Team Inspiration RobotX 2022 Technical Paper

D. Bahena Moctezuma, S. Boerhout, C. Chiang, J. Fausto, W. Gao, Amit Goel, Ashiria Goel, K. Goel, I. Gunawan, D. He, J. Hsieh, K. Jacob, K. Kim, M. Li, Z. Nebieridze, A. Samavedam, J. Silberman, P. Srivastava, A. Szeto, C. Szeto, M. Szeto, M. Tang, N. Tang, T. To, R. Veerepalli, A. Velasco, N. Velasco, E. Vij, L. Wright, J. Yu, and C. Yuan

Abstract— **Team Inspiration is focused on perpetually learning and improving, leveraging knowledge gained from FIRST robotics, RoboSub, collaboration with Triton-AI on autonomous cars, and industry internship experience to apply to Robotx Unmanned Surface Vessel (USV) and Unmanned Air Vehicle (UAV) development. With systems thinking, Team Inspiration applies engineering processes to understand RoboNation and Office of Naval Research (ONR) objectives, mission tasking requirements, and apply them to design, trade study, development, integration and test. Using modular development, incrementation prototyping and Virtual RobotX (VRX), the young team is able to demonstrate AUV waypoint navigation, perception with the support of UCSD capstone students. Team Inspiration learned the importance of teamwork, ownership, and time management.**

I. Introduction

Team Inspiration is a first-year RobotX team based in San Diego, California, USA. Conscious of the short time frame to go from ground zero to full autonomy, the team worked closely with the companies that furnished parts, consulted with local companies that work with USVs and UAVs, reached out to industry professionals for mentoring, talked with existing RobotX teams and coaches, and integrated UCSD Capstone Students into the team. This document outlines the decisions, designs, and testing for developing the USV and UAV for the 2022 Maritime RobotX competition.

II. Unmanned Surface Vehicle & Autonomous Aerial Vehicle

A. Design Strategy

Our engineering design decisions are based on analysis of individual tasks and the operation environment at the Sydney International Regatta Centre, Penrith in New South Wales, Australia.

- *Situational Awareness & Reporting:* requires communication between the Autonomous Maritime System (AMS) and Technical Director (TD) Server by transmitting a heartbeat message as described in the communications protocol.
- *Entrance and Exit Gates:* requires the AMS to detect the beacon, thus requiring the use of hydrophones. The WAM-V then passes through the gate, passes around the black beacon, and passes through the same gate.
- *Follow the Path:* requires vision perception of the buoys in order to pass through a set of colored buoys while avoiding black buoys.
- *Wildlife Encounter – React and Report:* requires a UAV with a hyperspectral camera to scan and search for targets and reports finding to the boat.
- *Scan the Code:* requires a vision system to perceive the three-color light sequence in an outdoor environment, as well as communication to the TD Network.
- *Detect and Dock:* requires a vision system to detect the designated color and the WAM-V requires a vision and LiDAR system to dock
- *Find and Fling:* requires a vision system to identify the correct target as well as a system to launch the ball with the correct velocity to cover the required system.
- *UAV Replenishment:* requires a vision system and gripper to identify colored tin and pick it up from the floating helipad.
- *UAV Search and Report:* requires a vision system to complete a search pattern and report the location of the objects, as well as to land in the designated landing site.

1. Autonomy Challenge strategy

We prioritized tasking by determining the minimum requirements for competition and sorted the mission tasks by how achievable they were based on the team's skill set. We compiled a list of requirements which different subteams would work on to complete the required tasks. For software, the main priority was navigation, as this was crucial for every mission. This included:

- Station Keeping
- Waypoint Navigation
- Accurate Compass and GPS Readings
- Sensor Fusion

We also worked to make these functions modular to use for multiple missions.

For mechanical, the main priorities were a dependable propulsion system, stable sensor mounts, and payloads to complete the missions.

For electrical, the main priorities were a reliable kill switch, robust power system, and modular interfacing of the electrical components to adapt to testing on multiple platforms.

2. Strategy to Sensor Selected

Precision navigation is key for RobotX, where 1ft accuracy is preferred as per the docking mission analysis. GPS sensors with Real Time Kinematics (RTK) capabilities of centimeters accuracy were selected to enable measurement of position drift caused by wind and tidal current effects.

For LiDAR selection, the team defaults to the solid state Livox LiDAR used in our team RoboCar competition to minimize team learning curve and cost since we can share them between the two vehicles.

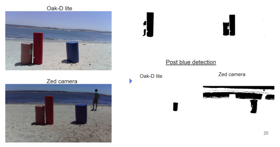

Fig 1. Color detection for Oak-D Lite vs Zed cameras. The Oak-D Lite camera is more specific in color detection than the Zed camera, along with being less susceptible to purple fringing.

The Oak–D camera from OpenCV AI Kit was selected over the 1080p Logitech webcam and the Zed camera to leverage the capability of the AI chip within the dual stereo camera. Capstone students discovered that bright lights cause chromatic aberration (aka purple fringing) in the Zed camera, eliminating it from our design considerations. The Oak–D can capture images at 120fps and at 60 fps, it can capture 12MP images while the webcam can at best capture 2MP at 60 fps. The Oak–D was the superior choice for our data intensive computer vision application.

The AS-1 Hydrophone was selected based on previous experience with RoboSub and existing material availability.

Communication systems were selected through a combination of searching on Ubiquiti's websites for radios, looking at the specs of max throughput, and verifying these specs against Ubiquiti forums confirming they work well. Team Bumblebee also gave feedback on the radios that worked well for them. The antennas were selected based on parameters (45° for ground to cover the area of competition (Omni for boat rotating 360°)). The switch and POE++ injector was based on budgeted items and mentor's preference

3. *Strategy to Propulsion Components*

The team selected four thruster configurations similar to mecanum wheels set up after watching the RobotX final videos over the years and seeing the advantage of lateral motions for fine tuning of the boat position. The holonomic drive configuration was proven in the prototype boat at the pool and lake. Torqeedo 2.0 motors were selected based on their compatibility with the WAM-V engine pod and their widespread use by WAM-V customers. The T-500 thruster was used based on the small form factor, efficiency and cost. The team has experience using T-200 thrusters on our prototype boat and RoboSub.

The team opts for a remote control propulsion system developed by WAM-V to provide assurance of safety operation while the team can concentrate on perception and control. The team converted the WAM-V remote control system baseline to autonomous and added the additional T-500 thrusters set.

a. *Detail for Propulsion Components*

i. Prototype Boat

The Mini-Me Blue Robotics T200 vector thruster configuration took heavy inspiration from previous experience in RoboSub and so did the electronics and control system. The final WAM-V configuration built off of that of the Mini-Me.

ii. WAM-V

Through research of previous competitors, the team established it was an essential requirement to implement front vectored thrusters to assist with position hold, in addition to the WAM-V provided Torqeedo Cruise 2.0 motors acting as primary thrusters in the back. A trade study of previous competitors' vector thrusters took weight, thrust, power required, and cost into account, but the most favorable motors in the trade study were discontinued. Additionally, Blue Robotics's new T500 motors could be controlled using the same servo driver system from the Mini-Me, making their integration simple for the programming team. As a result, the team chose to use T-500 motors at the front. There are 3D printed sacrificial mounts that shatter on great impact to preserve the motor.

A mount of 2" T-slotted profiles and bolts attaches two Blue Robotics T500 thrusters to the front of the pontoons to aid with steering.

The WAM-V propulsion control system is an adaptation of what was on the Mini-Me. The Me is driven by two main Torqeedo Cruise 2.0L outboard motors held in the rear as well as the four vectored T500 thrusters in the front. The outboards are controlled using serial commands sent from the AGX to the WAM-V ROBO-HELM unit which then sends it to the motors.

The T500s are configured with two facing the same direction on each side, essentially combining their power output. They point at a 45° angle inward vector to achieve holonomic drive, allowing for precision forward positioning and rough strafing. In order to drive the 24V T500s off the large 7000wH 30V Torqeedo battery bank on-board, we purchased two FLYCROSS HV ESCs rated for up to 50.4V compared to the Basic 500 ESCs only rated for up to 26V. We wired both motors on each side to one ESC to ensure they spin at the same speed. We then instituted a throttle limit on the ESCs so based on the duty cycle the ESCs would never send an average voltage above 24V to the motors.

Both the T500's and Torqeedo power systems are routed through the EMO, meaning both can be immediately disengaged.

Four Blue Robotics T500 thrusters placed at the front of the WAM-V were chosen to achieve holonomic drive. They are controlled in pairs using common 80A drone electronic speed controllers and are restricted to a maximum thrust of 60% to allow for drive at 6V above their nominal rating. Thus, additional voltage bucking isn't required and the thrusters can be run on the main power line on board the WAM-V.

4. *Strategy to Software Development*

Modularity, testing, and data collection were key strategies deployed to tackle the complex autonomous system. From low level motor controls to high level perception models, our modular sub systems allowed programmers to divide and conquer while iteratively reaching our goals. Functional and integration tests reduced debugging time for our LiDAR perception data collection, motor controls, mission planning, and wireless communication systems. As a data-driven team, we collected data on each sub system's performance by monitoring the memory, CPU usage, and network traffic, which was used to further improve performance and integrate with the rest of the system. These key strategies combined allowed rapid and effective deployment and further testing.

Initial tests on software were run using ArduPilot on a Pixahawk from RoboSub. This aided the team in collecting perception data while new code was written and new electronics were secured to convert the setup to the final version used on the WAM-V.

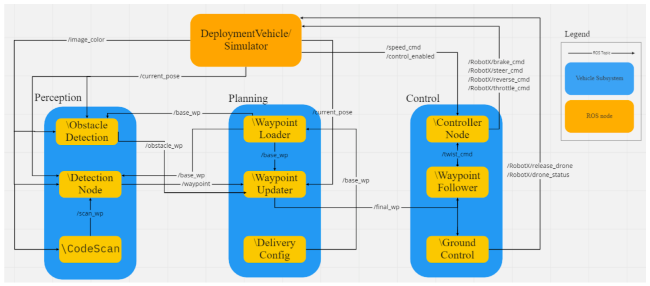

Fig 2. High Level Software Architecture Diagram

As per the software architecture diagram, ROS nodes hold minimal yet specific functions to divide the code into testable parts. Sensors such as the GPS, cameras, and LiDARs each had their own nodes devoted to data collection. A Mapper node fused the information from these nodes and created a grid-like map of the WAM-V's surroundings. Based on this knowledge of mission targets and obstacles, the Mission Planner node creates a safe path for the WAM-V to traverse, and passes waypoints to the waypoint navigator node. The waypoint navigator node, one of the key functionalities tested in VRX, calculates motor commands to move the WAM-V to the specified waypoint.

5. *Approach to Innovation*

As a rookie RobotX team, we were aware of our lack of experience with the dynamics of a water based USV, especially something as large as the WAM-V. This, in combination with the need for water time to test electronics and software while the mechanical team set up the WAM-V, drove the decision to create a working prototype boat that allowed for some development with less rigorous logistical requirements. This is similar to Team Inspiration's progression in RoboSub, where a smaller Blue Robotics submarine was used to test navigation and perception as the final customized submarine was being developed.

The aptly-named "Mini-Me" test bed allowed early testing in the backyard pool, easy deployment at the local lake. Team Inspiration plans to use it after the WAM-V shipping to maximize water test time. The electronic box is designed such that it fits on the mini-me and me. The wood shown on the boat is scrap material and the big electronic box is free. While the box is bigger than needed, it does help out with the thermal management aspect.

The final WAM-V design had two Torqeedo Cruise 2.0 thrusters as the main drivers in the back, with two pairs of Blue Robotics T500 motors in the front to aid with strafing. However, after disassembling and reassembling our WAM-V setup to move it onto a new trailer, the Torqeedo motors stopped working. While a Torqeedo repair shop troubleshooted our motors' electronics, we designed a backup solution for our thrusters with two T500s on each side at the back instead.

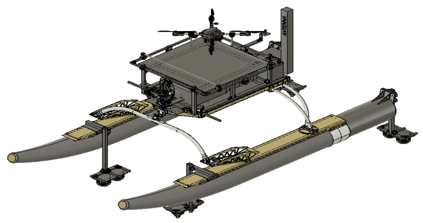

Fig 3. CAD view of backup propulsion system. This is identical to our final design, save for the Torqeedo motors

B. *Vehicle Design*

1. *Mechanical*

a. *Prototype USV - MiniMe*

For RobotX, the aptly-named "Mini-Me" was an Achilles LEX-96 inflatable with four vectored T200 thrusters mounted using an 80/20 T-Slotted profile frame. The Mini-Me included places to attach sensor and radio masts and retractable arms for Blue Robotics T200 thrusters in holonomic drive configuration.

b. *USV - Me*

A 1" square T-slotted profile serves as a perception rail for Oak-D Lite cameras and LiDARs. This also doubled as a GPS mount. The T-slotted profile is removable and can be transferred between the Mini-Me and Me setups.

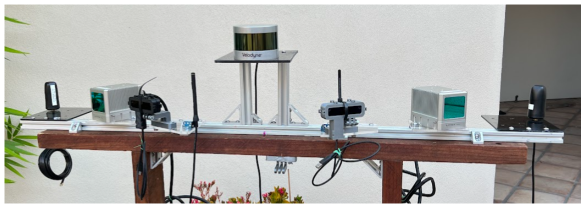

Fig 4. The perception rail enables mounting of perception sensors in one convenient mobile system for shared use between Team Inspiration and UCSD Capstone students.

Initially, a smaller WAM-V-provided radio was mounted directly to the "legs" of the WAM-V for convenience, but later a custom communications mount was riveted directly behind the WAM-V platform when the team upgraded to a larger radio.

The team chose the final design of the drone platform based on a trade study of various designs factoring in weight, simplicity, and reliability. As opposed to heavier designs which would passively catch the drone in turbulent waters, the team pursued development of a 4ft by 4ft drone landing platform with fiducial tags to assist with autonomous drone landing. This design was inspired by Planck Aerosystems, a small company local to California that is developing autonomous technology for drones.

The catcher mechanism consists of four 60:1 neverest motors. The motors were selected due to the team's proficiency with the model, as well as their availability at the lab. These motors were used to actuate eight carts which closed four PVC pipes, directing the drone to the center of the platform. The electrical interface consists of an Arduino controlling four BTS7960 motor drivers. Readings from the encoders of the motors enable the Drone landing platform to read current spikes to determine whether the carts have opened to the fullest extent or when the catcher has secured the drone legs.

The initial prototype of the racquetball launcher used 5205 Series Yellow Jacket Planetary Gear Motors, wheels, drive belts along with acrylic parts to build a turret launcher that can rotate 180 degrees from left to right, and linear rails to allow for easy angle adjustments. The launcher also holds a LiDAR and Oak-D Lite camera to aid with depth and color perception for aiming at targets, with its main controller being an Arduino Mega to control the motors using BTS7960 motor drivers for motor power. Limit switches are placed on two sides of the launcher to limit its movement within 180 degrees on each side.

c. *UAV (Drone)*

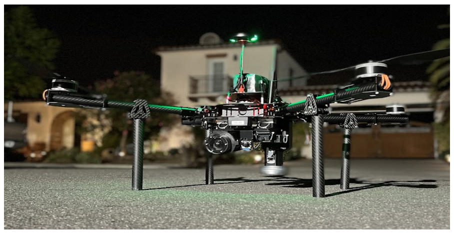

Fig 5. Full system view of assembled UAV. Note the camera positioned at the front of the perception rail.

The UAV took heavy inspiration from one of the companies which mentored Team Inspiration early on: Planck Aerosystems. By referencing a single photo of their medium-sized UAV, we custom designed and cut a three layer carbon fiber frame (ref main picture already on techpaper). Capable of 20kg of thrust and a 20 minute flight time with a 7lb payload, 30 minutes without, the drone is more than capable of fulfilling our competition requirements.

The UAV has a smart return-to-home feature and due to its large size and weight, an extremely stable position hold. The drone is also entirely water-resistant due to its layered design and is able to sustain crashes without significant physical damage due to its 2mm thick carbon fiber plates.

The drone can also fold itself into a compact 16”x16”x23” rectangle by removing the landing legs with a single screw and folding the arms down with a spring release mechanism. Fully expanded, it has a one meter stator-to-stator diameter and runs with 18” carbon fiber propellers.

The payload rail was 3D printed in a combination of ABS and PLA. This material was chosen to break during a crash, preserving the functionality of the sensors and gimbal motors.

There is one bottom facing camera, a gimbaled forward facing camera, one HSI camera, one bottom facing LiDAR, and the built in flight control found on the Pixhawk. This provides the drone with adequate information for autonomous takeoff and landing, aluminum can manipulation, and information about the wildlife mission.

The team determined it was advantageous to mount the HSI camera onto the drone rather than creating a mechanical arm to provide a top-down view of the wildlife mission models, providing more accurate location data quicker.

2. *Electrical*

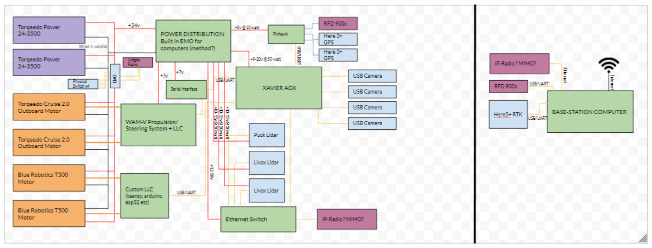

Fig 6. Electrical schematic of our WAM-V system

Preliminary electronics and control system development built off of Team Inspiration’s RoboSub designs.

a. *Prototype Boat*

With an initial goal of basic RC control, we developed our barebones electronics system within a 20” x 14” Pelican case. By drilling holes through the sides, we routed our motor wires into the box as well as let an antenna out for our radio receiver. Each T200 motor was connected to a Blue Robotics Basic ESC which then connected to our flight controller, a PixHawk Cube Orange. The Pixhawk contains several internal redundant sensors including both compasses and IMUs essential for navigation. We also used two Here3 GPSes with an RTK system to provide centimeter level positioning data to the Pixhawk. Our comms were established through an RFD900x Long Range telemetry system which also interfaced directly through the Pixhawk. Everything was powered with two Blue Robotics 16Ah batteries in parallel allowing for roughly four hours of run time during testing.

b. WAM-V

Now more familiar with the physical characteristics of the boat, Team Inspiration moved onto data collection and autonomous programming. The electronics were moved to a much larger box, and the Pixhawk in the motor control loop was replaced by an independently programmable PWM driver. The PWM connected directly to the ESCs and could be controlled by the Jetson AGX on-board.

The Electrical subsystem is powered by two 12V 50Ah Lithium LiFePO4 Deep Cycle Battery isolated from the main Torqeedo Power 24-3500. This provides the team with the assurance the performance of the electrical system will not dip during aggressive maneuvers.

The USV is powered through three main battery systems. The primary battery system is the two 3.5kWh Torqeedo batteries wired in parallel for a combined 7kWh of power. These batteries are used only to drive the two Torqeedo Cruise 2.0L and ROBO-HELM control held on the aft, and the four Blue Robotics T500 motors in the front, for a combined draw of 8kW during full throttle (very rare). The system can run for 32 hours before the Torqeedo batteries need charging, assuming normal usage. The secondary battery system involves two 600Wh LiFePO4 batteries held in parallel for a combined wattage of 1200Wh. This battery system is used to power all the control electronics and sensors on the USV besides the ROBO-HELM. Compared to the fluctuating power of the Torqeedos with its excessive current draw, the LiFePO4 batteries provide a much cleaner power to the systems in the electronics box. The third battery system is a single 18AH 12V motorcycle battery that powers the normally-open solenoid emergency stop. The switch is rated for 700A at 250V DC and designed for electric vehicles with much higher current draws, decreasing likelihood of e-stop failure. The switch requires a continuous 12V supply to be broken through one of the four push-buttons we have on the pylons of the boat.

The majority of the electrical infrastructure of the boat is housed within the large military grade air shipping container we have repurposed as our electronics box. Taking advantage of the box's removable top, we attached a 22 watt sealed active heat exchanger to the lid to ensure the thermal protection of the internal electronics. As power first enters the box, it passes through a 12V 20A regulator to a power distribution board. This board then supplies power to the Jetson AGX that is the main computer, the Livox Horizon LiDARs, the comms system, and the cooling system mentioned earlier. There is also a 12V rail which goes to a 5V 10A DC-DC converter to power the touchscreen display and powered USB hub for the AGX. The USB hub connects to two Oak-D Lite stereoscopic cameras, the servo driver for the T500, and our Ardusimple GPS RTK system for millimeter-level precision as well as GPS heading. Our T500 ESCs are also housed within the box although entirely isolated from the rest of the electronics besides a PWM wire going to the servo driver. To prevent electrical noise interference with the GPS system and antennas that are housed outside the box, we have covered the wall facing the GPS with grounded copper tape.

Within the box we have an ethernet switch which handles all of the boats comms, internally and to the ground station. The switch connects the AGX, LiDARs, ROBO-HELEM unit, and a POE injector for the groundstation onto a single local network.

c. Kill switch

The kill switch utilizes Interstate SLA1116 12V SLA Battery (Andymark) and red kill switches in series to detect when a button was actuated and kills the propulsion system. This circuit is only hooked up to the Power 24-3500 meaning the computer systems which are isolated still run even after the kill.

d. Racquetball Launcher

Initial prototypes used an Arduino and position control system to control the speed of the motors driving the loader belt and the flywheel using encoders. The motors are powered by a separate 12V battery connected to BTS7960 motor drivers and the LiDAR and limit switches are powered by the Arduino directly with a 5V voltage supply. Both the Arduino and the OAK-D Lite are connected to the AGX Xavier for power.

e. Drone

Everything is conformal coated. Great care was taken to prevent the wiring from making contact with the conductive carbon nanofiber drone body.

The lowest layer contains a majority of our electrical systems, featuring a 4-in-1 ESC, RC receiver, voltage regulators, cooling fans, a Raspberry Pi 4B 8GB (companion computer), and an arduino nano (LED system). All the wires are conveniently contained within the layer enclosure or routed through the carbon fiber arms to the motors and LEDs.

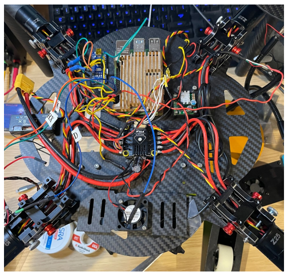
Fig 7. Top view of electronics plate

The second layer contains our flight controller, a Pixhawk Cube Orange, which connects to our ESCs, RC receiver, long range telemetry module, and companion computer among other sensors like a precision GPS, external barometer, and a downwards lidar. The Pixhawk takes care of the low-level stabilization and position hold of the drone through the sensors, allowing us to focus on the autonomy required to complete the missions. The second layer also contains a mosfet rated for 280A at 50V DC allowing us to use a power switch to immediately power off the drone.

The uppermost layer contains our GPS, Barometer, and our 13aH battery. It also has a safety switch which prevents the drone from arming unless it has been triggered.

f. Drone Landing Platform

To control the four rails that make up the catcher, four motors are installed on each of the four sides of the drone landing platform. The Arduino is used to control the motors using four BTS7960 motor drivers.

3. Software

a. Architecture

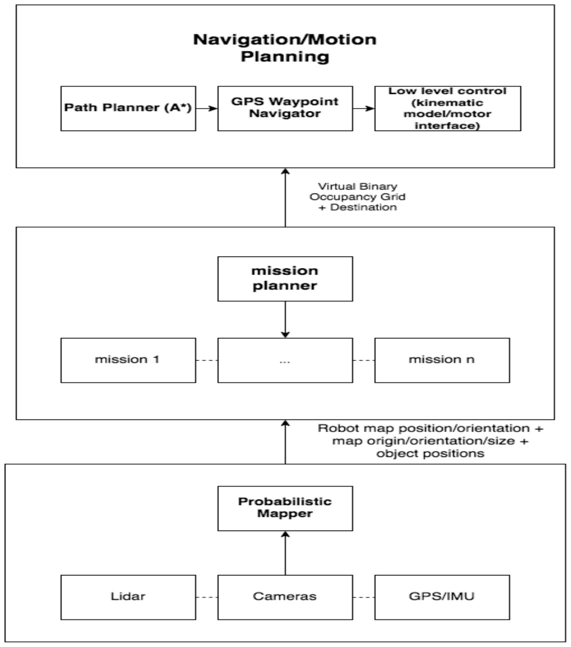
Navigation/Motion Planning
Path Planner (A*)
GPS Waypoint Navigator
Low level control (kinematic model/motor interface)
Virtual Binary Occupancy Grid + Destination
mission planner
mission 1
...
mission n
Robot map position/orientation + map origin/orientation/size + object positions
Probabilistic Mapper
Lidar
Cameras
GPS/IMU

Fig 8. The software architecture diagram divides software into the mapper that aids in situational awareness, the mission planner that decides the WAM-V's course of action, and the navigator that commands the thrusters

The robot uses sensor data from the LiDARs, cameras, and GPS/IMU in order to construct a map of objects in its surroundings and their associated probabilities. Initially, only the map was probabilistic. SLAM algorithms ID and list objects. After generating the list of objects, the mapper sends it over ROS to the mission planning and execution module; details can be found under Section D. Motion planning is handled by the path planner, waypoint navigator, and low-level control program; the path planner uses A* and by default converts the map into a binary occupancy grid; however, certain missions can choose to input a modified occupancy grid to specify additional movement constraints.

b. VRX Simulation

The VRX competition presented an opportunity to develop preliminary software. Participating in VRX allowed us to learn about the structure of the WAM-V code before the physical boat was ready for us to test on. We used VRX to test waypoint and station keeping functionality which serves as a baseline for the physical boat navigation.

Team Inspiration's final software architecture used the same structure of topic names as VRX. We also mimicked the approach of sending thruster commands to a topic. This similar structure allowed us to easily test the same software on the WAM-V and simulator with minimal effort.

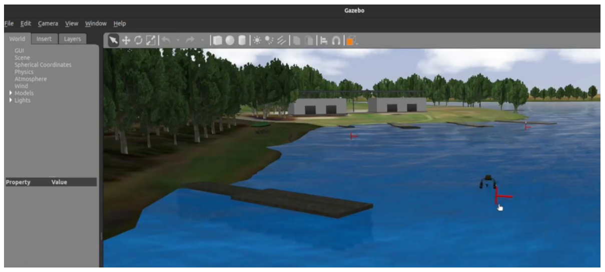

Fig 9. Waypoint Navigation in VRX. The structure of the software used in VRX translated to real life RobotX development.

c. Mission Planning and Mission Execution

The mission planner is responsible for maintaining and updating a list of missions that are to be executed, as well as information concerning how each mission is to be executed. Every mission follows the same general structure so the specific missions inherit from a more general generic mission. Associated with each mission is an arbitrary difficulty ranking, the general area in which a mission is located, and the mission status (the number of times a mission has been attempted as well as its success/failure). Also associated with each mission is the type of information that is required to complete it (such as various types of objects, or knowing where the entry point is), as well as rules to determine whether this information can be extrapolated, and the difficulty of extrapolation.

The mission *execution* algorithm for each mission assumes that all the necessary information has been provided, and operates on an ideal scenario. As a result, the mission planner itself must perform the necessary extrapolation, and weigh the risks accordingly. The mission execution algorithms will provide waypoints on the grid-map towards which the robot should navigate. Control is handed over to the path planning algorithm detailed in the *Path Planner* section, which determines the shortest path while avoiding obstacles.

Certain missions require more specific movement requirements than simply finding the shortest path without obstacles. Missions are permitted to provide a "virtual" binary occupancy map. The path mission is a salient example:

The path planning between the buoys cannot be just calculated from the occupancy map, as the spaces between the buoys are free. To solve this, the order of the buoys needs to be calculated, and virtual occupied space needs to be added. Given the start and end point of the buoy path, first the buoys at the ends are found by using the white buoys as reference. Then a traveling salesperson problem can be calculated to determine the correct crossing order for the rest of the buoys. When the correct order is found, then borders are applied connecting the buoys, which allows the A* algorithm to run as expected.

d. *Mapper*

Information about the map's origin (in GPS coordinates), orientation, density (the size of each map cell), and size (# cells * # cells) is created upon initialization and then published globally. This way, the rest of the software system can operate on objects represented in the internal map's reference frame, while the conversion from map coordinates to GPS coordinates can be relegated to low level motion control.

The first prototype of the mapper contains a probabilistic grid-map that holds information about the different types of objects in the robot's surroundings. To do this, the mapper kept track of the number of times that each "cell" of the map had been observed, using the field-of-vision, orientation, and maximum detection distance. It simultaneously counts the number of times that each class of object was sighted in each map cell (rudimentary position estimation of objects was performed using the position of the object in the camera frame, and the distance reported by the LiDAR). Using this information, a probability is assigned to each class of object in each cell. A threshold is then applied to eliminate unlikely candidates. To make sure that a single buoy is not perceived as multiple buoys, an erosion and dilation can be applied to the grid-map. The object type with the greatest occurrence within a cluster is chosen as the candidate for the object.

Our current mapper uses SLAM. The objects have their own ID and coordinate location that are stored in a data structure. Data from the drone and the WAM-V perception mount are inputs to the mapper.

Simultaneous localization and mapping problem formulation: Given a robot drove around and measured the following sensor data: LiDAR_data ∈ R^3×n series of points in 3D, gps_data of latitude, longitude, and heading.
Technical Approach: From the GPS reported state of the robot, we derive a transformation matrix ∈ R4×4 to project LiDAR measurements from LiDAR frame to body frame to world frame and finally to map frame.

Odds are updated in the map using the Bresenham algorithm for labeling empty space vs occupied space given LiDAR hit points. For example the light up area in the following picture will reduce the odds of occupancy in the final map:

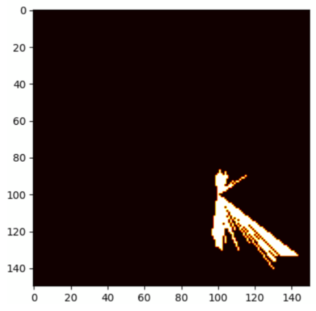
Fig 10. LiDAR based occupancy map

Waypoint navigation in standalone modeling and analysis to be integrated into the overall systems. The model is optimized based on the hypothesis of red/green buoy pairs with black buoy being the obstacle to be avoided. For the system to work, integrated perception data collected from the drone and the boat is an integral part of the solution.

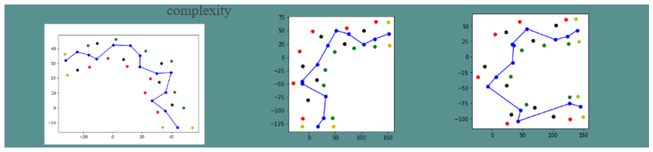

Fig 11. Waypoints with path planning; the drone can differentiate objects by color and type and collect coordinates of objects to plot on the WAM-V mapper.

e. *Perception*

The perception systems use a combination of OpenCV and YOLOv5 Object Detection to differentiate between various objects on the course. OpenCV was used for objects with easily discernible features (such as a distinct color) in order to be less computationally expensive. YOLOv5 was used for more complex objects that did not fit the above criteria, and was more computationally expensive and time consuming.

The perception algorithm combines both LiDAR and camera to output depth estimation and object recognition with calibration, projection to point cloud, ROS publication, image processing, bounding box and class detection, and distance measurements.

f. *Path Planner*

The path planner uses an occupancy map and information about the objects in that grid to determine the shortest path between desired waypoints. It takes in the information brought in from the SLAM and outputs the next location where the boat should head toward.

The chosen algorithm was A* for several purposes. It is guaranteed to return the shortest path, its heuristics allow faster calculations, and it can easily be changed to do quick recalculations of the map during executions. The A* algorithm finds the best path given two points. To do this it uses a Markov Decision Process where each free space in the grid is a member of the state space.

g. *Drone*

The lowest level of UAV autonomy is handled by the on-board pixhawk cube orange running PX4 commanded by a Raspberry Pi 4 companion computer. Over MAVROS, the Raspberry Pi is able to communicate to the pixhawk over mavlink communication in order to set local and global position setpoints, retrieve sensor data fused on the pixhawk (drone attitude) and absolute position in the global frame over GPS.

The high-level mission is commanded by the Raspberry Pi running ros noetic, allowing us to easily expand our system in an efficient and sustainable manner. It runs perception algorithms, communicating with two on-board cameras (downward and forward facing cameras) and handles communication with the WAM-V to ensure synchronization of the mission.

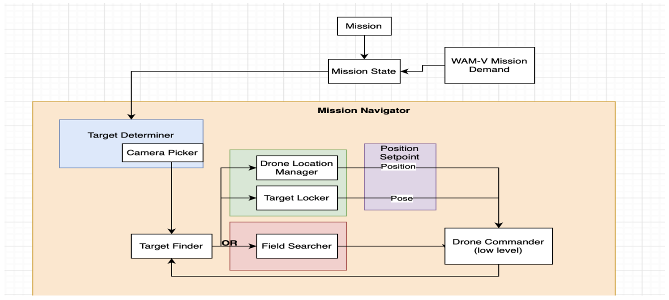

Fig 12. The aim of the mission architecture is to create a general, expandable algorithm that can be used for every mission. Since they are all in some way based on targeting buoy setpoints, there is a stress on determining the targets to fly towards based on the current mission, given to the drone by the WAM-V

The UCSD Capstone students developed a Python script running off a Raspberry Pi 4 as a baseline for vision informed autonomous landing.

In order to perform autonomous landing, we debated between multiple fiducial markers and configurations. Most notably, ArUco, ARTags, and AprilTags. Through research, we determined that AprilTags would be ideal for long ranges, computationally efficient, and eliminated rotational ambiguity. Within the AprilTag library, we experimented with each family and determined that the 36h11 family would be most effective for long distances. Initially, we planned to have four tags with a TagCircle49h12 on the inside for re-alignment (in terms of heading), however, we determined that it would be more effective to have four larger tags on the outside corners and to maintain drone heading through mavlink commands sent to the pixhawk, which has in-built GPS-based heading control. The detection algorithm provides the center coordinate of the AprilTags, and the drone utilizes a feedback loop to reach a position close to the coordinate before landing.

Initially, YOLOv3 models were trained to identify objects, but eventually the Capstone students upgraded to YOLOv5 to detect aluminum cans on the background of landing platforms and buoy recognition.

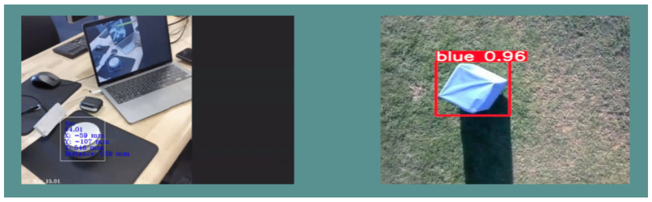

Fig 13. Recording coordinates of the buoy within the FOV of the drone camera for USV navigation.

h. Drone Landing Platform

In order to determine if the drone is secured by the catcher, the team debated between designing the drone landing platform to read voltage spikes from the motor encoders or to enable the catcher to detect the drone legs by fabricating the drone legs and the catcher from conductive material. Also, to determine if the carts have opened to their fullest extent, installation of limit switches was proposed. With encoders already available, we decided to use them. We knew that reading encoder spikes would cause the motors to stall for a few seconds after securing the drone legs before turning off; this introduces stress to the gears, but it was able to produce desirable results.

i. Racquetball Launcher

The C++ script running on the Arduino uses the flywheel motors' encoders to determine its RPM. When the flywheel motors rotate, the script receives encoder signals to derive the speed of the two motors' rotation in RPM based on the motors' encoder to gear ratio. A function that determines the required RPM based on the distance from the target provided by the LiDAR is derived beforehand. To keep the flywheel spinning at a constant desired RPM during battery power consumption, continuous calculation of the motors' RPM provides feedback for the Arduino to adjust the power provided to the motors by the BTS7960.

The Capstone students initially decided to use MATLAB Simulink to find PID constants for the constant flywheel speed control function in the script for a given flywheel RPM. Since different distances from the target require different RPM values, multiple sets of PID constants must be found. This is because each set of PID constants is good at a small RPM range, but their effectiveness decreases when a different RPM is required. Due to time restrictions, the capstone team relied on proportional control instead, adjusting power provided to the flywheel motors according to differences in the desired RPM and the flywheel RPM. A tachometer is used to confirm the reliability of the constant flywheel speed control function based on proportional control.

The Capstone students developed a Python script running on the AGX Xavier to analyze the visuals provided by the launcher's OAK-D Lite to determine the position of the target relative to where the launcher is facing. Initially, OpenCV was used to detect contours in the image, then identify which contour was correct. However, this method was susceptible to noise and often incorrectly detected the target.

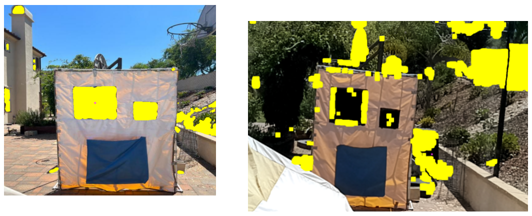
Fig 14. OpenCV target detection and detection failure due to noise

Therefore, OpenCV for target detection was replaced by a trained YOLOv5 model of the test target. Once the target is detected, the Python script sends commands serially to the Arduino to control the turret to align the launcher to face directly at the target and another command to the Arduino to turn on the flywheel and the loader.

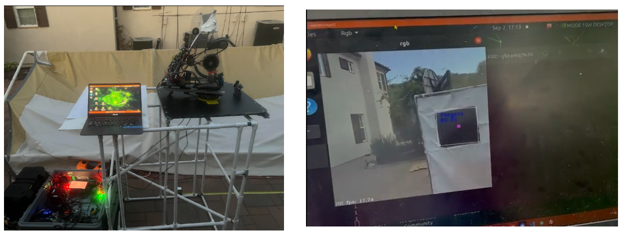
Fig 15. Running prototype launcher with the AGX Xavier using YOLOv5 target detection

4. *Design Process and Methodology*

Building off of what made our team successful in RoboSub, we used a mix of systems engineering and agile processes. For our project management we created a dynamic task list that was populated backwards from the final competition date. With our ever changing number of team members, this task list was extremely important for assigning new team members tasks and bringing them up to speed to increase efficiency and tracking.

We knew that we needed more technical depth than the team had when we were awarded the boat. To get that technical depth, we picked out tasks that our team was struggling with and proposed those tasks to the capstone students as an option for their final project. We conducted weekly meetings with the students and their advisor, Jack Silberman, to track their progress alongside the core team.

While the team worked in parallel with the capstone students, individual sub teams also worked in parallel. While the mechanical parts were being chosen and procured for the WAM-V, the electrical and software components were prototyped and tested on the Mini-Me and the first drone frame was being outfitted for testing.

5. *Lessons Learned*

Team Inspiration quickly learned how important logistics are in larger-scale competitions. Purchasing and machining RobotX parts took much longer compared to RoboSub and FTC, emphasizing the value of good prototyping and planning. With proper planning and review, parts could be ordered and arrived to integrate quickly into the existing systems. Prototyping with various materials became very important to make sure that the components would be waterproofed and non-conductive. Additionally, to make more efficient use of time and resources, team members learned to review designs with each other to determine whether their CAD models were over-designed or their software ideas were overly complex. The Team Inspiration lab currently has at least 50 ft of extra T-slotted profiles due to part procurement for an over-designed prototype.

Additionally, Team Inspiration learned that it is important to regularly meet with mentors and professional industry members, and to contact previous RobotX teams, to get feedback and necessary help.

Here are some other key points for young teams:

- Start using simulations for testing early. Working on Gazebo for VRX and beyond contributed significantly to the software development process.
- When stuck, think outside the box to find a solution
- Be prepared to spend a lot of hours on water testing. Water testing has a way to multiply to-dos on land, including searching for replacement tools when they get dropped in the lake or bay.
- Test out bottleneck issues that get in the way at each stage of the development process
- Merging different sets and LiDARs might be a valid alternative for sensor fusion with good results
- The system must be tuned according to the use case
- Design specifications are important in determining the tools to use
- Build off existing tools to solve problems
- Model and distance calculation comes with noise. Post-processing ensures accurate and reliable reading
- Data collection can be time consuming and error prone, so start early

III. Experimental Results

A. *HSI Camera testing*

The team tested the camera with a gray background at a representative of the UAV altitude at 10 meters. Capstone students wrote scripts that do the following:

- Update the calibration files to account for variations in sunlight
- Collect/process data cubes into raw digital numbers or reflectance
- Compute an average signature over a defined region
- Take a file containing signatures and generate a segmentation mask that tells you the most likely signature at each pixel of the scan
- Use the segmentation mask, start/end coordinates, altitude and camera angle to get the GPS coordinates of where each signature is likely to to map the area

B. *Drone Landing Platform Test*

Thirty six continuous opening-and-closings of the catcher without the drone was successful. However, when the drone is placed on the platform, unless it is directly in the center of the platform, the catcher rails are unable to push it towards the center due to its heavy weight. Because the rails were not designed to move independently, if one rail stops moving when it touches the drone's leg, the opposite rail will stop as well, preventing the other leg of the drone from being secured. Testing proved that we can either implement independent movement of each rail, replace the motors with motors of higher torque, or check if increasing the tension in the strings will produce better results.

C. *Scan the Code Test*

To test for scanning the code, the team first used a 32x16 Adafruit RGB LED matrix to build a LED panel for land and water testing. The panel is mounted on a floating device via pvc pipes for water testing. The Arduino Uno is used to control the LED panel using code provided by Adafruit. Images of the panel displaying red, green, blue, and off are uploaded to Roboflow for training using YOLOv7 and a trained model is used in a Python script to allow the OAK-D Lite to recognize LED color patterns. More than 3000 images were annotated for training and the results were successful. Remaining tests are to be done for the 64x32 RGB matrix utilizing the same strategy.

D. *Prototype Racquetball Launcher Test*

To test the launcher, a target with dimensions of 0.5m x 1.5m x 2m was built using pvc pipes with a small 0.25m^2 and a large 0.5m^2 target separated 0.25m apart, located 1.5m above the base. The large target is used for initial testing.

To confirm no detrimental effects by the wind on the ball after launch, wind testing was performed by utilizing an industrial air mover in the ball's path at speeds up to 82km/h.

Initially, a flywheel speed function based on the distance from the target was calculated based on physics. However, due to difficulty in determining the energy loss from friction between the ball and the flywheel, and the spinning of balls, a flywheel speed function based on direct measurement of the distance from the target and the RPM required is used. Using the LiDAR to determine the distance from the target, flywheel RPM values are tested until one hits the center of the target. Using data from a range of three to seven meters, a flywheel speed function is created using linear regression.

Testing is done to confirm that the OAK-D Lite can detect the large target and that the launcher will turn and face directly at the target continuously. The final test results for the prototype launcher's accuracy on the large target for three, five, and six meters by the capstone students are (29/33 - 87%), (27/33 - 81%), and (33/33 -100%), respectively.

IV. Acknowledgements

Team Inspiration would like to thank the RobotX organizers for giving us the opportunity to compete with high school students as team members and putting the competition together to challenge us. We also extend our gratitude to all of our mentors and sponsors that made our entry into RobotX possible. Finally, we are indebted to our parents for their continuous support.

A. *Mentors*

Andrew Raharjo, Anthony Tang, Arti Baldwin, Blake Baldwin, Brian Liu, Cris O'Bryon, Debra Kimberling, Handoko Gunawan, Jason Jerrell, Justin Dailey, Kevin Bowen, Kenzo Tomitaka, Kunal Srivastava, Leverett Bezanson, Michael Arnstein, Perla Myers, Phil Yao, Ralph Morales, Teresa To, Venkat Rangan, and Mike Jilka

B. *Sponsors*

Diamond: RoboNation, Biosero, and Northrop Grumman.

Platinum: Qualcomm, Intuit, Hologic, San Diego Foundation

Gold: Medtronic, Leverett Beznson, tinyVision.ai

Silver: WaterLinked, Nvidia, Team Kenautics, Blue Trail Engineering, Velodyne, ModalAI

Bronze: ePlastics, Sempra Energy, SolidWorks, MathWorks, Leidos, Jet Brains, Manna's Martial Arts, Roboctopi, Forecast 3D, Metal Master Inc., Harbor Freight, Costco

V. References

[1] RoboNation (2022, Sept.). Team Handbook. Robonation, United States of America. [Online]. Available: https://robonation.org/app/uploads/sites/2/2022/03/2022-RobotX_Team-Handbook_v2.pdf

[2] K. Poon, R. Fu, B. Zhao, P. Kliman (2022, Jun.). Ruggedization and Integration of Electronics for a Prototype Autonomous Boat. UC San Diego. CA. [Online]. Available: https://drive.google.com/file/d/1Xlo86G5RFGCU_Jgacj1yu62R6y18SlJq/view.

[3] K. Poon (2022, Jun.). RobotX Competition: Camera Testing - Oak-D Lite vs Zed Camera. UC San Diego. CA. [Online]. Available: https://docs.google.com/presentation/d/1ku2XyZWBBR2BSyKC04eDiT2mFoPXA7kRRoRS9GXjvVs/edit#slide=id.g13089135974_0_189.

[4] Z. Nebieridze, J. Yu, J. Hsieh (2022, Sept.). Autonomous Racquetball Launcher. UC San Diego. CA. [Online]. Available: https://docs.google.com/presentation/d/18TNruEHCynb7yBwBQi5aqd46XjwJvOnI2LYHdbtbLzg/edit#slide=id.p.

[5] C. Chiang, C. Yuan (2022, Sept.). Team B Perception: Final Presentation. UC San Diego. CA. [Online]. Available: https://docs.google.com/presentation/d/16TcoSU2I6hugb4phOIweBv4ngVn5DpiImXmWtyBlRjM/edit#slide=id.g15387ada36b_0_126.

[6] M. Tang, M. Li, D. Bahena Moctezuma (2022, Sept.). Drone Autonomous TakeOff Landing and Mapping. UC San Diego. CA. [Online]. Available: https://docs.google.com/presentation/d/1eYI6tUwh_iZZaEr4rzxoda6fp9du1X3dp2QftuKTvTM/edit#slide=id.g149f1c33aaf_0_554.

VI. Appendix a - STEM OUTREACHES

FIRST Global Coaching - Benin, Togo, Paraguay, & Ecuador robotics teams

Continued partnering with San Diego Fleet Science Center to teach and showcase robotics.

Showcased our WAM-V and drone at the Navy Gold Coast's 2022 event.

Sharing the benefits of and experience using computer science in majors that are not computer science with Del Norte High School computer science classes.

Hosted a FIRST Tech Challenges (FTC) scrimmage and mentoring numerous FTC/FIRST Lego League (FLL) teams to support our local FIRST community.

Continued to partner with University of San Diego STEAM Academy to teach week-long STEAM sessions, and provide students with a hands-on underwater robotics experience.

Hosted a summer camp at Manna's Martial Arts to teach students about FIRST, EV3 mechanics, and programming.

Co-hosted a FTC league meet at Poway High School.

Introduced land and underwater robotics to elementary and middle school students.