UM::Autonomy's *C-3PO* and *R2-D2*

UM::Autonomy 2021 RoboNation RoboBoat Competition Technical Design Report

Ryan Draves, Ryan Do, Jimmy Chen, Gregory Su, Albert Wei, Emma Shedden, Tom Gao, Michael Biek, Andreya Ware, Minh-Quan Nguyen, Rimaz Khan, Yi Lin Sim, Dora Guo, Eric Ma, Will Snedegar, Lance Bassett, Luke Lesh, Arjun Anand, Allen Ho, Abdeali Poonawala

UM::Autonomy, University of Michigan College of Engineering, Ann Arbor, MI, USA

Submitted May 23rd, 2021

Abstract

This paper serves to document the development and strategy behind UM::Autonomy's boat, *C-3PO*, and its drone, *R2-D2* for the 2021 Roboboat competition. To accomplish this complex task, the team worked in 5 separate subteams: AI, Hulls and Systems, Electrical, Drone, and Business. Each subteam operated remotely, only working in person when necessary. This meant keeping and making small improvements to areas of strength such as the sensor suite, electrical box system, carbon fiber fiber infusion process, and much of our software while also experimenting with more innovative approaches such as the new hull form, hydrophone system, and the addition of a CV Deep learning approach. As the year progressed we strived to keep our team and provide as many resources as possible to our members across the school year. These changes allowed the team to better tackle the RoboBoat competition challenges and ultimately push the organization to greater heights.

Team photo from our outdoor testing in April.

Introduction

UM::Autonomy is the University of Michigan's premier autonomous boat and drone team at the University of Michigan. The 2021 Competition marks the 15th year the organization has participated in RoboNation's Roboboat competition. With the continuation of the current roboboat format and the uncertainty of the global pandemic, the team wanted to revise the previous iteration of the boat/drone which was unfortunately never completed due to Covid-19. Because of the virtual format, we decided to keep and improve upon many aspects of the project while also subtly experimenting across the subteams. This approach to the design of the *C-3PO* and *R2-D2* allowed the team to adjust to the new virtual format while also providing enough time and resources for our members to balance the workload from their remote places of work. This paper elaborates on the changes our team made, how we approached each problem, and the results of our hard work in these unprecedented times.

Competition Strategy

Navigation Channel

The navigation channel might not require any fancy algorithms, but it still requires all parts of the boat to work together to perceive, localize, and move. Our approach would be to perceive the initial buoy gate and chart a course perpendicular to the gate. We could use LiDAR and our CV Deep Learning pipeline to perceive and keep a constant view of each gate until we pass through.

Speed Gate

The speed gate challenge takes integral movement and perception functionalities and assesses the boat's ability to perform them smoothly and quickly.

We can use a similar strategy to the navigation channel for perceiving the initial gates and localizing the boat relative to them. Using the normal vector of the gate, we can set a waypoint straight outward toward the Mark buoy and pick it up with our perception stack along the way. We can then engage the Task Planner's circle mode with the Mark buoy as our center, with a reasonable radius and fraction set to pi, navigate in a semicircle around the buoy and travel back to the saved gate location we set out from.

Obstacle Channel

The obstacle channel certainly tests every aspect of the boat on some level but focuses heavily on testing our ability to avoid obstacles, accurately identify colors, and change directions precisely and smoothly. In general, we would plan to perceive the initial (closest) gate as in navigation channel, then chart a spline using our theta* path planner to each subsequent gate.

Our LiDAR perception stack can accurately detect buoys and place them in the boat's costmap but lacks color information. We would use the color information obtained from the CV deep learning model to differentiate between yellow obstacle buoys and the red/green gates, avoiding the situation where an obstacle buoy might be treated as one side of a gate. With separate gate and obstacle detections, the path planner would easily be able to chart a spline from one gate to the next.

Strafing could also come in handy for this challenge. We think that beginning with the boat's yaw perpendicular to the first gate, then strafing through the channel would be much more controlled than changing the boat's yaw multiple times throughout the curved channel.

Obstacle Field

As one of the more intimidating and open-ended challenges on the course, the obstacle field would heavily test our boat's ability to find the path-of-least-resistance in and out of the relatively dense buoy field, and our ability to navigate in a circle under relatively close quarters.

We would plan to sit at the starting point on the edge of the field, which would keep much of the field within our boat's costmap. We would certainly be close enough for our new CV deep learning pipeline to perceive the OB buoy, so we could then pick a point just in front of the buoy and chart a spline through the front of the field and into the inner ring using the costmap generated by the boat's LiDARs. Once inside, we would use the task planner's newly implemented circle mode to navigate around the central buoy, then chart a spline back out to our starting point.

Acoustic Docking

Our main strategy for finding the pinger for acoustic docking is to use an array of 4 hydrophones in a rectangular fashion. We aimed to give an accurate measurement at 60 hertz, which will then be communicated with the main computer.

The first part to filter the hydrophone data is to find the specific wavelength of the pinger. To do this, we simply take the largest magnitude from computing the Fourier transformation against a series of possible wavelengths. Using the Fourier transformation also gives the phase shift of the data from the hydrophone, which can then be compared to the other hydrophones to find the difference in time when one hydrophone listens to the other hydrophones given that sound in water is constant.

Lastly, the differences in phase shift from the hydrophone data can create a hyperbola between every 2 pingers, which then creates 6 hyperbolas. We then use binary search to find the intersection of the hyperbola and determine the location of the pinger.

To achieve higher accuracy, we have decided to run the program to determine the location of the hydrophones 60 times a second, which allows for 3200 data points per hydrophone every cycle.

Object Delivery

With our previous success flying a drone at the competition, we felt that the most reliable method for object delivery was to use an aerial delivery system. Unlike a marine system, this can transport all objects at once and can deliver them to the center of the target, making it less likely for objects to fall off the dock.

We determined the key component of the design of this system to be the interoperability between the boat and drone. Thus, our main goal this year was to improve our system for returning to the boat after delivery, while a subgoal was to create a system to reliably communicate the boat's location to the drone. We accomplished this by creating a custom MAVLINK dialect, which includes specialized messages allowing the boat to communicate accurately and concisely with the drone. To improve the reliability of our landing system, the boat was designed with a larger deck size than previously used, enabling a larger landing pad for the drone and making it easier for the drone to find and land on the boat.

By choosing to use an aerial drone for object delivery, we committed to the long-term success of our team. We have found in the past that an autonomous aerial drone is eye-catching

for potential team members, which helps draw in and retain new recruits. We also formalized a dedicated drone subteam for our organization. This will help us accumulate knowledge and experience over time, which is particularly vital since some of our most experienced drone builders are graduating this year.

Design Creativity

Trimaran Design

For our hull design this year we decided to construct a trimaran hull form instead of the monohull design we had last year. The main reason for this change in hull form was due to concerns about the stability of the boat. Specifically, even though our previous monohull design was not significantly unstable, it did suffer from a very noticeable pitch downward when going forward, due to the additional weight that is typical at the front of our boat. This comes from the various sensors and mounts which must be placed at the front of the boat for visibility and operational reasons. This year with our trimaran design we were able to greatly reduce this issue. Utilizing a trimaran design also motivated our decision in naming the boat *C-3PO*, as we felt the 3 in the name was representative of our three identical hulls.

In our design, one of the first things you will notice is that the front of the boat has two hulls below the deck and in the back, there is only one hull. This was not an intuitive decision but doing this allowed for maximum buoyancy at the front of our boat which was our goal. As mentioned before, in last year's design having all our sensors at the front of the boat, caused the monohull design to pitch downward when going forward. With this decision to have two hulls in the front, we were able to have the maximum buoyancy of our ship just under where the sensors were mounted, so this eliminated any pitching moment we previously had. Eliminating this pitching problem greatly improved our boat's overall stability.

The third remaining hull in the back of the boat was kept to give the boat more length. We concluded to keep the third hull also based on our previous monohull design, which besides the pitching problems, was one of our most stable boats. We attributed this to the large area that the monohull was able to cover, so having a third hull allowed for similar effects. It added length which ultimately added to the surface area of the boat in contact with the water, allowing it to be more stable and buoyant. Having that surface area also helped with the spacing on deck equipment, because we were able to mount sensors at the very front of the deck and the electrical box more towards the middle of our boat where the two forward hulls and single rear hull met.

This additional length also served a dual purpose, because, in addition to adding stability, it was needed to allow the boat to have a deck space at the back. The additional deck space was used to give the drone a landing space.

The second most noticeable decision in our design is the three comprising hulls all being completely identical. This is quite different from the normal trimaran designs, which typically have a larger center hull and smaller outlying hulls. This decision allowed for a very quick and easy construction of the boat. Using only two identical molds to create all of our hulls, we were able to create the two front lying hulls at the same time and then the last hull very quickly just after.

Vacuum Infusion

This year, the hull for *C-3PO* was constructed using a carbon fiber vacuum resin infusion process. In previous years, the team utilized pre-resin-impregnated, high-temperature curing carbon fiber cured in an autoclave to construct the boat hulls, and while the team had experimented with vacuum infusion parts in the past, this was the first year that the team's competition hull was made using this process. Due to the pandemic, the team's normal access to our sponsor, Offshore Spars, which allows the team to utilize their industrial autoclave, was disrupted. The team was required to resort to entirely in-house methods of fabrication.

C-3PO's trimaran hull form was designed to utilize modular construction methods. Each of the three demi-hulls was only 36 inches long and were manufactured one by one in the team workspace within the university's Wilson Student Team Project Center. Constructing three individual demi-hulls was significantly more manageable than fabricating a single hull structure the entire 70-inch length of *C-3PO*.

Each demi-hull was fabricated using a machine-milled high-density foam mold. Dual-weave carbon fiber fabric was cut to shape and three layers of the fabric were placed within

the mold along with a layer of flow media on top. The entirety of the mold was enclosed within a vacuum bag and sealed. The bag was connected to a vacuum system via an excess-resin-collection pot, and a feed hose was added connected to a system of spiral-cut tubing placed beneath the vacuum bag and running around the top edge of the part. Slow-curing two-part resin epoxy was mixed in a pail and the open end of the feed hose was placed into the pail and secured in place. The vacuum system was activated, and the air was sucked out of the system. As the air within the vacuum bag depleted, the vacuum system began drawing the mixed resin from the pail, through the feed hose, and into the part within the mold. The resin flowed slowly through the part from the top of the mold where it flowed freely through the spiral-cut tubing, and then downward through the flow media while flowing into the weave of the carbon fiber fabric throughout the part. Upon flowing to the lowest point of the part, the excess resin was sucked up the vacuum outflow hose and collected in the resin-collection pot. Once resin was distributed throughout the part, the feed hose was sealed off and the system was left under vacuum. Through this process, resin becomes trapped within the weave of the carbon fiber fabric, where it cures over several days and gives the hull rigidity in its desired shape. Once the resin had cured, the hull was removed from the mold.

In comparison to using high-temperature curing, pre-impregnated carbon fiber fabric, and an autoclave, vacuum infusion produces parts that are slightly heavier due to slightly more resin than optimal being included in the part. However, it is easier to produce parts with more complex curvature with the vacuum infusion method, as the woven carbon fiber fabric used is more flexible than the stiff, pre-impregnated carbon fiber sheets used with the autoclave.

Electrical Box Organization and Reliability

Due to the pandemic, our sub team was not able to prototype new components as much as we wanted. Although we eventually got around to test a few concepts, our general mantra was to be efficient by completing tasks with the greatest effect on the end-product while being time-centric, as time is not a given in these uncertain times.

With this goal in mind, our main focus this year was to make the box more organized, neat, and reliable. In past years, such tasks were not desirable due to the many other tasks at hand. We were very happy with the computing power on-hand and decided that these aforementioned goals would make the biggest difference.

In addition, we wanted to smoothen out our competition experience. If our box was neater and more organized, diagnosing issues at competition, if they were to appear, would be extremely less stressful and far easier to debug. To achieve these goals, we introduced numerous new parts and custom PCB's. For example, in order to supply 5V, 12V, and 24V sources to outputs, internal parts, and miscellaneous components, we would simply cut an ATX cable from a power supply and splice the wires onto them directly. However, this year we adopted a better strategy - purchase a breakout board, which splits an ATX cable into its derivative voltage supplies.

In addition, for an "arduino shield" which controls the relays and stacklight, we used a very complex, contrived, and messy custom circuit board, probably created several years ago. In order to neaten this, we created a custom PCB in order to control these components. This PCB is currently in the process of being designed into its "v2" version which should come with numerous upgrades propelling it ahead of its v1 version and the monolithic circuit board of the past in terms of neatness, effectiveness, and reliability.

Lastly, we overhauled nearly every single electrical splice connection. In years prior, we soldered everything. Although the reliability was "probably ok", this was uncertain, not to mention the tens of hours needed to solder everything. Thus, we adopted a more crimp-oriented approach for this year. We utilized anderson connectors for most of our high-voltage connections, regular spades for some of our medium and low voltage connections, and small wago connectors for outputs and inputs from the box. While we still had some soldering joints, the time required to wire the entire box reduced from about 2 months, to a couple weeks. Not to mention, soldering cannot be done safely inside a house without proper precautions! Utilizing different types of heat-less connectors allowed us to work smarter as well as safer.

New Electrical Box

Over the past couple of years, we have been utilizing custom-cut polycarbonate to construct the main "shell" of the electrical box. However, this year, we wanted to be sure that we would have time to improve components that could have a more tangible effect on the end product.

Such improvements included greater neatness, reliability, and power efficiency, which was made possible due to the purchase of a pelican box, a rigid, waterproof, and air-tight crate. In last year's box, as it was custom, we were very unsure as to whether it was waterproof, banking on the fact that the box would never come into contact with water. However, if such a case happens in the future, we are certain that all of our expensive electronics will stay safe. In order to drill holes in the box for some inputs and outputs, we needed to expose members to different tools needed to penetrate the thick material. Although this process was time consuming, all of the sub-team members were able to contribute, learn a new skill, and the electrical box was completed almost two months ahead of where it was last year.

In order to choose the box itself, CAD programs needed to be utilized to map out the components and how they would fit within the box. In past years, we placed components wherever there was space without regard for specific placement. We wanted to change that this year. Components that need other components would be placed in closer proximity. For example, all the components needed to supply power to the thruster electronic speed controllers were all placed in a line: Circuit breaker - relay - buck converters - ESC. Although this slightly increased box dimensions, we felt it was a worthwhile goal. Simulating the components in the CAD program greatly allowed us to speed up building the box. In past years, we would figure it out as it goes, without strict adherence to the originally designed CAD, due to it not being "complete." We wanted to avoid that this year. CAD was completed in the summer, with ample time to look this over, consult with the Hulls and AI team, and to draw the layout on paper to see it first hand, much like how an architect sketches a building. Doing this allowed the box to be built seamlessly and quickly (within a couple days vs. a couple of weeks).

Hydrophone Array and Processing

Our design for the hydrophones is to use a Raspberry Pi as our processing power and then communicate via USB to the main computer.

We based our decision on using a Raspberry Pi on the processing power. Despite the fact that an FPGA could allow more parallel processing power, we opted to use a 192kHz sound card and then connect it to a Raspberry Pi. This decision was based on the amount of time it takes to program on an FPGA. The Raspberry Pi is fully capable of computing the hydrophone array while also being a cheaper option than the previous year's work.

In order to reduce interference, the hydrophones are placed at the very bottom of the boat. They are placed in a rectangular fashion to determine both the direction and the angle of the pinger, as compared to a straight line in which the pinger would have 2 possible locations.

CV deep learning

The objective of the CV deep learning team was to detect and classify objects of interest from the zed cameras using a convolutional neural network (CNN). The objects of interest were the spherical buoys, the cylindrical buoys, and the docks. The main advantage of using deep learning instead of the LIDAR alone to detect objects was that the data was richer - i.e., LIDAR cannot discern colors. Last year, we attempted to use classical CV methods (e.g., contours, color masking, segmentation, etc) to process the camera data, but it was too sensitive to noise - the accuracy of these methods was no better than random. This year, we decided that manual image processing was unnecessary, and that deep learning methods would be much more promising.

The deep learning models we explored were YOLOv3 and Faster RCNN. The motivation for choosing these models was that the model had to run in real time - speed was an important factor. Due to the novelty of these methods and the research required, we were unable to fully implement Faster RCNN. However, the results of YOLOv3 were a significant improvement from the classical methods. For the 9 object classes, we achieved an accuracy of 56% when tested on a dataset from a different source than the training

data. While this is a significant improvement, there is still work to be done.

LIDAR quantification

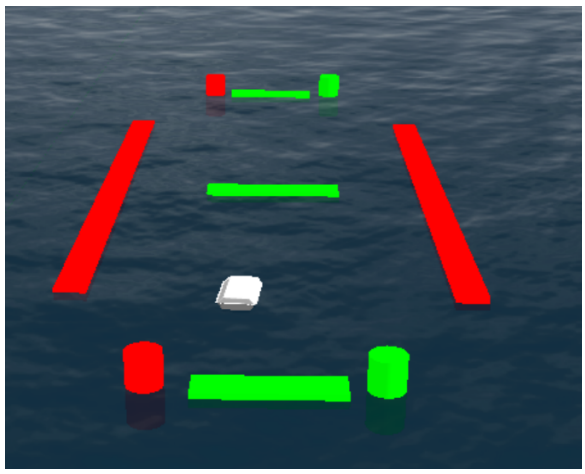
In previous years, our methods to evaluate the quality of our LIDAR object detector were not mathematical. Instead, we used rviz to visualize our raw, time-series point cloud data overlaid with detection markers, in order to roughly evaluate if the detections were reasonably accurate. In order to realize a baseline for our algorithm to use as a basis to improve in the future, we decided to explore more formal mathematical metrics to evaluate the quality of the LIDAR algorithm.

We primarily sought to evaluate our algorithm using a confusion matrix. In order to create the ground truth data necessary to generate it, we annotated 645 data points, each consisting of a raw point cloud at a certain timestamp containing one or more objects of interest. From the matrix, we saw that the algorithm achieved 45% total accuracy, a maximum precision of 65% with the “sphere buoy” class, and a maximum recall of 79% with the “cylinder buoy” class. Though our results were generally poor, we plan to utilize them as a baseline from which to continue developing our LIDAR-based classification in the future.

Gazebo Task Simulation

Gazebo task simulations allow for system-level tests of the boat’s functionality, putting the boat through a virtual competition course.

In order to quantify the success of a run, our simulated tasks include collisionless "green" and "red" zones that add and subtract from the current score on intersection with the boat, respectively. Course obstacles, such as buoys, double as "red" zones. Examplefied in the "autonomous navigation" task simulated below, red zones are added to either side of the challenge, while green zones are added to the middle. Thus, the boat will gain points by not veering off course.



Gazebo simulation of an "autonomous navigation" task.

The user has the option to dictate an order the boat must reach the green zones. In addition, the user can specify a time limit, after which the points for reaching a green zone start decreasing, and if left long enough will result in timing out the run.

This scoring functionality culminated with integrating the above "autonomous navigation" course into the Gitlab CI pipeline. Now, whenever a club member tries to push a change to the codebase, the above simulation will be executed, and if the run doesn't achieve a baseline score within the time limit, then the push will be rejected because something must be broken.

We wanted the ability to isolate one subsection of the boat’s code for testing by having the other components output some known, correct values. This way we can try to identify errors or inefficiencies at the source without worrying about cascading errors across multiple sections of code. When running the simulator, the user can choose to run a “fake” implementation of the high-level planner, path-planner, or controller instead of the real code for that section.

Task planning

Task planning implements the boat’s high-level decision making by interfacing with perception, path planning, and controls to understand the course and complete challenges. With no in-person competition last year, the task planner hadn’t been updated since 2019 so we took this year to implement some useful functionalities.

Our main goal and accomplishment was to implement the interface for general, common

movements usable throughout the course and rethink our challenge approaches based on these new maneuvers. Our chosen movements allow the boat to “anchor” itself, strafe, and circle around a fixed object.

Anchor mode allows the boat to choose a location and keep itself there. This might be useful for cases like a acoustic docking, where remaining stationary would make localizing the source of the much easier

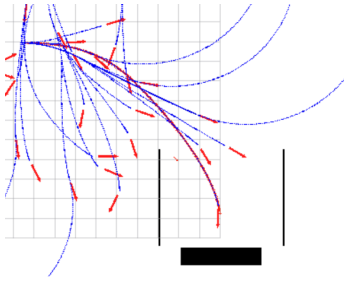
Strafing enables the boat to move sideways without changing its yaw. Strafing through the obstacle channel could help keep the yellow A-2 buoys in front of the boat and control our movements as we pass through the gates. We could also use strafing to sidle up to the dock in acoustic docking.

Circle mode allows us to specify a center, radius, and portion of a circle for the boat to traverse along. This would be useful for challenges like the obstacle field or speed gates where circular and semi-circular movement around a buoy is required.

Creating Paths for Nonholonomic Robots

Last year's path planner did not guarantee that the output trajectory could be feasibly followed. Even more problematic, the previous path planner did not take into account yaw value at all. Our new RRT based planner outputs a path feasible for our boat's dynamics, and guarantees the boat's finishes with the desired orientation.

The solution implemented this semester was an algorithm specific for non-holonomic vehicles, Theta*-RRT. This implementation is based on *RRT-Based Nonholonomic Motion Planning Using Any-Angle Path Biasing* (Palmieri 2016).



Path visualization of the boat.

Above is a visualization of the boat trying to find a path to the middle of the parking stall. Blue lines are the path segments feasible by our boat's dynamics connecting the red, random sample states.

Universal Dev Kit

Over the many years the team has existed, we have collected a large amount of unlabeled data in the form of ROS bags. To use this data to train machine learning models, it was crucial to converting the data into a usable format (png/jpg files for camera data and PCD files for LiDAR data). Additionally, we needed to label the data to identify instances of buoys or docks. We started with the vision team and LiDAR teams individually labeling data for their respective tasks. However, this resulted in the data only being usable for the group that labeled it. Additionally, it wasn't easy to organize the labeled data when we had multiple people labeling multiple bags, some of which were only partially completed.

We needed to develop a way to take all the data we had labeled in its various data formats (YOLO for the images and a format unique to the annotation tool we used for LiDAR) and provide a simple interface for producing data to train models on.

After exploring what public datasets are using for interfacing with their datasets, we decided to model our interface on the nuScenes' DevKit, a Python interface for the nuScenes dataset. The nuScenes' DevKit provides methods to easily query for specific attributes (which we could use if we wanted images of buoys of a certain color), allows tracking objects through multiple frames (which is useful if the team decides to use a temporal model in the future), and provides transformations from the 3D LiDAR annotations into 2D bounding boxes. This last capability means that the UMA teams could only annotate in 3D and receive both 3D annotations and automatically generated 2D annotations.

We had originally hoped to use the nuScenes DevKit to interface with our dataset directly, so our initial focus was on converting our dataset to match the format of nuScenes. There were no existing tools to do this, so we needed to develop our implementation. However, we eventually realized that we would be unable to achieve this because the nuScenes dataset uses GPS coordinates, maps of the road network, and other features that we did not have.

We decided to pivot our work from attempting to convert our data to a format usable by nuScenes to creating our devkit, called the Universal DevKit. Our idea for the Universal DevKit was to provide the features we liked the most from the nuScenes' DevKit but require as little input as possible (no requirements on GPS, maps, etc.). We open-sourced our work and made significant progress on the DevKit. Our current work can be found on GitHub, and we will finish up the project in the coming semester.

Drone

A custom designed H-frame was fabricated to accommodate the UAV portion of the competition tasks. The H-frame was chosen over the previous X-frame design to increase stability in flight and allow for a space-efficient payload delivery mechanism, and also provided a larger overall footprint designed around a stronger flight system. Due to COVID-19 restrictions, reduced access to our workshop prevented aluminum fabrication used in previous iterations of the team's UAV solution. Instead, prototyping and final fabrication of all components excluding 4 carbon fiber frame tubes were manufactured using the team's 3D printer. The central electrical system was reworked for easier access to the components, where all communication and computational elements were accessible from a single top plate, and all vision, rangefinders, and landing components were mounted onto a bottom plate. A 4S LIPO battery sat between these two plates to minimize overall footprint.

Fundamental differences in the drone's design stemmed from the inclusion of a new payload delivery system to deliver racquetball-sized objects into a landing zone. Two space-efficient drop bays were integrated into the existing H-frame to maintain stability and efficiently use the existing space. Bay doors were custom designed and manufactured using a 3D printer, while the frame itself prevented translational motion acting as walls. Each bay door was held in place by a micro servo, controlled through Raspberry Pi that also served as an extended computational unit. Servos were chosen based on their overall motor strength, and their PWM signals controlled both the speed and angle for bay doors to open in order for objects to roll towards the UAV's center in a controlled fashion.

Aerial Software System Design

R2-D2's software is centered around a finite state machine. This year, we have expanded its design by adding new search patterns for landing, improved the design of the computer vision system, and formalized the documentation of our software architecture. Our focus on design and documentation was driven by the need to collaborate remotely during the pandemic, but also helped us improve the safety and efficacy of our system design.

Our new search pattern for dock landing uses the provided waypoint of the dock, as in past years. However, the new design adds a grid of waypoints centered around this target, which *R2-D2* traverses until the dock is found using computer vision. This lowers the chance of failing to find the dock due to drift or other errors, and helps *R2-D2* land near the dock's center to avoid falling or dropping the payload into the water.

Our new search pattern for boat landing replaces the IR-lock used in the past with a computer vision system as the primary method to center on *C-3PO*. We made this change because the IR-lock was not accurate enough to reliably land the drone on target. Instead, our new design resembles that of the dock landing system, using the GPS coordinates of *C-3PO* (sent using our MAVLINK communication interface) as an initial target location then refined with computer vision. However, in case the CV system loses sight of *C-3PO* during the landing process, the IR lock is retained as a secondary landing procedure.

The computer vision system corrects *R2-D2*'s horizontal position during landing by identifying the landing zone and its center. It is designed to rely on a neural net model, which signals the finite state machine to strafe *R2-D2* towards the center of the target. To address the limited availability of training data for aerial vehicles in our competition, we have introduced an image generation script that applies transformations and color shifts to existing images of the dock and boat, thus increasing the effective size of the training set.

Drone Electrical System Design

This year's design consolidates R2-D2's electrical system to use one telemetry radio that

handles communication with both the boat and the ground control station, rather than separate radios for each external device. The change saves space and weight while simplifying the electrical system, reducing the number of points of failure. We have also replaced our light-based rangefinder with an ultrasonic sensor in order to more accurately measure *R2-D2*'s altitude over water.

Experimental Results

Marine Dynamics Model

In the past few years, UM::Autonomy has been working to add more advanced maneuvering and motion capabilities to the team's boat. These efforts have included the addition of what are essentially bow and stern thrusters, with which the team seeks to add lateral motion capabilities to the boat's maneuvering. However, these additions have forced UM::Autonomy to confront the present lack of sophistication in the team's dynamics model which is used to govern the boat's autonomous control system.

The issues with the existing dynamics model were partly due to the knowledge of UM::Autonomy's programming team being historically rooted in autonomous cars and land vehicles, with significantly less familiarity with the dynamics of marine surface vehicles. The goal of this project was to develop a new and improved marine-dynamics-based model for the team by integrating knowledge from marine dynamics, hydrodynamics, and computer modeling.

The model was built by members of the UM::Autonomy Hulls & Systems subteam during the Winter 2021 semester using a combination of MATLAB, Rhino, and OpenFOAM. The model was built using the team's 2020-21 season boat, *C-3PO*, but the main objective of the project was to build the model in such a way that it can be easily adapted to be applied to new boats in future seasons.

It was a goal of the project that the model should be able to account for non-linearity in the coefficients of the boat's motion. The model follows the standard marine dynamics form of:

$$([M] + [A])\,\overline{\eta}'' + [B]\,\overline{\eta}' + [C]\,\overline{\eta} = \sum \overline{F}_{ext}$$

with all six dynamic degrees of freedom included. The model accounts for nonlinearity by continuously updating the coefficients in the stiffness matrix, [C], and damping matrix, [B], based on the boat's position and velocity. The model also includes hydrodynamic added-mass effects, accounted for in the added-mass matrix, [A].

Several nonlinear regressions were performed using collected datasets encapsulating aspects of the boat's dynamic behavior. These regressions describe the nonlinear relationships between the boat's motion and forces resulting from hydrostatic and hydrodynamic sources, and facilitate the generation of the force coefficients populating the matrices in the model based on the boat's position and velocity.

Ultimately, the nonlinear dynamics model has proven to be capable of realistically representing *C-3PO's* dynamics and motion. It is vital that more marine-dynamics-informed methods be applied to UM::Autonomy's practices, and it was important to seize the opportunity to undertake this project this year while the team had a member with the necessary experience from taking senior-level marine dynamics courses within the University of Michigan's Department of Naval Architecture & Marine Engineering. It is unknown at this time whether what was produced this semester by the team working on this project will be able to be directly integrated into this or future UM::Autonomy boats' autonomous control systems or the team's virtual simulation efforts, but hopefully this project will have left behind a tool which will be useful to more programming-focused team members in helping them think about the boat's motion from a marine dynamics point of view.

Outdoor Testing

In the last few years, the team had been working towards a testing timeline in which the team can focus on integrating projects and establishing baseline functionalities in the spring. With the restrictions in place through the year, however, we put this testing goal on hold in favor of a relaxed timeline to accommodate the difficulties of gathering. We placed our focus on two testing periods; first, testing last year's boat at the beginning of the fall to recover operational knowledge, and second, having a brief testing period at the end of the spring to gauge where the team is at for future development.

With our fall testing period, our largest priority was to continue evaluating last year's boat, *Pass*, with our new team of officers. Without a traditional competition cycle, institutional knowledge about the testing and operation of the boat could not be handed down by senior members as normal, so this period was largely successful in getting that experience to our newer officers. After a few delays in hardware nuances, we had successful testing sessions whilst teleoperating *Pass*, although a few software flaws prevented meaningful autonomous navigation.

With our late spring testing, we were able to assemble *C-3PO* for one session on the campus pond. Thanks to diligent design and testing, our hardware worked flawlessly from the start, allowing us to teleoperate *C-3PO* and gauge its performance. Unfortunately, a critical error prevented our environment from starting, preventing us from evaluating our autonomous capabilities.

Remote Engagement

With many student organizations at the University of Michigan going remote, the team anticipated a drop in recruitment and member retention due to the decreased social engagement and lack of recruiting opportunities. To combat the potential decrease in members, the team experimented with various forms of virtual activities and structured the team to be more focused on new member onboarding and training. These activities included game nights, cooking nights, tech talks, virtual all-hands meetings where members could learn about other subteam projects, a team minecraft server, and more.

As anticipated, the recruitment season held much less traffic than usual; however, our efforts to focus on newer members held a higher retention rate among newer members. The social events had mixed results as the earlier social events were well received; however, as the school year became busier, participation in these optional events dropped. The results of a stronger focus on new member training was also mixed. For software oriented subteams, the focus yielded great new member recruitment and retention. For hardware oriented teams, it was much harder to train new members remotely.

Specifically for our hulls and systems team this was a challenge we tried to overcome. During a number of the hills and systems regular weekly zoom meetings, the team was able to incorporate virtual mentoring sessions which had some significant pay off. These sessions were held by more senior members of the team who would guide the newly joined members in learning more knowledge useful to their contributions to the team. One example of this was the ship terminology and basic principles learning session where the new members were able to get more acquainted with various terms and ideas that may otherwise never be explained but often used. Concepts such as ship degrees of motion, ship geometry, buoyancy, and centers of weight and buoyancy were all examples of topics discussed. Other instances of these mentoring sessions also incorporated the downloading and basic introduction to various modeling softwares we utilize on the team such as Rhino, AutoCad, SolidWorks, etc. The hulls and systems team is typically all hands-on, in person work but having knowledge of these softwares allowed our newer members, who were almost all remote, to be able to contribute to the design and construction of the boat even if it was only virtually.

Conclusion

The 2020-21 build season challenged our team's ability to adapt to a largely virtual format. We continued to uphold our commitment to member safety in the midst of a global pandemic and sought learning opportunities for the team to continue , experimenting and building together. Looking back at the team's progress the past year, we are proud of our growth and are excited about what's to come.

Acknowledgements

The existence and success of our team depends on the incredible support of the University of Michigan, our advisor, our committed alumni, and our industry sponsors. We must highlight the incredible support we receive from the Ford Motor Company. A special thanks to Professor Kevin Maki for being the team's advisor and his mentorship in naval architecture and Tony Lockwood at Ford AV LLC for his mentorship in software project management.

References

L. Palmieri, S. Koenig and K. O. Arras, "RRT-based nonholonomic motion planning using any-angle path biasing," *2016 IEEE*

International Conference on Robotics and Automation (ICRA), 2016, pp. 2775-2781, doi: 10.1109/ICRA.2016.7487439.