Advanced Control of an Autonomous Surface Vehicle

In Partial Fulfillment of MECH 498

Mechanical Engineering Honours Thesis

Alec Krawciw

V00220555

December 20, 2021

Supervisor: Professor Yang Shi

Abstract:

A small-scale autonomous boat was assembled to study the dynamic control implications of underactuated vessels. An existing remote-control boat was used as the research platform. The internal control electronics were replaced, and a camera, IMU, and GPS were added. The primary focus of the research was the development of controllers to guide the vessel into a dock. At the scale of the system, GPS precision was found to be insufficient. For this reason, visual-inertial odometry was implemented using fiducial markers to improve the localization of the system. This report will describe some of the different strategies used for path generation and control to achieve autonomous docking.

Table of Contents

Figures

Tables

Background

With the rise of inexpensive sensors and microcontrollers, many dynamic systems are being converted to run autonomously. Much of the research focuses on unmanned ground vehicles and multi-rotors. There are some common issues that are relevant to all unmanned vehicles: locomotion, perception, and navigation [1]. However, unmanned boats are gaining popularity as well. These systems are named Unmanned Surface Vehicles (USVs). Many challenges face all autonomous systems, but some are unique to the marine environment [2]. Surface vessels encounter complex hydrodynamic effects as well as aerodynamic effects. In addition, because of constant bobbing over waves, sensor data must be carefully fused for use in three dimensions. Another control challenge is the fact that most boats are underactuated [3]. Most boats use one thruster and rudder or two fixed thrusters to move in a three-degree of freedom plane.

Introduction

A micro unmanned surface vehicle was converted to run autonomously using visual odometry. The driving goal of this research was to understand some of the constraints associated with underactuated control. This vessel was customized to include an onboard computer, a camera, an IMU, and a GPS. Once the platform had been constructed, there were four focus areas:

1. Dynamics Modelling: Define a set of governing differential equations for the system.
2. Path Generation: Produce a good plan to move the boat from its starting location into a dock.
3. State Estimation: Use onboard sensors to evaluate the current state of the system, which consists of both position and velocity.
4. Control: Use the defined path and current state to follow the path as closely as possible.

The task of docking was chosen to represent a common purpose for all these research components. The boat is pictured in Figure 1.

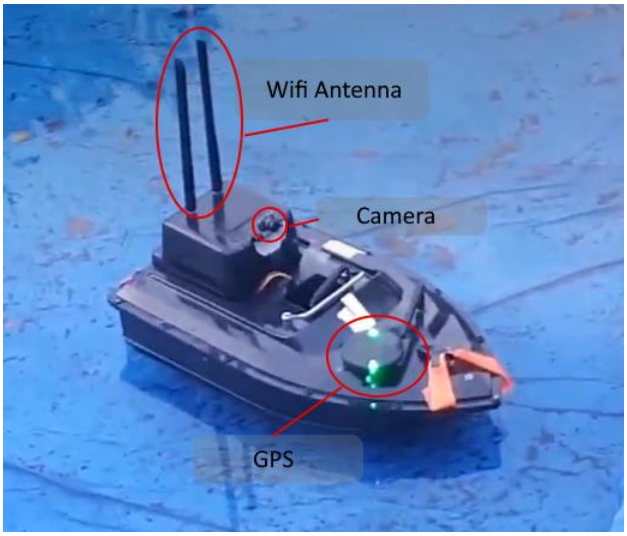

Figure 1: Completed USV with critical sensors highlighted.

System Architecture

A Pixhawk autopilot controller was chosen as the central vessel control unit (VCU). The Pixhawk contains a 32-bit ARM Cortex M4 core operating at 168 MHz with 256 KB RAM and 2 MB Flash [4]. The microcontroller runs the open-source ArduRover control stack. While this is

sufficient to manage sensor input and output, there is insufficient computational power to process complicated sensors such as cameras or LiDAR [5]. For this reason, a companion computer is required to interface with other sensors and perform navigation tasks. The companion computer used was an Nvidia Jetson Nano 4GB (board A01). In addition, for diagnostic purposes, the system must communicate with the ground station. It was determined that due to available components, a Wi-Fi link between the ground station computer and the companion computer was optimal. It may be advantageous for future iterations to use a long-range telemetry radio such as an RFD900x [6] for missions requiring a link further than 100m in range.

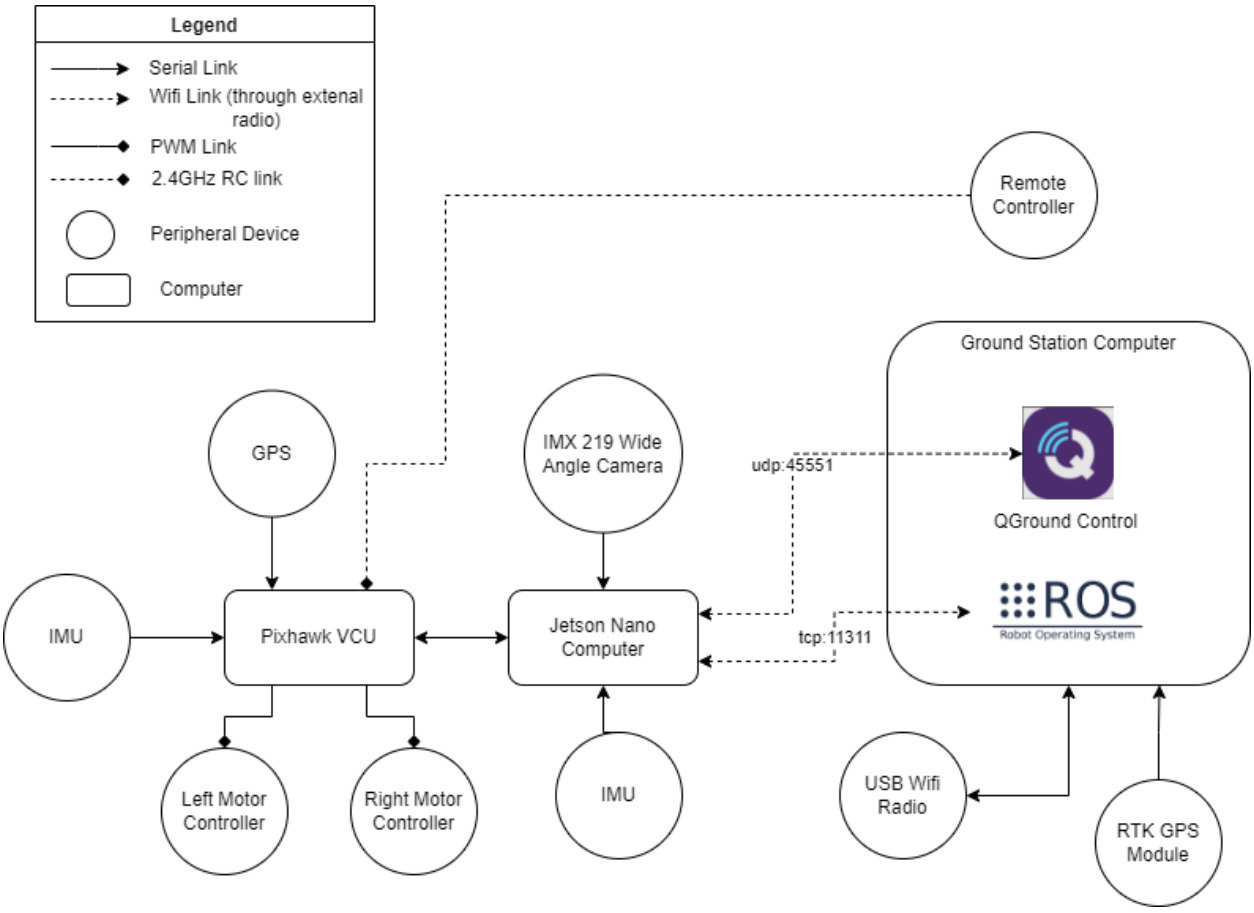

Figure 2: System architecture diagram

The Robot Operating System (ROS) [7] was selected as a middleware framework for advanced control because of the multitude of open-source introspection tools and hardware drivers. The version of ROS is Melodic. The Pixhawk is compatible with ROS through the mavros library [8]. The mavros library exists as a framework to enable hardware commands to be generated by any computer on the ROS network and converted to signals in the VCU.

QGround Control is an open-source ground station platform used to communicate with the Pixhawk VCU [9]. As described in Figure 2, wireless communication happens through a UDP port on the Jetson and is forwarded to the Pixhawk via USB. There can be more than one ground station connected to the Jetson at one time, and this allows multiple users to interact or enable

longer-range operations. QGround Control also provides a convenient dashboard to monitor system status and mission progress.

The Jetson Nano hosts the Wi-Fi network onboard. Using the vehicle as a router allows for fewer components at a mission site and reduces the complexity of networking. A high gain antenna must be used on both the ground station and boat for acceptable range. The details of the network can be found in Appendix B: USV Wifi Configuration Information.

Hardware Conversion

Before autonomous controllers could be investigated, the mobile platform needed to be constructed. A ZHM T088 remote-controlled boat was chosen to serve as the platform. The T088 has two 8V brushed DC motors. This configuration is called a differential drive. The boat was powered by a 2S (8.4V) lithium battery. The off the shelf controller board was replaced with two motor controllers, a voltage regulator, and an RC radio unit to interface the Pixhawk VCU to the motors.

Figure 3: USV hull with upgraded motor controllers and battery.

State Estimation/Localization Using Visual Odometry

One of the attractive primary features of ROS is the vast set of open-source packages that help to integrate sensors and common algorithms into the plug-and-play publisher-subscriber framework. This project selected the ar_track_alvar package [10]. This package is used to determine a camera's relative position and orientation (pose) with respect to a known target or fiducial marker. The targets, called AR tags, consist of square black and white regions that encode an id, like a QR code. Tag 0 is shown below (Figure 4). Based on camera calibration and prior knowledge of the target's size, the relative position can be extracted from the image distortion and size.

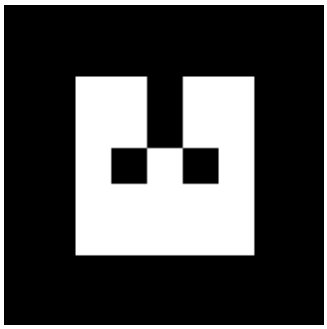

Figure 4: AR tag ID 0

Camera Calibration

The wide-angle lens used on the boat was calibrated using the ROS camera calibration package. The results of the calibration are shown below. As per the findings of Chou [11], the calibration was performed with a static checkerboard and a mobile camera. The lens was covered while moving to reduce the effect of motion blur during calibration.

AR Tag Layout

In this project, fifteen AR tags are mounted on and around the dock to provide a centimetre precision guide for the boat as it approaches the target. The size of the pool is 2.5m x 5m.

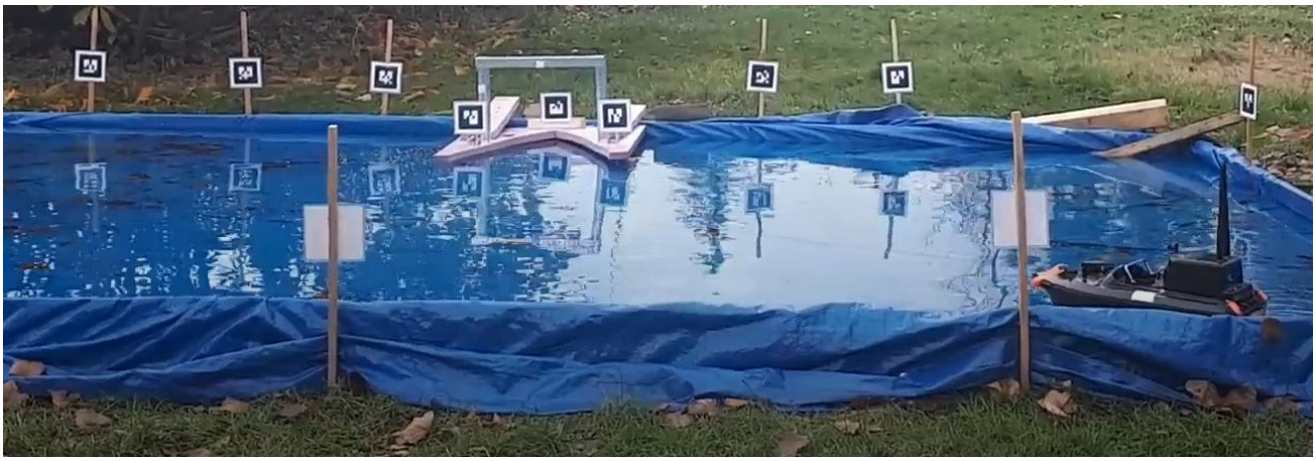

Figure 5: AR Tags located around the pool.

A set of four standard coordinate frames define the motion.

Table 1: Coordinate frame descriptions

Frame Name	Motion Type	Description
`cg_ned`	Body-Fixed	Attached to the boat at the centre of gravity
`base_link`	Body-Fixed	Attached to the boat at the Pixhawk
`csi_cam_link_0`	Body-Fixed	Attached to the boat at the camera
`map`	Earth-Fixed	Centred at the mouth of the dock

Figure 6 shows the local north-east-down (NED) frame (`cg_ned`) attached to the centre of mass of the boat and the fixed frame used for positioning (`map`). The origin of `map` is at the mouth of the dock centred in the berth coincident with the waterline.

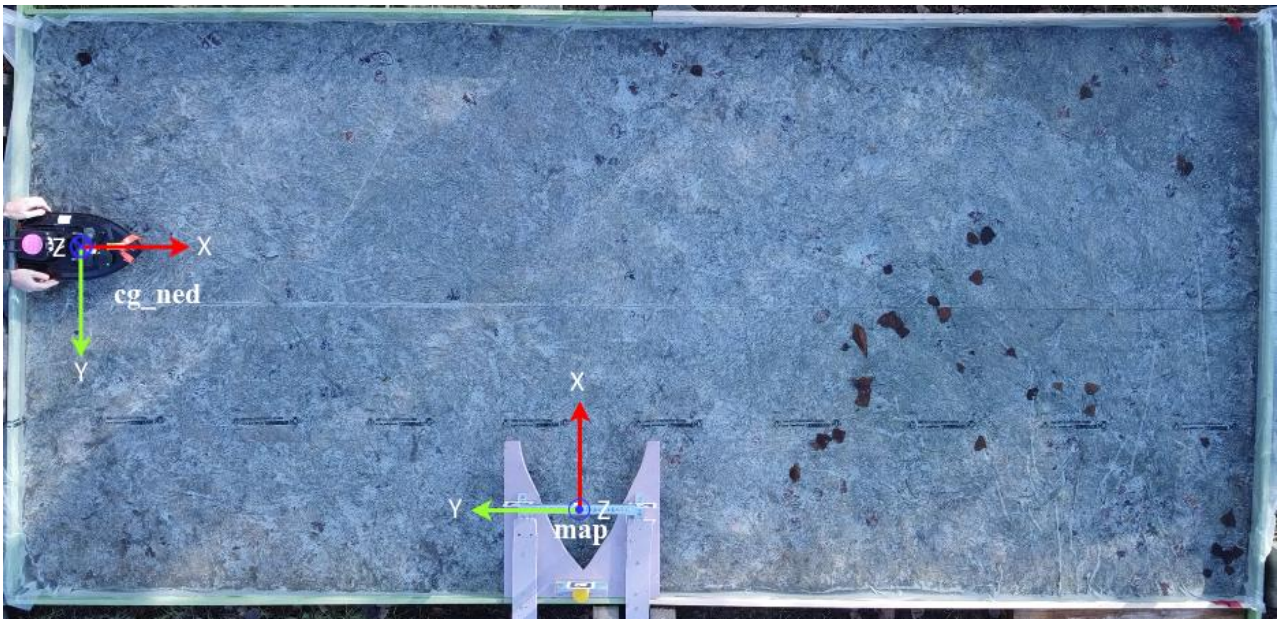

Figure 6: Coordinate frame definitions

As illustrated in Figure 7, the `base_link` frame is centred on the VCU, the `cg_ned` and `csi_cam_0_link` frames are connected statically to the `base_link`. These frames are body-fixed. AR tags are viewed from the `csi_cam_0_link` frame and transformed into the `map` frame, where motion analysis is performed. ROS handles most transformations internally.

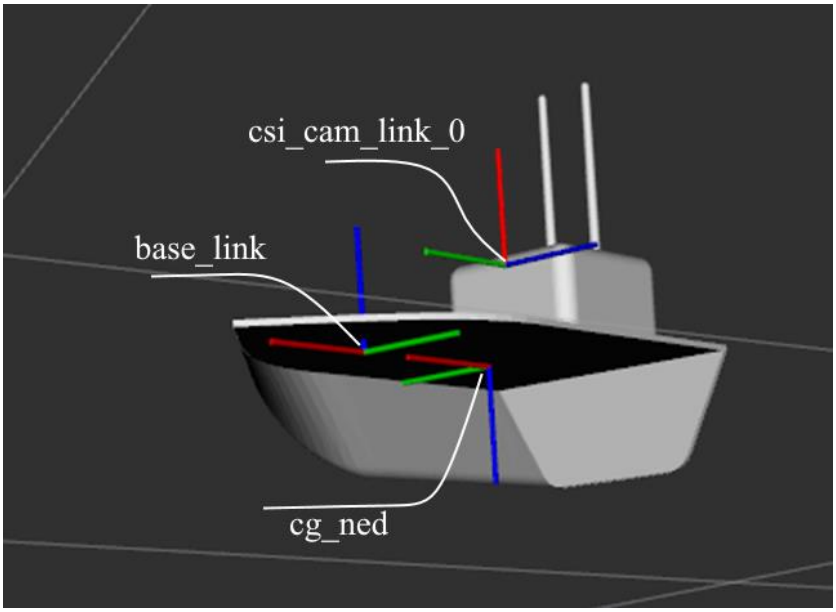

Figure 7: ROS body-fixed frames

State Estimation Validation

To quantify the magnitude of errors between the estimates and real life, an overhead camera attached to a DJI Mini SE drone was used. While the operation was in progress, the drone recorded a video of the test. Afterwards, OpenCV image processing was used to extract the position and orientation of the boat at 30 Hz. A bright yellow circle was placed on the dock to account for the lateral translation of the drone (top of Figure 8). A bright pink dot was placed on the rear of the boat, and a bright green stripe at the front (lower right of Figure 8). Colour filtering in the Hue-Saturation-Value colour space extracts the centroid of each region. From those, the angle and position are calculated. To fit the pool in the field of view of the drone's camera, the resolution of the camera works out to 2.8mm per pixel. The standard deviation of the measurements is 2.6mm and 0.5º for linear and angular data, respectively.

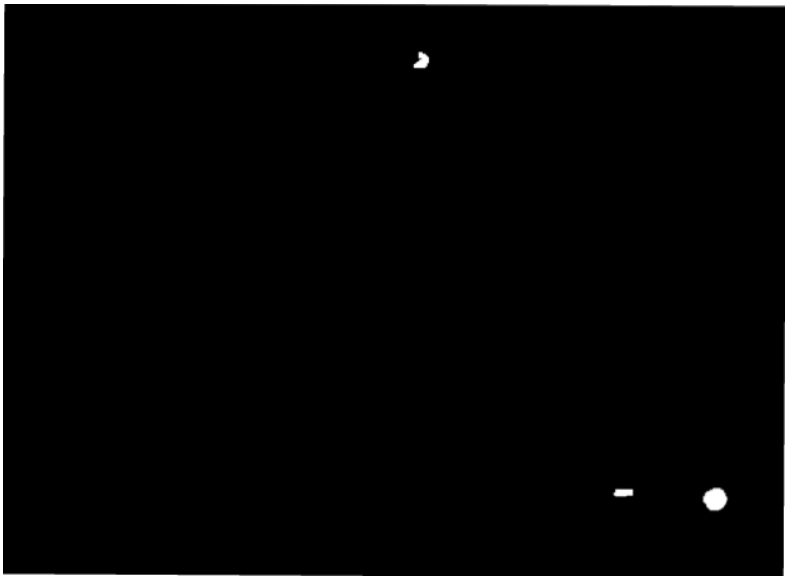

Figure 8: OpenCV colour filtering result

The position data is numerically differentiated to extract velocity curves. The core goal of a ground truth measurement is to assess the quality of the localization result

Dynamics Modelling

To compare different control schemes, an accurate model of the boat is essential. A rigid body model can be created after measuring the critical parameters of the boat. The mass was measured using a scale. The moment of inertia about the z-axis through the centre of gravity was measured using a torsional spring (Figure 9).

Figure 9: Autonomous boat suspended from torsional spring

The torsional spring constant was calibrated using a few objects of standard moments of inertia. The average period of oscillation was used to find the natural frequency. Damping was assumed to be negligible.

$$\omega_d = \sqrt{\frac{(1-\zeta^2)k_\theta}{I_{zz}}} \cong \omega_n = \frac{1}{T_n} = \sqrt{\frac{k_\theta}{I_{zz}}}$$

$$I_{zz} = k_\theta T_n^2$$

The test items were consistent with each other within 0.7%. This approach was applied to the boat. Only the I_{zz} moment was measured since this model only considers motion in the 2D plane. The results are summarized in Table 2.

Table 2: Boat model parameters

Model Parameter	**Value**
Mass	2.2 kg
Inertia (I_{zz})	0.0978 kg · m^2
Length	0.48 m
Beam (Width)	0.26 m
Draft (Depth underwater)	0.065 m
Thruster Separation (d_{sep})	0.165 m

A 2D rigid body dynamics approach was taken to model the problem [3].

$$M\dot{q} + C(q)q = \tau$$

$$q = \begin{bmatrix} u \\ v \\ r \end{bmatrix}$$

where u is the forward velocity of the boat, v is the lateral velocity of the boat, and r is the rotation rate, and M is the mass matrix:

$$M = \begin{bmatrix} m & 0 & 0 \\ 0 & m & 0 \\ 0 & 0 & I_{zz} \end{bmatrix}$$

C consists of two matrices, a rigid body Coriolis matrix (C_{rb}) and a drag matrix (D).

$$C = C_{rb} + D = \begin{bmatrix} 0 & 0 & -mv \\ 0 & 0 & mu \\ mv & -mu & 0 \end{bmatrix} + \begin{bmatrix} D_1|u| & 0 & 0 \\ 0 & D_2|v| & D_3|v| \\ 0 & D_4|r| & D_5|r| \end{bmatrix}$$

To simplify the controller, the τ vector is represented as $\tau = \begin{bmatrix} F \\ 0 \\ T \end{bmatrix}$. The forces generated by each thruster is calculated as $F_{left} = \frac{F}{2} + \frac{T}{d_{sep}}$ and $F_{right} = \frac{F}{2} - \frac{T}{d_{sep}}$.

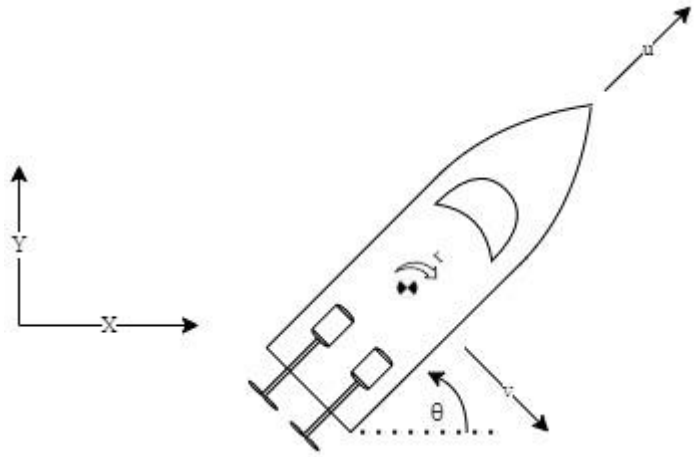

Figure 10: Boat coordinate frames

A Jacobian (J) is used to convert body velocities from the boat frame to the fixed frame.

$$J = \begin{bmatrix} \cos\theta & sin\theta & 0 \\ sin\theta & -cos\theta & 0 \\ 0 & 0 & -1 \end{bmatrix}$$

This approach is based on Klinger et al. [12]. However, the drag coefficients were experimentally determined instead of using cylindrical drag theory due to the hull shape. For initial value problem simulations, a six-state state-space model was used. The velocities and accelerations are represented in the frame `cg_ned`, and the position is recorded in the stationary `map` frame.

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{u} \\ \dot{v} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} 0_{3x3} & J(\theta) \\ 0_{3x3} & -M^{-1}C(q) \end{bmatrix} \begin{bmatrix} x \\ y \\ \theta \\ u \\ v \\ r \end{bmatrix} + \begin{bmatrix} 0_{3x3} & 0_{3x3} \\ 0_{3x3} & M^{-1} \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ F \\ 0 \\ T \end{bmatrix}$$

The model above requires tuning the drag coefficients to represent the boat while it is moving accurately. Several controlled tests were run with the physical boat in conditions of relatively still water to constrain the model parameters. These situations were then simulated using an initial value problem (IVP) approach in MATLAB.

Model Parameter Identification

In general, the five drag coefficients defined in the drag matrix D are functions of θ, but for the sake of simplicity, they are assumed to be constant. For this reason, this model will be unable to model the motion of the boat in reverse accurately. For the docking approaches discussed, this limitation should not be relevant.

$$D = \begin{bmatrix} D_1|u| & 0 & 0 \\ 0 & D_2|v| & D_3|v| \\ 0 & D_4|r| & D_5|r| \end{bmatrix}$$

The most reliable method of determining these parameters involves using a tow tank [13]. The boat would be pulled at known velocities and accelerations with different heading angles to fit the parameters. Without this, known input forces are used and compared to the velocity and acceleration profiles of the boat until they match.

Thrust Testing

In order to set up an accurate IVP simulation, data on the motor's transfer function is required. The motor thrusts were characterized by driving the boat into a load cell. Each motor was driven using PWM input (throttle). The resulting thrust was measured (Figure 11).

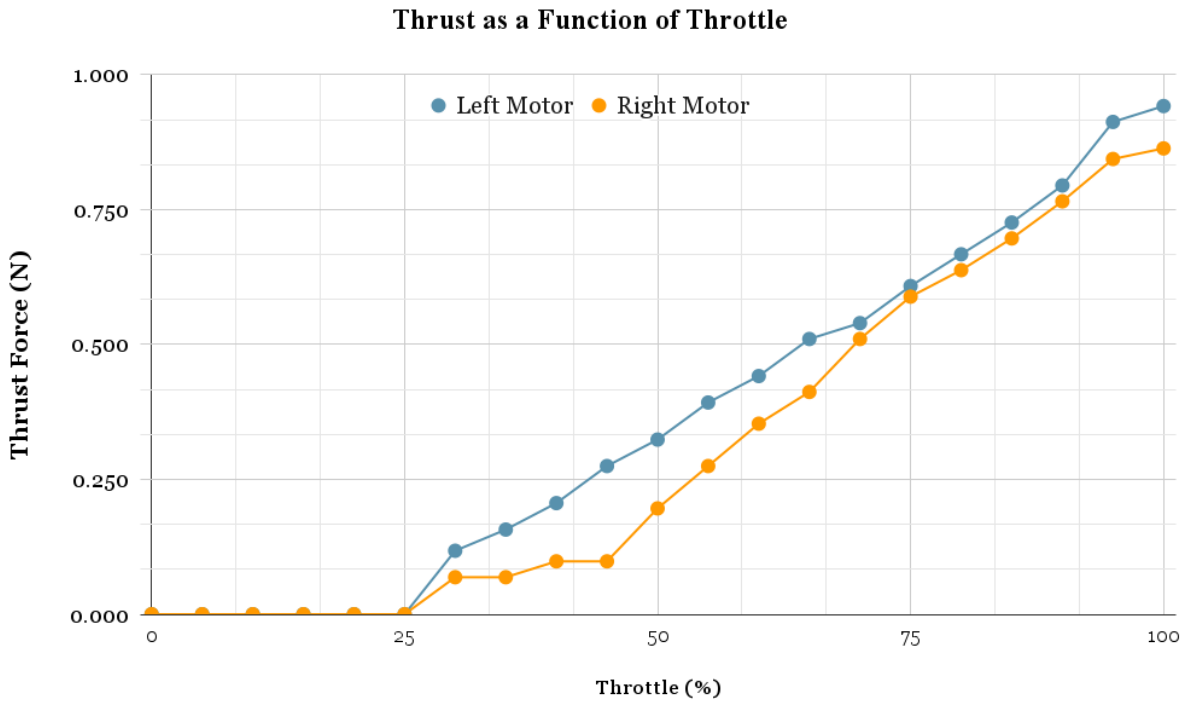

Figure 11: Thrust testing results

A few issues were observed. The primary issue is that the motor does not move until 25% throttle. This dead zone was corrected in the control code. In addition, the right motor is generally weaker than the left. The different slope of the right motor under 50% throttle may contribute to a commonly observed initial rotation during start-up.

Also, the performance of the motors degraded as the voltage dropped. Reduced motor power impacted the gain of the transfer function during extended operations. This system has the DC motor controller connected directly to the battery. It would be advantageous in future to regulate the voltage driving the motors.

Path Generation

Kalaitzakis and Carroll [14] propose many potential solutions for guiding robots within a constrained environment. Of the types proposed, a potential function gradient descent was applied to this system.

Constraints

The docking problem imposes some constraints on the endpoint point of a trajectory. For the sake of simplicity, the coordinate frame used in all discussions of the path is `map`. This frame's origin is placed at the center of the dock.

- The final position should be (0,0)
- The final angle should be π
- The final angular velocity $r = 0$
- The final linear velocity $u = 0, v = 0$

Approach

After initial experimentation in MATLAB, a gradient descent over an artificial potential field was selected. The core shape of the field is given below.

$$f(x, y) = x^2 + Ay^2$$

$$\nabla f = -\begin{bmatrix} 2x \\ 2Ay \end{bmatrix}$$

The target heading of the boat is defined to be tangent to the path. The overall path is published as an array of vectors of the form:

$$\begin{bmatrix} x_d \\ y_d \\ \theta_d \end{bmatrix} = \begin{bmatrix} x_d \\ y_d \\ \tan^{-1}\left(\frac{-x_d}{-Ay_d}\right) \end{bmatrix}$$

The scaling factor A is used to adjust the relative strength of aligning the boat with the dock and approaching the port in a straight line. Empirically, it was found that a value of A = 5 was well suited to the task.

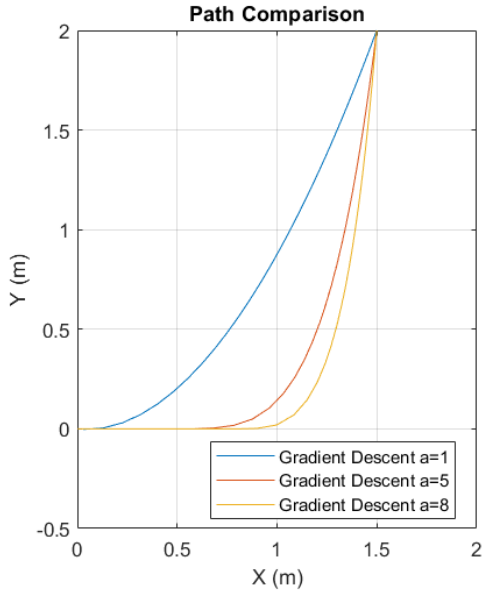

Figure 12: Effect of parameter A on the path

In addition to the basic gradient shape, obstacles can be added to the field to cause avoidance. Two semi-circular virtual obstacles were added to the field to move the boat away from the dock and prevent impossible approaches. The outer regions represent inflation of the outer walls of the pool. Inflation ensures that the proposed path will not allow the edge of the boat to touch the wall. The final potential field is shown below.

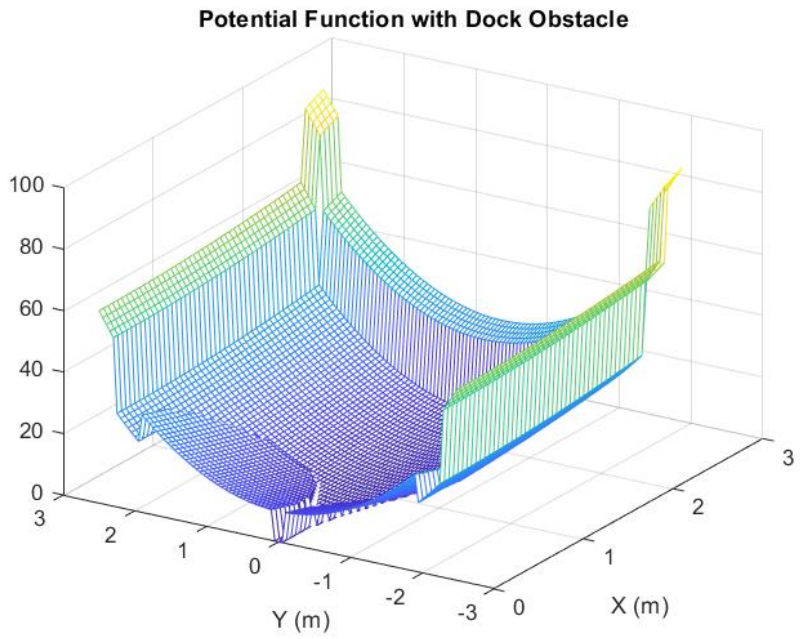

Figure 13: Artificial Potential Field

The potential field does have some issues. Most notably, the gradient descent method is susceptible to getting stuck in local minima. For example, in the example above, a start position of (0, 2.5) would get stuck at the edge of the artificial obstacle instead of avoiding it. A global optimization approach could be more resilient against these issues.

Docking Controllers

Once the camera recognizes the dock using the AR tags on it, navigation can be used to approach the dock. A valid controller needs to smoothly bring the boat to the dock with a velocity near zero and the heading pointing straight into the target. Different methods of guiding the boat into the dock and the different strengths and weaknesses will be examined.

Forward Angle Controller

The first approach performs analysis within the body-fixed frame (`cg_ned`) to move the boat on the shortest path to the target. The forward thrust is proportional to the distance to the target, and the torque is proportional to the angular error between the target and the boat's current heading.

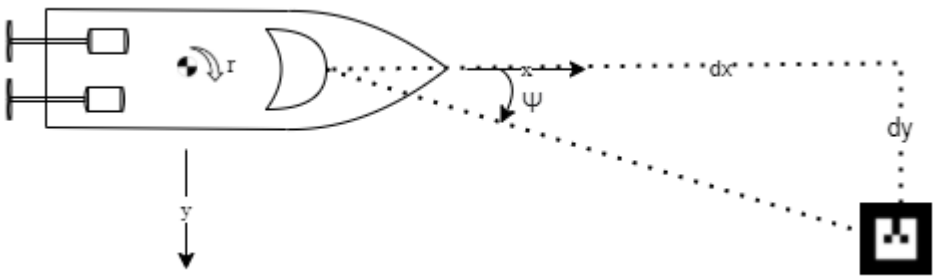

Figure 14: Forward Angle Controller Geometry

$$\tau = \begin{bmatrix} k_p d_x \\ 0 \\ k_\psi \psi \end{bmatrix}$$

An initial issue with this approach is that this does not reliably constrain the final angle parallel with the dock. Any radial approach will be valid. In addition, the system is unstable near the target because the angle increases asymptotically as d_x approaches 0. However, a major advantage of this system is that it does not require any prior knowledge of the dock geometry.

Path Following Controller

The primary method for navigating into the dock involved generating a dynamically acceptable path and following it. For this reason, the problem was decomposed into two compatible problems: path generation and path regulation control. The controller used three primary actions to correct the heading. First, the system minimizes the Euclidian norm of the position vector relative to the path to find the closest point. The system will regulate its heading to match the trajectory (K_1).

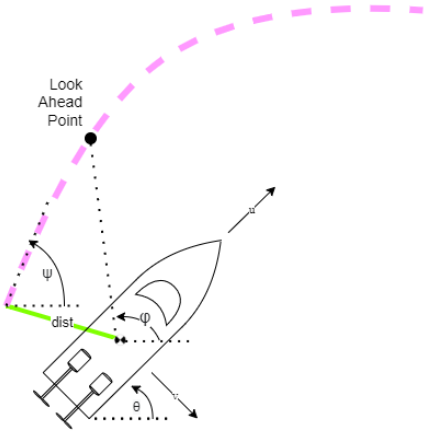

Figure 15: Look ahead path regulation

Once the closest point is found, a lookahead point is used to guide the boat back to the path. Empirically a lookahead point of 0.3m of arc length ahead provided good performance. The forward thrust is proportional to the distance from the path. This means that the boat will

prioritize turning when it is further away from the path (K_2). A third term (K_3) was added to regulate the angular velocity proportional to the curvature of the path. Curvature was used instead of an r(t) term because the path regulation is not adequate. Curvature is time-invariant, so it remains valid regardless of the speed that the boat achieves. The description of the controller force is given below.

$$\tau = \begin{bmatrix} F_x \\ F_y \\ \mathrm{T}_z \end{bmatrix} = \begin{bmatrix} \dfrac{0.75}{2 + dist} \\ 0 \\ K_1(\psi - \theta) + K_2(\phi - \theta) + K_3(\kappa - r) \end{bmatrix}$$

Results and Discussion

The system performed well enough to achieve the primary goals of autonomous conversion and docking control. The following sections will discuss the performance of the system and explore why it did not perform as well as desired.

Dynamics Modelling: Drag Parameter Identification

The completed drag matrix used in the simulations is populated below.

$$\begin{bmatrix} D_1|u| & 0 & 0 \\ 0 & D_2|v| & D_3|v| \\ 0 & D_4|r| & D_5|r| \end{bmatrix} = \begin{bmatrix} 4.17|u| & 0 & 0 \\ 0 & 15|v| & 1|v| \\ 0 & -0.5|r| & 0.12|r| \end{bmatrix}$$

There is significant uncertainty in all these values, although D_1 is relatively well constrained.

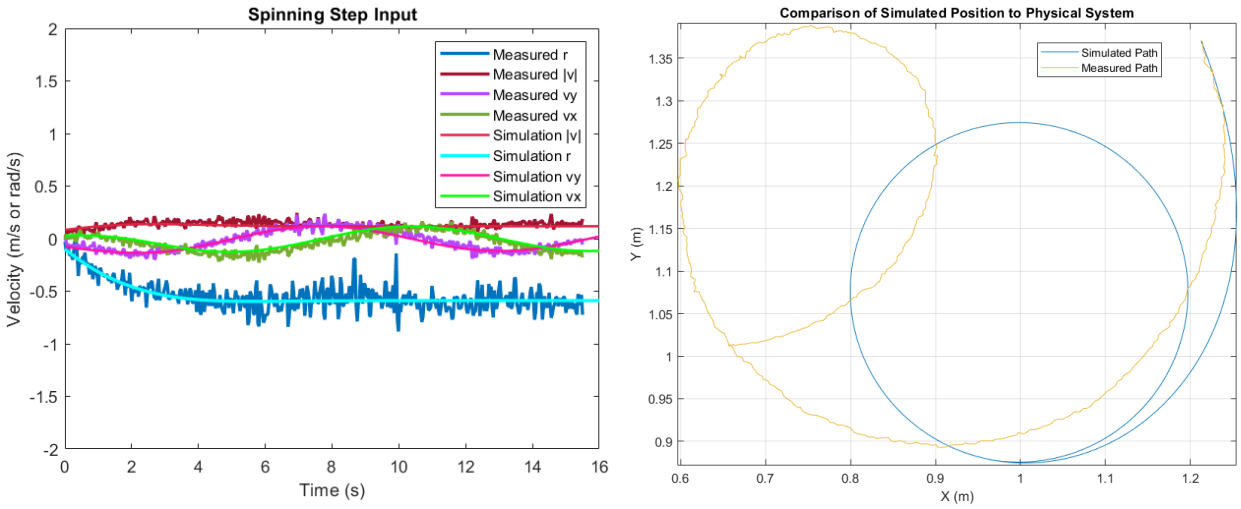

Figure 16: Circular step input response

A straight-line test is given below. The curvature is based on mismatched motors.

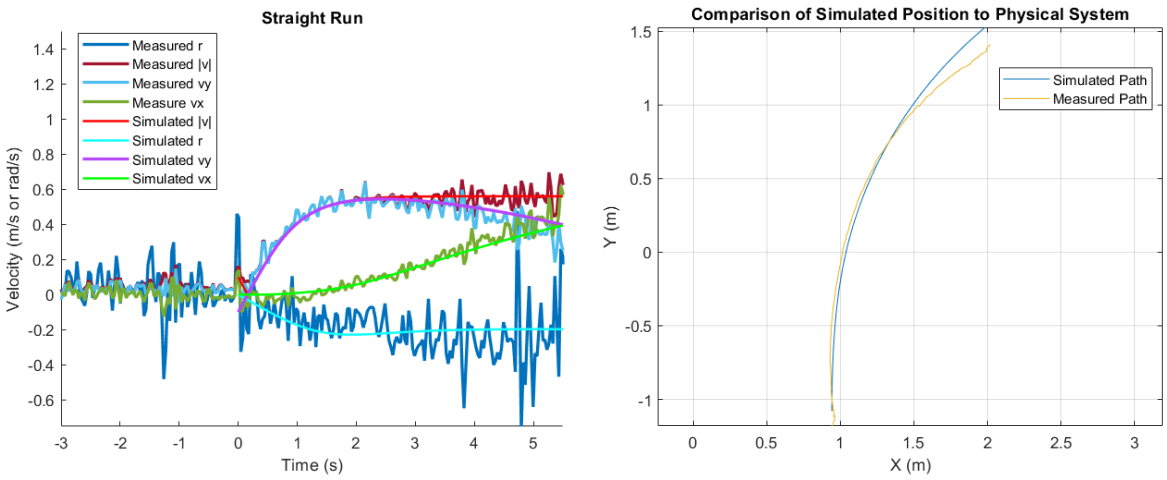

Figure 17: Straight line step input response

The simulations revealed a few interesting features of the boat. Firstly, even if the boat moves with uniform circular motion, the heading of the boat will not point along the tangent line. This dynamic artifact can be intuitively understood by the generation of centripetal forces. Unlike a car that has static friction to generate the normal acceleration, the lateral drag force on the boat is based on the y velocity in the `cg_ned` frame. The boat must be offset from the actual tangent to move a component of the total velocity into the lateral direction. Although this was not considered in path planning, a future version could do so.

Another facet is the consistent lateral motion of the boat. In the simulation, there is a consistent quasi-equilibrium of uniform circular motion. However, this effect is not clearly observed in the water. There are quite a few possibilities for this. It is possible that this could be a simple result of comparing the dynamics at the center of gravity versus a point that is slightly back of it. This seems unlikely, given that the magnitude of this error is not more than 2 cm. It is also possible that this is the result of wind disturbances. Hong et al. included wind and current effects in their simulations to account for this effect [13]. Their outputs look similar. Whenever the thrusters are inactive, the boat drifts at speeds of up to about 0.2 m/s, depending on the wind situation. This dynamic model does not consider aerodynamic effects, including wind. It is quite possible that wind would be able to provide input disturbances to account for the motion observed.

Finally, a third option is a poor characterization of the drag across the hull as the heading changes. Mohamed and Kchaou [15] describe how the drag coefficient across the bow is not simply related to the $D_1\sin(\theta)$ and $D_2\cos(\theta)$. To ensure that the boat spins on the spot, a reverse thrust must be applied as well. It is possible that there is additional coupling between the forces and velocities that are not well understood. This would be the subject of a study unto itself, but it would be interesting to construct a tow tank to evaluate these parameters. A detailed tow tank system would allow the boat to be pulled through the water at any fixed angle attached with a rope. A 3-axis dynamometer would measure the force and moment at the attachment point, ideally with a high enough refresh rate to capture the first-order dynamics of the system.

Path Generation

The gradient descent path generation approach was effective given the scale of this project. The paths were reasonable for the boat to follow and guided it into the dock reliably. In addition to the final gradient descent style, some other approaches were considered. First, some explicit polynomial paths that satisfy the constraints were considered. Any polynomial of degree 3 or higher would be sufficient. However, these are inflexible because such paths cannot have discontinuities or loop over themselves.

It would be interesting to try generating paths that are optimized over different sets of criteria. For example, [16] optimized over the steering angle of the truck. In the AR tag case, an optimization that maximizes the quality of AR tag feedback could provide benefits to the robustness, even if the path itself is slower than the current one. This would involve essentially including the field of view of the camera and the tag locations. This might reduce the chance that the system loses track of its current position.

The primary future work would be to involve full state feedback. Currently, the trajectory tracking is poor, so only position plans are defined. It would be better if entire state trajectories, including angular velocity and linear velocities, could be defined. It might be more interesting to use a model-based approach to generate a full force and torque trajectory to try to regulate around. This might not work, but regular full state feedback could be used because over that two-input space, everything can be controlled. The model could be used in reverse to estimate the current input torque and force, and then this estimate could be compared to the desired trajectory.

State Estimation

Initially, it was the plan to use an out-of-the-box Extended Kalman filter that used both GPS and an IMU to estimate the complete state of the boat. However, the discreet jumps in GPS were too significant and unpredictable to use in a control loop. This was the original motivation switch to using an AR tag array for visual odometry. The AR tags have their own set of challenges. The two most prevalent are range issues and position singularities.

The camera has a limited field of view through which to observe the environment. The first iteration of the system had a Raspberry Pi Cam V2 with a horizontal Field of View of about 60º. This was problematic because the boat would often move into a position where no AR tag would be completely visible. For this reason, a wide-angle 160º FOV camera was purchased to replace it. This improved the performance. Only if the boat is at the edge of the pool and pointing out will there be no tags in the field of view. At a resolution of 1920x1080 pixels, the system can detect tags from approximately 4m. This is sufficient for pool operation. Even with this improvement, the core AR tag library remains susceptible to discreet singularity position.

It is not clear which positions will cause a singularity. In the AR Track Alvar Library, Schweighofer and Pinz discuss how a global optimization solves for the most likely pose based on the visible tags. Local minima can lead to erroneous results. This effect has been observed by other researchers [17] [18] as well. Interestingly, these singularity positions appear to be consistent. Figure 18 shows some of these jumps in action. The most significant jump appears at the same true distance away from the dock (Figure 19). For this situation, the error position

appears to be based on reflective symmetry. The smaller jumps reflect uncertainty in the measurement. It is possible that motion blur impacts the measurements as well.

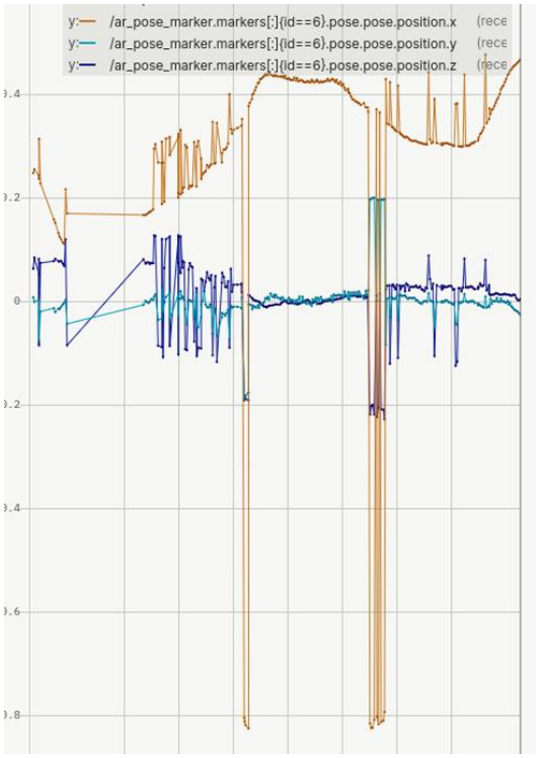

Figure 18: Singularity positions of the dock

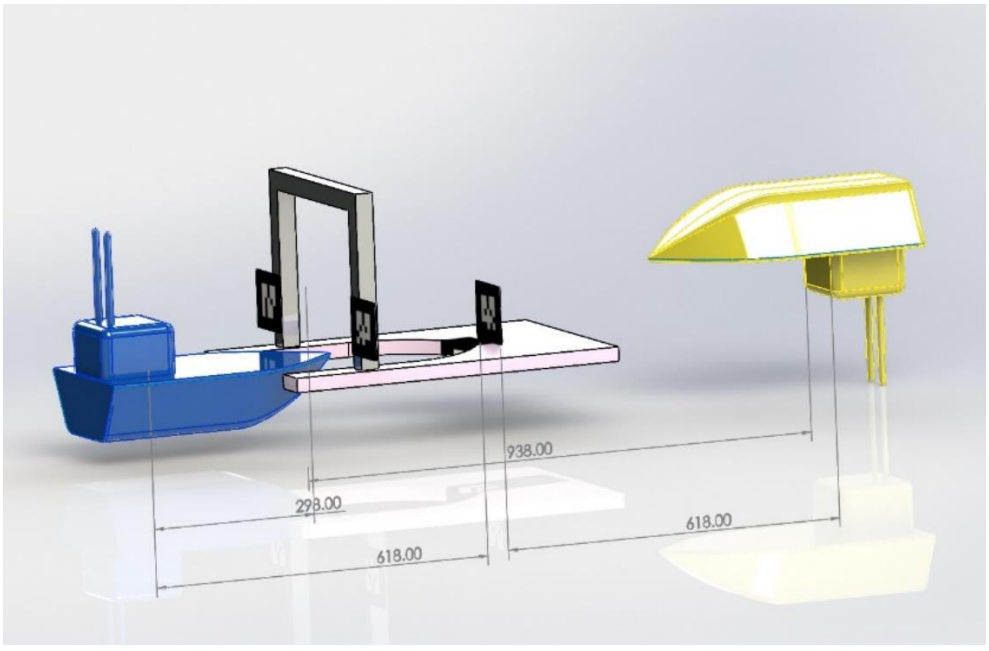

Figure 19: Blue shows true position during singularity, yellow shows the position reported during the singularity. It is a rotation around the rear AR tag.

To reduce the impact of measurement jumps, a Kalman filter fusion of the IMU acceleration and angular velocity was investigated. The root of the issue involves the variance of the sensor measurements. A median filter rejected transient AR tag failures, but overall, the Visual-Inertial odometry fusion suffered from the jumps just as much. Ultimately the EKF led to more problems than it solved. At least when the AR tags jump, the error does not propagate. However, when these values were fed into an EKF, they would perturb the measurements for about 5 seconds which rendered a longer-term impact than the raw data

Although the IMU had some issues with not settling to an average of 0, its noise is normal, and variance could be reasonably estimated for the stochastic process. Correct measurements from the camera are routinely accurate within a centimetre with a standard deviation of about 0.5-3cm, depending on the distance of the tag from the camera. In contrast, singularity situations have no upper bound on the error.

One question is whether the localization error is an effect of motion blur. By increasing the controller gain (K_1), the boat spins more quickly as it follows the path. Motion blur is most commonly caused by the rotation of the boat. For five tests where the boat followed a straight-line path, the ground truth was compared to the AR localization. Figure 20 demonstrates one such path. There are significant issues with large magnitude jumps

Table 3: Path Localization Errors

Controller Gain K_1	Maximum Angular Velocity (rad/s)	Path Error (cm RMS)
2.5	0.20	11.07
5	0.62	39.59
10	1.01	53.0
20	1.19	19.97
100	1.35	31.76

Table 3 supports the earlier theory that localization singularities are position-driven instead of an effect of motion blur. The RMS error in localization is not proportional to the angular velocity. Even with these errors, all these situations are more accurate than the GPS. Even though the errors can be significant, at least they are not accumulated.

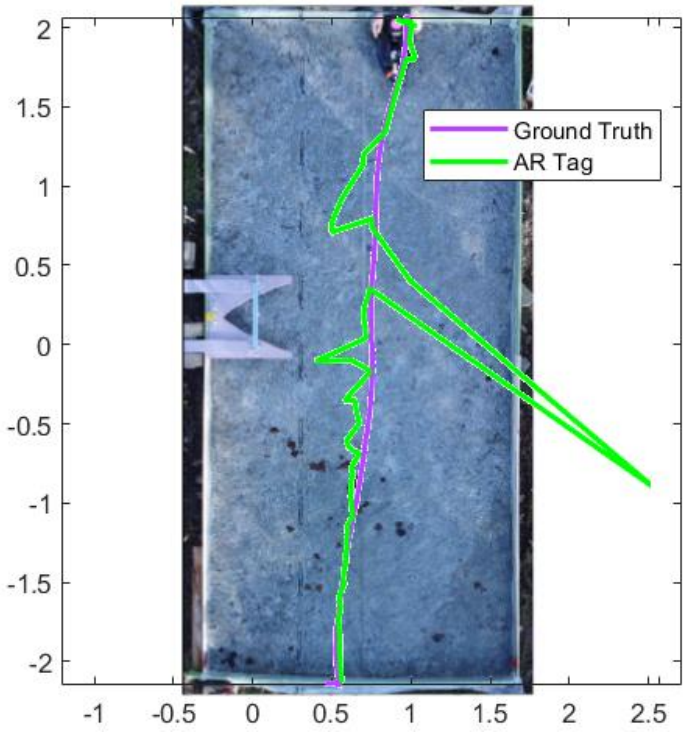

Figure 20: Comparison of ground truth (purple) to perceived position (green) for K_1=5, K_2=2, K_3=0.1

There are several improvements that could be made to this system, both at this scale and in general. The first one would be experimenting with different tag libraries instead of Alvar. A recent survey paper comparing fiducial systems found that the Stable Tag (STag) system was ideal [17]. It had better computational efficiency for large bundles. Another interesting feature is that detection will still work for partially obstructed tags, meaning that it is more robust against partial shadows. This change would likely not require significant changes to the overall structure of the code because of the ROS publisher subscribe architecture.

Further improvement could come from using a velocity sensor. Either a pair of pitot tubes or an ultrasonic velocity sensor on the underside of the hull would help the Kalman filter operate under a much more constrained set of parameters. Other novel methods of velocity estimation using sonar or underwater optical flow could be applied as well.

Controller

The system model was used to experiment with control schemes before they were implemented on the boat. The model was tweaked based on the results found comparing the physical system tests to the simulations. A few sample simulations for different step inputs are shown below.

Controller Simulations

MATLAB was used to test controller configurations before they were implemented on the physical system. Even though the model is not ideally constrained, it should be sufficient to investigate how controllers behave. The three controller gains used for target heading, path correction, and rotation rate regulation were first tuned in MATLAB. However, it should be noted that due to the poor-quality localization on the boat and slow refresh rate, solutions that

worked well in MATLAB did not necessarily translate to real life. For example, the result of a simulation is given below:

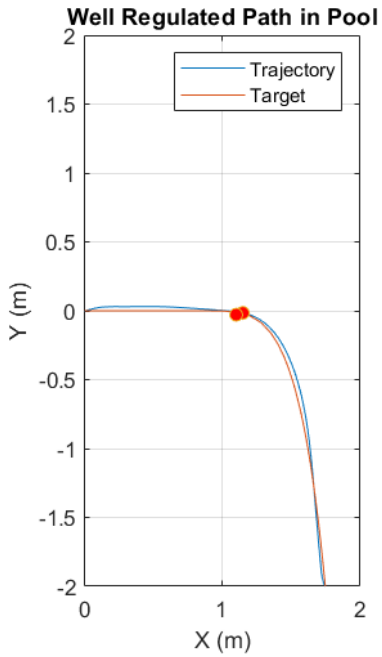

Figure 21: Well tuned MATLAB simulation

The gains used for this simulation are given in Figure 22: Path error propagation sample. See Figure 15: Look ahead path regulation for the definition of each angle.

$$\tau = \begin{bmatrix} F_x \\ F_y \\ \mathrm{T}_z \end{bmatrix} = \begin{bmatrix} \dfrac{0.75}{1 + dist} \\ 0 \\ K_1(\psi - \theta) + K_2(\phi - \theta) + K_3(\kappa - r) \end{bmatrix}, \qquad \begin{matrix} K_1 = 0.5 \\ K_2 = 20 \\ K_3 = 5 \end{matrix}$$

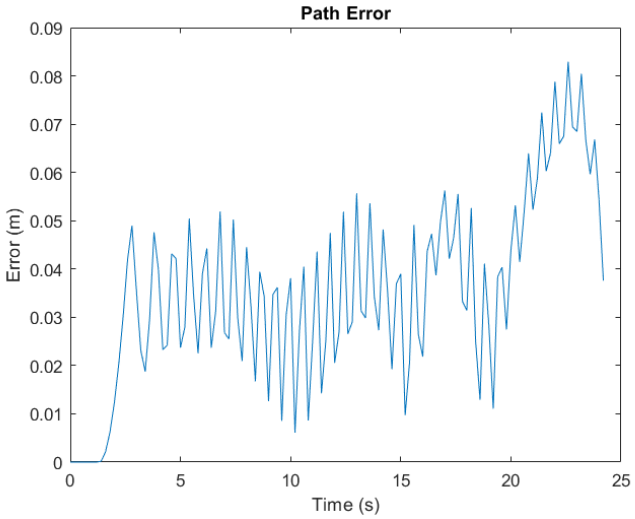

Figure 22: Path error propagation sample for well-tuned controller

Table 4: Comparison of path following for different gains

Scenario	K_1	K_2	K_3	RMS Error (m)	Maximum Error (m)
Well-Tuned	0.5	20	5	0.04	0.083
Too High K_3	0.5	5	5	0.214	0.38
Stable Boat (Figure 20)	5	2	0.1	0.122	0.205
Unstable Boat (Figure 24)	20	2	0.1	0.122	0.241

It is interesting to note that in real life, the results of the docking are more accurate than in the simulation. Additionally, the well-tuned scenario diverges when tested in the pool. However, it is challenging to determine if this means the model is insufficient or if this is a lucky outcome based on unreliable sensor readings. On the water, the refresh rate of the localization system is only 5 Hz. This sampling rate should be better incorporated into the simulation. It is possible that the high resolution of the full state information changes how the system responds. In addition, the error of the localization system is significant, and this will affect the control output in ways that cannot be easily simulated. Figure 23 shows the localization spikes in action. The heading data clearly shows high angular acceleration when these spikes occur. This proves that spikes in the localization have a measurable impact on the controller.

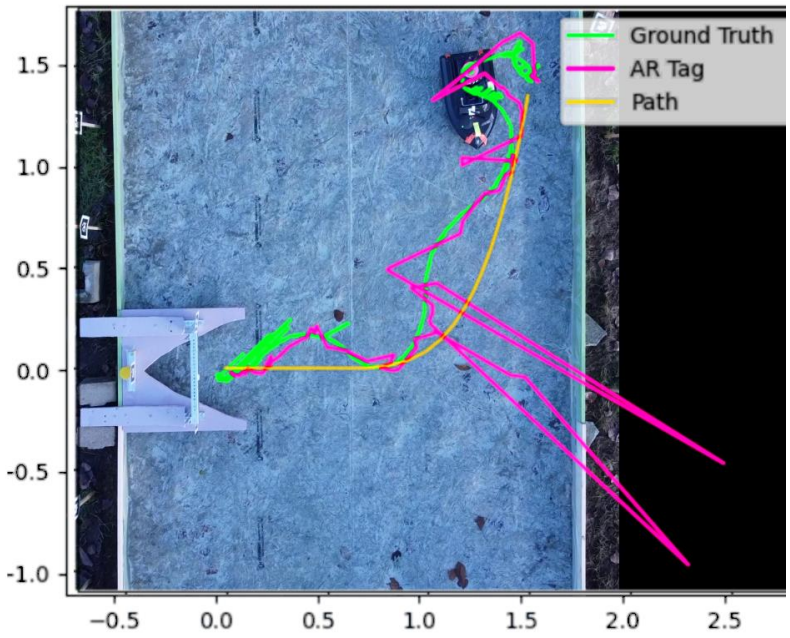

Figure 23: Comparison of ground truth to boat localization with K_1=5, K_2=2, K_3=0.1

With more time, repetition of different controller gains could be carried out to understand the discrepancies between the MATLAB model and the robot.

Natural Frequency Study

To better understand the effect that changing the K_1 value had on the control system, both MATLAB simulations and real testing were performed to characterize the relationship between the frequency of the angular velocity oscillations and K_1. To reduce the complexity of the test,

straight-line paths were used in both cases. The interest in this test came by observing unstable oscillatory behaviour as the gain was increased during earlier tests.

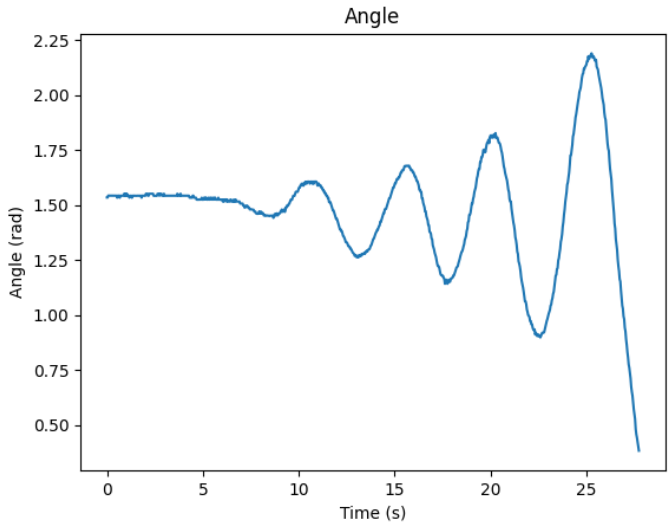

Figure 24: Unstable angle during path tracking

The behaviour looks similar to an unstable second-order system, so it was hypothesized that there might be a similar dependence relationship between the gain, moment of inertia and natural frequency. It should be noted that due to the nonlinear equations of motion, this is not a true analytical result.

$$\omega_n \cong \sqrt{\frac{K_1}{I_{zz}}}$$

$$\log(\omega_n) = \frac{1}{2}\log(K_1) - \frac{1}{2}\log\left(I_{zz}\right)$$

Six different K_1 values were simulated (Table 5). A Fourier transform on the output response was used to determine the peak harmonic frequency within the signal. A linear fit of a log (ω) and $\log(K_1)$ was used to extract a relationship.

Table 5: Simulated oscillation frequencies for the model parameters

$\mathbf{K_1}$	$\boldsymbol{\omega_n}$ (rad /s)
5	0.934
10	1.25
20	1.67
40	2.29
80	3.33
100	3.74

The best linear fit line had coefficients of 0.464 and -1.165

$$\log(\omega_n) = 0.464 \log(K_1) - 1.165$$

The 0.464 is encouraging, given that it is similar to the proposed exponent of 0.5. The y-intercept implies that there are more than just these two values. The value of the y-intercept predicts a value of 0.064 kgm^2 for I_{zz}. This is 70% of the true 0.09 kgm^2. A similar trend was observed experimentally.

Table 6: Observed oscillation frequencies for the boat

K_1	ω_n (rad/s)
2.5	0.452
10	1.09
20	1.36

The experimental linear fit was

$$\log(\omega_n) = 0.544 \log(K_1) - 1.345$$

The exponent was slightly larger than 0.5 instead of slightly smaller. In addition, the measured inertia was 0.147kgm^2 which was larger than the actual value. An experiment varying the inertia would be insightful as well.

Future Work

In addition to the specific forms of improvement listed in the discussion, the project could also move in a different direction. The non-linearities of the boat were particularly hard to constrain. It would be interesting to model the system using a machine learning approach to model generation. A physics-aware machine learning model would be an interesting test of whether a higher-level approach could provide good results. Undoubtedly there is much room for improvement within the controller, and a model-based control would be helpful.

Before embarking on any kind of longer-term analysis of machine learning, the localization system would need to be overhauled to work reliably. In addition, it would be advantageous to use a covered pool so that wind and rain would not factor into the performance as much during initial testing. A covered environment would allow for the overhead camera system to be permanently applied. This data could be processed in real-time and wirelessly transmitted to the boat during operations within a constrained environment.

Conclusions

The study had four primary components: Dynamics Modelling, Path Generation, State Estimation, and Control. Weaving them together was the common task of autonomous docking. Overall, docking was achieved reliably within the outdoor pool facility. However, due to inadequate state estimation, applying advanced control techniques was hampered.

Dynamics Modelling

A three degree of freedom system was implemented in MATLAB and constrained appropriately. Key parameters were either measured directly or estimated by fitting to experimental data. The simulations demonstrate consistency with the physical system, but errors accumulated over long

simulations. Sufficient confidence was achieved in the model to simulate different controller configurations. However, the simulation was not accurate enough to tune controller gains. It is believed that this reflects the slow sensor rates on the boat.

Path Generation

An artificial potential field was used for path generation. Following the gradient allowed for a flexible implementation that could account for obstacles in the environment. Paths that satisfy the requirements for a valid trajectory could be generated efficiently. Separating this component of the operation from the control was advantageous because it allowed the system to define any path for testing. A future investigation would involve using the dynamic constraints of the boat to define full state trajectories that would include velocities.

State Estimation

Although a significant effort was expended to improve this aspect of the system, it is the least effectively developed. The first attempt using an IMU and GPS in an Extended Kalman Filter was ineffective. The precision of GPS was about 300% of the size of the boat, leading to unsatisfactory results. The visual odometry system using AR tags generally provided precise measurements. The error cases were significantly inaccurate. However, visual odometry represented a significant improvement with average RMS errors on the order of 30cm instead of 1m.

The most effective method of measuring the state was recording overhead drone video and extracting the position data from the image. Unfortunately, these results could only be extracted in post-processing and could not be used for control. Variable lighting conditions affected the ease of extraction as well. A stationary camera in the future would likely solve many of these problems, particularly if a roof covered the pool to manage the sunlight and rain.

Control

Although precision control was the original purpose of this project, it could not be fully explored without good state information. However, even with the unreliable state information, the boat could follow an arbitrary path to a reasonable quality. The Forward Angle Controller was reliably able to station keep in front of stationary targets. It could also follow an AR tag through an arbitrary trajectory. The Path Following Controller works on an arbitrary path. However, the system should have a higher-level watchdog that decides when to generate a new path. There were situations when the boat had strayed sufficiently far away from the path that a new one should have been generated and the system restarted. It would be interesting to design controllers to estimate full state feedback instead. In this case, other controller designs, such as a backstepping controller, could be employed.

Acknowledgements

I would like to thank Henglai Wei for support throughout the term, as well as Dr. Shi and Dr. Buckham for providing input.

References

[1] M. Alatise and G. Hancke, "A Review on Challenges of Autonomous Mobile," *IEEE Access,* vol. 8, pp. 39830-39846, 2020.

[2] M. Caccia, M. Bibuli and R. Bono, "Basic navigation, guidance and control of an Unmanned Surface Vehicle," *J. of Autonomous Robotics,* vol. 25, pp. 349-365, 2008.

[3] C. Sonnenburg and C. Woolsey, "Modeling, Identification, and Control of an Unmanned," *Journal of Field Robotics,* vol. 30, no. 3, pp. 371-398, 2013.

[4] ArduPilot Dev Team, "ArduPilot Pixhawk Overview," ArduPilot, [Online]. Available: https://ardupilot.org/copter/docs/common-pixhawk-overview.html. [Accessed 27 10 2021].

[5] ArduPilot Dev Team, "ArduPilot - Companion Computers Overview," ArduPilot, [Online]. Available: https://ardupilot.org/dev/docs/companion-computers.html#companion-computers. [Accessed 27 10 2021].

[6] RFDesigns, "RFD 900x Modem," [Online]. Available: https://store.rfdesign.com.au/rfd-900x-modem/. [Accessed 27 10 2021].

[7] Stanford Artificial Intelligence Laboratory et al., , "Robotic Operating System," 2018.

[8] V. Ermakov, "mavros," 03 03 2018. [Online]. Available: http://wiki.ros.org/mavros. [Accessed 08 09 2021].

[9] Drone Code Foundation, "QGroundControl," 2019. [Online]. Available: http://qgroundcontrol.com/. [Accessed 06 09 2021].

[10] S. Niekum, "ar_track_alvar," 19 07 2016. [Online]. Available: http://wiki.ros.org/ar_track_alvar. [Accessed 01 10 2021].

[11] G. Chiou, "Reducing The Variance Of Intrinsic Camera Calibration Results In The Ros Camera_Calibration Package," University of Texas at San Antonio, San Antonio, 2017.

[12] W. B. Klinger, I. R. Bertaska, K. D. von Ellenrieder and M. R. Dhanak, "Control of an Unmanned Surface Vehicle with Uncertain Displacement and Drag," *IEEE Journal of Oceanic Engineering,* vol. 42, no. 2, pp. 458-476, 2017.

[13] S. M. Hong, K. N. Ha and J.-Y. Kim, "Dynamics Modelling and Motion Simulation of USV/UUV with Linked Cable," *Journal of Marine Science and Engineering,* vol. 8, no. 318, 2020.

[14] R. Seigwart, I. Nourbakhsh and D. Scaramuzza, Intrduction to Autonomous Mobile Robots, Cambridge, Massachusetts: The MIT Press, 2011.

[15] A. Mohamed, H. Kchaou, A. Mohamed and Z. Driss, "Numerical Study of Attack's Angle Effect on Drag Coefficient of AUV Hull Design," *American Journal of Mechanical Engineering,* vol. 5, no. 1, pp. 8-13, 2017.

[16] B. T., A. Brodnik, H. Jonsson, M. Staffanson and I. Soderkvist, "Planning Smooth and Obstacle-Avoiding B-Spline Paths for Autonomous Mining Vehicles," *IEEE Transactions on Automation Science and Engineering,* vol. 7, no. 1, pp. 167-172, 2010.

[17] M. Kalaitzakis, B. Cain, S. Carroll and A. Ambrosi, "Fidual Markers for Pose Estimation," *J. of Intelligent and Robotic Systems,* vol. 101, no. 71, 2021.

[18] M. Agel, M. Marhaban and I. Saripan, "Review of visual odometry: types, approaches, challenges, and applications," *SpringerPlus,* vol. 5, no. 1, pp. 1-26, 2016.

[19] M. Breivik and J.-E. Loberg, "A Virtual Target-Based Underway Docking Procedure for Unmanned Surface Vehicles," in *18th IFAC World Congress*, Milano, Italy, 2011.

[20] G. Schweighofer and A. Pinz, "Robust Pose Estimation from a Planar Target," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 28, no. 12, pp. 2024-2030, 2006.

[21] O. Volden, A. Stahl and T. Fossen, "Vision--based positioning system for auto-docking of unmanned surface vehicles (USVs)," *International Journal of Intelligent Robotics and Applications,* 2021.

Appendix A Pixhawk Autopilot Testing

The first step to verifying autonomy was to tune the built-in PID controller in the ArduRover firmware. Tuning the controller essentially involved using trial and error on a small course until the oscillations around the target trajectory were reduced. This process was challenging because it required access to a large open water area, namely Elk Lake. For this reason, the work was not completed as much as expected. The following images show the progressive improvement of the path following as tuning progressed.

Figure 25: Sample trajectory after initial calibration.

Figure 26: Current tuning after additional PID tuning.

Appendix B: USV Wifi Configuration Information

Network Parameter	**Value(s)**
Network SSID	Ubuntu
Passphrase	miniBoat
Jetson IP Address	10.0.60.1
ROS_MASTER_URI	http://10.0.60.1:11311
Client IP Address	10.0.60.[2-254]
QGroundControl Link	udp://10.0.60.1:14450