Oyster Farming by Autonomous Surface Vehicles
An Implementation of Control and Perception Systems

Daniel Klahn
Massachusetts Institute of Technology
Cambridge, MA, USA
dklahn@mit.edu

Fiona Gillespie
Massachusetts Institute of Technology
Cambridge, MA, USA
fgillesp@mit.edu

Sebastian Monsalvo
Massachusetts Institute of Technology
Cambridge, MA, USA
monsalvo@mit.edu

Annemarie Dapoz
Massachusetts Institute of Technology
Cambridge, MA, USA
adapoz@mit.edu

Andrew Bennett
MIT Sea Grant
Massachusetts Institute of Technology
Cambridge, MA, USA
abennett@mit.edu

Michael S. Triantafyllou
MIT Sea Grant
Massachusetts Institute of Technology
Cambridge, MA, USA
mistetri@mit.edu

***Abstract*—With rising population and limited land, aquaculture is an increasingly important method of meeting worldwide seafood demand. A local New England aquaculture business, Ward Aquafarms, raises oysters in rigid plastic mesh bags that float on the surface of the water. These bags can weigh up to 27kg (60lb) and must be flipped over every 10-14 days to prevent biofouling from accumulating. This process is currently done manually, but there are several problems associated with this: ergonomic difficulty, potential for injury, and the lack of workers willing to do the job. A team at MIT Sea Grant created the Oystermaran in 2021 as a solution to this problem. While this was a promising proof-of-concept, the system was entirely remote-controlled and could only flip baskets in one direction which led to excessive twisting in the connecting ropes.**

This paper builds on the previous work by implementing several low level safety, perception, and control systems to enable autonomous operation. The 2023 Oystermaran is shown in Figure 1. We created an oyster basket localization algorithm with a custom motor controller. We demonstrate their successful integration via a basket-centering routine that uses visual feedback to move the boat to a target point relative to a basket between the hulls.

A new flipping mechanism was designed to enable flipping baskets in both directions to avoid excessive twisting. During the testing of the mechanism, we discovered that it is essential to evenly redistribute the oysters within the bag after flipping in order to keep the basket correctly balanced. Future iterations of the Oystermaran should include a flipping mechanism that addresses re-balancing the bag and extend the autonomy system to traverse throughout the basket array.

This paper demonstrates implementation and integration of key perception and control systems as steps toward full autonomy in oyster farming and aquaculture.

***Index Terms*—aquaculture, ASV, autonomy, low level control, oyster, perception, robotics**

I. Introduction

Oysters have become one of the main products in the global aquafarming sector over the past decades. They are grown in floating mesh baskets on the surface of rivers or bays, and they require a surprising amount of care in order to grow to their prime. A particularly intensive step of the oyster farming process is the need for a worker to manually flip each oyster basket over every 10-14 days. Figure 2 shows the current process of a person in a kayak flipping oyster baskets.

Fig. 1. The 2023 Oystermaran in the lab. Note the oyster basket between the hulls and the two flipping arms.

If the baskets are not periodically turned over, unwanted algae, barnacles, and other biofouling may build up on the underside of the basket. This accumulation can prevent the oysters from getting enough water flow, which in turn can prevent the oysters from accessing enough nutrients to grow [1]. However, periodically flipping the baskets allows the sun to kill the biofouling and restore proper water flow.

While the current manual flipping procedure is quite effective, it is also very repetitive and physically demanding as workers must flip hundreds of oyster bags weighing up to 27kg (60lbs). Thus, our group at the MIT Sea Grant College Program is developing an autonomous surface vehicle (ASV) armed with an oyster basket flipping mechanism in order to automate this arduous task. The ASV, nicknamed the

Fig. 2. Local New England oyster farmers kayak out to their offshore oyster farms and manually flip each oyster basket.

Oystermaran, was originally conceived as a class project and a first prototype was completed in 2021. This first iteration was remote controlled by a human operator and featured a flipping mechanism that was only able to flip the oyster baskets one direction. Field testing of this original prototype also revealed that the oyster baskets are very tightly packed. It was difficult even via remote control to maneuver the ASV through the crowded oyster farm.

With the lessons learned from the initial Oystermaran, we present a completely rebuilt edition of the Oystermaran in this work. The current version of the Oystermaran features a redesigned flipping mechanism and implementations of control and perception subsystems which will enable a more capable autonomy system. Section II describes the mechanical and electrical systems. Section III discusses the new basket flipping mechanism designed to flip baskets in both directions to avoid excessive twisting in the connecting ropes as well as new findings on the basket flipping problem. Section IV details the controls and perception subsystems, and Section V examines the results of evaluating each subsystem.

II. Vehicle Platform

A. Mechanical Setup

As seen in Figure 1, the Oystermaran is a catamaran-style vessel. The two hulls are made of layered styrofoam encased in fiberglass to add rigidity and a smooth finish. The primary motivation behind the hull redesign was to improve maneuverability in crowded oyster farms, so the hulls are coated in a layer of low-friction marine paint and the redesigned bow is shaped to guide the baskets smoothly between the hulls.

The frame was also designed to be modular and easy to disassemble whenever needed. Oystermaran's frame is a 3 foot tall structure made of 80/20 stainless steel bars meant to bridge the two hulls together. It offers the possibility of adjusting the distance between the two hulls while not sacrificing rigidity.

More information on the hull design and maneuverability can be found in *A Prototype USV Design for Maneuvering in a Crowded Oyster Field.*

B. Electronics

The Oystermaran is powered by a 12V 75 AH lead acid battery and incorporates several resettable fuses to protect the electronic speed controllers, thrusters and onboard computers. The 12V thruster power rails run through a power relay which allows the onboard computer to cut power to the thrusters via a software emergency stop.

The electronics are held within several water resistant boxes. Connections between the boxes are provided via waterproof bulkhead connectors which make it easy to assemble, disassemble, and debug the electrical and computational systems.

C. Computation

Our system uses a Raspberry Pi 4 for the main autonomy components and a Pixhawk 4 for communication with the hardware. The Raspberry Pi runs the ROS-based software systems for locating baskets, computing target velocities, and calculating thruster commands. These thruster commands are then sent to the Pixhawk which handles sending the correct PWM signals to the thruster electronic speed controllers. The Pixhawk also provides easy remote control of the thrusters via a DX18 transmitter.

III. Flipping Mechanism

This mechanism is meant to physically flip the oyster baskets one by one as the catamaran moves along the array of baskets. The previous flipping mechanism design involved a prismatic arm with a hook that would latch on to one side of the baskets, contract, and flip. This design had some limitations, one of them being that the boat could only interact with the baskets from one side. The new design was developed with the intention of having control of the baskets from both sides. The newer design (Figure 3) consists of a pair of arms running alongside both hulls facing the baskets. Additionally, a third biaser arm hangs above the baskets in order to guarantee they fall on the correct side. The mechanism is comprised of aluminum and steel framing, and it uses stepper motors, and a stainless steel gear train. Testing this design using lightly weighted baskets was successful. However, when testing with maximally weighted baskets, we found that evenly redistributing the weights inside the basket after flipping is critical. Without this redistribution, the baskets remain unevenly weighted and sink below the reach of the flipping arms.

IV. Autonomy System Design

A. Architecture Overview

The autonomy system enables the Oystermaran to detect a basket, determine the relative position between the basket and ASV, and maneuver itself to center on the basket. As this work is constantly evolving, it is important to create a structure that allows for modularity. A middleware helps to create a system that clearly defines the behavior of its components. This allows flexibility to improve any singular component without worrying about large changes to the entire system.

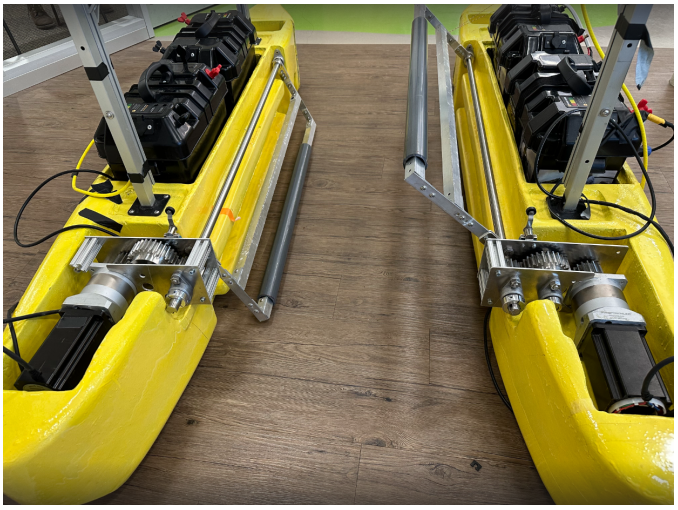

Fig. 3. The two flipping arms of the flipping mechanism. Also shown, the stepper motors and gear boxes.

The Robot Operating System (ROS) is used as the middleware for this work. ROS is an industry standard and provides strong tools for working with cameras. The Mission Oriented Operating Suite with Interval Programming (MOOS-IvP) was also considered for its specific focus on autonomous marine vehicles. However, ROS was chosen as the main middleware for this work due to its better integration with cameras and other sensors [2].

The autonomy system requires perception, planning, and motion control capabilities. These large processes are further divided into modules shown in Figure 4.

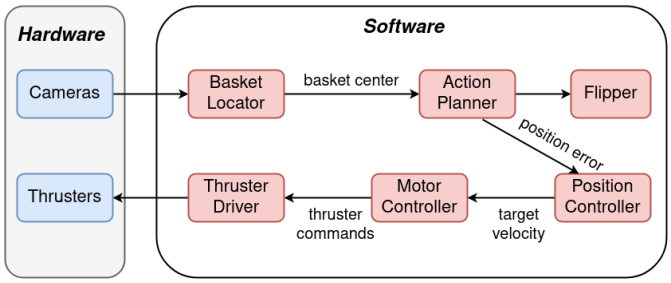

Fig. 4. Overview of software architecture modules.

To handle perception, two EMEET 1080p webcam cameras are used for the sensor input. This camera data is sent to the basket locator algorithm, which finds the basket center if enough of a basket is visible. For planning, the action planner takes in the basket center and determines if the Oystermaran is in position to flip or still needs to readjust. For motion control, the action planner can readjust by sending the basket position error to the position controller to calculate a target velocity. The motor controller then converts this target velocity into thruster commands. The thruster driver sends these commands directly to the thrusters.

B. Perception

An essential component of the perception system is the detection and localization of oyster baskets between the hulls. Although the flipping mechanism is being redesigned, it is apparent that in order to successfully flip a basket the Oystermaran will need to be able to move itself relative to the basket in order to position the basket correctly within the flipping mechanism. Furthermore, the oyster basket array has some freedom to drift which can cause the rows of baskets to be curved rather than perfectly straight. The freeform shape of the array can be seen in Figure 2. Since the array is flexible, GPS-based navigation alone would prove inadequate for robust navigation. Instead, augmenting the GPS and accelerometer data with camera measurements of basket positions allows the Oystermaran to more accurately determine its velocity and progress through the array.

We place red straps on the baskets and use two cameras to determine the basket's position using traditional image processing techniques based on findings from previous work on this problem [3] [4].

1) Camera System: The Oystermaran uses a camera-based system to detect oyster baskets between the hulls. The EMEET 1080P Webcam was chosen for this system. At first, a single camera was mounted above the center of the frame, but this only provided visibility within the hulls of the Oystermaran. This range was too limited, so a second EMEET camera was added. To add extra protection against glare, polarized filters were attached over the cameras.

The two cameras views overlab so that at least one camera can detect a basket continuously throughout the range of 2.95m (9 ft 8 in). Figure 5 shows the full range that the cameras can see. This allows the Oystermaran to see approximately a full basket in front of the noses to the back of the hulls. The cameras are mounted 1m above the waterline, and towards the front of the Oystermaran to allow for earlier detection of baskets. The camera views are shifted towards the front of the boat so that the Oystermaran can see farther in front than behind. This is because in practice the Oystermaran will be moving forward more often than backward.

Each camera takes pictures at a frequency of 20 Hz. Every image is then sent to the basket locator to check if a full basket is in frame. To get two cameras running on the same USB bus, the resolution had to be slightly decreased from 640x480 pixels to 640x360 pixels. This decrease in resolution still allows full basket detection.

2) Basket Locating Algorithm: The Oystermaran uses traditional image processing techniques to achieve accurate and low-latency basket locating. In this section, we present a method for locating baskets by marking them with colored straps, applying filtering in Hue Saturation Value (HSV) color space, and matching the resulting image regions to an expected marker pattern.

Previous work on a camera-based basket location system explored training an object detection model to detect and locate oyster baskets, but found that this approach suffers from a lack of large training sets necessary to achieve robust

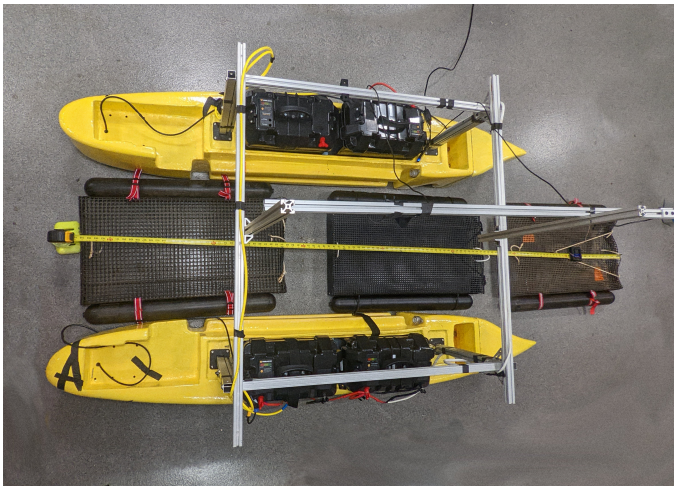

Fig. 5. The camera system can see a range of over 3 baskets at the same time, spanning 9 feet 8 in or 2.95 m

performance [4]. In contrast, traditional image processing techniques provide an accurate and low-latency solution to basket locating.

Each oyster basket is comprised of a rectangular plastic mesh with two floats on either side. The floats are attached via black rubber ties that keep them firmly secured to the basket. Our solution involves replacing these rubber ties with thick red straps as seen in Figure 6. These red straps not only hold the floats to the basket, but also provide an easily distinguishable marker that can be consistently placed in the same location on each basket and will boldly stand out against the water and bio-fouling in a variety of light conditions.

Fig. 6. Oyster basket with original straps (left) and basket with new red straps (right)

Our image processing algorithm performs simple value thresholding in the HSV color space (Figure 7 panel 2) to determine which pixels belong to the straps and which do not. This technique is computationally fast and achieves quite good results in a variety of lighting conditions.

The algorithm then performs a closing and small object removal operation to remove spurious pixels (Figure 7 panel 3) and calculates the centroid of each connected region (Figure 7 panel 4).

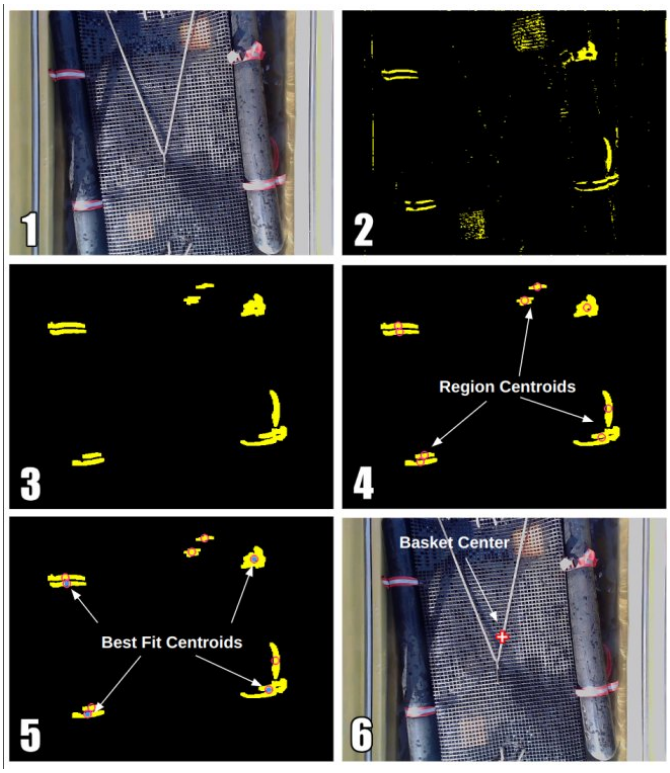

Fig. 7. Intermediate stages of the basket locating algorithm: 1.) The raw image from the camera 2.) The result of HSV filtering. 3.) A cleaned up image after small object removal and gap closing. 4.) Determining the centroids of each disconnected region. The centroids are shown as pink rings. 5.) Determining the set of centroids that most closely match the expected basket shape. These best fit centroids are marked with a blue dot. 6.) Calculate the center of the basket as the average of the best fit centroids. The basket center is marked with a red and white plus.

The final step is to find the set of region centroids that most closely match the expected pattern of basket markers based on the mean squared error between their differences (Figure 7 panel 5). The center of the basket is then found by taking the average of the positions of the best fit centroids (Figure 7 panel 6).

Finding the best fit centroids is equivalent to finding the point correspondences between the region centroids and the expected marker pattern. There are multiple existing algorithms for a similar problem in robotics: point cloud registration. However, many of these algorithms assume the point correspondences are already known and are therefore not useful in this case. A notable exception is the Iterative Closest Point algorithm (ICP) which attempts to determine point correspondences and the transformation between them [5]. We found that in our use case with a small number of points that are relatively close to each other, it was difficult

to filter out outliers and they would skew the result enough to reduce the accuracy beyond what was useful.

We developed an algorithm to determine the point correspondences between the known marker pattern and the region centroids extracted from the camera data. This algorithm takes in the set of measured region centroids and the set of known strap positions and returns a mapping of which measured region centroid most likely corresponds to each known strap position. A high level description of the algorithms steps is given below:

1) Consider the possibility that region centroid n corresponds to known point 0
2) Assuming the initial correspondence from step 1, calculate the expected positions of the other known points
3) For each expected position find the nearest neighbor in the set of measured region centroids
4) Build the rest of the correspondence map by associating each known point with its nearest neighbor from step 3
5) Repeat steps 1-4 for each n and return the correspondence map with the lowest mean squared error between the region centroids and the expected positions from step 2.

Pseudocode for finding the point correspondences is given in Algorithm 1. This algorithm makes the assumption that the basket is not rotated significantly from the expected marker pattern. This is a reasonable assumption when the Oystermaran is within the basket array as the baskets are mechanically constrained by the array and by the Oystermaran's hulls.

Data:
region_centroids: the set of measured region centroids stored as a KDTree
target_points: the set of known strap points

correspondences ← empty list
foreach *r* ∈ *region_centroids* **do**
  shift ← target_points[0] - r
  expected_points ← target_points - shift
  nearest_centroids ← nearest neighbors of expected_points in region_centroids
  error ← mean squared error between expected_points and nearest_centroids
  append (nearest_centroids, error) to correspondences
end
Result: Return the correspondence with the lowest error

Algorithm 1: An algorithm for finding the point correspondences between the measured region centroids and the expected strap points

The Oystermaran ROS system packages this algorithm into a node that subscribes to the camera image topic and publishes to a basket location topic.

C. Planning

Once the system knows the relative basket location, the action planner decides what the Oystermaran should do next based on the difference between the current basket position and the target basket position. The current system has the choice between two actions: move to be more centered on the basket or try to flip the basket. If the current basket position is close enough to the desired position, then the flipping is activated. If the position is too far away, the action planner triggers the motion control stack to align the Oystermaran better with the basket. The proximity threshold was determined experimentally.

Due to the redesign of the flipping mechanism, "flipping" is currently indicated by an LED which lights up when the flipping action should be engaged. The desired basket position, shown in Figure 8, was chosen since it is in the center of the camera range.

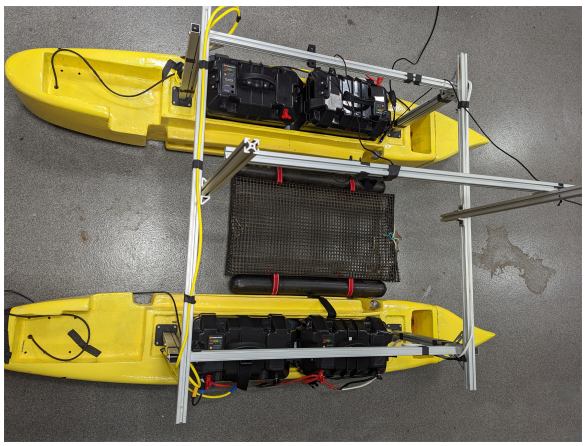

Fig. 8. The desired basket location is towards the front of the hulls

D. Motion Control

1) Position Controller: The position controller takes in the current basket position error and uses feedback control to determine a target velocity to recenter the Oystermaran over the basket. The position controller uses PID control in order to compute the target velocity. The gains were tuned experimentally. The position and motion controllers were tested together to ensure that the Oystermaran would move smoothly and safely when centering itself. The target velocity computation with tuned gains is shown in Equation 1.

$$v_{\text{TARGET}}(t) = 2e(t) + 0.1\int_0^t e(\tau)\,d\tau + 4\frac{d}{dt}e(t) \qquad (1)$$

The Oystermaran ROS system packages the position controller into a node that subscribes to the `/position_error` topic and publishes the target velocity as a Twist message to `/cmd_vel`.

2) Motor Controller: The Oystermaran has four thrusters: two Blue Robotics T200 thrusters in the front and two Blue Robotics T100 thrusters in the back. Each hull has a thruster

in the stern for forward/backward movement and a cross-tunnel thruster near the bow which is rotated 90 degrees and enables side-to-side movement. The Oystermaran implements a dynamics-based motor controller to convert target velocities into motor speeds. The motor controller formulates this problem as the matrix equation shown in Equation 2

$$Bu = Cv^2 \tag{2}$$

For the Oystermaran, u is a 4x1 vector representing the motor speed outputs, v is a 3x1 vector representing the linear and angular velocities $[v_x, v_y, \omega]$, C is the 3x3 drag matrix, and B is a 3x4 matrix that converts the 4x1 u vector into a 3x1 force/torque vector $[F_x, F_y, \tau]$.

The motor controller uses an optimization (using the magnitude of u as the cost function) to determine the smallest thruster output that will achieve the target velocity.

The result is that the control output u is proportional to v^2 with the B matrix encoding the relative contributions of the 4 thrusters to the resulting velocity. In theory the B and C matrices can be calculated analytically based on the physical configuration of the thrusters and the shape of the hulls, but in practice they require some experimental fine-tuning.

The hardware constraints also required some additional processing to the control outputs. The Blue Robotics T200 thrusters are unreliable and intermittent at low speeds due to a dead-zone from 0-10% commanded thrust [6]. To improve the motion controller performance, the control output is clipped such that any control output between 2% and 13% is instead set to 13%.

The Oystermaran ROS system packages this motor controller into a node that subscribes to the */cmd_vel* topic and publishes thruster speeds to the mavros node which relays them to the thruster control electronics. As a safety feature, the motor controller node includes a watchdog timer that halts the thrusters if nothing is published to */cmd_vel* within a given time period.

V. Results and Evaluation

A. Flipping Mechanism Testing

To test the flipping mechanism, we affixed a basket in our indoor test tank and weighted it to accurately resemble a full oyster bag. The basket was filled with 14kg (30lbs) of water containers and 14kg (30lbs) of lead weights. This mixture was chosen to closely resemble the dynamics of semi-buoyant oysters.

While testing the mechanics of the flipping mechanism, we discovered a new factor in the basket dynamics that we had not considered: the uneven distribution of weight after a flip. When the basket turns over, the contents shift to one side. In maximally loaded baskets, this causes the basket to sink dramatically below the reach of the flipping arms causing the flip to fail and requires human intervention to re-balance the bag.

We found that shaking the basket by hand was an effective method of redistributing the weight, but the flipping mechanism was not capable of replicating this motion. A new flipping mechanism will need to be designed to address the problem of uneven weight distribution.

B. Basket Locator

1) Accuracy: The basket locating algorithm was successful on 92.6% (175/189) of the test images. The test set was comprised of 189 images and included significant numbers of images from a variety of conditions. The test images included a mixture of images where the basket straps were all in frame, only some were in frame, straps from multiple baskets were in frame, and none were in frame. The test images were also taken in direct sunlight, full shade, and uneven shade. Figure 9 shows some example images from the test dataset. The test images were hand-labeled with the location of the basket center if all straps were in frame, or with `None` if all straps were not in frame. The basket location was considered to have succeeded if either the estimated center position was within a 20 pixel radius of the labeled center position if it existed, or if it correctly excluded images in which there was not a basket with all four straps in frame. Figure 10 shows a more detailed breakdown of performance across different subsets of the test images.

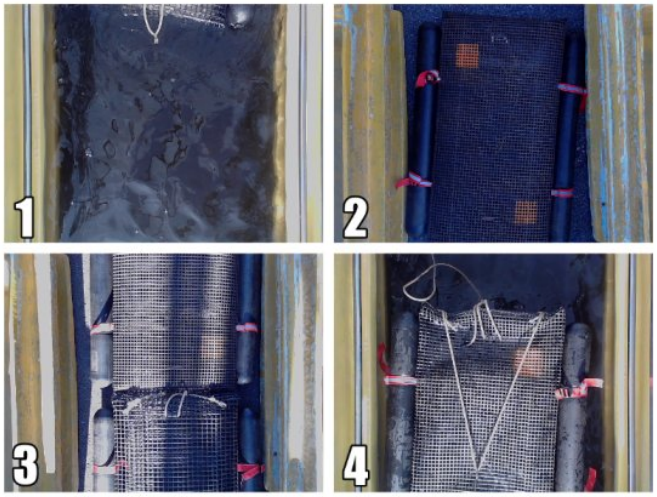

Fig. 9. A few example images from the test data set: 1.) direct sunlight, basket not in frame 2.) fully shaded, basket in frame 3.) direct sunlight, basket partially in frame 4.) direct sunlight, basket partially in frame

2) Execution Speed: The execution speed of our implementation was tested on a Raspberry Pi 4 Model B as this is what will be used for the main computer on the Oystermaran. Across all test images, the basket center was found in 49 milliseconds on average with a maximum time of 67 milliseconds. This means that on average the Oystermaran can generate 20.4 basket position updates per second to feed into the behavior control system.

C. Motion Control

The performance of the motor controller was evaluated qualitatively, as small errors in speed or direction can be compensated for in the future based on sensor feedback. The

		Lighting Condition			
		Fully Shaded	*Dappled Sunlight*	*Direct Sunlight*	***Total***
Basket Configuration	*Fully in Frame*	88% (15/17)	90% (9/10)	85% (41/48)	87% (65/75)
	Partially in Frame	90% (9/10)	100% (11/11)	96% (72/75)	96% (92/96)
	Not in Frame	100% (3/3)	100% (3/3)	100% (12/12)	100% (18/18)
	Total	90% (27/30)	96% (23/24)	93% (125/135)	**92.6% (175/189)**

Fig. 10. A breakdown of the performance of the basket locator algorithm across different lighting conditions and basket configurations. Running the test bench on our image set indicates a high degree of accuracy across all subsets of the test images.

motor controller was able to successfully produce translation and rotation Notably, we found that given a sideways target velocity, the Oystermaran is able to translate side-to-side in a straight line despite not having dedicated lateral thrusters. Actuating the side thrusters on their own produces sideways motion, but also creates a torque that spins the boat. The motor controller is able to successfully counteract the torque from the side thrusters with torque from the forward/backward thrusters to produce exclusively sideways motion.

D. Integration

The integration of the system was tested by evaluating if the Oystermaran could identify a basket's location, determine what at velocity to move to get closer to the basket, and successfully send thruster outputs to achieve the target velocity. To test the control system, the Oystermaran attempted to center itself on a stationary basket in the MIT Sea Grant Float Tank. A basket was tied to the sides of the tank with rope to roughly constrain the position, as seen in Figure 11.

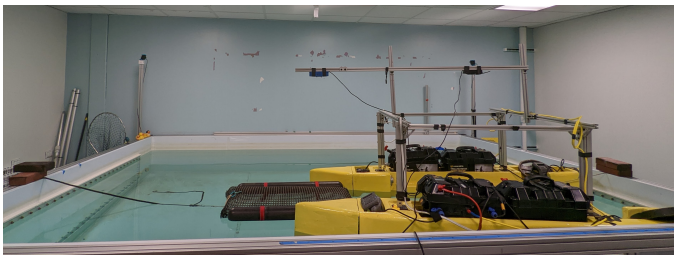

Fig. 11. In the MIT Sea Grant Float Tank, the Oystermaran starts about 1m away from a stationary basket and will attempt to center over the basket during an integration test.

The basket's flexibility in one direction and rope attachments in another direction mimic the constraints of the oyster farm arrays. The relative position of the basket with respect to the ASV was recorded during the basket centering process as the basket position error. The oyster basket was detected as starting about 0.8m away from the target point in the center of the hulls, in view of the camera system. After some manual tuning of PID gains, Figure 12 shows the results of the integration test. The purple line displays the error: 0.25m of overshoot with small oscillations around the target point. This was an encouraging result and validated the integration of the perception and motion control systems.

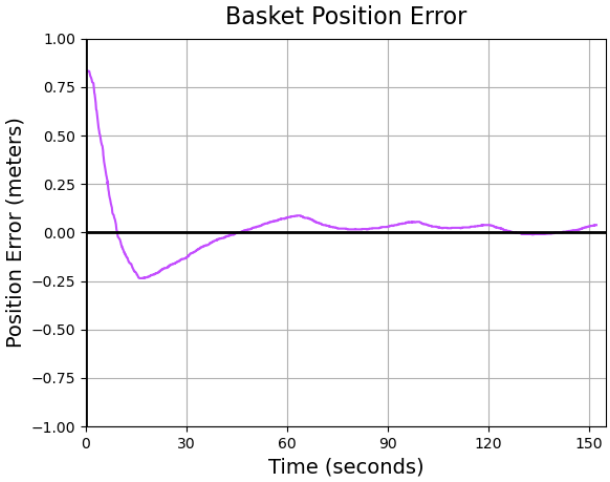

Fig. 12. A plot of the position error during an integration test of the perception and controls systems. The Oystermaran starts roughly 0.8m away from the stationary oyster basket and converges to a target position relative to the basket.

While this test occurred under controlled conditions, it was an exciting first step in verifying that the fundamental perception and motion control systems are working together properly. Now that the Oystermaran can successfully localize itself relative to a basket, it will be ready to accurately measure its progress through the array and position baskets properly within the flipping mechanism.

E. Media

To learn more about the Oystermaran, please visit our website. Photographs and videos of our designing and testing processes can be found at: seagrant.mit.edu/Oystermaran.

VI. Future Work

While this work details a promising new development in Oystermaran, we are looking forward to expanding the capabilities in the next stage of research.

A. Flipping Mechanism

As previously discussed in Section VI-A, tests of the flipping mechanism revealed a need to shake the baskets to redistribute the weight. This prevents the baskets from sinking beyond the reach of the flipping arms. Either this feature will be added or another method of dealing with uneven weight distribution will be developed. This could entail designing and building an entire new flipping mechanism.

B. Basket Locator

1) Camera Calibration: In the current system, the red straps are detected by performing value thresholding in HSV color space. Right now these threshold values are hand-picked and are dependent on lighting conditions. This means that the performance of the basket locating system can vary significantly based on lighting. In the future, we plan on adding some calibration spots on the hull in view of the cameras in order to dynamically adjust the threshold values.

2) Multi-basket Detection: One major limitation of the basket locating algorithm is that it makes the assumption that there is only one basket in frame at a time. This will not be true in the full array and the algorithm will need to be adapted to accommodate multiple baskets.

Multibasket detection will also be key for future higher level behaviors and state estimation. The rows of baskets in the array have a significant amount of freedom to curve and will typically not be in straight lines (see Figure 2). Therefore when maneuvering along a row of baskets, rather than representing the position in a Cartesian coordinate system, it will be more natural to represent position as progress through the array (i.e. as the number of baskets passed). Tracking multiple baskets through the camera view will be an essential part of precisely determining the Oystermaran's position in the array.

C. Motion Control

1) Hardware Performance: The thruster commands were clipped to work around the 0-10% dead-zone, but more work can be done on further characterization. A potential improvement would be to PWM at higher thrust commands to effectively reach thrust levels within the dead-zone.

2) Feedback Control: The Oystermaran does not currently have any way to measure its velocity in order to provide feedback to the motor controller. Next steps for the motion control subsystem will include integrating the visual perception system with the Pixhawk's sensor suite in order to implement a feedback-based motor controller.

3) Simulation: As we work to improve the Oystermaran control systems it will be useful to have a representative simulation to enable faster iteration and testing. We plan on using the Gazebo simulation environment as it is easy to integrate with ROS and there are existing plugins for modeling aquatic vehicle dynamics. Additionally there exist software tools for running a hardware-in-the-loop (HITL) Gazebo simulation with the Pixhawk.

D. Autonomy System

For the autonomy system, we plan to focus next on traversing through the basket array. A huge component of this is the capability to localize the relative position of the Oystermaran in the array. We plan to look into SLAM techniques and augment our camera input with the Pixhawk's local and global position estimates to better determine the relative positioning. Another aspect of maneuvering through the crowded array is the ability to handle when the Oystermaran gets stuck on baskets. This entails creating a behavior system that can use perception to detect when the ASV is stuck and can then perform a sequence of tasks to maneuver itself out of being stuck.

VII. Conclusion

The overall Oystermaran system is able to successfully locate and follow an oyster basket. This was an important step for the project in testing the basket location algorithm and proving the integration of the whole ROS system. This framework can be extended to the needs of the new flipping mechanism to enable autonomous flipping for oyster baskets. This work is just one more step in the larger process of creating an autonomous surface vehicle to assist in oyster farming and aquaculture.

VIII. Acknowledgments

We would like to acknowledge the many people who have made this work possible. Thank you to Dan Ward and Ward Aquafarms for allowing us to use his farm and answering any questions about the logistics and current state of oyster farming. Thank you to the contributors of the previous iteration of the Oystermaran [7] for their work in starting this project. Thank you to our current undergraduates who have assisted with this poject: Ashley Margetts, Toya Takahashi, and Sreeja Thipireddy. And thank you to Andrew Bennett who has provided immeasurable guidance since the conception of the Oystermaran. We would also like to thank the entire MIT Sea Grant for their support in answering questions, informing design decisions, and helping in the development of this research.

References

[1] E. Kenworthy, "UGA part of regional collaboration studying Oyster Farming," Oct 2018. [Online]. Available: https://gacoast.uga.edu/uga-part-regional-collaboration-studying-oyster-farming

[2] "ROS: Why ROS? — ros.org," https://www.ros.org/blog/why-ros/, [Accessed 22-Mar-2023].

[3] M. C. Tung, "Oystamaran: An implementation of autonomy in surface vehicles for oyster farming," Master's thesis, Massachusetts Institute of Technology, June 2021. [Online]. Available: https://dspace.mit.edu/handle/1721.1/139084

[4] J. Zhang, "Perception and motion planning for autonomous surface vehicles in aquaculture," Master's thesis, Massachusetts Institute of Technology, May 2022. [Online]. Available: https://dspace.mit.edu/handle/1721.1/144936

[5] T. S. H. K. S. Arun and S. D. Blostein, "Least-squares fitting of two 3-d point sets," pp. 698–700, Sept 1987.

[6] B. Robotics, "T200 thruster: Performance charts," Apr 2023. [Online]. Available: https://bluerobotics.com/store/thrusters/t100-t200-thrusters/t200-thruster-r2-rp/

[7] M. Kornberg, A. Patton, M. Sullivan, A. Badillo, A. Kriezis, A. Lastra, and H. Turner, "Autonomous vehicle for oyster aquaculture," in *OCEANS 2021: San Diego – Porto*, 2021. [Online]. Available: https://ieeexplore.ieee.org/document/9705669