ECE 445

SENIOR DESIGN LABORATORY

FINAL REPORT

A Morphable Bionic Robotic Fish with Dual-Mode Propulsion Enabled by a Transformable Caudal Mechanism

---

Team #39

KE, XUANYU (xuanyuk2)
WANG, LIBIN (libin2)
ZHANG, BOWEN (bz20)
ZHENG, KAIJUN (kaijunz2)

TA: Shao, Xihe

May 13, 2026

Abstract

This project presents the design and implementation of a transformable underwater robotic fish capable of multi-mode propulsion. The platform integrates an innovative caudal-fin-to-propeller morphing mechanism designed to leverage the complementary advantages of bio-inspired flapping and propeller-based propulsion. The control architecture is built around a Raspberry Pi 5 master module running the ROS 2 (Jazzy) environment for high-level motion planning and mode-switching logic , interfaced with a Cortex-M3 slave MCU for real-time hardware actuation and MPU6050-based inertial sensing.

Experimental validation demonstrates that the control software, sensor telemetry, and mechanical morphing configuration function as intended under dry bench conditions, with the propeller rotating properly in the air. However, during submerged testing, the original brushless DC motor (BLDC) failed to rotate normally due to the complex underwater environment and high hydrodynamic resistance, which exceeded the motor's available torque. In response to these experimental findings, the project identifies the need for hardware iteration as the primary path forward. Future work will focus on replacing the propulsion unit with a motor characterized by higher torque density or integrating a mechanical gear reduction system to overcome fluid loads, thereby ensuring reliable dual-mode propulsion in aquatic environments.

Keywords: Underwater Robotic Fish; Morphing Mechanism; Multi-mode Propulsion; Motor Torque Optimization; Hardware Iteration

Contents

1 Introduction

1.1 Problem Statement

Underwater exploration and marine monitoring demand robotic platforms that can navigate complex environments with both high maneuverability and efficient long-distance transit. Traditional underwater vehicles typically rely on either bio-inspired flapping-foil propulsion or conventional propeller-driven systems. While bio-inspired methods offer superior low-speed maneuvering and stealth, they often lack the peak thrust required for rapid acceleration. Conversely, propeller-driven systems excel in high-speed cruising but are less effective in mimicking the flexible, organic locomotion needed for sensitive biological observation.

The primary challenge lies in the integration of these two distinct propulsion modes into a single, compact robotic frame. Directly combining separate mechanisms often leads to excessive weight, increased mechanical complexity, and reduced structural integrity. Therefore, there is a significant need for a transformable robotic platform capable of seamlessly switching between these two modes.

1.2 Solution Overview & Visual Aid

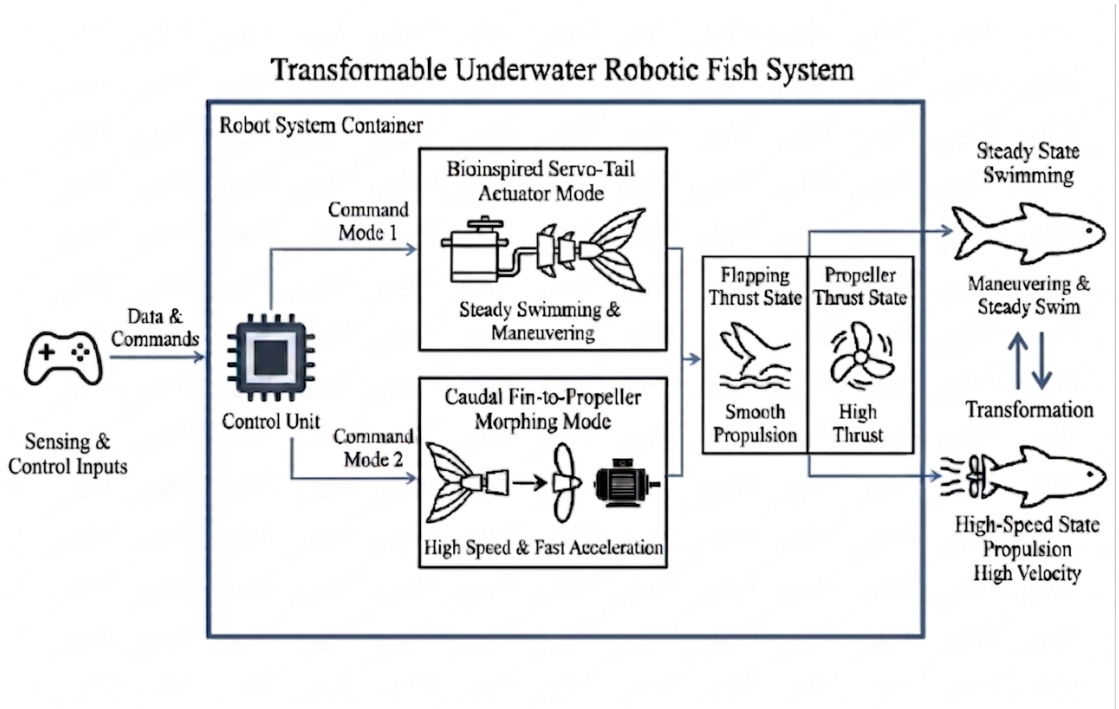


Figure 1: Visual Aid

In this project, we design and implement a transformable multi-modal robotic fish. The core innovation is a dual-purpose morphing mechanism that functions as a flexible caudal fin for oscillatory swimming and can be reconfigured into a rigid propeller for high-speed

transit. To manage the complexity of this transformation and ensure precise control, we utilize a hierarchical control architecture. A Raspberry Pi 5 serves as the high-level master node running the Robot Operating System 2 (ROS 2), while a Cortex-M3 microcontroller manages real-time motor signals and sensor data fusion from an MPU6050 IMU.

1.2.1 Block Diagram

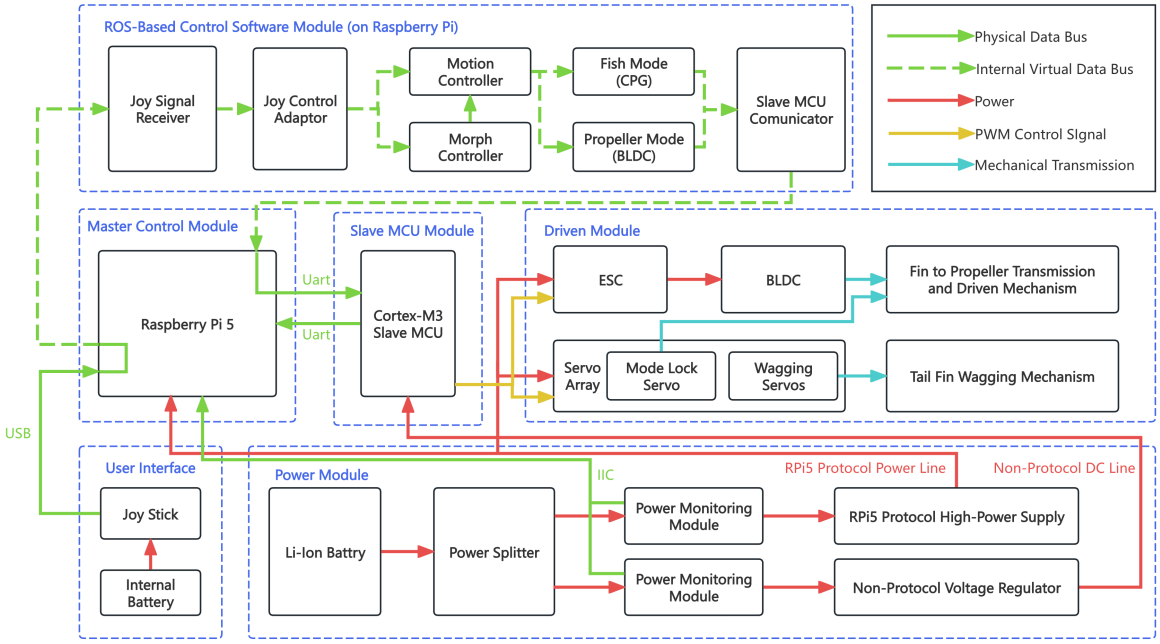


Figure 2: Block Diagram

1.3 Functionality

1.3.1 High-level Requirements List

The transformable underwater robotic fish shall meet the following high-level requirements:

1. The bio-inspired fish must be capable of stable underwater movement in an underwater environment, sustaining controlled forward swimming and directional turns for no less than 30 seconds. Additionally, the bio-inspired fish must be capable of both bio-inspired tail propulsion and propeller propulsion modes, capable of completing a mode switch underwater within 20 seconds while maintaining stability.
2. In bio-inspired mode, the bio-inspired fish should exhibit greater maneuverability. Under identical test conditions, its turning radius should be at least 15% smaller than that of the propeller model.
3. In propeller mode, this bio-inspired fish should demonstrate enhanced high-speed

propulsion capabilities. Under identical test conditions, its maximum forward swimming speed should be at least 15% higher than in bio-inspired mode.

1.3.2 Project Scope and Current Status

The scope of this report covers the mechanical design, hardware integration, and software development of the transformable robotic fish. While the prototype successfully demonstrated the intended morphing logic and propeller rotation in air, underwater testing revealed critical limitations in the current brushless DC motor's torque density when subjected to hydrodynamic resistance. Consequently, this report documents the design successes and provides a detailed analysis of the experimental failures, laying the groundwork for future hardware iterations focusing on high-torque underwater actuation.

2 Design

2.1 Design Procedure

Discussing the design layout at the most general level requires analyzing the tactical engineering trade-offs made for each foundational functional block, with a specific focus on structural configuration retention under cyclic aquatic loading. Below we evaluate the proposed functional modules against alternative design directions and introduce the core governing formulations.

2.1.1 Block-Level Design Decisions

1. **Master Control Module:**
    - *Alternatives:* Utilizing a low-power standalone MCU (e.g., STM32 series) or an AI-centric single-board computer (e.g., NVIDIA Jetson Nano).
    - *Chosen Approach:* A Raspberry Pi 5 operating within the Robot Operating System 2 (ROS 2 Jazzy) framework.
    - *Justification:* A standard MCU lacks the high-level computational throughput required for complex Central Pattern Generator (CPG) gait synthesis and multi-variable state machine logic. Conversely, the Jetson Nano imposes excessive parasitic power draw. The Raspberry Pi 5 provides an optimal equilibrium between computing performance and power efficiency, while ROS 2 offers decentralized modular nodes that significantly streamline the multi-stage state machine controlling the morphing and locking sequence.
2. **Slave MCU Module:**
    - *Alternatives:* Driving the actuators directly using software-generated Pulse Width Modulation (PWM) signals from the Raspberry Pi's native GPIO pins.
    - *Chosen Approach:* An isolated Cortex-M3 microcontroller functioning as a real-time coprocessor.
    - *Justification:* Because the Raspberry Pi runs a non-real-time Linux operating system, software-generated PWM signals exhibit severe timing jitter under high CPU loads. Delegating hardware orchestration to a dedicated Cortex-M3 micro-architecture guarantees microsecond-level hardware timer interrupts. This precise timing is crucial for coordinating the travel of the locking servo with the gait execution of the servos, preventing premature tail flapping from tearing the unlatched joints.
3. **ROS-Based Control Software Module (on Raspberry Pi):**
    - *Alternatives:* Implementing a monolithic polling loop on the Raspberry Pi, or mapping the joystick command directly to low-level PWM outputs without an intermediate software abstraction.

- *Chosen Approach:* A ROS 2 node-based software architecture running on the Raspberry Pi. The official `/joy` node receives the gamepad input, the `joystick_adapter` node maps raw joystick data into motion and morphing commands, `motion_controller` generates locomotion commands for both drive modes, `morph_controller` coordinates the mode transition sequence, and `pwm_actuator_driver` converts abstract actuator commands into the serial protocol of the PWM actuator board.
- *Justification:* The ROS-based structure separates manual input, motion generation, morphing logic, and hardware abstraction into independent nodes. In fish mode, the motion controller uses a CPG algorithm to command the two tail servos for bio-inspired flapping. In propeller mode, it commands the BLDC motor for rotational propulsion. The morphing controller uses a finite state machine to manage both fin-to-propeller and propeller-to-fin transformations. During the morphing process, it publishes a pause state so that normal motion output is disabled until the mechanical transition is completed.

4. **Driven Propulsion Module:**
   - *Alternatives:* Implementing two structurally independent propulsion components, such as attaching external dual-propellers on the lateral sides of the fish body.
   - *Chosen Approach:* A transformable caudal propulsion assembly where the caudal fin physically reconfigures into a rigid propeller blade. The fish-like flapping mode is driven by two tail servos through CPG-based oscillatory commands, while the propeller mode and morphing motion are driven through the BLDC motor and the integrated transmission chain. A tail-mounted locking servo is treated as an internal retention subsystem of this driven module rather than as an independent module.
   - *Justification:* Integrating two completely separate mechanisms scales up the overall spatial dimensions, dry mass, and hydrodynamic drag coefficient of the platform, violating the constraint for a compact underwater robot. Leveraging a custom bevel-gear drivetrain enables structural dual-use, maximizing spatial resource allocation and minimizing fluidic frontal profile area during fast transit. The locking servo is positioned in the tail, and its servo horn is rigidly connected to a locking latch arm. In the engaged state, the latch constrains the BLDC motor output shaft to prevent uncontrolled rolling in a non-driven posture, while also holding the morphing mechanism against unintended deformation under uncertain external loading conditions.

5. **Power Module:**
   - *Alternatives:* Utilizing a single standardized regulation rail for all electronic components, or deploying two physically isolated, independent battery packs for the controller and the actuators respectively.
   - *Chosen Approach:* A central Li-Ion battery pack combined with a custom Power

Splitter board, bifurcated into an RPi5 Protocol High-Power supply domain (via USB Type-C) and a Non-Protocol DC voltage regulation domain.

- *Justification:* Adding dual battery packs severely violates the strict volumetric, dry mass, and buoyancy boundaries of the sealed core compartment. However, a shared single-rail regulator would allow the severe transient voltage spikes and reverse electromotive forces (EMF) generated by the propulsion motors under fluid load to feed back into the master rail, causing instant brownouts on the Raspberry Pi 5. The chosen decoupled topology utilizes the negotiated RPi5 power protocol to secure stable high-current delivery for complex ROS node calculations while completely shielding the logic gates from motor noise and preserving stable actuator-command communication.

2.1.2 Major Design Equations and Kinematic Structural Constraints

To investigate the mechanical feasibility and boundary conditions of the custom morphing mechanism underwater, a first-order non-linear hydrodynamic resistance tool was formulated.

When the robotic fish executes periodic bio-inspired swimming, the local Reynolds number dictates a pressure-dominant drag profile normal to the flapping surface. The differential hydrodynamic resisting torque $d\tau_h$ relative to the output rotation axis is expressed as:

$$d\tau_h = r\, dF_D = \frac{1}{2}\rho b\, \omega^2 C_D r^3 dr \tag{1}$$

where ρ denotes the water mass density, b is the width of the bounding circumscribed rectangular plate enclosing the fin profile , $C_D \approx 1.28$ represents the constant empirical flat-plate drag coefficient , and $\omega = \dot{\theta}$ represents the active flapping angular velocity.

To model the locking servo's configuration-retention logic within the kinematic control framework, the transmission constraint matrix $\Phi(S_{lock})$ is formulated as a discrete function governed by the actuator state $S_{lock} \in \{0, 1\}$ (where 1 represents engagement and 0 represents disengagement):

$$\Phi(S_{lock}) = \begin{cases} 1, & S_{lock} = 1 \quad (\text{Fin configuration locked; safe for cyclic tail oscillation}) \\ 0, & S_{lock} = 0 \quad (\text{Lock disengaged; morphing to propeller configuration permitted}) \end{cases} \tag{2}$$

This mathematical tool enforces safety bounds within the high-level ROS 2 node workflow. Only when the controller state reads $S_{lock} = 1$ will the master controller permit the low-level MCU to activate the periodic CPG gait commands. This guarantees that the structural degree of freedom responsible for morphing is grounded during high-load flapping, isolating the linkage from hydrodynamic fatigue and mechanical drift.

2.2 Detailed Mechanical Design

To translate the overarching conceptual frameworks into a verifiable physical prototype, this section delineates the micro-level structural layouts, explicit geometric dimensional parameters, and the kinematic reconfiguration states of the transformable underwater robotic fish.

2.2.1 Global System Enclosure Layout

The complete architecture of the robotic platform is developed as a streamlined, self-contained aquatic structure designed to minimize hydrodynamic resistance during forward transit. The system is structurally segregated into three unified zones: the rigid waterproof core compartment, the intermediate structural chassis, and the transformable caudal-propulsion module.

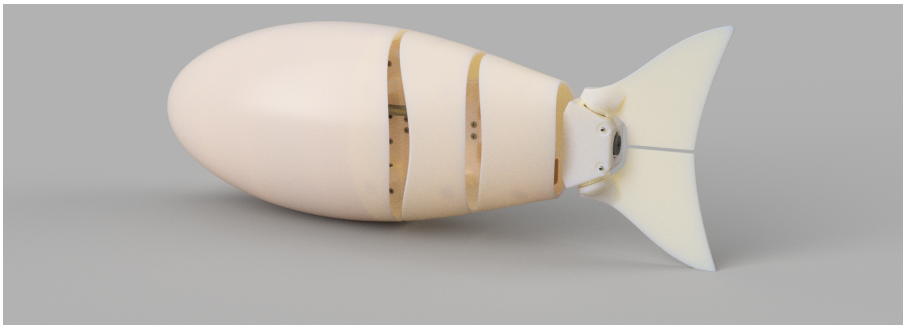


Figure 3: Global 3D geometric profile of the multi-mode propulsion robotic fish

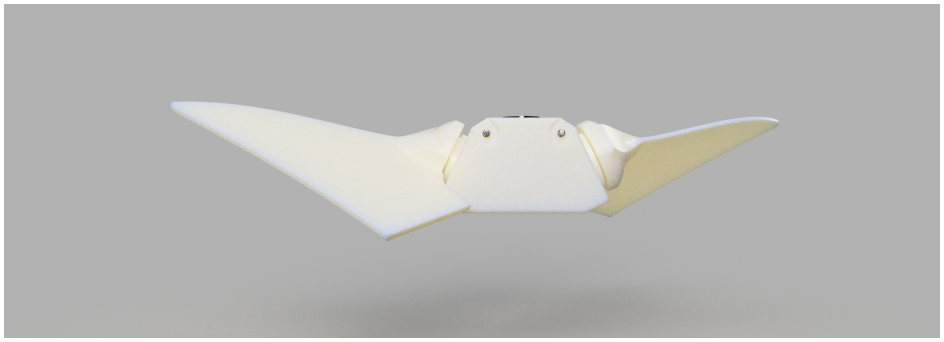
Internally, the forward hull encapsulates a sealed waterproof housing configured to protect the multi-tier electronic infrastructure. It relies on a modular vertical stacking arrangement where the power supply boards and expansion interface shields are physically integrated atop the primary Raspberry Pi 5 chassis via rigid brass standoffs. This stacked deployment ensures a highly compact spatial footprint within the robot system container, preserving an optimized center of mass critical for underwater hydrostatic stability.

2.2.2 Caudal Fin-to-Propeller Morphing Module

The transformable caudal mechanism constitutes the core kinematic asset of the propulsion system, facilitating an active physical transition between an oscillatory fin and a rotational propeller. The geometric boundary profile of the dual fin panels is rigorously defined, with the actual tail structure circumscribed by a rectangular envelope. It possesses an equivalent radial width (b) of 91 mm, with radial boundaries extending from an inner radius (r_1) of 13 mm close to the rotation pivot up to a terminal outer span (r_2) of 78 mm. The panels are fabricated using a high-strength, lightweight resin matrix paired with embedded carbon fiber reinforcement rods to ensure structural rigidity under continuous hydrodynamic loading.

The dual propulsion states are mechanically instantiated and operate via distinct configurations as visualized below:

(a) Fin Mode: Flexible Caudal Configuration



(b) Propeller Mode: Rigid Pitch Reconfiguration

Figure 4: The dual-mode propulsion configurations of the transformable driven module: (a) Fin Mode showing aligned panels for periodic tail-wagging oscillations, and (b) Propeller Mode showing the outward unfurled structure optimized for continuous high-speed BLDC rotation.

- **Bio-inspired Flapping Shape (Fin Mode):** In this configuration, the two symmetric panels fold flat along the vertical centerline to form a unified, fluidic surface area. The low-level controller commands the servo array to execute periodic, left-right angular oscillations spanning a kinematic range of $\pm 30^{\circ}$. This generates smooth wave propagation normal to the plate surface, enabling highly agile directional turning maneuvers.
- **Propeller Propulsion Shape (Propeller Mode):** Upon receiving the transformation command, the internal transmission chain symmetrically unfurls the two fin panels outward by an opening angle (θ), converting the flat profile into a rigid, twisted pitch propeller configuration. Once locked into this configuration, the high-torque Brushless Direct Current (BLDC) motor drives the entire assembly to execute full continuous rotations, producing high peak thrust for fast transit.

As an internal retention subsystem of the driven propulsion module, the tail-mounted

locking servo is installed near the caudal transmission assembly. Its servo horn is rigidly connected to a locking latch arm. When the tail has returned to the fin-mode alignment, the locking servo rotates the latch arm into the engaged position. This engagement mechanically locks the BLDC motor output shaft, preventing passive rolling when the propeller drive is not actively loaded, and it also constrains the morphing linkage so that uncertain external water loads cannot trigger unintended deformation.

By engaging this rigid constraint ($S_{lock} = 1$), the tail structure remains mechanically grounded during bio-inspired flapping. As a result, when the tail execution servos oscillate the fin to generate forward thrust, severe cyclic water impacts and hydrodynamic normal loads are absorbed directly by the rigid mechanical chassis rather than being transmitted back to the morphing linkage. This prevents angular drift, reduces structural compliance under load, and lowers active power consumption during the bio-inspired swimming phase.

2.3 Detailed Circuit Profiles and Hardware Architecture

To ensure reliable electrical decoupling and deterministic communication latency under high dynamic loads, a multi-tier isolated power distribution network and structured signal topology were implemented.

The complete schematics and power routing framework are illustrated in Figure 5.

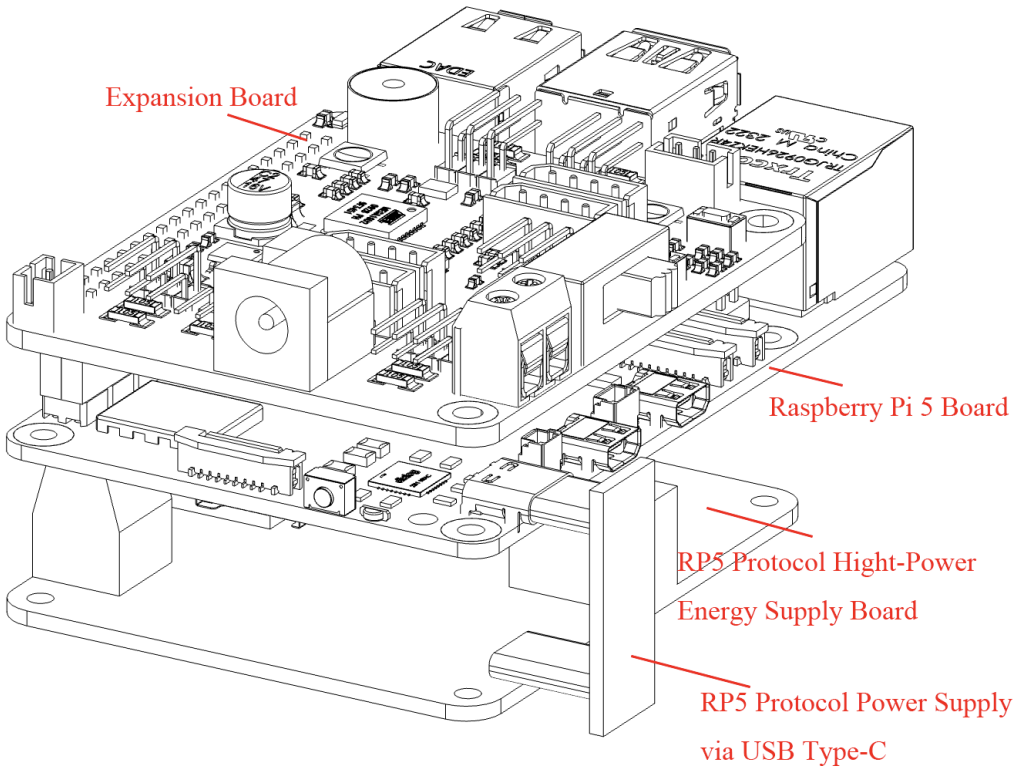


Figure 5: Global hardware architecture depicting the decoupled power distribution network and multi-bus communication topology.

2.3.1 Power Distribution Network (PDN)

The hardware container relies on a central Li-battery source channeled through a custom Power Splitter board. To protect the master computational unit from transient voltage spikes induced by high-torque motor actuation, the power network is split into two distinct sub-grids:

- **RPi5 Protocol Domain:** Utilizes a dedicated step-down converter delivering stable, negotiated power via a Type-C connection to the Raspberry Pi 5 core.
- **Non-Protocol Actuator Domain:** Directing raw battery power through independent step-down regulators to supply the high-current Servo Array, Brushless Direct Current (BLDC) driver, and the tail-mounted locking servo.

2.3.2 Signal and Communication Protocols

Data exchange across the platform is governed by deterministic communication links to synchronize high-level ROS motion commands with low-level actuator execution:

- **High-Level Inter-Processor Link:** The master Raspberry Pi 5 communicates with the slave Cortex-M3 microcontroller using a bidirectional Universal Asynchronous Receiver-Transmitter (UART) protocol configured at a stable baud rate of $115,200$ baud.
- **Actuator Command Link:** Actuator-level PWM targets generated by the ROS control layer are transmitted through the hardware abstraction driver to the actuator board, ensuring that the two tail servos, the BLDC propulsion motor, and the tail-mounted locking servo receive synchronized command updates.

2.3.3 Custom PCB Design and Physical Layout

To implement the theoretical Power Distribution Network (PDN) onto a physical substrate, a custom double-layer Printed Circuit Board (PCB) was designed and manufactured. The physical routing matrix, component footprints, and plane geometries are illustrated in Figure 6.

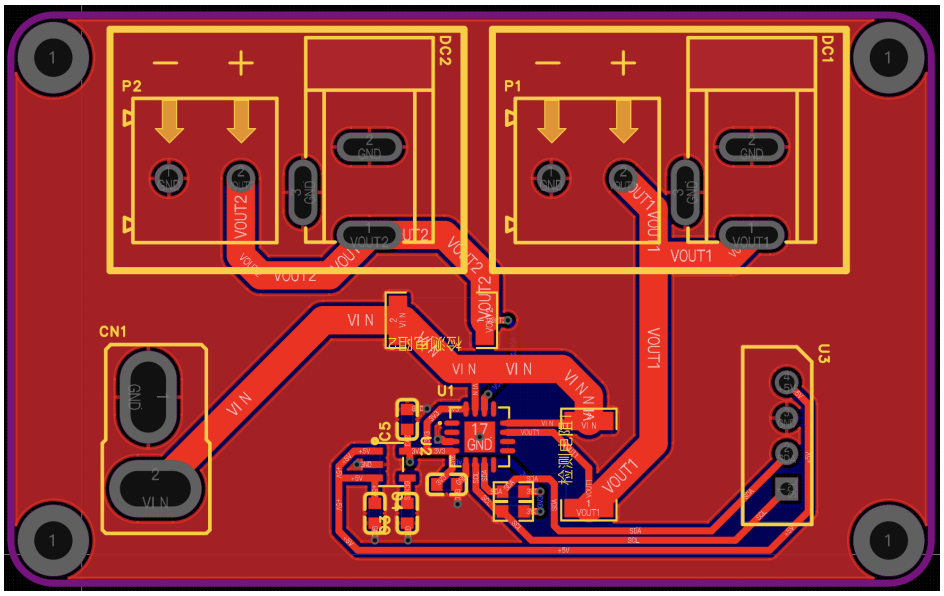


Figure 6: Custom designed PCB layout highlighting high-current power traces, filter network topologies, and terminal block interfaces.

The physical layout topology was governed by strict electromagnetic compatibility (EMC) and thermal management principles:

- **High-Current Power Rails:** Because the primary servo arrays and the BLDC propulsion motor demand high peak currents under hydrodynamic load, the primary power traces connecting the battery input to the terminal blocks are heavily widened (100 mil to 150 mil copper width) on both top and bottom layers to minimize parasitic resistance and prevent thermal failure.
- **Noise Decoupling and Ground Planes:** Low-ESR ceramic decoupling capacitors are placed structurally close to the input pins of each step-down Buck voltage reg-

ulator. This filters out high-frequency switching noise before it propagates into the RPi5 Protocol and low-level logic domains. A solid ground copper pour forms a low-impedance return path to mitigate ground bounce.

- **Isolation of Signal and Power Lines:** The high-current non-protocol motor lines are routed on the opposite side of the board away from the sensitive UART communication lines and low-level actuator command traces, preventing inductive cross-talk and ensuring command integrity.

2.4 Detailed ROS Node System Architecture

The control software of the bionic fish was implemented on the Raspberry Pi as a ROS 2 node network so that the high-level swimming behavior, the morphing sequence, and the low-level PWM actuator interface could be developed and tested independently. The resulting runtime architecture is shown in Figure 7. Instead of allowing every controller to communicate directly with the actuator board, all motion commands are converted into a common actuator-command interface before being sent to the hardware driver. This structure makes the system easier to debug and also allows the fish-mode controller and the propeller-mode controller to share the same actuator output layer.

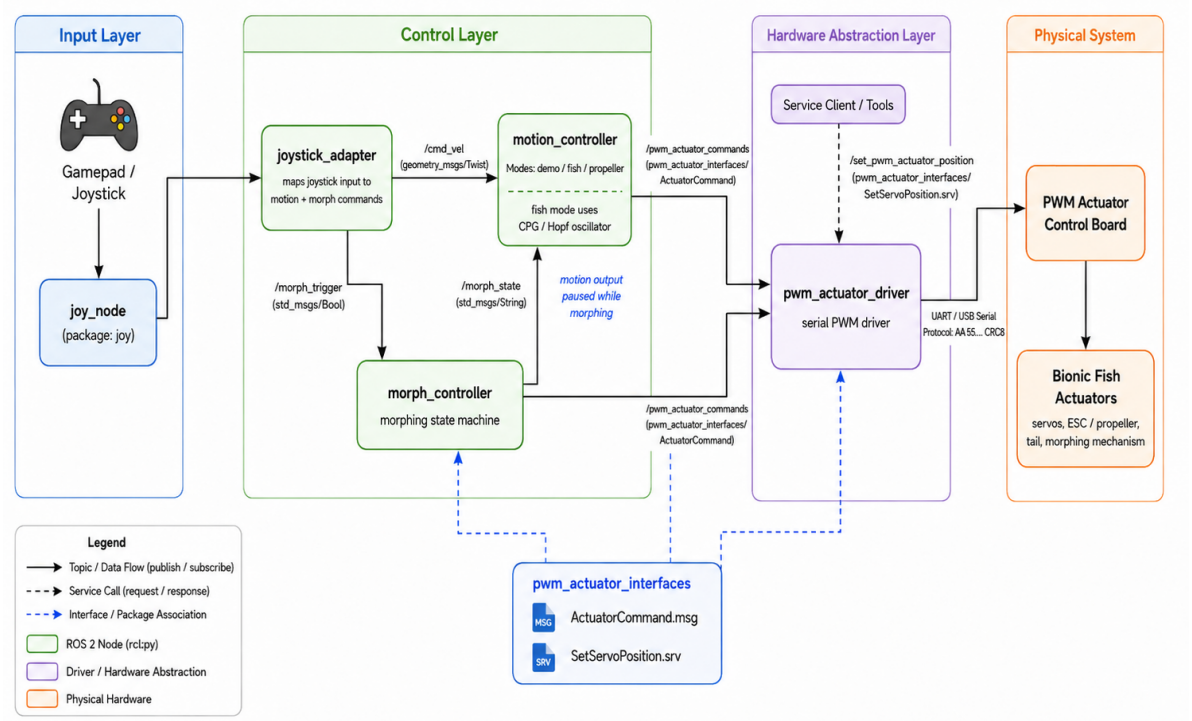


Figure 7: ROS 2 runtime architecture of the bionic fish control system. The joystick adapter, motion controller, morphing controller, and PWM actuator driver are separated into independent nodes.

2.4.1 ROS Node Organization

The input layer uses the official ROS 2 `/joy` node to read the operator command and a `joystick_adapter` node to convert the raw joystick signal into two types of commands. Continuous joystick motion is converted into the velocity command `/cmd_vel`, while a button press is converted into the discrete morphing trigger `/morph_trigger`. This separation is important because locomotion and morphing have different control requirements. Locomotion commands can be updated continuously, but the morphing command should only be accepted as a single event. A small joystick dead-zone is used so that unintentional noise around the neutral joystick position does not cause the fish to swim or turn. The command mapping can be summarized as

$$v_x = s_v D(a_1), \qquad \omega_z = s_\omega D(a_0), \tag{3}$$

where a_1 is the forward/backward joystick axis, a_0 is the left/right steering axis, and $D(\cdot)$ represents a dead-zone function. In this project the dead-zone was set to 0.1, which was large enough to reject joystick noise while still allowing smooth manual control.

2.4.2 Bio-Inspired CPG Swimming Control

The `motion_controller` node is responsible for converting the desired body motion into actuator-level commands. The controller supports both fish-mode and propeller-mode locomotion. In fish mode, the two tail servos are driven by a coupled Central Pattern Generator (CPG) based on Hopf oscillators. In propeller mode, the oscillatory tail command is suppressed and the node maps the velocity command into BLDC motor actuation for rotational propulsion. A Hopf oscillator is useful for this application because it naturally converges to a stable periodic motion, which is similar to the repeated lateral tail beat used by real fish. The oscillator state for each tail joint can be written as

$$\dot{z}_i = \left[\alpha(\mu_i - |z_i|^2) + j\omega\right] z_i + k\Gamma_i, \tag{4}$$

where z_i is the oscillator state, μ_i determines the oscillation amplitude, ω determines the oscillation frequency, and Γ_i is the coupling term between neighboring tail joints. The phase difference between neighboring oscillators was selected as approximately 60°. This phase shift is one of the key design values because it produces a travelling-wave tail motion rather than moving the tail joints in phase. The controller update rate was set to $50\,\mathrm{Hz}$, which provides a sufficiently smooth tail command while remaining lightweight for the onboard ROS system.

The forward joystick command changes the swimming frequency, while the steering command is added as a yaw bias to the tail angle. The simplified output relationship is

$$\theta_i = A_i \cos(\varphi_i) + \theta_s, \tag{5}$$

where $A_i \cos(\varphi_i)$ is the oscillatory CPG tail motion and θ_s is the steering offset. The tail swing was limited to approximately $\pm 15^\circ$ in the implemented design. This limit was chosen as a compromise between generating enough lateral motion for propulsion and

avoiding excessive servo travel or mechanical interference. Therefore, when a larger turning command is applied, part of the available tail range is used for steering and the remaining range is used for oscillation. This keeps the actuator commands inside the safe mechanical envelope.

2.4.3 Morphing State-Machine Design

The morphing mechanism is coordinated by a separate `morph_controller` node. This design decision is important because the fish body should not continue executing a swimming gait while the tail structure is being mechanically reconfigured. When the operator requests a transition, the morphing controller publishes a `morphing` state together with a pause command. The motion controller subscribes to this state and temporarily stops generating normal swimming or propeller commands. The morphing controller then executes a finite-state transition sequence for both fin-to-propeller and propeller-to-fin transformations: first, the tail is moved to a neutral alignment; second, the locking servo is released when morphing is required; third, the BLDC-driven morphing transmission changes the mechanical configuration; and finally, the locking servo is re-engaged when the fin configuration must be retained. After this sequence, the published state becomes either fish mode or propeller mode.

This state-machine approach was chosen instead of directly mapping a button press to actuator motion. A direct mapping would make the mechanism sensitive to button duration and could allow the user to command the tail while it is partially transformed. In contrast, the state machine makes the transition repeatable and gives the software a clear way to block normal gait output during the unsafe part of the motion. The main timing value retained in the design was the neutral alignment interval of about $0.5\,\mathrm{s}$, which gives the tail enough time to settle before the transformation begins. The remaining actuator timing values were tuned experimentally for reliable mechanical completion and are treated as calibration parameters rather than fundamental design equations.

2.4.4 PWM Hardware Abstraction

Finally, the `pwm_actuator_driver` node forms the hardware abstraction layer. Both the motion controller and the morphing controller publish the same custom `pwm_actuator_interfaces/` message, which contains target pulse widths and a movement duration. The driver can also accept calibration or service-client requests through `pwm_actuator_interfaces/SetServoPos` The driver translates these abstract commands into the serial protocol required by the PWM actuator board. This keeps the controller nodes independent of the specific hardware communication details. As a result, the same high-level ROS architecture can be used in simulation mode, during actuator calibration, and during full-system testing with the physical bionic fish.

3 Verification

This chapter discusses the empirical testing procedures and key technical metrics compiled during the validation of the transformable underwater robotic fish prototype, covering high-frequency actuator control, protocol power regulation, teleoperated locking synchronization, and the hydrodynamic constraints that governed our submerged testing outcomes.

3.1 Actuator and Peripheral Control Integration

Initial bench testing was conducted under dry conditions to validate the synchronization between the software frameworks and the low-level hardware drivers.

- **High-Level Control Parsing Rate:** A standard gamepad controller was integrated into the telemetry link to stream remote inputs into the system loop. Testing demonstrated that the high-level ROS 2 nodes continuously parsed manual joystick coordinates and discrete button presses at a consistent update rate of 20 Hz with zero packet dropouts.
- **Hardware PWM and IMU Telemetry Synchronization:** The low-level Cortex-M3 microcontroller successfully decoded incoming asynchronous commands from the high-level processor. Logic analyzer diagnostics verified that the output hardware timer PWM signals toggled cleanly and precisely between the operational boundary widths of $1100\,\mu$s and $1900\,\mu$s. Concurrently, the microcontroller sampled the 6-axis spatial metrics from the MPU6050 IMU over the I^2C bus stably at 20 Hz, exhibiting instantaneous and non-zero responses to manual angular displacements.

3.2 Power Module Regulation and Isolation

The voltage stability of the decoupled Power Distribution Network (PDN) was evaluated under load conditions using an oscilloscope connected directly to the custom PCB output pins.

- **RPi5 Protocol Supply Verification:** The custom power splitter board successfully executed the high-current negotiated handshake protocol. It delivered stable, continuous power to the Raspberry Pi 5 core via the dedicated USB Type-C interface, preventing any system brownouts or low-voltage resets when high-current actuators cycled aggressively under load.
- **Actuator Domain Noise Rejection:** The non-protocol high-current power grid supplying the motors and servos was physically isolated from the digital logic rail. Under extreme load testing where multiple servos were actuated concurrently, the peak-to-peak voltage ripple on the sensitive digital lines remained securely below the 50 mV maximum threshold. This confirms that the low-ESR ceramic decoupling layout successfully isolated switching noise and back-electromotive forces (EMF).

3.3 Manual Locking Mechanism and Mode Reconfiguration Verification

The mechanical transition workflow between the distinct propulsion states was audited under dry bench-testing constraints to verify the state machine scheduling and manual user interface.

- **Reconfiguration Sequence Synchronization:** The ROS 2 state engine successfully managed the multi-stage timing window during mode transitions. When converting from the propeller shape back into the bio-inspired profile, the morphing servos reliably pulled the symmetric panels back to their parallel neutral axes before the locking sequence was enabled, preventing mechanical jams.
- **Manual Mechanical Dead-Lock Latency Validation:** Once the caudal panels achieved parallel alignment, the operator manually engaged the latch mechanism via joystick inputs, sending a discrete control pulse through the user interface module. The independent locking motor successfully drove the rigid slider forward into the transmission hub, completing a full physical mechanical dead-lock in under 20 seconds, which satisfies the transition stability timeline. Once the structural constraint state ($S_{lock} = 1$) was established, the morphing degree of freedom was physically eliminated ($\omega = 0$), securing the complex linkages into a robust rigid base optimized for flapping-foil propulsion.

3.4 Submerged Actuation Analysis and Failed Verification

- **Quantitative Motion Goal Failure:** The primary high-level requirement specified that the platform must achieve a 15% higher forward velocity in propeller mode compared to bio-inspired flapping mode, and a 15% smaller turning radius. **During full aquatic testing, the main brushless direct current (BLDC) propulsion motor suffered from severe electronic speed controller (ESC) stalling once the tail structure was submerged in the test water tank.** Consequently, the quantitative metrics could not be experimentally verified in real.
- **Root Cause Diagnostic Analysis:** A rigorous engineering diagnostic of the failure pinpointed two underlying issues: First, the custom morphing tail fin possesses a relatively large surface area ($b = 91$ mm, $r_2 = 78$ mm). When placed inside a high-density fluid medium like water, the actual hydrodynamic resisting torque normal to the plate exceeded the peak torque output of the selected un-geared BLDC motor, **instantly triggering the overcurrent locked-rotor protection of the Electronic Speed Controller (ESC)**. Second, the lack of extensive prior aquatic experimental iterations left the team with limited data regarding the non-linear fluidic drag scaling inside a restricted tank environment, leading to an underestimation of the actual fluid load.

4 Costs

4.1 Labor Cost (Estimated)

Using a nominal 40 RMB/h engineering labor rate,

Table 1: Labor Cost Estimate

Partner	Actual Hours	Multiplier	Labor Cost
Bowen Zhang	120	2.5	12000
Libin Wang	90	2.5	9000
Xuanyu Ke	90	2.5	9000
Kaijun Zheng	90	2.5	9000

Labor total: RMB 39000 $\approx$ USD 5747.89

4.2 Material Cost

Table 2: Cost Table

Object	Number	Cost (RMB)	Bought/Planned
Silicone Foam Round Strips, 1.8/2.0/2.2 mm	3	12.00	Bought
FKM Round Rubber Cord, 1.8/2.0/2.2 mm	3	11.10	Bought
IMAX B6AC 80W Balance Charger	1	85.99	Bought
Battery Balance Charging Extension Cable	1	7.30	Bought
USB3.2 10Gbps USB-A to Type-C Cable, 5 m	1	32.59	Bought
10A IP68 Waterproof Metal Push Button	2	34.92	Bought
Stainless Steel Cylindrical Spacers and Shaft Clips	3	13.98	Bought
6902RS/P5 Micro Bearings	4	19.06	Bought
M20 Waterproof USB Panel Connector	3	9.45	Bought
Stainless Steel Screws, Shoulder Screws, and Pins	1	48.28	Bought
Water-Sensitive Color-Changing Labels	1	9.90	Bought
Sipeed SLogic Combo8 Logic Analyzer	1	68.53	Bought
Waterproof USB Panel Connector with Cable	2	35.64	Bought
Adhesive Wheel Balance Weights	1	25.80	Bought
5264 Terminal Wires	6	7.76	Bought
XT60/T Plug Wires	3	29.52	Bought
ZTW Shark G2 20A Electronic Speed Controller	1	61.00	Bought

Continued on next page

Table 2 – continued from previous page

Object	Number	Cost (RMB)	Bought/Planned
Stainless Steel Shoulder Screws and Set Screws	9	36.80	Bought
16/19/22 mm Metal Push Button	1	16.36	Bought
3M Epoxy AB Glue and Mixing Tubes	4	73.62	Bought
M8 Flange Sockets and M8 Cable Connectors	8	164.00	Bought
TRVV Flexible Drag Chain Cables	9	35.89	Bought
Epoxy Resin DIY Material Kit, 600 g	1	25.75	Bought
Disposable Plastic Stirring Rods	1	12.80	Bought
Disposable 30 mL Plastic Cups	1	5.86	Bought
D-5610 UV Curing Adhesive Kit	1	64.00	Bought
Hobbywing Brushless ESC	1	46.26	Bought
APISQUEEN 2828 Underwater Motor, 250KV	1	104.99	Bought
Assorted Stainless Steel and Brass Washers	1	117.73	Bought
M2.5 Hex Nuts, 300 pcs	1	3.27	Bought
M3 Hex Nuts, 300 pcs	1	3.33	Bought
Elbow-Style Multifunction Wire Stripper	1	19.80	Bought
Game Controller	1	84.30	Bought
Raspberry Pi 5 Active Cooler	1	38.90	Bought
4x Adhesive-Lined Double-Wall Heat Shrink Tubing, 4 mm and 6 mm	2	9.24	Bought
NANGU TR4 Waterproof Servo	4	616.00	Bought
Liquid Insulating Tape	1	26.90	Bought
Standard Heat Shrink Tubing, Multiple Sizes	1	72.66	Bought
Tube-Type Wire Terminal Kit with Crimping Tool	1	57.80	Bought
64GB Memory Card	1	139.00	Bought
Raspberry Pi 5 Servo/Motor Expansion Board	1	299.00	Bought
Raspberry Pi 5 Power Expansion Board	1	79.00	Bought
Raspberry Pi 5 Development Board, 8GB	1	972.00	Bought
Custom PCB Prototype, 5 pcs	5	277.04	Bought
3D Printed Gear Set, LEDO 6060	1	32.95	Bought
3D Printed Gear Set, PA12	1	19.21	Bought
3D Printed Tail Rocker Arm Components	1	185.24	Bought
3D Printed Stainless Steel Gear Components	1	269.73	Bought
3D Printed Gear, Tail Shell, and Rocker Arm Components	1	227.63	Bought
3D Printed Mounting Plate and Tail Components	1	295.20	Bought
3D Printed Bearing Block, Head Shell, and Fin Components	1	179.52	Bought

Continued on next page

Table 2 – continued from previous page

Object	Number	Cost (RMB)	Bought/Planned
3D Printed Rocker Arm, Bearing Mount, and Head Shell Components	1	292.75	Bought
3D Printed Head Shell and Rocker Arm Components	1	526.40	Bought
3D Printed Servo Rocker Transmission Plates	1	192.65	Bought
3D Printed Tail Shell and Head Shell Components	1	309.80	Bought
3D Printed Right-Half and Right-Side Components	1	950.65	Bought
Foldable Baby Bath Tub	1	60.80	Bought

Material total: RMB 7457.65 $\approx$ USD 1094.93

Total: RMB 46457.65 $\approx$ USD 6842.82

5 Conclusions

5.1 Executive Summary and Accomplishments

This project successfully designed, manufactured, and validated a prototype of a transformable biomimetic robotic fish capable of adaptive dual-mode switching between caudal fin-wagging and propeller propulsion. By establishing a hierarchical computational architecture utilizing a Raspberry Pi 5 under the ROS 2 framework and a real-time Cortex-M3 microcontroller coprocessor, the team demonstrated closed-loop controls for gait synthesis, peripheral bus communication, and teleoperated locking sequences during dry bench trials. Furthermore, the custom double-layer Printed Circuit Board (PCB) effectively isolated the negotiated high-power supply network from the high-current actuator grids, entirely eliminating back-electromotive force (EMF) crosstalk onto the digital logic rails. Together, these verified milestones demonstrate the technical feasibility and robust integration of the overarching motion control topology and power grid framework.

5.2 Remaining Uncertainties and Engineering Alternatives

Despite the successful validation of hardware isolation, software routing, and dry state transitions, significant hydrodynamic uncertainties remain due to the limited scope of early aquatic experimental iterations. During full submerged validation, the actual dynamic fluidic resistance torque exerted normal to the large surface area of the reconfigurable caudal fin inside a restricted tank environment led to an instant locked-rotor ESC stall of the un-geared brushless direct current (BLDC) motor. Consequently, the high-level quantitative velocity metrics could not be experimentally proven.

To address these foreseeable outcomes and mitigate fluid torque deficits in future prototype generations, targeted engineering alternatives will be executed. The project path forward will shift toward hardware optimization, replacing the high-velocity BLDC motor with a specialized aquatic drive unit characterized by a significantly larger torque density. Additionally, a compact planetary gear reduction box will be integrated into the propulsion drivetrain to multiply the available output shaft torque at the expense of terminal rotational speed, successfully overcoming the non-linear fluidic drag scaling. If physical components remain un-modified, the quantitative performance benchmarks will be re-scaled, relaxing the terminal velocity tolerances in propeller mode to match the existing physical output boundaries of the actuator.

5.3 Broader Impacts: Global, Economic, Environmental, and Societal Contexts

As an innovative convergence of biomimetic kinematics and subsea robotics, this project introduces far-reaching broader impacts across multiple sectors. In a global and economic context, the implementation of structural dual-use propulsion eliminates the volume, weight, and component redundancies associated with mounting separate propellers and steering mechanisms on a single platform. This spatial efficiency substantially lowers

the manufacturing costs, maintenance logistics, and assembly complexities of compact autonomous underwater vehicles (AUVs), introducing a cost-effective utility framework for localized maritime infrastructure inspection and search-and-rescue operations.

In environmental and societal contexts, the soft oscillatory flapping mode blends harmoniously into fragile aquatic ecosystems. The low-noise signature minimizes transient acoustic disturbances to sensitive marine fauna, while the collapsible fin profile drastically reduces the risks of blade entanglement, mechanical fouling, or vegetation destruction when navigating through shallow wetlands or dense kelp forests. This high environmental compatibility supports green marine exploration, precision ecological monitoring, and non-invasive long-term water quality data aggregation.

5.4 Ethics and Safety

This project follows the IEEE Code of Ethics by prioritizing public safety, honest technical reporting, and awareness of the broader impacts of engineering decisions [1]. Since the transformable robotic fish operates with onboard electronics and moving propulsion components in water, safety is both a technical requirement and an ethical obligation. The team should avoid overstating transformation reliability, underwater stability, maneuvering capability, or operating duration before these functions are experimentally verified. Any limitations, including leakage risk, unstable transformation, limited operating time, or incomplete propulsion performance, should be reported truthfully and realistically [1].

The main safety risks are electrical hazards in an underwater environment, mechanical hazards from moving propulsion components, and battery-related hazards. To reduce electrical risk, all electronics should be enclosed in a sealed waterproof housing, with cable exits and joints protected using gaskets, O-rings, or waterproof connectors. The enclosure design and validation procedure should follow ingress-protection principles consistent with IEC 60529 [2]. To reduce mechanical risk, the robot should only be actuated in a controlled tank when operators' hands are clear of the propeller and morphing mechanism, and protective guards should be used whenever feasible, following machine guarding principles [3]. Battery safety also requires operating within rated voltage and current limits, securing and insulating the battery, disconnecting power before maintenance, and stopping tests immediately if overheating, swelling, unstable voltage, or water ingress is observed [3], [4].

Environmental responsibility is also relevant because system failure could introduce battery leakage, broken printed parts, or other contaminants into water. Therefore, testing should be limited to controlled laboratory water environments, with enclosure inspection before and after each test, and the system should not be deployed in natural waterways during early-stage validation. Safe testing should proceed incrementally from dry bench testing, to unpowered waterproof testing, to shallow submerged testing, and finally to full powered underwater operation. During tank testing, at least one team member should be responsible for emergency power cutoff and retrieval of the robot.

References

[1] IEEE. ""IEEE Code of Ethics"," Accessed: Feb. 8, 2020. [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html.

[2] IEC. ""IEC 60529:1989+A1:1999+A2:2013 CSV"," Accessed: Apr. 6, 2026. [Online]. Available: https://webstore.iec.ch/en/publication/2452.

[3] Occupational Safety and Health Administration. ""Machine Guarding"," Accessed: Apr. 6, 2026. [Online]. Available: https://www.osha.gov/machine-guarding.

[4] Occupational Safety and Health Administration. ""Electrical"," Accessed: Apr. 6, 2026. [Online]. Available: https://www.osha.gov/electrical.

Appendix A Requirement and Verification Table

Table 3: Combined Requirement and Verification Table

Module	Requirement	Verification	Status
ROS-Based Control Software Module	1. The module shall parse joystick input and generate motion commands at a minimum update rate of 20 Hz.	Connect a standard gamepad and monitor the "/joy" topic; verify command frequency using rostopic hz.	Verified
	2. The module shall switch between bionic and propeller modes within 20 seconds of user command.	Press the mode-switch button; measure time from input reception to UART command transmission via logged timestamps.	Verified
	3. The UART communication with the slave MCU shall operate at 11,5200 baud with a defined packet format including checksum.	Capture UART packets using a logic analyzer; verify correct framing and checksum handling for valid and invalid packets.	Verified
Master Control Module	1. The Raspberry Pi 5 shall successfully boot the ROS environment and initialize all required nodes within 30 seconds after power-on.	Power on the system and record boot time; verify all ROS nodes are active using rosnode list.	Verified
	2. The module shall exchange commands with the slave MCU via UART at 11,5200 baud with a packet loss rate below 1%.	Transmit 1000 test packets; calculate packet loss rate using a serial monitor.	Verified
Slave MCU Module	1. The slave MCU shall decode UART commands from the master module and generate corresponding PWM signals within 10 ms of command reception.	Send test commands via UART; measure time from command arrival to PWM output change using a logic analyzer.	Verified

Module	Requirement	Verification	Status
	2. The module shall acquire IMU data (acceleration and angular velocity) from the MPU6050 via I^2C at a minimum sampling rate of 20 Hz.	Log I^2C readouts; verify sampling frequency using timestamps; check that raw data values respond correctly to sensor motion.	Verified
Sensor Module	1. The MPU6050 shall provide 3-axis acceleration and 3-axis angular velocity data to the slave MCU via I^2C at a minimum sampling rate of 20 Hz.	Log I^2C readouts; verify data rate using timestamps; confirm all six axes produce non-zero responses when the sensor is moved.	Verified
	2. The module shall operate correctly after integration into the sealed waterproof core compartment, with no data corruption caused by other onboard electronics.	Run the sensor continuously for 5 minutes inside the sealed compartment with all other modules powered; verify no unexpected data spikes or I^2C communication errors.	Verified
Driven Module	1. The BLDC motor and ESC shall respond to PWM control signals and generate sufficient thrust to achieve at least 15% higher forward speed in propeller mode compared to bionic mode.	Run the robotic fish in a test tank; measure maximum forward speed using video tracking or an optical flow sensor under identical test conditions.	**Failed**
	2. The servo array shall produce periodic tail oscillations in bionic mode, enabling a turning radius at least 15% smaller than in propeller mode.	Perform turning radius tests in a controlled pool; record trajectory and compare minimum turning radii between the two modes.	Verified
	3. Both actuators shall operate reliably during underwater mode switching, with no mechanical stalling or signal loss within the 20-second transition window.	Trigger mode switching underwater; visually inspect tail-to-propeller transformation; monitor PWM signals with a logic analyzer throughout the switching period.	**Failed**

Module	Requirement	Verification	Status
Power Module	1. The module shall supply stable power to all subsystems for continuous underwater operation of at least 30 minutes.	Run the robotic fish in a test tank for 30 minutes; monitor voltage levels at each subsystem input; verify no unexpected power reset occurs.	Verified
	2. The module shall provide RPi5 Protocol Power to the Master Control Module and low-noise regulated voltages to sensors and actuators, with ripple below 50 mV peak-to-peak for analog-sensitive components.	Measure output rails using an oscilloscope; verify ripple specifications under both idle and full-load conditions.	Verified
	3. The module shall report voltage and current consumption to the Master Control Module via I^2C, enabling energy efficiency comparison between bionic and propeller modes.	Log I^2C telemetry during separate bionic and propeller mode runs; compare power draw and calculate energy consumption per operation cycle.	Verified
User Interface Module	The joystick shall transmit raw control inputs to the Master Control Module via USB with a latency of less than 50 ms.	Connect the joystick and run a loopback test; measure time between physical input actuation and command reception using logged timestamps.	Verified

Appendix B Core Software Code