Comparison of Cascaded Feedback Linearization with PID Control Law with PID for Underactuated AUV

***Abstract*— This paper focuses on the control of Underactuated autonomous Underwater vehicles (AUVs). These vehicles inherently have non-linear dynamics and limited actuation, which present significant challenges. We have proposed a cascaded controller comprising an inner loop feedback linearization with an outer loop PID. The proposed method is compared against a conventional PID controller to evaluate its performance. Simulations are carried out using the ROS2 middleware and the Gazebo simulation environment. The performance of the controller is assessed under various scenarios, including trajectory tracking accuracy, robustness to initial condition variations.**

I. INTRODUCTION

Robotics is becoming part of our daily life, industry, and defense. Robots are being used for the exploration of new areas and for protecting geographically tough areas. One such area is Underwater. Both for exploration and security purposes. Underwater Vehicles, such as submarine,s have been used for the past few decades. But with advancements, the focus is on autonomous underwater vehicles. For achieving autonomy, one major aspect is control of the AUV. The higher DoF compared to a ground vehicle makes it challenging. Apart from making their operating time longer, usually they are underactuated, which makes their control more complex. We have studied the literature related to this problem. We have found that Bashir et al. [1] reviewed control algorithms including model predictive control, adaptive control, H control, fuzzy logic, PID, and backstepping for trajectory control of unmanned underwater vehicles.

Makam et al. [2] studied the mathematical modelling of underwater vehicles in detail, along with controllers. Specifically, they discussed the depth pitch controller, the Yaw Controller for the linearized model. In nonlinear control, they had compared the SMC with conventional controllers like PID. Steenson et al. [3] focused on model predictive control for depth regulation of AUVs. In particular they have used Delphin2 AUV which is an overactuated thrusters and control surfaces.

Vadapalli et al. [4] focused on 3D path following of AUV using a backstepping approach for robust control. They have used Linear Matrix Inequality for controller design. They have not discussed about feedback linearization this paper. Esfahani et al. [5] worked on improving the tracking accuracy for motion control of AUVs under external changes.To make a robust controller they have used Backstepping,Integral Sliding Mode Control and time delay control. Zhou et al. [6] focused on 5 DoF motion model with input delays. They focused on underactuated vehicles with time delays and proposed a controller based on sliding mode control.

Vu et al. [7] focused on adaptive controller based on Sliding Mode for underactuated AUVs. They focused on bringing robust control and adaptive learning together. They proposed a two level hierarchical controller.

Lainag X .[8] addressed the path following of an underactuated AUV with external parameters. They proposed an adaptive robust control system using fuzzy logic,backstepping an sliding mode control.

There has been work on the control of underwater vehicles. Feedback linearization also has been employed, but not with respect to underactuated underwater vehicles, which poses new challenges. We will address these new challenges and will design a cascaded controller using feedback lineraization in outer loop and PID in inner loop. Will try to control major DoF like pitch, yaw, and surge under various scenarios. It demonstrates a clear control challenge (nonlinearity, underactuation, coupling via fins, dependence on surge velocity), and a structured control solution. While nonlinear and adaptive controllers have been widely studied, feedback linearization has received comparatively less attention in the context of underactuated AUVs, especially those actuated via surge-dependent fins. Most works assume fully actuated vehicles or neglect the coupling effects introduced by fin-based actuation.

II. MATHEMATICAL MODELLING AND CONTROLLER DESIGN

A. Mathematical Modelling of Underwater Vehicle

Following Fossen 's book the following kinematics and Dynamics have been used:- Following the Manuvering Theory Given by T. I. Fossen [9], We have worked on kinematics and dynamics of surface and underwater vehicles.

Kinematics of Six DOF Underwater Vehicle:- It describes how the vehicle's position (x, y, z) and orientation (roll ϕ, pitch θ, yaw φ) change over time based on its control inputs like linear velocities (u, v, w) and angular velocities (p, q, r). Essentially, it captures the complex motion dynamics of the vehicle as it navigates underwater.

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \dot{\phi} \\ \dot{\theta} \\ \dot{\varphi} \end{bmatrix} = \begin{bmatrix} c(\varphi)c(\theta) & -s(\phi)c(\phi)+c(\varphi)s(\theta)s(\phi) & s(\varphi)s(\phi)+c(\varphi)c(\phi)s(\theta) & 0 & 0 & 0 \\ s(\varphi)c(\theta) & c(\varphi)c(\phi)+s(\phi)s(\theta)s(\varphi) & -c(\varphi)s(\phi)+s(\theta)s(\varphi)c(\phi) & 0 & 0 & 0 \\ -s(\theta) & c(\theta)s(\varphi) & c(\theta)c(\phi) & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & s(\phi)t(\theta) & c(\phi)t(\theta) \\ 0 & 0 & 0 & 0 & c(\phi) & -s(\phi) \\ 0 & 0 & 0 & 0 & s(\phi)se(\theta) & c(\phi)se(\theta) \end{bmatrix} \begin{bmatrix} u \\ v \\ w \\ p \\ q \\ r \end{bmatrix}$$

As per craig's model:-

$$\dot{\eta} = J(\eta)\nu$$

B. Three DOF Dynamic Model

The kinetic equations of motion for an Autonomous Underwater Vehicle (AUV) in 3 Degrees of Freedom (DOF) are given by:

$$M\dot{\nu} + C(\nu)\nu + D(\nu)\nu + g(\eta) = \tau$$

Where:

- M is the inertia matrix (including added mass),
- $\dot{\nu}$ is the time derivative of the velocity vector (linear and angular accelerations),
- $C(\nu)$ is the Coriolis and centripetal matrix (depends on velocity ν),
- $D(\nu)$ is the damping matrix (due to hydrodynamic drag),
- $g(\eta)$ is the vector of gravitational and buoyant forces and moments (depends on the position and orientation vector η),
- τ is the control input (forces and moments from thrusters or control surfaces),
- ν is the velocity vector, typically:

$$\nu = \begin{bmatrix} u \\ w \\ r \end{bmatrix}$$

  where u is the surge velocity, w is the heave velocity, and r is the yaw rate,
- η is the position and orientation vector, typically:

$$\eta = \begin{bmatrix} x \\ z \\ \varphi \end{bmatrix}$$

  where x is the position in surge, z is the position in heave, and φ is the yaw angle.

Inertia Matrix (M):

$$M = \begin{bmatrix} m_{11} & 0 & 0 \\ 0 & m_{22} & 0 \\ 0 & 0 & m_{33} \end{bmatrix}$$

where m_{11}, m_{22}, m_{33} are the mass and added mass terms in surge, heave, and yaw, respectively.

Coriolis and Centripetal Matrix ($C(\nu)$):

$$C(\nu) = \begin{bmatrix} 0 & 0 & -m_{22}w \\ 0 & 0 & m_{11}u \\ m_{22}w & -m_{11}u & 0 \end{bmatrix}$$

Damping Matrix ($D(\nu)$):

$$D(\nu) = \begin{bmatrix} d_{11} & 0 & 0 \\ 0 & d_{22} & 0 \\ 0 & 0 & d_{33} \end{bmatrix}$$

where d_{11}, d_{22}, d_{33} are the damping coefficients in surge, heave, and yaw.

Restoring Forces and Moments ($g(\eta)$):

$$g(\eta) = \begin{bmatrix} 0 \\ -(W - B) \\ 0 \end{bmatrix}$$

where W is the weight of the vehicle, B is the buoyancy force.

C. PID Controller

The following PID control laws are designed, where:
- Surge motion is directly actuated by a thruster.
- Pitch and yaw are controlled by fins that produce hydrodynamic moments only when there is sufficient forward (surge) velocity.

Surge Control (Thruster)

$$e_u(t) = u_d(t) - u(t) \tag{1}$$

$$T = K_{P,u}\, e_u(t) + K_{I,u} \int_0^t e_u(\tau)\, d\tau + K_{D,u}\, \frac{de_u(t)}{dt} \tag{2}$$

where T is the control thrust applied, and $K_{P,u}, K_{I,u}, K_{D,u}$ are the PID gains for surge.

Pitch Control (Fin, Surge-Dependent)

$$e_\theta(t) = \theta_d(t) - \theta(t) \tag{3}$$

$$\tau_\theta^{des} = K_{P,\theta}\, e_\theta(t) + K_{I,\theta} \int_0^t e_\theta(\tau)\, d\tau + K_{D,\theta}\, \frac{de_\theta(t)}{dt} \tag{4}$$

$$\delta_p = \frac{\tau_\theta^{des}}{k_\theta u^2 + \epsilon} \tag{5}$$

Here, τ_θ^{des} is the desired pitch moment, and δ_p is the pitch fin deflection. The term $k_\theta u^2$ captures the nonlinear relationship between surge speed and control effectiveness, and ϵ is a small constant to prevent division by zero.

Yaw Control (Fin, Surge-Dependent)

$$e_\psi(t) = \psi_d(t) - \psi(t) \tag{6}$$

$$\tau_\psi^{des} = K_{P,\psi}\, e_\psi(t) + K_{I,\psi} \int_0^t e_\psi(\tau)\, d\tau + K_{D,\psi}\, \frac{de_\psi(t)}{dt} \tag{7}$$

$$\delta_y = \frac{\tau_\psi^{des}}{k_\psi u^2 + \epsilon} \tag{8}$$

δ_y is the yaw fin deflection, dependent on the available surge velocity u.

Notes on Interdependence

The effectiveness of the pitch and yaw controllers is directly influenced by the surge velocity u, making it essential to maintain a minimum forward speed for adequate control authority. This interdependence is a hallmark of underactuated AUV control dynamics.

D. Cascaded Controller

Cascaded control offers a structured way to manage challenges of underactuation by decomposing the control problem into two nested loops. The *outer loop* regulates the vehicle's position by comparing the desired position p_d to the measured position p, and uses a PID regulator to generate a commanded velocity v_d. The *inner loop* then regulates velocity: it compares v_d to the measured velocity v and uses a second PID to produce a desired acceleration a_d. This separation simplifies tuning—each loop handles one integration order of the vehicle's kinematics—and improves robustness by limiting the bandwidth of each controller.

To handle the AUV's nonlinear dynamics in the inner loop, we employ *feedback linearization*. Given the standard rigid-body model

$$M(q)\,\ddot{q} + C(q,\dot{q})\,\dot{q} + D(q,\dot{q})\,\dot{q} + g(q) \;=\; \tau \qquad (9)$$

where q is the configuration vector, M the inertia matrix, C the Coriolis/centripetal terms, D the hydrodynamic damping, and g the restoring forces, we choose

$$\tau \;=\; M(q)\,a_d \;+\; C(q,\dot{q})\,\dot{q} \;+\; D(q,\dot{q})\,\dot{q} \;+\; g(q) \qquad (10)$$

This exact inversion cancels all nonlinearities, yielding the linear input–output relationship $\ddot{q} = a_d$. The inner PID then effectively controls a double-integrator, vastly simplifying stability and performance analysis.

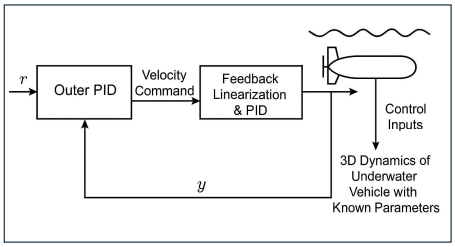

Fig. 1: AUV State Control Flow-chart

In short, the cascaded PID structure decouples position and velocity regulation, while feedback linearization cancels the AUV's nonlinear terms. The result is a controller that is both easy to tune and capable of high-performance trajectory tracking, even in the presence of strong hydrodynamic effects.

E. Controller Design

AUV Parameters Physical Properties

- Mass of AUV: $m = 148.3571$ kg
- Neutral buoyancy assumed: $W = B$
- Propeller diameter: $d_p = 0.2$ m

State Variables

- u: Linear velocity in surge (x-axis)
- q: Angular velocity in pitch (y-axis)
- r: Angular velocity in yaw (z-axis)
- θ: Pitch angle
- ψ: Yaw angle

Dynamic Equations (3 DOF)

TABLE I: AUV Parameters

AUV Parameters	
Inertia	
I_{xx}	$3.0000\ \text{kg}\cdot\text{m}^2$
I_{yy}	$41.980233\ \text{kg}\cdot\text{m}^2$
I_{zz}	$41.980233\ \text{kg}\cdot\text{m}^2$
Added Mass	
$X_{\dot{u}}$	-4.876161
$M_{\dot{q}}$	-33.46
$N_{\dot{r}}$	-33.46
Hydrodynamic Drag	
$X_{\lvert u\rvert u}$	-6.2282
$M_{\lvert q\rvert q}$	-632.698957
$N_{\lvert r\rvert r}$	-632.698957

Let the total surge, pitch, and yaw inertias be:

$$m_u = m - X_{\dot{u}} = 153.2333\ \text{kg} \qquad (11)$$

$$I^*_{yy} = I_{yy} - M_{\dot{q}} = 75.440233\ \text{kg}\cdot\text{m}^2 \qquad (12)$$

$$I^*_{zz} = I_{zz} - N_{\dot{r}} = 75.440233\ \text{kg}\cdot\text{m}^2 \qquad (13)$$

Surge

The decoupled equation in surge

$$m_u\dot{u} = -m_u ur + X_{|u|u}|u|u + \tau_u \qquad (14)$$

Where

m : Mass of the vehicle
$X_{\dot{u}}$: Added mass in surge direction
X_u : Linear damping coefficient in surge
$X_{|u|}$: Quadratic damping coefficient in surge
$|u|$: Magnitude of Surge Velocity
τ_u : Thrust along x-axis (from main propeller)

Pitch

$$(I_y - M_{\dot{q}})\dot{q} + M_q q + M_{|q|}|q|q = \tau_q \qquad (15)$$

I_y : Moment of inertia around y-axis (pitch)
$M_{\dot{q}}$: Added moment of inertia in pitch
$q = \dot{\theta}$: Pitch rate (time derivative of pitch angle)
M_q : Linear damping coefficient in pitch
$M_{|q|}$: Quadratic damping coefficient in pitch
τ_q : Pitching moment (from horizontal fins)

Yaw

$$(I_z - N_{\dot{r}})\dot{r} + N_r r + N_{|r|}|r|r = \tau_r \qquad (16)$$

Where align* I_z : Moment of inertia around z-axis (yaw)
$N_{\dot{r}}$: Added moment of inertia in yaw
N_r : Linear damping coefficient in yaw
$N_{|r|}$: Quadratic damping coefficient in yaw
τ_r : Yaw moment (from rudders/fins)

Under neutral buoyancy and symmetric design, restoring moments are negligible.

1) Feedback Linearization: **Surge (u)**

$$f_u(u,r) = -m_u ur + X_{|u|u}|u|u \tag{17}$$

Writing Surge Dynamics in standard non-linear form

$$\dot{u} = \frac{1}{m_u}\left(\tau_u + f_u(u,r)\right) \tag{18}$$

Define virtual control v_u:

$$\dot{u} = v_u \quad \Rightarrow \quad \tau_u = m_u v_u - f_u(u,r)$$

$$\tau_u = m_u v_u + m_u ur - X_{|u|u}|u|u \tag{19}$$

Pitch (q)

$$f_q(q) = M_q q + M_{|q|q}|q|q \tag{20}$$

Writing Pitch Dynamics in standard non-linear form

$$\dot{q} = \frac{1}{I^*_{yy}}\left(\tau_q - f_q(q)\right) \tag{21}$$

Define virtual control v_q:

$$\dot{q} = v_q \quad \Rightarrow \quad \tau_{pitch} = I^*_{yy} v_q - f_q(q) \tag{22}$$

$$\tau_q = I^*_{yy} v_q + M_q q + M_{|q|q}|q|q \tag{23}$$

Yaw (r)

$$f_r(r) = N_r r + N_{|r|r}|r|r \tag{24}$$

$$\dot{r} = \frac{1}{I^*_{zz}}\left(\tau_r - f_r(r)\right) \tag{25}$$

Define virtual control v_r:

$$\dot{r} = v_r \quad \Rightarrow \quad \tau_r = I^*_{zz} v_r + f_r(r) \tag{26}$$

$$\tau_r = I^*_{zz} v_r + N_r r + N_{|r|r}|r|r \tag{27}$$

PID Controller on Virtual Inputs

Once the nonlinearities are canceled, the system behaves like a set of integrators, where the virtual input directly controls the rate of change of the state. At this point, our goal becomes: Control the system states by properly shaping their rate of change — i.e., the virtual inputs. To achieve this, we design a PID controller on the virtual inputs. The PID controller computes the appropriate virtual input based on the tracking error (e.g., $(u_d - u)$ to ensure that the actual state converges to the desired value smoothly and accurately.

Surge PID

$$v_u = K_{p_u}(u_d - u) + K_{i_u}\int (u_d - u)\,dt + K_{d_u}(\dot{u}_d - \dot{u}) \tag{28}$$

Pitch PID

$$v_q = K_{p_q}(q_d - q) + K_{i_q}\int (q_d - q)\,dt + K_{d_q}(\dot{q}_d - \dot{q}) \tag{29}$$

Yaw PID

$$v_r = K_{p_r}(r_d - r) + K_{i_r}\int (r_d - r)\,dt + K_{d_r}(\dot{r}_d - \dot{r}) \tag{30}$$

Outer-Loop PID Controller (for Position Control)

Once the system is linearized using feedback linearization and the inner-loop PID controllers are applied to track the desired velocities, we can further extend the control architecture to perform **position tracking**. Since velocity is the time derivative of position, we treat the *velocity reference* (e.g., u_d, q_d, r_d) as a virtual input to a **second (outer) PID controller** that operates on **position error**.

This forms a *cascaded control structure*, where the outer-loop PID controller generates the *desired velocity* based on position tracking error, and the inner-loop PID controller ensures that the system follows that desired velocity accurately.

The outer-loop PID controller computes the velocity command by minimizing the position error over time. This allows the system states (e.g., surge position, pitch angle, yaw angle) to *converge to their respective desired values smoothly and with minimal steady-state error*.

Position PID Controllers: **Surge Position PID :**

$$u_d = K_{px}(x_d - x) + K_{ix}\int (x_d - x)\,dt + K_{dx}(\dot{x}_d - \dot{x}) \tag{31}$$

Pitch Angle PID (θ control):

$$q_d = K_{p\theta}(\theta_d - \theta) + K_{i\theta}\int (\theta_d - \theta)\,dt + K_{d\theta}(\dot{\theta}_d - \dot{\theta}) \tag{32}$$

Yaw Angle PID (ψ control):

$$r_d = K_{p\psi}(\psi_d - \psi) + K_{i\psi}\int (\psi_d - \psi)\,dt + K_{d\psi}(\dot{\psi}_d - \dot{\psi}) \tag{33}$$

Each outer-loop PID controller ensures that the vehicle moves toward its *position or orientation setpoint*. The output of this PID (desired velocity) becomes the *setpoint* for the *inner-loop controller*, which then commands the system accordingly.

Notes

- Restoring moments are neglected due to neutral buoyancy and symmetric configuration.
- PID gains (K_p, K_i, K_d) should be tuned based on desired tracking performance.

III. Simulation

A. Environment Setup

In this section, we describe the simulation environment that integrates Gazebo with ROS2 to implement a multi-agent framework with obstacle avoidance for marine robots.

a) Justification for Using Gazebo:: Gazebo is chosen as the simulation platform due to its realistic physics, robust ROS2 integration, and extensive sensor support. Its ability to simulate complex environmental effects—such as waves, currents, and drag forces—provides a highly realistic testing ground for evaluating the performance of our autonomous systems.

b) Simulation World:: The simulation world is designed to replicate an underwater and surface environment with carefully modeled physical properties.This includes factors like fluid dynamics and environmental disturbances to mimic real-world underwater conditions.I

TABLE II: Simulation Parameters

Parameter	Value/Description
Water Density	1025 kg/m³
Drag Coefficient	0.8
Gravity	9.81 m/s²
Gazebo Version	Gazebo Harmonic
ROS2 Version	ROS 2 Jazzy

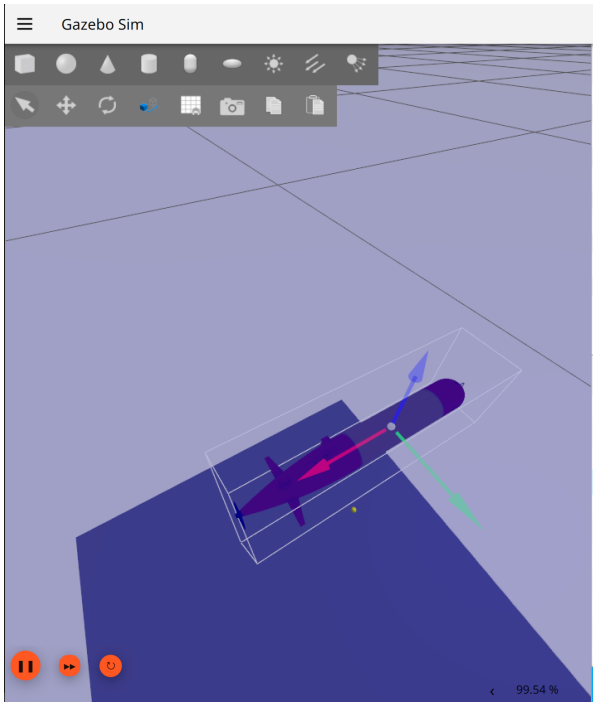

Fig. 2: Gazebo Sim Underwater World With AUV

c) Robot Model:: The robot models, such as AUVs and ASVs, are constructed using SDF/URDF. These models incorporate detailed representations of physical dimensions, mass properties, and sensor placements, ensuring accurate simulations of robot dynamics and interactions.[10]

d) Physics Engine:: For physics simulation, we employ the Open Dynamics Engine (ODE), which provides realistic dynamics and collision handling essential for underwater simulations.

e) Sensor Models:: The environment includes multiple sensor models (IMU, camera, LiDAR,GPS and Compass) integrated via ROS2 topics. This setup ensures comprehensive perception data for tasks such as navigation and obstacle avoidance.

f) Visualization Tools:: The simulation setup is complemented by several visualization tools:

- ROS2 Node Graph using `rqt_graph`
- Gazebo World Screenshots
- RViz visualization of the robot and sensor data
- Velocity/Trajectory Graphs for performance analysis

g) ROS2 Communication:: The simulation leverages ROS2 for inter-node communication.

B. Results

Performance Metrics to Evaluate Controllers

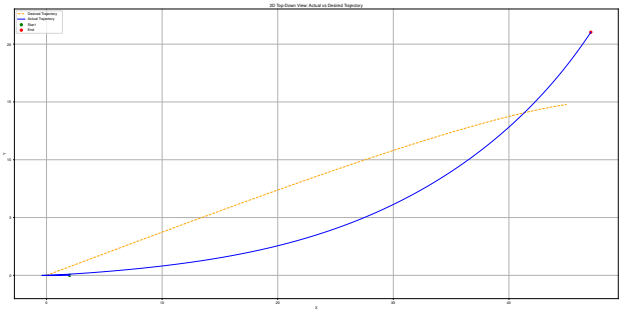

Fig. 3: Trajectory Tracking Using Cascaded Controller

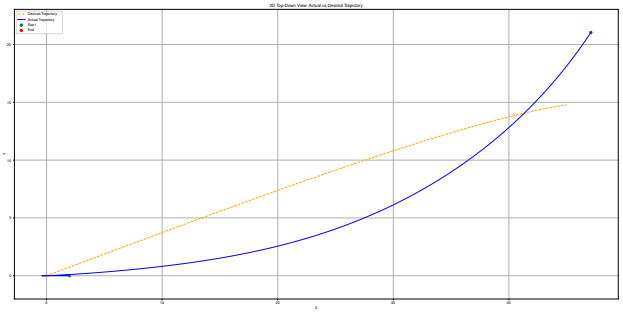

Fig. 4: Trajectory Tracking Using PID Controller

1) **Accuracy and Tracking Performance**

- **Tracking Error:**
  - Mean Squared Error (MSE) or Root Mean Squared Error (RMSE) between desired and actual trajectory.
  - Maximum Error (e.g., overshoot, peak deviation).
  - Final Position Error (steady-state error).
  - Integral of Squared Error)
  - Integral of Absolute Error
  - Integral of Time-weighted Squared Error
  - Integral of Time-weighted Absolute Error
- **Time to Reach Target:** Time taken to converge to within a tolerance of the target state or trajectory.

Metric	Definition	Cascaded	PID
ISE	$\int_0^\infty e(t)^2\,dt$	32242.4039	29157.9052
IAE	$\int_0^\infty \lvert e(t)\rvert\,dt$	1973.2149	1858.1065
ITSE	$\int_0^\infty t\cdot e(t)^2\,dt$	2433310.4169	2119309.2168
ITAE	$\int_0^\infty t\cdot \lvert e(t)\rvert\,dt$	153288.3224	139126.2826
RMSE	$\sqrt{\frac{1}{T}\int_0^T e^2(t)\,dt}$	14.2135	13.9290

TABLE III: Integral Performance Criteria for Tracking Error

IV. CONCLUSIONS

–

REFERENCES

[1] Bashir, A.; Khan, S.; Iqbal, N.; Bashmal, S.; Ullah, S.; Fayyaz; Usman, M. A Review of the Various Control Algorithms for Trajectory Control of Unmanned Underwater Vehicles. Sustainability 2023, 15, 14691. https://doi.org/10.3390/su152014691

[2] R. Makam, P. Mane, S. Sundaram, and P. B. Sujit, "A Comprehensive Study on Modelling and Control of Autonomous Underwater Vehicle," arXiv preprint arXiv:2312.02690, 2024. [Online]. Available: https://arxiv.org/abs/2312.02690

[3] L. V. Steenson, L. Wang, A. B. Phillips, S. R. Turnock, M. E. Furlong, and E. Rogers, "Experimentally Verified Depth Regulation for AUVs Using Constrained Model Predictive Control," *IFAC Proc. Volumes*, vol. 47, no. 3, pp. 11974–11979, 2014, doi: 10.3182/20140824-6-ZA-1003.01497.

[4] Vadapalli, S.; Mahapatra, S. 3D Path Following Control of an Autonomous Underwater Robotic Vehicle Using Backstepping Approach Based Robust State Feedback Optimal Control Law. J. Mar. Sci. Eng. 2023, 11, 277. https://doi.org/10.3390/jmse11020277

[5] H. N. Esfahani, B. Aminian, E. I. Grøtli, and S. Gros, "Backstepping-based Integral Sliding Mode Control with Time Delay Estimation for Autonomous Underwater Vehicles," arXiv preprint arXiv:2111.10179, 2021. [Online]. Available: https://arxiv.org/abs/2111.10179

[6] Zhou J, Zhao X, Feng Z, Wu D. Trajectory tracking sliding mode control for underactuated autonomous underwater vehicles with time delays. International Journal of Advanced Robotic Systems. 2020;17(3). doi:10.1177/1729881420916276

[7] Vu, Q.V.; Dinh, T.A.; Nguyen, T.V.; Tran, H.V.; Le, H.X.; Pham, H.V.; Kim, T.D.; Nguyen, L. An Adaptive Hierarchical Sliding Mode Controller for Autonomous Underwater Vehicles. Electronics 2021, 10, 2316. https://doi.org/10.3390/electronics10182316

[8] Liang X, Wan L, Blake JIR, Shenoi RA, Townsend N. Path following of an Underactuated AUV Based on Fuzzy Backstepping Sliding Mode Control. International Journal of Advanced Robotic Systems. 2016;13(3). doi:10.5772/64065

[9] T.I.Fossen, "Maneuvering Theory," in Handbook of Marine Craft Hydrodynamics and Motion Control, Ch. 6, pp. 109–132, John Wiley Sons, Ltd, 2011. [Online]. Available: https://doi.org/10.1002/9781119994138.ch6

[10] , "MBARI Tethys LRAUV," Open Robotics, Apr. 2, 2021. [Online]. Available: https://fuel.gazebosim.org/1.0/accurrent/models/MBARI