

Autonomous Depth Control for the MuBot: PID Regulation of the Buoyancy Control Device

Arnab Gupta

Department of Mechanical Engineering
Pennsylvania State University
Email: apg5799@psu.edu

Micah DeLattre

Department of Mechanical Engineering
Pennsylvania State University
Email: mwd5376@psu.edu

Abstract—This report presents a finalized implementation of Proportional-Integral-Derivative (PID) control applied to an adaptive buoyancy control device (BCD) on a bio-inspired robotic swimmer, the MuBot. This PID implementation addresses challenges such as system dynamics modeling, trajectory tracking, and disturbance rejection by optimizing control inputs to track a reference depth trajectory. Simulation results demonstrate the controller's effectiveness in achieving precise trajectory tracking while maintaining robustness against potential underwater disturbances.

I. INTRODUCTION AND LITERATURE REVIEW

A. Motivation and Background

In this project, Proportional-Integral-Derivative (PID) control is applied to a bio-inspired robotic swimmer, the MuBot, equipped with an adaptive buoyancy control device (BCD). The buoyancy of the MuBot will be regulated by adjusting the dynamically changing volume of the BCD. This will allow the robot to follow a specified reference depth trajectory. The key goals of the project are to achieve precise tracking of a reference trajectory within a simulated underwater environment, ensure robust responses to external disturbances, and successfully validate the control strategy through simulation. The challenges to be addressed include accurately modeling the system dynamics, developing a complete Simulink simulation with PID-controlled closed-loop feedback, and ensuring stability despite underwater and pressure sensor disturbances. Additionally, tuning the PID parameters to optimize trajectory tracking and robust disturbance rejection is explored.

B. Literature Review

PID control is still used as one of the most dominant methods for underwater robotic system regulation as it is simple, easy to program, and provides enough control performance in most working conditions. Although a classical control architecture, PID has shown its relevance in underwater environments where the dynamics are nonlinear and the model is uncertain.

In the work by Ajmal et al., they proposed PID for velocity control of an AUV (autonomous underwater vehicle) tracking 3-D trajectories in the ocean by rejecting disturbances [1]. Similarly, Hernández Beleño et al. presented a software platform for a Hardware-in-the-loop (HIL) simulation based on a dynamic underwater robotic manipulator model. Their

approach enabled the identification of the real-time system and demonstrated the real-world effectiveness of PID controllers in handling underwater dynamics [2].

Zhao and Yuh developed a PID autopilot for an AUV rejecting real-time ocean current disturbances. They argue that PID can reliably maintain depth and heading amid environmental changes [3].

Additionally, Kuo et al. considered a gain-scheduled PID method for depth control in an underwater glider. Their strategy updated the controller parameters depending on the glider velocity, which they argue is significantly better than a fixed-gain PID control [4].

There have been several extensions to the PID control setup. According to Hu et al., their fuzzy-PID method improves system responsiveness and robustness against external perturbations compared to traditional PID control for AUV depth control [5]. Maalouf et al. also conducted real-time experimental comparisons among various real-time depth control schemes, reaffirming the viability of PID-based control strategies, especially when used in hardware and real-time computational constraints [6].

Akkizidis et al. designed a fuzzy-like PD controller that would be expected to provide an improvement in adaptability while preserving PID simplicity. Their results showed that such PD-like controllers could be tuned to an acceptable level of tracking and disturbance rejection for underwater environments [7]. Finally, we must mention the robust PD control scheme developed by Suarez et al. for depth regulation of an underwater vehicle, achieving reliable performance in real time in experimental trials [8].

For this project, PID control will be implemented for regulation of the variable buoyancy control device of the MuBot, similar to methodologies developed by Ajmal et al. [1] and Zhao et al. [3]. The control system will attempt to accurately follow a reference depth trajectory while rejecting external disturbances.

PID control has proven to be successful in underwater robotics, mainly in real-time implementation with limited computational resources. In this scenario, considering real-world hardware limitations of the MuBot, ease of use, and straightforward tuning, PID emerges as a favorable practical option.

output $u(t)$ from the controller drives the BCD to adjust the MuBot's buoyancy, thereby controlling its depth.

C. Simulink Simulation Setup

As mentioned, the main objective in this control formulation is to enable the simulated MuBot to track a reference depth trajectory by adjusting the input voltage applied to the BCD system. The controller must also maintain stability while rejecting external disturbances.

The complete system dynamics and control architecture were modeled and simulated in Simulink. Figure 2 showcases the complete Simulink block diagram.

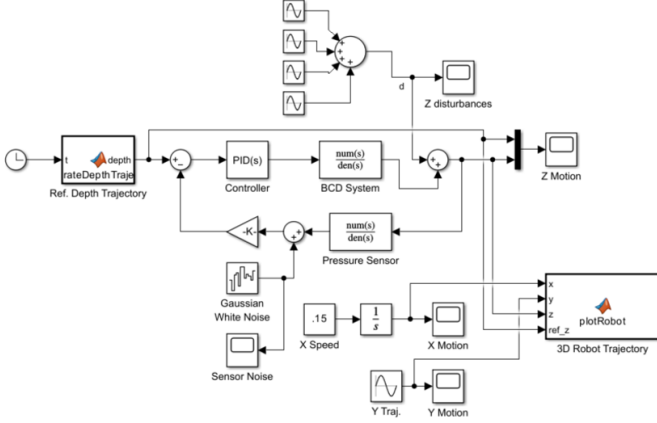


Fig. 2. Simulink Block Diagram for MuBot Depth Control

Referencing the above figure, a reference depth is fed to the PID controller, which then calculates the desired control input. This control input then goes into the BCD system plant model, which actuates the MuBot BCD system, changing the entire robot's depth in the simulated underwater environment. The robot's actual depth is then measured by a simulated pressure sensor and fed back to the PID controller, where the control input is computed based on the depth error. This effectively closes the feedback control loop. Notice that there are also disturbances applied to the robot's depth and pressure sensor measurements; these will be covered in detail in following subsections.

In addition, some predefined, uncoupled motions in the longitudinal and lateral directions enabled the simulated robot to follow a 3D trajectory in the underwater environment. For this work, these had no effect on the depth control dynamics, which is likely unnatural in the real world. Finally, the 3D simulated robot trajectory was plotted to visualize its motion.

1) *Reference Depth Trajectory*: A random reference depth trajectory is generated at the start of each simulation to mimic realistic operational conditions for the MuBot in a dynamic underwater environment. The trajectory is defined over a total simulation time of 20 seconds, with random level depth changes occurring at specified intervals and constrained between 0.35 and -0.35m. The randomized reference trajectory allows evaluation of the controller's effectiveness in tracking depth changes while rejecting disturbances.

2) *External and Sensor Disturbances*: To evaluate the robustness of the depth controller, simulated underwater disturbances were created by combining four different sinusoidal components with randomized frequencies, phase shifts, and amplitudes. These underwater disturbances aim to mimic, and are inspired by, realistic, dynamic wave and current disturbances the MuBot might encounter in a natural ecological environment. This resultant disturbance signal was fed into the output signal of the plant model in the Simulink simulation. Additionally, a simulated pressure sensor was added to the Simulink architecture to provide depth feedback for the simulated MuBot. To mimic the behavior of real-life sensors, low-amplitude, high-frequency Gaussian white noise was added to the pressure sensor system signal output.

Figures 3 and 4, below, are examples of external and sensor disturbance signals present in the simulation.

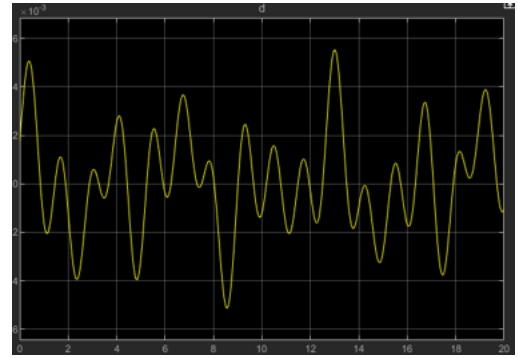


Fig. 3. External Underwater Current Disturbances

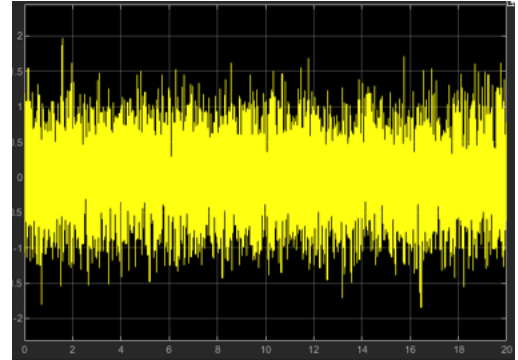


Fig. 4. Simulated Pressure Sensor White Noise Disturbance Signal

III. SIMULATION RESULTS

A. Simulation Examples

Presented here are two different simulation examples of MuBot depth control.

The first example is the MuBot responding to a reference trajectory with 2 high-amplitude step changes. The second example includes a reference trajectory with 3 lower amplitude step changes. Both simulation examples last 20 seconds and include disturbances.

Example 1: Figures 5 and 6 showcase the robot's 3D tracking of the reference trajectory and the robot depth vs. the target depth trajectory over time, respectively.

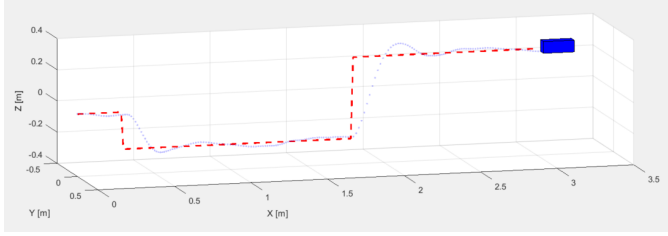


Fig. 5. Example 1 3D Robot Trajectory

As seen, the robot performs relatively well when tracking the reference trajectory. At time = 2 seconds, the reference depth steps down to approximately -0.23m. The MuBot is then driven downwards towards this reference and reaches it less than two seconds later. But then it overshoots to almost -0.3m at $t=4$ seconds. The controller adjusts, and the robot reapproaches the reference at around $t=6$ seconds. The robot mostly settles at the reference but slightly oscillates due to constant disturbance rejection.

At time = 12 seconds, the reference depth steps up to approximately 0.3m. The MuBot is driven upwards towards this reference and reaches it quickly once again. A larger overshoot is observed here, likely due to the large magnitude depth change. The controller adjusts for the overshoot, and the robot reapproaches the reference at around $t=16$ seconds. Again, the robot mostly settles at the reference, but some slight oscillation is observed.

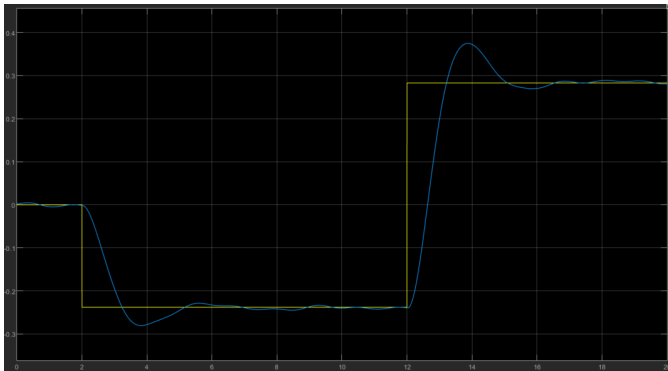


Fig. 6. Example 1 Robot Depth vs. Reference Depth Over Time

Example 2: Figures 7 and 8 showcase the robot's 3D trajectory and the robot depth vs. the target depth over time for the 3-step change example.

Again, in this example, the robot performs moderately well when tracking the reference trajectory.

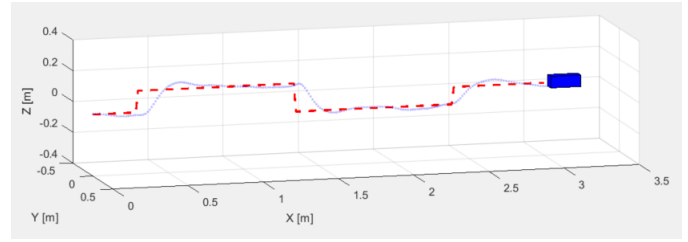


Fig. 7. Example 2 3D Robot Trajectory

At time = 2 seconds, the reference depth steps up to approximately 0.14m. The MuBot is then driven upwards towards this reference and reaches it less than two seconds later. But then it overshoots to almost 0.18m at $t=4$ seconds. The controller adjusts, and the robot reapproaches the reference at around $t=6$ seconds. Similarly, the robot mostly settles at the reference but slightly oscillates due to disturbance rejection.

At time = 9 seconds, the reference depth steps down to approximately -0.05m. The MuBot is driven downwards towards this reference, and again, a larger overshoot is observed here from the greater magnitude depth change. The controller mostly adjusts for the overshoot, and the robot begins to approach the reference at around $t=16$ seconds. Larger oscillation is observed here as the robot attempts to steady out.

Lastly, at $t = 16$ seconds, the reference depth steps up to approximately 0.07m. The MuBot rises upwards but overshoots slightly once again. The controller attempts to adjust for this overshoot, but due to the simulation ending at $t = 20$ seconds (and continuous disturbance rejection), the robot is not fully able to recover and steady at its reference depth.

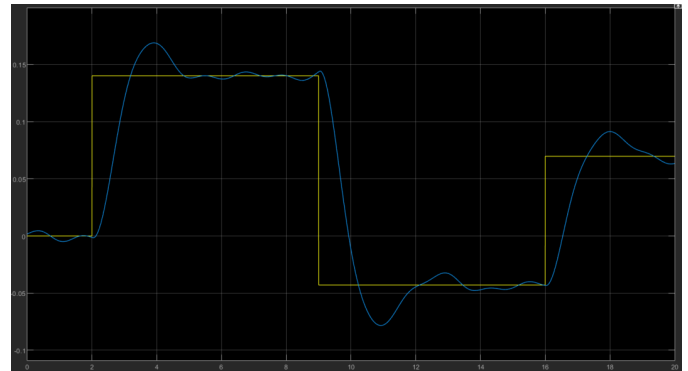


Fig. 8. Example 2 Robot Depth vs. Reference Depth Over Time

B. Key Ideas and Takeaways

The depth control results across both simulation examples demonstrate that the PID controller successfully regulates the MuBot's vertical position in response to changing reference depths, even in the presence of external disturbances.

In both plots, the robot achieves decent tracking performance with relatively fast rise times and stable settling behavior, indicating effective baseline controller tuning. However, the responses consistently exhibit overshoot and slight oscillations, particularly during abrupt reference transitions and disturbance

rejection. These behaviors are characteristic of an underdamped system and could be mitigated through further tuning of the proportional, integral, and derivative gains. Specifically, refining the derivative action may help suppress overshoot and dampen oscillations, while appropriate adjustment of the proportional gain could enhance responsiveness without introducing instability.

Another notable limitation is the lack of predictive behavior inherent to PID control. The robot consistently lags slightly behind the reference depth changes, as the controller reacts only to current and past errors rather than anticipating future system behaviors.

Despite these limitations, the PID controller provides reliable closed-loop performance and is an effective foundation for baseline depth control in realistic underwater environments.

IV. CONCLUSIONS

This report presents the autonomous depth control simulation of the MuBot, a bio-inspired robotic swimmer equipped with an adaptive buoyancy control device regulated by PID control. The modeling of the MuBot and BCD system was covered in detail, along with the controller definition and Simulink simulation setup. Lastly, simulation example results showcase the controller's effectiveness in tracking reference depth trajectories even in the presence of disturbances.

While the MuBot tracked reference depth trajectories effectively, there were response issues, as mentioned, such as overshoot, oscillatory behavior, and poor anticipatory action. While this formulation was largely successful, potential adjustments of system and PID controller parameters could lead to more promising results. Alternative control strategies could also be considered rather than this baseline PID controller for future applications.

A. Future Work

Potential future work will focus on tuning PID controller parameters to address previously stated limitations. As mentioned, other control strategies could be considered, such as Model Predictive Control (MPC) or Robust control. MPC specializes in anticipatory action control while adhering to system and state operational constraints. Robust control specializes in robust disturbance rejection and handling unmodeled or uncertain system dynamics. These control strategies would likely improve simulated depth tracking performance.

Additionally, this simulation can be expanded to properly incorporate the forward swimming dynamics of the MuBot, coupling x, y, and z motion. Likely, the lateral and longitudinal motion of the MuBot will affect its depth control capabilities. Lastly, we hope to replicate these simulation examples in real life with the physical MuBot. Similar to this work, a PID controller could be implemented to evaluate the real-world applicability and feasibility of MuBot autonomous depth control with pressure sensor feedback. If this sim-to-real transfer proves successful, this paves the way for future, possibly ecologically immersive applications of the MuBot.

REFERENCES

- [1] M. S. Ajmal, M. Labeeb, and D. V. Dev, "Fractional Order PID Controller for Depth Control of Autonomous Underwater Vehicle Using Frequency Response Shaping Approach," *2014 Annual International Conference on Emerging Research Areas: Magnetics, Machines and Drives (AICERA/iCMMD)*, vol. 1, pp. 1–6, Kottayam, India, 2014. doi:10.1109/AICERA.2014.6908219.
- [2] R. D. Hernández Beleño, P. A. Mora Gonzales, O. F. Avilés Sánchez, and J. Vaqueiro Ferreira, "Dynamic Modeling and PID Control of an Underwater Robot Based on the Hardware-in-the-Loop Method," *International Review of Mechanical Engineering (IREME)*, vol. 10, no. 7, pp. 9037, 2016. doi:10.15866/ireme.v10i7.9037.
- [3] S. Zhao and J. Yuh, "Experimental Study on Advanced Underwater Robot Control," *IEEE Transactions on Robotics*, vol. 21, no. 4, pp. 695–703, 2005. doi:10.1109/TRO.2005.844682.
- [4] C.-L. Kuo, C.-K. Tsui, N.-S. Pai, C.-H. Lin, S.-C. Chen, and P.-W. Li, "A PID Controller for the Underwater Robot Station-Keeping," *2016 IEEE 14th International Conference on Industrial Informatics (INDIN)*, vol. 1, pp. 1242–1246, Poitiers, France, 2016. doi:10.1109/INDIN.2016.7819357.
- [5] B. Hu, H. Tian, J. Qian, G. Xie, L. Mo, and S. Zhang, "A Fuzzy-PID Method to Improve the Depth Control of AUV," *2013 IEEE International Conference on Mechatronics and Automation*, vol. 1, pp. 1528–1533, Takamatsu, Japan, 2013. doi:10.1109/ICMA.2013.6618141.
- [6] D. Maalouf, V. Creuze, A. Chemori, et al., "Real-Time Experimental Comparison of Two Depth Control Schemes for Underwater Vehicles," *International Journal of Advanced Robotic Systems*, vol. 12, no. 2, 2015. doi:10.5772/59185.
- [7] I. S. Akkizidis, G. N. Roberts, P. Ridao, and J. Battle, "Designing a Fuzzy-like PD Controller for an Underwater Robot," *Control Engineering Practice*, vol. 11, no. 4, pp. 471–480, 2003. doi:10.1016/S0967-0661(02)00055-2.
- [8] A. E. Z. Suarez, A. M. Magallanes, M. A. G. Rangel, R. L. Leal, S. S. Cruz, and F. M. Palacios, "Depth Control of an Underwater Vehicle Using Robust PD Controller: Real-Time Experiments," *2018 IEEE/OES Autonomous Underwater Vehicle Workshop (AUV)*, vol. 1, pp. 1–6, Porto, Portugal, 2018. doi:10.1109/AUV.2018.8729783.
- [9] D. Aborisade, "DC Motor with Load Coupled by Gears Speed Control using Modified Ziegler-Nichols Based PID Tunings," *Control Theory and Informatics*, vol. 4, no. 5, 2014.

APPENDIX A

Parameter	Value [unit]
R_a , motor resistance	13.5 [Ω]
L_a , motor inductance	0.121 [H]
J_m , motor torque inertial	2e-5 [$kg - m^2$]
B_m , motor coefficient of friction	2e-4 [$N - m - sec/rad$]
J_L , inertial momentum of load	5e-5 [$kg - m^2$]
B_L , viscous friction of load	1e-4 [$N - m - sec/rad$]
K_a , motor constant	0.2 [$N - m/amp$]
K_b , back emf constant	0.2 [$V - sec/rad$]
$n = n_1/n_2$, gear ratio (motor/load)	3/7
c_d , drag coefficient of robot	0.2
m , mass of robot	78 [g]
r , radius of BCD dynamic volume	7 [mm]
V_f , fixed volume of robot	1e5 [mm^3]
A , Cross-sectional area of robot	45 [cm^2]
ρ_w , Density of water	1000 [kg/m^3]
p , pitch of lead screw	0.4 [mm]
g , gravity acceleration	9.81 [m/s^2]

TABLE I
ROBOT AND BCD PARAMETERS