Designing and Simulating an Underwater Exploration Robot Project Report

Tharumal Dinuranga
Department of Electrical Engineering
University of Moratuwa
Colombo, Sri Lanka
senevirathnekdytd.21@uom.lk

Dishara Senesh
Department of Electrical Engineering
University of Moratuwa
Colombo, Sri Lanka
seneshwad.21@uom.lk

Shapthamuky Jeevakaran
Department of Electrical Engineering
University of Moratuwa
Colombo, Sri Lanka
shapthamukyj.21@uom.lk

***Abstract*—The purpose of this project is to design a moving mechanism of a human controlled underwater robot with the ability to capture the surroundings and avoid obstacles. The robot will be equipped with a remote control, to be directed by the human according to the requiring movements and obstacles. The robot's behavior is determined by a set of instructions that gives the information about the direction of movements and turns. This project will demonstrate the feasibility of designing a moving mechanism of a human controlled robot that can navigate through a track directed remotely from the outside while capturing the surrounded environmental images and videos for research and other purposes. In this project, the robot is developed mainly under three mechanisms for different types of movements such as Mechanism of vertical movements, Mechanism of forward and backward movements and Mechanism of turning movements. For every mechanism, Design techniques and operation methods, and MATLAB codes and Simulations for the processes are included in this report**

***Index Terms*—Underwater robotics, Depth control system, Propulsion system, Turning mechanism**

I. Introduction

Numerous research and development activities on underwater robotics have been conducted worldwide from the period of 1990s. The need and use of underwater robotics system has become more obvious as the ocean gets a great attention on environmental issues and resources for scientific as well as military purposes. As a result, the creation of underwater robots, also known as Autonomous Underwater Vehicles (AUVs), has gained a significant attention in both robotics and underwater developments fields. Also, the idea of underwater robots has sparked the creation of many modern machines that can do difficult activities in dangerous or difficult-to-reach locations. In most areas like scientific discoveries, medical and industrial researches, resource exploration and environmental monitoring, plants, animals, mineral resources and many other things from the marine world have to be searched and collected for samples for the research and development activities. In early years, these exploration tasks were executed with human presence but in present situation, such tasks are widely proceeded by human controlled robots in areas where human presence is risky and impossible such as tight caves, deep oceans or the area where dangerous living beings live. Their ability to navigate and operate in challenging underwater environment, often characterized by low visibility, strong currents, and complex underwater terrain, necessitates the development of robust and efficient moving mechanisms.

Moreover, the cost for proceeding underwater researches with human interferences was a big challenge as we had an additional expenditure for researchers' safety measures, which is now sorted out with the robots.

Design of effective moving mechanisms for underwater robots is a complex and multifaced endeavor, requiring careful consideration of the robot's intended mission, hydrodynamic constraints, and the interplay between various propulsion and control systems. A well-designed moving mechanism should not only enable precise and maneuverable movement but also optimize energy consumption, minimize noise generation and withstand the harsh underwater conditions.

As a group of undergraduate students, we are proposing a design of a moving mechanism of such an underwater robot with a water-filling mechanism, as a main goal of this project.

A. Literature Review

Several amphibious robots are now commercially available which use different techniques to move around on land and in water. Many types of robots such as quadruped robots, snake robots and bipedal robots have been designed with the inspiration from animal morphologies. Commonly the moving mechanism in such robots are considered as three mechanisms for vertical movements, horizontal movements and turning movements.

The vertical movement mechanism, also known as depth control system, enables precise control of vertical position of the robot for ascending, descending or maintaining a specific depth. For this mechanism, buoyancy control, thrusters or propellers and variable-pitch propellers methods are mainly considered. In 2021, Zhang et al. proposed a buoyancy control

Identify applicable funding agency here. If none, delete this.

system using combination of a ballast tank and a high-precision pressure sensor for precise depth control of an underwater robot. This system demonstrated an effective depth regulation with minimal oscillations and energy consumption. As a continuation of that, Xiang et al. (2022) developed a depth control system using thrusters and a fuzzy control algorithm for an underwater robot. As a result, this system achieved an accurate depth tracking and robust performance under various disturbances. Then wang et al. (2023) investigated the use of variable-pitch propellers for depth control of an underwater robot. This investigation demonstrated the feasibility of this approach and its potential for improved efficiency and maneuverability.

The forward and backward movement mechanism which is part of horizontal movement mechanism generates thrust to propel the robot in the desired direction. This mechanism is also known as propulsion system and commonly it includes propellers, thrusters and water jets. For the development of this mechanism Liu et al. (2021) developed a novel thruster design for underwater robots, that is utilizing a magnetically levitated impeller for reduced noise and improved efficiency. Moreover, Li et al. (2022) investigated the use of water jets for propulsion of underwater robots which is exploring the trade-offs between thrust, efficiency and noise reduction.

The turning mechanism enables the robot to change its orientation into the water for navigating around obstacles, changing direction and following specific paths. Generally, this mechanism includes rudder or steering vanes, differential thrust and independent propellers or thrusters. In 2020, Zou et al. proposed a rudder design for an underwater robot, optimizing its shape and placement for improved maneuverability and reduced resistance. Chen et al. (2021) developed a differential thrust control algorithm, that enables precise turning maneuvers and enhanced stability. For further improvement, Sun et al. (2022) investigated the use of independent propellers or thrusters for turning control of the robots, achieving agile and versatile turning capabilities.

Despite these advancements, challenges remain in optimizing energy efficiency for longer mission durations, minimizing noise generation for environmental considerations and designing mechanisms adaptable to diverse tasks and environments. Future research efforts should focus on addressing these challenges wile incorporating advancements in control systems and materials to propel AUV capabilities and performance to new heights.

B. Project Statement

This project focuses on the design and simulation of an underwater exploration robot, resembling a drone, with manual control through a dedicated panel. The primary aim is to streamline and enhance underwater exploration by providing users with a tool for precise navigation and operation. The control panel facilitates human-machine interaction, ensuring adaptability to different underwater environments. Simulation techniques are employed to analyze the robot's performance in various scenarios, optimizing its design for maximum efficiency in the field of underwater exploration. This project marks a significant advancement in the field of underwater robotics, offering a versatile solution for scientific, environmental and industrial applications.

C. Background Study

Precise movement and control are critical for underwater robotic vehicles to carry out useful exploration and data collection missions. As noted by Smith et al. (2021), underwater environments lack reliable GPS signals, meaning these robots must rely on inertial guidance systems and manual control inputs to maneuver. Even minor errors in rotation or translation can cause the robot to veer off course. Additionally, complex underwater terrain, currents and low visibility conditions present significant navigation and positioning challenges(Wang et al., 2022)

Therefore, developing accurate and responsive control mechanism is an essential requirement for underwater robotics. The thruster configurations and control algorithms must translate manual inputs into smooth, proportional movements in all directions in order to maintain a desired depth and heading (Wernli, 2020). Prior researchers have indicated that simulation platforms are highly useful tools for designing and refining these control parameters before deploying underwater vehicles in real marine environments (Nicholson and Healey, 2008).

In parallel, quadcopters face similar challenges with stabilization, manual control responsiveness, and avoiding deviations from the flight path intended by the user. As outlined by Luukkonen (2011), quadcopter movement is controlled by precisely varying the thrust and torque generated by each of its four rotors. This project leveraged experience gained in modeling and tuning control systems for quadcopter simulations. The knowledge was applied to develop proportional equations and parameters for controlling the underwater robot's vertical thrusters and rotational propellers.

These control schemes were implemented and refined in a 3D simulation environment to exhibit smooth, accurate manual control of movement along all axes prior to potential future implementation on real hardware. The capability for precise control will be necessary for this underwater robot to be viable for tasks such as photographic surveying, sensor data collection, and close inspection of submerged structures.

II. Design and Drafting

The underwater exploration robot developed in this project resembles a sleek, modern drone design with six propellers enabling maneuverablity along multiple axes. Prior to drafting in SolidWorks, initial sketches were made to conceptualize a quadcopter-like form factor with four vertically oriented propellers supplementing two forward-facing, horizontally oriented thrusters.

This configuration enables both rotation around the z-axis from the upper propellers that function similar to a quadcopter, along with horizontal thrust generation from the front-facing rotors. The symmetrical, centered positioning of all six rotors grants stability and balanced control response for manual navigation in close proximity to submerged objects or structures.

Following hand sketches to illustrate the basic desired framework, the full 3D model was developed using SolidWorks CAD software. Precise dimensions were applied to design the body and rotor diameters to appropriate scales for an unmanned underwater vehicle intended for activities such as photographic surveying or sensor data collection missions. The completed SolidWorks model realistically represents the envisioned drone shape wit all six rotors placed at appropriate positions based on the initial sketch concepts.

This 3D model was then imported to a physics-based simulation program to develop and tune control systems that translate manual inputs into smooth, proportional movement of the underwater robot in all directions. The detailed drafting in SolidWorks was an essential foundation for proceeding to behavioral implementation and testing in the simulation environment, evaluating capability for robust control and stable hovering or navigation as intended for the real-world application.

A. Mechanical Design

During the designing and modeling process, several factors were taken into account like functionality, defining axes, mass distribution, etc. The mechanical layout focused on achieving balanced weight placement and positioning six rotors to enable smooth manual control of movement along all axes. Symmetry played a key role in allowing analogous thrust generation and torque response on both sides for straightforward pilot input translation.

The four vertically oriented propellers draw inspiration from quadcopter configurations, as this grants established flight stability and control. Building upon this foundation, the addition of two forward rotors supplements lateral and frontal motion capacity. Distributing the propulsive elements across three axes tailors the design specifically for the marine environment, where movement is not constrained a single plane. Propeller sizing and weight allowances left room for embedding necessary electronics within the waterproof fuselage in the future.

- Functionality
  The primary functional elements enabling mobility are the six propulsion rotors, with four vertically-mounted units and two forward facing thrusters. All utilize concentric bases allowing free rotation along the central axis to provide thrust in both directions. This supports reversing movements motions as needed for fine maneuvering. A small integrated camera module centered on the front supplies first-person visualization without impacting hydrodynamic form. While currently non-functional in simulation, this component provides for future remote control and navigation to survey underwater environments.
- Axes
  The coordinate system and rotation axes were clearly defined to translate propeller thrusts into controlled directional motions. For the main chassis, the z-axis runs vertically with positive upward, x-axis points forward, and y-axis completes the right-handed triad to the side. The four upper propellers adopt the same body-frame axes to induce torques for rotation around z. The forward rotors utilize a separate frame with z aligned along the front-facing thrust direction. This alternative alignment allows those units to generate lateral forces rather than axial torque effects. Overall the suite of six propellers focus thrust production along the local z-directions, while strategic placement converts these spinning forces into full 3-axis control authority. Defining these coordinated body-fixed and propulsion reference frames constituted a key portion of the design process to support subsequent control implementation and simulated flight demonstration.

B. Part 1 (Robot Body)

The foundation of the underwater robot is the main body chassis shown in Figure 1. Extending upwards from this are two short, wide front propeller arms (Figure 2) and two longer, narrower back propeller arms (Figure 3). These attachments contain an enclosed slot and base to mount the vertical thrusters. For lateral thrust, separate side mount assemblies (Figure 4) are fixed to the bottom of the central hub. Each of these modular components – the main body, vertical rotor arms, and lateral rotor mounts – possess a streamlined, hydrodynamic shape. The propeller arm widths and lengths were optimized to retain maneuvering stability once fully assembled, as depicted in the final model (Figure 5). The centralized main body retains symmetry while distributing a variety propulsive sources, with the suites of vertical and horizontal rotors placed at strategic distances and orientations to balance motions along all axes. This multi-part assembly and the standalone components depicted in the figures constitute the full mechanical design of the underwater robot's dynamic framework.

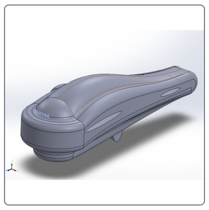


Fig.1 Main Body

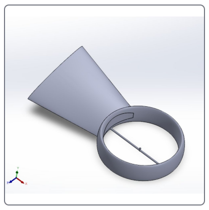


Fig.2 Front Propeller Arm

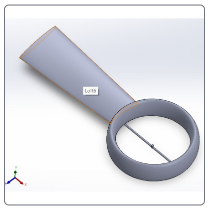


Fig.3 Back Propeller Arm

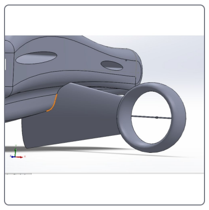


Fig.4 Down Propeller Arm

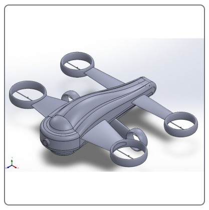


Fig.5 Assembled Robot Body

C. Part 2 (Front Propeller)

The front vertical thrusters utilize an eight-fin propeller configuration mounted inside the protective propeller arms. As depicted in Figure 6, the rotor assembly contains eight angled fins evenly distributed around and secured to a central hub. This arrangement maximizes thrust generation efficiency compared to simpler two or four blade designs. The direction and pitch of the fins helps counteract torque during spinning motion. The central hub component provides a robust attachment point to anchor the propeller within the slotted housing. It also enables low-friction rotation around the vertical axis when powered. The eight fin rotor presents a balance of stability, propulsive efficacy, and linkage durability within the confined front arm spaces to enable responsive manual thrust control. This complex geometry packed into a compact design typifies the propeller unit visible within the transparent housings of the forward vertical thruster assemblies.

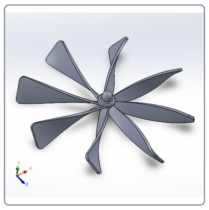


Fig.6 Front Propeller Design

D. Part 3 (Back Propeller)

The rear vertical thrusters employ a nearly identical propeller configuration as the forward units. This also consists of eight angled fins distributed evenly around a sturdy central hub, as shown in Figure 7. Maintaining this consistency in the rotor geometry between the front and back supports balanced thrust generation and torque response from both sets of vertical propellers. The rear propellers spin within an enclosed housing in the long, narrow propeller arms appended to the rear of the main body. The eight-fin design presents equivalent efficiency and stability benefits suited for the rear rotor mounts as for the frontal sections. This ensures smooth, proportional control of ascent, descent, and spinning movements through symmetrical thrust forces. The rear propellers exemplify the same robust construction and hub-anchored attachment to the protective surround intended to enable sustained operation in submerged conditions across all four vertical lift mechanisms.

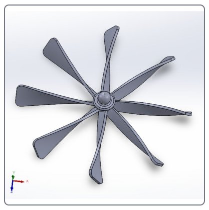


Fig.7 Back Propeller Design

E. Part 4 (Down Propeller)

The forward lateral thrusters utilize the same propeller design as the vertical units, adapted to a smaller scale. As shown previously in Figure 7, these also spin 8 angled fins distributed evenly around a central hub. The key difference from the vertical thrusters is the overall reduced diameter and aspect ratio. This condenses the proven rotor configuration to fit within the compact side mounts fixed below the main body. Despite the size decrease, the 8-fin layout maintains efficient thrust generation, balance, and torque resistance for responsive steering. The central hub continues to provide robust installation and low-friction spinning. So while more compact than the vertical thrusters, the forward rotors still produce ample uni-directional thrust for maneuvers by employing the same number of optimized angled fins. This miniature adaptation exemplifies right-sizing of the core propeller assembly to integrate horizontal thrusters facing the x-axis within the space limitations.

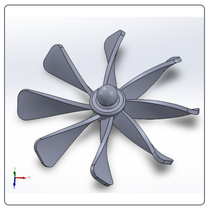


Fig.8 Down Propeller Design

F. Assembly

The complete assembly drawing culminates the full underwater robot design by integrating each modular component into a unified framework. As depicted in the assembled view (Figure 9), the central main body now supports two front and two rear vertical propeller arms. The lateral side-mounted thruster units tie below. Within each propeller cover housing, the motors fix concentrically to the bases, preventing axial misalignments while permitting intended rotational motions. This retains all six rotors in their respective orientations to generate balanced multi-axial thrusts and torques.

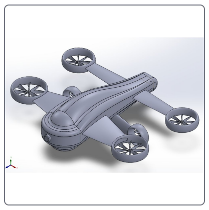


Fig.9 Assembled Robot

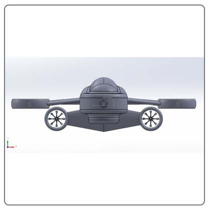


Fig.10 Front View

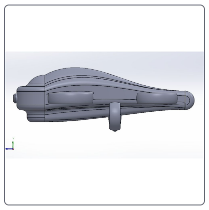


Fig.11 Side View

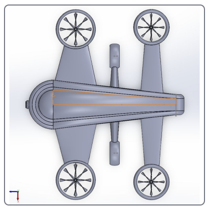


Fig.12 Top View

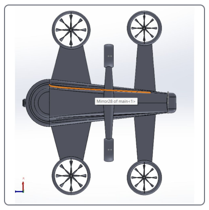


Fig.13 Bottom View

From a centered frontal view (Figure 10), the forward-facing camera component is visible mounted centrally among the protective housings and front crossed rotor arms. Meanwhile, the side angle (Figure 11) reveals the distanced rear arms and subtle contours of the tapered main body shape. The rearward perspective also shows the distributed stability granted by the landing legs on each corner. The top angle (Figure 12) looks directly down upon the upper propeller system layout in a visually balanced symmetric alignment. Finally, the bottom view (Figure 13) gazes up under the core body with lateral rotors protruding distinctly to supplement the vertical control mechanisms. Collectively, the figures showcase the model in full three-dimensional completeness. The integrated design retains aesthetic cohesiveness from all viewing angles, while positioning propulsive and stability components for simulated underwater mobility.

G. Surface Finished Robot

The final surface finished product (Figure 14) has a sleek and modern aesthetic with its black and blue color scheme. The central main body provides stability while the propeller arms branching out from it allow for maneuverability across six axes. The broader front two propeller arms contrast with the thinner rear arms to balance weight distribution. The additional forward-facing propellers generate x-axis rotation. All propellers sit securely in their own slots and spin smoothly within enclosed covers, protecting them as the robot explores underwater. The high-quality surface finishes and attention to ergonomic design result in an elegant underwater drone suited for exploration even in tight spaces or turbulent marine environments. Overall, the drone has both visual appeal and functional practicality with its aerodynamic form and multiple high-powered propulsion mechanisms.

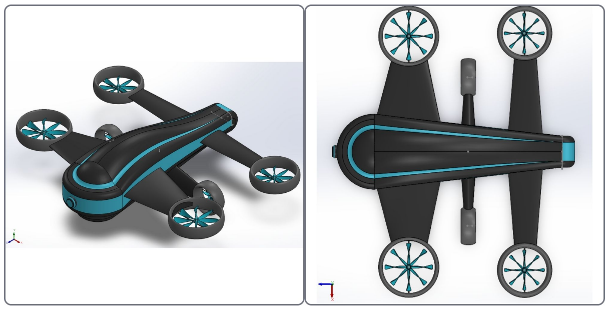


Fig.14 Final Surface Finished Robot

III. Simulation using Matlab Simulink

A. Background Research

Our goal was to simulate the dynamics of an underwater drone robot using MATLAB Simulink. We began by researching techniques for modeling basic quadcopters to understand their thrust mechanisms, inertia, and stabilization controllers.

Resources like MathWorks tutorials [1] and conference papers [2,3] provided great insights into quadcopter simulation fundamentals.

We then set out to expand beyond quadcopters to a more complex robot with six propellers enabling motion across six axes. Tuning the controllers for this kind of free range maneuverability was challenging. We developed custom thruster models so that the force inputs from each propeller would accurately reflect on the central body dynamics.

After many iterations, we could realistically simulate motions along the x, y, z axes as well as rotational motions on all axes. Both auto-stabilizing and manual control systems were tested. The self-balancing PID loops would correct the orientation against outside forces. The manual remote controls then allowed free navigation based on joystick and keyboard commands.

In the end, by building on basic quadcopter simulation techniques, we created a flexible underwater drone prototype in Simulink. The virtual modeling enabled rapid testing of dynamics and control approaches before committing to a physical build. This project was great practice for learning how to simulate complex robots through custom propulsion and inertial calculations.

Some of the sources that we have used during the research and learning process are mentioned below.

- https://www.mathworks.com/help/sl3d/simulate-a-quadcoptor.html
- https://www.mathworks.com/discovery/drone-simulation.html
- https://www.mathworks.com/content/dam/mathworks/mathworks-dot-com/company/events/conferences/matlab-virtual-conference/2015/proceedings/introduction-to-simulink-quadcopter-simulation-and-control.pdf

B. Simulation Mechanism

The robot simulation centered around independently defined thruster models to represent each of the six rotors - four vertical lift propellers arranged in a quadcopter layout and two forward-facing horizontal propellers. By customizing the voltage signals feeding into each thruster subsystem, precise control of rotor directionality and variable speed settings could be digitally sent simulating a range of manual remote control inputs.

For the quadcopter set, increasing or decreasing the net voltage difference between diagonal pairs generated asymmetrical thrust forces and torques along the z-axis and about the z-axis respectively. So for pure vertical heave, all rotors could be set to a positive value for upward movement; reversing the sign on all four signals would induce equivalent downward force. For twisting rotations on the yaw axis alone, the top left and bottom right thrusters would receive incrementally higher positive voltages than the opposite diagonal pair given slightly lower or negative signals instead.

The two horizontal propellers pointing forward along the x-axis were configured to spin in opposite directions always so their thrust forces directly canceled out. Then amplifying the voltage, and hence RPMs to the front prop while proportionally cutting back on the rear, successfully tilted the net thrust vector to a forward angle. ramping rear voltages higher than front would reverse the direction. So precise coordination of this thrust imbalance between front and back rotors enabled full acceleration control to surge backwards and forwards.

Through the customizable voltage patterns fed to all six rotors, full maneuverability along and about all axes was achieved to navigate any desired 3D motion trajectories without needing inertial physics calculations. The unmanned vehicle could be digitally piloted to climb, descend, pitch, roll, yaw, reverse, turn while moving, and more. The Simulink simulation thereby provided a virtual yet realistic testing environment for designs prior to fabrication.

C. Simulation Procedure

1) Importing the Assembled Robot into Simscape Multibody Environment
    a) Generating the Simscape Multibody link from the CAD model
    b) Importing into MATLAB Simulink and creating basic simulation blocks
2) Integrating a 6 DOF (Degrees of Freedom) Joint
    a) Defining voltage inputs to control manoeuvring in 6 axes
3) Activating the Propellers
    a) Integrating revolute joints for each propeller
    b) Adding control voltage sources
    c) Separating 4 vertical thrusters diagonally
4) Adding Internal Mechanics Parameters
    a) Calculating mass properties and buoyant forces
    b) Adding gravitational force and buoyant force
    c) Defining fluid damping coefficients
5) Simulating the 4 Vertical Propellers
    a) Simulating Up/Down Motion
        i) Setting default rotor speeds
        ii) Calculating thrusts for upward and downward directions
        iii) Determining voltage levels for heave
    b) Simulating Rotational Motion
        i) Setting propeller speeds for clockwise and counter-clockwise torques
        ii) Determining input levels for rotation
    c) Combining 2 Motions
        i) Calculating rotor speeds and Controlling 4 rotors simultaneously using functions
6) Simulating 2 Horizontal Propellers
    a) Forward/Backward Motion
        i) Setting thrust values in surge and sway
        ii) Calculating rotor speeds for longitudinal control
    b) Adding Rotational Maneuvers

i) Introducing angled rotation of horizontal thrust vectors

7) Creating 6-Axis Control Interface
   a) Enabling manual control of all degrees of freedom

1) Importing the Assembled Robot into Simscape Multibody Environment:

1) Generating the Simscape Multibody link from the CAD model

   The assembled 3D robot model designed in SolidWorks comprised of multiple part files for the central body, arms, propellers, and other components. This complete assembly was imported into Simulink in order to integrate a physics-based simulation using Simscape Multibody. A direct link was created between the CAD geometry and the mathematical model through a built-in conversion pipeline.
   The robot assembly's visual appearance, constraints, mass properties, and degrees of freedom mapped directly from SolidWorks into a Simulink block where the multibody dynamics calculations could be defined. This enabled our real-world physical robot prototype to have a virtual equivalent within the Simulink simulation environment, facilitating rapid testing in a modular, parametric model before committing to hardware iterations. The connected workflow from 3D model to 2D simulation thereby provided the flexibility to evaluate and refine the robot's maneuvering dynamics through subsequent iterations of both the virtual and physical constructs.

2) Importing into MATLAB Simulink and creating basic simulation blocks

   After converting the full three-dimensional SolidWorks assembly into a Simscape-compatible link, it was imported into the Simulink environment to set up the simulation framework. As illustrated in Figure 15, basic configuration blocks were created including a Joint Actuator block for commanding maneuvers, Rigid Transform sensor to output pose data, and 3D visualization to dynamically render movements. The foundational underwater drone robot subsystem encapsulated all necessary mechanical parameters translated from the CAD geometry. By extracting just the essential components into Simulink, the streamlined model facilitated rapid testing by exposing only the key control inputs and outputs for tuning the robot's thrust response and stability in the virtual world. Building up from this simplified base representation, additional capability blocks were incrementally incorporated while retaining an efficient workflow between complex modeling and dynamic simulation.

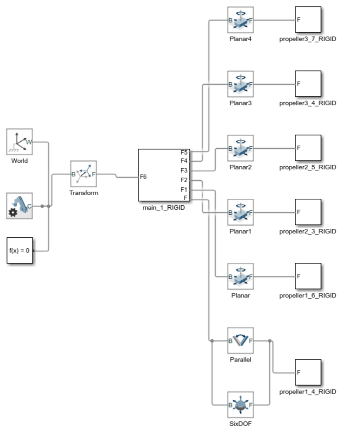


Fig.15 Basic underwater robot simulation blocks in Simulink

2) Integrating a 6 DOF (Degrees of Freedom) Joint:

1) Defining inputs to control manoeuvring in 6 axes

   A Six Degrees of Freedom (6 DOF) joint module connects the main body of the underwater drone allowing unconstrained translation on all X, Y, and Z Cartesian axes as well as angular rotation about each axis. This enables implementing commands for the robot to surge forward/back, sway left/right, heave up/down, pitch nose up/down, roll side to side, and yaw clockwise/counter-clockwise.
   By exposing control parameters on the 6 DOF joint, explicit force and moment inputs could be defined to induce the desired 6 dimensional motions:

   Px - x coordinate target setpoint
   Py - y coordinate target setpoint
   Pz - z coordinate target setpoint
   Fx - Translational force input for X axis
   Fy - Translational force input for Y axis
   Fz - Translational force input for Z axis
   vz - Vertical direction velocity input
   wz - Rotational velocity input for yaw axis

   Through managing this small set of intuitive navigation parameters on a single 6 DOF Driver block, our underwater robot simulation gained complete freedom to maneuver along any trajectories in all dimensions. The interactive inputs enabled testing complex navigation paradigms with changing position setpoints, external disturbances compensating forces, gently aligned orientation shifts, controlled depth changes and more.

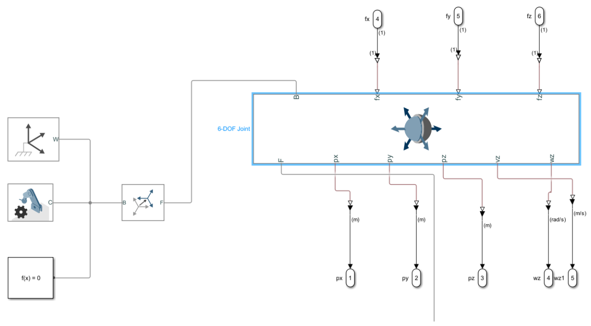


Fig.16 6-Direction of Freedom Joint

3) *Activating the Propellers:*

1) Integrating revolute joints for each propeller

In order to activate rotation of the six propellers, revolute joints were connected to each allowing free spin along one rotational axis. PID controllers were then implemented in a closed feedback loop to control the torque applied on each joint and achieve the commanded angular velocity.
Three propeller speed input parameters were defined as control setpoints for the PID blocks:

a) Front-facing propellers speed (front-back)
b) Diagonal right propellers speed(Right)
c) Diagonal left propellers speed(Left)

As shown in Figure 17, the four vertical thrusters were paired diagonally, so their respective PID controllers tracked the two diagonal speed inputs. The two horizontal propellers pointing forward and back had their revolute joints set up independently to follow the horizontal input as visualized in Figure 18.
By integrating revolute joints powered by PID control loops for all six rotors, full authority was established over the individual spin rate of each propeller. Their rotational maneuvers could be digitally activated and tuned.

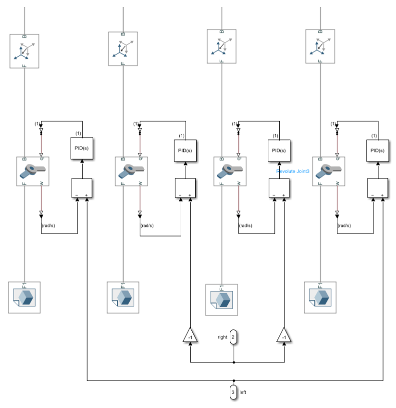


Fig.17 Simulink model of the 4 Vertical Propellers

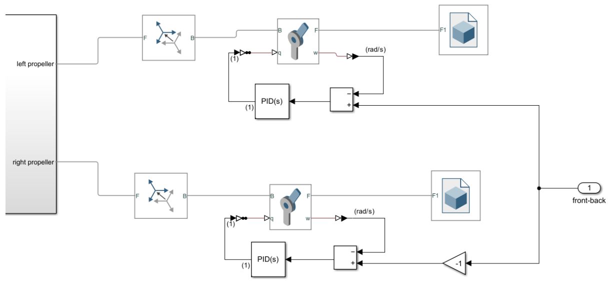


Fig.18 Simulink Model of the 2 Horizontal Propellers

2) Adding control voltage sources

Three key input parameters were created to control the rotation rates of the propellers:

a) Front-facing propellers speed
b) Diagonal right propellers speed
c) Diagonal left propellers speed

These reference values fed into the PID controllers associated with each revolute propeller joint. For the front horizontal thrusters, their PID loops tracked the "front-back" setpoint to determine the voltage sent to the motors. Likewise, the diagonal right and left rotor pairs were controlled in tandem, matching the corresponding "right" and "left" inputs.
By simply adjusting these three global speed variables, coordinated commands were propagated downstream to tune the individual RPMs of all six propellers simultaneously. This enabled intuitive digital control of the rotor mechanisms for thrust maneuvering through higher-level velocity setpoints rather than manipulating the internals of each motor subsystem independently.

3) Separating 4 vertical thrusters diagonally

The quadcopter configuration of the 4 vertical lift rotors was arranged with two thrusters occupying diagonal positions on opposite sides of the main robot body. As will be elaborated further in analyzing the rotation con-

trol principles, modeling the propeller pairs diagonally simplified the simulation of angular motions about the vertical axis.
By locking the two diagonal thrusters to operate in tandem at matched speeds, rotating the robot's orientation could be achieved through imbalance between the diagonal pairs. For example, to induce a clockwise yaw, the "left" pair speed input would be set greater than the "right" pair input, generating an unequal torque.
This diagonal grouping and paired speed control was key in coding efficient scripts to pivot the drone's orientation on the spot without inducing unnecessary horizontal or vertical forces. The modular separation made incrementing the counter-rotational torques straightforward through manipulating just two propeller speed variables rather than four unique rotor commands.
In summary, the diagonal configuration enabled simpler integration of rotational dynamics in the 6-DOF simulation while keeping generalized speed input control across all four vertical thrusters. As mentioned in the Figure 17 the same output is given to diagonally placed rotors through PID controllers to keep the rotor rotating throughout the input time.

4) *Adding Internal Mechanics Parameters:*

1) Calculating mass properties and buoyant forces

Properly accounting for the robot's mass and volume was critical in applying accurate gravity and buoyancy physics within the dynamic model. The total mass was computed by summing the contributions from the primary central body plus smaller additional masses from the propeller components:

Mass of body = 4.43261 kg
Mass of front propeller = 0.00214824 kg
Mass of back propeller = 0.00142175 kg
Mass of front-facing propeller = 0.000911896 kg
Total Mass = 4.43261 + 2(0.00214824 + 0.00142175 + 0.000911896) kg
= 4.441573772 kg

With a known uniform density of 900 kg/m3 set for the robot's materials, the volume was derived from the mass:

Volume = Mass / Density
= 4.441573772 kg / (900 kg/m3)

The gravitational force depended directly on the total mass, while buoyant force relied on overall robot volume fully submerged, as the vertical thrust needed to counterbalance.

W = mg
U = Volume * Density(water) * g

These forces are calculated inside matlab functions and integrated with the Robot's dynamics using the codes mentioned below.

Calculating Gravitational Force:
```
function y = fcn(u)
y = u*9.81;
```

Calculating Buoyancy Force:
```
function y = fcn(u)
y = -(u/900)*1000*9.81;
```

2) Adding gravitational force and buoyant force

After deriving the total robot mass and volume values, the gravitational and buoyant forces were computed through MATLAB math functions based on the standard equations:

W = mg
U = Volume * Density(water) * g

These force quantity equations were represented as variable blocks in Simulink, referencing the latest mass and volume parameters calculated for the robot. As shown in the model diagram Figure 19, the Gravity block output a 9x1 vector with 0 x/y forces and a downward z-axis force based on mass.
Likewise, the Buoyancy Force block generated 0 x/y forces and an upward z-axis force based on volume displacement. By wiring these vectors into the multibody dynamics, realistic vertical loads were imposed on the robot for sinking/rising responses.

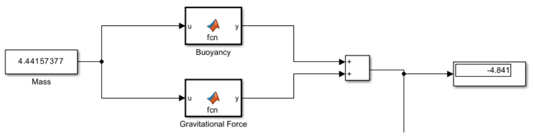
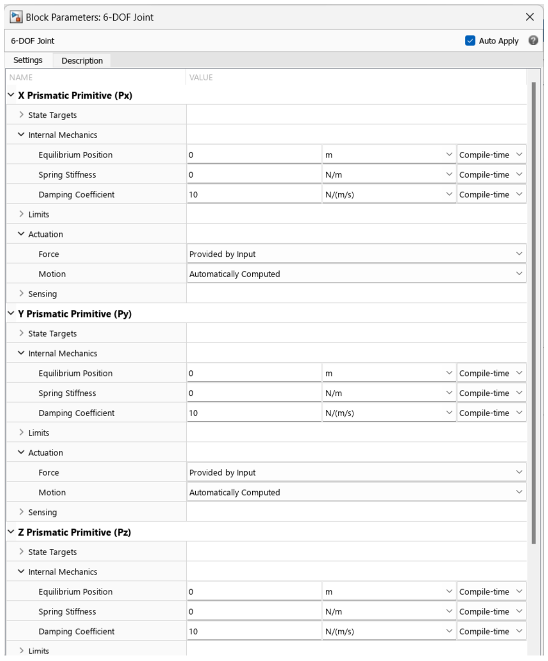


Fig.19 Simulink Model of adding Internal Mechanics to the system

3) Defining fluid damping coefficients

To account for the viscous fluid resistance and frictional forces acting on the robot as it maneuvers underwater, a damping coefficient was incorporated into the physics of the dynamic model. This represents the drag impacts from effects like shear stresses in the water that dissipate kinetic energy from components in motion.
A damping coefficient value of 10 Ns/m was empirically tuned and set uniformly across the x, y, and z axes. This means a linear damping force proportional to the robot's velocity along each axis resists motion in the direction of travel. The higher the intended speed, the greater the

counter force applied.
By coding an appropriate damping coefficient into the equations of motion, the real-world energy losses and flow disruptions caused by the underwater environment could be simulated. This allows testing control stability with more realistic external disturbances and loading effects on the multi-rotor design.


Block Parameters: 6-DOF Joint

6-DOF Joint — Auto Apply [✓]

Settings | Description

NAME	VALUE		
X Prismatic Primitive (Px)			
State Targets			
Internal Mechanics			
Equilibrium Position	0	m	Compile-time
Spring Stiffness	0	N/m	Compile-time
Damping Coefficient	10	N/(m/s)	Compile-time
Limits			
Actuation			
Force	Provided by Input		
Motion	Automatically Computed		
Sensing			
Y Prismatic Primitive (Py)			
State Targets			
Internal Mechanics			
Equilibrium Position	0	m	Compile-time
Spring Stiffness	0	N/m	Compile-time
Damping Coefficient	10	N/(m/s)	Compile-time
Limits			
Actuation			
Force	Provided by Input		
Motion	Automatically Computed		
Sensing			
Z Prismatic Primitive (Pz)			
State Targets			
Internal Mechanics			
Equilibrium Position	0	m	Compile-time
Spring Stiffness	0	N/m	Compile-time
Damping Coefficient	10	N/(m/s)	Compile-time
Limits			

Fig.20 Damping Internal Mechanics Applied to the Robot

5) *Simulating the 4 Vertical Propellers:*

1) Simulating Up/Down Motion
   a) Setting default rotor speeds

A key preliminary step before introducing vertical actuation was to determine the constant rotation rate for the four lift rotors that would generate adequate thrust to precisely counterbalance the net upward force from having the robot submerged.
The net vertical force was calculated by subtracting the gravitational pull down from the buoyant force up which depended on displaced fluid weight:

Total Upward Force
= Buoyancy Force - Gravitational Force
= 4.841 N

This 4.841 N lift must be matched by the total thrust output from the four rotors to avoid sinking or rising:

Thrust/Rotor =
Total Upward Force / 4 Rotors
= 4.841 N / 4
= 1.21025 N

Knowing that thrust and RPM have a squared relationship allowed back-calculating for the implicit thruster constant K:

F = K * V^2
$K = F/V^2$

Assuming a default speed of 500 RPM,

K = 1.21025 / (500 RPM)^2
$= 4.84417e - 6$

This process established the requisite idling speed and underlying thrust efficiency parameter needed before then adjusting RPMs to controllably ascend or descend.

   b) Calculating thrusts for upward and downward directions

After defining the base equilibrium speed, the next task was formulating rotor velocities for controlled heaving manuevers. As shown in the subsystem of Figure 21, directional thrust commands were created using switches, applying either +1N upwards force or -1N downwards.
When this force input is enabled, integrating the robot dynamics yields a net force imbalance, causing acceleration along the z-axis. To consciously induce these intentional disturbances, the propeller speeds must be adjusted to overpower the default thrust.
Knowing total lift/weight differentials, the additional/reduced thrust per rotor was derived by dividing by 4 blades:

Upward Total = 5.841 N
Upward per Rotor = 5.841 N / 4 = 1.46025 N
Downward Total = 3.841 N
Downward per Rotor = 3.841 N / 4 = 0.96025 N

Then using the established model:

F = KV^2

The upward and downward speeds were calculated per rotor, coded into a Matlab function for live control signal output to all four thrusters simultaneously, proportionally tuning velocity to generate target thrust forces.

Code for calculating Rotor speeds for Up/Down Motion

```
function [left_p,right_p]= fcn(ud)
  K = (4.841/4)/((500)^2);
  if ud == 1
    left_p = sqrt((5.841/4)/K);
    right_p = sqrt((5.841/4)/K);
  elseif ud == -1
    left_p = sqrt((3.841/4)/K);
    right_p = sqrt((3.841/4)/K);
  else
```

```
        left_p=500;
        right_p=500;
    end
end
```

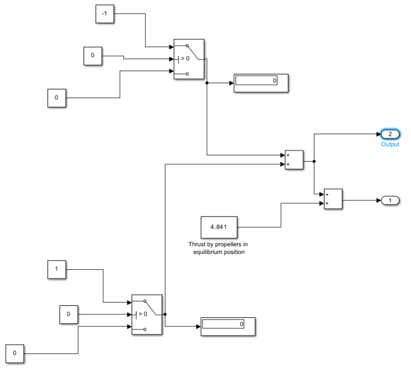


Fig.21 Simulink Model for Setting the thrust in Up/Down motion

c) Determining input levels for heave

User-initiated ascent and descent commands were enabled through virtual "latch" push buttons wired to digitally trigger the switch subsystems from Figure 22.
Pressing the upward control assigns a value of 5, toggling the upper switch and engaging +1N input until later released. This sustained upward force persists by maintaining the finite assignment value even when the button itself reverts to 0 after a press.
Similarly, the downward latch button assigns a value of 5 to throw the lower switch, continually applying its -1N descent load until reset.
This mechanism allows the user to intuitively direct sustained heaving motions in either vertical direction by tapping to initiate and terminate directional thrust inputs. The latch buttons avoid the need to hold the controls for entire maneuvers through automated retention of digital switch positions.
In this way, upward/downward movement commands were straightforwardly implemented through momentary taps tied to the pre-configured force inputs and corresponding speed control loops.

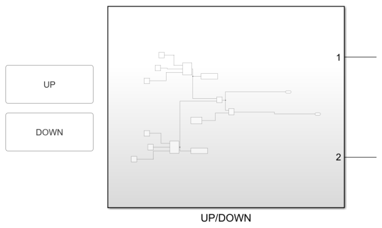


Fig.22 Simulink Model of Value Input of Up/Down Motion

2) Simulating Rotational Motion

a) Setting propeller speeds for clockwise and counter-clockwise torques

Inducing a yaw rotational moment on the underwater drone involves creating an imbalance in thrust torques between the diagonal vertical rotor pairs. To accomplish this without lateral motion, the total lift must remain evenly distributed.
Assuming a 1:9 speed ratio between slowed and accelerated propellers respectively, the total 4.841N z-force is divided:

Thrust per Fast Rotor = (81*TotalThrust)/164
= (814.841 N)/164
= 2.7059 N

Thrust per Slow Rotor = (1*TotalThrust)/164
= (14.841 N)/164
= 0.0295 N

Then resolving speeds from thrust:
Fast Speed = $\sqrt{Thrust/K}$
$= 702.8RPM$

Slow Speed = $\sqrt{Thrust/K}$
$= 78.09RPM$

This 78 vs 703 RPM diagonal difference generates rotational torque while keeping net lift equalized. As depicted in the dual switch control model, clockwise and counter-clockwise commands trigger the defined fast/slow speed sets to the respective rotor pairs through MATLAB code.
In this way, spinning the vehicle orientation could be accomplished through directly manipulating the relative velocities of only two sets of motors rather than tuning all four uniquely. The automated thrust distribution and torque calculations thereby simplified integration of rotational maneuvering.

Code used in the Matlab Function

```
function [left_p,right_p]= fcn(rot)
K = (4.841/4)/((500)^2);
if rot == 1
        left_p = sqrt(((81*4.841)/164)/K);
        right_p = sqrt(((1*4.841)/164)/K);

elseif rot == -1
        left_p= sqrt(((1*4.841)/164)/K);
        right_p= sqrt(((81*4.841)/164)/K);

else
        left_p=500;
        right_p=500;
end
```

b) Determining input levels for rotation

Manual control of rotation along the vertical axis is enabled through two latching push buttons designated for clockwise and counter-clockwise commands. As depicted in Figure 23, these digital inputs feed into switch boxes that toggle between on/off states when the respective button is pressed or released. The clockwise button assigns a value of 5 to its associated switch, turning it “on” which then outputs a 1 signal. Releasing the button resets the switch box to 0, removing the spin torque. The counter-clockwise button follows an analogous process, except it maps to a switch value of -5 when latched. This cascades into calling for -1 to induce rotation in the opposite direction. Through this dual push button architecture, the user can initiate and halt positive or negative rotation as desired to control heading under manual operation. The latching behavior avoids the need to constantly hold the buttons during maneuvering. So this digital input arrangement provides an intuitive, easy-to-control method to command rotation along the vertical axis by incrementally dialing in torque levels.

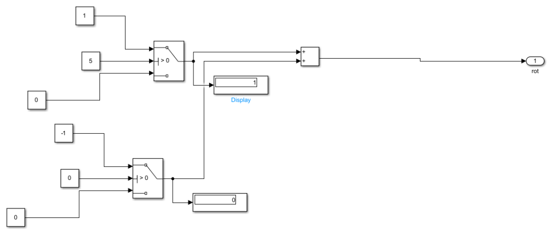


Fig.23 Simulink Model
for the Clockwise/Anticlockwise Rotation

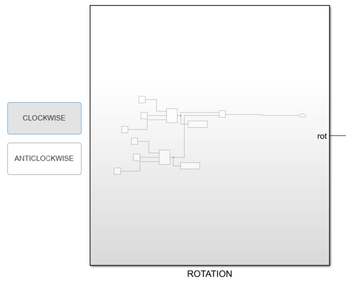


Fig.24 Simulink Model
of Value input for Rotational Motion

3) Combining 2 Motions

a) Calculating rotor speeds and Controlling 4 rotors simultaneously using functions

Executing simultaneous motions combining vertical translation and axial rotation introduces additional complexity in determining the differential rotor speeds. The thrust requirements calculated for the four paired motion scenarios are:

Thrust needed for Upward + Clockwise Motion: 5.841 N
Thrust needed for Downward + Clockwise Motion: 3.841 N
Thrust needed for Upward + Anti-Clockwise Motion: 5.841 N
Thrust needed for Downward + Anti-Clockwise Motion: 3.841 N

To blend these compound movements, the rotors must divide into ”fast” and ”slow” groups based on rotation direction while still collectively meeting the vertical thrust needs. After defining a desired fast:slow speed ratio of 1:9, nested if-then logic was incorporated into the MATLAB control function. This first checks spin state to assign certain rotors to fast or slow sets. Next, it calculates the speeds for each set to produce both the rotation ratio and hit the net thrust targets. For instance, during Up + Clockwise, front and right rotors become fast units while rear and left slow down - but cumulative speeds satisfy the 5.841 N ascent force.
By solving for the two rotor thrust sets, the expanded algorithm enables continuous blending of spinning and translating through more dynamic velocity coordination. This showcases the capability to combine axial and vertical maneuvers through selective speed variation.

Finalized Code for Combined Motion

```
function [left_p,right_p]= fcn(ud, rot)
K = (4.841/4)/((500)^2);
if ud == 1
   if rot == 1
       left_p = sqrt(((81*5.841)/164)/K);
       right_p = sqrt(((1*5.841)/164)/K);
   elseif rot == -1
       left_p = sqrt(((81*3.841)/164)/K);
       right_p = sqrt(((1*3.841)/164)/K);
   else
       left_p = sqrt((5.841/4)/K);
       right_p = sqrt((5.841/4)/K);
   end
elseif ud == -1
   if rot == 1
       left_p = sqrt(((81*3.841)/164)/K);
       right_p = sqrt(((1*3.841)/164)/K);
   elseif rot == -1
       left_p = sqrt(((81*5.841)/164)/K);
       right_p = sqrt(((1*5.841)/164)/K);
   else
       left_p = sqrt((3.841/4)/K);
       right_p = sqrt((3.841/4)/K);
   end
else
```

```
    if rot == 1
        left_p = sqrt(((81*4.841)/164)/K);
        right_p = sqrt(((1*4.841)/164)/K);

    elseif rot == -1
        left_p= sqrt(((1*4.841)/164)/K);
        right_p= sqrt(((81*4.841)/164)/K);

    else
        left_p=500;
        right_p=500;
    end
end
```

6) Simulating 2 Horizontal Propellers:

1) Forward/Backward Motion

a) Setting thrust values in surge and sway

Lateral front-back translation is enacted by both forward-facing horizontal thrusters mounted on the sides spinning at equal speeds. As with other axes, latching push buttons connect to switches that activate +1 Newton forward thrust or -1 Newton reverse thrust. Rather than directional variation, in this case the commanded positive or negative force signals apply synchronously to both side rotors. Pressing “Forward” spins up both thrusters to generate symmetrical 1N propulsion, while “Backward” reverses rotation in unison to achieve -1N rearward force. Releasing the buttons halts axial impulses.
So unlike rotation where rotor speeds split, the paired lateral propellers turn together to add or subtract identical incremental lateral impulses between -1 to 1 Newtons. This synchronicity applies the digital input control architecture to simplify lateral maneuvering. Both side thrusters activate equivalently for each toggled forward/backward latch command, using concurrent speeds to contribute to the net x-axis force dictated by the button presses.

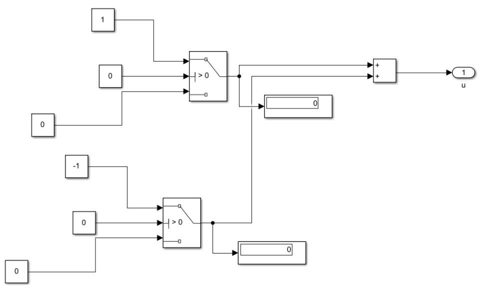


Fig.25 Simulink Model
for the Forward/Backward Rotation

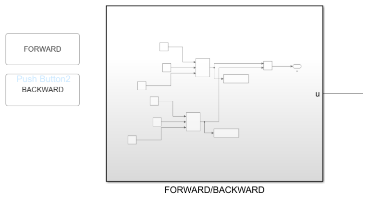


Fig.26 Simulink Model
of Value input for Forward/Backward Motion

b) Calculating rotor speeds for longitudinal control

Translational motion along the forward-backward x-axis is achieved by adjusting lateral rotor speeds in unison. Based on initial experiments, a rotor velocity of 500 RPM was correlated to producing 1 Newton of inline force. By applying this predetermined relationship, the Matlab control function simply sets both side thrusters to 500 RPM when 1N thrust input command is received. To generate reverse thrust, both rotors spin at -500 RPM synchronously by changing only the direction while maintaining speed magnitude. This algorithm outputs the corresponding RPM signal to enforce the proper unified velocity on the lateral thrusters.
Rather than calculating differential sets, the simplified logic directly encodes the empirical understanding that ±500 RPM yields ±1 N through both identically behaving propellers. Toggling between these precise RPM setpoints enables the main simulation model to then physically realize smooth acceleration, constant velocity, and deceleration phases along the x-axis under manual button control. So uniform rotational velocity coordination replaces the need for more complex math to govern lateral movement.

Code used for calculating rotor speed

```
function y = fcn(u)
if u == 1
   y = 500;
elseif u == -1
   y = -500;
else
   y = 0;
end
```

2) Adding Rotational Maneuvers

a) Introducing angled rotation of horizontal thrust vectors

Performing lateral movements during banked turns requires redirecting uniform bilateral thrust to stay

aligned with the instantaneous forward plane. The current z-axis rotation rate feeds into an integrator, generating real-time orientation angle. Using this angle, the singular commanded lateral force gets geometrically projected into y-axis and x-axis components. This keeps the equal-RPM force vector pushing along the symmetric plane even as orientation changes.
For example, during a clockwise yaw while commanding 1N forward thrust, the control scheme tilts output force direction by deducting a portion of thrust in the y-direction. This synchronization transform aligns net translation with the moment-by-moment front-back axis.
So rather than producing differential rotor speeds, the innovation is projecting the fixed symmetric thrust magnitude laterally based on closed-loop rotation feedback. Taking orientation state into account allows smoother trajectory tracking without external intervention. This demonstrates an augmentation still allowing unified manual commands, but with assistive redirection for tracking accuracy.

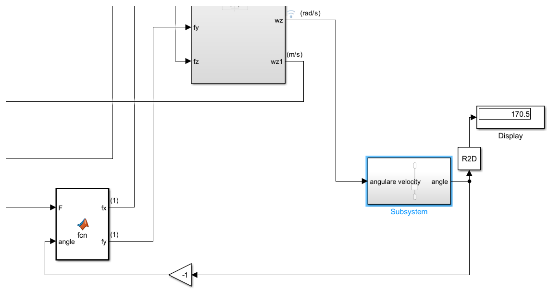


Fig.27 Simulink Model
for Introducing angled movement Rotation

Code used for supplying angled force

```
function [fx,fy]= fcn(F,angle)
fx = F*sin(angle);
fy = F*cos(angle);
```

7) Creating 6-Axis Control Interface:

1) Enabling manual control of all degrees of freedom

An intuitive manual control interface was developed containing six latching buttons corresponding to movement along each major axis. This includes vertical up/down translation, clockwise/counter-clockwise rotation, and forward/backward lateral translation. The simplistic push button architecture mirrors typical remote control interfaces for aerial and underwater drones. The latching behavior enables incrementally ramping velocity up or down along a particular axis by toggling the desired direction's button. Releasing then gradually halts motion. This provides a natural control methodology conformal to manual flight manipulation paradigms. The six inputs then feed into the complex differential rotor calculations, thrust redirections, and closed-loop orientation adjustments handled automatically by the UUV's embedded controllers. So the straightforward user-facing panel reduces cognitive burden during man-in-the-loop operation while enabling full 6-degree freedom by interfacing with the coordination algorithms working in the background.

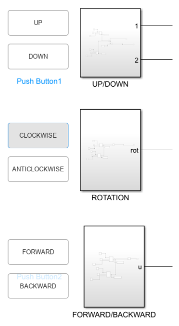


Fig.28 Simulink Model
for 6 Input Control Panel

IV. Results Discussions

The underwater robot simulation results demonstrate controllable stability and maneuverability within the modeling environment. The mechanical design supports manual manipulation along all primary axes, including combinatorial motions blending vertical translation, axial rotation, and lateral translation. The control interface and embedded coordination logic enable intuitive directional commands while performing the complex thrust calculations necessary to achieve responsive multi-axis movements. Visual verification confirms smooth velocity ramps, constant speed cruising, and deceleration across each axis individually or in tandem.

While conceptual at present, this framework provides a foundation for future experimental prototyping by proving consistent stabilized behavior in simulation. Additional sensor integration and automated navigation algorithms could augment the basic teleoperated functionality exhibited here. The simulation constitutes an initial feasibility study, collecting key learnings around multi-thruster underwater vehicle stability dynamics, control parameterization, and manual piloting. These insights guide requirements for eventually transitioning to physical systems. Expanding the platform's autonomy and operational resilience further may better equip versions for tasks such as surveillance, mapping, or exploration.

Additionally, we can observe how each output values varie with time by changing different inputs from the simulink data Inspector.

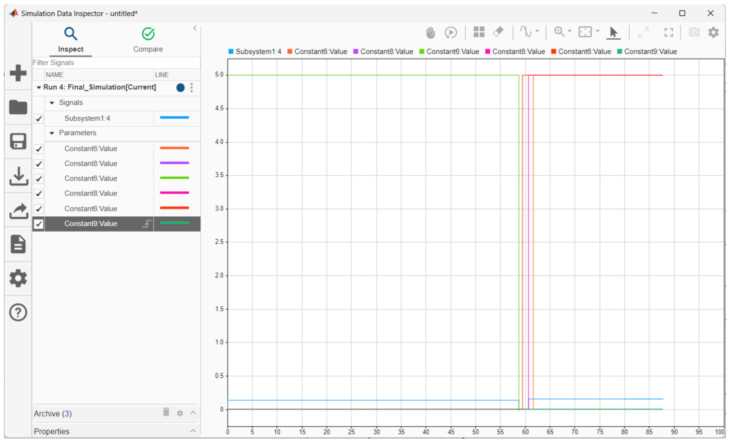


Fig.29 Simulink Data Inspector

V. Summary and Conclusions

This project successfully demonstrated a simulated conceptual design for a six-thruster underwater drone platform controlled manually across six axes. The CAD modeling and physics simulations verified stable flight behavior within the virtual environment using a set of balanced thrust configurations. This offers a foundation for future real-world prototyping and testing. Next steps may include fabricating a physical model to validate and refine maneuvering capabilities. Additional opportunities exist to expand upon the piloting systems for waypoint-based navigation. Onboard sensor integration could also enable terrain following, object detection, and environment reconstruction.

In conclusion, the learning outcomes around designing for underwater environments, modeling multi-rotor stability dynamics, and implementing multi-variable control systems provide a toolkit for progressing further. The simulation serves as an initial feasibility demonstrator for a versatile, omnidirectional UUV suited for tight navigation spaces. Building upon this base by transitioning concepts proven digitally into physical platforms can expand manual and autonomous operational capabilities alike. The outcomes showcase an array of future research directions to progress this modern, multi-propeller marine robot.

References

[1] Yuh, J. (2004). Design and Control of Autonomous Underwater Robots: A Survey. Autonomous Robots, 8(1), 7-24.

[2] S. K. Deb, J. H. Rokky, T. C. Mallick and J. Shetara, ”Design and construction of an underwater robot,” 2017 4th International Conference on Advances in Electrical Engineering (ICAEE), Dhaka, Bangladesh, 2017, pp. 281-284, doi: 10.1109/ICAEE.2017.8255367.

[3] A P Nyrkov et al 2017 J. Phys.: Conf. Ser. 803 012108

[4] Smith, A. et al. (2021). Navigation Challenges for Underwater Robotic Vehicles. Oceanic Engineering Journal. 46(1).

[5] Wang, L. et al. (2022). A Review of Navigation Solutions for Underwater Robots. Annual Reviews in Control. 46.

[6] Wernli, R.L. (2020). Thruster Systems for ROVs and AUVs. Marine Technology Society Journal. 54(2).

[7] Nicholson, J. and Healey, A. (2008). The Present Use and Future Role of Simulation Tools for Underwater Vehicles. Marine Technology Society Journal. 42(3).

[8] Luukkonen, T. (2011). Modelling and Control of Quadcopter. Independent research project in applied mathematics. Aalto University School of Science.