University of Toulon

Marine and Maritime Intelligent Robotics (MIR)

Non-linear Control Report

Akira Techapattaraporn

Hyejoo Kwon

Mukesh Reddy

Submitted on October 23, 2025

1 Assignments

1. Program the control techniques for the inverted pendulum seen in the introductory lectures (basic analysis, linear state-space approach with pole placement or LQR). Leave the feedback linearization for later.
Pay attention to the fact that the analysis is done on the linear system while the control is meant to be applied to the nonlinear system.
To perform the simulation, it is asked not to use an already made solver instead try to implement at least two solvers in the list (ie Euler explicit and Runge-Kutta 4) and compare their capacity to deliver a decent simulation.

A custom RK4 solver was implemented for solving the LQR control of an inverted pendulum. The plots (Figure 1) show the system's angular position (θ), angular velocity ($\dot{\theta}$), control input (torque), and the phase portrait, illustrating the pendulum's stabilization around the upright equilibrium from an initial displacement.

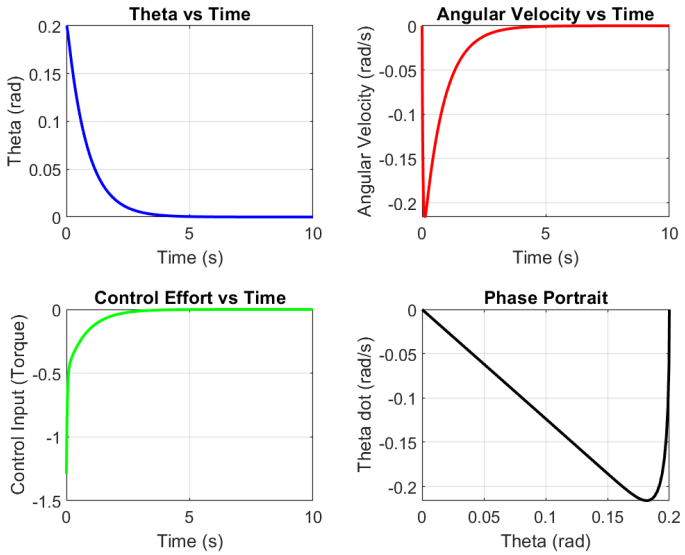

Figure 1: LQR Control Performance for Inverted Pendulum using Runge-Kutta 4 Solver

2. Draw the two vector fields given as examples during the lecture (items (ii) and (iii) after the vector fields definition, the second on being the simplified Lotka-Volterra system). They are both dimension two vector fields, the domain over which you draw them and the resolution of the meshgrid are up to you.

(ii)

$$x(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}, \quad f(x,u) = \begin{bmatrix} -\frac{6x_1^2}{(1+x_1)^2} + 2x_2 \\ -\frac{2x_1+2x_2}{(1+x_1^2)^2} \end{bmatrix}$$

(iii) Lotka-Volterra equation

$$x(t) \in (\mathbb{R}^{+x})^2$$

$$x(t) = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \quad \dot{x}(t) = \begin{bmatrix} x_1(1-x_2) \\ x_2(x_1-1) \end{bmatrix}$$

We visualized the 2D vector fields for the examples (ii) and (iii). Those graphs provides us to understand the system dynamics and behaviors around equilibrium points. For Example (ii), we chose a symmetric domain of: $x_1, x_2 \in [0,\ 2]$. For Example (iii), the simplified Lotka–Volterra system is only defined for positive population values. Therefore, we restrict the domain to $x_1, x_2 \in [0,\ 2]$. The resolution of the meshgrid was set to 0.1 in both cases to get a clear view of the field structure.

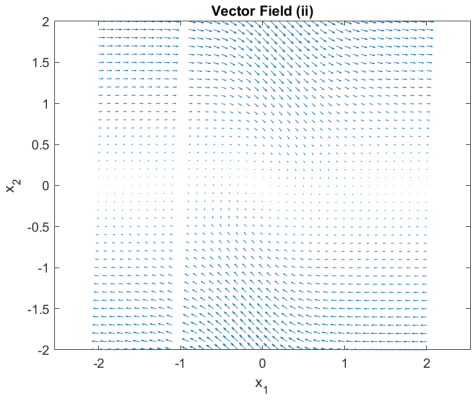

Figure 2: vector fields

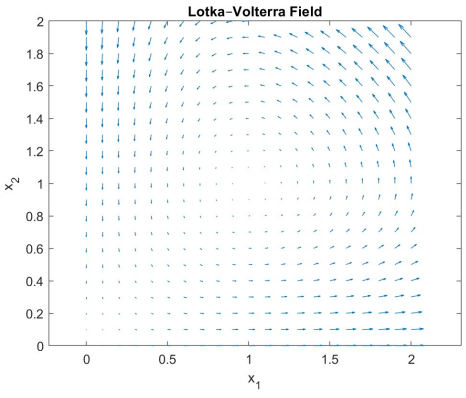

Figure 3: Lotka–Volterra field

As you see in Figure 2, the vector field show rotational and nonlinear shape around the origin. Near the $x1 = 0$ and $x2 = 0$, the magnitudes of vector are small but more aggeressive. In Figure 3, the vector field showing the Lotka-Volterra equation has closed orbit behavior around the point (1,1). It indicates that this system is conservative. In conclusion, each system reveals unique stability properties and dynamic structures.

3. Numerically solve this system (simulation end time and step size are up to you) both by means of an implicit and an explicit method (ie Euler). Tets the implicit method in order to check whether or not the corresponding solution cans witch between the two solutions.

$$\dot{x}(t) = x^{\frac{1}{3}}, \quad x(0) = 0$$

We numerically solved the initial value problem using both explicit and implicit Euler methods. As shown in the Figure 4, the explicit Euler method kept the value at $x(t) = 0$ during the whole simulation.

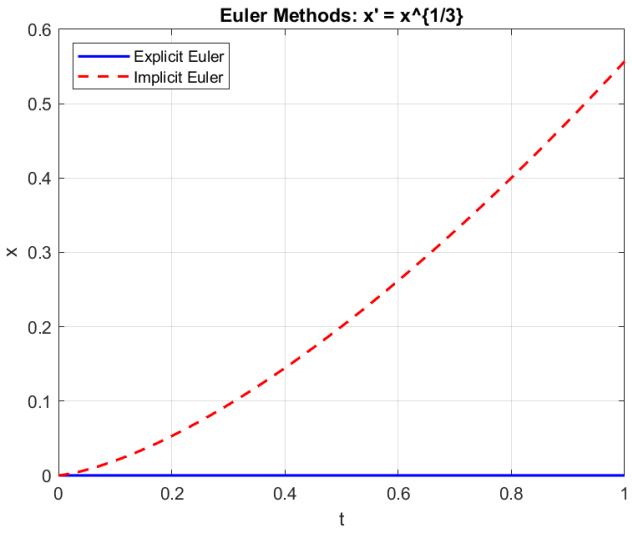

Figure 4: Result of Euler Methods

On the other hand, the implicit Euler method showed a different result. It drew an upward curve and gave the other possible solution.
This means that when there are multiple solutions, the implicit method can show us more options of the behavior of solution, while the explicit method may only stick with the simplest one.

4. Draw the level sets of the fourth example of the Stability section. Which property absence does it highlight ?)

The fourth example is:

$$\dot{x}(t) = \begin{bmatrix} -\dfrac{6x_1^2}{(1+x_1^2)^2} + 2x_2 \\ -\dfrac{2x_1 + 2x_2}{(1+x_1^2)^2} \end{bmatrix} = f(x), \quad \text{where } x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \in \mathbb{R}^2$$

We consider the Lyapunov function:

$$V(x) = \frac{x_1^2}{1+x_1^2} + x_2^2$$

It is clear that $V(x) \geq 0$ for all x, and $V(0,0) = 0$. Moreover, $V(x) > 0$ for any $x \neq 0$, so the function is positive definite.
The derivative of $V(x)$ along the system is:

$$\dot{V}(x) = \nabla V(x) \cdot f(x) = \frac{-12x_1^2 - 4(1+x_1^2)^2 x_2^2}{(1+x_1^2)^2} < 0 \quad \text{for } x \neq 0$$

This shows that the system is Locally Asymptotically Stable. However, $V(x)$ is not a proper function. As $\|x\| \to \infty$, the function $V(x) \to \infty$ only in specific directions.
For example:

$$\lim_{x_1 \to \infty,\, x_2 \to 0} V(x) = \lim_{x_1 \to \infty} \left(\frac{x_1^2}{1+x_1^2} \right) = 1$$

So, the function remains bounded even as $\|x\| \to \infty$, meaning:

$$V(x) \not\to \infty \quad \text{as } \|x\| \to \infty$$

This means the function is not proper and we cannot ensure that this is Global Asymptotic Stability.
As you can see in Figure 5, the level sets of $V(x)$ show that values are minimum at the origin and increase away from it. However, the function flattens in the x_1 direction. This means that $V(x)$ does not grow to infinity as $\|x\| \to \infty$. This supports our conclusion that the system is locally asymptotically stable, but not globally asymptotically stable.

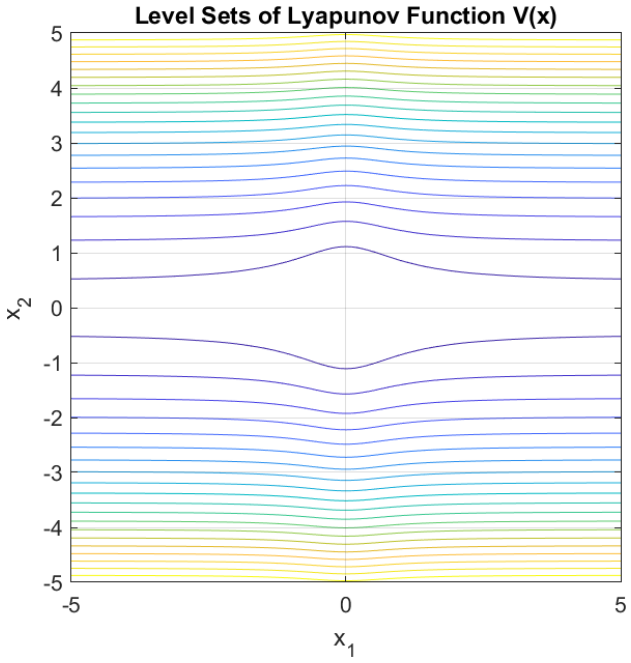

Figure 5: Level set of the fourth example

5. Implementation example 1 (Jurdjevic-Quinn technical)

The example system is :

$$\dot{x} = f(x) + g(x)u = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u, \quad x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}.$$

To study the stability of this system, we define a Lyapunov function:

$$V(x) = \frac{1}{2}(x_1^2 + x_2^2).$$

The gradient of the function is:

$$\nabla V(x) = \begin{bmatrix} x_1 & x_2 \end{bmatrix}.$$

This function is positive definite, reaching its minimum at the origin ($V(0) = 0$) and positive elsewhere. It is also a proper map, as $V(x) \to \infty$ when $\|x\| \to \infty$.
The Lie derivative along $g(x)$ is:

$$L_g V(x) = \nabla V \cdot g(x) = x_2.$$

We apply the Jurdjevic-Quinn stabilization method and define the feedback control:

$$u(x) = -\alpha \cdot L_g V(x) = -\alpha x_2, \quad \alpha > 0.$$

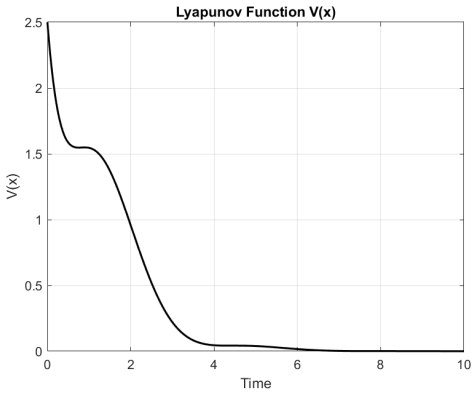

Figure 6: Lyapunov Function V(x)

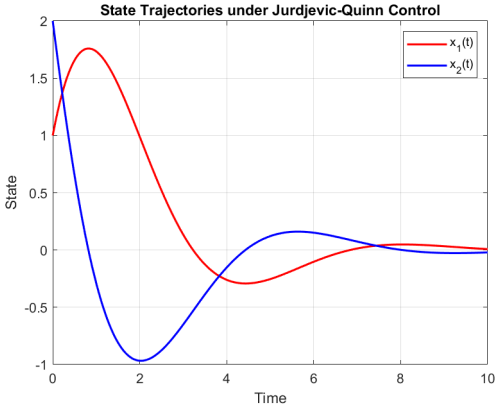

Figure 7: Trajectorie

We simulated the system with initial condition $x(0) = [1,\, 2]^T$ and $\alpha = 1$. In Figure 6, the value is 2.5 at the beginning, but it keeps going down and gets very close to 0. This means the energy of the system is decreasing. In addition, the states are moving toward the origin. So, we can say the controller works well and stabilizes the system. Looking at the state trajectories, both $x_1(t)$ and $x_2(t)$ oscillates at the beginning, but the amplitude decreases over time. Eventually, both states converge to zero, which matches the expected behavior for a locally asymptotically stable system under Jurdjevic–Quinn control.

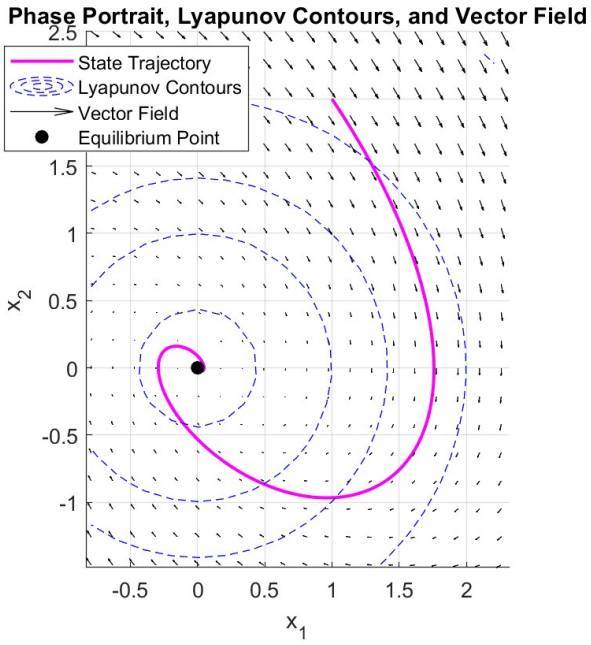

Figure 8: Phase Portrait, Lyapunov Contours, and Vector Field of the Controlled System

6. Finish example 2 (mass-spring)

First, we define the energy and lagrangian equations. 1. Kinetic Energy:

$$E_K = \frac{1}{2}m(\dot{x}_1^2 + \dot{x}_2^2)$$

2. Potential Energy :

$$E_P = \frac{1}{2}k(x_1 - \ell_0)^2 + \frac{1}{2}k(x_2 - x_1 - \ell_0)^2$$

3. Lagrangian:

$$\mathcal{L} = E_K - E_P$$

The Euler-Lagrange equations is following :

$$\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{x}_i}\right) - \frac{\partial \mathcal{L}}{\partial x_i} = Q_i$$

- For x_1 (no external force):

$$m\ddot{x}_1 + k(x_1 - \ell_0) - k(x_2 - x_1 - \ell_0) = 0$$

- For x_2 (control force u):

$$m\ddot{x}_2 + k(x_2 - x_1 - \ell_0) = u$$

Let the state vector be:

$$X = \begin{bmatrix} x_1 \\ x_2 \\ v_1 \\ v_2 \end{bmatrix}, \quad \dot{X} = \begin{bmatrix} v_1 \\ v_2 \\ \frac{-k(x_1-\ell_0)+k(x_2-x_1-\ell_0)}{m} \\ \frac{-k(x_2-x_1-\ell_0)}{m} + \frac{u}{m} \end{bmatrix} = f(X) + g(X)u$$

where

$$g(X) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \frac{1}{m} \end{bmatrix}$$

The equilibrium state (in absence of control) is:

$$\bar{X} = \begin{bmatrix} \ell_0 \\ 2\ell_0 \\ 0 \\ 0 \end{bmatrix}$$

We define the total energy as the Lyapunov function:

$$V(X) = \frac{1}{2}m(v_1^2 + v_2^2) + \frac{1}{2}k(x_1 - \ell_0)^2 + \frac{1}{2}k(x_2 - x_1 - \ell_0)^2$$

Then, the gradient of $V(X)$ is as following :

$$\nabla V = \begin{bmatrix} -k(x_1 - \ell_0) + k(x_2 - x_1 - \ell_0) \\ k(x_2 - x_1 - \ell_0) \\ mv_1 \\ mv_2 \end{bmatrix}$$

Since the system is conservative and there's no damping:

$$\dot{V} = \nabla V^\top f(X) = 0$$

This implies Lyapunov (but not asymptotic) stability. At equilibrium:

$$\dot{V} = 0 \iff \nabla V = 0 \iff X = \bar{X}$$

For Lyapunov asymptotic stability, we require:

$$\dot{V}(x) < 0 \quad \forall x \in \Delta(x) \setminus \{0\}$$

However, in this conservative system:

$$\dot{V}(x) = 0$$

So only Lyapunov stability (not asymptotic) is achieved.
If we compute feedback law $u(x) = -k \cdot \mathcal{L}_g V(x)$
We have:

$$g(x) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \frac{1}{m} \end{bmatrix}, \quad \nabla V^\top g(x) = m v_2 \cdot \frac{1}{m} = v_2$$

Therefore, the control input is:

$$u(x) = -k \cdot v_2$$

The initial Conditions and Parameters are :

- Mass $m = 1$, spring constant $k = 1$, natural spring length $\ell_0 = 1$
- Control gain $\alpha = 1$
- Initial state: $x_1(0) = 1.5$, $x_2(0) = 2$, $v_1(0) = 0$, $v_2(0) = 0$

This control input is designed to reduce the total energy of the system over time. The Lyapunov function is the total kinetic + potential energy, which is expected to decrease when the system is stabilizing.
In the first plot (Figure 9), we observe the positions and velocities of the two masses. Initially, they exhibit oscillatory behavior due to spring forces, but these oscillations gradually decrease in amplitude. This reduction is attributed to the control input damping the motion, effectively guiding the system toward equilibrium In the second plot (Figure 10), the value of the Lyapunov function $V(x)$ is shown. It starts around 0.25 and decreases smoothly. This graph shows that the energy is steadily disappearing. This behavior aligns with the theoretical expectation that $V(x) \to 0$ as $t \to \infty$, meaning that it is asymptotic stability. Moreover, we also add the phase portrait plots for proving the convergence. Figure 11 confirm the convergence of the

system under the applied control where the initial points are marked as 'o' and final points are masked as 'x'.

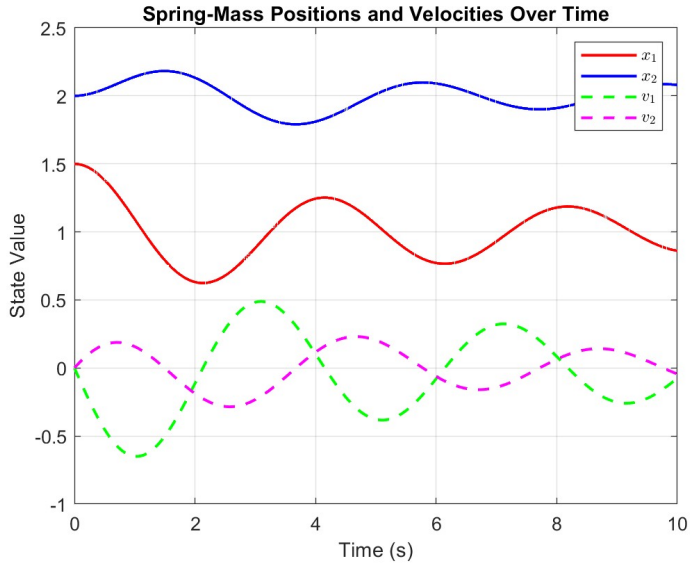

Figure 9: States Plot

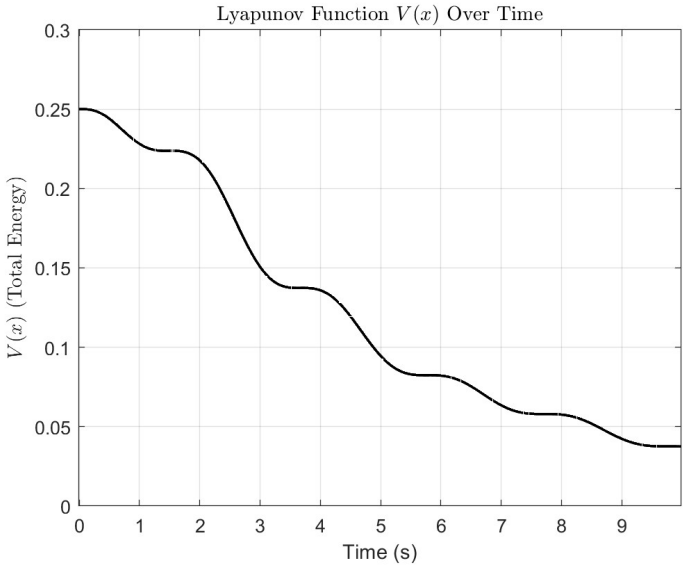

Figure 10: Lyapunov Function Over Time

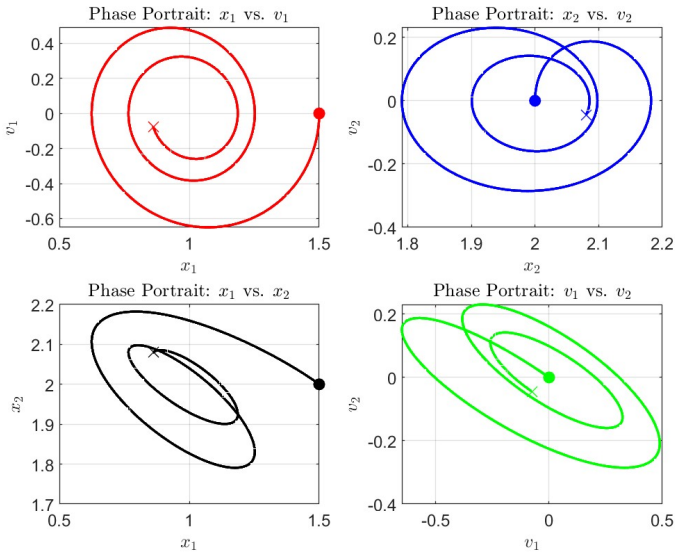

Figure 11: 2D Phase Portrait Plot

7. Implement block diagram (feedback linearization of turtlebot)

The differential-drive robot model was developed according to [1]. The state variables of the robot dynamics are: x-position (x_1), y-position (x_2), and heading angle (θ). The inputs are linear velocity (u_1) and angular velocity (u_2).
To address the ill-conditioned nature of the system's differential delay matrix, an integrator was introduced for the linear velocity input. This augmented the system, creating a new state variable z (where $z = u_1$) and a new control input $\bar{u}_1$ (where $\dot{z} = \bar{u}_1$). The augmented system dynamics are:

$$\begin{aligned} \dot{x}_1 &= z\cos\theta \\ \dot{x}_2 &= z\sin\theta \\ \dot{z} &= \bar{u}_1 \\ \dot{\theta} &= \bar{u}_2 \end{aligned}$$

Feedback linearization was then applied to derive the control law for the actual inputs $(\bar{u}_1, \bar{u}_2)$ in terms of virtual control inputs (v_1, v_2) where:

$$\begin{pmatrix} \ddot{y}_1 \\ \ddot{y}_2 \end{pmatrix} = A(x) \begin{pmatrix} \bar{u}_1 \\ \bar{u}_2 \end{pmatrix}$$

and

$$A(x) = \begin{pmatrix} \cos\theta & -z\sin\theta \\ \sin\theta & z\cos\theta \end{pmatrix}$$

The actual control inputs were thus calculated as $\begin{pmatrix} \bar{u}_1 \\ \bar{u}_2 \end{pmatrix} = A(x)^{-1} \begin{pmatrix} v_1 \\ v_2 \end{pmatrix}$.
A PD controller was designed to generate the virtual control inputs (v_1, v_2) to track the desired trajectory $(x_{1,d}, x_{2,d})$. The control law for the PD controller was given by $v = \alpha_0(y_d - y) + \alpha_1(\dot{y}_d - \dot{y}) + \ddot{y}_d$, with suggested gains $\alpha_0 = 1$ and $\alpha_1 = 2$ according to the Pascal's triangle.
The desired trajectory for the robot was defined as a combination of sinusoidal functions:

$$x_{1,d}(t) = R\sin(f_1 t) + R\sin(f_2 t)$$
$$x_{2,d}(t) = R\cos(f_1 t) + R\cos(f_2 t)$$

We used $R = 15$, $f_1 = 0.02$, $f_2 = 0.12$.
The overall system diagram is represented in Figure 12.

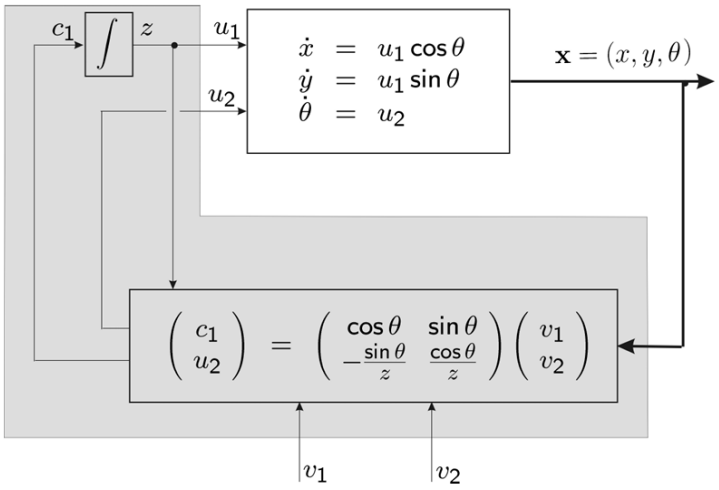

Figure 12: Overall System Diagram

Simulation

The simulation was implemented in Simulink as shown in Figure 13.

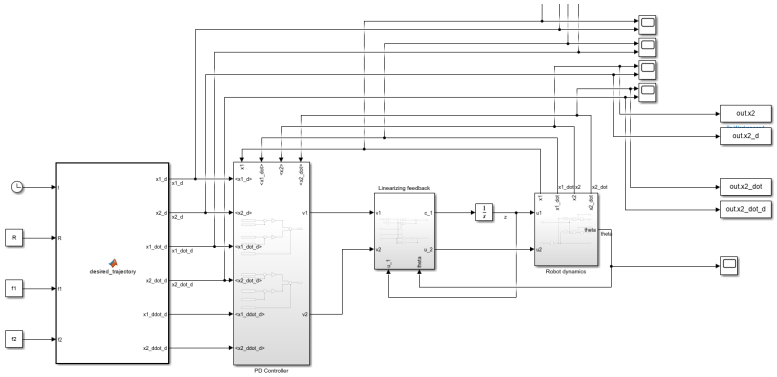

Figure 13: Overall Simulation Model

THe following results are obtained:

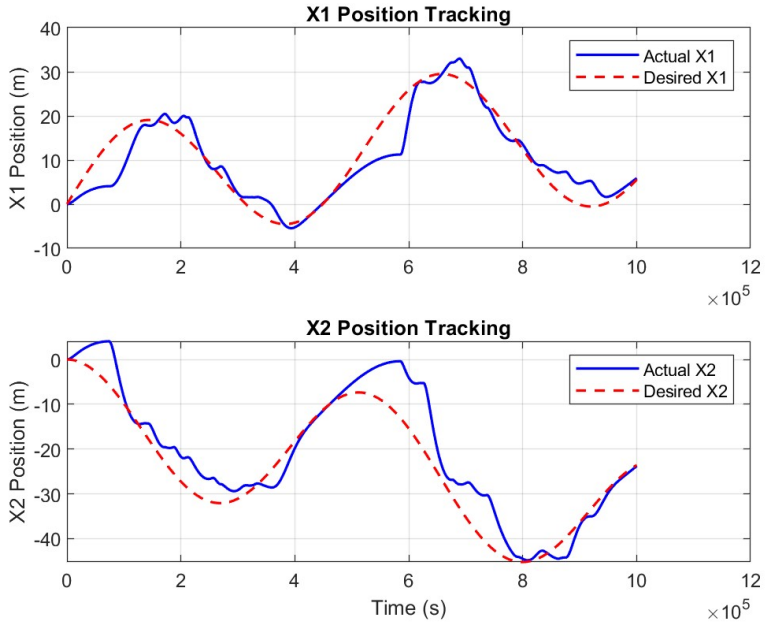

Figure 14: Position Tracking: Actual vs. Desired for X_1 and X_2

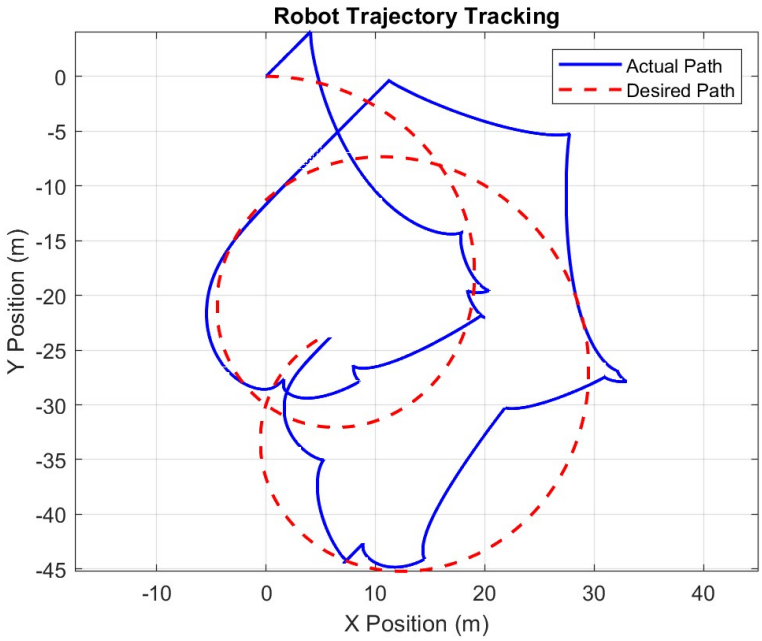

Figure 15: Robot Trajectory Tracking: Actual Path vs. Desired Path

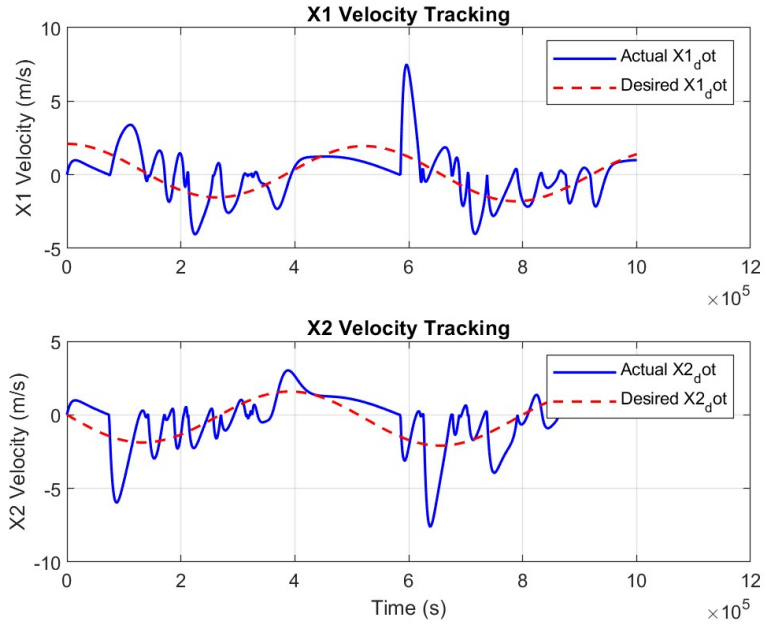

Figure 16: Velocity Tracking: Actual vs. Desired for $\dot{x}_1$ and $\dot{x}_2$

- **Robot Trajectory Tracking (Figure 15):** The actual path (blue

line) qualitatively attempts to follow the desired path (red dashed line). However, there are noticeable deviations, particularly at points of high curvature in the desired trajectory. The actual path exhibits significant oscillations and a general lag relative to the desired path.

- **Position Tracking (Figure 14):**
  - x_1 **Position Tracking:** The actual x_1 (blue line) largely follows the general trend of desired x_1 (red dashed line) but with a consistent lag and smaller peaks/troughs. This indicates the robot is struggling to keep up with the desired position changes.
  - x_2 **Position Tracking:** Similar to x_1, actual x_2 shows a lag and a damped response compared to desired x_2.
- **Velocity Tracking (Figure 16):**
  - x_1 **Velocity Tracking:** actual $\dot{x}_1$ (blue line) attempts to follow desired $\dot{x}_1$ (red dashed line). However, the actual velocity is highly oscillatory and shows significant spikes and undershoots, particularly when the desired velocity changes direction or magnitude rapidly.
  - x_2 **Velocity Tracking:** actual $\dot{x}_2$ exhibits similar oscillatory and spiky behavior, indicating difficulty in smoothly matching the desired Y-velocity.

Despite correct implementation, tracking performance is suboptimal due to several factors. A primary issue is the mismatch between the robot's initial state and the desired trajectory's starting conditions, leading to large initial errors. Furthermore, the control law for $\bar{u}_2$ involves division by the linear velocity z, which can cause control signals to become unstable when z approaches zero. While initial measures were taken, robust handling of this singularity is critical. Finally, the suggested PD controller gains ($\alpha_0 = 1$, $\alpha_1 = 2$) may not be optimally tuned for the desired trajectory, contributing to the observed oscillations and tracking deviations.

2 J-Q Stabilization

Stabilize the Lotka-Volterra system to its equilibrium point

$$x = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \quad u = 0$$

using the **Jurdjevic-Quinn (J-Q) approach**.

System Description:

$$X = \begin{pmatrix} x \\ y \end{pmatrix} \in \mathbb{R}^2, \quad \dot{X} = \begin{pmatrix} x(1-y) \\ y(x-1) \end{pmatrix} + \begin{pmatrix} 1 \\ 0 \end{pmatrix} u = f_0(X) + f_1(X)u$$

Candidate Lyapunov Function:

$$V(x, y) = x - \ln(x) + y - \ln(y)$$

Stage 1: Validate the J-Q assumptions and determine the feedback control law $u = u(x, y)$.

First, let's validate the Lyapunov candidate function by checking if the function is 0 at $\bar{x} = 1$ and $\bar{y} = 1$.

$$V(\bar{x}, \bar{y}) = 1 - ln(1) + 1 - ln(1) = 2$$

The following substitution doesn't satisfy the condition. So, the candidate function needs to be modified. The proposed new candidate function is:

$$V(x, y) = x - \ln(x) - 1 + y - \ln(y) - 1$$

The new candidate function would make the evaluation at the equilibrium be:

$$V(\bar{x}, \bar{y}) = 1 - ln(1) - 1 + 1 - ln(1) - 1 = 0$$

which would satisfy the first condition of Lyapunov stability.
Next, the candidate function needed to be verify if $V(x, y) \geq 0$ for $x, y > 0$. Let

$$f(x) = x - ln(x) - 1$$

then,

$$f(1) = 0$$

$$f(x) > 0 \quad \text{for all} \quad x > 0, x \neq 1$$

The full function:

$$V(x, y) = f(x) + f(y)$$

applies the same properties where it is zero only at $(1, 1)$, strictly positive everywhere, and is a positive definite with respect to the equilibrium point. Moreover, as shown in the surface plot (Figure 17), the function is positive definite.

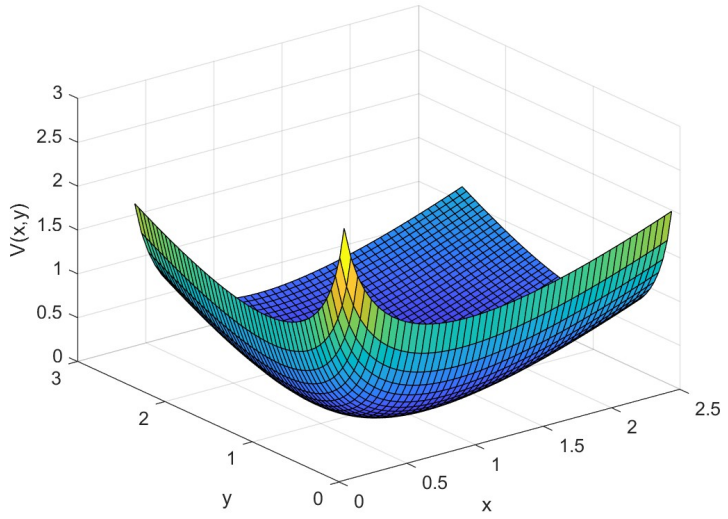

Figure 17: Surface Plot of Lyapunov Candidate Function

Lie Bracket Computation:

To apply the J-Q stabilization, we need to verify that the control vector field and its Lie bracket with the drift vector field span the state space.
Let:

$$f_0(x, y) = \begin{pmatrix} x(1-y) \\ y(x-1) \end{pmatrix}, \quad f_1(x, y) = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

Then the Lie bracket is given by:

$$[f_0, f_1](x, y) = -\frac{\partial f_0}{\partial x} f_1$$

Compute the Jacobian:

$$\frac{\partial f_0}{\partial x} = \begin{bmatrix} 1-y & -x \\ y & x-1 \end{bmatrix}$$

Then:

$$[f_0, f_1](x, y) = -\begin{bmatrix} 1-y \\ y \end{bmatrix} = \begin{bmatrix} y-1 \\ -y \end{bmatrix}$$

Next, we check if the determinant to confirm the local accessibility.

$$M(x, y) = \begin{bmatrix} f_1(x, y) & [f_0, f_1](x, y) \end{bmatrix} = \begin{bmatrix} 1 & y-1 \\ 0 & -y \end{bmatrix}$$

$$\det(M) = -y$$

Since $\det(M) \neq 0$ for all $y > 0$, the vector fields are linearly independent almost everywhere in the domain $x > 0, y > 0$. Thus, the Lie algebra rank condition is satisfied, and the system is locally accessible.

Stage 2: Implement the control and verify the stabilization of the system.

First we calculate the derivative of our Lyapunov function.
The gradient is:

$$\nabla V(x, y) = \begin{pmatrix} 1 - \frac{1}{x} \\ 1 - \frac{1}{y} \end{pmatrix}$$

The system is given by:

$$\dot{X} = f_0(x, y) + f_1(x, y)u$$

Then the time derivative of the Lyapunov function is:

$$\dot{V} = \nabla V^\top f_0 + \nabla V^\top f_1 \cdot u = L_{f_0} V + L_{f_1} V \cdot u$$

Compute each term:

$$L_{f_0} V(x, y) = \left(1 - \frac{1}{x}\right) x(1-y) + \left(1 - \frac{1}{y}\right) y(x-1)$$

$$L_{f_1} V(x, y) = 1 - \frac{1}{x}$$

Then:

$$\dot{V} = L_{f_0} V(x, y) + \left(1 - \frac{1}{x}\right) u$$

To ensure $\dot{V} \leq 0$, we define the feedback control law:

$$u(x, y) = -\alpha_0 \cdot \left(1 - \frac{1}{x}\right), \quad \alpha_0 > 0$$

Substituting this into the expression for $\dot{V}$, we obtain:

$$\dot{V} = L_{f_0} V(x, y) - \alpha_0 \left(1 - \frac{1}{x}\right)^2$$

This guarantees that $\dot{V} \leq 0$, and $\dot{V} < 0$ away from the equilibrium.

Stage 3: Stabilized the L-V system to a closed orbit.

The uncontrolled Lotka-Volterra system exhibits closed orbits (periodic trajectories) in phase space that repeat over time. Instead of stabilizing the system to a single equilibrium point, we want to stabilize it to one of these periodic orbits.
To do this, we define a control law that maintains the system at a constant level set of the Lyapunov function. Let V^* denote the desired orbit level, and define the feedback as:

$$u(x, y) = -\alpha(V(x, y) - V^*) \left(1 - \frac{1}{x}\right)$$

This control law ensures that:

- When $V(x, y) > V^*$, the control damps the system toward the orbit
- When $V(x, y) < V^*$, the control energizes the system
- When $V(x, y) = V^*$, the control vanishes

This creates a damping effect that converges the system to a closed orbit defined by $V(x, y) = V^*$. In contrast to Stage 2, where $\dot{V} < 0$, here we aim for $\dot{V} \to 0$, which corresponds to orbital stabilization rather than point convergence.
A simulation was implemented with following parameters and initial conditions:

- Control gain: $\alpha = 2$
- Target: $V^* = 1.5$
- Initial condition: $x_0 = [2, 0.3]$

As can be observed from the plots, the phase plane plot shows that the desired control converges to the closed orbit rather than the equilibrium point. The Lyapunov plot also confirms $V(x, y)$ stabilizes to the target value V^*.

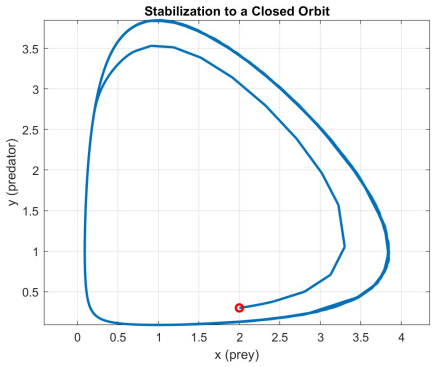

Figure 18: Phase portrait showing stabilization to a closed orbit.

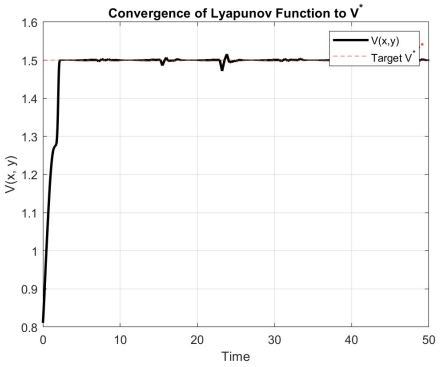

Figure 19: Lyapunov function convergence to the desired energy level V^*.

3 ROV feedback linearization

The objective for this assignment is to stabilize the ROV's depth and pitch using a PID controller. We use [2] as the reference and also compare the system to the direct PID method.

System description and dynamics

According to [2], the simplified 2 DoF model for ROV is represented by:

$$\begin{bmatrix} M_z^* & 0 \\ 0 & M_\theta^* \end{bmatrix} \begin{bmatrix} \ddot{z} \\ \ddot{\theta} \end{bmatrix} + \begin{bmatrix} D_z^* & 0 \\ 0 & D_\theta^* \end{bmatrix} \begin{bmatrix} \dot{z} \\ \dot{\theta} \end{bmatrix} + \begin{bmatrix} -(W-B) \\ Wr_{gz}cos(\phi)sin(\theta) \end{bmatrix} = \begin{bmatrix} \tau_z^* \\ \tau_\theta^* \end{bmatrix}$$

Our state variables are $x = [z, \dot{z}, \theta, \dot{\theta}]^T$. M^*s are inertial and added mass, D^*s represent damping, $(W - B)$ is the net buoyancy force, and the term with Wr_{gz} is the gravitational restoring moment.

Feedback Linearization

Feedback linearization aims to transform the nonlinear ROV dynamics into a simpler, equivalent linear system. This is achieved by designing a specific control input τ^* that precisely cancels out the inherent nonlinear terms of

the system. For our 2-DOF ROV model, the goal is to obtain a decoupled linear relationship of the form $\ddot{z} = v_z$ and $\ddot{\theta} = v_\theta$, where v_z and v_θ are new, linear 'virtual' control inputs. The τ_z^* and τ_θ^* are defined as:

$$\begin{bmatrix} \tau_z^* \\ \tau_\theta^* \end{bmatrix} = \begin{bmatrix} D_z^* & 0 \\ 0 & D_\theta^* \end{bmatrix} \begin{bmatrix} \dot{z} \\ \dot{\theta} \end{bmatrix} + \begin{bmatrix} -(W-B) \\ Wr_{gz}cos(\phi)sin(\theta) \end{bmatrix} + \begin{bmatrix} M_z^* & 0 \\ 0 & M_\theta^* \end{bmatrix} \begin{bmatrix} v_z \\ v_\theta \end{bmatrix}$$

PID Controller

The PID controller is used to generate the virtual control inputs for the linearized system. The controller calculates an output based on the error ($e(t)$) between the desired reference and the measured output, its integral, and its derivative:

$$v(t) = K_p e(t) + K_i \int_0^t e(\tau)d\tau + K_d \frac{de(t)}{dt}$$

These are the gain parameters that we used:

- Depth (z) Controller: $K_{p_z} = 50$, $K_{i_z} = 10$, $K_{d_z} = 5$.
- Pitch (θ) Controller: $K_{p_\theta} = 20$, $K_{i_\theta} = 2$, $K_{d_\theta} = 1$.

Overall Architecture

The complete control system architecture integrates the PID controller with the feedback linearization module, ensuring effective control of the nonlinear ROV dynamics. Figure 20 illustrates the conceptual block diagram of this integrated control system.

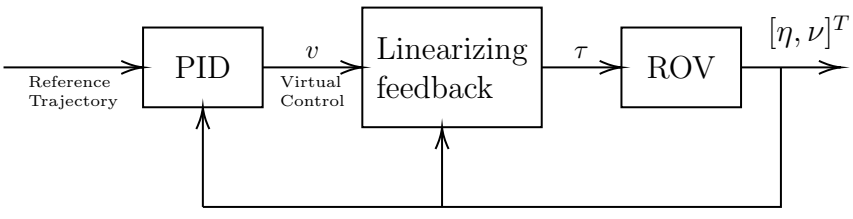

Figure 20: Overall System Diagram

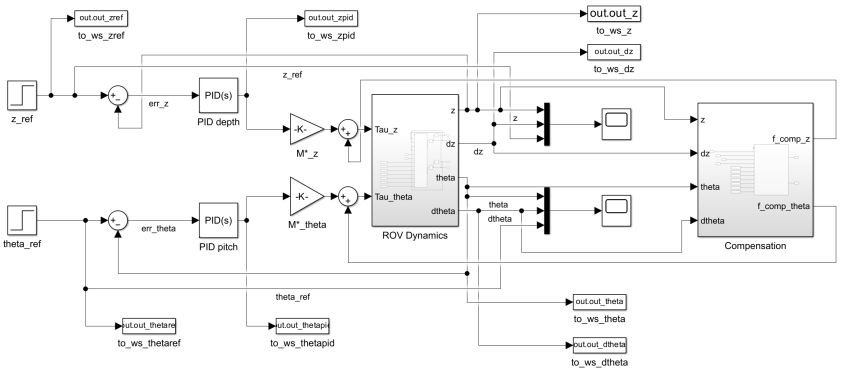

Figure 21: Overall Simulink Diagram

Simulation and Result

Simulations were conducted in MATLAB/Simulink using the ROV model parameters: $M_z^* = 10$, $D_z^* = 5$, $W = 3$, $B = 2.5$, $M_\theta^* = 1$, $D_\theta^* = 0.5$, and $r_{g_z} = 0.05$. The ROV initiated from a resting state ($z = 0, \theta = 0$). Reference trajectories included a step input for depth from 0m to 0.8m at $t = 0.1s$, and a step input for pitch from 0 to 10 degrees ($10 \times \pi/180$ rad) at $t = 0.1s$. The simulations were run for 20 seconds.
To create a baseline, an initial simulation was performed where a PID controller directly commanded the nonlinear ROV dynamics, bypasssing the linearization feedback module.

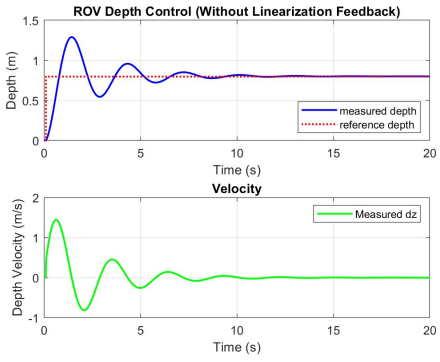

Figure 22: ROV Depth Control with PID: Position and Velocity Response.

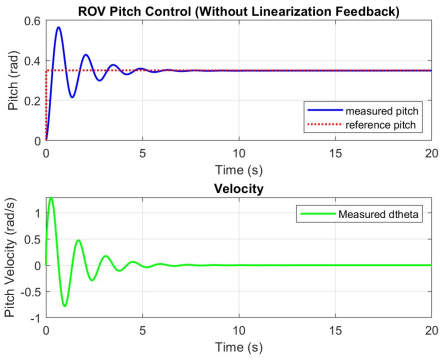

Figure 23: ROV Pitch Control with PID: Position and Velocity Response.

As depicted in Figure 22 and Figure 23, the plots show an oscillatory response. This shows the limitation of PID controller without applying com-

pensation as it gives aggressive transient responses.
Now, we add the compensation terms and see the feedback of the system. By adding this module, the system is expected to be improved.

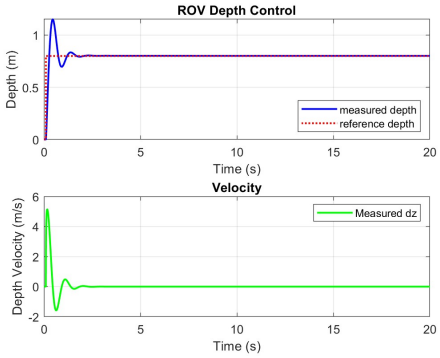

Figure 24: ROV Depth Control with PID and Linearization Feedback: Position and Velocity Response

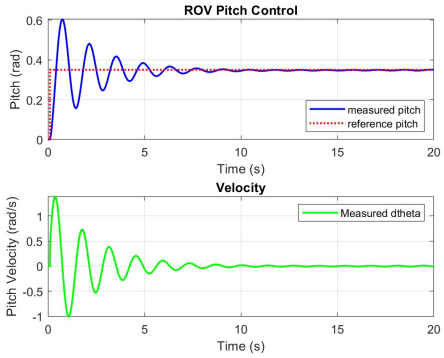

Figure 25: ROV Pitch Control with PID and Linearization Feedback (Initial Gains): Position and Velocity Response.

Figure 24 shows a significant improvement in depth control response. The system overshooted; however, settles much faster than the system without the compensation term, based on the same parameters.
However, as can be observed in Figure 25, the control output exhibit an oscillation. This suggested that the gains are not optimal. So, we re-tune the parameters for pitch to be: $K_{p_\theta} = 4.2$, $K_{i_\theta} = 1$, and $K_{d_\theta} = 3.7$. The tuned response is shown in Figure 27. Moreover, we also tuned the depth controller for the better results, the parameters are: $K_{p_z} = 6$, $K_{i_z} = 1$, $K_{d_z} = 5$.

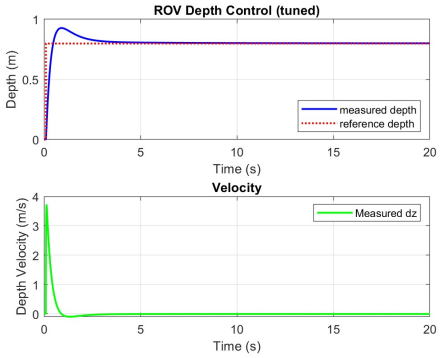

Figure 26: ROV Depth Control with PID and Tuned Linearization Feedback: Position and Velocity Response

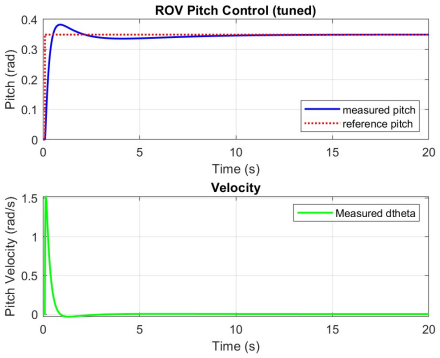

Figure 27: ROV Pitch Control with PID and Tuned Linearization Feedback: Position and Velocity Response

Discussion and Conclusion

This study explored applying feedback linearization with PID control to stabilize a simplified ROV model. Simulations were conducted across three scenarios: direct PID, PID with initial linearization, and PID with tuned linearization. The quantitative results are shown in Table 1. Without the linearization feedback, the controller struggled significantly with the nonlinearities. The results show high overshoot and long settling time. After we add the linearization feedback, the result of the depth controller with the same gains as the experiment without the compensation terms slightly improved; however, the pitch controller response is worsen, which indicate that the parameters gains are not suitable in this system. Therefore, we tuned the controllers and resulted in more suitable gains to both depth and pitch controller.

	No feedback		**With feedback**		**With feedback + tuned**	
	Depth	***Pitch***	***Depth***	***Pitch***	***Depth***	***Pitch***
Steady-state error	-0.07%	0.08%	-0.00%	0.19%	-0.05%	0.04%
Settling time	10.36 s	5.82 s	1.58 s	8.72 s	3.16 s	7.54 s
Overshoot	61.34%	61.95%	43.72%	72.98%	15.92%	9.49%
Rise Time	0.51 s	0.24 s	0.15 s	0.25 s	0.28 s	0.32 s

Table 1: Performance Metrics Comparison of ROV Control Strategies

References

[1] L. Jaulin, *Mobile robotics*. John Wiley & Sons, 2019.

[2] D. Maalouf, A. Chemori, and V. Creuze, "L1 adaptive depth and pitch control of an underwater vehicle with real-time experiments," *Ocean Engineering*, vol. 98, pp. 66–77, 2015.