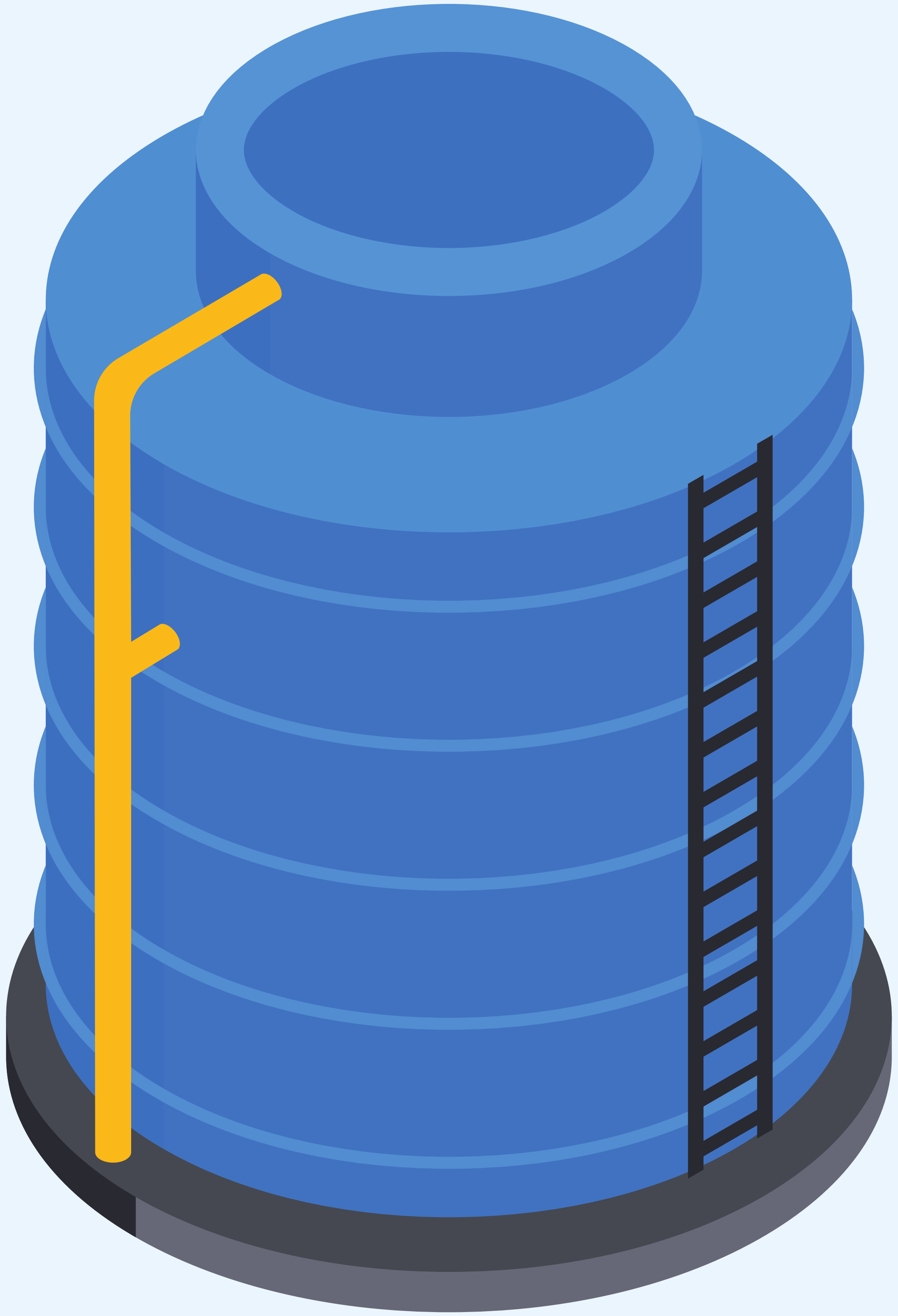
MCTA4362

MACHINE LEARNING - MINI PROJECT

WATER LEVEL CONTROL IN TANK



Team Members

Adlin
(2111324)

Azliyana
(2210620)

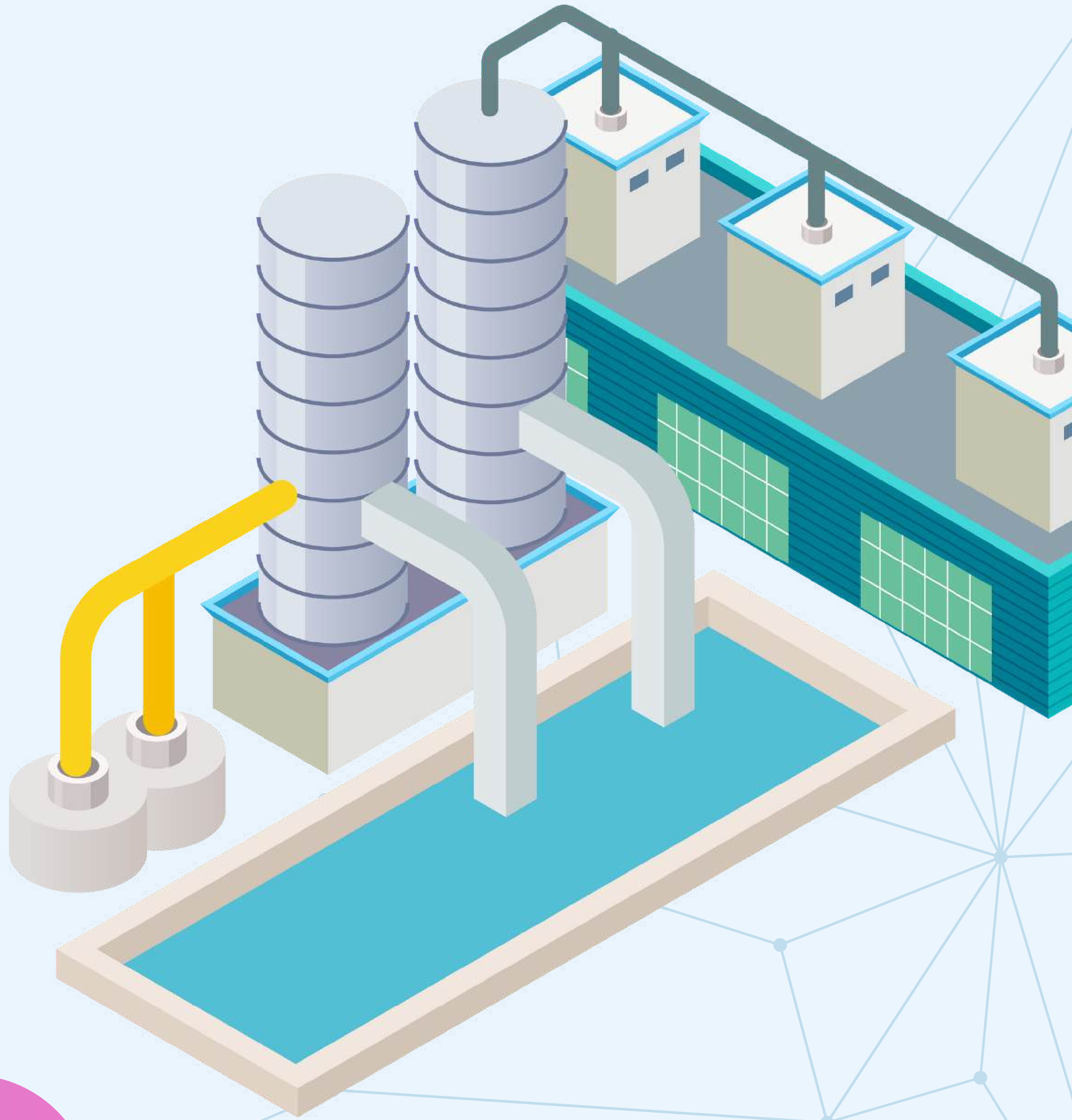
Huda
(2210776)

Maisara
(2217856)

Problem Statement

The nonlinear nature of water flow and dynamics in a tank system makes it difficult for **conventional PID controllers** to maintain accurate water level regulation under varying conditions .

To solve this problem, we aim to implement a **reinforcement learning-based controller** for a nonlinear water tank level system and evaluate its performance against a traditional PID controller.



System Modelling

- Simulates a water tank with controlled inflow and gravity-driven outflow.
- Commonly used in fluid level control problems (chemical plants, irrigation, etc.).
- Nonlinearity introduced by outflow being proportional to sqrt H (Torricelli’s law).
- Serves as a benchmark system for comparing PID and Reinforcement Learning controllers.
- Implemented in Simulink for real-time simulation and control analysis.

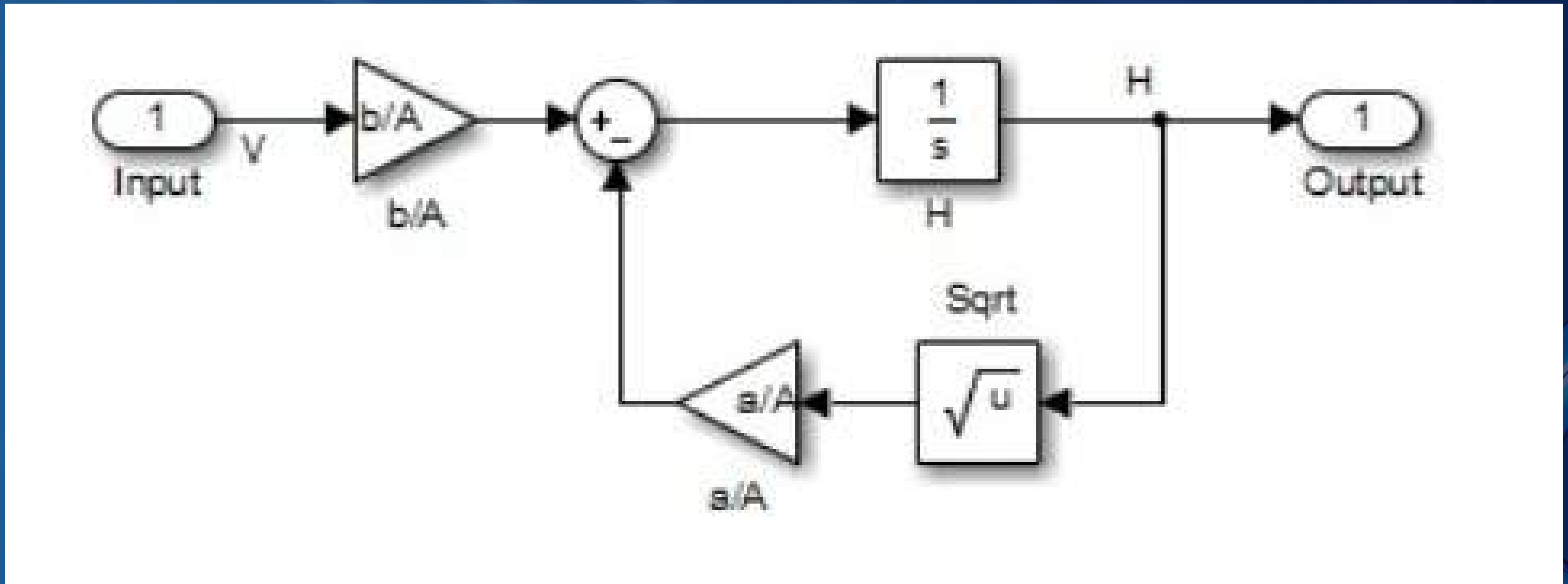
Mathematical Model

$$\frac{dH(t)}{dt} = \frac{1}{A}\left(V(t) - a\sqrt{H(t)}\right)$$

FIRST ORDER NONLINEAR DIFFERENTIAL EQUATION

- $H(t)$: Tank water level (output)
- $V(t)$: Inflow (control input)
- A: Tank cross-sectional area
- a: Outflow coefficient (gravity-driven)
- Captures balance between controlled inflow and nonlinear outflow

Simulation Block Diagram



- In1 (Input): Receives control input V(t) from the controller (e.g. PID or RL)
- Gain (b/A): Scales the inflow according to the tank's cross-sectional area
- Sum (+ -): Computes net flow: inflow – outflow
- Integrator: Integrates net flow to calculate water level H(t)
- Sqrt: Calculates sqrt H to model gravity-driven outflow (Torricelli's law)
- Gain (a/A): Scales the outflow rate using outflow coefficient a
- Out1 (Output): Outputs the current water level for feedback or display

Control Objectives

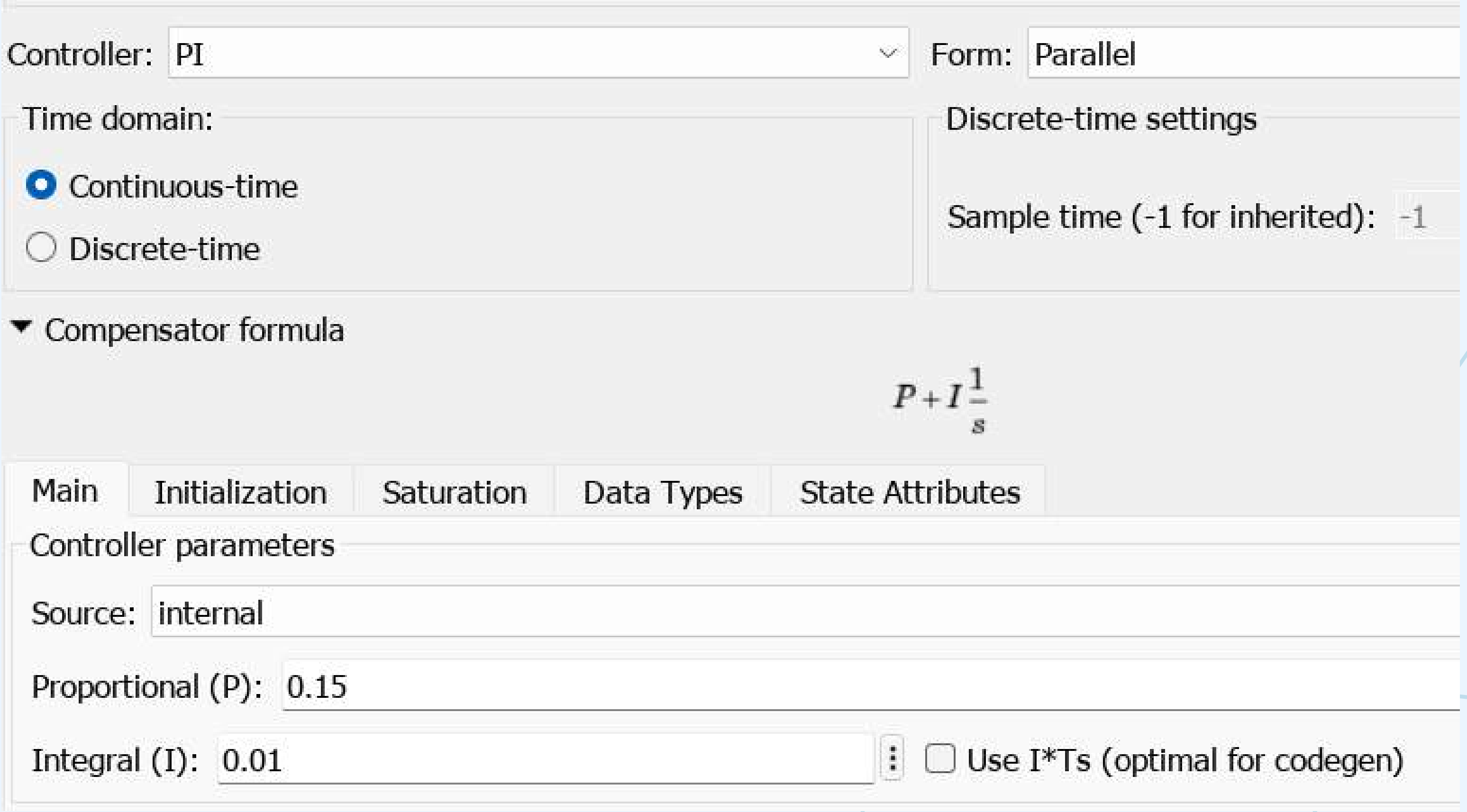
- To implement and tune a PID controller
- To train an RL agent for adaptive control
- To evaluate and compare their performance. Use key metrics like rise time, overshoot, and settling time

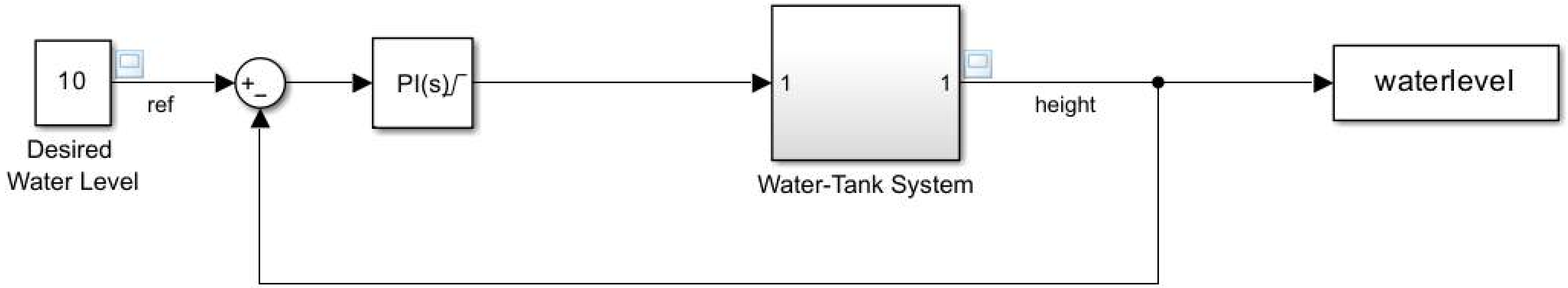
Assumptions & Parameters

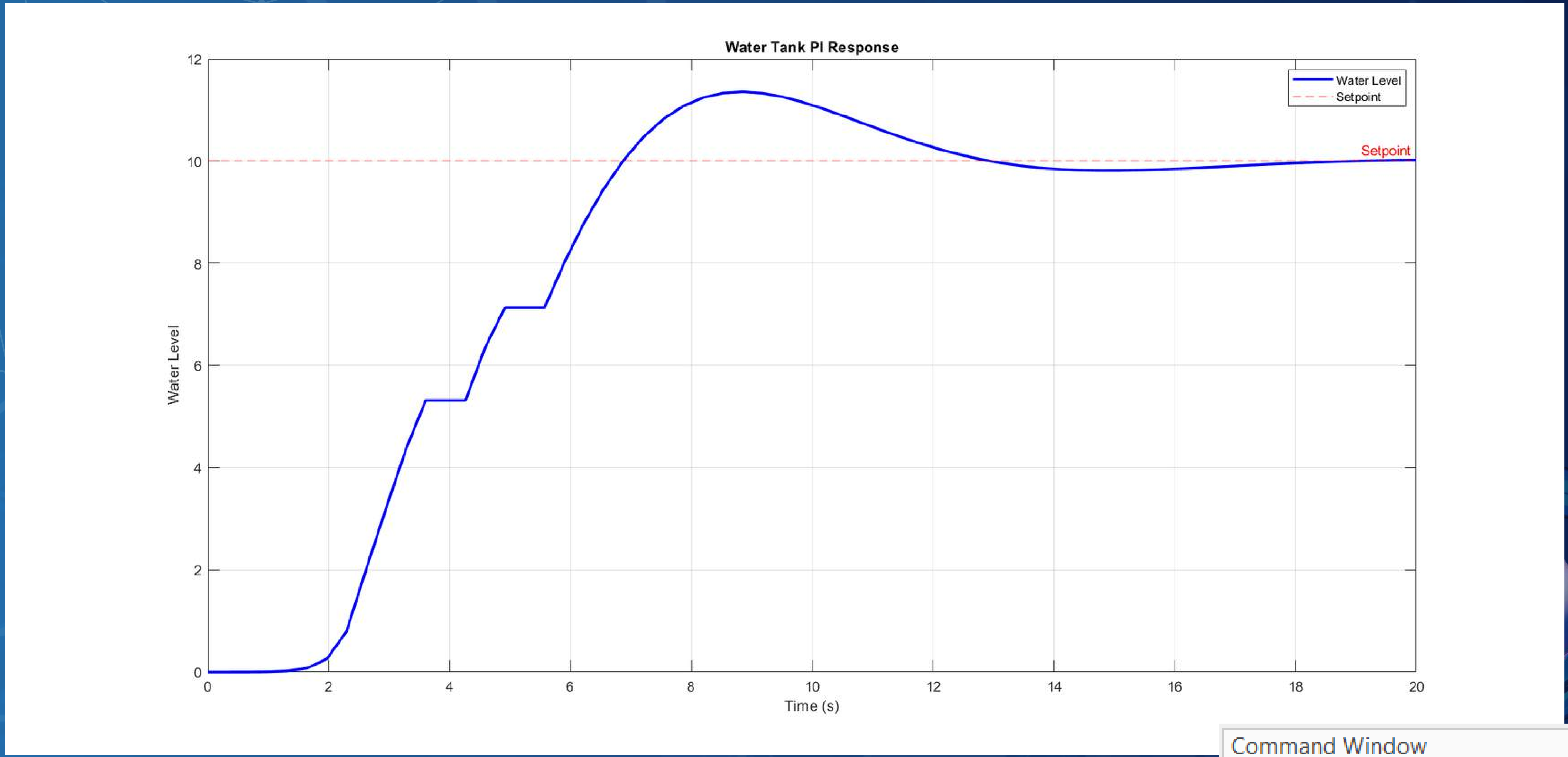
- Ideal actuator and sensor (no delay/noise)
- No leakage or external disturbances
- Constant fluid properties (incompressible water)
- Vertical tank with constant area
- Tank area $A=4.0\ m^2$
- Outflow coefficient $a=0.4$
- Inflow gain $b=1.0$
- $b/A=0.25$, $a/A=0.1$, and initial level $H0=0$
- These values help simulate realistic yet controllable water level behavior

PI Controller

- The tank dynamics include a square-root relationship in the outflow, introducing nonlinearity that complicates analytical control.
- PI controller was selected to regulate the water level in a nonlinear tank system.







Plot showing water tank PI controller response

Metrics performance

Command Window

```
>> watertank_pi_metrices
Performance Metrics:
Rise Time: 3.9791 seconds
Settling Time: 12.2020 seconds
Overshoot: 13.55 %
Peak Time: 8.8525 seconds
Mean Squared Error (MSE): 16.846315
>>
```

RL Controller for Water Tank System using DDPG

MCTA4362

Objective: Design a controller to regulate water level in a tank using RL

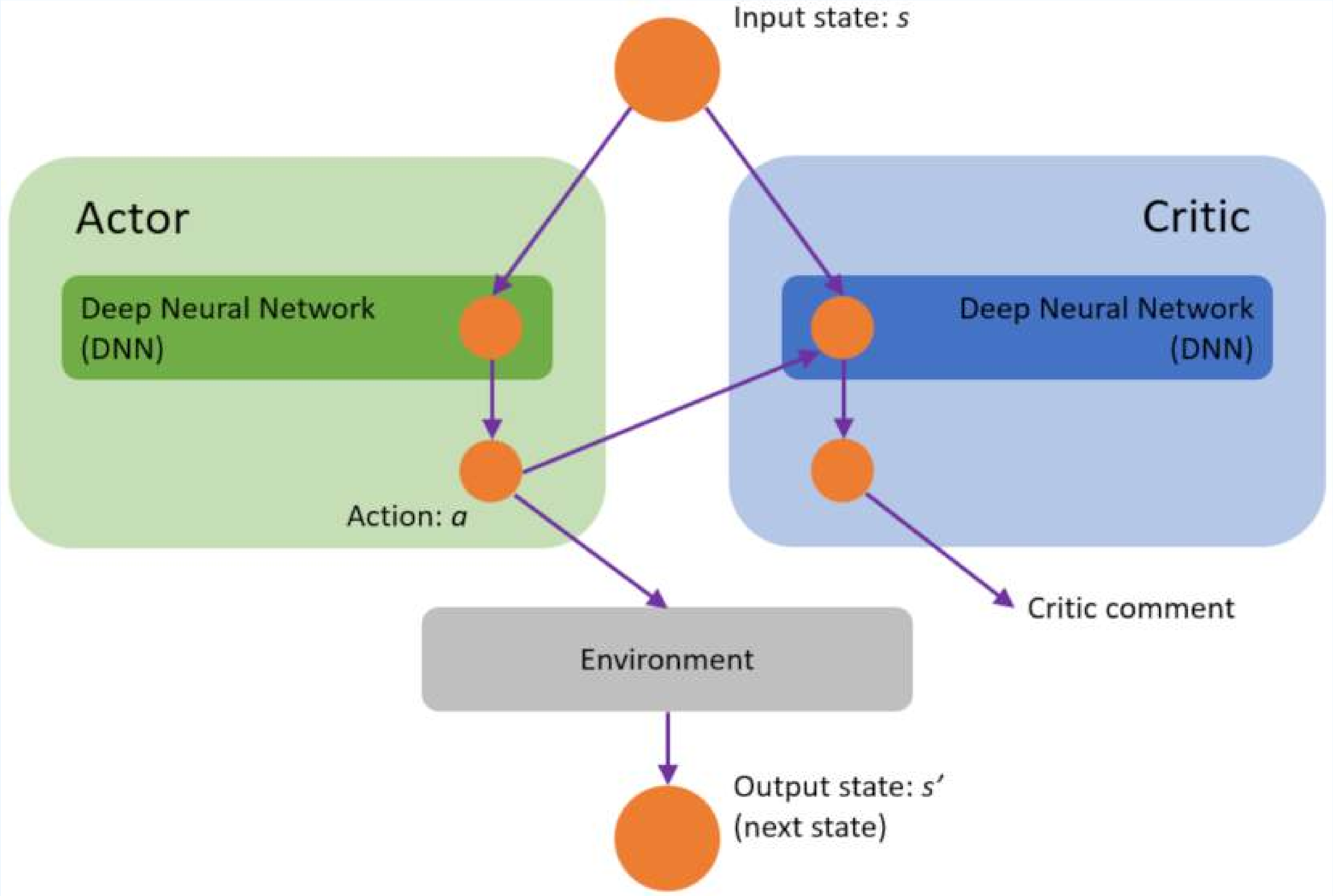
Approach: Implement Deep Deterministic Policy Gradient DDPG in Simulink environment

Key Components:

- RL Agent Design
- Reward Engineering
- Simulation and Training
- Performance Evaluation

Why DDPG and Environment Design

Deep Deterministic Policy Gradient (DDPG) architecture



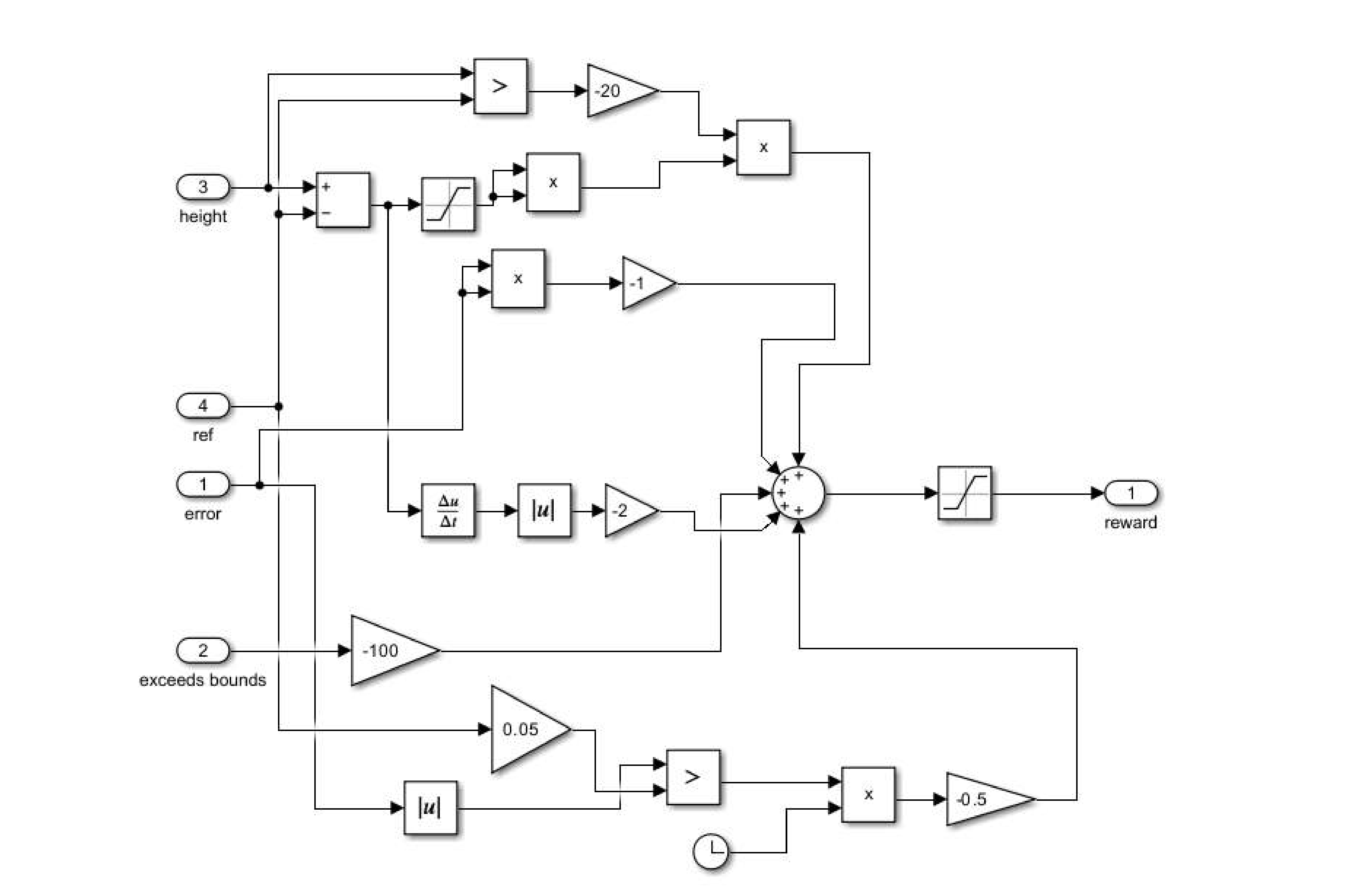
Why Deep Deterministic Policy Gradient (DDPG):

- Suitable for continuous state/action spaces
- Proven in physical control tasks
- Supported by MATLAB RL Toolbox

Environment Design:

- State Space:
  - Water height
  - Error from reference
  - Derivative of error
  - Boolean: within +-5% tolerance
- Action space
  - Valve opening: continuous scalar [0,1]

Reward Design

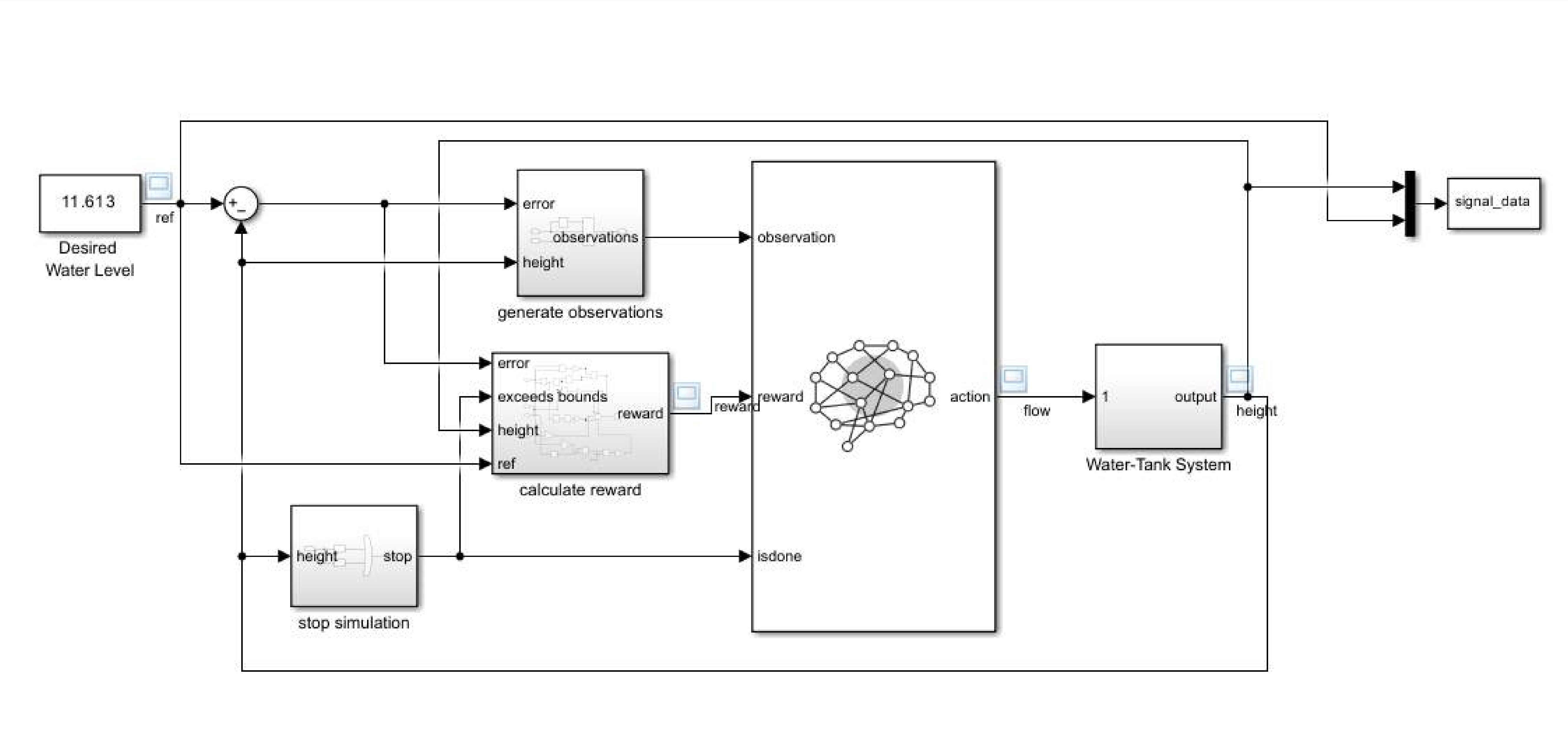


- Accurate reference tracking
- Penalize
  - Overshoot
  - Oscillation
  - Settling time delay
  - Exceeding bounds

- Reward Formula:

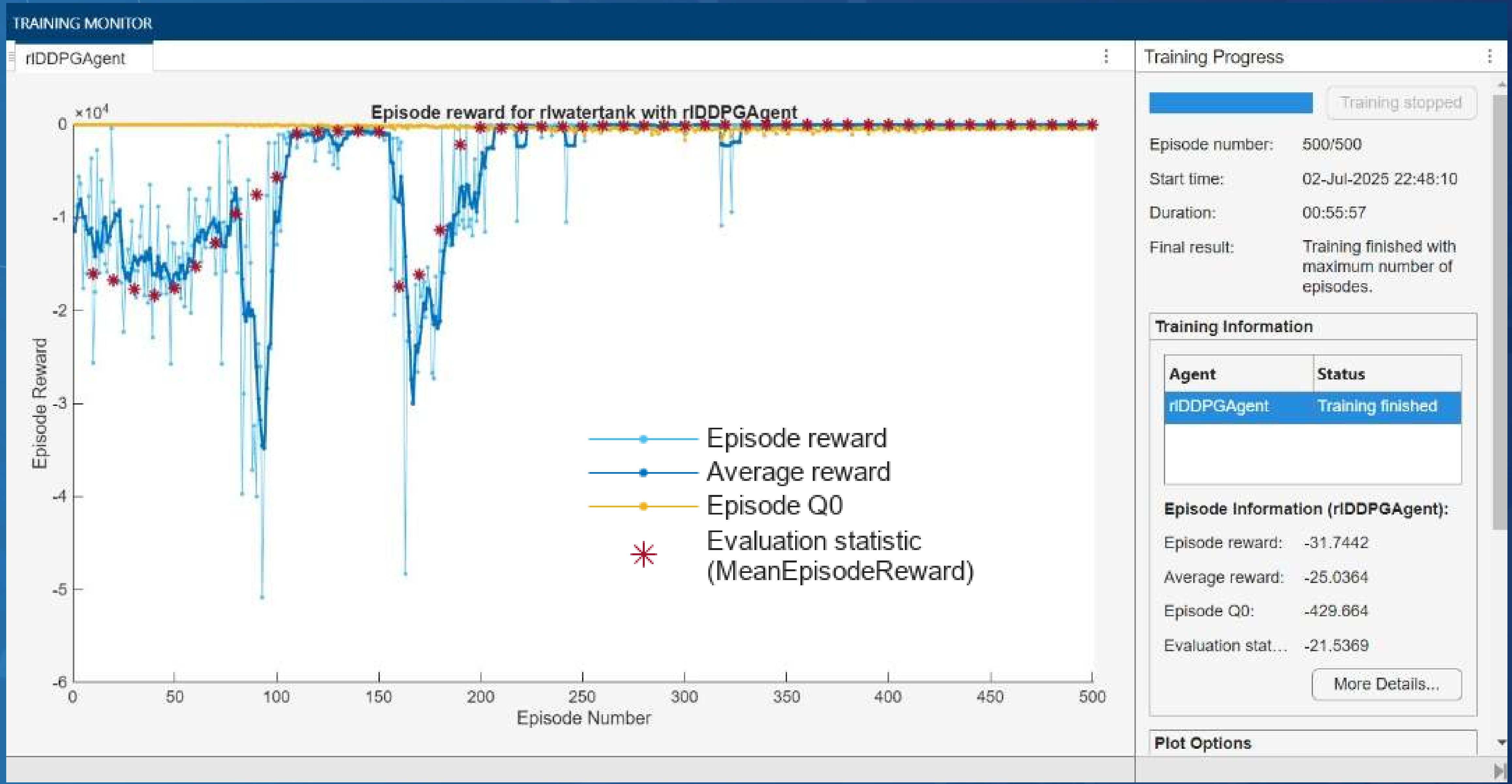
$R = -[(h - href)^2 + \alpha u^2 + \beta(\text{overshoot})^2 + \gamma(\text{dError})^2 + \delta t \times \text{unsettled} + \text{hard penalty}]$

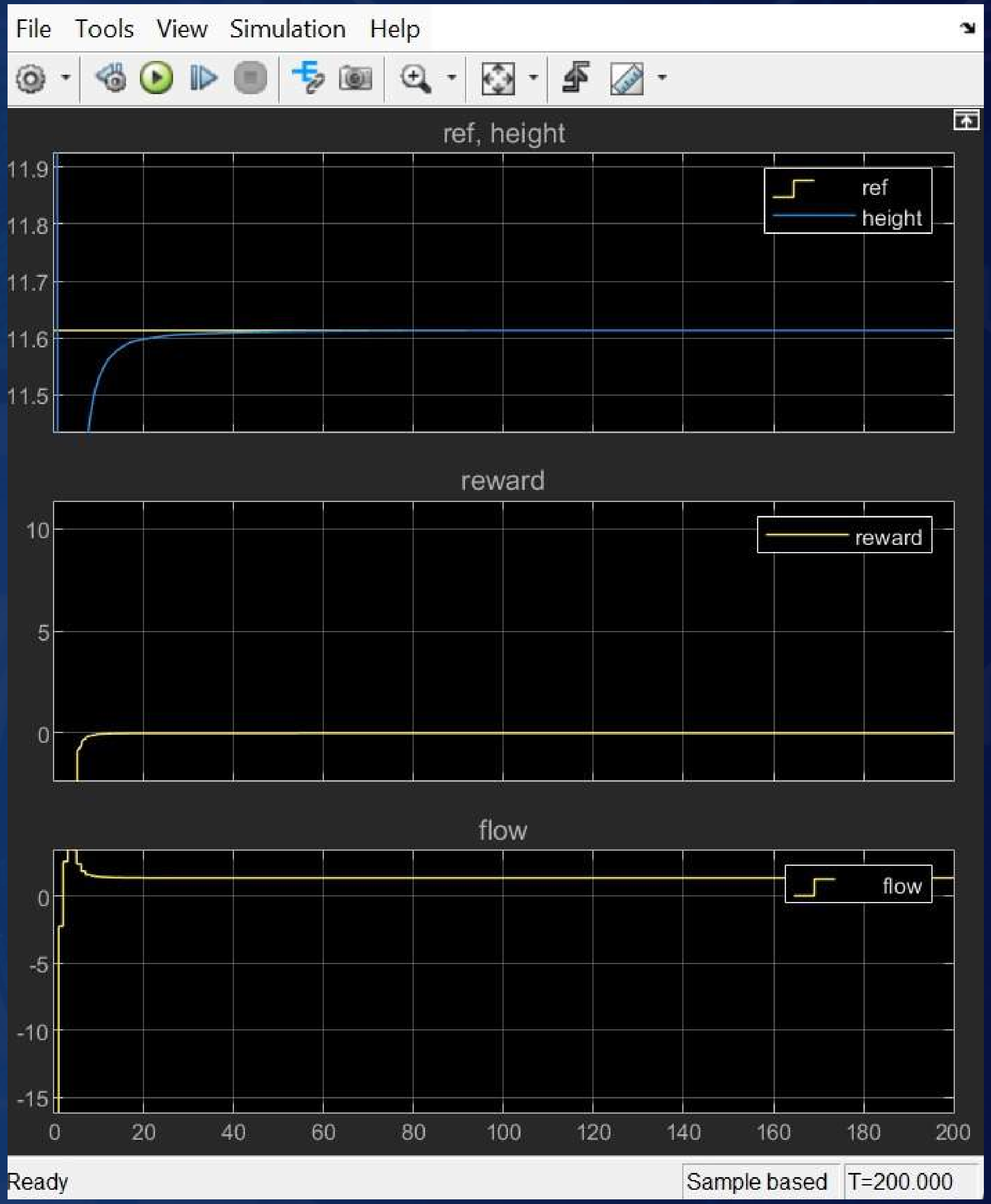
Training Setup



- RL Agent Block
  - contains actor and critic networks
  - input: observation, reward, isdone
  - output: action (flow rate)
- Generate observations
- Desired water level
- Calculate Reward
- stop simulation
- water-tank system

Simulation and Learning Process





Performance metric

Performance Metrics (RL Controller):

- ➤ Rise Time : 0.00 seconds
- ➤ Settling Time : 200.00 seconds
- ➤ Overshoot : 54.96 %
- ➤ Undershoot : 11.39 %
- ➤ MSE : 0.1909

PI vs RL

Testing phase

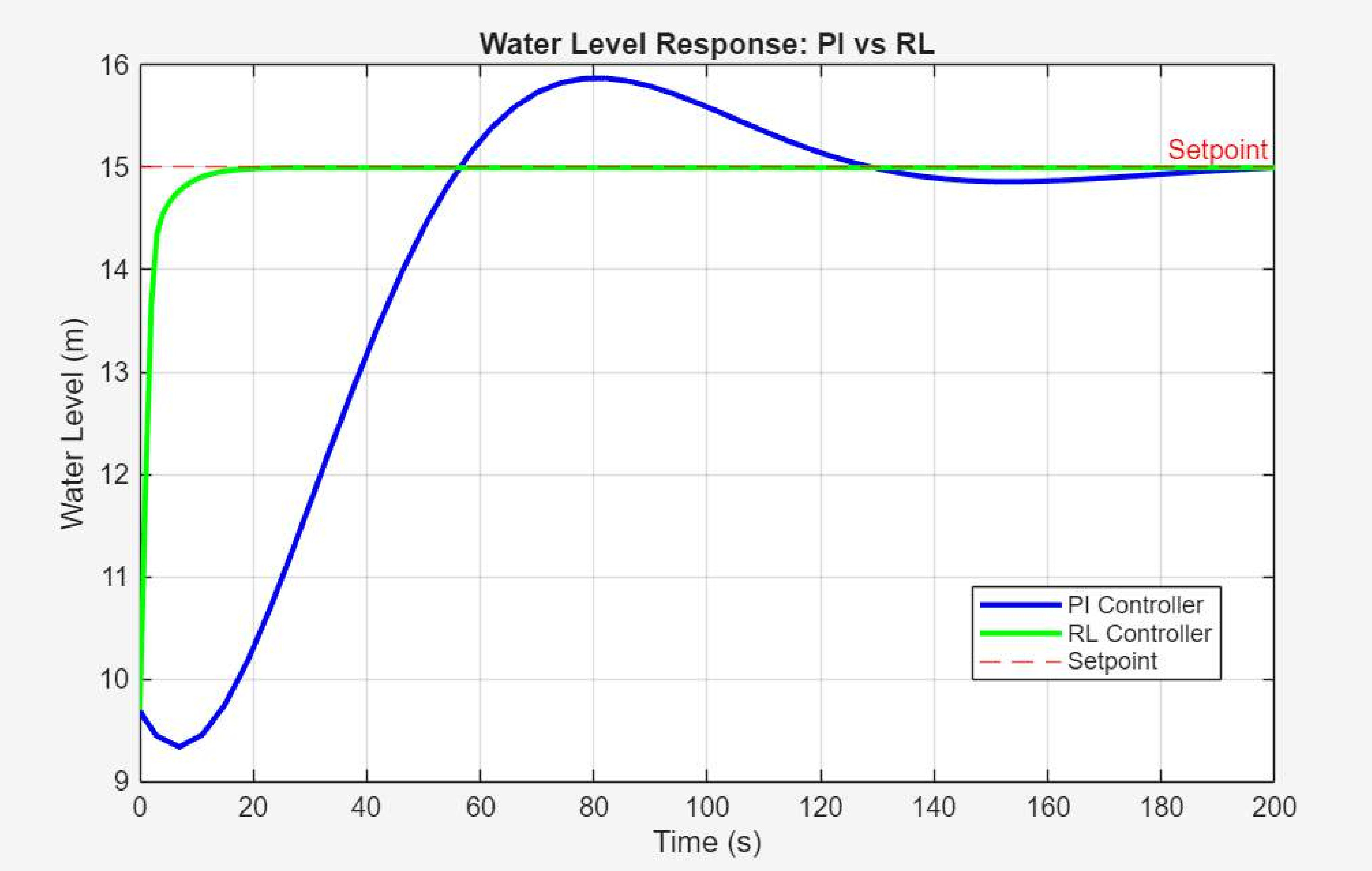
- Comparing between PI and RL controller in a controlled environment
- Desired Water Level : 15 m
- Initial Water Level : 9.5 m

Output:

Performance Metrics Comparison:

Metric	PI	RL
Rise Time (s)	NaN	NaN
Settling Time (s)	110.08	5.00
Overshoot (%)	5.81	-0.02
Mean Squared Error	6.17	0.20

>>

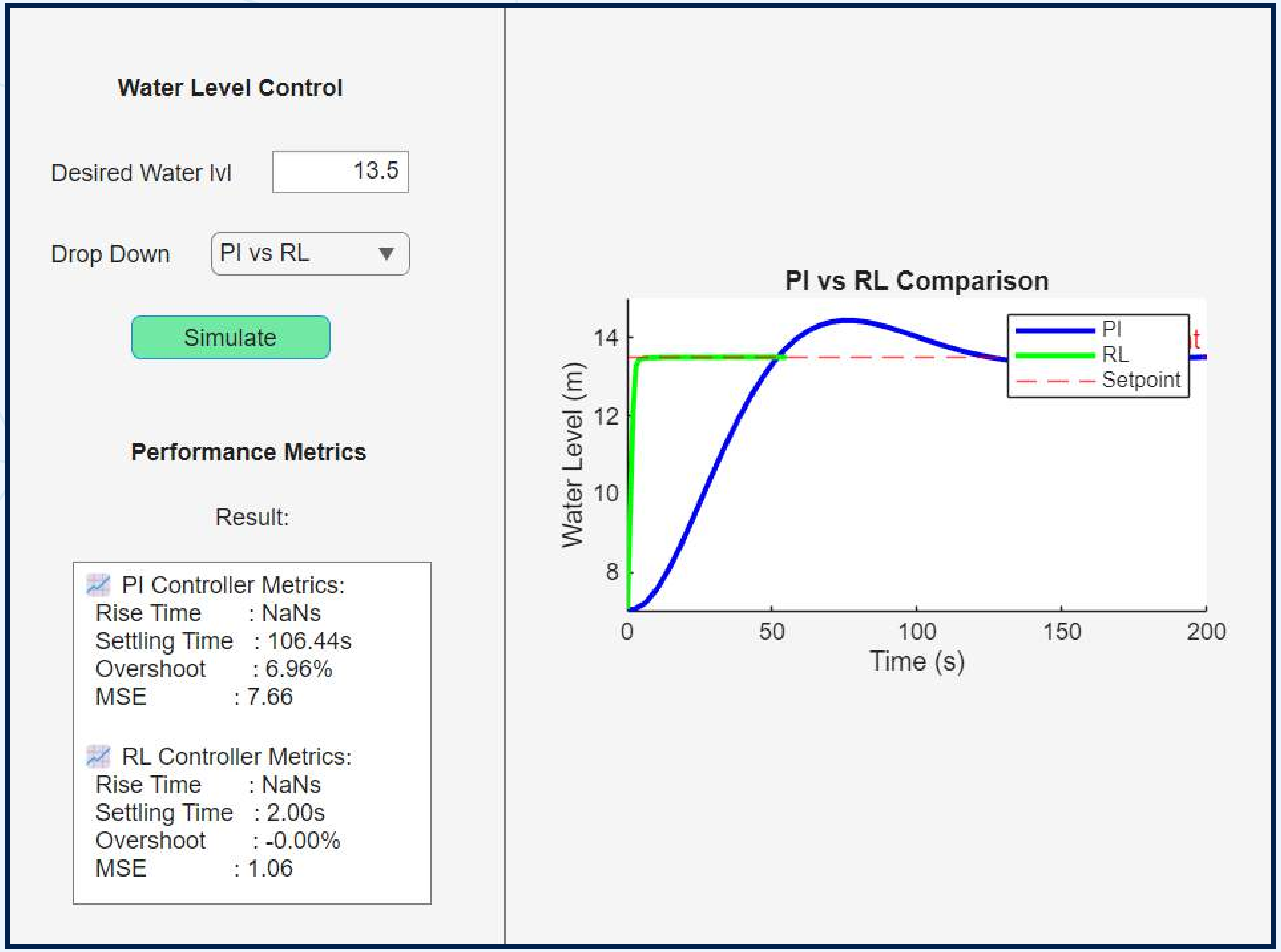


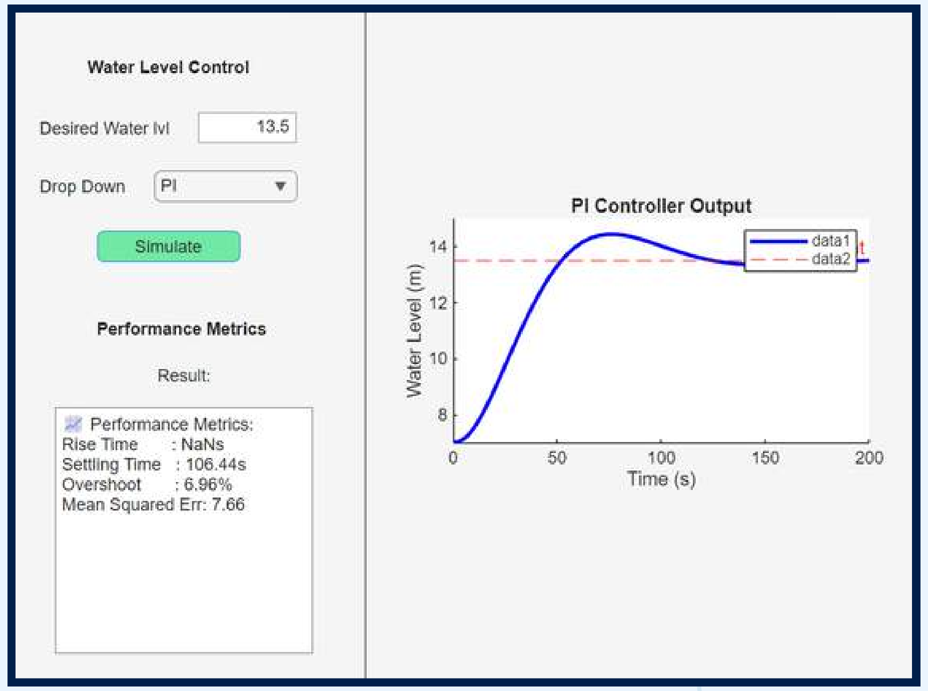
Comparison between RL and PI Controller

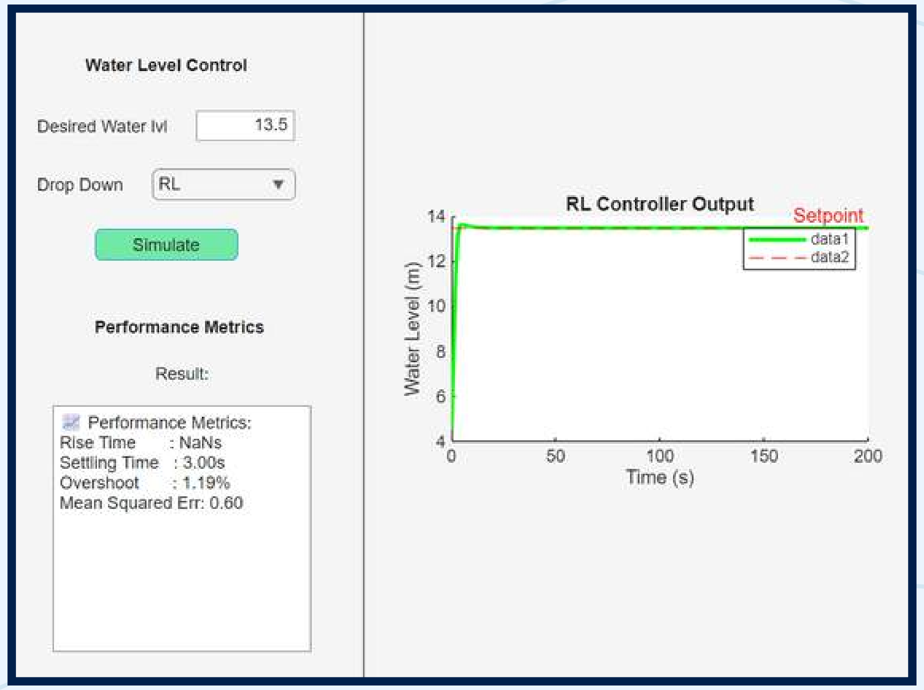
Controller	Advantages	Disadvantages
PI Controller	- Simple to design and tune - Predictable behaviour - Low overshoot	- Very slow response - Higher average error (MSE) - Not adaptive to changes in dynamics
RL Controller	- Fast and accurate tracking - Extremely low MSE - Adaptive to environment	- Requires extensive training -Complex to design and tune -Less explainable logic

- PI Controller: Reliable and easy to implement, but responds slowly and has higher error due to limited adaptability.
- RL Controller: Delivers fast, precise, and adaptive control with minimal error, but requires complex setup, long training, and careful design to ensure safe performance.

GUI Simulation Result - via MATLAB app







THANK YOU!

gulduck final!