FRA503: Deep Reinforcement Learning Homework 0

- Chantouch Orungrote 66340500011
- Sasish Keawsing 66340500076

1. Part 1: Look at Cartpole RL Agent

1.1. Train the Cartpole RL Agent

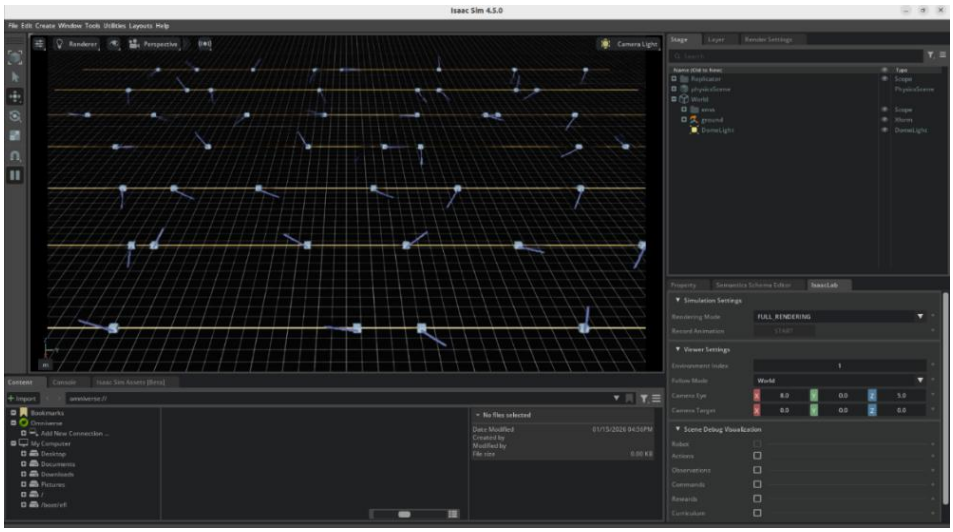

Figure 1 Exploration of Isaac-Cartpole-v0

The first figure illustrates a simulation environment in Isaac Sim, where a cartpole interacts with a grid-based environment. The visual representation includes the agent, grid, and environmental elements such as the camera light. This environment serves as the setup for model training, and alterations to reward functions or training parameters will influence how the agent behaves within the simulated space.

1.2. Visualize the Result

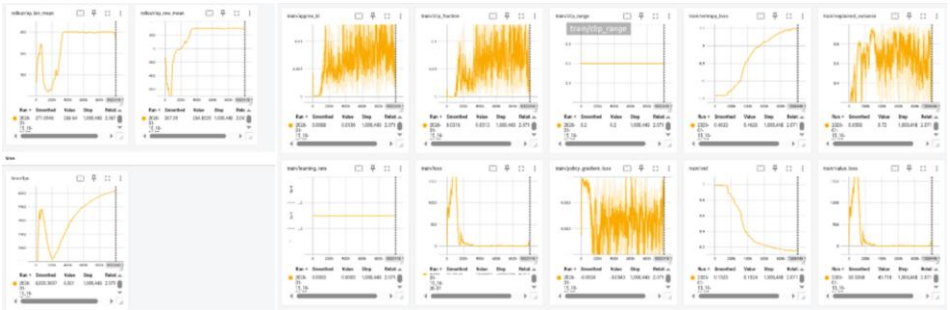

Figure 2 TensorBoard Visualization

Figure 2 represents plots from tensor board, providing insights into key performance metrics like cumulative reward during the agent's training. These graphs reflect different aspects of the agent's behavior.

In Part 2, both IsaacSim and TensorBoard are used to analyze the agent's behavior. IsaacSim simulates the agent’s interactions within the environment, while TensorBoard tracks performance metrics. Together, they help evaluate how changes in reward weights impact the agent’s actions and performance, identifying any improvements or degradations.

1.3. Part 1 Questionnaires

Question 1: According to the tutorials, if we want to edit the environment configuration, action space, observation space, reward function, or termination condition of the Isaac-Cartpole-v0 task, which file should we look at, and where is each part located?

Answer: IsaacLab/source/isaaclab_tasks/isaaclab_tasks/manager_based/classic/cartpole/cartpole_env_cfg.py

```python
@configclass
class CartpoleEnvCfg(ManagerBasedRLEnvCfg):
    """Configuration for the cartpole environment."""

    # Scene settings
    scene: CartpoleSceneCfg = CartpoleSceneCfg(num_envs=4096, env_spacing=4.0)
    # Basic settings
    observations: ObservationsCfg = ObservationsCfg()
    actions: ActionsCfg = ActionsCfg()
    events: EventCfg = EventCfg()
    # MDP settings
    rewards: RewardsCfg = RewardsCfg()
    terminations: TerminationsCfg = TerminationsCfg()

    # Post initialization
    def __post_init__(self) -> None:
        """Post initialization."""
        # general settings
        self.decimation = 2
        self.episode_length_s = 5
        # viewer settings
        self.viewer.eye = (8.0, 0.0, 5.0)
        # simulation settings
        self.sim.dt = 1 / 120
        self.sim.render_interval = self.decimation
```

Figure 3 Cartpole Environment Configurations

```python
@configclass
class ActionsCfg:
    """Action specifications for the MDP."""

    joint_effort = mdp.JointEffortActionCfg(asset_name="robot", joint_names=["slider_to_cart"], scale=100.0)
```

Figure 4 Action Space Configurations

```python
@configclass
class ObservationsCfg:
    """Observation specifications for the MDP."""

    @configclass
    class PolicyCfg(ObsGroup):
        """Observations for policy group."""

        # observation terms (order preserved)
        joint_pos_rel = ObsTerm(func=mdp.joint_pos_rel)
        joint_vel_rel = ObsTerm(func=mdp.joint_vel_rel)

        def __post_init__(self) -> None:
            self.enable_corruption = False
            self.concatenate_terms = True

    # observation groups
    policy: PolicyCfg = PolicyCfg()
```

Figure 5 Observation Space Configurations

```python
@configclass
class RewardsCfg:
    """Reward terms for the MDP."""

    # (1) Constant running reward
    alive = RewTerm(func=mdp.is_alive, weight=1.0)
    # (2) Failure penalty
    terminating = RewTerm(func=mdp.is_terminated, weight=-2.0)
    # (3) Primary task: keep pole upright
    pole_pos = RewTerm(
        func=mdp.joint_pos_target_l2,
        weight=-1.0,
        params={"asset_cfg": SceneEntityCfg("robot", joint_names=["cart_to_pole"]), "target": 0.0},
    )
    # (4) Shaping tasks: lower cart velocity
    cart_vel = RewTerm(
        func=mdp.joint_vel_l1,
        weight=-0.01,
        params={"asset_cfg": SceneEntityCfg("robot", joint_names=["slider_to_cart"])},
    )
    # (5) Shaping tasks: lower pole angular velocity
    pole_vel = RewTerm(
        func=mdp.joint_vel_l1,
        weight=-0.005,
        params={"asset_cfg": SceneEntityCfg("robot", joint_names=["cart_to_pole"])},
    )
```

Figure 6 Reward Configurations

```python
@configclass
class TerminationsCfg:
    """Termination terms for the MDP."""

    # (1) Time out
    time_out = DoneTerm(func=mdp.time_out, time_out=True)
    # (2) Cart out of bounds
    cart_out_of_bounds = DoneTerm(
        func=mdp.joint_pos_out_of_manual_limit,
        params={"asset_cfg": SceneEntityCfg("robot", joint_names=["slider_to_cart"]), "bounds": (-3.0, 3.0)},
    )
```

Figure 7 Termination Configurations

Question 2: What are the action space and observation space for an agent defined in the Isaac-Cartpole-v0 task?

Answer: According to the ActionsCfg and ObservationsCfg

- Action space: The agent outputs a single continuous value that is applied as a force (joint_effort) to the cart's horizontal joint (slide_to_cart) which is scaled by 100.0

- Observation space: A continuous 4-dimensional space formed by concatenating joint (self.concatenate_terms = True):
    - Positions: cart position (x) and pole angle (θ) from joint_pos_rel.
    - Velocities: cart velocity $(\dot{x})$ and pole angular velocity $(\dot{\theta})$ from joint_vel_rel.

Question 3: How can episodes in the Isaac-Cartpole-v0 task be terminated?

Answer: According to the TerminationsCfg

- Time out: The episode ends automatically when the max episode length is reached.
- Cart out of bounds: The episode terminates if the cart's horizontal position (slider_to_cart) goes outside the range −3.0 to 3.0.

Question 4: How many reward terms are used to train an agent in the Isaac-Cartpole-v0 task?

Answer: According to the RewardCfg, there are 5 reward terms with default values included.

- **(Weight +1.0) Constant running reward:** Gives a small positive reward at every step if the task is not finished, encouraging the agent to keep the system running.
- **(Weight -2.0) Termination penalty:** Gives a negative reward when the episode ends or actions that cause failure, which is cart out of bound.
- **(Weight -1.0) Pole position penalty (Primary task):** Penalizes the agent when the pole moves away from the upright position.
- **(Weight -0.01) Cart velocity penalty (Shaping tasks):** Penalizes the cart for moving too fast.
- **(Weight -0.005) Pole angular velocity penalty (Shaping tasks):** Penalizes the pole for rotating too quickly.

2. Part 2: Playing with Cartpole RL Agent

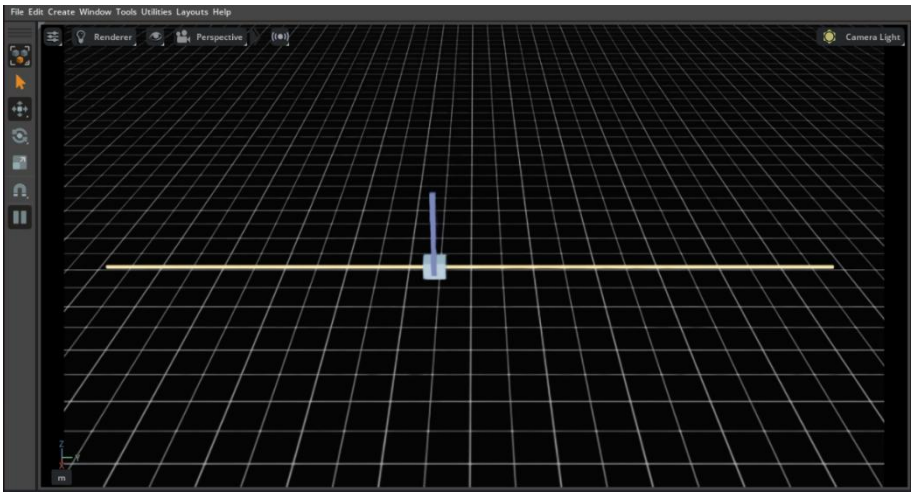

Figure 8 Cartpole Model Visualization at 448,000 Steps of Original Setup

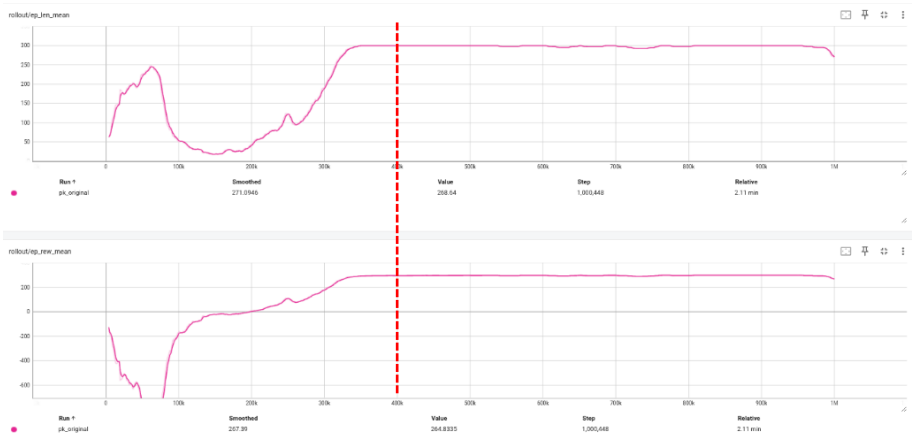

Figure 9 Progress Default Model 448,000 Steps of the Model Showing Mean Episode Length and Cumulative Reward

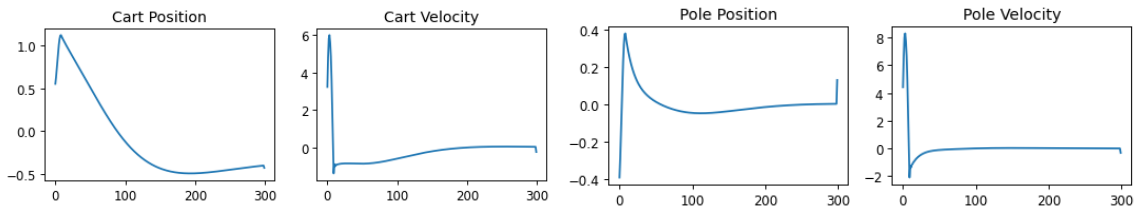

Figure 10 Cartpole Position and Velocity vs. Timestep of the Default Model

In this section, we will analyze the cartpole system's behavior with adjusted reward weights. TensorBoard will track metrics like rollout/ep_rew_mean and rollout/ep_len_mean, while Matplotlib will visualize key variables such as $\mathbf{x}, \dot{\mathbf{x}}, \theta$ and $\dot{\theta}$

2.1. Experiment Objective

Guided by the **Reward Hypothesis**, which states that all AI goals are defined by maximizing a total reward, these experiments test **"how changing specific reward weight shapes the agent's behavior and performance in Isaac Sim".** We compare two extremes: **non-significant (0.0)**, where a goal is removed, and **highly significant (x10)**, where a goal dominates the agent's actions. This proves that an agent's behavior and control style are direct results of its reward settings. Full details of reward changing setup are in Section 2.2.1.

2.2. Model and Parameters Setup

The model at 448,000 steps was selected as the baseline for this experiment. At this stage, the agent's cumulative reward reached its maximum potential, and its physical behavior was fully aligned with the defined goals. This point was chosen because **it represents the great performance and stable behavior observed during the original training process.**

2.2.1. Reward Weight

The reward will be only for the interest reward terms and others will be the same as the default of the model 448,000 steps. The experiment involves adjusting the reward weight in 2 conditions:

- **0.0:** When the reward term is removed entirely by multiply 0.
- **x10:** When the reward term is significantly increased, the **default weight multiplied by 10.**

2.2.2. Observed Timestep

The timestep for each experiment will be selected based on the time required for the default model 448,000 steps. Which will be observed over a **single episode consisting of 300 timesteps**, And the single episode is calculated from the formula below, according to figure 3:

$$\text{single episode timesteps} = \frac{\text{self. episode\_length\_s}}{\text{self. sim. dt} \times \text{self. decimation}}$$

2.3. Experiment 1: Adjust the Reward Weight of Staying Alive

2.3.1. Hypothesis

- With a reward weight of **0.0**, the agent will lack the fundamental motivation, resulting in immediate failure as there is no reward to counteract the termination penalty which is -2.0.
- With a reward weight of **+1.0**, the agent will be trying to stay alive cause from the revenue reward from every steps it still alive in this simulation.
- With a reward weight of **+10.0**, the agent will prioritize episode duration so aggressively that it will ignore the physical style penalties, leading to chaotic.

2.3.2. Define Variables

Independent Variable	Dependent Variables	Control Variables
• Alive Reward Weight (w_{alive})	• Agent's Performance (Total Reward) • Agent's Behavior (Physical Stability) • Stabilization Time	• Model (448,000) • Timestep (300) • Other Reward Weights

2.3.3. Training Results Visualization

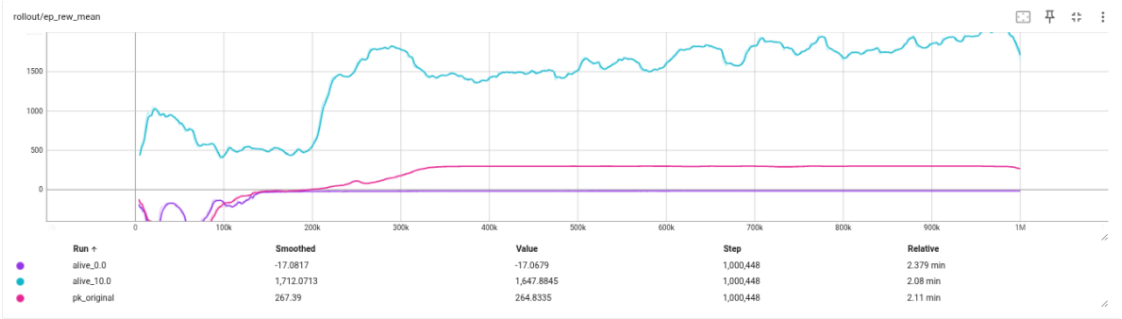

Figure 11 Reward vs. Timestep Graph for Experment 1

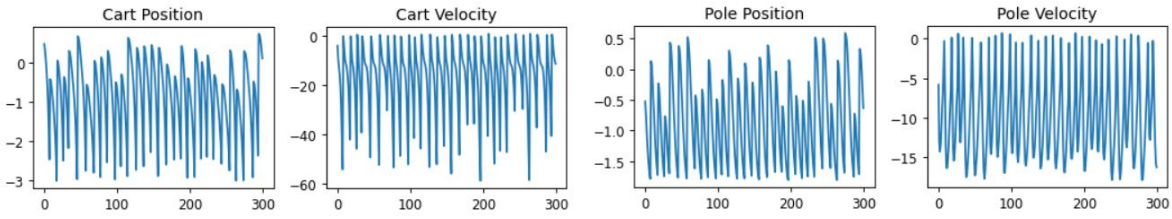

Figure 12 Cartpole Position and Velocity vs. Timestep for a Alive Reward Weight of 0.0

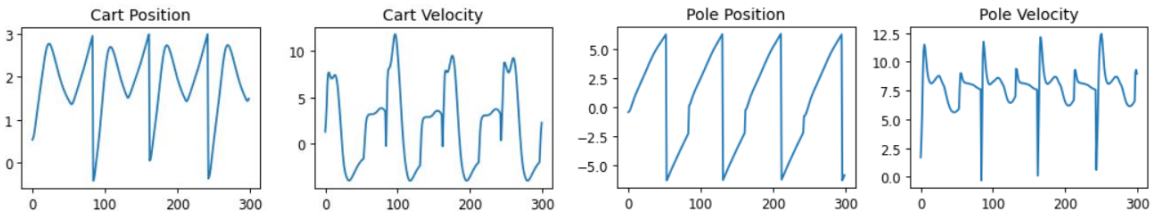

Figure 13 Cartpole Position and Velocity vs. Timestep for a Alive Reward Weight of +10.0

2.3.4. Comparative Analysis of Alive Reward Scaling

2.3.4.1. Condition 1: Weight 0.0

- **Reward Per Timesteps Graph Analysis:** The purple reward curve for 0.0 remains flat and near zero throughout training. And the mean episode length remains extremely low, indicating the agent never learns to survive or fail.
- **IsaacSim Physical Observation Analysis:** The observation plots show high frequency oscillations that lead to immediate failure. Without a positive incentive to remain active, **the agent cart slides to the left side almost every time they spawn cause from lacking a clear reason to resist the termination penalty which is the highest value (-2.0)**, resulting in an easy but immediate surrender to failure.

2.3.4.2. Condition 2: Weight +1.0 (Baseline)

- **Reward Per Timesteps Graph Analysis:** The baseline or pink curve shows a steady learning trajectory, reaching maximum episode length after approximately 400k steps.
- **IsaacSim Physical Observation Analysis:** This weight establishes a healthy motivation for the agent to balance the pole. It creates an easy baseline where the survival reward outweighs minor shaping penalties, allowing the agent to achieve a stable control steady state.

2.3.4.3. Condition 3: Weight +10.0

- **Reward Per Timesteps Graph Analysis:** The light blue curve shows rapid growth in reward but displays high variance and spiky behavior.
- **IsaacSim Physical Observation Analysis:** The state plots reveal erratic behavior where the massive survival reward causes the agent to **prioritize longevity over stability**. For example, the agent attempts to balance the pole, but if a correction risk sliding the cart out of bounds, **it abandons the pole to stay within the safety zone.** This results in a reckless policy that maximizes numerical scores through survival at all costs while remaining physically unstable.

2.3.5. Conclusion

The experiment makes it clear that the alive reward is the core reward of the agent's motivation cause of the reward feeding in every step.

- With a reward weight of **0.0**, the agent demonstrates no incentive to counteract gravity or avoid termination and slide only one direction, resulting in immediate failure and extremely short episode durations, as originally predicted.
- With a reward weight of **+1.0 (baseline)**, the agent has balance prioritize, it makes the pole upright, smooth cart movement, which is the same as hypothesis predicted.
- With a reward weight of **+10.0**, the agent excessively prioritizes episode longevity, producing highly erratic and unstable motions as the dominant survival reward overwhelms the physical shaping penalties, matching the hypothesized chaotic behavior.

2.4. Experiment 2: Adjust the Reward Weight of Termination

2.4.1. Hypothesis

- With a reward weight of **0.0**, the agent will lack a significant negative incentive to avoid failure, potentially leading to a policy that does not prioritize long-term survival when exploration becomes difficult.
- With a reward weight of **-2.0**, the agent will be trying to avoid terminating but still doing the shaping task or making the pole upright and balancing it.
- With a reward weight of **-20.0**, the agent will be under extreme pressure to avoid termination, which may result in a very conservative control style or a significantly faster stabilization time to ensure the pole stays within safe bounds at all costs.

2.4.2. Define Variables

Independent Variable	Dependent Variables	Control Variables
• Termination Penalty Weight (w_{term})	• Agent's Performance (Total Reward) • Agent's Behavior (Physical Stability) • Stabilization Time	• Model (448,000) • Timestep (300) • Other Reward Weights

2.4.3. Training Results Visualization

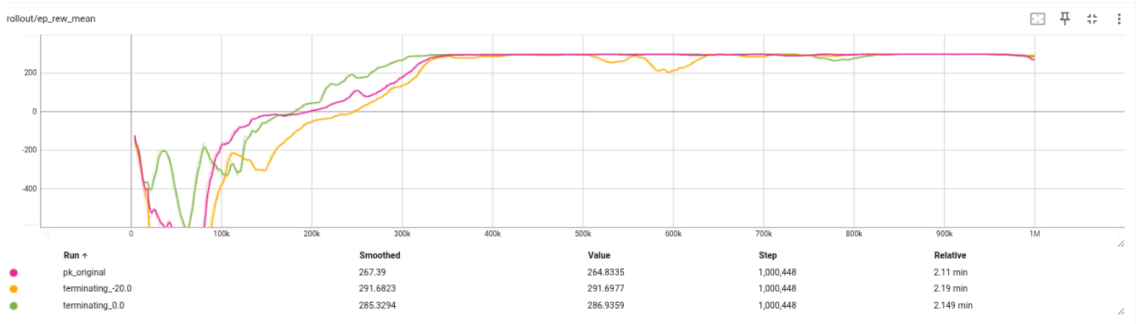

Figure 14 Reward vs. Timestep Graph for Experment 2

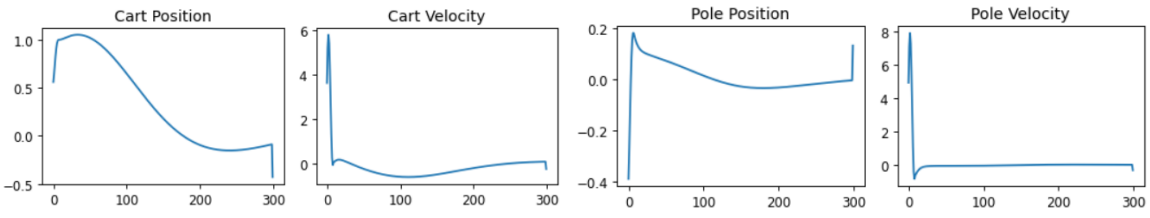

Figure 15 Cartpole Position and Velocity vs. Timestep for a Termiantion Penalty Weight of 0.0

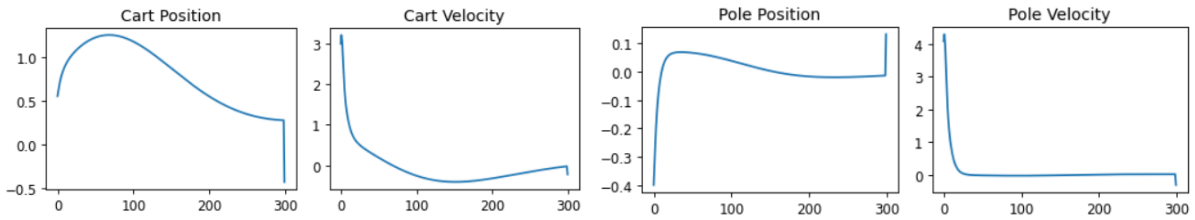

Figure 16 Cartpole Position and Velocity vs. Timestep for a Termiantion Penalty Weight of -20.0

2.4.4. Comparative Analysis of Termination Penalty Scaling

2.4.4.1. Condition 1: Weight 0.0

- **Reward Per Timesteps Graph Analysis:** The green reward and episode length curves show a very fast initial rise and fall, which cause from the no termination cost. But encounter a significant decreasing graph and instability around 200k steps. The agent eventually recovers but lacks the steady pressure for reliable survival seen in the baseline.
- **IsaacSim Physical Observation Analysis:** The observation plots reveal that the agent is capable of balancing, but the motion is less refined. Without a penalty for failing, it is clear that the agent takes an easy but less disciplined approach to control, **leading to wider swings in pole position before correction at first period of the graph.**

2.4.4.2. Condition 2: Weight -2.0 (Baseline)

- **Reward Per Timesteps Graph Analysis:** The pink baseline curve provides a more consistent learning path with a smoother transition toward maximum survival time compared to the null penalty.
- **IsaacSim Physical Observation Analysis:** This weight creates an easy and balanced pressure for the agent. It establishes enough of a consequence for failure that the agent learns to prioritize the primary balancing task while maintaining the smoothness provided by shaping weights.

2.4.4.3. Condition 3: Weight -20.0

- **Reward Per Timesteps Graph Analysis:** The orange curve shows a much more volatile learning process with several sharp drops in performance as the agent explores. Each failure is so costly that it creates a noisy training signal, though it ultimately reaches the survival ceiling.
- **IsaacSim Physical Observation Analysis:** The resulting policy is extremely conservative. **The agent has learned to lock the pole into position very quickly to avoid the massive penalty**, resulting in the most rigid state plots of the three conditions and a very fast stabilization time for the pole angle.

2.4.5. Conclusion

The experimental results largely support the initial hypothesis, confirming that the termination penalty serves as the primary deterrent mechanism that shapes the agent's risk sensitivity and failure avoidance behavior.

- When the termination penalty is set to **0.0**, the agent exhibits reduced concern for failure, leading to less disciplined control behavior and **inconsistent performance**, particularly during early training, as the absence of negative reinforcement weakens long-term survival prioritization, which is almost correct according to the prediction.
- When the termination penalty of **-2.0**, the agent understands the goal and **avoids the termination modesty** which is the same as a hypothesis.
- When the termination penalty is increased to **-20.0**, the agent experiences extreme pressure to avoid failure, resulting in a **highly conservative control** policy characterized by rapid stabilization and rigid motion patterns, consistent with the hypothesized behavior. This is very consistent with the hypothesis.

2.5. Experiment 3: Adjust the Reward Weight Keep the Pole Upright

2.5.1. Hypothesis

- With a reward weight of **0.0**, the agent will minimize movement to avoid penalties, **focusing on staying alive without prioritizing pole stability, which should be free falls the pole.**
- With a reward weight of **-1.0**, the agent will be trying to keep the pole upright but not forget to avoid the termination, stay alive, and move cart and pole slowly.
- With a reward weight of **-10.0**, the agent will focus on keeping the pole upright to avoid the penalty, but this may lead to more aggressive adjustments, **increasing the risk of termination.**

2.5.2. Define Variables

Independent Variable	Dependent Variables	Control Variables
• Keep the Pole Upright Penalty Weight (w_{pole_pos})	• Agent's Performance (Total Reward) • Agent's Behavior (Physical Stability) • Stabilization Time	• Model (448,000) • Timestep (300) • Other Reward Weights

2.5.3. Training Results Visualization

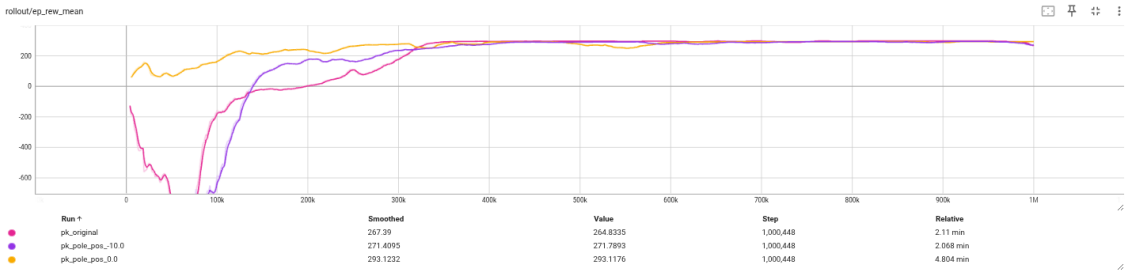

Figure 17 Reward vs. Timestep Graph for Experiment 3

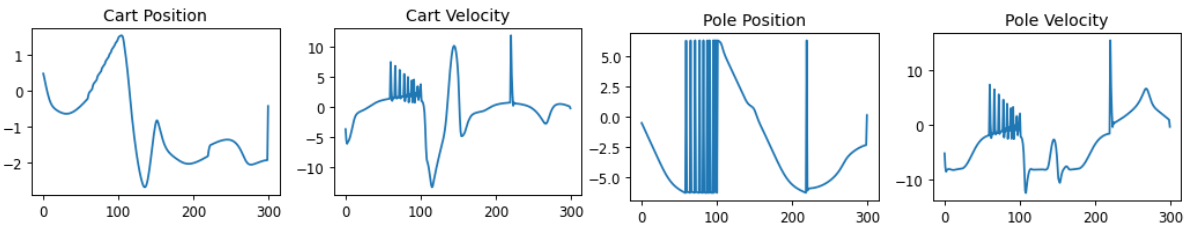

Figure 18 Cartpole Position and Velocity vs. Timestep for a Keep Pole Upright Reward Weight of 0.0

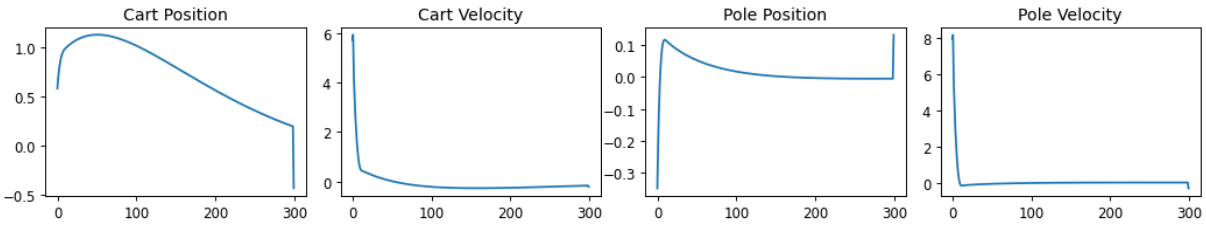

Figure 19 Cartpole Position and Velocity vs. Timestep for a Keep Pole Upright Reward Weight of -10.0

2.5.4. Comparative Analysis of Keep the Pole Upright Penalty Scaling

2.5.4.1. Condition 1: Weight 0.0

- **Reward Per Timesteps Graph Analysis:** The yellow curve for 0.0 appears superior initially, maintaining a high, stable mean reward. This is a false positive in reinforcement learning, the agent is maximizing the alive reward (+1.0) without incurring any cost for pole deviation. **Which the reward and behavior are not make sense**
- **IsaacSim Physical Observation Analysis:** The state plots reveal chaotic oscillations. Because the penalty term for the pole position is nullified $(0.0 \times \text{penalty} = 0)$, the agent lacks the term to learn pole balancing. It purely optimizes for episode duration, resulting in a failed controller that cannot stabilize the pole despite high numerical scores

2.5.4.2. Condition 2: Weight -1.0 (Baseline)

- **Reward Per Timesteps Graph Analysis:** The pink curve starts with significant negative values. This indicates the agent is being penalized for initial failures, creating the necessary pressure to explore corrective actions
- **IsaacSim Physical Observation Analysis:** The agent achieves a steady state. The penalty is sufficient to prioritize the primary task while balancing the shaping rewards for velocity

2.5.4.3. Condition 3: Weight -10.0

- **Reward Per Timesteps Graph Analysis:** The purple curve shows the steepest recovery after the initial exploration phase. By increasing the penalty tenfold, the cost of deviation becomes the dominant signal in the loss function
- **IsaacSim Physical Observation Analysis:** This results in the most stable observation plots. The high weight forces the agent to minimize the pole angle with high precision, effectively tightening the control loop and reducing the steady-state error to the best it can.

2.5.5. Conclusion

- Contrary to the initial hypothesis, setting the reward weight to 0.0 does not cause the agent to allow the pole to simply collapse. Instead, **the agent continues to swing and attempts to stabilize the pole. This behavior is driven by the termination terms**, since doing nothing would cause the cart to slide out of bounds, the agent erratically maneuvers the cart back toward the center to avoid failure. Consequently, the agent maintains active control despite having no explicit penalty for pole deviation, **which differs from the predicted outcome.**
- When the reward weight of **-1.0**, the agent shows a great reaction to the goal, they have a good stabilize and explore.
- When the reward weight is increased to **-10.0**, the penalty becomes the dominant objective, forcing the agent to minimize pole angle deviation with high precision and resulting in the most stable and well physical behavior observed in this experiment.

2.6. Experiment 4: Adjust the Reward Weight of Lowering the Cart Velocity

2.6.1. Hypothesis

- With a reward weight of **0.0**, the agent will not prioritize reducing cart velocity, resulting in **larger and more variable movements.**
- With a reward weight of **-0.01**, the agent will act smoothly on the cart movement and avoid the termination.
- With a reward weight of **-0.1**, the agent will strongly focus on minimizing cart velocity, leading to **more constrained and conservative movements**, which may reduce overall stability or negatively affect pole balance

2.6.2. Define Variables

Independent Variable	Dependent Variables	Control Variables
• Lower Cart Velocity Penalty Weight (w_{cart_vel})	• Agent's Performance (Total Reward) • Agent's Behavior (Physical Stability) • Stabilization Time	• Model (448,000) • Timestep (300) • Other Reward Weights

2.6.3. Training Results Visualization

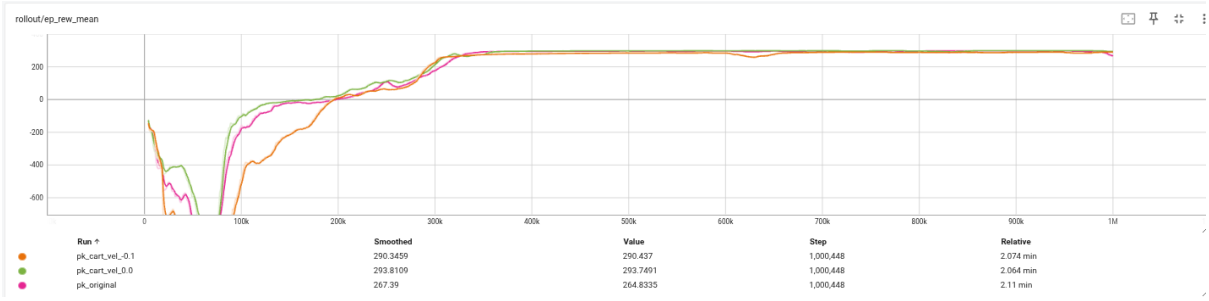

Figure 20 Reward vs. Timestep Graph for Experiment 4

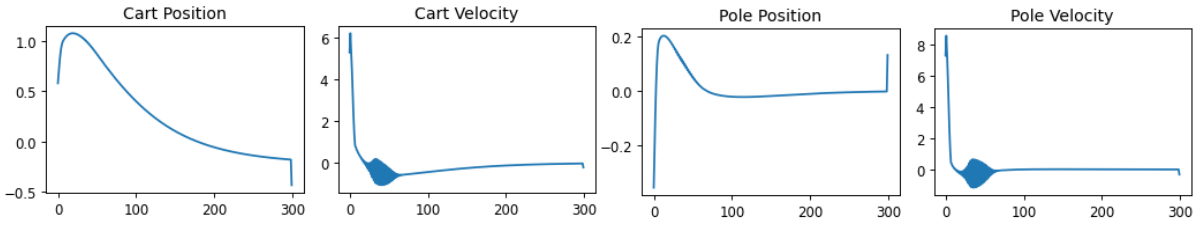

Figure 21 Cartpole Position and Velocity vs. Timestep Graph for a Cart Velocity Reward Weight of 0.0

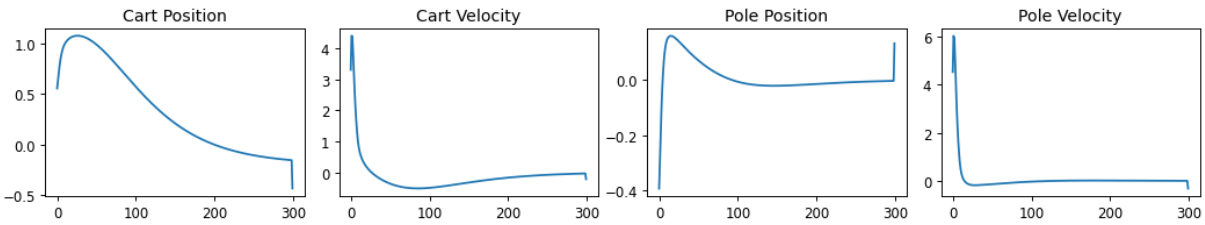

Figure 22 Cartpole Position and Velocity vs. Timestep Graph for a Cart Velocity Reward Weight of -0.1

2.6.4. Comparative Analysis of Lowering the Cart Velocity Penalty Scaling

2.6.4.1. Condition 1: Weight 0.0

- **Reward Per Timesteps Graph Analysis:** The green curve **converges quickly to the highest reward. Because movement costs nothing**, the agent is not penalized for high-speed adjustments.
- **IsaacSim Physical Observation Analysis:** The agent displays aggressive, high-variance movements. Without a velocity constraint, t**he agent over-corrects to balance the pole, resulting in noisy velocity graphs with high peaks.** The cart position in this instance exhibits no fluctuations, as the pole's initial position is already upright. As a result, the cart moves rapidly in a single direction, which is reflected in the observed curve.

2.6.4.2. Condition 2: Weight -0.01 (Baseline)

- **Reward Per Timesteps Graph Analysis:** The pink curve shows a standard learning trajectory, settling slightly lower than the 0.0 case.
- **IsaacSim Physical Observation Analysis:** This weight acts as a subtle shaping reward. **It encourages calmer pole motion and smoother cart transitions** without being so restrictive that it hinders the primary balancing task. It strikes a balance between stability and the freedom to move.

2.6.4.3. Condition 3: Weight -0.1

- **Reward Per Timesteps Graph Analysis:** The orange curve takes the longest to recover, showing a deeper dip during the exploration phase. The agent must work harder to find a policy that balances the pole while staying under the strict speed limit.
- **IsaacSim Physical Observation Analysis:** The agent develops a conservative way. T**he cart remains almost stationary in the position graph because the high velocity penalty discourages any significant travel.**

2.6.6. Conclusion

- When the reward weight is set to **0.0**, the agent exhibits no incentive to limit cart speed, resulting in aggressive, high-variance movements characterized by rapid over-corrections and noisy velocity profiles, as originally anticipated, which same as a predicted.
- When the reward weight is at **-0.01**, the agent acts regular cart movement with the pole stabilizing because **the penalty for cart movement is minimal, the agent is not discouraged from utilizing the full length of the track to maintain balance, leading to a very easy and successful convergence.**
- When the reward weight is increased to **-0.1**, the velocity penalty becomes a dominant constraint, **significantly restricting cart movement and forcing the agent into a highly conservative policy that minimizes motion even when corrective action would be beneficial.**

2.7. Experiment 5: Adjust the Reward Weight of Lowering the Pole Angular Velocity

2.7.1. Hypothesis

- With a reward weight of **0.0**, the agent will not prioritize reducing the pole's angular velocity, which may result in more frequent oscillations and larger corrective actions
- With a reward weight of **-0.005**, the agent will not prioritize reducing the pole's angular velocity as much, **which the agent will not focus on the pole's velocity but making it stabilize and take focus on taking effort on the cart will be better priorities.**
- With a reward weight of **-0.05**, the agent will strongly focus on minimizing the pole's angular velocity, **leading to smoother and more damped pole motion**. However, this may cause slower reactions to disturbances and could reduce overall task performance

2.7.2. Define Variables

Independent Variable	Dependent Variables	Control Variables
• Keep the Pole Upright Penalty Weight (w_{pole_vel})	• Agent's Performance (Total Reward) • Agent's Behavior (Physical Stability) • Stabilization Time	• Model (448,000) • Timestep (300) • Other Reward Weights

2.7.3. Training Results Visualization

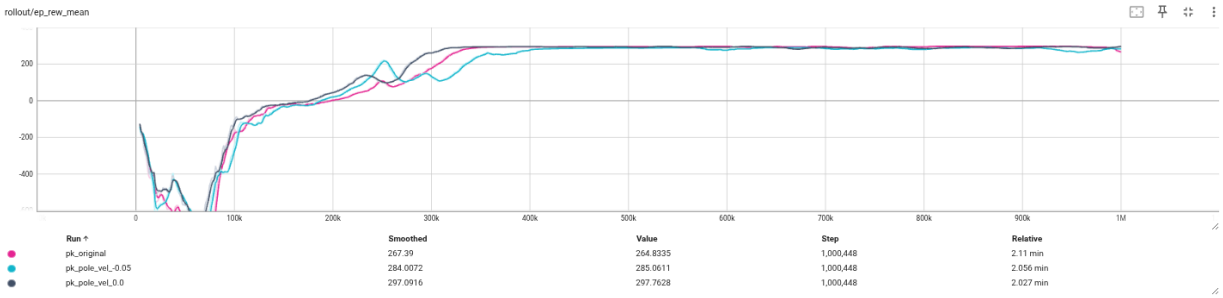

Figure 23 Reward vs. Timestep Graph for Experiment 5

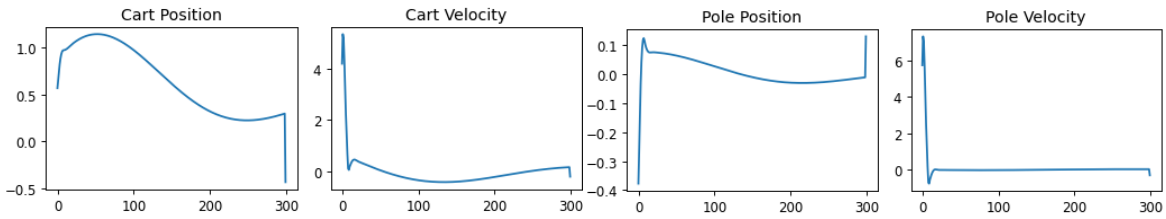

Figure 23 Cartpole Position and Velocity vs. Timestep Graph for a Pole Velocity Reward Weight of 0.0

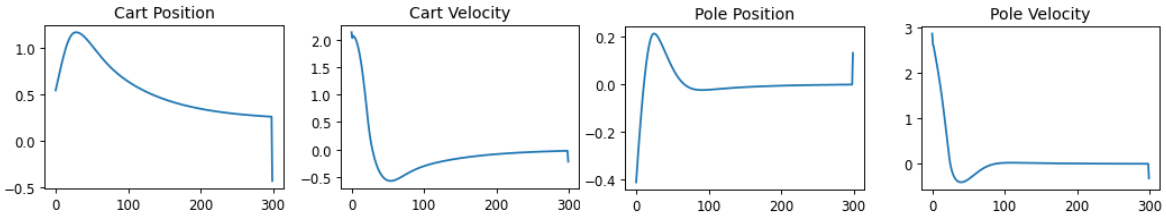

Figure 24 Cartpole Position and Velocity vs. Timestep Graph for a Pole Velocity Reward Weight of -0.05

2.7.4. Comparative Analysis

2.7.4.1. Condition 1: Weight 0.0

- **Reward Per Timesteps Graph Analysis:** The dark blue curve for 0.0 reaches a high reward quickly. Like previous null-weight experiments, this represents a clear path of least resistance where the agent is not penalized for erratic pole rotations.
- **IsaacSim Physical Observation Analysis:** The agent displays **more frequent oscillations and larger corrective actions**. Because there is no cost for the pole spinning quickly, the control loop is noisy, leading to less efficient stabilization.

2.7.4.2. Condition 2: Weight -0.005 (Baseline)

- **Reward Per Timesteps Graph Analysis:** The pink curve represents the standard baseline, showing a steady learning trend and reliable stabilization
- **IsaacSim Physical Observation Analysis:** This weight provides an easy balance between speed and stability. It encourages enough dampening to keep the motion smooth while allowing the agent to move fast enough to catch the pole if it begins to fall.

2.7.4.3. Condition 3: Weight -0.05

- **Reward Per Timesteps Graph Analysis:** The blue curve shows a lower overall reward and a more complex path to stabilization. The high penalty for angular velocity makes the agent fear moving the pole too quickly**.**
- **IsaacSim Physical Observation Analysis**: This results in highly damped, slow-motion pole behavior. While the motion is clear and smooth, it creates a risk: the agent may react too slowly to sudden disturbances, potentially reducing overall task performance because it prioritizes calmness over aggressive recovery**.**

2.7.6. Conclusion

- When the reward weight is set to **0.0**, the agent exhibits no incentive to suppress rapid pole rotation, resulting in frequent oscillations and large corrective actions, producing a noisy and inefficient stabilization process as anticipated.
- When the reward weight is set to **-0.005**, the agent **tries to swing the pole upright more than concern about the pole velocity** because the reward weight is less significant.
- When the reward weight increases to **-0.05**, the agent **starts concern to prioritize minimizing angular velocity as the pole velocity in observation plot,** leading to highly smooth and visually stable pole motion.

3. Part 3: Mapping RL Fundamentals

3.1. Part 3 Questionnaires

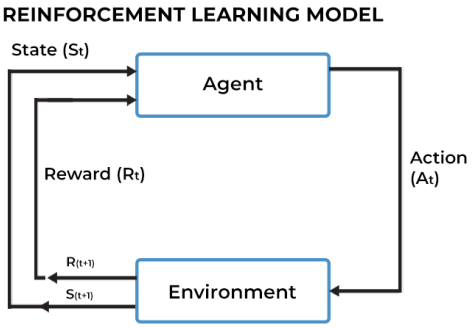

Figure 25 Reinforcement Learning Model

Question 1: What is reinforcement learning and its components according to your understanding? Giving examples of each component according to the diagram consider the Cartpole problem

Answer: Reinforcement Learning is a branch of machine learning where an agent interacts with an environment by observing the state, taking an action, and receiving a reward, with the goal of selecting the actions to maximizing total future reward.

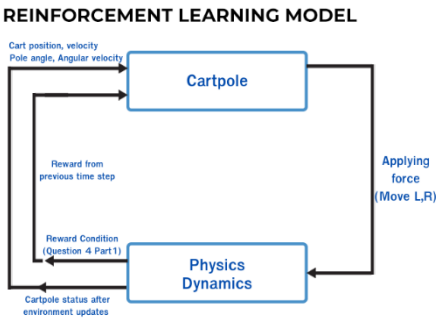

Figure 26 Reinforcement Learning Model of Cartpole

- **Agent:** the controller that decides how to push the cart
- **State** (S_t)**:** cart position, cart velocity, pole angle, and pole angular velocity
- **Action** (A_t)**:** applying a force to the cart
- **Environment:** cart, pole, gravity, and physics dynamics
- **Reward** (R_{t+1})**:** positive reward for keeping the pole upright, penalty if it the pole moves away from the upright position (fall), etc.
- **Next State** (S_{t+1})**:** the new state after the environment updates

Question 2: What is the difference between the reward, return, and the value function?

Answer:

Concept	Temporal Scope	Provided By	Function
Reward (R_{t+1})	One step	Environment	Feedback signal
Return (G_t)	Overall steps	Reward	Optimization objective
Value Function $(V_\pi(S))$	Expected future steps	Agent	Decision guidance

Definition

- **Reward** (R_{t+1})**:** The immediate scalar feedback signal received after taking an action
- **Return** (G_t)**:** Total cumulative reward over time
- **Value Function** $(V_\pi(S))$**:** Estimates the expected return of being in a particular state

Question 3: Consider policy, state, value function, and model as mathematical functions, what would each one take as input and output?

Answer:

Concept	Mathematical Function	Input	Output
Policy	$\pi(s) = A$	State (S_t)	Action (A_t)
Environment State	$S_t = O_t$	Environment (O_t)	State (S_t)
Agent State	$S_{t+1} = u(S_t, A_t, R_{t+1}, O_{t+1})$	History (u)	Next State (S_{t+1})
Value Function	$v_\pi(s) = \mathbb{E}[G_t \mid S_t = s, \pi]$	State (S_t)	Expected Return
Model (State & Reward)	$P(s, a, s') \approx p(s' \mid S_t = s, A_t = a)$	State (S_t), Action (A_t)	Expected Next State (s')
	$R(s, a) = \mathbb{E}[R_{t+1} \mid S_t = s, A_t = a]$		Expected Reward (R)