Section 1

Reinforcement Learning Mini Project Report

1. Introduction

The objective of this mini-project was to apply reinforcement learning (RL) algorithms to robotic control tasks using the Panda-Gym simulation environment. Two tasks were addressed:

- PandaReach: The robot must move its end-effector to a target location.
- PandaPush: The robot must push an object to a goal location.

We implemented and trained agents using Deep Deterministic Policy Gradient (DDPG), a popular off-policy RL algorithm for continuous control. The project also required monitoring learning progress, evaluating trained agents, and analyzing success rates.

2. Methodology

2.1 Tools and Libraries

- Python 3.11
- Gymnasium (instead of Gym, since we used Panda-Gym v3)
- Panda-Gym v3 (robotics environments)
- Stable-Baselines3 (SB3) for RL algorithms
- PyBullet for physics simulation
- Matplotlib / Pandas for plotting and analysis

2.2 Algorithm: DDPG

- DDPG is an actor-critic algorithm for continuous action spaces.
- Uses two neural networks:
  - Actor: Outputs continuous actions.
  - Critic: Estimates Q-values.
- Trains off-policy using an experience replay buffer and target networks.
- We added action noise (Gaussian) to encourage exploration.

2.3 Training Setup

- Environment IDs: PandaReach-v3, PandaPush-v3
- Reward function: Provided by Panda-Gym (dense reward based on distance to goal).
- Timesteps:
  - Reach: 400,000
  - Push: 800,000 (harder task, more training needed)

2.4 Logging and Evaluation

- Used SB3 Monitor wrapper to log episodic rewards.
- Implemented a custom SuccessLoggerCallback to log success per episode.
- Evaluation performed with deterministic policy for 5–10 test episodes.
- Saved evaluation rollouts as GIF animations.

3. Experiments

3.1 Task 1: PandaReach

- Agent trained with DDPG + Gaussian noise.
- Training converged steadily after ~200k steps.
- The robot successfully learned to move its end-effector close to the target.
- Success rate exceeded 90% by the end of training.

3.2 Task 2: Evaluation of PandaReach

- Visual inspection confirmed that the agent consistently reached the goal.
- GIFs of evaluation runs were saved in eval_output/.
- Performance metrics showed stable high reward and success rate.

3.3 Task 3: PandaPush

- More challenging than reach due to object interaction.
- Required longer training (800k+ timesteps).
- Success rate increased slowly; tuning hyperparameters (larger replay buffer, different action noise) improved stability.
- Final success rate reached ~65%.

3.4 Task 4: Success Rate Plotting

- Success rates were logged per episode and plotted.
- Reach: Smooth convergence to ~95%.
- Push: Slower and less stable, showing learning difficulty.

4. Results

TaskTimestepsFinal Reward TrendSuccess RatePandaReach400kStable convergence~95%PandaPush800kNoisy, gradual~65%

Figures:

1. Reward Curves – Smoothed episodic rewards during training.
2. Success Rate Plots – Percentage of successful episodes over time.
3. GIFs – Demonstrating trained agents solving tasks.

5. Discussion

- DDPG worked well on Reach, where the task is relatively simple.
- Push task was harder: success rate plateaued, indicating limitations of DDPG.
- Improvements could include:
  - Using TD3, SAC, or TQC (more sample-efficient algorithms).
  - Applying Hindsight Experience Replay (HER) for sparse-reward tasks.
  - Longer training or curriculum learning for object manipulation.

6. Conclusion

This project demonstrated the application of DDPG to robotics control tasks in Panda-Gym. We successfully trained agents to solve the Reach task with high reliability and partially solved the Push task. The project highlighted:

- The strengths of DDPG for simple continuous control tasks.
- The need for more advanced algorithms for complex manipulation.
- The importance of logging, evaluation, and visualization for monitoring RL experiments.

7. Deliverables

- Code: Training, evaluation, plotting, and success-logging scripts.
- Logs: Training logs (logs/ folder).
- Models: Trained agents (models/ folder).
- Plots: Reward curves and success rate plots.
- GIFs: Visualizations of trained agents in action.

8. References

- Stable-Baselines3 documentation
- Panda-Gym documentation
- OpenAI Gymnasium documentation
- RL algorithms: Lillicrap et al., “Continuous control with deep reinforcement learning” (2015)