Course Project

Code of Honor. All external resources used in the project, including research papers, open-source repositories, datasets, and any content or code generated using AI tools, e.g., ChatGPT, GitHub Copilot, Claude, Gemini, must be *clearly cited* in the final submission. The final report must also include *a clear breakdown of individual group member contributions.* Any lack of transparency in the use of external resources or in reporting group contributions will be considered academic dishonesty and will significantly impact the final evaluation.

Topic	Deterministic versus Stochastic Policy Gradients: DDPG and SAC
Category	Deep RL from Scratch

Objective Design and implement two advanced actor–critic algorithms for continuous control: Deep Deterministic Policy Gradient (DDPG) and Soft Actor-Critic (SAC) [2, 1]. Compare their performance in terms of stability, sample efficiency, and robustness on continuous-action Gym environments.

Motivation DDPG and SAC represent two influential approaches to reinforcement learning with continuous action spaces. DDPG applies deterministic policy gradients combined with replay buffers and target networks, while SAC introduces maximum entropy regularization with stochastic policies, leading to improved stability and robustness. By implementing both from scratch, you will gain hands-on experience with advanced actor–critic architectures, off-policy training, and entropy-based exploration. These skills are highly valued in both research and applications such as robotics and control.

Requirements The final submission should address the following requirements while the details can be freely decided by the group members.

1. Implementation: in this respect, you should
   - Implement DDPG with replay buffer, target networks, and soft updates.
   - Implement SAC with stochastic actor, double Q-networks, and entropy regularization.
   - Use an actor–critic structure with separate networks for policy and value functions.
2. Environment modification: you can use a pre-implemented Gym environment that is *required to be controlled by Deep RL*. Note that basic environments that are handled easily by tabular RL are **not accepted**. To give the implementation some level of novelty, you **must** modify standard Gym environments with **at least one** of the following modifications:
   - Adding *noise* to observations.
   - *Perturbing* actions before execution.
   - Injecting irrelevant *distractor* features into the state vector.
   - Adding *stochasticity or delays* to rewards.
3. Evaluation: the final project should report key evaluation of the implemented algorithms in the modified environment. In this respect, the results should

- compare DDPG and SAC learning curves under normal and modified conditions,
- quantify stability, convergence speed, and sample efficiency, and
- report how environment modifications affect algorithm performance.

4. The results should be elaborated through
   - performing ablation studies, e.g., remove target networks in DDPG or investigate effect of entropy regularization in SAC, and
   - providing discussion on trade-offs between complexity and performance.

Milestones The following milestones are to be accomplished through semester.

1. Literature Review and Setup
   - Review DDPG and SAC.
   - Choose environments and finalize environment modifications.
2. Implementation
   - Implement DDPG with replay buffer and target networks.
   - Implement SAC with entropy-regularized objectives.
   - Validate both algorithms on simple continuous environments.
   - Train both on modified environments.
3. Evaluation and Analysis
   - Collect and plot learning curves.
   - Compare DDPG and SAC under normal and modified conditions.
   - Perform ablation experiments.
4. Final Report and Presentation

Submission Guidelines The main body of work is submitted through Git. In addition, each group submits a final paper and gives a presentation. In this respect, please follow these steps.

- Each group must maintain a Git repository, e.g., GitHub or GitLab, for the project. By the time of final submission, the repository should have
  - Well-documented codebase
  - Clear `README.md` with setup and usage instructions
  - A `requirements.txt` file listing all required packages or an `environment.yaml` file with a reproducible environment setup
  - Demo script or notebook showing sample input-output
  - *If applicable,* a `/doc` folder with extended documentation
- A final report (maximum *5 pages*) must be submitted in a PDF format. The report should be written in the provided formal style, including an abstract, introduction, method, experiments, results, and conclusion.
  Important: Submissions that do not use template are considered *incomplete.*
- A 5-minute presentation (maximum *5 slides including the title slide*) is given on the internal seminar on Week 14, i.e., *Dec 1 to Dec 5,* by the group. For presentation, any template can be used.

Final Notes While planning for the milestones please consider the following points.

1. You are encouraged to explore innovative approaches as long as the core objectives are met.
2. While computational resources are limited, the chosen environment can keep training feasible. Trade-offs (e.g., fewer episodes, smaller networks) are expected and should be justified.
3. Teams are expected to manage their computing needs and are advised to perform early tests to estimate runtime and training feasibility. As graduate students, team members can use facilities provided by the university, e.g., ECE Facility. Teams are expected to inform themselves about the limitations of the available computing resources and design accordingly.

References

[1] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning (ICML)*, pages 1861–1870. PMLR, 2018.

[2] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.