Name and email:
- Sophia Wang (sophiawang705@g.ucla.edu)

Prior experience:
- Won the ACM AI mini hackathon beginner track spring quarter 2025. But, beginner experience with RL, Deep Learning, and coding. I am most familiar with R and C++, with beginner knowledge in Python and SQL.

Hours spent and a breakdown of how they were spent:
- Hours 1-2: Understanding the prompt, researching articles, environment setup, and initial PPO baseline training.
- Hours 3-4: SDE implementation and hyperparameter running. Was able to reach 283 before dropping. Researched on the result, identifying a left leg limp from the video, which led me to further research into symmetric locomotion.
- Hours 5-6: Investigated even more into the collapse, dropping from 283 to 180s. Implemented recovery with action smoothing to reduce jitter.
- Hours 7-8: Stabilized agent to final performance of 271 training mean. Generated videos and completed a technical report and GitHub repository.
- Final performance: 271 training mean, .955 explained variance.

Compute resources: Google Colab, Intel Xeon @ 2.20GHz GPU
- A video of your walker's best performance (this should list the total number of time steps to train), this is included in the README

Techniques used:
The following articles provide the frameworks that serve as the foundation for my implementation.
- Because of its "Trust Region" clipping, which keeps the policy from collapsing during high-torque motions, Proximal Policy Optimization (PPO) was selected. It worked because the clipping mechanism prevented the agent from forgetting to stand after a bad update.
- State-Dependent Exploration (SDE): SDE offers temporally correlated noise in place of independent Gaussian noise. This is essential for bipedalism in order to prevent white noise from canceling out a "leg-lift" instruction over several timesteps. It provided a slight breakthrough to 283 because of the temporal consistency that it provided, needed for the legs to keep swining rather than just vibrating.
- Normalization of Observations: By using VecNormalize, gradient dominance was avoided by ensuring that the MLP weighted Lidar data (0–1.0) and hull angular velocity (scaled differently) were equal. This stabilization was able to achieve the final 0.955 variance in training logs.
- Action Smoothing ($\alpha=0.70$): In accordance with Mysore et al. (2021), actions were temporally filtered to avoid "joint chattering," which frequently results in the agent tripping at high speeds. By using this, the agent was able to have a stable center of gravity, but still have a left leg limp.

Ablation studies:

- The impact of VecNormalizing made the most difference. During initial tests, I removed the normalization of the environment's observations. Without it, the agent failed ot learn basic balance, due to the raw LIDAR data and hull angles leading to unstable gradients. By reintroducing normalization, the variance was able to reach /955. Proving that scaling the input data was needed to achieve the reward landscape.
- A huge challenge was the agent settling for a safe but inefficient way of walking by having the left leg as a limp. I was able to find a conservative rate that trapped this limping behavior. But in the end, the combination of a higher learning rate and extended training duration was able to achieve the score of 271, though the limp was still there.
- I also monitored the standard deviation of the agent's actions as a form of ablation on how it explores. Early on, the high variance was necessary for the agent to discover leg-swing momentum, but then the standard deviation dropped to 0.156 at the 1M step mark. Meaning it was able to remove its own noise for a more precise, mastered walking cycle.

Issues encountered:

- Again, the primary problem was the agent's tendency to fall into a stable limp. The walker developed a walk where it dragged its left leg to minimize the risk of falling. Though it secured a baseline positive reward by avoiding the -100 point penalty for a crash, it capped performance near 150 because of the jerky move, etc. I tried to break this cycle by going from Gaussian noise to SDE. This provided the baseline consistency for it to continue and be a little more symmetrical. Though it still kept the drag.
- Another problem was when I attempted to accelerate learning by increasing the learning rate, where the mean reward plummeted from 283 to 180s. The problem was that the trust region clipping in PPO was being overwhelmed by updates too large for this training. I implemented action smoothing at alpha = .70 to reduce the learning rate. This eliminated some of the joint shaking that caused the agent to trip. I think this was most helpful in getting the mean of 271.
- Early on, the agent was extremely unstable and failed almost immediately. I discovered this was because different sensors were talking at different volumes. The hull speed sensors gave large numbers, while LIDAR distance sensors gave smaller ones. The loud speed numbers overshadowed the quiet LIDASR numbers. I used a tool called VecNormalize, which acts like a volume balancer, and it scaled all the sensor data to match the intensity. Once I did this, the explained variance was able to go to around 95%. So the mismatched data wasn't an issue anymore, and the sensors were able to stay more balanced and walk.

Conclusion:

- The most successful part was transitioning from the standard bipedal gait to SDE and action smoothing. They were able to turn the agent's walking a bit smoother, although many improvements can still be made. If I had more time, I would definitely focus on Gait

Symmetry. Though the current walker is a bit faster than before, it still has the limp, which is common in bipedal reinforcement learning.

- I would probably research more on symmetry-based reward functions, which penalize the agent if the motor torques for the left leg don't mirror the right leg over a specific timeframe. Hopefully, that would push the score to over 300.
- Finally, I would want to see how the agent would handle different environments, like different terrains, gravity, or friction, if I had more time.
- This project definitely tested my capabilities by moving from a beginner's understanding to managing more complex topics that I wasn't familiar with. However, I really enjoyed the process and was able to learn a lot!

Citations for papers used:

- **Abdolhosseini, F., Ling, H. Y., Xie, Zhaoming, Peng, X. B., & van de Panne, M. (2019).** *On Learning Symmetric Locomotion.* Proceedings of the 12th ACM SIGGRAPH Conference on Motion, Interaction and Games. https://xbpeng.github.io/projects/SymLoco/index.html
- **Chou, P. W., Maturana, D., & Scherer, S. (2017).** *Improving Stochastic Policy Gradients in Continuous Control with Deep Reinforcement Learning using the Beta Distribution.* Proceedings of the 34th International Conference on Machine Learning (ICML). https://proceedings.mlr.press/v70/chou17a.html
- **Heess, N., Dhruva, T. B., Sriram, S., Lemmon, M., Merel, J., Wayne, G., Tassa, Y., Erez, T., Wang, Z., Eslami, S. M. A., Riedmiller, M., & Silver, D. (2017).** *Emergence of Locomotion Behaviours in Rich Environments.* arXiv preprint arXiv:1707.02286. https://arxiv.org/abs/1707.02286
- **Henderson, P., Islam, R., Bachman, P., Pineau, J., Precup, D., & Meger, D. (2018).** *Deep Reinforcement Learning That Matters.* Proceedings of the AAAI Conference on Artificial Intelligence, 32(1). https://doi.org/10.1609/aaai.v32i1.11694
- **Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., & Wierstra, D. (2015).** *Continuous Control with Deep Reinforcement Learning.* arXiv preprint arXiv:1509.02971. https://arxiv.org/abs/1509.02971
- **Mysore, S., Mabsout, B., Mancuso, R., & Saenko, K. (2021).** *Regularizing Action Policies for Smooth Control with Reinforcement Learning.* IEEE International Conference on Robotics and Automation (ICRA). http://ai.bu.edu/caps/
- **Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017).** *Proximal Policy Optimization Algorithms.* arXiv preprint arXiv:1707.06347. https://arxiv.org/abs/1707.06347
- **OpenAI.** (2018). *Spinning Up in Deep RL.* https://spinningup.openai.com. (Justifies the practical use of PPO and Observation Normalization).
- **Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017).** *Proximal Policy Optimization Algorithms.* https://arxiv.org/abs/1707.06347.