A Universal Framework for Ensembled Deep Federated Reinforcement Learning and Applications to Micro UAVs

Kevin Han[1], James Anderson[2], Brian Plancher[3]
[1]Department of Physics and Mathematics, University of Texas at Austin; kevinhan@utexas.edu
[2]Department of Electrical Engineering, Columbia University; [3]Department of Computer Science, Barnard College, Columbia University

Introduction

Federated Reinforcement Learning (FRL) enables multiple agents with identical state and action spaces in independent and varied environments to collaboratively learn an optimal policy. This approach is beneficial in scenarios where agent privacy is crucial, such as in energy grids or medicine. Using PyTorch, I develop a deep FRL framework capable of supporting and ensembling any RL algorithm, and enhance two a novel momentum-based algorithm, FEDSVRPG-M[1], alongside other state-of-the-art (SotA) RL algorithms, to train crazyflie drones. The novelty in momentum-based algorithms are shown with their guaranteed convergence to a stationary point of the average performance function, despite environment heterogeneity. However, this is under the assumption that they are the only local algorithms.

For the other local algorithms, I use Proximal Policy Optimization (PPO) [2], Soft Actor-Critic (SAC) [3], and Twin-Delayed Deep Deterministic Policy Gradient (TD3) [4]. Each of these algorithms have their independent strengths. PPO is more of an on-policy algorithm that is stable and provides reliable performance but can be sample inefficient and sensitive to hyperparameters. SAC and TD3 are off-policy algorithms. SAC offers excellent exploration and sample efficiency but is computationally demanding. TD3 is robust against overestimation bias and sample efficient but can be slower to converge. Combined with deep learning, my ensemble method aims to improve learning by leveraging the strengths of each other SotA sub-algorithm without losing too much of the convergence benefits of momentum-based FRL. I also explore the benefits of aggregation in value function estimation to determine if critics benefit from FRL too.

Furthermore, on the simulation side, I modify the gym-pybullet-drones[5] platform to include domain randomizations for wind and mass conditions, enhancing sim2real transfer. Using this FRL platform, I train crazyflie drones for various tasks and plan to incorporate layer freezing and LQR-based supervised learning for subtasks like hovering to advance robot learning.

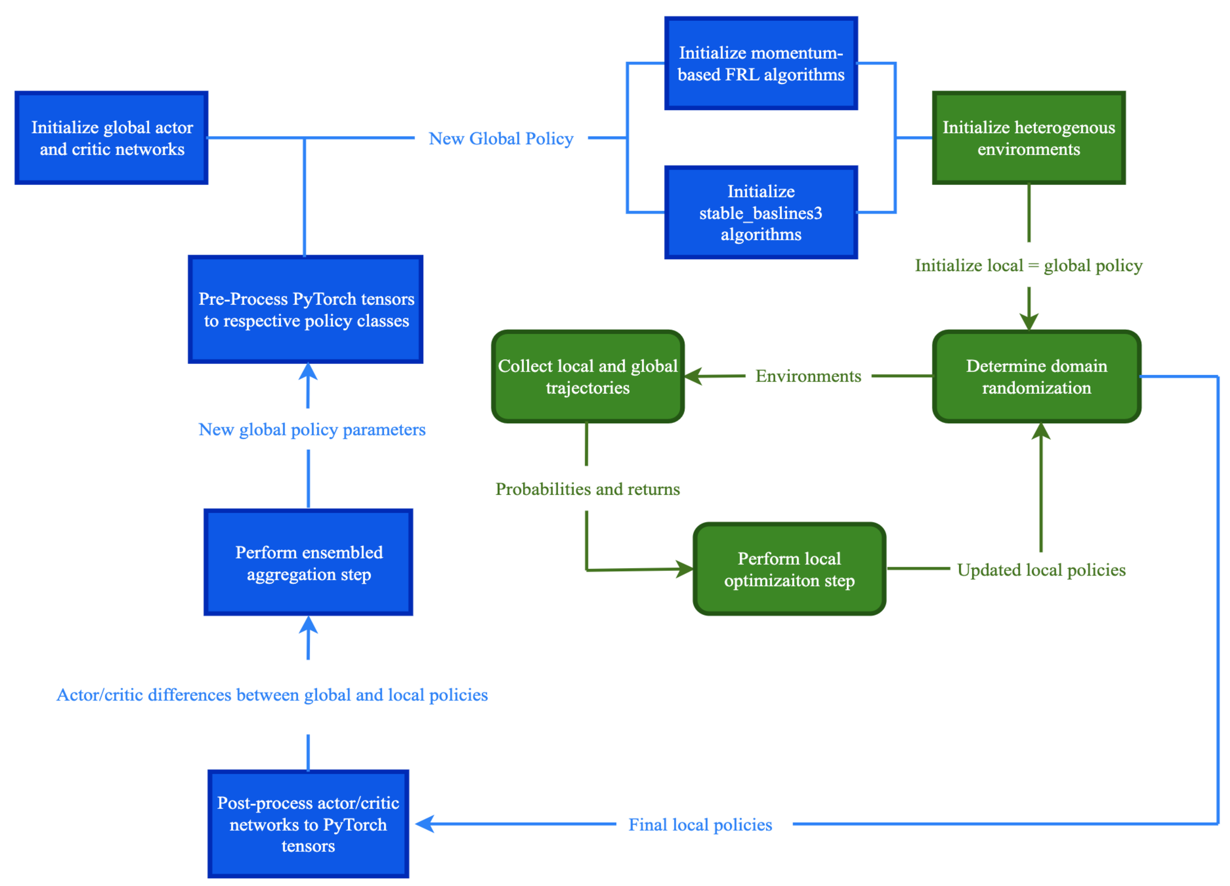
Research Goal

With this project I aim to answer 2 key questions and explore their consequential degrees of freedom for optimization:

1) Does implementing a federated reinforcement learning framework ensembling existing state-of-the-art algorithms with momentum-based algorithms provide any benefit over their respective vanilla versions?
2) How does environmental heterogeneity combined with domain randomization aid in more robust learning of chaotic dynamics in physical systems?

Software Architecture

General Workflow:



Federated RL Algorithm

LOCAL ITERATIONS *k* (for each agent *i*)

Initialize local policy $\theta_{r,k}$ to be identical to global policy θ_r

Policy Gradient Estimates:

Classic (REINFORCE)

$$g(\tau \mid \theta)=\sum_{t=0}^{H-1}\left(\sum_{h=t}^{H-1} \gamma^h \mathcal{R}\left(s_h, a_h\right)\right) \nabla \log \pi_\theta\left(a_t \mid s_t\right)$$

Momentum-Based (FEDSVRPG-M)

$$u_{r,k} = \beta g(\tau_{r,k}|\theta_{r,k}) + (1-\beta)[u_r + g(\tau_{r,k}|\theta_{r,k}) - w(\tau_{r,k}|\theta_{r-1},\theta_{r,k})g(\tau_{r,k}|\theta_{r-1})],$$

Objective Functions:

PPO

$$\underset{\theta}{\operatorname{maximize}} \quad \hat{\mathbb{E}}_t\left[\frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)}\hat{A}_t\right]$$
$$\text{subject to} \quad \hat{\mathbb{E}}_t[\mathrm{KL}[\pi_{\theta_{\text{old}}}(\cdot \mid s_t), \pi_\theta(\cdot \mid s_t)]] \le \delta$$

SAC

$$J_\pi(\phi) = \mathbb{E}_{\mathbf{s}_t\sim\mathcal{D},\epsilon_t\sim\mathcal{N}}\left[\log \pi_\phi(f_\phi(\epsilon_t;\mathbf{s}_t)|\mathbf{s}_t) - Q_\theta(\mathbf{s}_t, f_\phi(\epsilon_t;\mathbf{s}_t))\right]$$

TD3

$$J(\phi) = N^{-1}\sum \nabla_a Q_{\theta_1}(s,a)|_{a=\pi_\phi(s)} \nabla_\phi \pi_\phi(s)$$

Run local optimization step $k = 1,2, \ldots, K$

Send policy displacements: $\Delta_r = \theta_{r,K} - \theta_r$

GLOBAL SERVER STEP *r*

Evaluate all agents i and obtain mean rewards ρ_i

FRL Server Aggregation Step:

Standard

$$u_{r+1} = \frac{1}{\eta NK}\sum_{i=1}^{N}\Delta_r^{(i)}$$

Ensembled (weighted by reward)

$$u_{r+1} = \frac{1}{\eta NK\sum_{i=1}^{N}\rho_i}\sum_{i=1}^{N}\rho_i\Delta_r^{(i)}$$

Update global policy $\theta_{r+1} = \theta_r + \lambda u_{r+1}$

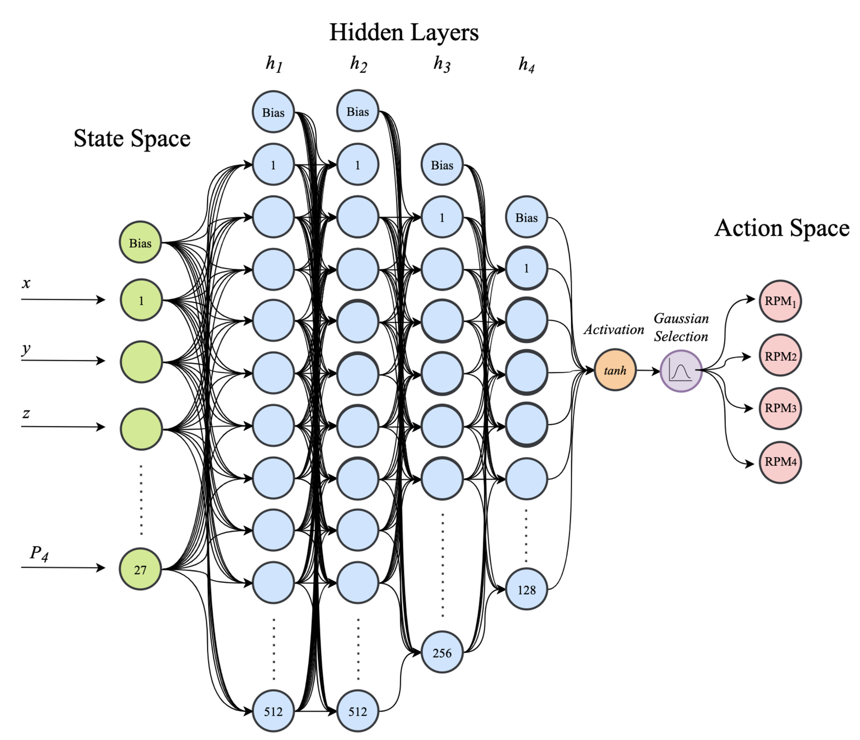
Repeat from top $r = r + 1, k = 0$ until $r = R$

Notes:

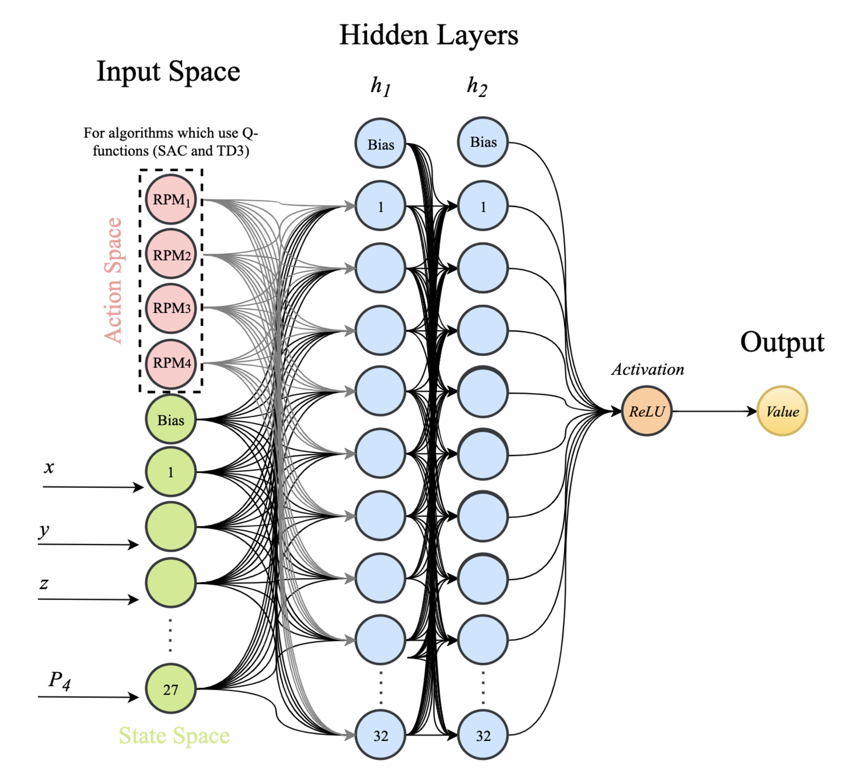
1. For algorithms which use target networks (SAC & TD3), I consider the non-target network during global updates and reinitialize them to be identical to the new global policy after
2. Policies which use state-action Q-functions have action parameters weighted separately outside of the gradient step and reinitialized as the average at the start of a new global iteration

Network Architectures

Actor Network: Diagonal Gaussian MLP



Critic Networks: Value Function Estimation



Simulation Design

State space:

$$S \subseteq \mathbb{R}^{27} \ni \{\mathbf{x}, \mathbf{q}, r, p, y, \dot{\mathbf{x}}, \boldsymbol{\omega}, \mathbf{P}\}$$

where:

- $\mathbf{x}$ represents the positions in x, y, z coordinates,
- $\mathbf{q}$ represents the quaternions,
- r represents the roll,
- p represents the pitch,
- y represents the yaw,
- $\dot{\mathbf{x}}$ represents the linear velocities in x, y, z directions,
- $\boldsymbol{\omega}$ represents the angular velocities,
- $\mathbf{P}$ represents the 4 RPMs for 4 motors.

Wind Disturbances:

$$F = \begin{cases} F_{ap}(t) & \text{if } p(e) < 0.5 \\ 0 & \text{if } p(e) \ge 0.5 \end{cases}$$

$$F_{ap}(t) = \begin{cases} \operatorname{sgn}(f(t)) \times |f(t)| & \text{if } p(s) < 0.3 \\ 0 & \text{if } p(s) \ge 0.3 \end{cases}$$

where:

- $F_{ap}(t)$ is the force induced by wind, bounded by a magnitude of 0.005 Newtons.
- $p(e)$ is the probability of domain randomization occurring for the episode.
- $p(s)$ is the probability of wind occurring in each step of a domain randomized episode.
- $f(t)$ is a vector whose magnitude is bounded by 0 and 0.005 Newtons.

Action space:

$$A \subseteq \mathbb{R}^4 \ni \mathbf{P}$$

where:

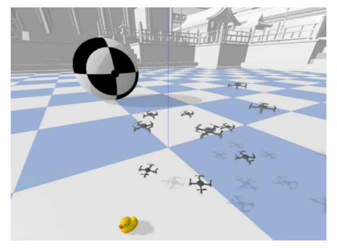
- $\mathbf{P}$ represents the 4 RPMs for 4 motors.

Reward Model:

For the task of hovering in the presence of wind, the reward is determined by the Euclidean distance from a target position. The episode terminates and reward is set to 0 if the drone flips, determined by its roll and pitch.

Mass Randomization:

$$\Pr(0.027\text{ g} \le m \le 0.042\text{ g} \mid p(e) < 0.5) = 1$$



A rendering of the gym-pybullet-drones simulator [5]

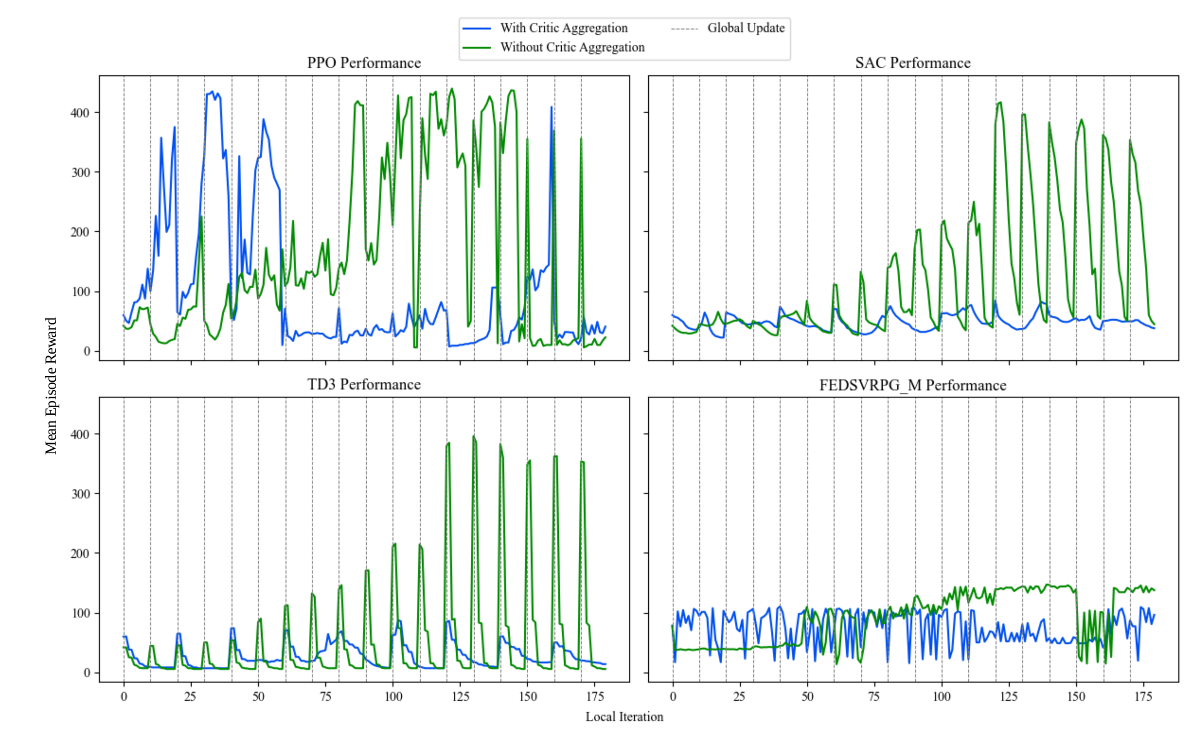
Results and Conclusion

Hyperparameters:

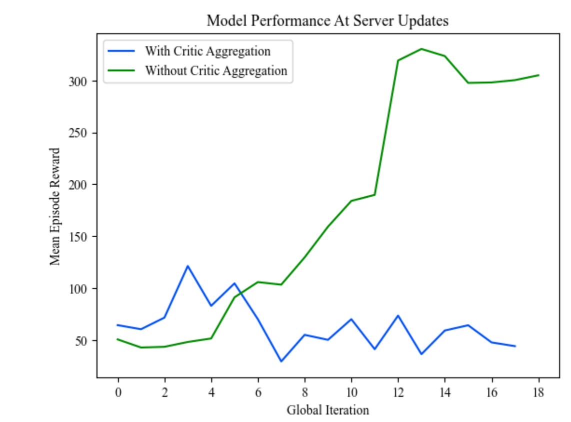
Global:
- Global Update Frequency: 10
- Total local iterations: 180
- Maximum Episode Length: 2048
- Global Learning Rate: 0.001
- Total Agents: 4

FedSVRPG-M:
- β = 0.2
- η=0.001

Stable_baslines3 Algorithms:
- Default



Results: The federated algorithm without critic network aggregation generally performed significantly better. It consistently outperformed the aggregated version for algorithms using double Q-Networks (SAC & TD3), though it showed a slower start for PPO. Additionally, SAC and TD3 exhibited a spike in performance with each global update, indicating potential benefits of ensembling. The momentum-based algorithm performed the worst overall but showed less pronounced effects from global updates compared to the other algorithms.



Conclusion: Due to time constraints, I had to limit total local iterations to 180. However, I believe it takes significantly more iterations in RL to obtain dependable results, especially when combining ensembling with deep learning. This would also explain why the critic aggregation performed significantly worse, and why every local iteration seemed to only worsen performance after a global update for some algorithms. Although only a basic hover task in the presence of wind was displayed through this project, the domain randomization and deep ensembled FRL platform provides for a lot of customizability, which shows potential for more complex tasks in different continuous environments. Furthermore, compatibility with physics simulators such as PyBullet and OpenAI Gymnasium environments [9] allows this platform to be widely open-sourced and serve as a baseline for federated or ensembled RL in further studies. Future work could include fine-tuning hyperparameters for each of the algorithms and for the actor/critic networks, employing more sophisticated statistical methods to match algorithm choice to heterogenous environments, and a sim2real transfer on real-life Crazyflie 2.1 drones in different environments.

References

[1] Han Wang, Sihong He, Zhili Zhang, Fei Miao, and James Anderson, "Momentum for the win: Collaborative federated reinforcement learning across heterogeneous environments," 2024.
[2] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov, "Proximal policy optimization algorithms," 2017.
[3] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," 2018.
[4] Scott Fujimoto, Herke van Hoof, and David Meger, "Addressing function approximation error in actor-critic methods," 2018.
[5] Jacopo Panerati, Hehui Zheng, SiQi Zhou, James Xu, Amanda Prorok, and Angela P. Schoellig, "Learning to fly – a gym environment with pybullet physics for reinforcement learning of multi-agent quadcopter control," 2021.
[6] Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann, "Stable-baselines3: Reliable reinforcement learning implementations," Journal of Machine Learning Research, vol. 22, no. 268, pp. 1–8, 2021.
[7] Junzi Zhang, Jongho Kim, Brendan O'Donoghue, and Stephen Boyd, "Sample efficient reinforcement learning with reinforce," 2020.
[8] Robinroy Peter, Lavanya Ratnabala, Demetros Aschu, Aleksey Fedoseev, and Dzmitry Tsetserukou, "Tornado-drone: Bio-inspired drl-based drone landing on 6d platform with wind force disturbances," 2024.
[9] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba, "Openai gym," 2016.

My Code

Acknowledgments

I would like to thank my primary supervisor, Dr. Brian Plancher, for his thorough guidance and inspiration throughout this project. I would also like to thank Dr. James Anderson for his guidance on Federated Reinforcement Learning. Finally, I would like to thank the program manager, Tiffany Moore, for a wonderful summer experience and allowing me to switch to Dr. Plancher's lab nearly halfway through the program for a better match.