COMSW4733 Computational Aspects of Robotics Milestone1

Hua Hsuan Liang (hl3811), Jaisel Singh (js6897) and Edward Zhang (cz2874)

I. Problem Overview and Objectives

Deformable object manipulation is a long-standing challenge in robotics, with applications ranging from cloth folding and rope tying to surgical assistance. Unlike rigid-body control, deformable objects exhibit high-dimensional, nonlinear, and partially observable dynamics that are difficult to capture with standard reinforcement learning (RL) policies such as Gaussian-distributed multilayer perceptrons (MLPs). Recent benchmarks like DaXBench highlight the gap between existing policy learning algorithms and the requirements of real-world deformable manipulation.

Our project addresses this gap by exploring flow-based policy representations within reinforcement learning. Specifically, we aim to integrate Conditional Flow Matching (CFM) with policy gradient optimization to leverage the expressivity of generative flows while maintaining the stability of on-policy RL.

We define the following objectives for this project:

- Implement and train FPO for deformable manipulation tasks, ensuring actions remain valid and learnable within continuous action spaces.
- Evaluate whether FPO improves sample efficiency and training stability compared to standard PPO (on-policy) and SAC (off-policy) with MLP policies.
- Compare Imitation learning(Flow matching model) and reinforcement learning(FPO) performance in the deformable manipulation tasks.

II. Technical Progress and Implementation

A. Training Algorithm

We adopt the FPO surrogate [6], which preserves the PPO clipped objective but replaces the likelihood ratio with a ratio computed from per-sample conditional flow-matching (CFM) losses. Let o_t denote the observation (state) at time t and $\hat{A}_t$ the advantage (e.g., GAE). The surrogate objective is

$$\max_{\theta} \; E\Big[\min\Big(\hat{r}_{\text{FPO}}(\theta)\,\hat{A}_t,\; \text{clip}(\hat{r}_{\text{FPO}}(\theta),\,1-\varepsilon_{\text{clip}},\,1+\varepsilon_{\text{clip}})\,\hat{A}_t\Big)\Big], \tag{1}$$

where the FPO ratio is

$$\hat{r}_{\text{FPO}}(\theta) = \exp\Big(\hat{L}_{\text{CFM},\,\theta_{\text{old}}}(a_t; o_t) \;-\; \hat{L}_{\text{CFM},\,\theta}(a_t; o_t)\Big). \tag{2}$$

The per-sample CFM loss is estimated with Monte Carlo:

$$\hat{L}_{\text{CFM},\,\theta}(a_t; o_t) = \frac{1}{N_{\text{mc}}} \sum_{i=1}^{N_{\text{mc}}} \ell_\theta(\tau_i, \epsilon_i), \tag{3}$$

with

$$\ell_\theta(\tau, \epsilon) \;=\; \tfrac{1}{2}\left\|\hat{v}_\theta(a_t^\tau, \tau; o_t) - (a_t - \epsilon)\right\|_2^2, \tag{4}$$

and the partially noised action

$$a_t^\tau = \alpha_\tau\, a_t + \sigma_\tau\, \epsilon, \qquad \tau \sim \mathcal{U}(0,1),\;\; \epsilon \sim \mathcal{N}(0, I). \tag{5}$$

We share the same (τ_i, ϵ_i) draws when evaluating $\hat{L}_{\text{CFM},\,\theta_{\text{old}}}$ and $\hat{L}_{\text{CFM},\,\theta}$ for variance reduction. Intuitively, decreasing the CFM/denoising loss for an action increases its ELBO and thus its effective likelihood under the flow policy; plugging the ratio in (2) into the clipped surrogate (1) yields a PPO-style, advantage-weighted update without computing exact log-likelihoods.

B. System figure

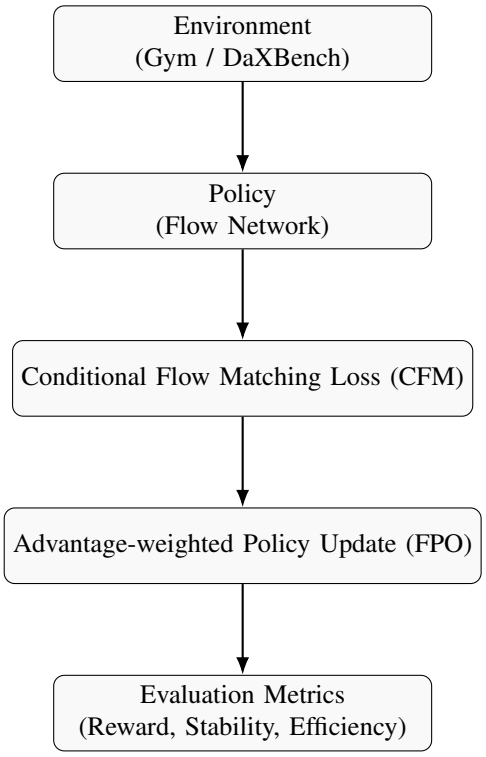

Fig. 1. System Architecture Overview for Flow Policy Optimization (FPO).

III. Preliminary Results and Analysis

In our experiments, we conducted four different training processes: **PPO**, **FPO**, and two variants of **Soft Actor-Critic (SAC)**. The first variant, **SAC-S**, updates the policy and critic

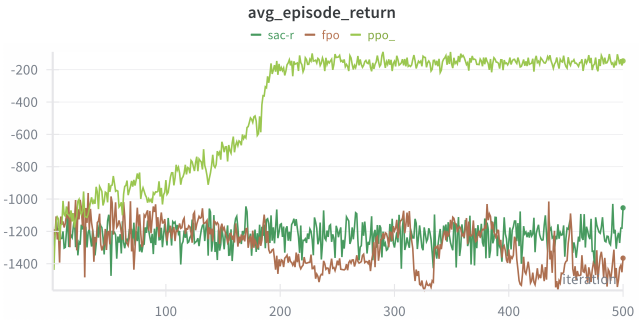

Fig. 2. The average return of each policy according to iteration on PENDULUM-V1, in this environment all reward is negative, as SAC-S not sharing the same training process, not showing in this plot, but it converged much early than PPO

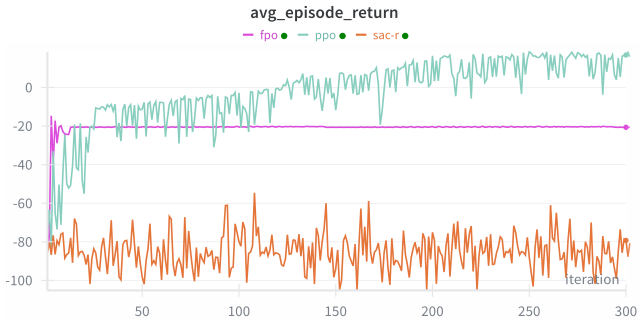

Fig. 3. The average return of each policy according to iteration on BipedalWalker, SAC-S not sharing the same training process, not showing in this plot, but it outperformed all the result we should in this plot.

after every environment step, while the second variant, **SAC-R**, performs updates only after each rollout. We implemented both SAC versions to align the update frequency with the on-policy methods (PPO and FPO), allowing for a more comparable evaluation across algorithms.

From the results, show in Table. 1 and Fig. 2,3,4, we observed that **SAC-R** performed poorly, which is expected since its replay buffer tends to reuse outdated samples, leading to delayed and unstable learning updates [3, 2], which also a possible that there are some bugs in our code. In contrast, **SAC-S** achieved the best performance, benefiting from the stable and well-tuned implementation provided by the **OpenAI Spinning Up** framework [1]. **PPO** also demonstrated competitive and consistent results, serving as a strong baseline for on-policy training [8].

However, **FPO** underperformed in our current setup. We suspect that this degradation stems from the flow policy's high sensitivity to action and state scaling. Improper normalization can easily destabilize the conditional flow matching objective, as reported in prior work on flow-based policy learning [6, 5]. To address this, we plan to incorporate both **action and state normalization** before feeding data into the policy network and conduct a deeper analysis to identify additional factors contributing to FPO's instability.

	SAC-S	SAC-R	PPO	FPO
Pendulum	-130	-1150	-120	-1400
Bipedal Walker	270	-70	14.89	-20
Reacher v4	-	-65	-56	-52

TABLE I
PERFORMANCE COMPARISON ON LAST 10 EPISODES AVERAGE RETURN.

IV. PLAN FORWARD

In the next phase of the project (Weeks 8–12), our primary objective is to **make the Flow Policy Optimization (FPO) framework functional and stable** before extending it to more complex environments. Although our preliminary experiments confirm that PPO and SAC-S perform reliably, the current

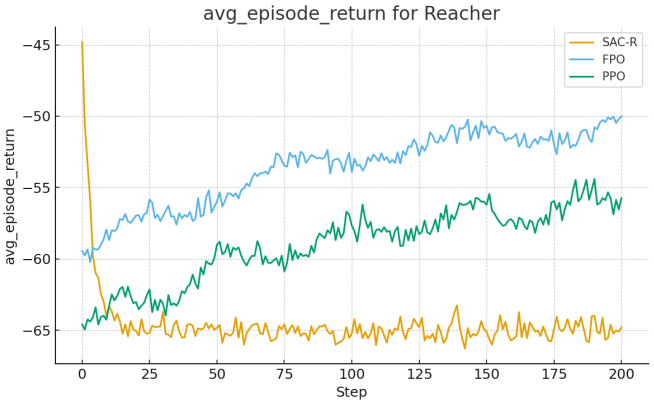

Fig. 4. Average return for Reacher-v4 benchmark comparing SAC, FPO, and PPO from Openai Gym, why SAC have high return value at the every beginning, it is because at the early stage we use the random action instead of use policy output.

implementation of FPO exhibits unstable learning dynamics and poor overall performance. Therefore, our immediate focus will be on diagnosing the root causes of these issues and improving FPO to achieve competitive results on simple rigid-body tasks.

To this end, we plan to conduct an in-depth analysis of the FPO training pipeline, including the action scaling, loss normalization, and noise scheduling components. We suspect that improper normalization and sensitivity to action magnitude are contributing to instability, as observed in prior work on flow-based policy optimization [5, 6]. Our next steps include: (1) implementing both **state and action normalization** layers, (2) verifying the correctness of the conditional flow matching loss computation, and (3) experimenting with smaller flow architectures to reduce overfitting and improve training stability.

Once a stable FPO implementation is established, we will extend our experiments to deformable object manipulation tasks in **DaXBench**, starting with the *RopeStraightening* and *ClothFolding* environments. At that stage, we will perform systematic hyperparameter tuning and evaluate **FPO** alongside **PPO**, **SAC-S**, and **Flow Matching Policy** baselines.

Here, the Flow Matching Policy will serve as an *imitation*

learning representation model. Since we do not have access to expert demonstrations, we will collect trajectory data from the trained FPO agent to serve as pseudo-expert samples. This approach is inspired by iterative imitation learning frameworks such as DAgger [7] and GAIL [4], where data generated by an existing policy can be reused to improve or evaluate subsequent models. Although our setup is unconventional, it allows us to examine whether the imitation learning model can successfully replicate or refine the behaviors discovered by FPO. If the imitation learning policy fails to achieve comparable performance under this setup, it would further indicate that the FPO framework possesses unique representational potential worth improving and extending.

The evaluation will include quantitative metrics such as average episode reward, convergence speed, and robustness to random initialization, along with qualitative assessments of policy behavior across different deformable manipulation tasks.

Finally, our goal for the next milestone is to deliver (1) a working and stable version of FPO validated on rigid tasks, (2) preliminary progress on DaXBench integration, and (3) a comprehensive comparison with baseline algorithms. This will ensure that subsequent work can focus on scaling FPO to complex deformable environments with a strong and reliable foundation.

REFERENCES

[1] Joshua Achiam. Spinning up in deep reinforcement learning. https://spinningup.openai.com, 2018.

[2] Scott Fujimoto, Herke van Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning (ICML)*, pages 1582–1591, 2018.

[3] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *arXiv preprint arXiv:1801.01290*, 2018.

[4] Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 29, pages 4565–4573, 2016.

[5] Thibault Lipman, Jiaming Song, and Stefano Ermon. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

[6] David McAllister, Songwei Ge, Brent Yi, Chung Min Kim, Ethan Weber, Hongsuk Choi, Haiwen Feng, and Angjoo Kanazawa. Flow matching policy gradients. In *arXiv preprint arXiv:2507.21053*, 2025.

[7] Stéphane Ross, Geoffrey J. Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 627–635, 2011.

[8] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. In *arXiv preprint arXiv:1707.06347*, 2017.