Final Project Phase I Report

Predictive Machine Learning - CS395T Spring 2025

Name: Hasif Shaikh
Date: April 21, 2025

1 Approach to the Problem

1.1 Problem Formulation

The Safe-PDP framework [Jin et al., 2021] is a framework for finding the optimal trajectory while obeying constraints. In the original Safe-PDP paper, the known system dynamics $f(x_t, u_t)$ and cost function $J(\theta)$ are fixed, and the setup is deterministic. In this setting, given an initial state and a sequence of actions, the trajectory is completely predictable.

The model prior is incapable of representing some aspects of real-world dynamics due to its purely deterministic nature. Oftentimes, there is noise in the form of sensor noise, environmental factors, or other factors that can lead to variability. As a result, the deterministic Safe-PDP framework sometimes struggles in modeling real-world control problems that have a significant amount of noise.

The goal is to introduce a stochastic optimal control framework that uses a system that evolves stochastically instead of using deterministic transitions. This system can be modeled $x_{t+1} \sim f(x_t, u_t) + \epsilon_t$ with the addition of ϵ_t representing the Gaussian noise for every timestep. As well, due to it being stochastic, the cost function becomes an expectation of all realizations of a trajectory under $\pi(u_t, x_t)$ by $J(\theta) = E_{u_t \sim \pi(u_t | x_t)}[\sum_{t=0}^{T-1} c_t(x_t, u_t) + c_t(x_t)]$. Safety constraints are incorporated in the stochastic setting by making sure the probability of constraint violations is very low across various sample trajectories.

1.2 Learning Method

To solve this problem, I propose using Differential Policy Optimization (DPO) [Bajaj and Nguyen, 2024]. DPO can handle stochastic environments using only sample-based access to the system. Thus, DPO is more suitable for real-world control modeling where transitions can be impacted by noise.

DPO works by iteratively collecting on-policy sampling and using those samples to calculate gradients meant to improve policy parameters. It can ensure an increase in policy quality in each iteration. Eventually, the policy learns to generate action sequences that minimize expected total cost while also adapting to the stochastic environment.

DPO relies only on collected data from interacting with its environment. This is different from model-based methods like Safe-PDP and Neural PMP [Bajaj et al., 2024] that require an explicit model of dynamics. As a result, DPO can be more robust to model misspecification compared to the model-based approaches. With DPO, the goal is to train a policy that optimizes behavior from noisy rollouts without having to custom-make a model of the stochastic system. Additionally, by penalizing trajectories that violate safety, there is a built-in way to incorporate safety constraints into the DPO training loop.

1.3 State/Action/Policy/Reward Setup

I plan on designing a simplified environment to model a stochastic control system, meant to model noisy settings. The state x_t is represented as a 2-dimensional vector. Having a 2-dimensional vector simplification makes it easy to visualize and interpret the policy behavior. The action u_t is a continuous-valued 2-dimensional vector. This allows the action to represent control forces that can alter the trajectory of the system. Actions are chosen each timestep by policy and affect future states through stochastic dynamics. The policy $\pi(u_t | x_t)$ is a stochastic mapping from states to distributions over actions. Currently, the policy is just sampling actions conditioned on the current state, but in the future, I aim to implement a fully learnable policy network that can map states to actions even with the presence of noise. To keep it simple, the reward function is a negative squared Euclidean distance between the agent's state and the desired target state $c_t(x_t, u_t) = ||x_t - x_{target}||_2^2$.

1.4 Flowchart

The figure below summarizes the learning dynamics employed in my stochastic Safe-PDP extension using DPO. The current state uses a stochastic policy to sample actions that transition the state through noisy dynamics. Afterwards, I calculate the cost and update the policy using DPO

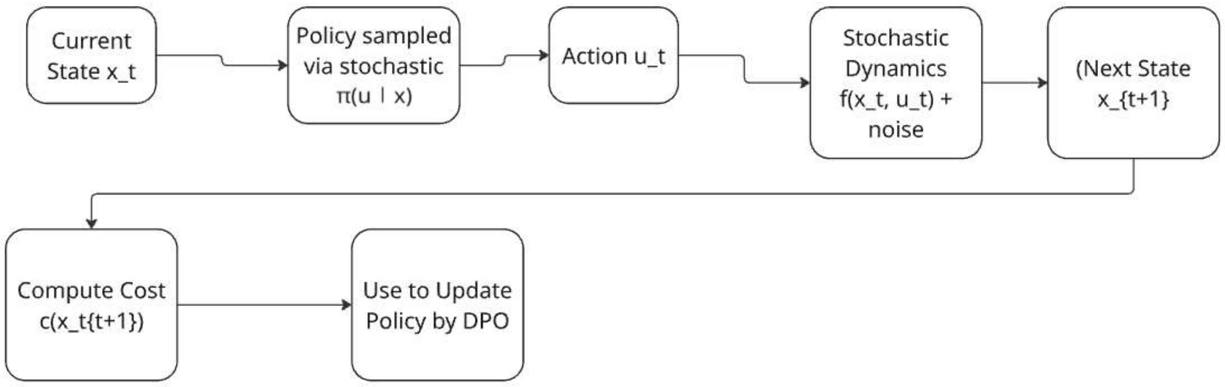
Current State x_t
Policy sampled via stochastic π(u | x)
Action u_t
Stochastic Dynamics f(x_t, u_t) + noise
(Next State x_{t+1}
Compute Cost c(x_t{t+1})
Use to Update Policy by DPO

Figure 1: Learning Dynamics Overview for Stochastic Safe-PDP via DPO.

1.5 Data and Experiment Plan

To test the approach, the goal will be to simulate trajectories of a 2D mass particle in a 2D stochastic point-mass environment. The state will keep track of the position and the velocity in two dimensions, and in each timestep, the control action will perturb the velocity. The dynamics of the system will add Gaussian noise, which would emulate real-world disturbances mentioned previously.

The training data is generated dynamically by episodes rolling out, where the agent tries to navigate towards a fixed location. Each of these rollouts produces sequences of data (state, action, cost). $J(\theta)$ will represent the total deviation from the target across an episode (the expected cumulative cost). Additionally, I will monitor safety metrics with constraint violations. For the experiment to succeed, I must see a consistent reduction in expected cost over training epochs, and ensure that the system is robust by testing the policy over new random noise to ensure the system did not learn a specific noise.

In future work for Phase II, I plan on extending the implementation to OpenAI Gym environments such as MountainCar to test the system in a more complex environment.

2 Review of Prior Methods

2.1 Prior Methods Table

Method	Model-based or model-free	Limitation	Chosen status
Safe-PDP [Jin et al., 2021]	Mode-Based	Safe-PDP is effective in deterministic systems that require strict constraint satisfaction. Safe-PDP struggles in a non-ideal environment and deterministic dynamics, which is extremely rare in a real-world setting with noise	This is the starting point
Neural PMP [Bajaj et al., 2024]	Model-Based	Neural PMP extends Hamiltonian-based methods to a stochastic setting via neural networks. The downsides of Neural PMP, however, are in the computational complexity required due to the incorporation.	Eliminated
DPO [Bajaj and Nguyen, 2024]	Model-Free	DPO can optimize a policy using sample-based rollouts without requiring a dynamics model. It has the downside of being sample inefficient and slower to converge compared to the model-based approaches. It does excel however, in being more scalable and robust.	Selected

2.2 Discussion

I explored using Safe-PDP, Neural PMP, and DPO for solving constrained stochastic optimal control problems. The Safe-PDP framework [Jin et al., 2021] is a good foundation for constrained optimal control. It excels in purely deterministic environments while guaranteeing hard constraint satisfaction by optimizing Pontryagin Hamiltonian conditions. However, Safe-PDP requires the system to be known and deterministic, making it difficult to use in real-world applications. In noisy environments, its extension is fairly non-trivial.

Neural PMP [Bajaj et al., 2024] improves robustness by introducing neural networks to approximate co-state variables in stochastic settings. Neural PMPs allow for handling stochasticity in the theory. However, it can be very computationally inefficient due to the requirement of solving complex backward dynamic programming equations as well as the forward trajectories. Additionally, it can struggle in high-noise environments due to inaccurate co-state estimation.

Differentiable Policy Optimization (DPO) [Bajaj and Nguyen, 2024] is a model-free approach by directly optimizes policies through sample-based rollouts. It guarantees monotonically increasing policy improvement by using observed costs from sample trajectories to update the policy parameters. DPO has downsides in requiring a large number of samples to achieve adequate performance, which would slow down the training time. However, the scalability and robustness of DPO make it a compelling option as the choice of algorithm for this project.

3 Proposed Extension and Experimental Plan

I propose to expand upon Safe-PDP by developing a training pipeline based on DPO. In Phase II, I will implement a parameterized stochastic policy that outputs actions distributed on the current state. Training will be done by collecting the rollouts in the current policy, estimating costs, and updating policy parameters using policy gradient updates.

To test robustness, I plan on introducing varying noise intensities in training to evaluate how much the policy performance degrades in the presence of noise. Additionally, the safety constraint will be measured by tracking violation frequencies across stochastic trajectories. In the future, I plan to use a more complex benchmark environment using OpenAI Gym to test the scalability of the model.

Finally, I plan to benchmark the effectiveness of the DPO-based approach against a Safe-PDP baseline under stochastic perturbations.