Plan Your Target And Learn Your Skills: Transferable State-Only Imitation Learning via Decoupled Policy Optimization

ICML 2022 Publication

Minghuan Liu, Zhengbang Zhu, Yuzheng Zhuang, Weinan Zhang, Jianye Hao, Yong Yu, Jun Wang

Shanghai Jiao Tong University　　Huawei Noah's Ark Lab

Motivation

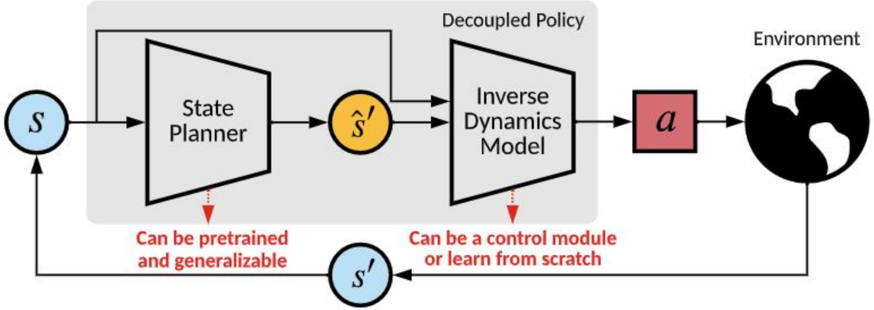
Can we imitate the high-level planning mode and transfer to different action spaces?

- How to model and generalize the high-level planning mode?
- How to transfer such an ability?
- A state-to-action policy is ad-hoc to the action space and dynamics

Observation:

- A policy can be seen as a composition of planner and inverse dynamics
- The planner can be universal and the inverse dynamics can be ad-hoc to different action spaces

Decouple the policy



Given a global inverse dynamics, we can learn from state-only demostrations by matching state planner (need an accurate inverse dynamics so it can obtain a correct action to interact)!

Formulation:

$$\pi = \underbrace{\mathcal{T}_\pi^{-1}}_{\text{inverse dynamics}} \left(\underbrace{\mathcal{T}(\pi_E)}_{\text{state planner}} \right)$$

$$\begin{aligned}\pi_E(a|s) &= \int_{s'} \mathcal{T}(s'|s,a)\pi_E(a|s)\,\mathrm{d}s' \\ &= \int_{s'} \frac{\rho_{\pi_E}(s,s') I_{\pi_E}(a|s,s')}{\rho_{\pi_E}(s)}\,\mathrm{d}s' \\ &= \int_{s'} h_{\pi_E}(s'|s) I_{\pi_E}(a|s,s')\,\mathrm{d}s'\,.\end{aligned}$$

inverse dynamics

state planner

Definition 4.2. A hyper-policy $\Omega \in \Lambda$ is a maximal set of policies sharing the same state transition occupancy such that for any $\pi_1, \pi_2 \in \Omega$, we have $\rho_{\pi_1}(s,s') = \rho_{\pi_2}(s,s')$.

$$h_\Omega(s'|s) = \frac{\rho_\Omega(s,s')}{\int_{\tilde{s}} \rho_\Omega(s,\tilde{s})\,\mathrm{d}\tilde{s}} = \int_a \pi(a|s)\mathcal{T}(s'|s,a)\,\mathrm{d}a\,.$$

state planner

$$\arg\min_\Omega \ell\left(\rho_{\Omega_E}(s,s'), \rho_\Omega(s,s')\right)$$

$$\Rightarrow \arg\min_{h_\Omega} \mathbb{E}_{s\sim\Omega}[\ell\left(h_{\Omega_E}(s'|s), h_\Omega(s'|s)\right)]\,.$$

Learn by supervised learning

1. Learn state planner from state-only demonstrations!
   - Can cause compounding error problem!

$$\min_{\psi} L^h = \mathbb{E}_{(s,s')\sim\Omega_E}[\mathrm{D_f}(h_{\Omega_E}(s'|s)\|h_\psi(s'|s))] \, .$$

2. Learn (global) inverse dynamics from sampled data!
   - This is the only way to learn this module!
   - State planner is only meaningful when given an accurate inverse dynamics model to sample in the environment and reach a desired state.

$$\min_{\psi} L^I = \mathbb{E}_{(s,s')\sim\pi_{\mathcal{B}}}[\mathrm{D_f}(I_{\pi_{\mathcal{B}}}(a|s,s')\|I_\phi(a|s,s'))] \, .$$

Decoupled policy gradient

- We can also do policy gradient to learn the state planner! This will alleviate compounding error when doing imitation learning.

$$\begin{aligned}\nabla_{\phi,\psi}\mathcal{L}^{\pi} &= \mathbb{E}_{(s,a)\sim\pi}\left[Q(s,a)\nabla_{\phi,\psi}\log\pi_{\phi,\psi}(a|s)\right] \\ &= \mathbb{E}_{(s,a)\sim\pi}\Big[\frac{Q(s,a)}{\pi(a|s)}\Big(\int_{s'} I(a|s,s')\nabla_{\psi}h_{\psi}(s'|s)\,\mathrm{d}s' \\ &\quad + \mathbb{E}_{s'\sim h}\left[\nabla_{\phi}I_{\phi}(a|s,s')\right]\Big)\Big] .\end{aligned}$$

- We should let the inverse dynamics to be an accurate control module at least **for the current policy** when learning the planner, the inverse dynamics function is static when optimizing the policy function.

Decoupled policy gradient

- If given an accurate (at least local) inverse dynamics model

$$s' = h(\epsilon; s), \;\; \pi(a|s) = \mathbb{E}_{\epsilon \sim \mathcal{N}} \left[I(a|s, h(\epsilon; s)) \right]$$

$$\nabla_\psi \mathcal{L}^\pi = \mathbb{E}_{(s,a) \sim \pi, \epsilon \sim \mathcal{N}} \left[\frac{Q(s,a)}{\pi(a|s)} (\nabla_h I(a|s, h_\psi(\epsilon; s)) \nabla_\psi h_\psi(\epsilon; s)) \right]$$

- **Explain**: we are taking the knowledge from the inverse dynamics about action to update parameters of the state planner by updating the prediction about the next state with error $\Delta s' = \alpha \nabla_h I(a|s, h(\epsilon; s))$
- Reward can be obtained through GAN-like methods for imitation learning, such as GAIfO - using discriminator $D_\omega(s, s')$ to obtain the reward $r(s,a) \triangleq r(s,s')$

Problem: agnostic decoupled PG

Simply applying DePG to obtain the desired high-level target planner is faced with serious learning challenges, especially when the inverse dynamics model is approximated by NNs.

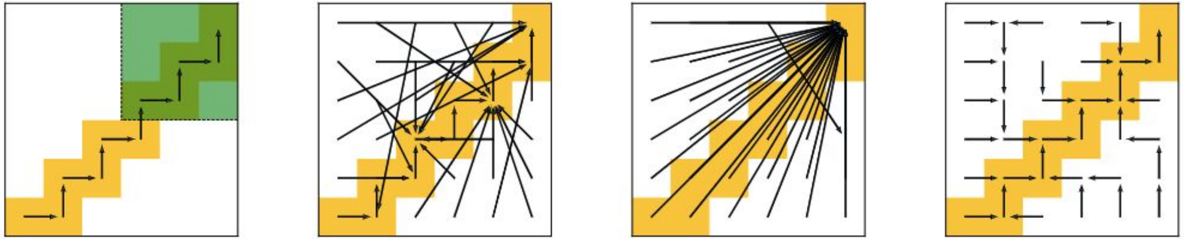
Why?

- A **constraint** on the inverse dynamics model's input - the prediction of the target planner **MUST** be a legal neighbor state.
- However simply apply decoupled PG does not ensure that.
- An **illegal** state transition could still be a **legal** input to the approximated inverse dynamics model and it may still provide a **feasible** action to interact with the environment.

Problem: agnostic Decoupled PG (DePG)

Simply applying DePG to obtain the desired high-level target planner is faced with serious learning challenges, especially when the inverse dynamics model is approximated by NNs.

An illustrative case:



(a) Grid World Setting (b) Supervised DePO (c) Agnostic DePG (d) DePO

Figure 2: Grid world environment and the prediction of the learned state planner. (a) The expert starts from the left-bottom corner (0,0) to the right-upper (6,6) and the arrows on the yellow grid depicts the path of the expert. The agent is required to start at any grid on the map except the shaded zone. We test three variants under the decoupled policy structure where both modules are learned from scratch. (b) Supervised learning only from the dataset results in predictions of the target state on expert paths, even if not a neighboring (legal) one. (c) Agnostic DePG learns to predict arbitrary states, while the inverse dynamics can still give a legal action to reach a neighbor state. (d) The proposed DePO algorithm, which generalizes the planning into every out-of-demonstration state (white blocks) with legal transitions.

Solution: Constraint by Calibrated Decoupled PG (CDePG)

$$\nabla_\psi \mathcal{L}^\pi = \mathbb{E}_{(s,a,s')\sim\pi}\left[Q(s,a)\nabla_\psi \log h_\psi(s'|s)\right]$$

Explain: Optimizing CDePG can be realized as maximizing the probability to target state *s′* on state *s* if *a* is a good action regarding the inverse dynamics is accurate.

Trade-off:

- CDePG has a severe exploration problem since the planner is only allowed to predict a visited state.
- DePG provides a way to explore the most promising actions although it is not responsible for getting legal state transition.

Practice: Optimize DePG and CDePG jointly.

$$\min_\psi \mathcal{L}^{\pi,h} = \mathcal{L}^\pi_{\mathrm{DePG}} + \lambda_h(\mathcal{L}^h + \mathcal{L}^\pi_{\mathrm{CDePG}})$$

Algorithm

Algorithm 1 Decoupled Policy Optimization (DePO)

```
Input: State-only expert demonstration data 𝒟 = {(s_i)}_{i=1}^N, empty replay buffer ℬ, randomly initialized discriminator model D_ω, state transition predictor h_ψ and parameterized inverse dynamics model I_φ
for k = 0, 1, 2, ··· do
    ▷ Pre-training stage
    Collect trajectories {(s, a, s′, r, done)} using a random initialized policy π = E_{ε∼𝒩}[I_φ(a|s, h_ψ(ε; s))] and store in ℬ
    Sample (s, a, s′) ∼ ℬ and update φ by L^I (Eq. (9))
end for
for k = 0, 1, 2, ··· do
    ▷ Online training stage
    Collect trajectories {(s, a, s′, r, done)} using current policy π = E_{ε∼𝒩}[I_φ(a|s, h_ψ(ε; s))] and store in ℬ
    Sample (s, a, s′) ∼ ℬ, (s, s′) ∼ 𝒟
    if Learn inverse dynamics function then
        repeat
            Update φ by L^I (Eq. (9))
        until Converged
    end if
    Update the discriminator D_ω with the loss:
```

$$\mathcal{L}_\omega^D = -\mathbb{E}_{(s,s')\sim\mathcal{B}}[\log D_\omega(s,s')] - \mathbb{E}_{(s,s')\sim\mathcal{D}}[\log(1 - D_\omega(s,s'))] \,,$$

```
    Update ψ by ℒ_ψ^{π,h} (Eq. (17))
end for
```

First update inverse dynamics model!

Then update D to get reward! (If imitation)

At last update state planner

Experiments

Imitation and transferring to different action dynamics / spaces:

Keep the planner and learn the inverse dynamics by supervised learning only.

Discrete space (grid world):

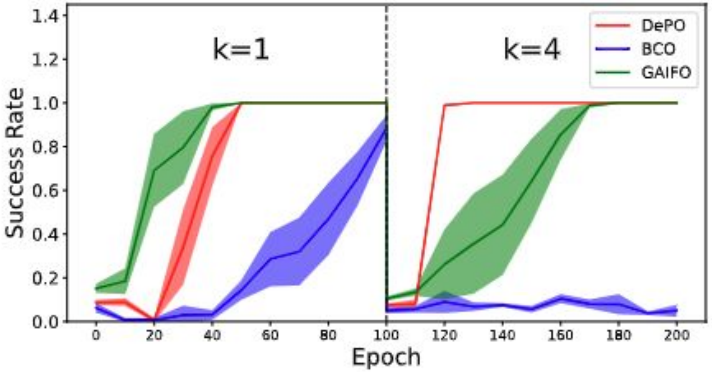


Figure 3: Transferring experiment on grid world environment. The y-axis denotes the success rate of reaching (5,5). The solid line and the shade shown in this and following figures represent the mean and the standard deviation of the results over 5 random seeds.

Experiments

Imitation and transferring to different action dynamics / spaces:

Continous space (mujoco): given the orginal action space as *m* and dynamics as *s'=f(s,a)*

Transfer action dynamics: 0.8 gravity, *n=2m* and *s'=f(s,h(a))*

$$h = -\exp(a[0:n/2]+1)+\exp(a[n/2:-1]))/1.5$$

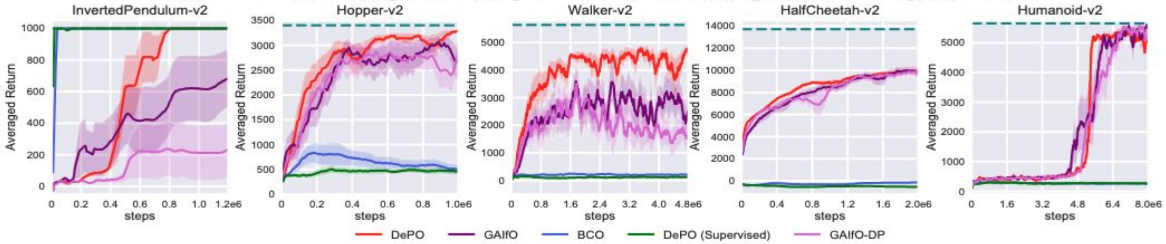


Figure 4: Learning curves on easy-to-hard continuous control benchmarks, where the dash lines represent the expert performance.

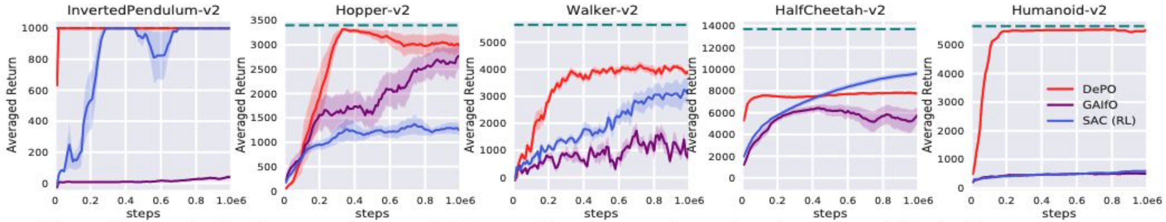
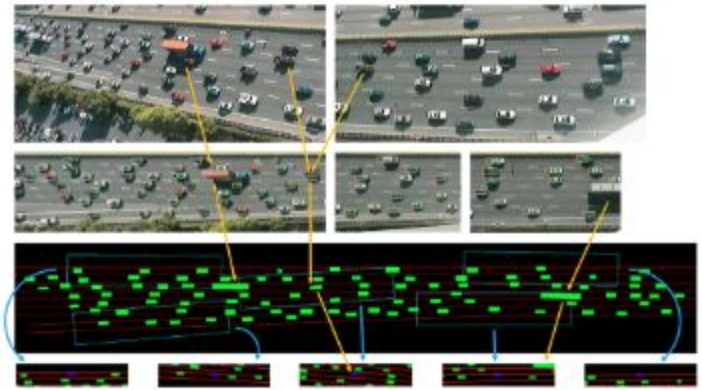


Figure 5: Transferring by pre-training on Mujoco tasks with complex action dynamics within 1e6 interaction steps.

Experiments

Co-training for different action dynamics / spaces (NGSIM dataset)



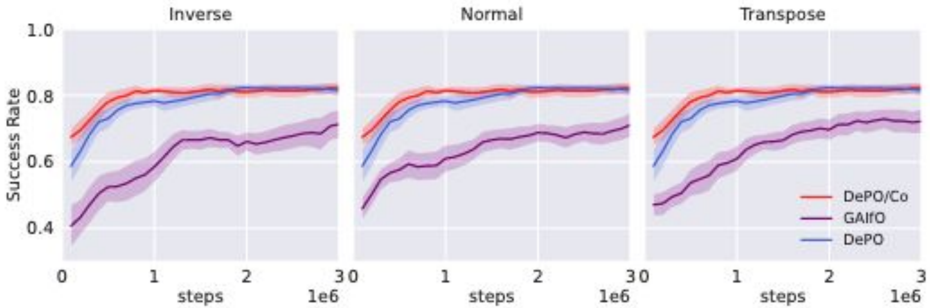


Figure 6: Co-training for different vehicles.

Experiments

Our method is general, can also be used for RL and transfer!

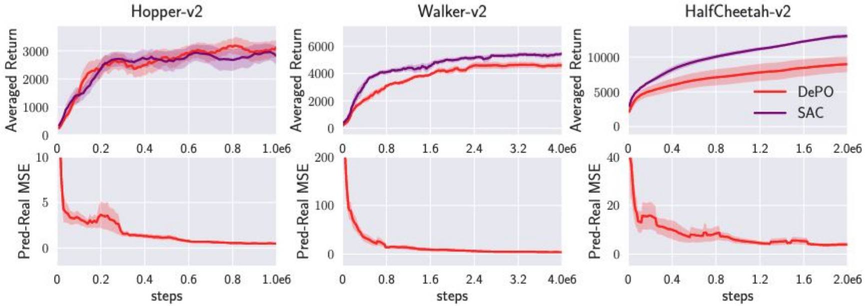


Figure 16: RL experiments on Mujoco tasks over 5 random seeds.

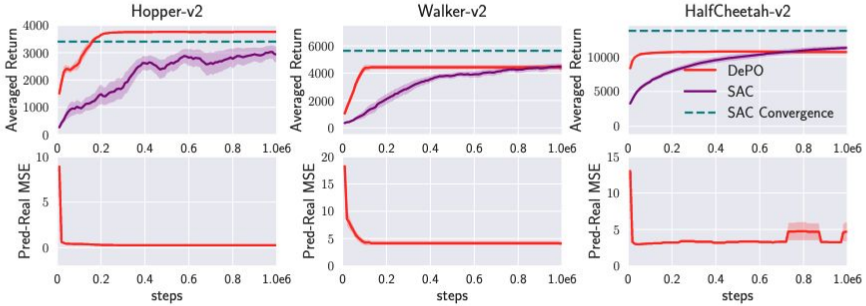
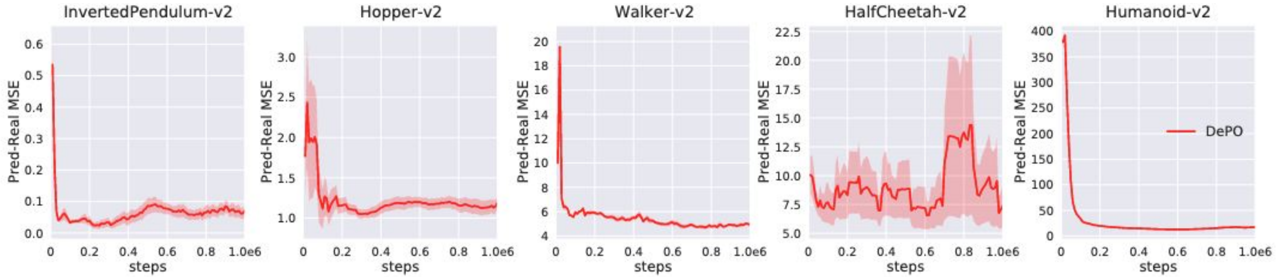


Figure 17: One-shot Transferring by pre-training RL agents on Mujoco tasks with inverted action dynamics over 5 random seeds.

Experiments

State planner is accurate for multi-step planning!



Pred-Real MSE

Figure 10: MSE curves of the one-step prediction of the state planner and the real state that the agent achieves in the environment in **complex** transfer experiments. The target state predictions are stably accurate along the whole training stage.

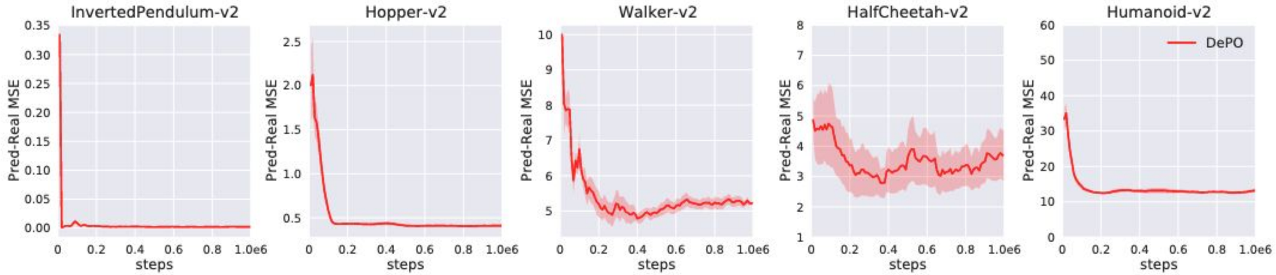
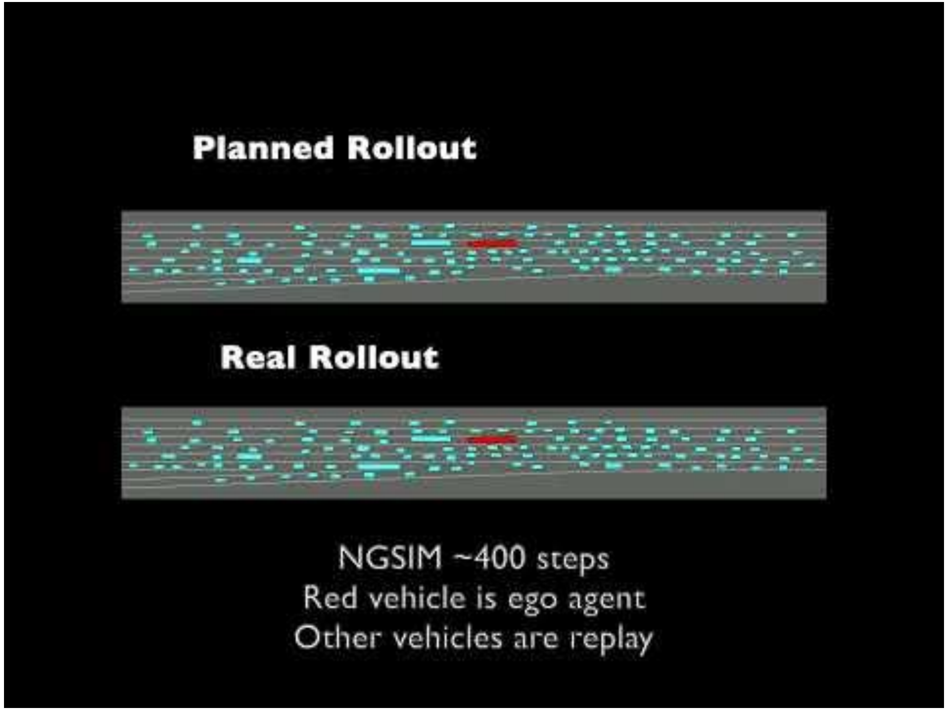


Figure 11: MSE curves of the one-step prediction of the state planner and the real state that the agent achieves in the environment in **simple** transfer experiments. The target state predictions are stably accurate along the whole training stage.

Experiments

State planner is accurate for multi-step planning!



https://www.youtube.com/watch?v=WahVjjvcYYM