A Missing Proofs

A.1 Proof of Proposition 1

Definition 2. *A metric space is an ordered pair (M, d) where M is a set and d is a metric on M, i.e., a function $d\colon M \times M \to \mathbb{R}$ such that for any $x, y, z \in M$, the following holds:*

1. $d(x, y) \geq 0, d(x, y) = 0 \Leftrightarrow x = y$,
2. $d(x, y) = d(y, x)$,
3. $d(x, z) \leq d(x, y) + d(y, z)$.

The first two properties are obviously guaranteed by $\overline{D}_W^{\rho}$. As for the triangle inequality,

$$
\begin{aligned}
&\mathbb{E}_{s\sim\rho(s)}[D_W(\theta_i(s), \theta_k(s)] \\
&= \mathbb{E}_{s\sim\rho(s)}[\sum_{l=1}^{|\mathcal{A}|} |\theta_i(s) - \theta_k(s)|] \\
&= \mathbb{E}_{s\sim\rho(s)}[\sum_{l=1}^{|\mathcal{A}|} |\theta_i(s) - \theta_j(s) + \theta_j(s) - \theta_k(s)|] \\
&\leq \mathbb{E}_{s\sim\rho(s)}[\sum_{l=1}^{(|\mathcal{A}|} |\theta_i(s) - \theta_j(s)| + |\theta_j(s) - \theta_k(s)|)] \\
&= \mathbb{E}_{s\sim\rho(s)}[\sum_{l=1}^{|\mathcal{A}|} |\theta_i(s) - \theta_j(s)|] + \mathbb{E}_{s\sim\rho(s)}[\sum_{l=1}^{|\mathcal{A}|} |\theta_j(s) - \theta_k(s)|] \\
&= \mathbb{E}_{s\sim\rho(s)}[D_W(\theta_i(s), \theta_j(s)] + \mathbb{E}_{s\sim\rho(s)}[D_W(\theta_j(s), \theta_k(s)]
\end{aligned}
$$

B Proof of Proposition 2

$$
\begin{aligned}
\rho_\theta(s) &= P(s_0 = s|\theta) + P(s_1 = s|\theta) + ... + P(s_T = s|\theta) \\
&\overset{L.L.N.}{=} \lim_{N\to\infty} \frac{\sum_{i=1}^N I(s_0 = s|\tau_i)}{N} + \frac{\sum_{i=1}^N I(s_1 = s|\tau_i)}{N} + ... + \frac{\sum_{i=1}^N I(s_T = s|\tau_i)}{N} \\
&= \lim_{N\to\infty} \frac{\sum_{j=0}^T \sum_{i=1}^N I(s_j = s|\tau_i)}{N} \\
\overline{\rho}_\theta(s) &= \sum_{i=1}^N \sum_{j=0}^T \frac{I(s_j = s|\tau_i)}{N} \\
\mathbb{E}[\overline{\rho}_\theta(s) - \rho_\theta(s)] &= 0
\end{aligned}
$$

C Implementation Details

C.1 More Details on WSR

WSR: Penalty Method The Penalty Method considers the constraints of Eq.(6) by putting constraint $g(\theta)$ into a penalty term, followed by solving the following unconstrained problem in an iterative manner,

$$
\max_{\theta\in\Theta} \quad f(\theta) + \frac{1-\alpha}{\alpha} \min\{g(\theta), 0\}, \tag{8}
$$

The limit of the above unconstrained problem when $\alpha \to 0$ then leads to the solution of the original constrained problem. As an approximation, WSR chooses a fixed weight α, and uses the gradient of $\nabla_\theta f + \frac{1-\alpha}{\alpha}\nabla_\theta g$ instead of $\nabla_\theta f + \frac{1-\alpha}{\alpha}\nabla_\theta \min\{g(\theta), 0\}$, thus the final solution will intensely rely on the selection of α.

C.2 Calculation of D_W

We use deterministic part of policies in the calculation of D_W, i.e., we remove the Gaussian noise on the action space in PPO and use $D_W(a_1, a_2) = |a_1 - a_2|$.

C.3 Network Structure

We use MLP with 2 hidden layers as our actor models in PPO. The first hidden layer is fixed to have 32 units. We choose to use 10, 64 and 256 hidden units for the three tasks respectively in all of the main experiments, after taking the success rate, performance and computation expense (i.e. the preference to use less unit when the other two factors are similar) into consideration.

C.4 Training Timesteps

We fix the training timesteps in our experiments. The timesteps are fixed to be 1M in Hopper-v3, 1.6M for Walker2d-v3 and 3M for HalfCheetah-v3.

D Visualize Diversity

D.1 Mujoco Locomotion

In this section, we provide some qualitative results of IPD on the Mujoco locomotion tasks. In all of our experiments we use the vanilla Mujoco locomotion benchmarks, with the default settings on defining healthy states. Although otherwise the visualization of learned policies might become more diverse (e.g., a Hopper agent may learn to stand-up after falling down while another agent may learn to move forward on the ground if we set the z-axis healthy threshold as 0).

With the method of IPD, the Hopper policies (Figure 5) learns to jump further and avoids falling down rather instead of just jumping and falling down (Figure 6). In the Walker2d environment, the color of purple indicates the left leg is visible. It can be seen that the IPD policies (Figure 7) learn to use both left and right legs in walking, while the PPO policies usually learn jumping. (Figure 8). In HalfCheetah, the IPD policies (Figure 9) perform much better than the PPO policies (Figure 10). The IPD policies leran to run with head-downward (Figure 9 line 1), head-upward (Figure 9 line 3), and forward (Figure 9 line 5) while the PPO policies are always head-downward.

In Hopper and HalfCheetah, IPD is able to improve the primal task performance by avoiding always getting trapped in some certain sub-optimal behaviors.

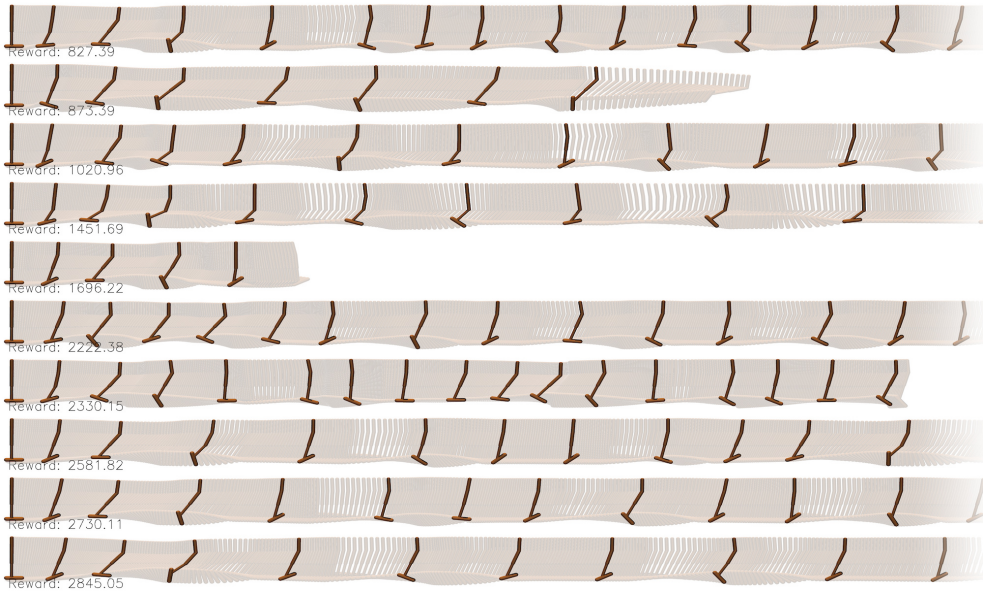

Figure 5: The visualization of policy behaviors of agents trained by our method in Hopper-v3 environment. Agents learn to jump with different strides.

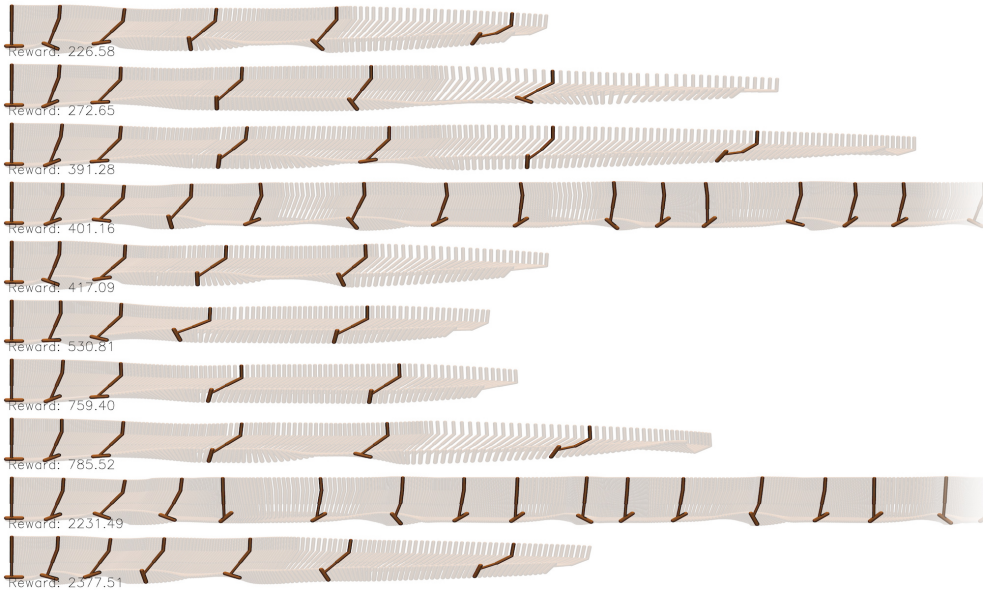

Figure 6: The visualization of policy behaviors of agents trained by PPO in Hopper-v3 environment. Most agents learn a policy that can be described as *Jump as far as possible and fall down*, leading to relative poor performance.

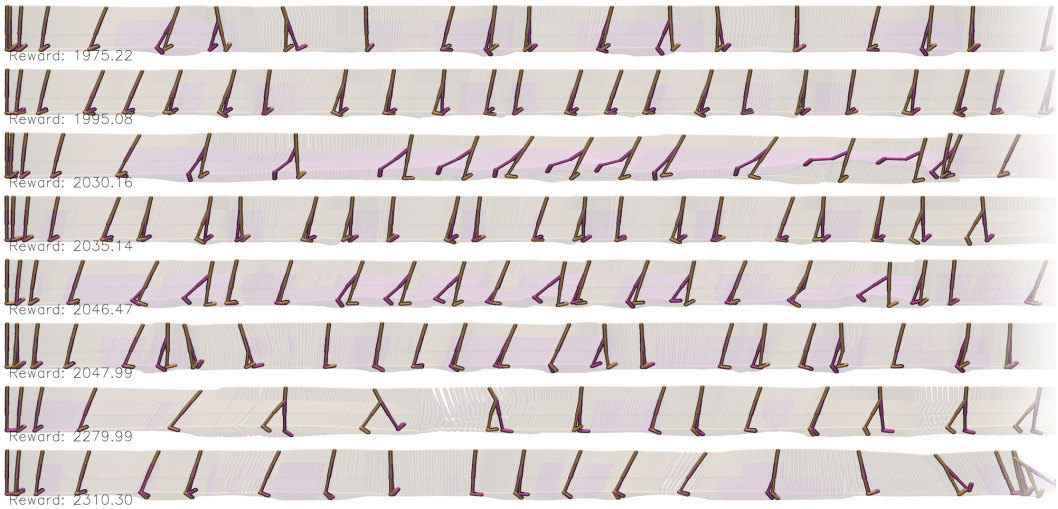

Figure 7: The visualization of policy behaviors of agents trained by our method in Walker2d-v3 environment. Instead of bouncing at the ground using both legs, our agents learns to use both legs to step forward.

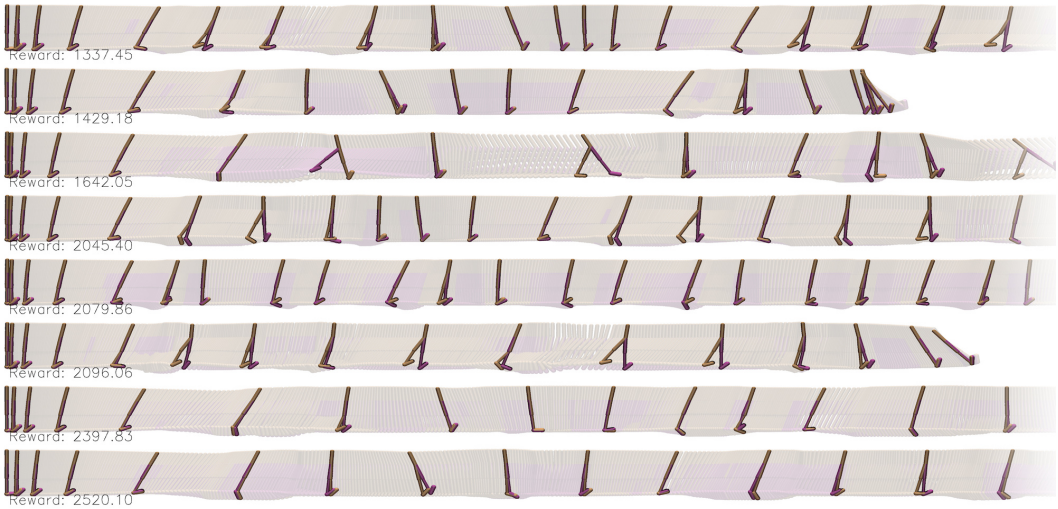

Figure 8: The visualization of policy behaviors of agents trained by PPO in Walker2d-v3 environment. Most of the PPO agents only learn to use their right leg to support the body and jump forward.

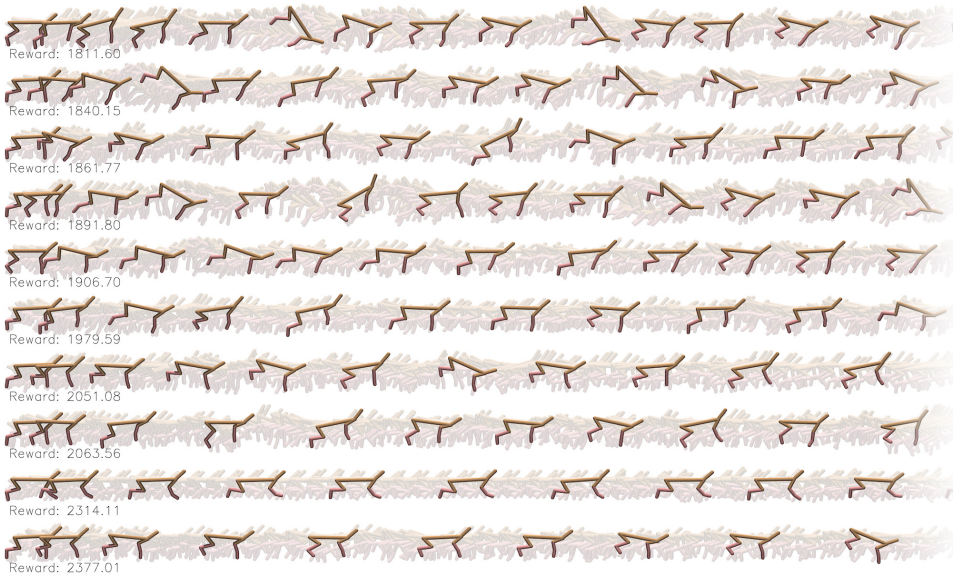

Figure 9: The visualization of policy behaviors of agents trained by our method in HalfCheetah-v3 environment. Our agents run much faster compared to PPO agents and at the mean time several patterns of motion have emerged.

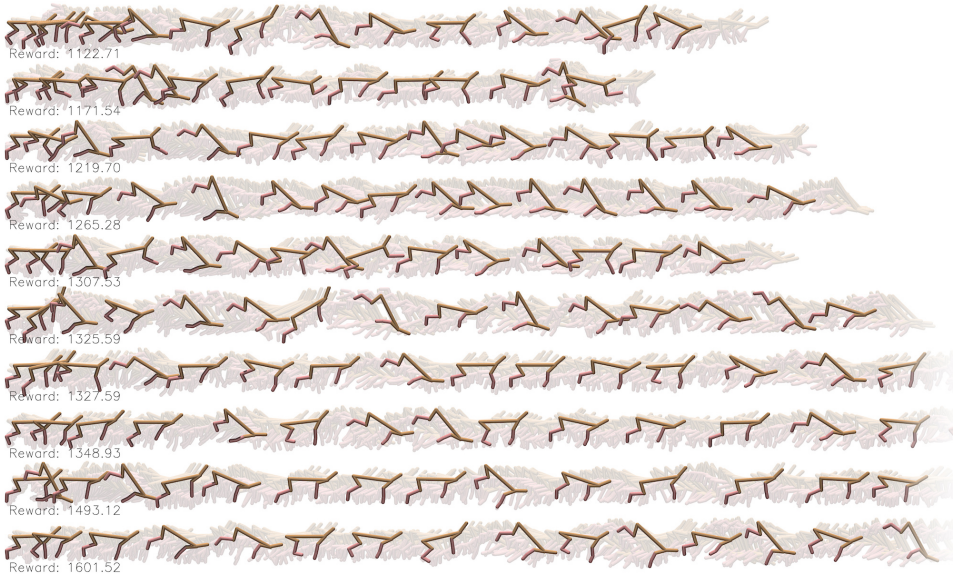

Figure 10: The visualization of policy behaviors of agents trained by PPO in HalfCheetah-v3 environment. Since we only draw fixed number of frames in each line, in the limited time steps the PPO agents can not run enough distance to leave the range of our drawing, which shows that our agents run much faster.