Appendix: Multi-policy Grounding and Ensemble Policy Learning for Transfer Learning with Dynamics Mismatch

A Interpretations of MPG Objective Function

We begin with the definitions of marginal transition distributions.

$$\rho_{\pi,\pi_g}(s,a,s') = (1-\gamma)\pi(a|s)P_g(s'|s,a)\sum_{t=0}^{\infty}\gamma^t p(s_t=s|\pi,P_g)$$

where $P_g(s'|s,a) = \sum_{\tilde{a}\in A} P_{src}(s'|s,\tilde{a})\pi_g(\tilde{a}|s,a)$, and $p(s_t=s|\pi,P_g)$ is the probability of being at state s at time t under dynamics P_g and π. Similarly, for target dynamics P_{trg},

$$\rho_{\pi,trg}(s,a,s') = (1-\gamma)\pi(a|s)P_{trg}(s'|s,a)\sum_{t=0}^{\infty}\gamma^t p(s_t=s|\pi,P_{trg})$$

Objective function of GARAT is derived from minimizing the convex conjugate function $\psi^*_{GA}(\rho_{\pi,\pi_g}-\rho_{\pi,trg})$ of the cost regularizer ψ_{GA} [Ho and Ermon, 2016; Desai *et al.*, 2020]. Also, optimal value to this objective function is known to minimize Jensen-Shannon divergence of two distributions $D_{JS}(\rho_{\pi,\pi^*_g},\rho_{\pi,trg})$ with constant differences [Goodfellow *et al.*, 2014].

$$\begin{aligned} J_{GARAT} &= \min_{\pi_g\in\Pi_g}\psi^*_{GA}(\rho_{\pi,\pi_g}-\rho_{\pi,trg}) \\ &= \min_{\pi_g\in\Pi_g} D_{JS}(\rho_{\pi,\pi_g},\rho_{\pi,trg}) + C \end{aligned} \tag{A.1}$$

when C is constant scalar value and $\psi_{GA}:\mathbb{R}^{S\times A\times S}\mapsto\mathbb{R}\cup\{\infty\}$ is defined as follows:

$$\psi_{GA}(c) = \begin{cases}\mathbb{E}_{\pi,P_{trg}}[g(c(s,a,s'))] & \text{if } c<0\\ +\infty & \text{otherwise}\end{cases}$$

$$\text{where } g(x) = \begin{cases}-x-\log(1-e^x) & \text{if } x<0\\ +\infty & \text{otherwise}\end{cases}$$

This ψ^*_{GA} measures how close the marginal transition distribution ρ_{π,π_g} in grounded environment is to the target distribution $\rho_{\pi,trg}$.

Objective function for multi-policy grounding Dynamics matching via IfO algorithms means finding π_g through the transition samples of the target environment by interpreting that the samples were obtained with the unknown expert policy π^*_g for the augmented MDP. Since the dynamics of augmented MDP depends on the rollout policy, an IfO problem for dynamics matching can be defined for each rollout policy $\pi_i\in\{\pi_1,...,\pi_K\}$. Therefore, Equation (A.1) is an objective function to find π_g for a particular rollout policy π.

The purpose of MPG is simultaneously matching the marginal transition distributions of K rollout policies through a common π_g. MPG considers the mixture of marginal transition distributions of K rollout policies for this purpose.

$$\begin{aligned} J_{MPG} &= \min_{\pi_g\in\Pi_g}\max_D \sum_{i=1}^{K}\big[\mathbb{E}_{\pi_i,P_g}[\log D(s,a,s')] \\ &\qquad + \mathbb{E}_{\pi_i,P_{trg}}[\log(1-D(s,a,s'))]\big] \end{aligned} \tag{A.2}$$

$$= \min_{\pi_g\in\Pi_g} K\cdot\psi^*_{GA}\Big(\frac{1}{K}\sum_{i=1}^{K}\rho_{\pi_i,\pi_g} - \frac{1}{K}\sum_{i=1}^{K}\rho_{\pi_i,trg}\Big) \tag{A.3}$$

$$= \min_{\pi_g\in\Pi_g} K\cdot D_{JS}\Big(\frac{1}{K}\sum_{i=1}^{K}\rho_{\pi_i,\pi_g},\frac{1}{K}\sum_{i=1}^{K}\rho_{\pi_i,trg}\Big) + C \tag{A.4}$$

where $D(s,a,s'):S\times A\times S\mapsto(0,1)$ distinguishes if the observed transition (s,a,s') is from the grounded or target environment when K rollout policies has been deployed. Equation (A.2), presented in the main draft, is an objective function for a practical algorithm based on generative adversarial networks. MPG iteratively trains discriminator D and π_g based on Equation (A.2). Equation (A.3) express the equation (A.2) through the convex conjugate function ψ^*_{GA} in the same way as the objective formula J_{GARAT} of single policy grounding (SPG), and this convex conjugate function measure the discrepancy between the mixture distributions in the target dynamics and in the grounded dynamics. This implies that MPG reduces the Jensen-Shannon divergence of two mixture distributions which becomes zero if all individual marginal transition distributions are matched.

B Proofs

Proposition 1. *If there exists* $\bar{\pi}_g \in \Pi_g$ *such that* $P_g(s'|s,a) = P_{trg}(s'|s,a)\ \forall s, s' \in S, a \in A$*, then* J_{MPG} *is minimized by* $\bar{\pi}_g$*.*

Proof. Jensen-Shannon divergence of mixture of marginal transition distributions is bounded by the average of Jensen-Shannon divergence of distributions.

$$D_{JS}(\frac{1}{K}\sum_{i=1}^{K}\rho_{\pi_i,\pi_g}, \frac{1}{K}\sum_{i=1}^{K}\rho_{\pi_i,trg}) \leq \frac{1}{K}\sum_{i=1}^{K} D_{JS}(\rho_{\pi_i,\pi_g}, \rho_{\pi_i,trg})$$

Hence,

$$\begin{aligned} J_{MPG} &= \min_{\pi_g\in\Pi_g}\max_{D}\sum_{i=1}^{K}\big[\mathbb{E}_{\pi_i,P_g}[\log D(s,a,s')] \\ &\qquad + \mathbb{E}_{\pi_i,P_{trg}}[\log(1-D(s,a,s'))]\big] \\ &= \min_{\pi_g\in\Pi_g} K\cdot D_{JS}(\frac{1}{K}\sum_{i=1}^{K}\rho_{\pi_i,\pi_g}, \frac{1}{K}\sum_{i=1}^{K}\rho_{\pi_i,trg}) \\ &\qquad + C \\ &\leq \min_{\pi_g\in\Pi_g}\sum_{i=1}^{K} D_{JS}(\rho_{\pi_i,\pi_g},\rho_{\pi_i,trg}) + C \end{aligned}$$

For $\bar{\pi}_g$ satisfying the condition and any $\pi \in \Pi$, $D_{JS}(\rho_{\pi,\bar{\pi}_g}, \rho_{\pi,trg}) = 0$ which is the achievable lowest value. Thus,

$$\bar{\pi}_g = \operatorname*{argmin}_{\pi_g\in\Pi_g} K\cdot D_{JS}(\frac{1}{K}\sum_{i=1}^{K}\rho_{\pi_i,\pi_g}, \frac{1}{K}\sum_{i=1}^{K}\rho_{\pi_i,trg})$$

which implies J_{MPG} is minimized by $\bar{\pi}_g$. □

C Details of Experiment Settings

C.1 Environments

We have used default InvertedPendulum, Hopper, and HalfCheetah of OpenAI gym [Brockman *et al.*, 2016] with Mujoco physic simulator [Todorov *et al.*, 2012] as the source environments.

InvertedPendulum is a task to control the pendulum standing on the cart note to fall over, and the state is a 4-dimensional (D) vector representing the position and velocity of the cart and pendulum. The cart is controlled by applying a 1-D force, and we defined the target task in which an extra force of $\epsilon = 1.5$ acts to cart compared to the source environment. Therefore, the optimal action transformation is $\tilde{a}^* = a + \epsilon$.

Hopper is a planar monopod robot consist of a torso, two legs, and feet, and the goal of the task is moving forward without falling over during episode length (1000). We control the robot with three actuated joints, hence having 3-D action space, and 11-D state space represents joint angles, joint velocities, and more. The target task, BrokenHopper, is an environment in which the first of the three actuators does not work, and the optimal action transformation is $\tilde{a}^* = (0, a_2, a_3)$ for $a = (a_1, a_2, a_3)$.

In HalfCheetah, we aim to move forward a robot consists of a torso and two legs as fast as possible without falling over. The action space is a 6-D space representing the control of the 6 actuators, and the state space is a 17-D space representing the joint angles, joint velocities, and more. BrokenCheetah is similarly defined to BrokenHopper by disabling the first one of the six actuators; $\tilde{a}^* = (0, a_2, a_3, a_4, a_5, a_6)$ for $a = (a_1, a_2, a_3, a_4, a_5, a_6)$ [Eysenbach *et al.*, 2020]. HeavyCheetah is defined as increasing the torso mass from 14 to 20 [Desai *et al.*, 2020]. Optimal action transformation in HeavyCheetah is not trivially defined.

C.2 Hyperparameters

Grounding-and-then-Policy Learning (GPL) approach consists of two steps: (1) grounding the source environment and (2) policy learning in the grounded environment. For grounding, we solve the Imitation from Observation (IfO) problem through the generative adversarial approach. Table C.1 shows the parameters to learn action transformation policy π_g with TRPO algorithm, and Table C.2 shows the parameters for training discriminative classifier which distinguishes transition samples. After grounding, we learn policy to be deployed at target task with SAC algorithm. Table C.3 shows the parameters when we learn a policy with SAC. We use the same parameters to learn the source optimal policies (rollout policies) and the target optimal policies.

Parameters	**Value**
Total time steps	5e6 for Inverted Pendulum 20e6 for the others
Hidden layers	2
Layer size	64
Time steps per batch	2048
Maximum KL divergence threshold	0.01
Iterations for conjugate gradient computation	15
Conjugate gradient damping factor	0.1
Weight for entropy loss	0.0
Discount factor	0
λ for GAE [Schulman *et al.*, 2015]	0.95
Value function iterations	5
Value function step size	3e-4 for Inverted Pendulum 1e-3 for the others

Table C.1: Parameters of TRPO algorithm for learning π_g

D Overfitting of SPG

Figure 1 shows heatmap of action transformation error averaged over $100,000$ samples, where x- and y-axis represent

Parameters	Value
Batch size per update	256
Learning rate	3e-4
Hidden layers	2
Layer size	256
Entropy coefficient	0.001
Gradient penalty coefficient	10

Table C.2: Parameters for discriminator

Parameters	Value
Batch size per update	256
Learning rate	3e-4
Replay buffer size	1000000
Learning starts	1000
Training frequency	1
Gradient steps	1
Hidden layers	2
Layer size	256
Entropy coefficient	Automatic adjustment [Haarnoja *et al.*, 2018]

Table C.3: Parameters of SAC algorithm for learning agent policy

the indices of grounded environment and those of rollout policy used to compute the error, respectively. Specifically, x- and y-axis mean policy π_g and policy π in computing action transformation error: $\mathbb{E}_{(s,a,\tilde{a})\sim\pi,\pi_g,P_{src}}[\|\tilde{a}^* - \tilde{a}\|_2]$. The left column of three graphs are for SPG and the right for MPG applied to three environments. Note that the interpretation of the x-axis is different between SPG and MPG. In SPG, grounded environment i is obtained (grounded) using samples collected by a single rollout policy π_i; whereas in MPG, grounded environment i is obtained using samples collected by all five rollout policies. The five grounded environments of MPG are generated by solving the same augmented MDP using five random seeds in RL algorithm used. The rollout policy in y-axis, on the other hand, has the same interpretation between SPG and MPG: a policy used in sampling trajectories from the grounded environment for the purpose of computing the expectation.

In Figure 1, MPG has smaller errors for all transfer scenarios compared to SPG. In particular, SPG is overfitted to the rollout policy used for grounding. Each grounded environment obtained through SPG has the least error if the transition samples are obtained with the rollout policy (grounded environment index is equal to the evaluation policy index). Otherwise, action transformation error increases when other policies different from the rollout policy are used for evaluation. MPG, on the other hand, grounds the dynamics across multiple rollout policies so avoids being overfitted to the particular policy. The poor grounding of SPG may, though not always, lead to poor performance of policy, which was found to be optimal in the grounded environment, at the target task.

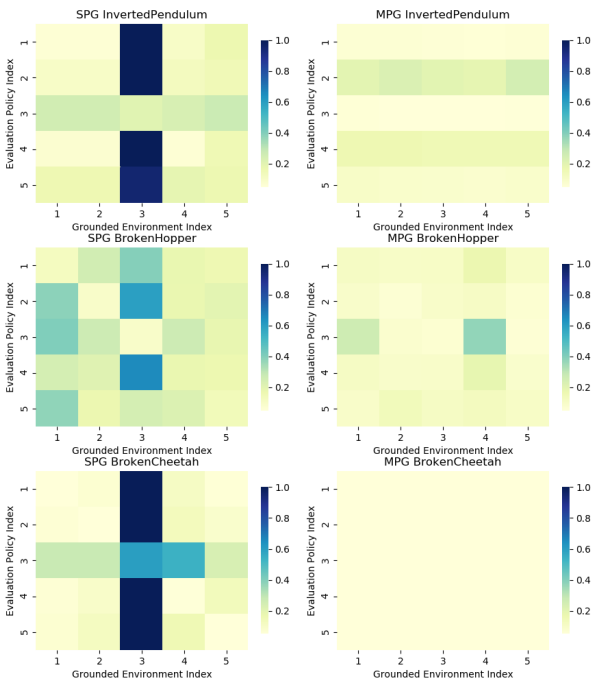

Figure 1: Comparison of action transformation errors when using 100,000 target transition samples

E Full Results of Numerical Experiments

We compared the performance at the target task of obtained agent policies by each algorithm, and part of the results are shown in Figure 4. Figure 4 selectively shows the results for SPG, MPG with $K = 5$, and MPGE with $K = 5, N = 5$. We provide the entire test results for each algorithm. Table E.1, Table E.2, Table E.3, and Table E.4 show the performance of the proposed algorithms in InvertedPendulum with $\epsilon = 1.5$, BrokenHopper, BrokenCheetah, and HeavyCheetah, respectively. Average episodic return values are presented with target optimal values and source optimal values in the target task without normalization.

References

[Brockman *et al.*, 2016] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

[Desai *et al.*, 2020] Siddharth Desai, Ishan Durugkar, Haresh Karnan, Garrett Warnell, Josiah Hanna, Peter Stone, and AI Sony. An imitation from observation approach to transfer learning with dynamics mismatch. *Advances in Neural Information Processing Systems*, 33, 2020.

[Eysenbach *et al.*, 2020] Benjamin Eysenbach, Swapnil Asawa, Shreyas Chaudhari, Ruslan Salakhutdinov, and

Target samples	SPG	MPG-ER, $K = 5$	MPGE-ER, $K = 5, N = 5$	MPG-MS, $K = 5$	MPGE-MS, $K = 5, N = 5$
500	630.0 (492.9)	738.1 (396.9)	1000.0 (0.0)	681.6 (440.1)	989.8 (14.8)
1,000	292.9 (260.1)	547.6 (362.8)	1000.0 (0.0)	635.8 (414.3)	993.6 (5.7)
5,000	807.8 (412.5)	804.4 (385.7)	972.9 (35.6)	971.1 (52.8)	1000.0 (0.0)
10,000	515.8 (393.8)	790.9 (415.6)	996.1 (8.8)	968.3 (48.2)	990.8 (20.6)
50,000	788.3 (437.5)	970.8 (50.6)	811.5 (411.2)	993.6 (14.2)	970.9 (50.9)
100,000	606.0 (371.0)	964.0 (56.0)	982.0 (40.3)	995.1 (8.6)	983.9 (29.2)
Target optimal policy: 1000.0 (0.0), Source optimal policy at target task: 33.9 (31.6)					

Table E.1: Performances at target task for **InvertedPendulum with** $\epsilon = 1.5$. Parenthesis indicates standard deviation over 5 different random seeds.

Target samples	SPG	MPG-ER, $K = 5$	MPGE-ER, $K = 5, N = 5$	MPG-MS, $K = 5$	MPGE-MS, $K = 5, N = 5$
500	223.1 (193.7)	454.7 (135.6)	409.5 (175.2)	444.8 (123.0)	568.0 (106.4)
1,000	197.4 (221.0)	399.6 (207.5)	621.2 (20.6)	528.4 (160.9)	645.4 (65.5)
5,000	355.3 (207.1)	450.1 (182.5)	677.4 (22.2)	550.6 (30.9)	624.3 (58.8)
10,000	341.4 (198.7)	416.8 (159.0)	669.9 (101.0)	478.5 (187.0)	578.0 (81.0)
50,000	267.7 (188.6)	512.0 (125.3)	682.1 (68.6)	497.1 (167.8)	609.0 (113.0)
100,000	374.0 (195.0)	397.8 (114.8)	481.8 (191.9)	389.7 (191.3)	623.8 (109.8)
Target optimal policy: 862.2 (117.8), Source optimal policy at target task: 164.5 (101.4)					

Table E.2: Performances at target task for **BrokenHopper**. Parenthesis indicates standard deviation over 5 different random seeds.

Target samples	SPG	MPG-ER, $K = 5$	MPGE-ER, $K = 5, N = 5$	MPG-MS, $K = 5$	MPGE-MS, $K = 5, N = 5$
500	3456.5 (768.9)	2437.2 (1652.2)	4764.9 (600.7)	2782.7 (838.9)	4653.4 (673.7)
1,000	3848.6 (1487.1)	3780.1 (467.8)	4820.4 (941.5)	4336.3 (707.3)	5296.3 (686.2)
5,000	4474.4 (2070.5)	5368.9 (604.8)	6069.5 (484.1)	5284.2 (599.5)	5870.1 (736.9)
10,000	4378.7 (2439.6)	4936.7 (2161.6)	5777.7 (457.1)	5865.5 (508.8)	5569.6 (911.5)
50,000	4839.8 (2494.2)	5258.0 (1402.4)	6034.9 (612.7)	5793.8 (813.9)	5801.0 (1039.2)
100,000	5117.1 (1300.3)	5762.6 (773.9)	6301.9 (294.5)	5650.0 (829.5)	5920.4 (1004.7)
Target optimal policy: 6470.2 (289.2), Source optimal policy at target task: 1611.7 (1317.8)					

Table E.3: Performances at target task for **BrokenCheetah**. Parenthesis indicates standard deviation over 5 different random seeds.

Target samples	SPG	MPG-ER, $K = 5$	MPGE-ER, $K = 5, N = 5$	MPG-MS, $K = 5$	MPGE-MS, $K = 5, N = 5$
500	2643.4 (850.6)	2994.4 (447.3)	5245.6 (1013.7)	3621.8 (634.2)	5303.6 (850.5)
1,000	3116.3 (1058.9)	4437.5 (675.3)	6636.4 (711.7)	3625.0 (649.5)	5179.4 (869.3)
5,000	4969.7 (1460.0)	4372.6 (929.7)	6394.3 (603.1)	4772.9 (1099.6)	5804.8 (1218.7)
10,000	4795.7 (402.2)	5238.5 (814.9)	7082.9 (846.1)	4711.2 (381.7)	6094.0 (471.5)
50,000	4986.1 (1223.7)	5548.9 (856.1)	5878.8 (508.6)	5583.6 (486.0)	5839.9 (551.3)
100,000	4965.3 (1148.2)	5796.8 (818.4)	6270.0 (795.1)	5043.8 (720.5)	5503.8 (543.2)
Target optimal policy: 5226.9 (1063.8), Source optimal policy at target task: 3396.2 (870.1)					

Table E.4: Performances at target task for **HeavyCheetah**. Parenthesis indicates standard deviation over 5 different random seeds.

Sergey Levine. Off-dynamics reinforcement learning: Training for transfer with domain classifiers. *arXiv preprint arXiv:2006.13916*, 2020.

[Goodfellow *et al.*, 2014] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27:2672–2680, 2014.

[Haarnoja *et al.*, 2018] Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, et al. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018.

[Ho and Ermon, 2016] Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. *Advances in neural information processing systems*, 29:4565–4573, 2016.

[Schulman *et al.*, 2015] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

[Todorov *et al.*, 2012] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE, 2012.