Appendix: Latent Wasserstein Adversarial Imitation Learning

Our appendix is organized as follows. In Sec. A, we discuss ICVF and provide a more detailed explanation of Eq. (6). In Sec. B, we provide the details of the environment in our experiments (Sec. B.1), the dataset used in our experiments (Sec. B.2), the hyperparameters we used for our method (Sec. B.3), and the details for our baselines (Sec. B.4). In Sec. D, we summarize the notation used in our paper. Finally, in Sec. E, we state the computational resource used for running our experiments.

A Extended Preliminaries

Intention Conditioned Value Function (ICVF). Intuitively, $V(s, s_+, z)$ is designed to evaluate the likelihood of the following question: *How likely am I to see s_+ if I act to perform z from state s?* The learning of ICVF is similar to other value-learning algorithms. ICVF satisfies the following Bellman equation:

$$\begin{aligned} V(s, s_+, z) &= \mathbb{E}_{a\sim\pi_z^*}\left[\mathbb{I}(s = s_+) + \gamma\mathbb{E}_{s'\sim P_z(\cdot|s_t)}\left[V(s', s_+, z)\right]\right], \\ \text{where } \pi_z^* &= \arg\max_a r_z(s) + \gamma\mathbb{E}_{s'}\left[V(s', z, z)\right]. \end{aligned} \tag{10}$$

Here, (s, s') is a transition and $P_z(s_{t+1}|s)$ is the transition probability from s_t to s_{t+1} when acting according to intent z. Further, r_z defines the agent's objective for a particular intention z. Note, $r_z(s)$ is not the ground truth reward signal. Instead, it describes whether a state s is desirable by intent z and thus depends on data; in other words, the agent aims to maximize the reward specified by r_z when pursuing intention z. The original reward is not needed in ICVF training.

The original paper adopts implicit Q-learning (IQL) for ICVF learning. In one update batch, we sample transition (s, s'), potential future outcome s_+, and intent z. Similar to the original IQL (Kostrikov et al., 2022), we update the critic with asymmetric critic losses to avoid out-of-distribution overestimation. To do this, we apply different weights on critic loss with respect to the positivity of *advantage*. Note, as we care about whether the transition (s, s') corresponds to acting with intention z, our goal s_+ is equal to z. Thus, the advantage A is defined as:

$$A = r_z(s) + \gamma V_\theta(s', z, z) - V_\theta(s, z, z). \tag{11}$$

Following that, the critic loss is defined as:

$$\mathcal{L}(V_\theta) = \mathbb{E}_{(s,s'),z,s_+}\left[|\alpha - \mathbb{I}(A < 0)|(V_\theta(s, s_+, z) - \mathbb{I}(s = s_+) - \gamma V_{\text{target}}(s', s_+, z))^2\right]. \tag{12}$$

B Experimental Details

B.1 Environments

We use five MuJoCo (Todorov et al., 2012) and D4RL (Fu et al., 2020) environments: Maze2d, hopper, halfcheetah, walker2d and ant. The environment specifications for maze2d are provided in Sec. 4.1. In this section, we will briefly introduce the other MuJoCo environments. Fig. 7 provides an illustration of those environments.

1. **Hopper.** The hopper environment (as well as the other three environments) is a locomotion task. In hopper, the agent needs to control a single-legged robot leaping forward in a 2D space with x- and z-axis. The 11-dimensional state space encompasses joint angles and velocities of the robot, while the 3-dimensional action space corresponds to torques applied on each joint.
2. **Halfcheetah.** In the Halfcheetah environment, the agent needs to control a cheetah-shaped robot to sprint forward. It also operates in a 2D space with x- and z-axis, but has a 17-dimensional state representing joint positions and velocities, and a 6-dimensional action space that modulates joint torques.

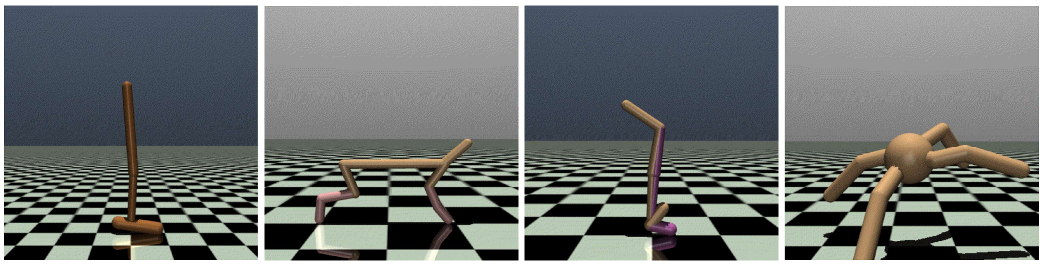

Figure 7: Illustration of the MuJoCo (Todorov et al., 2012) environments we test in Sec. 4.2.

3. **Walker2d.** As implied by its name, in Walker2d, the agent needs to control a 8-DoF bipedal robot to walk in the two dimensional space. It has a 27-dimensional state space and an 8-dimensional action space.
4. **Ant.** Different from the other three environments, the Ant environment is a 3D setting where the agent navigates a four-legged robotic ant moving towards a particular direction. The state is represented by 111 dimensions, including joint coordinates and velocities, while the action space has 8 dimensions.

B.2 Datasets

For expert datasets of the MuJoCo environments, we use 1 trajectory from the D4RL expert dataset, which has 1000 steps. Some baselines such as PWIL (Dadashi et al., 2021) employ a *subsampling* hyperparameter, which creates a low-data training task by taking only one state/state-action pair from every 20 steps of the expert demonstration. For fairness, we set all baselines' subsampling factors to be 1, i.e., no subsampling.

Dataset	Size	Normalized Reward (Expert is 100)
Hopper-random-v2	999996	1.19 ± 1.16
HalfCheetah-random-v2	1000000	0.07 ± 2.90
Walker2d-random-v2	999997	0.01 ± 0.09
Ant-random-v2	999930	6.36 ± 10.07

Table 2: The basic statistics of the random datasets from D4RL (Fu et al., 2020) applied in our experiments. It is apparent that all these data are of very low quality compared to an expert, yet our ICVF-learned metric still works well.

B.3 Hyperparameters

Tab. 3 summarized the hyperparameters for our method. We use the same settings for all environments, and keep hyperparameters identical to TD3 (Fujimoto et al., 2018) and ICVF (Ghosh et al., 2023) whenever possible.

B.4 Baselines

We use several different github repositories for our baselines. We use default settings of those repos, except for the number of expert trajectories (which is set to 1) and the subsampling factor (see Appendix B.2). Below are the repos we used in our experiments for each baseline:

- *BC (Ross et al., 2011), GAIL (Ho & Ermon, 2016), AIRL (Fu et al., 2018):* We use the *imitation* (Gleave et al., 2022) library, which provides clean implementations of several imitation learning algorithms and has a MIT license.
- *OPOLO (Zhu et al., 2020), DACfO (Kostrikov et al., 2019), BCO (Torabi et al., 2018a), GAIfO (Torabi et al., 2018b):* We use OPOLO's official code (`https://github.com/`

Type	Hyperparameter	Value	Note
ICVF.	Network Size of ϕ	[256, 256]	same as original paper
Disc.	Network Size	[64, 64]	
	Activation Function	ReLU	
	Learning Rate	0.001	
	Update Epoch	40 steps	
	Update interval	4000	
	Batch Size	4000	
	Optimizer	Adam	
	Gradient Penalty coefficient	10	
Actor	Network Size	[256, 256]	
	Activation Function	ReLU	
	Learning Rate	0.0003	
	Training length	1M steps	
	Batch Size	256	
	Optimizer	Adam	
Critic	Network Size	[256, 256]	
	Activation Function	ReLU	
	Learning Rate	0.001	
	Training Length	1M steps	
	Batch Size	256	
	Optimizer	Adam	
	γ	0.99	discount factor

Table 3: Summary of the hyperparameters of LWAIL.

	Hopper	**HalfCheetah**	**Walker**	**Ant**	**Average**
1 trajectory	110.52 ± 1.06	86.71 ± 5.67	105.30 ± 2.33	80.56 ± 13.09	95.77
5 trajectories	107.65 ± 7.47	93.28 ± 1.97	107.32 ± 1.36	87.23 ± 10.43	98.87
All expert dataset	109.34 ± 3.87	94.18 ± 3.12	104.37 ± 1.97	90.81 ± 9.61	99.67

Table 4: Ablation on using multiple trajectories as expert demonstrations. Our method shows consistent expert-level performance regardless of the number of expert demonstrations.

`illidanlab/opolo-code`), where DACfO, BCO and GAIfO are integrated as baselines, which does not have a license.

- *OLLIE (Yue et al., 2024):* We tried to use the official code but it can't be executed due to non-trivial typos. Thus we use their reported numbers on random dataset instead.
- *PWIL:* We use another widely adopted imitation learning repository (Arulkumaran & Ogawa Lillrank, 2023) (`https://github.com/Kaixhin/imitation-learning`), which has an MIT license.
- *WDAIL:* We use their official code (`https://github.com/mingzhangPHD/Adversarial-Imitation-Learning/tree/master`), which does not have a license.
- *IQlearn:* We use their official code (`https://github.com/Div99/IQ-Learn/tree/main`) with a research-only license.

C More Ablations

In this section, we provide additional ablation results of our method. We report normalized reward (higher is better) for all results.

C.1 Multiple trajectories

To demonstrate robustness of our method even if the expert data is scarce, we test our method with 5 expert trajectories and the whole expert dataset (1M transitions). Tab. 4 summarizes the results. We observe consistent compelling performance regardless of the number of expert trajectories.

C.2 Embeddings

In this section, we compare our method with ICVF embeddings to use of other embeddings. It is worth noting that while there are embedding methods for RL/IL, most of them are not applica-

	Hopper	HalfCheetah	Walker	Ant	Average
LWAIL	110.52 ± 1.06	86.71 ± 5.67	105.30 ± 2.33	80.56 ± 13.09	**95.77**
PW-DICE	110.60 ± 0.77	46.07 ± 27.95	106.63 ± 1.03	85.36 ± 8.12	87.16
CURL	105.70 ± 1.22	87.62 ± 5.10	102.97 ± 4.19	52.03 ± 8.33	87.08
No Embedding	108.34 ± 3.42	85.98 ± 3.42	62.39 ± 20.43	40.72 ± 18.95	74.36

Table 5: Ablation of different embedding methods with LWAIL. The result shows that ICVF embeddings outperform other contrastive learning-based embeddings.

	Hopper	HalfCheetah	Walker	Ant	Average
LWAIL	110.52 ± 1.06	86.71 ± 5.67	105.30 ± 2.33	80.56 ± 13.09	**95.77**
LWAIL_subsample	109.00 ± 0.46	86.73 ± 7.02	106.13 ± 2.47	83.21 ± 8.80	**96.27**
WDAIL_subsample	108.21 ± 4.90	35.41 ± 2.07	114.32 ± 2.07	83.87 ± 10.92	85.45
IQlearn_subsample	60.26 ± 14.21	4.12 ± 1.03	8.31 ± 1.48	5.32 ± 3.87	19.50

Table 6: Ablation on subsampled expert trajectories. The result shows that LWAIL is robust to subsampled expert demonstrations and outperforms other baselines with subsampled expert demonstrations.

ble to our scenario. For instance, most empirical state embedding methods are for visual environments (Meng et al., 2023; Sermanet et al., 2018) or for cross-domain dynamics matching (Duan et al., 2017; Franzmeyer et al., 2022). Among theoretical state embedding methods, low-rank MDPs (Modi et al., 2024) are not applicable to the MuJoCo environment, and bisimulation (Zhang et al., 2020a) requires a reward signal which is not available in imitation learning.

Nonetheless, we identify two contrastive learning-based baselines that are most suitable for our scenario: CURL (Laskin et al., 2020) and PW-DICE (Yan et al., 2024). Both methods use InfoNCE (Oord et al., 2018) as their contrastive loss for better state embeddings. Their difference: 1) CURL updates embeddings with an auxiliary loss during online training, while PW-DICE updates embeddings before all other training; 2) CURL compares the current state with different noises added as positive contrast examples, while PW-DICE uses the next states as positive contrast samples. Tab. 5 summarizes the results. The result shows that 1) state embeddings generally aid learning; and 2) our proposed method works best.

C.3 Subsample

To validate the robustness of our policy, we provide results with subsampled expert trajectories, a widely-adopted scenario in many prior works such as PWIL and IQ-learn. Only a small portion of the complete expert trajectories are present. Our subsample ratio is 10, i.e., we take 1 expert state pair out of adjacent 10 pairs. Tab. 6 summarizes the results, which show that 1) our method with subsampled trajectories outperforms Wasserstein-based baselines such as WDAIL (Zhang et al., 2020b) and IQlearn (Garg et al., 2021), and 2) the performance of our method is not affected by incomplete expert trajectories.

C.4 Downstream RL algorithm

We used TD3 as our downstream RL algorithm rather than PPO with entropy regularizer. Our choice is motivated by better efficiency and stability, especially because TD3 is an off-policy algorithm which is more robust to the shift of the reward function and our adversarial training pipeline. We ablate this choice of the downstream RL algorithm and show that TD3 outperforms PPO in our framework. Tab. 7 summarizes the results.

C.5 ICVF Embedding with Other Methods

We also show that our proposed solution outperforms existing methods with ICVF embedding, both Wasserstein-based (IQlearn, WDAIL) and f-divergence based. The results are summarized in Tab. 8 (using average reward; higher is better). We find that 1) our method outperforms prior methods with ICVF embedding, and 2) ICVF does not necessarily improve the performance of prior methods,

	Hopper	HalfCheetah	Walker	Ant	Average
LWAIL+TD3 (original)	110.52 $\pm$ 1.06	86.71 $\pm$ 5.67	105.30 $\pm$ 2.33	80.56 $\pm$ 13.09	**95.77**
LWAIL+PPO	65.21 $\pm$ 4.81	1.02 $\pm$ 0.21	24.13 $\pm$ 2.14	9.12 $\pm$ 0.85	24.87

Table 7: Ablation on downstream RL algorithms. The result shows that TD3 works much better than PPO.

	Hopper	HalfCheetah	Walker	Ant	Average
LWAIL	110.52 $\pm$ 1.06	86.71 $\pm$ 5.67	105.30 $\pm$ 2.33	80.56 $\pm$ 13.09	**95.77**
WDAIL+ICVF	110.02 $\pm$ 0.53	30.07 $\pm$ 2.32	68.68 $\pm$ 9.16	3.42 $\pm$ 1.01	53.04
IQlearn+ICVF	29.80 $\pm$ 10.12	3.82 $\pm$ 0.98	6.54 $\pm$ 1.23	8.91 $\pm$ 0.45	12.27
GAIL+ICVF	8.96 $\pm$ 2.09	0.12 $\pm$ 0.40	3.98 $\pm$ 1.41	-3.09 $\pm$ 0.85	2.49

Table 8: ICVF with other methods. Our method far outperforms other methods with ICVF embeddings.

due to other components of our method (e.g., normalized input for the Wasserstein discriminator, downstream RL algorithm).

C.6 Mismatched Dynamics

It is worth noting that the very motivation of LWAIL is to find a latent space which aligns well with the environment's true dynamics. Despite this, we agree that there might be cases where the latent space employed in LWAIL does not align with the true dynamics due to inaccurate data, e.g., mismatched dynamics between expert demonstrations and the actual environment. To test such cases, we use the halfcheetah mismatched experts scenario analyzed in SMODICE (Ma et al., 2022): for expert demonstration, the torso of the cheetah agent is halved in length, thus causing inaccurate alignment. We compared our methods with the results reported in the SMODICE paper. Tab. 9 summarizes the final average normalized reward (higher is better). Results show that 1) our method works better than several baselines including SMODICE; and 2) our method is robust to mismatched dynamics.

C.7 Sigmoid Reward Mapping

We adopt the sigmoid function to regulate the output of our neural networks for better stability (similar to WDAIL (Zhang et al., 2020b)). However, one cannot naively apply the sigmoid to the reward function for better performance. To show this, we compare to TD3 with a sigmoid function applied to the ground truth reward. The result is illustrated in Tab. 10. The result shows that a naive sigmoid mapping of the reward does not improve TD3 results.

C.8 Pseudo-Reward Metric Curve

To validate the effect of using sigmoid and ICVF embedding for our pseudo-reward generated by f, we conduct two experiments:

1) Run a standard setting of LWAIL, and compare pseudo-rewards generated by f with the sigmoid function, and pseudo-rewards without the sigmoid function for the MuJoCo environments. This is illustrated in Fig. 8.

2) Run standard LWAIL and LWAIL without ICVF embedding, and compare pseudo-rewards (with the sigmoid function) for the MuJoCo environments. This is illustrated in Fig. 9.

The result clearly shows that both ICVF-embedding and sigmoid function are very important for pseudo-reward stability and positive correlation with ground-truth reward.

D List of Notations

Tab. 11 summarizes the symbols which appear in our paper.

	Normalized Reward
LWAIL	**24.31** ± **4.51**
SMODICE	**23.2** ± **7.43**
SAIL	0 ± 0
ORIL	2.47 ± 0.32

Table 9: Performance on the Halfcheetah environment with mismatched dynamics. Our method outperforms baselines.

Environment	**Hopper**	**HalfCheetah**	**Walker**	**Ant**	**Maze2D**	**Average**
TD3	105.54	76.13	89.68	89.21	120.14	96.14
TD3+Sigmoid reward	84.23	30.76	42.55	34.79	119.03	62.27

Table 10: Results of TD3 with and without sigmoid applied on the ground truth reward. The results show that applying the sigmoid function does not yield better performance.

E Computational Resources

All our experiments are performed with an Ubuntu 20.04 server, which has 128 AMD EPYC 7543 32-Core Processor and a single NVIDIA RTX A6000 GPU. With these resources, our method needs about 65 − 75 minutes for the MuJoCo environments.

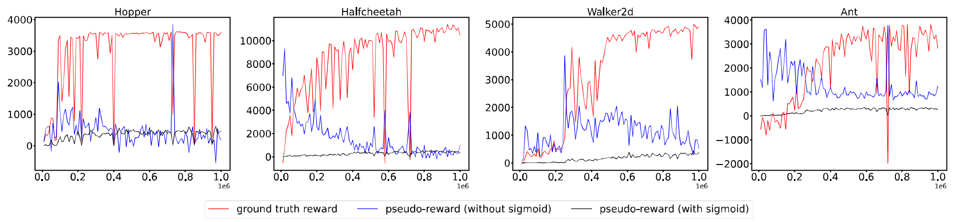

Figure 8: The reward curves of pseudo- and ground-truth reward in a single training session, where pseudo-reward is generated by f following Alg. 1 and serves as the reward signal for our downstream TD3. We note that the pseudo-reward is much more stable and positively correlated with ground-truth reward when using a sigmoid function.

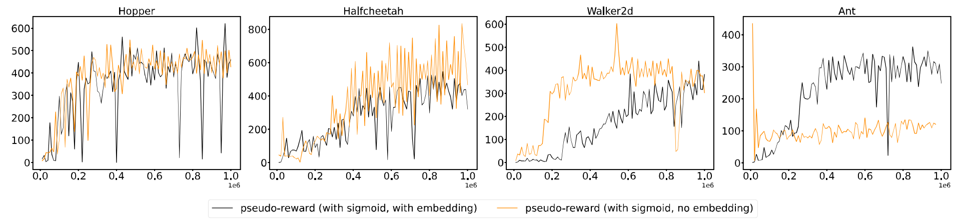

Figure 9: The pseudo-reward curves with and without ICVF embedding in a single training session. We note that without ICVF, the pseudo-reward is generally less stable (e.g. fluctuation in halfcheetah and sudden drop in walker2d and ant) and sometimes less correlated with ground-truth reward (e.g. ant environment).

Name	Meaning	Note
$\mathcal{S}$	State space	
s	State	$s \in \mathcal{S}$
$\mathcal{A}$	Action space	
a	Action	$a \in \mathcal{A}$
t	Time step	$t \in \{0, 1, 2, \dots\}$
γ	Discount factor	$\gamma \in [0, 1)$
r	Reward function	$r(s, a)$ for single state-action pair
P	Transition	$P(s'\|s, a) \in \Delta(\mathcal{S})$
E	Expert dataset	state-only expert demonstrations
I	Random dataset	state-action trajectories of very low quality
π	Learner policy	The policy we aim to optimize
d_s^π	State occupancy of π	$d_s^\pi(s) = (1-\gamma)\sum_{i=0}^{\infty}\gamma^i \Pr(s_i = s)$, where s_i is the i-th state in a trajectory
d_{ss}^π	State-pair occupancy of π	$d_s^\pi(s, s') = (1-\gamma)\sum_{i=0}^{\infty}\gamma^i \Pr(s_i = s, s_{i+1} = s')$, where s_i is the i-th state in a trajectory
d_{ss}^E	State-pair occupancy of expert policy	The expert policy here is empirically induced from E
c	Underlying metric for Wasserstein distance	
f	Dual function / Discriminator	Dual function in Rubinstein dual form of 1-Wasserstein distance; also a discriminator from adversarial perspective and a reward model from IRL perspective
Π	Wasserstein matching variable	In our case, $\sum_{s\in\mathcal{S}}\Pi(s, s') = d_s^E(s')$, $\sum_{s'\in\mathcal{S}}\Pi(s, s') = d_s^\pi(s)$
$\mathcal{W}_1$	1-Wasserstein distance	
s_+	Outcome state	
z	Latent intention	
V	Value function	takes s, s_+, z as input in ICVF; only takes s in normal RL
V_{target}	Target value	target value function in the critic objective of RL
$\mathbb{I}$	indicator function	$\mathbb{I}[\text{condition}] = 1$ if the condition is true, and $= 0$ otherwise
ϕ	State representation (embedding)	the embedding function we use for f; $\phi(s) \in \mathbb{R}^d$
T	Counterfactual intention	$T(z) \in \mathbb{R}^{d\times d}$
ψ	Outcome representation	$\psi(s_+) \in \mathbb{R}^d$
α	ICVF constant	$\alpha \in (0.5, 1]$
σ	Sigmoid function	

Table 11: A list of symbols used in the paper. The first part focuses on RL-specific symbols. The second part details Wasserstein-specific notation. The third part summarizes ICVF-specific symbols (Sec. 3.2).