Time Your Rewards: Learning Temporally Consistent Rewards from a Single Video Demonstration

Huaxiaoyue Wang[*], **William Huey**[*], **Anne Wu**,
Yoav Artzi, **Sanjiban Choudhury**
Cornell University

Abstract: Designing reward functions for tasks with high-dimensional motion sequences, such as controlling humanoid robots, is difficult. A more intuitive approach is to use video demonstrations to specify the desired behavior. Recently, optimal transport (OT) has become popular for learning rewards by aligning learner and demonstration trajectories. However, OT faces two key challenges. First, it lacks temporal constraints, which are crucial for tasks where subgoals must be completed in a specific order. Second, poorly designed reward functions can lead to local minima, allowing the agent to exploit undesired behaviors. Our key insight is to structure the reward function to enforce temporal consistency. We propose a novel class of reward functions `SDTW+`, which uses Soft Dynamic Time Warping (SDTW) to align trajectories in the correct order and adds a cumulative reward bonus to encourage continuous progress. In experiments, agents trained with `SDTW+` achieve a 91.7% success rate on six sequence-following tasks in the Mujoco `Humanoid-v4` environment, significantly outperforming OT-based methods.

1 Introduction

Designing reward functions for reinforcement learning (RL) agents is a tedious process, especially when controlling humanoid robots with high degrees of freedom. For example, training a humanoid to perform specific motion sequences, such as coordinated hand and arm signals, requires explicitly defining high-dimensional subgoals and the conditions for transitioning between them. A more natural and efficient alternative is to provide a video demonstration, which directly conveys the desired behaviors without the need for manually crafted task specifications.

Inverse reinforcement learning (IRL) [1, 2] provides a foundational framework for learning reward functions from expert demonstrations. At its core, IRL can be viewed as a distribution matching problem between the learner and the demonstrator, often formulated as optimizing Integral Probability Metrics (IPMs) [3, 4]. Recent works have leveraged optimal transport (OT) [5] to do such matching in high-dimensional feature space [6–12]. However, current approaches face two significant challenges: First, certain tasks require completing subgoals in a specific order. OT fails to enforce this ordering because its formulation lacks temporal constraints, allowing it to ignore the order of subgoals when matching distributions between the learner and the demonstration. Second, the reward function must avoid local minima that may lead to reward hacking, where the agent learns to exploit undesired loopholes [13–15]. A naive reward function that only enforces temporal constraints can lead to an agent that lingers at subgoals and collects immediate rewards without making meaningful progress.

Our key insight is to structure the reward function to enforce temporal consistency. Instead of using OT, which matches trajectories without considering temporal order, we build on Soft Dynamic Time Warping (SDTW) [16] to constrain alignment paths, ensuring that the agent follows the correct

[*]Denotes equal contribution. Correspondence to: Huaxiaoyue Wang (yukiwang@cs.cornell.edu) and William Huey (wph52@cornell.edu).

sequence of actions. However, we find that SDTW alone does not sufficiently drive task completion. To address this, we introduce a cumulative reward bonus that encourages the agent to make continuous progress, preventing it from stalling at intermediate subgoals.

We propose a method, Soft-DTW Plus (`SDTW+`), that (1) uses a visual encoder to compute the distance for all pairs of frames between the learner and demonstration trajectories, (2) based on the distances, solves for the optimal alignment path using SDTW and accumulates reward bonuses to calculate the per-timestep reward. We show that agents trained with the `SDTW+` rewards outperform OT by 56.2%, achieving 91.7% success rate in the Mujoco `Humanoid-v4` environment.

Our key contributions are:

1. A novel sequence-matching reward function class `SDTW+` that enforces the agent to follow the demonstration in the correct order and encourages it to continuously make meaningful progress.
2. A robust visual encoder, finetuned on privileged robot state, that estimates its uncertainty to calculate reliable visual distance between two frames.
3. Experiments showing that the `SDTW+` reward can effectively and efficiently train RL agents to achieve 100.0% success rate in the 2D-Navigation environments and 91.7% success rate in the MujoCo `Humanoid-v4` environment.

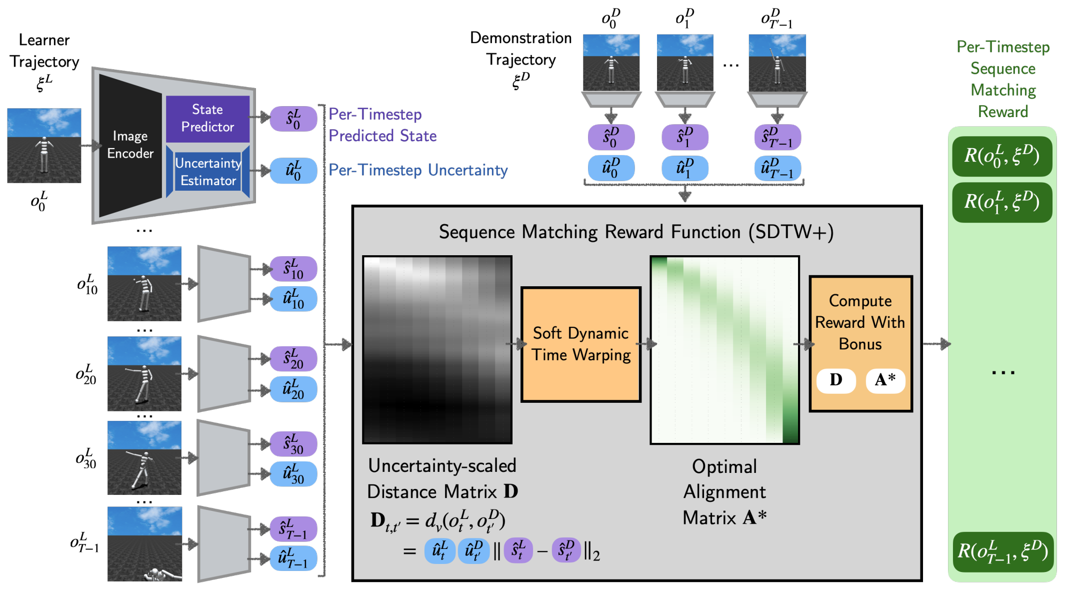

Learner Trajectory ξ^L
o_0^L
Image Encoder
State Predictor
Uncertainty Estimator
$\hat{s}_0^L$
$\hat{u}_0^L$
Per-Timestep Predicted State
Per-Timestep Uncertainty
o_{10}^L
o_{20}^L
o_{30}^L
o_{T-1}^L
$\hat{s}_{10}^L$ $\hat{u}_{10}^L$
$\hat{s}_{20}^L$ $\hat{u}_{20}^L$
$\hat{s}_{30}^L$ $\hat{u}_{30}^L$
$\hat{s}_{T-1}^L$ $\hat{u}_{T-1}^L$
Demonstration Trajectory ξ^D
o_0^D o_1^D o_{T-1}^D
$\hat{s}_0^D$ $\hat{s}_1^D$ $\hat{s}_{T-1}^D$
$\hat{u}_0^D$ $\hat{u}_1^D$ $\hat{u}_{T-1}^D$
Sequence Matching Reward Function (SDTW+)
Soft Dynamic Time Warping
Compute Reward With Bonus
D A*
Uncertainty-scaled Distance Matrix **D**
$\mathbf{D}_{t,t'} = d_v(o_t^L, o_{t'}^D)$
$= \hat{u}_t^L \hat{u}_{t'}^D \| \hat{s}_t^L - \hat{s}_{t'}^D \|_2$
Optimal Alignment Matrix **A***
Per-Timestep Sequence Matching Reward
$R(o_0^L, \xi^D)$
$R(o_1^L, \xi^D)$
$R(o_{T-1}^L, \xi^D)$

Figure 1: **Approach overview.** The goal is to define a per-timestep sequence-matching reward for a visual learner trajectory $\xi^L = \{o_t^L\}_{t=0}^{T}$ given a visual demonstration trajectory $\xi^D = \{o_{t'}^D\}_{t'=0}^{T'}$. (1) Each frame is passed through the visual encoder to predict the robot state and estimate uncertainty. (2) Given the uncertainty-scaled visual distance matrix, SDTW computes the optimal alignment matrix. (3) This matrix is used to compute per-timestep rewards and apply reward bonuses, resulting in the final sequence-matching reward.

2 Preliminaries

2.1 Problem Formulation

We model the problem as a Markov Decision Process (MDP) defined by the tuple $(\mathcal{S}, \mathcal{A}, P, r, \gamma)$, where $\mathcal{S}$ is the state space and $\mathcal{A}$ is the action space. At each timestep t, the agent occupies a state $s_t \in \mathcal{S}$ and receives a visual observation $o_t \in \mathcal{O}$, which is an image of the agent's state and the environment. Upon taking action $a_t \in \mathcal{A}$, the environment transitions to a new state $s_{t+1} \in \mathcal{S}$ according to the transition dynamics $P(s_{t+1}|s_t, a_t)$, and produces a new observation $o_{t+1} \in \mathcal{O}$.

However, the reward function for this task is unknown. Instead, we assume access to a *single* visual demonstration of the task, $\xi^D = \{o^D_{t'}\}_{t'=0}^{T'-1}$, which the robot policy must imitate. Importantly, this demonstration may be temporally unaligned with the learner's trajectory. Additionally, the demonstration lacks state and action labels, rendering classical imitation learning inapplicable. Instead, we approach this as an inverse reinforcement learning (IRL) problem, where the reward function $\mathcal{R}(o^L_t, \xi^D)$ is defined as a function of the visual demonstration ξ^D and the learner's observations. The goal is to learn a policy $\pi^*(a|s)$ that maximizes the expected discounted sum of rewards:

$$\pi^* = \arg\max_{\pi} \mathbb{E}_{\xi^L \sim p(\xi^L|\pi)} \left[\sum_{t=0}^{T-1} \gamma^t \mathcal{R}(o^L_t, \xi^D) \right]$$

where $\xi^L = \{o^L_t\}_{t=0}^{T-1}$ is the visual learner trajectory, and γ the discount factor.

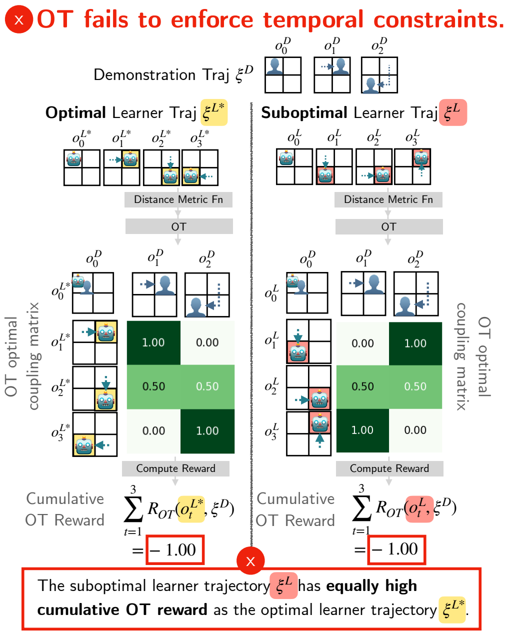
2.2 Reward Function Candidates and Failure Modes

To construct the reward function $\mathcal{R}(o^L_t, \xi^D)$, we examine the failure cases of optimal transport (OT) and consider other sequence-matching algorithms similar to OT as potential candidates.

Optimal Transport fails to enforce temporal constraints. Optimal Transport (OT) [5] finds the optimal coupling to transport one distribution to another distribution. Prior works in imitation learning specifically leverage the Wasserstein distance to measure how closely a learner trajectory matches a demonstration trajectory [6–12]. Given a learner trajectory $\xi^L = \{o^L_t\}_{t=0}^{T-1}$ and a demonstration trajectory $\xi^D = \{o^D_{t'}\}_{t'=0}^{T'-1}$, the corresponding learner distribution is defined as $\rho^L = \frac{1}{T}\sum_{t=0}^{T-1} \delta_{o^L_t}$, where $\delta_{o^L_t}$ is a Dirac distribution centered on o^L_t, and the demonstration trajectory's distribution is $\rho^D = \frac{1}{T'}\sum_{t'=0}^{T'-1} \delta_{o^D_{t'}}$. Given a distance metric $d(o^L_t, o^D_{t'})$, OT solves for the optimal μ^* within the set of coupling matrices $M = \{\mu \in \mathbb{R}^{T\times T'} : \mu\mathbf{1} = \rho^L, \mu^T\mathbf{1} = \rho^H\}$: $\mu^* = \arg\min_{\mu\in M} \sum_{t=0}^{T-1}\sum_{t'=0}^{T'-1} d(o^L_t, o^D_{t'})\mu_{t,t'}$.

Then, the reward function is defined as

$$\mathcal{R}_{OT}(o^L_t, \xi^D) = -\sum_{t'=0}^{T'-1} d(o^L_t, o^D_{t'})\mu^*_{t,t'}. \tag{1}$$



OT fails to enforce temporal constraints.
Demonstration Traj ξ^D
o_0^D o_1^D o_2^D
Optimal Learner Traj ξ^{L*}
Suboptimal Learner Traj ξ^L
o_0^{L*} o_1^{L*} o_2^{L*} o_3^{L*}
o_0^L o_1^L o_2^L o_3^L
Distance Metric Fn
OT
OT optimal coupling matrix
1.00 0.00
0.50 0.50
0.00 1.00
0.00 1.00
0.50 0.50
1.00 0.00
Compute Reward
Cumulative OT Reward
$\sum_{t=1}^{3} R_{OT}(o_t^{L*}, \xi^D)$
$= -1.00$
$\sum_{t=1}^{3} R_{OT}(o_t^{L}, \xi^D)$
$= -1.00$
The suboptimal learner trajectory ξ^L has **equally high cumulative OT reward** as the optimal learner trajectory ξ^{L*}.

Figure 2: **Failure cases for OT reward in the 2D-Navigation environment.** The suboptimal learner trajectory ξ^L moves in the counter-clockwise direction while optimal one ξ^{L*} moves clockwise. Unlike the learner, the agent in the demonstration trajectory can move multiple cells per timestep.

However, OT fails to penalize violation of temporal constraints. Fig. 2 shows that OT fails for tasks that require completing subgoals in specific orders. A suboptimal trajectory that visits the subgoals in the *reversed* order has the same OT reward as the optimal trajectory. Appendix A.1 generalizes this observation.

Soft Dynamic Time Warping enforces temporal constraints. Soft Dynamic Time Warping (SDTW) overcomes OT's limitation by temporally aligning the learner and demonstration trajectories. Specifically, it imposes the time consistency constraints on alignment paths $A \in \Omega$ such that a path $A \in \mathbb{R}^{T\times T'}$ must align the beginning and the end of two trajectories (i.e., $A_{0,0} = 1$ and $A_{T-1,T'-1} = 1$), and its alignment indices must be monotonically non-decreasing in time. Given two trajectories ξ^L and ξ^D, and a distance metric $d(o^L_t, o^D_{t'})$, SDTW solves for the optimal probability distribution p^* over all alignment paths $A \in \Omega$. Mensch and Blondel [17] shows that the SDTW objective function can be written in an entropy regularized formulation controlled by a temperature term $\lambda \geq 0$: $p^* = \arg\max_p \sum_{A\in\Omega} \sum_{t=0}^{T-1} \sum_{t'=0}^{T'-1} [p(A)A]_{t,t'} d(o_t^L, o_{t'}^D) - \lambda H(p)$. Because of its time consistency constraints, SDTW can be solved in polynomial time via dynamic programming [16] using the soft-min operator: $\min^\lambda\{x_1, \ldots, x_n\} = -\lambda \log \sum_{i=1}^n e^{-x_i/\lambda}$. Note that Dynamic Time Warping (DTW), which uses the minimum operator instead, is a special case of SDTW when $\lambda = 0$ [16].

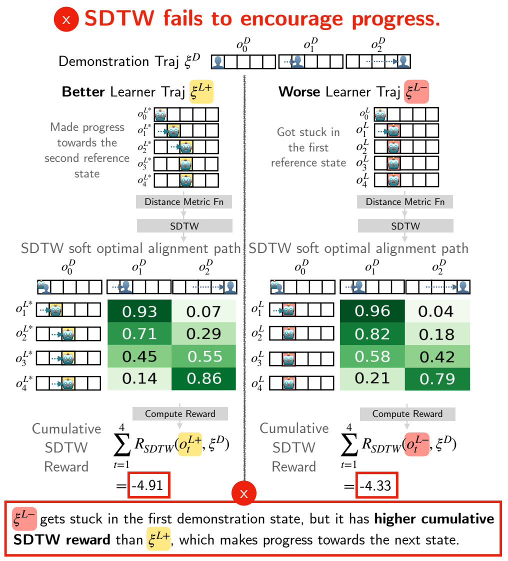
Given the optimal probability distribution p^* over all paths, the SDTW reward function is:

$$\mathcal{R}_{SDTW}(o_t^L, \xi^D) = -\sum_{t'=0}^{T'} d(o_t^L, o_j^D) A^*_{t,t'} \text{ where } A^* = \sum_{A\in\mathcal{A}} p^*(A)A \tag{2}$$

where A^* is the optimal soft alignment path defined based on the optimal distribution p^*.

SDTW's time consistency constraints allow it to avoid the example failure case of OT (shown in Fig. 9). However, Fig. 3 demonstrates SDTW's limitation: *it fails to incentivize progress along the demonstrated trajectory*. Consider a trajectory that makes one move of progress past the first demonstration state and another trajectory that just gets stuck in the first demonstration state. Even if the first trajectory does not reach the next demonstration state yet, it should have a higher reward because it makes more progress. Instead, the second trajectory has a higher SDTW reward. This is an example of a local minimum that could cause the agent to stall at intermediate states instead of making meaningful progress during RL training. Appendix A.2 proves that, when SDTW has temperature 0, there will always exist an unsuccessful trajectory with the same total DTW reward as the successful trajectory in the 2D-Navigation environment.

This investigation reveals two key desiderata for the reward function: (1) It must *enforce temporal constraints*, and (2) It must *assign higher reward to trajectories that make more progress*.


SDTW fails to encourage progress.
Demonstration Traj ξ^D
o_0^D
o_1^D
o_2^D
Better Learner Traj ξ^{L+}
Made progress towards the second reference state
o_0^{L*}
o_1^{L*}
o_2^{L*}
o_3^{L*}
o_4^{L*}
Worse Learner Traj ξ^{L-}
Got stuck in the first reference state
o_0^L
o_1^L
o_2^L
o_3^L
o_4^L
Distance Metric Fn
SDTW
SDTW soft optimal alignment path
0.93 0.07
0.71 0.29
0.45 0.55
0.14 0.86
0.96 0.04
0.82 0.18
0.58 0.42
0.21 0.79
Compute Reward
Cumulative SDTW Reward
$\sum_{t=1}^{4} R_{SDTW}(o_t^{L+}, \xi^D)$
$= -4.91$
$\sum_{t=1}^{4} R_{SDTW}(o_t^{L-}, \xi^D)$
$= -4.33$

ξ^{L-} gets stuck in the first demonstration state, but it has **higher cumulative SDTW reward** than ξ^{L+}, which makes progress towards the next state.

Figure 3: **Failure cases of soft dynamic time warping (SDTW).** The worse learner trajectory is stuck at the first demonstration state, while the better learner trajectory makes more progress towards the second demonstration state.

3 Approach

We propose `SDTW+`, a sequence-matching reward function that enforces temporal constraints while encouraging continuous progress in Section 3.1, and we introduce a finetuning framework to train a robust visual encoder that allows `SDTW+` to take videos as input and measure the visual distance $d_v(o_t^L, o_{t'}^D)$ between the learner and demonstration video frames in Section 3.2.

3.1 `SDTW+`: Sequence-Matching Reward with SDTW and Reward Bonus

We introduce the `SDTW+` reward function, which augments the SDTW reward (2) with a reward bonus to mitigate SDTW's issues in encouraging progress. After the SDTW reward is calculated and converted to non-negative values by computing the exponential of it, `SDTW+` accumulates a per-timestep reward bonus based on how far the learner makes progress along the demonstration trajectory in the SDTW alignment path. Given the SDTW reward $\mathcal{R}_{SDTW}(o_t^L, \xi^D)$, the `SDTW+`

reward function is:

$$\mathcal{R}_{SDTW+}(o_t^L, \xi^D) = \begin{cases} \mathcal{R}_{SDTW}(o_0^L, \xi^D) & \text{if } t = 0, \\ \mathcal{R}_{SDTW}(o_t^L, \xi^D) + \mathcal{B}(o_t^L, \xi^D) & \text{if } t > 0. \end{cases} \tag{3}$$

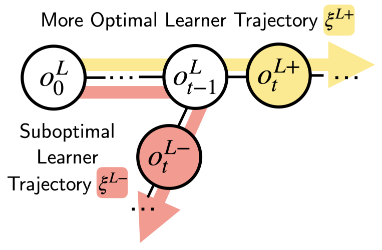
where $\mathcal{B}(o_t^L, \xi^D)$ is the cumulative reward bonus at each timestep t calculated via Algorithm 1.

Algorithm 1 `SDTW+` Reward Bonus $\mathcal{B}(o_t^L, \xi^D)$ Calculation

```
1: Input: SDTW per-timestep rewards $\{\mathcal{R}_{SDTW}(o_t^L, \xi^D)\}_{t=0}^{T-1}$, the SDTW optimal assignment
   path $A^*$, the timestep $t$ to calculate the bonus for.
2: Output: Accumulated reward bonus at timestep $t$: $\mathcal{B}(o_t^L, \xi^D)$
3: $\mathcal{B}(o_t^L, \xi^D) = 0$ // Initialize the reward bonus
4: // At learner's step 0, it is matched to the demo step 0 by temporal constraint
5: prev_best_match = 0
6: for $i \leftarrow 1$ to $t-1$ do
7:     // At step $i$, the demo step that the learner is primarily matched to
8:     curr_best_match = $\arg\max_{t'} A^*_{i,t'}$
9:     if curr_best_match > prev_best_match then
10:        // Made progress by matching to a later demo step, so adding reward from
   the previous step $i-1$
11:        $\mathcal{B}(o_t^L, \xi^D) \leftarrow \mathcal{B}(o_t^L, \xi^D) + \mathcal{R}_{SDTW}(o_{i-1}^L, \xi^D)$
12:    prev_best_match ← curr_best_match
13: Return $\mathcal{B}(o_t^L, \xi^D)$
```

We formalize the SDTW failure cases and show how `SDTW+`, which accumulates the SDTW reward from the previous timestep when the learner makes progress, mitigates this issue.

Problem Setup. Let $\xi^{L-} = \{o_0^L, \dots, o_{t-1}^L, o_t^{L-}, \dots\}$ and $\xi^{L+} = \{o_0^L, \dots, o_{t-1}^L, o_t^{L+}, \dots\}$ be two learner trajectories where observations from 0 to $t-1$ are identical but differ at o_t^{L-} and o_t^{L+}. We define ξ^{L-} to be a suboptimal trajectory, where, at timestep t, the learner stays close to the previous timestep's frame o_{t-1}^L instead of making progress, so $\mathcal{R}_{SDTW}(o_t^{L-}, \xi^D) \le \mathcal{R}_{SDTW}(o_{t-1}^L, \xi^D)$. In contrast, we define ξ^{L+} to be a more optimal trajectory, where at timestep t, it makes progress as its frame o_t^{L+} matches to the next demonstration frame compared to the previous timestep. Fig. 4 conceptually illustrates these two trajectories. Appendix A.3 shows that, depending on the visual distance between the more optimal frame o_t^{L+} and demonstration frames, o_t^{L+} can have a lower SDTW reward compared to the suboptimal frame o_t^{L-}. However, `SDTW+` can solve this problem:



More Optimal Learner Trajectory ξ^{L+}
o_0^L
o_{t-1}^L
o_t^{L+}
...
Suboptimal Learner Trajectory ξ^{L-}
o_t^{L-}
...

Figure 4: Two learner trajectories that have identical observations from timestep 0 to $t-1$. At timestep t, the more optimal trajectory ξ^{L+} makes progress, while the suboptimal trajectory's frame o_t^{L-} just stays close to the previous timestep's frame.

Proposition 1. *Under the problem setup stated above, the `SDTW+` cumulative reward bonus guarantees that the more optimal frame o_t^{L+} has a higher `SDTW+` reward than the suboptimal frame o_t^{L-}, i.e., $\mathcal{R}_{SDTW+}(o_t^{L+}, \xi^D) > \mathcal{R}_{SDTW+}(o_t^{L-}, \xi^D)$.*

The proof of Proposition 1 is deferred to Appendix A.3. In conclusion, the `SDTW+` reward function enforces temporal constraints because of its SDTW formulation and also encourages the agent to make progress, thereby satisfying the two key desiderata for sequence-matching reward functions.

3.2 Visual Distance Metric

The sequence-matching reward (Section 3.1) requires a function $d_v(o_t^L, o_{t'}^D)$ that computes the distance between two images o_t^L and $o_{t'}^D$. Empirically, we find that the representations of pretrained

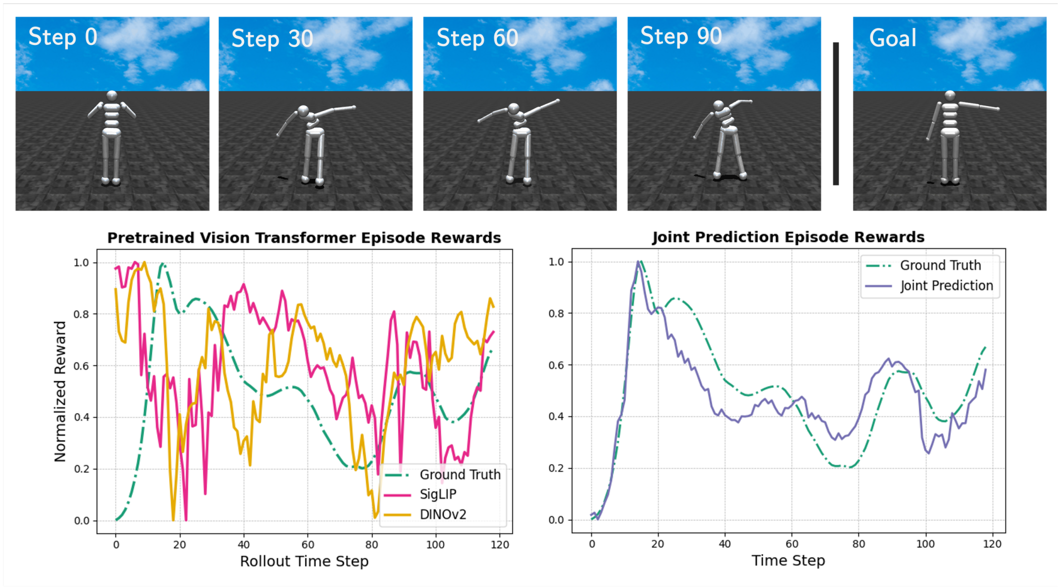


Figure 5: **(Left) Rewards of two pretrained models and (right) our joint prediction model on an example learner trajectory for the left arm out task**. To emphasize their shape, all rewards are normalized along the trajectory dimension. Rewards using the pretrained SigLIP-ViT-B-16 and DINOv2-ViT-B-14-reg models are based on the cosine similarity between the learner and demonstration embeddings. The fine-tuned joint prediction model (Section 3.2) provides a smoother reward curve. The trajectories are in the MuJoCo `Humanoid-v4` environment [18, 19], which is visually modified to mimic the setup of [20].

visual encoders produce noisy rewards. Fig. 5 shows that the reward signals sometimes are the opposite of the ground-truth reward (e.g., at around timestep 20). Even at timesteps when these rewards roughly follow the shape of the ground-truth, there is a significant amount of noise, matching the observations of [21] in the Meta-world environment.

We hypothesize that the pretrained encoders struggle because the environment is out of their training distribution. To address the domain gap, we collect a set of robot play data $\mathcal{D} = \{(o_i, j_i)\}_{i=1}^N$ containing images o_i of various poses and corresponding joint positions j_i. We attach a joint position prediction head f to an existing visual encoder ϕ , which we train using regression. Additionally, we train an autoencoder $g_{dec} \circ g_{enc}$ that minimizes the reconstruction loss $\mathcal{L}_{reco}(\mathbf{z}_i)$ given a visual embedding $\phi(o_i) = \mathbf{z}_i$. The reconstruction loss is defined as the squared Euclidean distance between the original and reconstructed embedding: $\mathcal{L}_{reco}(\mathbf{z}_i) = ||g_{dec}(g_{enc}(\mathbf{z}_i)) - \mathbf{z}_i||_2^2$.

Because out-of-distribution images are common during RL training, we utilize the reconstruction loss to estimate the model's uncertainty. First, we compute the mean μ_{reco} and standard deviation σ_{reco} of the loss on the offline dataset. Then, we use a formulation inspired by Frey et al. [22] to compute an uncertainty score $u(\mathbf{z_i})$:

$$u(\mathbf{z_i}) = \begin{cases} \epsilon, & \text{if} \mathcal{L}_{reco}(\mathbf{z_i}) < \mu_{reco} \\ 1 - \exp\left(\frac{(\mathcal{L}_{reco}(\mathbf{z_i}) - \mu_{reco})^2}{2(\sigma_{reco} k_\sigma)^2}\right) + \epsilon, & \text{otherwise} \end{cases} \tag{4}$$

where k_σ is a hyperparameter that controls the spread of the uncertainty function and ϵ ensures that the uncertainty is greater than 0. In our experiments, we set $k_\sigma = 2$ and $\epsilon = 1$.

Finally, let $d_j(o_t^L, o_{t'}^D)$ be the Euclidean distance between the predicted joint positions for both images. We use an uncertainty-scaled visual distance $d_v(o_i^L, o_j^D)$ for the sequence matching function:

$$d_v(o_t^L, o_{t'}^D) = u(\phi(o_t^L))u(\phi(o_{t'}^D))d_j(o_t^L, o_{t'}^D) \tag{5}$$

See Appendix B for more details on the visual dataset, model, training, and uncertainty scaling.

4 Experiments

4.1 Experimental Setup

Environments. We evaluate our approach across two environments (details in Appendix C.1):

- **2D-Navigation.** This discrete environment tests the sequence-matching reward function on demonstrations with different pacing from the learner (shown in Fig. 6). At each timestep, the learner can move one cell in a cardinal direction or stay at the current cell.
- **Humanoid.** The MuJoCo `Humanoid-v4` environment [18, 19] examines how well the sequence-matching reward function works with the visual distance metric. We define 6 tasks (shown in Fig. 8), and the goal is to follow the demonstration motion within the maximum 120 timesteps.

RL Policy. For the 2D-Navigation environment, we train PPO [23] for 100k steps and evaluate the policy's performance every 2000 steps on one environment (because the environment does not change). For the Humanoid environment, we train SAC [24] for 2M steps and evaluate the policy every 20k steps on 8 environments. Appendix C.2 contains RL training details and hyperparameters.

Baselines. We compare `SDTW+` (with the temperature parameter $\lambda = 5$) against baselines that use other sequence-matching algorithms: `OT` [25], `DTW` (SDTW with $\lambda = 0$), `SDTW` ($\lambda = 5$), and `DTW+` (which augments the DTW rewards by adding bonuses). For the 2D-Navigation environment, all approaches use a shortest-path distance metric based on the agent location. For the Humanoid environment, they use the visual distance metric in Section 3.2. We also include RoboCLIP [26], a transformer-based approach that uses a pretrained video-and-language model [27] to directly encode the video. It defines the reward for the last timestep as the cosine similarity between the learner video's and the demonstration video's embeddings, while all previous timesteps have 0 as the reward.

Metrics. We define the **success rate** as the percentage of a demonstration trajectory that the learner trajectory *matches in the same order as the demonstration*. This metric is calculated using the privileged state information that all reward functions do not have access to. For the 2D-Navigation environment, a learner state matches a demonstration state if the agent locations are identical. For the Humanoid environment, a learner trajectory has successfully matched a demonstration state if the agent can remain standing (torso height above 1.1) and its arm joint position s_t^L is close to the demonstration's arm joint position s_t^D ($e^{-||s_t^L - s_t^D||_2} > 0.50$) for $N = 3$ consecutive frames.

4.2 How well does `SDTW+` perform when demonstrations are faster than the learner?

In Fig. 6, the learner trained with `SDTW+` reward is the only one that converges to 1.0 success rate for both tasks, as it reaches all the demonstration states and follows the right order. We visualize the learner trajectories after being trained on each reward function in Fig. 12 in the Appendix. With the constant speed demonstration (where the demonstrator moves an equal amount of cells to reach each state), the `OT` learner nearly reaches all the states that the demonstration visits, but it travels in the opposite direction, thereby supporting the hypothesis that OT fails to enforce temporal constraints. Only the `DTW+` and `SDTW+` learners are able to reach the first state in the demonstration, but the `DTW+` learner gets stuck at this point. With the variable speed demonstration, similar behaviors occurred (i.e., the `OT`, `DTW+`, and `SDTW` learners all get stuck at one of the intermediate states in the demonstration).

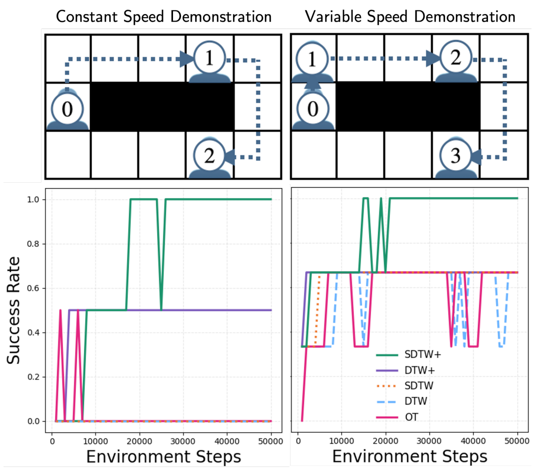


Figure 6: **Learning curves on 2D-Navigation environment tasks with a shortest-path based distance metric.** The policy and environment are deterministic, so confidence intervals are not shown.

4.3 How well does SDTW+ perform with demonstration videos in the humanoid domain?

Fig. 7 shows that, across all tasks, the learner trained with the SDTW+ reward achieves the highest IQM success rate compared to other baselines. Fig. 8 shows the average success rates over training for SDTW+ and three baselines. SDTW+ performs better than all baselines on $4/6$ tasks (Both Arms Out, Both Arms Down, Left/Right Arm Out) and performs slightly worse than DTW+ on $2/6$ tasks (Left/Right Arm Up). Meanwhile, RoboCLIP has the worst performance, suggesting that a sparse reward based on video embeddings is insufficient for sequence-following tasks.

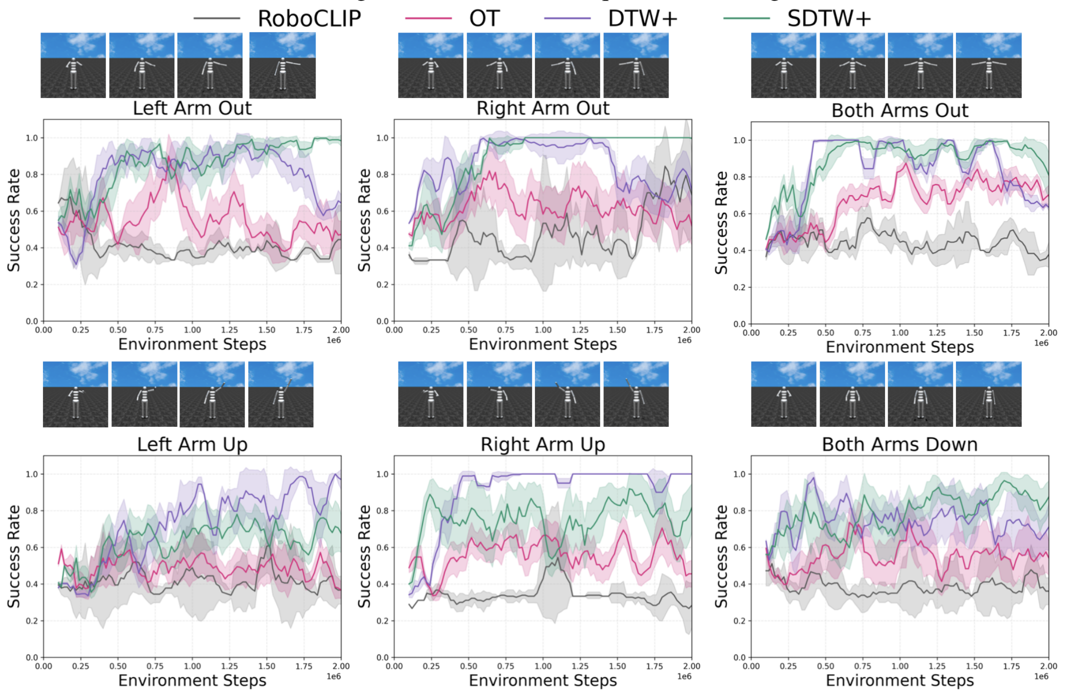


Figure 8: **Per-task learning curves on Humanoid Tasks with visual distance metrics.** SDTW+ performs the best on $4/6$ tasks. Results are averaged across 8 seeds and shown with 95% confidence intervals.

Empirically, the DTW+ approach matches a large portion of the learner's trajectory to the final frame in the demonstration (see Fig. 14), thereby also training an agent to achieve a relatively high success rate. We hypothesize that this leads to especially good performance on tasks like Left/Right Arm Up, where the final frame is much further from the initial frame. However, this unbalanced matching also causes training instability (where the performance degrades in $4/6$ tasks) because the DTW+ agent focuses mainly on reaching the final joint positions and ignores the need to match the previous positions in the demonstrations well. In contrast, the SDTW+ reward with a more distributed matching trains the agent more stably.

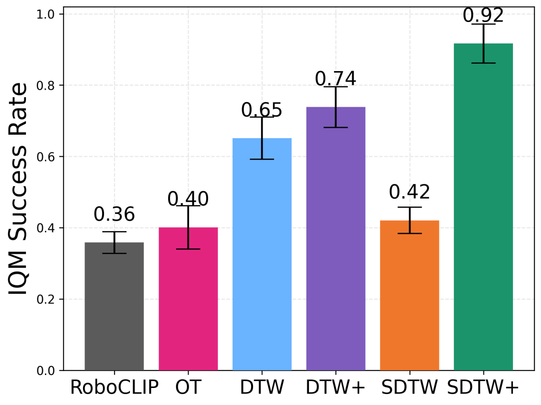


Figure 7: **IQM success rate for all approaches after training.** The SDTW+ reward function successfully trains the RL agent to achieve the highest success rate. Error bars show the 95% confidence intervals.

The results also validate our hypothesis about the failure case of OT. We observe that the OT coupling matrix is not temporally consistent, leading to a poorly shaped reward that causes worse training. For example, when the agent explores, it is temporarily closest to later frames in the demonstration, causing OT to match the agent's frame to these later frames. Then, when the agent loses its progress and moves closer to the earlier demonstration frames due to instability, `OT` matches the agent frame to earlier frames, leading to no reduction in the episode return. Appendix C.4 shows visualizations of this phenomenon.

Overall, with the `SDTW+` reward, the agent achieves a **better success rate** than the baselines and trains more stably.

5 Related Works

5.1 Sequence-Matching Algorithms Used in Robotics

Prior works use optimal transport [5], a sequence-matching algorithm, to define rewards for robotic tasks. In the imitation learning setting, given a teleoperated demonstration dataset, the reward or the loss function is defined as the optimal coupling between the learner trajectories' and the demonstrations' state or action distributions [6–8, 10, 28]. Without access to privileged states, recent works exploring using optimal transport to match the visual embedding between the learner and the demonstration trajectory [9, 11, 12, 29]. Similar to our work, they use a visual encoder (e.g. pretrained ResNet [30]) to embed the images in the trajectories. However, none of these focus on the sequence-following task that requires the agent to closely follow all steps of the demonstration. Meanwhile, Dynamic Time Warping (DTW) has mostly been used as an evaluation metric to quantify how well a robot trajectory can be matched to demonstration trajectories [31, 32] or a filtering metric to create a higher quality imitation learning demonstration dataset [33]. To the best of our knowledge, our work is the first to systematically explore how effectively each sequence-matching algorithm can function as a reward function for sequence-following tasks, and we are the first to effectively utilize soft dynamic time warping in reward functions.

5.2 Visual Rewards

Recent advancements in pretrained vision and multimodal models have raised interest in leveraging their features as reward sources for RL, especially when traditional reward functions are challenging to define. Prior works use reward functions derived from these models to directly specify tasks [20, 34, 35] or to provide supplementary reward signals [36, 37]. Similar to our goal, Sontakke et al. [26] learn a policy for manipulation tasks from a demonstration video using a pretrained video model [27] to generate trajectory-level sparse rewards. However, embeddings derived from these models can produce noisy rewards that hinder RL training [21, 38], and sparse rewards can fail to effectively train a policy in more complex environments.

6 Discussion

We investigate how to define the reward function for tasks that require an RL agent, such as a humanoid robot, to follow the sequence specified in a video demonstration. We present the `SDTW+` reward function, which utilizes SDTW to enforce time constraints so that the learner must follow the order in the demonstration and accumulates a reward bonus to encourage the learner to make meaningful progress. From eight tasks across two distinct domains, our approach outperforms baselines that use other sequence-matching algorithms in two distinct domains. With the `SDTW+` reward function, the RL agent is able to learn how to follow the video demonstration effectively.

In future directions, we are looking to (1) apply the `SDTW+` reward function to longer-horizon, more difficult tasks, such as those requiring periodic motions. Solving these tasks would require incorporating a memory module into the RL policy or a temporal encoding into the state representation because the action at a state will differ depending on the history of states visited. We also plan to (2) relax the dataset assumption needed to finetune the visual encoder. Currently, we rely on the offline robot play dataset to finetune the encoder and estimate the model's uncertainty. Future work would explore using online finetuning during RL training, inspired by [21], to improve performance.

References

[1] P. Abbeel and A. Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*, page 1, 2004.

[2] B. D. Ziebart, A. L. Maas, J. A. Bagnell, A. K. Dey, et al. Maximum entropy inverse reinforcement learning. In *Aaai*, volume 8, pages 1433–1438. Chicago, IL, USA, 2008.

[3] W. Sun, A. Vemula, B. Boots, and D. Bagnell. Provably efficient imitation learning from observation alone. In *International conference on machine learning*, pages 6036–6045. PMLR, 2019.

[4] G. Swamy, S. Choudhury, J. A. Bagnell, and S. Wu. Of moments and matching: A game-theoretic framework for closing the imitation gap. In *International Conference on Machine Learning*, pages 10022–10032. PMLR, 2021.

[5] G. Peyré and M. Cuturi. Computational optimal transport, 2020. URL https://arxiv.org/abs/1803.00567.

[6] H. Xiao, M. Herman, J. Wagner, S. Ziesche, J. Etesami, and T. H. Linh. Wasserstein adversarial imitation learning, 2019. URL https://arxiv.org/abs/1906.08113.

[7] R. Dadashi, L. Hussenot, M. Geist, and O. Pietquin. Primal wasserstein imitation learning. *arXiv preprint arXiv:2006.04678*, 2020.

[8] G. Papagiannis and Y. Li. Imitation learning with sinkhorn distances. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 116–131. Springer, 2022.

[9] S. Cohen, B. Amos, M. P. Deisenroth, M. Henaff, E. Vinitsky, and D. Yarats. Imitation learning from pixel observations for continuous control, 2022. URL https://openreview.net/forum?id=JLbXkHkLCG6.

[10] Y. Luo, Z. Jiang, S. Cohen, E. Grefenstette, and M. P. Deisenroth. Optimal transport for offline imitation learning. *arXiv preprint arXiv:2303.13971*, 2023.

[11] S. Haldar, J. Pari, A. Rai, and L. Pinto. Teach a robot to fish: Versatile imitation from one minute of demonstrations. *arXiv preprint arXiv:2303.01497*, 2023.

[12] S. Haldar, V. Mathur, D. Yarats, and L. Pinto. Watch and match: Supercharging imitation with regularized optimal transport. In *Conference on Robot Learning*, pages 32–43. PMLR, 2023.

[13] J. Clark and D. Amodei. Faulty reward functions in the wild, 2016. URL https://openai.com/index/faulty-reward-functions/.

[14] D. Amodei, C. Olah, J. Steinhardt, P. Christiano, J. Schulman, and D. Mané. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*, 2016.

[15] Y. Yuan, Z. L. Yu, Z. Gu, X. Deng, and Y. Li. A novel multi-step reinforcement learning method for solving reward hacking. *Applied Intelligence*, 49:2874–2888, 2019.

[16] M. Cuturi and M. Blondel. Soft-dtw: a differentiable loss function for time-series. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 894–903. JMLR.org, 2017.

[17] A. Mensch and M. Blondel. Differentiable dynamic programming for structured prediction and attention. In J. Dy and A. Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 3462–3471. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/mensch18a.html.

[18] E. Todorov, T. Erez, and Y. Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033, 2012. doi:10.1109/IROS.2012.6386109.

[19] Humanoid-v4. URL https://gymnasium.farama.org/environments/mujoco/humanoid/.

[20] J. Rocamonde, V. Montesinos, E. Nava, E. Perez, and D. Lindner. Vision-language models are zero-shot reward models for reinforcement learning. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=N0I2RtD8je.

[21] Y. Fu, H. Zhang, D. Wu, W. Xu, and B. Boulet. Furl: Visual-language models as fuzzy rewards for reinforcement learning, 2024. URL https://arxiv.org/abs/2406.00645.

[22] J. Frey, M. Mattamala, N. Chebrolu, C. Cadena, M. Fallon, and M. Hutter. Fast traversability estimation for wild visual navigation, 2023. URL https://arxiv.org/abs/2305.08510.

[23] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms, 2017. URL https://arxiv.org/abs/1707.06347.

[24] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor, 2018. URL https://arxiv.org/abs/1801.01290.

[25] T. Tian, C. Xu, M. Tomizuka, J. Malik, and A. Bajcsy. What matters to you? towards visual representation alignment for robot learning. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=CTlUHIKF71.

[26] S. Sontakke, J. Zhang, S. Arnold, K. Pertsch, E. Bıyık, D. Sadigh, C. Finn, and L. Itti. Roboclip: One demonstration is enough to learn robot policies. *Advances in Neural Information Processing Systems*, 36, 2024.

[27] S. Xie, C. Sun, J. Huang, Z. Tu, and K. Murphy. Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification. In *Proceedings of the European conference on computer vision (ECCV)*, pages 305–321, 2018.

[28] M. Bobrin, N. Buzun, D. Krylov, and D. V. Dylov. Align your intents: Offline imitation learning via optimal transport, 2024. URL https://arxiv.org/abs/2402.13037.

[29] I. Guzey, Y. Dai, B. Evans, S. Chintala, and L. Pinto. See to touch: Learning tactile dexterity through visual incentives. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13825–13832. IEEE, 2024.

[30] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition, 2015. URL https://arxiv.org/abs/1512.03385.

[31] L. Gong, B. Chen, W. Xu, C. Liu, X. Li, Z. Zhao, and L. Zhao. Motion similarity evaluation between human and a tri-co robot during real-time imitation with a trajectory dynamic time warping model. *Sensors*, 22(5):1968, 2022.

[32] N. Taghavi, J. Berdichevsky, N. Balakrishnan, K. C. Welch, S. K. Das, and D. O. Popa. Online dynamic time warping algorithm for human-robot imitation. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3843–3849. IEEE, 2021.

[33] Anonymous. Action-constrained imitation learning. In *Submitted to The Thirteenth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=UlAkM88Vum. under review.

[34] P. Mahmoudieh, D. Pathak, and T. Darrell. Zero-shot reward specification via grounded natural language. In *International Conference on Machine Learning*, pages 14743–14752. PMLR, 2022.

[35] K. Baumli, S. Baveja, F. Behbahani, H. Chan, G. Comanici, S. Flennerhag, M. Gazeau, K. Holsheimer, D. Horgan, M. Laskin, et al. Vision-language models as a source of rewards. *arXiv preprint arXiv:2312.09187*, 2023.

[36] E. S. Lubana, J. Brehmer, P. De Haan, and T. Cohen. Fomo rewards: Can we cast foundation models as reward functions? *arXiv preprint arXiv:2312.03881*, 2023.

[37] M. Klissarov, P. D'Oro, S. Sodhani, R. Raileanu, P.-L. Bacon, P. Vincent, A. Zhang, and M. Henaff. Motif: Intrinsic motivation from artificial intelligence feedback. *arXiv preprint arXiv:2310.00166*, 2023.

[38] Y. Wang, Z. Sun, J. Zhang, Z. Xian, E. Biyik, D. Held, and Z. Erickson. Rl-vlm-f: Reinforcement learning from vision language foundation model feedback. *arXiv preprint arXiv:2402.03681*, 2024.

[39] S. Chen and Y. Wang. Neural approximation of wasserstein distance via a universal architecture for symmetric and factorwise group invariant functions, 2023. URL https://arxiv.org/abs/2308.00273.

[40] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[41] J. Andrews, E. Morton, and L. Griffin. Detecting anomalous data using auto-encoders. *International Journal of Machine Learning and Computing*, 6:21, 01 2016.

Part I

Appendix

Table of Contents

A Sequence Matching Reward Function

A.1 Proof: Limitation of Optimal Transport (OT)

Detailed Problem Setup. Let ξ^{L*} and ξ^{L} be trajectories with the same embedded occupancy distribution. One such example is if ξ and ξ^{L} have the same length and reach the same states, such as the left of Fig. 9. Assume that ξ^{L*} reaches the demonstration states in the correct order, and ξ^{L} reaches them in the incorrect order.

Proposition 2. *The cumulative OT reward is the same for ξ^{L*} and ξ^{L}.*

Proof. Chen and Wang [39] show that the Wasserstein distance is a function of the embedded occupancy distribution of its point sets, so it is invariant to permutations in the ordering of these points. Thus, for the trajectories ξ^{L*} and ξ^{L}, the Wasserstein distance is the same. Consequently, the cumulative OT rewards are also the same, even though ξ^{L} is not a successful rollout. □

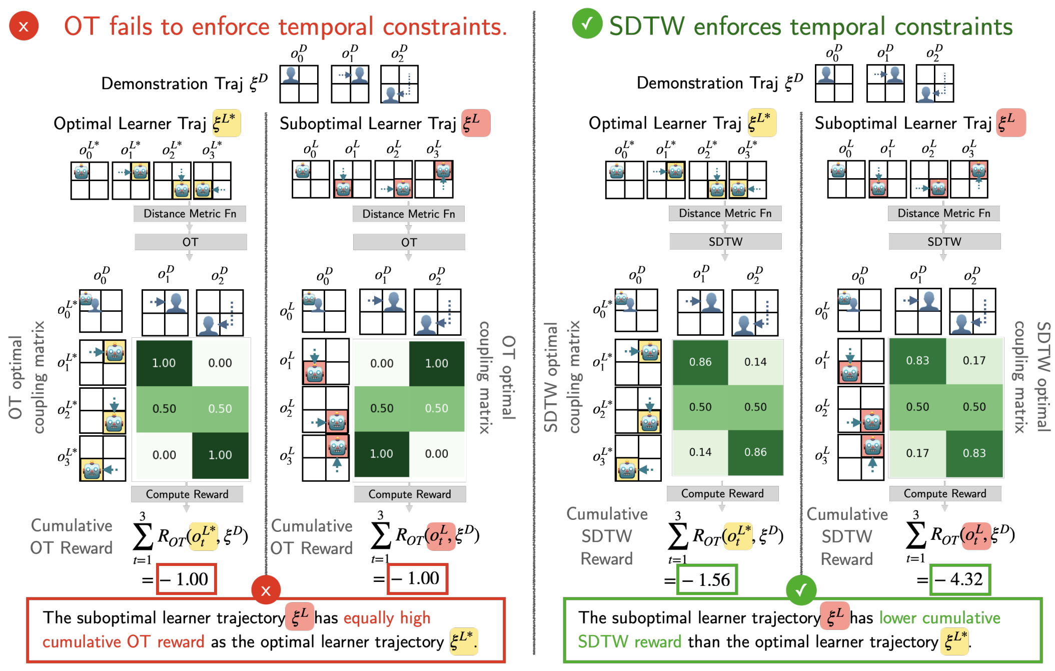
Note that SDTW can overcome this limitation because of its time consistency constraints, as shown in the right figure of Fig. 9. It aligns the two trajectories in a time-consistent way, ensuring that ξ^{L*} (the trajectory in the correct order) is rewarded more than ξ^{L} (the trajectory in the incorrect order), thereby overcoming the failure case of OT.

A.2 Proof: Limitation of Dynamic Time Warping (SDTW with $\lambda = 0$)

Detailed Problem Setup. Assume $\mathcal{S}$ is a finite discrete state space: $S \subset \mathbb{Z}^N$ and $\mathcal{S}$ is bounded in all dimensions. Let (S, d) be a metric space, where d is the Manhattan distance. Assume $\mathcal{A}$ is an action space that allows the agent to move 1 space in any cardinal direction. 2D-Navigation is an example of such an environment.

Let the learner trajectory be $\xi^{L} = \{s_t^L\}_{t=0}^{T-1}$ of length T and $\xi^{D} = \{s_t^D\}_{t'=0}^{T'-1}$ of length T'. Let the sequence matching cost function be SDTW with temperature 0 (this is equivalent to DTW):

$$c_{\text{dtw}}(\xi^L, \xi^D) = \frac{1}{T'T}||\mathbf{A_{dtw}}^*(\xi^L, \xi^D) \odot \mathbf{D}(\xi^{\mathbf{L}}, \xi^{\mathbf{D}})||$$



OT fails to enforce temporal constraints.
SDTW enforces temporal constraints
Demonstration Traj ξ^D
Optimal Learner Traj ξ^{L*}
Suboptimal Learner Traj ξ^L
Distance Metric Fn
OT
SDTW
OT optimal coupling matrix
SDTW optimal coupling matrix
Compute Reward
Cumulative OT Reward
Cumulative SDTW Reward
$\sum_{t=1}^{3} R_{OT}(o_t^{L*}, \xi^D)$
$\sum_{t=1}^{3} R_{OT}(o_t^{L}, \xi^D)$
= – 1.00
= – 1.00
= – 1.56
= – 4.32
The suboptimal learner trajectory ξ^L has equally high cumulative OT reward as the optimal learner trajectory ξ^{L*}.
The suboptimal learner trajectory ξ^L has lower cumulative SDTW reward than the optimal learner trajectory ξ^{L*}.

Figure 9: **Failure cases for OT reward in the 2D-Navigation environment. SDTW overcomes OT's limitation.** The suboptimal learner trajectory ξ^L moves in the counter-clockwise direction while optimal one ξ^{L*} moves clockwise. Unlike the learner, the agent in the demonstration trajectory can move multiple cells per timestep.

where $\mathbf{A_{dtw}}^*(\xi^L, \xi^D)$ is the optimal assignment matrix between ξ^L and ξ^D, and $\mathbf{D}(\xi^L, \xi^D)$ is the distance matrix between ξ and ξ^D such that $\mathbf{D_{t,t'}} = d(\xi^L_t, \xi^D_{t'})$.

Let $\Xi^*(\xi^D)$ denote the set of all learner trajectories with minimum total sequence matching cost given ξ^D: $\Xi^*(\xi^D) = \{\arg\min_{\xi^L} c_{\text{dtw}}(\xi^L, \xi^D)\}$. Finally, we assume that there exists $\xi^{L*} \in \Xi^*(\xi^D)$ such that ξ^{L*} visits every state in ξ^D (i.e., $\forall s^D_{t'} \in \xi^D \; \exists s^L_t \in \xi^{L*}$ s.t. $d(s^D_{t'}, s^L_t) = 0$).

Proposition 3. *There exists $\xi^{L-} \in \Xi^*(\xi^D)$ such that ξ^{L-} does not reach every state in ξ^D.*

Proof. An adversarial learner trajectory ξ^{L-} is constructed by visiting all the same states as ξ^{L*} except that it gets stuck in the second-to-last state of the demonstration trajectory, $s^D_{T'-2}$, and it moves one state towards the the final demonstration state $s^D_{T'-1}$ in its last timestep. Formally, partition the optimal learner trajectory ξ^{L*} into $\xi^{L*}_{\leq}$ and $\xi^{L*}_{>}$, where $\xi^{L*}_{\leq} = \{s^{L*}_0, \ldots, s^{L*}_i\}$ s.t. $d(s^{L*}_i, s^D_{T'-2}) = 0$, and $\xi^{L*}_{>} = \{s^{L*}_{i+1}, \ldots, s^{L*}_{T-1}\}$ is the set of all states afterwards. Then, the adversarial trajectory is $\xi^{L-} = \xi^{L*}_{\leq} + \{s^D_{T'-2}\}_{i=1}^{|\xi^{L*}_{>}|-1} + \{s^{L-}_{T-1}\}$, where s^{L-}_{T-1} is the state such that $d(s^{L-}_{T-1}, s^D_{T'-1}) = d(s^{L-}_{T-2}, s^D_{T'-1}) - 1$.

It follows from the dynamic programming formulation of DTW that $c_{dtw}(\xi^{L*}_{\leq}, \xi^D)$ is only a function of $\xi^{L*}_{\leq}$ and ξ^D, and not any states that occur after $\xi^{L*}_{\leq}$. In other words, the DTW assignment matrix is equal for ξ^{L-} and ξ^{L*} up to and including timestep i.

Let $d_{rem} = d(s^D_{T'-2}, s^D_{T'-1})$. Then, a temporally consistent assignment matrix $\mathbf{A}^-$ for the remaining entries in ξ^{L-} can be constructed by matching s^{L-}_{T-1} to $s^D_{T'-1}$ and the learner states after i and before the last timestep to the second-to-last demonstration state $s^D_{T'-2}$. Because $d(\xi^D_{T'-1}, \xi^D_{T'-1}) = 0$ and $d(s^{L-}_{T-1}, s^D_{T'-1}) = d(s^D_{T'-2}, s^D_{T'-1}) - 1 = d_{rem} - 1$, the assignment and distance matrices are as follows:

$$\mathbf{A}^{-}(\xi^{\mathbf{L}-},\xi^{D}) = \begin{bmatrix} \mathbf{A_{dtw}}(\xi^{\mathbf{L}*}_{\leq},\xi^{\mathbf{D}}) & 0 & 0 \\ 0 & \mathbf{1} & 0 \\ \vdots & \vdots & \vdots \\ 0 & \mathbf{1} & 0 \\ 0 & 0 & \mathbf{1} \end{bmatrix} \qquad \mathbf{D}(\xi^{\mathbf{L}-},\xi^{D}) = \begin{bmatrix} \mathbf{D}(\xi^{\mathbf{L}*}_{\leq},\xi^{\mathbf{D}}) & 0 & 0 \\ 0 & \mathbf{0} & 0 \\ \vdots & \vdots & \vdots \\ 0 & \mathbf{0} & 0 \\ 0 & 0 & d_{rem}-1 \end{bmatrix}$$

This results in the following upper bound on the cost of the adversarial trajectory:

$$c_{dtw}(\xi^{\mathbf{L}-},\xi^{D}) \leq c_{dtw}(\xi^{L*}_{\leq},\xi^{D}) + d_{rem} - 1 \tag{6}$$

To establish a lower bound on the optimal trajectory, observe that d_{rem} moves are necessary to reach $\xi^{D}_{T'}$ from $\xi^{D}_{T'-1}$. Each of these moves incurs at least 1 additional cost for ξ^{L*} except for the last move, which incurs 0 cost, resulting in:

$$c_{dtw}(\xi^{L*},\xi^{D}) \geq c_{dtw}(\xi^{L*}_{\leq},\xi^{D}) + d_{rem} - 1 \tag{7}$$

It follows from equations 6 and 7 that $c(\xi^{L-},\xi^{D}) \leq c_{dtw}(\xi^{L*},\xi^{D})$. Thus, $\xi^{L-} \in \Xi^{*}(\xi^{D})$, despite ξ^{L-} failing to reach the final state $\xi^{D}_{T'}$. □

A.3 Proof: the Effect of Reward Bonus in `SDTW+`

We formalize how `SDTW+` (Section 3.1), which accumulates the SDTW reward from the previous timestep when the learner makes progress, mitigates this failure mode of SDTW.

Detailed Problem Setup. Let $\xi^{L-} = \{o^{L}_{0},\dots,o^{L}_{t-1},o^{L-}_{t},\dots\}$ and $\xi^{L+} = \{o^{L}_{0},\dots,o^{L}_{t-1},o^{L+}_{t},\dots\}$ be two learner trajectories where observations from 0 to $t-1$ are identical but differ at o^{L-}_{t} and o^{L+}_{t}. Let A^{*-} and A^{*+} be the optimal alignment matrix for ξ^{L-} and ξ^{L+} respectively. Let the learner frame o^{L}_{t-1} be primarily matched to $o^{D}_{t'-1}$ (i.e., $t'-1 = \arg\max_j A^{*-}_{t-1,j} = \arg\max_j A^{*+}_{t-1,j}$).

We define ξ^{L-} as a suboptimal trajectory, where, at timestep t, the agent stays close to the previous timestep's frame o^{L}_{t-1} instead of making progress. Thus,

$$\underbrace{t'-1 = \arg\max_{j} A^{*-}_{t,j}}_{o^{L-}_{t+1}\text{primarily matches to the same demonstration frame } o^{D}_{t'-1}\text{ as the previous timestep}} \tag{8}$$

$$\mathcal{R}_{SDTW}(o^{L-}_{t},\xi^{D}) \leq \mathcal{R}_{SDTW}(o^{L}_{t-1},\xi^{D}) \tag{9}$$

In contrast, we define ξ^{L+} to be a more optimal trajectory, where at timestep t, it makes progress as its frame o^{L+}_{t} matches to the next demonstration frame compared to the previous timestep. Thus,

$$\underbrace{t' = \arg\max_{j} A^{*+}_{t,j}}_{o^{L+}_{t+1}\text{primarily matches to the next demonstration frame } o^{D}_{t'}} \tag{10}$$

Under the visual distance metric, when $d_v(o^{L-}_{t},o^{D}_{t'-1}) < d_v(o^{L+}_{t},o^{D}_{t'})$, the suboptimal frame o^{L-}_{t} has higher SDTW reward than the optimal frame because o^{L-}_{t} is primarily matched to $o^{D}_{t'-1}$, o^{L+}_{t} is primarily matched to $o^{D}_{t'}$, and lower distance implies higher reward.

We restate Proposition 1 below:

Proposition 1. *Under assumptions (8) and (9) about o^{L-}_{t} and (10) about o^{L+}_{t}, the `SDTW+`'s cumulative reward bonus guarantees that the more optimal frame o^{L+}_{t} has a higher `SDTW+` reward than the suboptimal frame o^{L-}_{t}, i.e., $\mathcal{R}_{SDTW+}(o^{L+}_{t},\xi^{D}) > \mathcal{R}_{SDTW+}(o^{L-}_{t},\xi^{D})$.*

Proof. By (10) and Algorithm 1, the more optimal learner trajectory ξ^{L+} accumulates an additional reward bonus of $\mathcal{R}_{SDTW}(o^{L}_{t-1},\xi^{D})$ at timestep t because it makes progress by matching o^{L+}_{t} to the next demonstration frame $o^D_{t'}$. Then, the difference between the SDTW+ reward of the more optimal frame $\mathcal{R}_{SDTW+}(o^{L+}_t, \xi^D)$ and the less optimal frame $\mathcal{R}_{SDTW+}(o^{L-}_t, \xi^D)$ is

$$= (\mathcal{R}_{SDTW}(o^{L+}_t, \xi^D) + \mathcal{R}_{SDTW}(o^{L}_{t-1}, \xi^D)) - \mathcal{R}_{SDTW}(o^{L-}_t, \xi^D) > 0 \tag{11}$$

where the equality follows by (8) and Algorithm 1 that the less optimal learner trajectory ξ^{L-} does not accumulate new reward bonus at timestep t; the inequality follows by (9) and positive rewards $\mathcal{R}_{SDTW}(o^{L+}_t, \xi^D) > 0$. Thus, the more optimal frame o^{L-}_t has a higher SDTW+ reward than the less optimal frame $\mathcal{R}_{SDTW+}(o^{L+}_t, \xi^D) > \mathcal{R}_{SDTW+}(o^{L-}_t, \xi^D)$ □

B Visual Distance Metric

We hypothesize that the pretrained encoders fail to pick up fine grained differences between simulated images because the environment is out of their training distribution. Thus, we choose to train a model that can predict the ground truth states (which corresponds to joint positions in the Mujoco environment).

B.1 Dataset Details.

To train the model for visual rewards, we collected a new dataset of MuJoCo images paired with corresponding joint states for each image. The dataset includes in total 9,038 samples.

To build the dataset, we utilized a set of rollout trajectories covering a set of goal reaching tasks (such as different hand poses, doing splits, etc.), We included both successful and unsuccessful trajectories. To ensure diversity among samples representing different stages of a trajectory, we selected one frame every k frames (here $k = 5$), encouraging the network to differentiate between similar images. Given the similarity of initial trajectories, we retained the first four frames only 25% of the time, and in those cases, selected a random frame from a five-frame interval.

B.2 Joint Predictor Training Details.

To train our joint predictor $f \circ \phi$, we fully fine-tune a ResNet50 backbone [30] pre-trained on ImageNet-1K [40] with a 3-layer MLP head that projects to the joint dimension. The MLP head has layers of shape (2048, 1024), (1024, 1024), and (1024, 54), where 54 represents the number of joints (18) multiplied by the dimension per joint (3). Optimization is performed over 100 epochs using SGD with learning rate .008, batch size 16, and momentum 0.875. After training the joint predictor, we freeze the backbone weights, and train a shallow autoencoder architecture with two linear layers of shapes $(d_\phi, 32)$ and $(32, d_\phi)$ using the same parameters, where d_ϕ is the dimension of the backbone (2048 in this case). This provides the reconstruction loss that is used for confidence estimation. For the fine-tuning, we also experimented with Dinov2 on the task of interest and didn't observe, thus we chose to use ResNet.

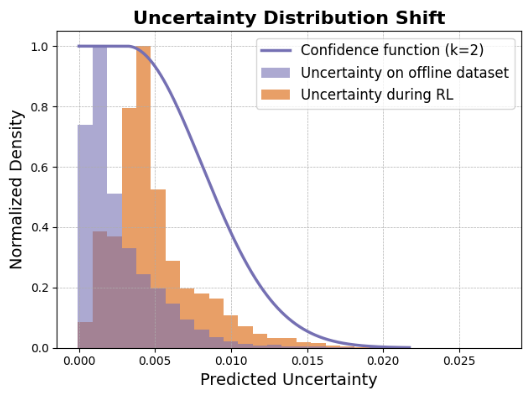


Figure 10: **The distribution of uncertainty predictions on the offline training dataset and over an RL training run for the right arm extend wave task.**

B.3 Epistemic Uncertainty Estimate as Reward Scaling

The use of a joint-position predictor results in an additional challenge: in an environment with unstable dynamics, there is a large space of image observations with very different joint positions, many of which are difficult to reach through robot play. During RL training, a policy can reach a state outside of $\mathcal{D}$, resulting in noisy joint predictions and rewards. To solve this problem, prior work has estimated the epistemic uncertainty of the visual model using an autoencoder architecture [22, 41]. Given that the training converges, if an embedding **z** is in the domain, then the autoencoder will be able to achieve low reconstruction loss. Contrapositively, if it has high reconstruction loss on an embedding **z'**, then **z'** must not be in domain.

We observe that the reconstruction losses over a set of trajectories sampled from partially trained policies are skewed towards high uncertainties. In-domain images tend to have low loss that is tightly clustered around the average offline loss, while out-of-domain images tend to have higher and more spread out loss. Figure 10 shows that uncertainties increase during RL training, indicating that there is distribution shift. We additionally visualize the confidence function $c(\mathbf{z})$, which we define as $1 - (u(\mathbf{z}) + \epsilon)$. The confidence uses the mean and standard deviation of the offline uncertainties and $k = 2$. This demonstrates that the confidence function (and thus uncertainty function) is appropriate for the uncertainty distribution encountered during RL.

Figure 11 shows a qualitative example of how uncertainty scaling fixes incorrect reward predictions on out-of-distribution images. The shape of the uncertainty scaled reward curve better matches the ground truth.

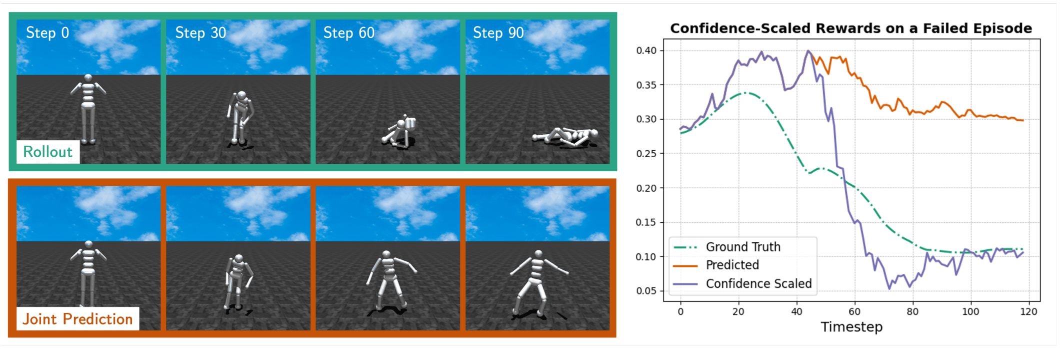


Figure 11: **The impact of confidence scaling on rewards for a trajectory where the robot falls down**. The ground truth trajectory is shown on top and renders of the model's joint predictions given the top frames are shown on the bottom. The rewards are computed with respect to the final reference image of the left arm wave task. Once the robot starts falling, there is a distribution shift in the images, and the model predicts that the robot is upright. This leads to high reward predictions, but also high uncertainty estimates. When corrected using the confidence function, the reward shape improves significantly.

C Experiments

C.1 Environment Details.

- **2D-Navigation.** The discrete environment has a 3×5 grid with the agent starting at $(1, 0)$ and obstacles spanning from $(1, 1)$ to $(1, 3)$. The state space is the current position of the agent. At each timestep, the agent can move to one cell in a direction (up, down, left, or right) or remain in its current cell.
  The agent's goal is to follow the reference sequence within a maximum of 8 timesteps. We design two demonstration trajectories (shown in Fig. 6) to test how well the sequence matching reward function handles a demonstration trajectory with varying pacing (i.e. the demonstrator can move multiple cells per timestep). The first demonstration sequence has 2 timesteps, with the number of cells between each timestep in the demonstration trajectory remaining the same. The second sequence has 3 timesteps, but the number of cells between timesteps varies.

- **Humanoid.** We use the MuJoCo `Humanoid-v4` environment [18, 19]. At the beginning of an episode, the humanoid is spawned upright, slightly above the ground, with its arms curled towards its chest.
  The humanoid's goal is to follow the motion of a demonstration trajectory within a maximum of 120 timesteps. We define 6 motions, corresponding to 6 demonstration trajectories:
  1. left arm up
  2. right arm up
  3. left arm out
  4. right arm out
  5. both arms out
  6. both arms down

  These demonstration trajectories each have a length of 10 and are generated by interpolating between the initial and final poses. Fig. 8 shows snapshots of these trajectories. We assume access to a stability reward function, which includes a control cost and a reward for remaining standing.

C.2 RL Policy Details.

Table 1: Training hyperparameters used for experiments on both environments.

Parameter	**2D Navigation Env. (PPO)**	**Humanoid Env. (SAC)**
Total environment steps	100,000	2,000,000
Learning rate	0.00025	0.001
Batch size	64	256
Gamma (γ)	0.99	0.99
Entropy coefficient	0.25	-
Value function coefficient	1	-
Actor/Critic architecture	-	(256, 256)
Episode length	5	120
Seed	9	9

C.3 Toy Environment Experiments

Figure 12 shows example trajectories for each method on the two 2D-Navigation environment tasks. In both cases, `SDTW+` is the only method that reaches every reference state in the correct order.

C.4 OT Failure in the Humanoid Environment

Figure 13 shows an example of failure case for OT reward in the Mujoco `Humanoid-v4` environment.

C.5 Unbalanced Matching in DTW+

Figure 14 shows an example of trajectory, where the DTW+ coupling matrix assigns most of the learner timesteps to the frames towards the end of the reference trajectory.

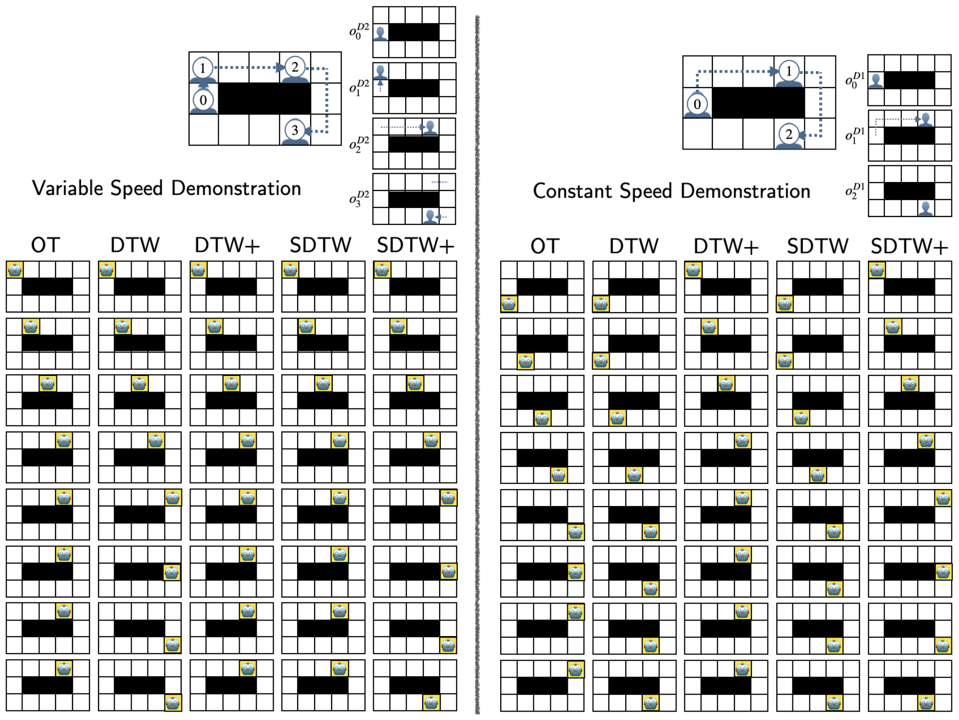


Figure 12: **Final trajectories of PPO-trained agents on the two 2D-Navigation sequence following tasks.**

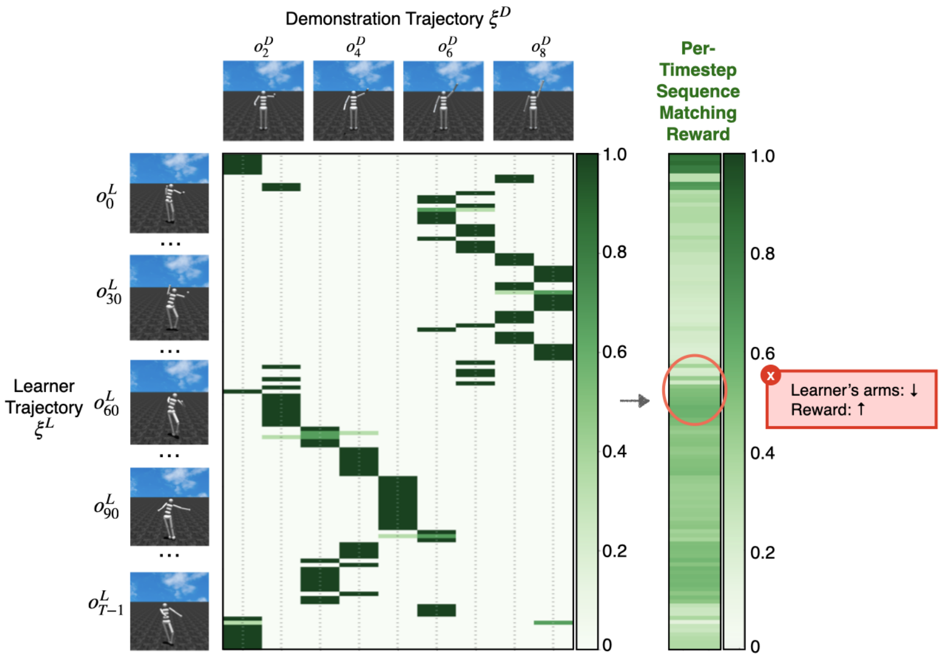


Figure 13: **Failure case for OT reward in the Humanoid environment.** The OT optimal coupling matrix (left matrix) shows that when the learner's arms are moving downward (around timestep 60, denoted by o_{60}^L on the left), the corresponding assignments in the matrix also progress forward along the demonstration trajectory. However, the OT reward increases from approximately 0.2 to 0.6, suggesting a higher reward when the humanoid's movements do not accurately match the intended complex motions.

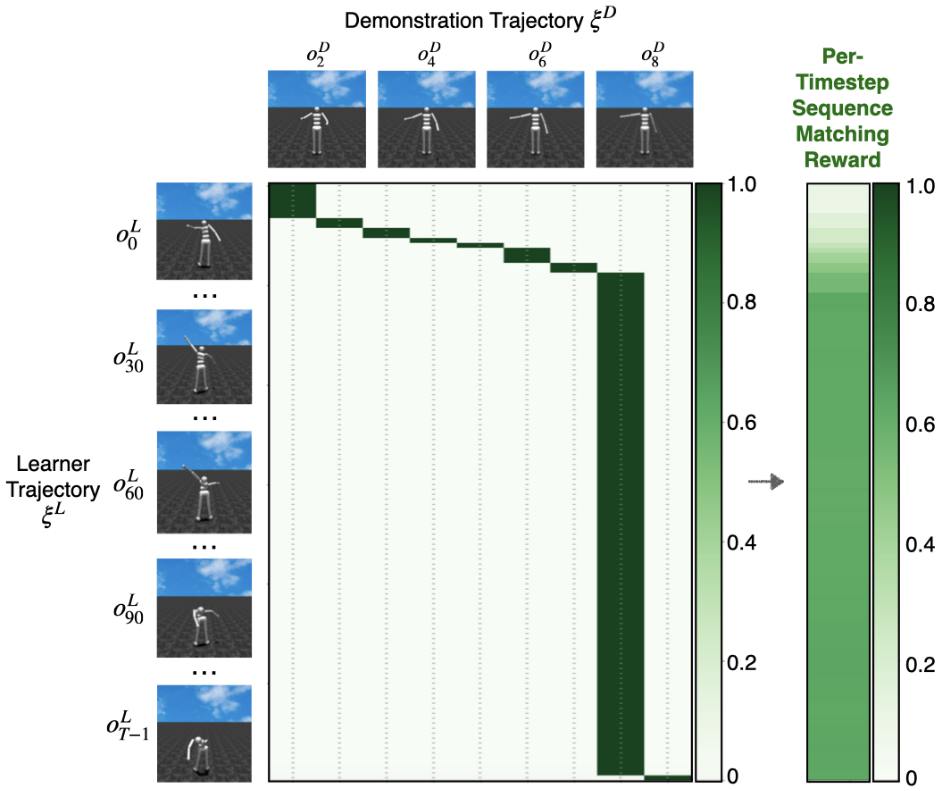


Figure 14: **Example of trajectory with unbalanced matching with DTW+.**