Robot Policy Learning with Temporal Optimal Transport Reward

Yuwei Fu

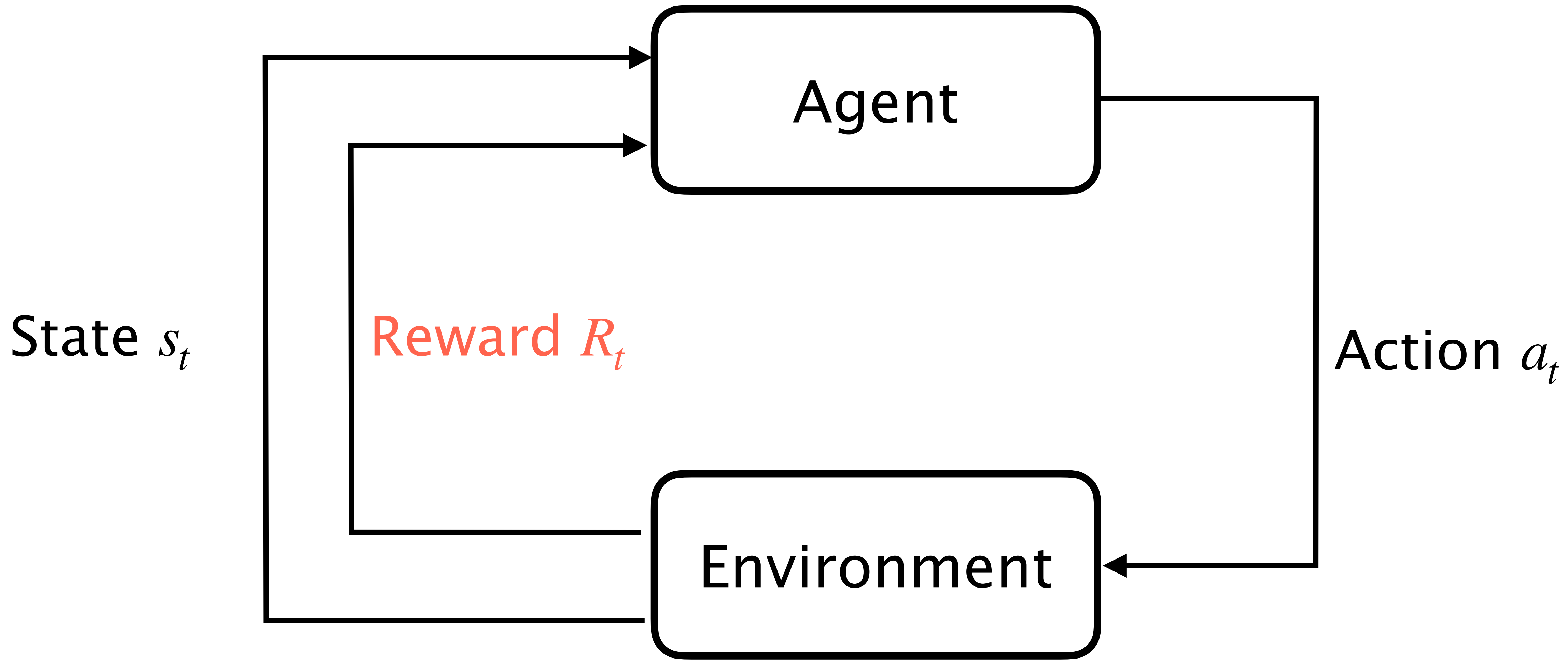
2024-12

NEURAL INFORMATION PROCESSING SYSTEMS

McGill

1. Inverse RL

- Reward specification is tricky in RL.
- Inverse RL first learns an approximate reward function from expert data, and then uses the proxy reward function to learn the agent.
- How to learn effective robot policies with only a few expert demos?



$$\mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty}\gamma^t R(s_t, a_t)\right]$$

$$\mathbb{E}\left[\sum_{t=0}^{\infty}\gamma^t R^*(s_t, a_t)\middle|\pi^E\right] \geq \mathbb{E}\left[\sum_{t=0}^{\infty}\gamma^t R^*(s_t, a_t)\middle|\pi\right], \forall \pi \in \Pi$$

2. OT-based RL

- Applying Optimal Transport (OT) based proxy reward in RL by solving the following optimization problem:

$$\mathcal{W}(p,q) = \inf_{\mu} \int_{\mathcal{X}\times\mathcal{Y}} c(x,y)\,d\mu$$

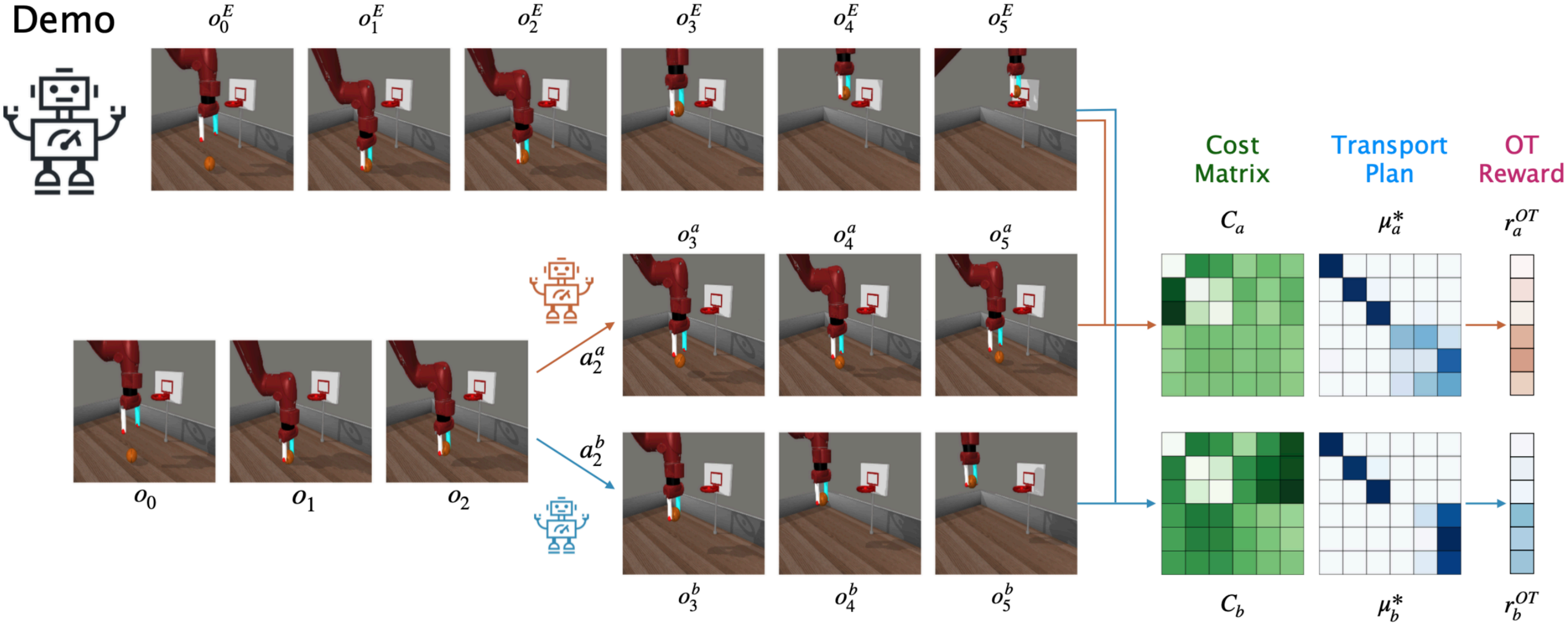
$$\tau^E = \{o_1^E, \cdots, o_T^E\}$$

$$\tau = \{o_1, \cdots, o_T\}$$

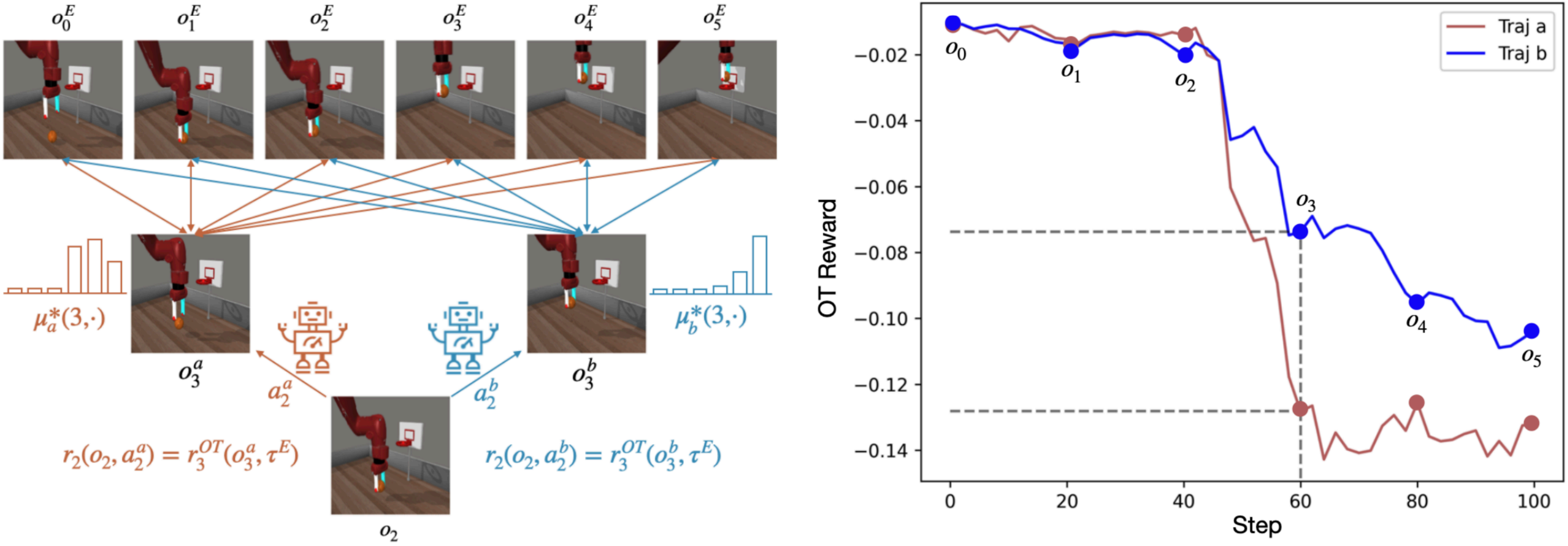
$$\mathcal{W}(\tau,\tau^E) = \min_{\mu\in\mathbb{R}^{T\times T}} \sum_{i=1}^{T}\sum_{j=1}^{T} c(o_i, o_j^E)\mu(i,j),$$

$$\text{s.t.} \sum_{i=1}^{T}\mu(i,j) = \sum_{j=1}^{T}\mu(i,j) = \frac{1}{T}.$$

$$r_i^{OT} = -\sum_{j=1}^{T} c(o_i, o_j^E)\mu^*(i,j)$$



Demo
o_0^E
o_1^E
o_2^E
o_3^E
o_4^E
o_5^E
Cost Matrix
Transport Plan
OT Reward
C_a
μ_a^*
r_a^{OT}
o_3^a
o_4^a
o_5^a
a_2^a
a_2^b
o_0
o_1
o_2
o_3^b
o_4^b
o_5^b
C_b
μ_b^*
r_b^{OT}



o_0^E
o_1^E
o_2^E
o_3^E
o_4^E
o_5^E
$\mu_a^*(3,\cdot)$
$\mu_b^*(3,\cdot)$
o_3^a
o_3^b
a_2^a
a_2^b
o_2
$r_2(o_2, a_2^a) = r_3^{OT}(o_3^a, \tau^E)$
$r_2(o_2, a_2^b) = r_3^{OT}(o_3^b, \tau^E)$
Traj a
Traj b
OT Reward
Step
o_0
o_1
o_2
o_3
o_4
o_5
−0.02
−0.04
−0.06
−0.08
−0.10
−0.12
−0.14
0
20
40
60
80
100

3. Observations

- OT reward is order invariant.
- The order information is discarded and the frames from the demo trajectory are treated as bag-of-temporally-collapsed frames.
- The policy is likely to converge to undesired solutions.

$$(s_1, s_2, s_1)$$

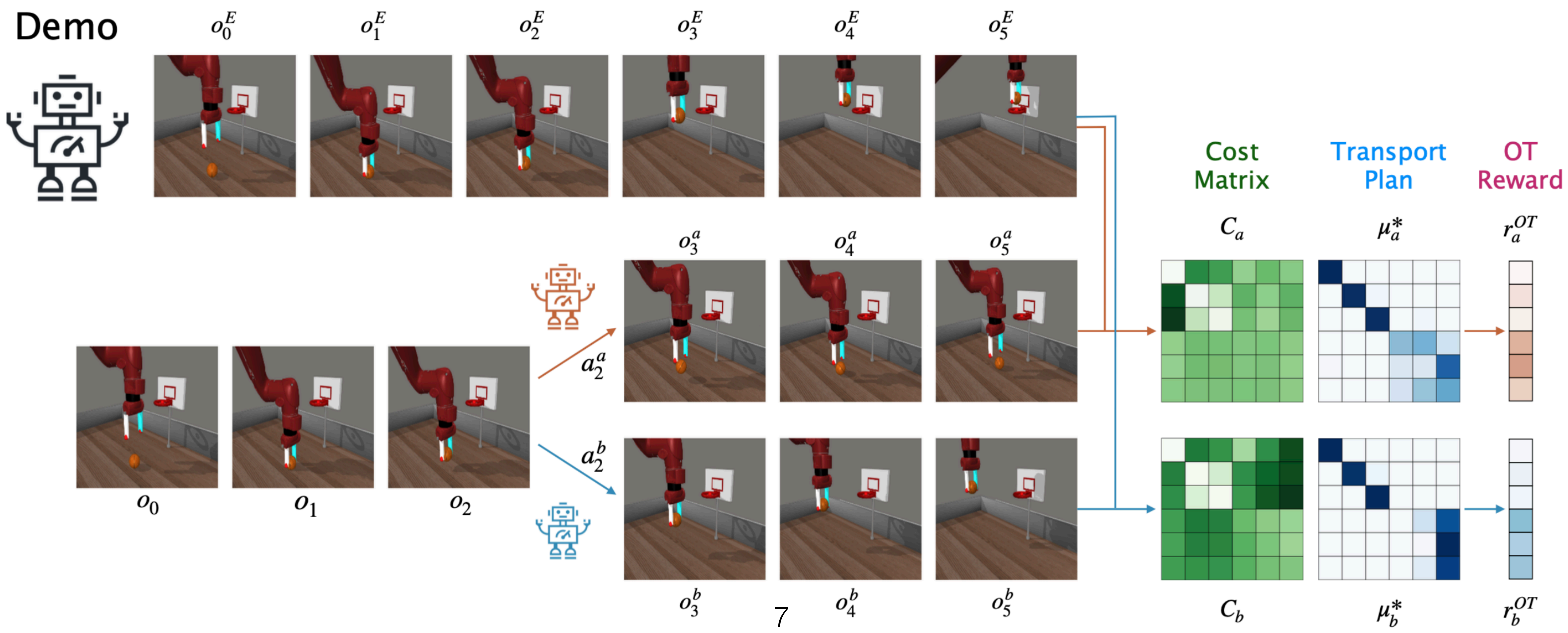
$$(s_1, s_1, s_2)$$

$$r_i^{OT} = -\sum_{j=1}^{T} c(o_i, o_j^E)\mu^*(i, j)$$

3. Observations

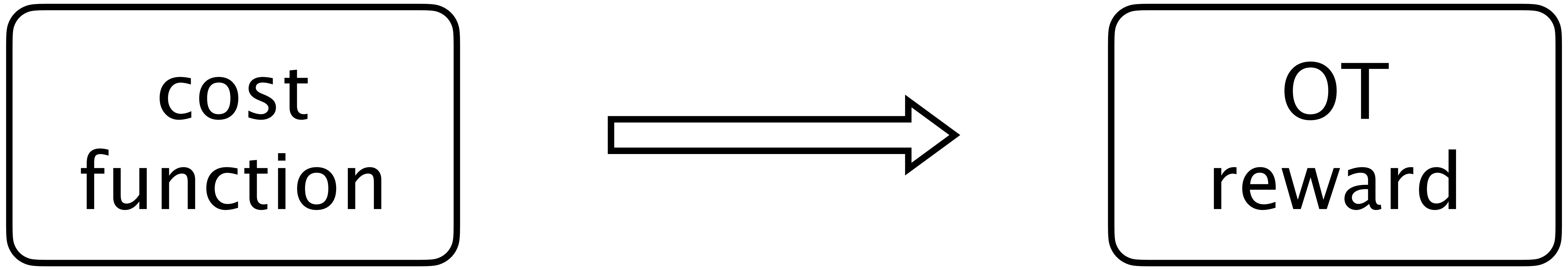
- OT reward is influenced by the later steps so that two transitions with the same state-action pair could have different values.

$$\mathbb{E}_{(s,a,r,s')}\left[\left(r+\gamma\max_{a'}Q^{\pi}_{\hat{\theta}}(s',a')-Q^{\pi}_{\theta}(s,a)\right)^2\right]$$



4. TemporalOT

- Stage 1: defines a transport cost between two states.
- Stage 2: solves the OT optimization problem to approximate the optimal transport plan and computes the OT reward.

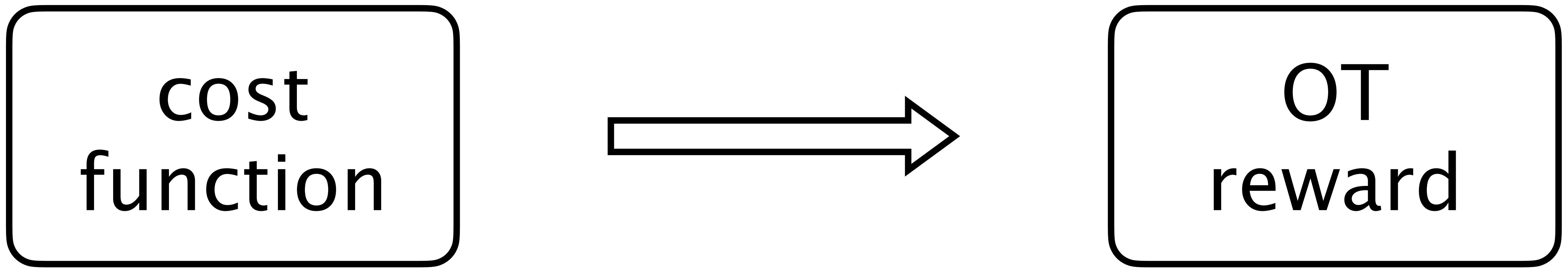


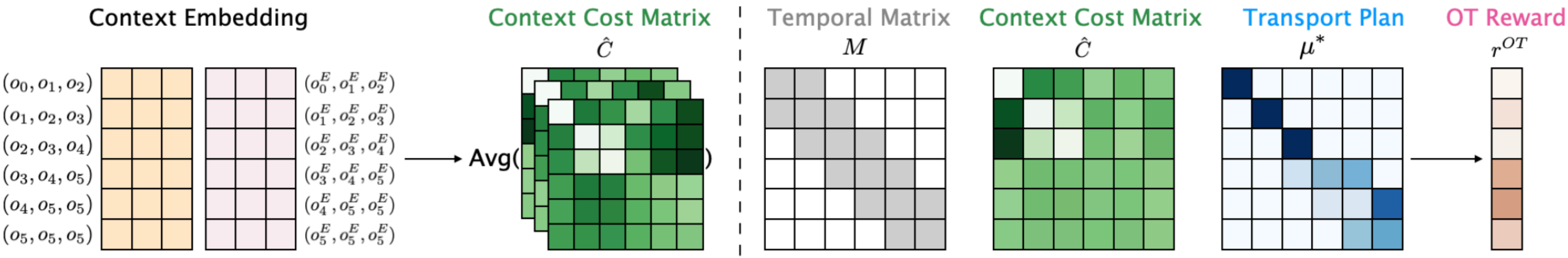
$$c(o_i, o_j^E) = 1 - \frac{\langle f(o_i), f(o_j^E) \rangle}{\|f(o_i)\| \|f(o_j^E)\|}$$

$$r_i^{OT} = -\sum_{j=1}^{T} c(o_i, o_j^E) \mu^*(i, j)$$

4. TemporalOT

- Context embedding-based cost matrix for improving stage 1.
- Temporal-masked OT objective for improving stage 2.





cost function $\Longrightarrow$ OT reward

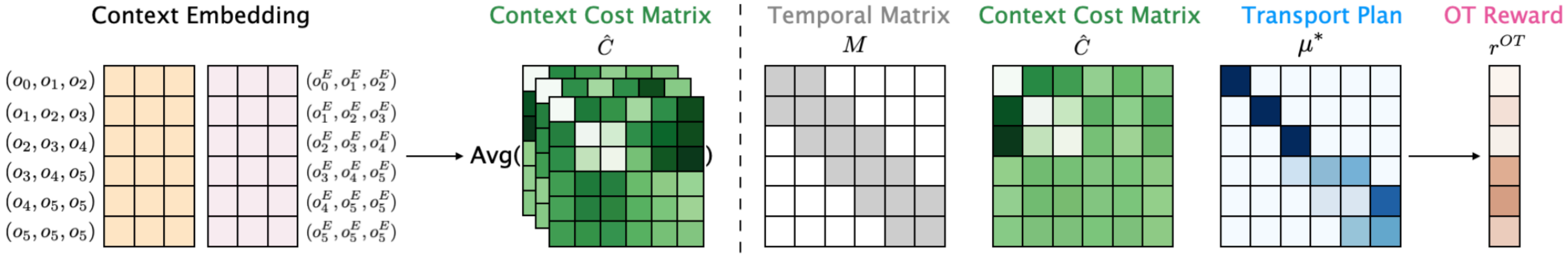
$$c(o_i, o_j^E) = 1 - \frac{\langle f(o_i), f(o_j^E)\rangle}{\|f(o_i)\|\|f(o_j^E)\|}$$

$$\hat{c}(o_i, o_j^E) = \frac{1}{k_c}\sum_{h=0}^{k_c-1}\left(1 - \frac{\langle f(o_{i+h}), f(o_{j+h}^E)\rangle}{\|f(o_{i+h})\|\|f(o_{j+h}^E)\|}\right)$$

$$r_i^{OT} = -\sum_{j=1}^{T} c(o_i, o_j^E)\mu^*(i,j)$$

$$\mu^* = \arg\min_{\mu}\langle M \odot \mu, \hat{C}\rangle_F - \epsilon\mathcal{H}(M \odot \mu)$$

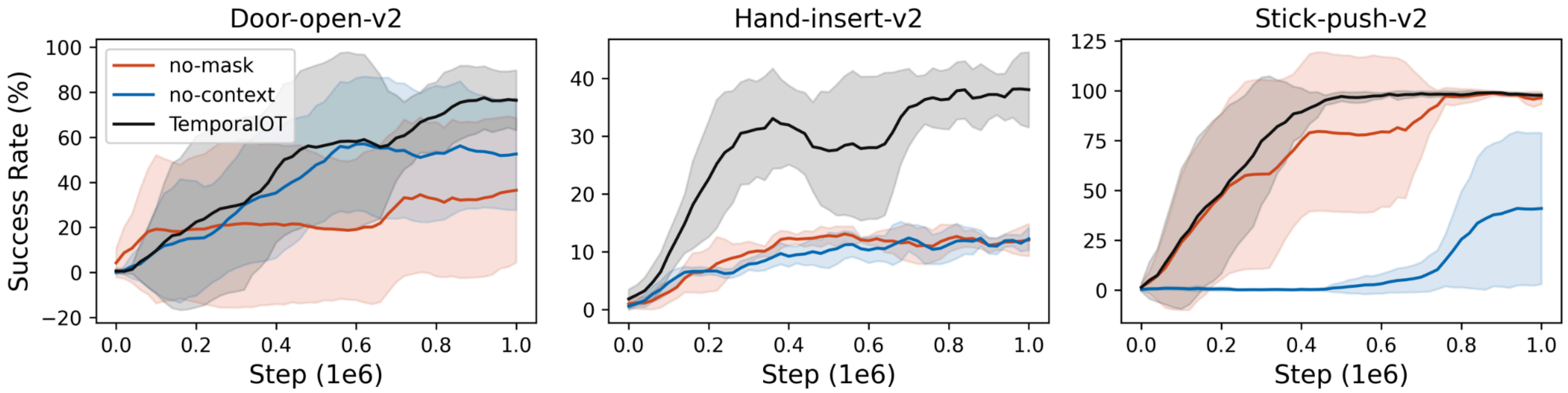
$$M(i,j) = \begin{cases} 1, & \text{if } j \in [i-k_m, i+k_m], \\ 0, & \text{otherwise,} \end{cases}$$



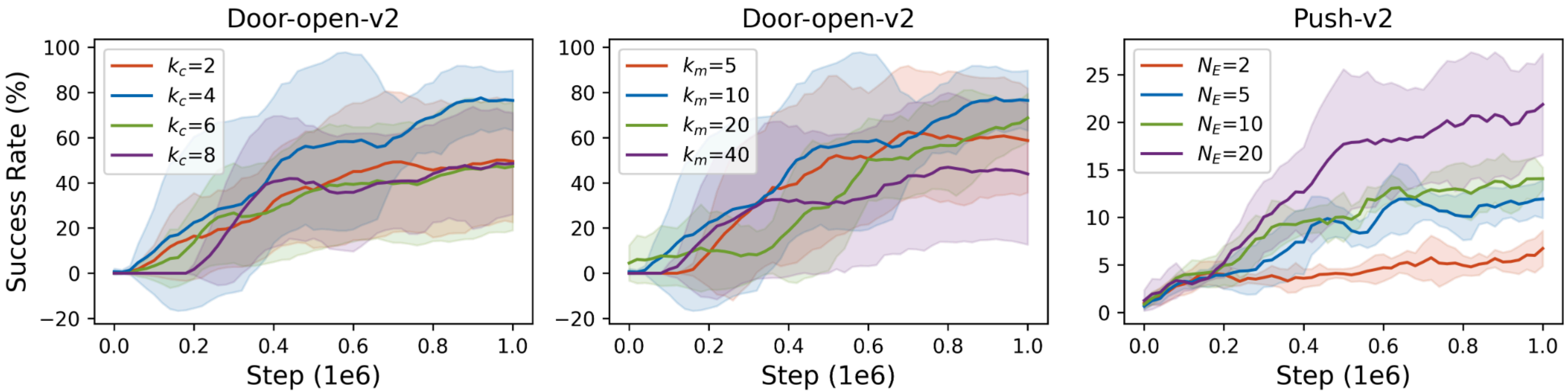
5. Experiments

Environment	TaskReward	BC	GAIfO	OT0.99	OT0.9	ADS	**TemporalOT**
Basketball	0.0 (0.0)	0.2 (0.4)	0.0 (0.0)	0.0 (0.0)	76.6 (27.4)	42.2 (44.5)	**94.4 (4.7)**
Button-press	14.0 (18.5)	1.7 (2.4)	1.0 (1.1)	**88.8 (2.5)**	85.2 (3.3)	**89.0 (3.8)**	**92.4 (3.6)**
Door-lock	86.2 (12.4)	4.6 (7.0)	8.8 (12.2)	3.0 (5.5)	2.8 (2.0)	3.2 (2.7)	**33.4 (2.8)**
Door-open	0.0 (0.0)	10.7 (10.3)	2.2 (1.7)	46.2 (33.6)	30.2 (34.5)	52.0 (42.7)	**78.4 (12.4)**
Hand-insert	0.8 (1.6)	2.3 (2.1)	8.6 (4.4)	29.0 (9.7)	11.2 (2.3)	**35.0 (5.3)**	**36.8 (6.6)**
Lever-pull	0.0 (0.0)	0.8 (1.6)	3.4 (1.9)	15.4 (15.5)	35.6 (12.8)	21.2 (12.0)	**53.6 (7.7)**
Push	1.0 (0.7)	0.4 (0.8)	0.0 (0.0)	**14.2 (7.5)**	7.0 (2.6)	**17.2 (5.6)**	8.4 (1.7)
Stick-push	0.0 (0.0)	0.0 (0.0)	18.8 (22.9)	0.0 (0.0)	48.8 (41.5)	20.0 (40.0)	**97.6 (2.6)**
Window-open	85.6 (12.2)	1.6 (2.7)	4.0 (4.7)	**54.0 (28.0)**	22.4 (22.9)	43.6 (20.5)	**55.2 (2.3)**
Average	20.8	2.5	5.2	27.8	35.5	35.9	**61.1**

5. Experiments



5. Experiments



6. Summary

- TemporalOT improves the two stages in OT-based RL, respectively.
  - Learning a more accurate cost function.
  - Using a temporal mask to incorporate temporal order information.