
Enhancing Tactile-based Reinforcement Learning for Robotic Control – Supplementary Material

Project page with videos: See the provided `index.html` for agent videos. We recommend changing the playback speed to `x0.5` to better see the learned gaits.

A Results as real-world metrics

Figure A1 shows the average number of seconds it takes the agent to locate the object within different tolerances. The distance d is measured between the Franka end-effector (imaginary fixed frame in the center of the parallel-jaw gripper) and the object center. While the performance between a proprioceptive and proprioceptive-tactile agent is similar for a $d = 3$ cm threshold, the benefits of tactile data become more pronounced with smaller tolerances. The relative performance between SSL objectives is consistent except for $d = 0.05$ cm, where the tactile dynamics objective is on average the fastest. Figure A2 shows learning through the number of bounces the agent achieves in 10 seconds, and Figure A3 through the number of complete Baoding rotations achieved. Due to the overlapping performance distributions in *Baoding*, we provide alternative figure versions with only a subset of runs and/or no bad seeds. Across all seeds, from Figure A3 (bottom left) we can see applying tactile reconstruction or full dynamics self-supervision approximately doubles the number of complete rotations achieved.

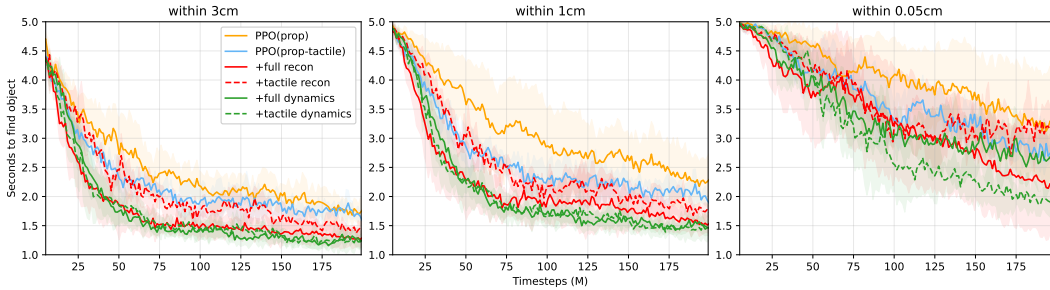


Figure A1: Number of seconds to find the object within 3, 1, and 0.05 cm in the *Find* environment.

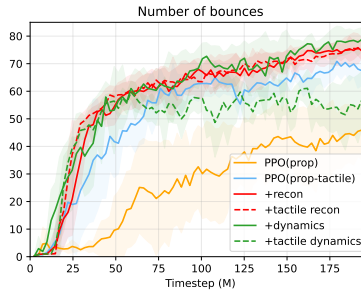


Figure A2: Number of bounces in 10 seconds.

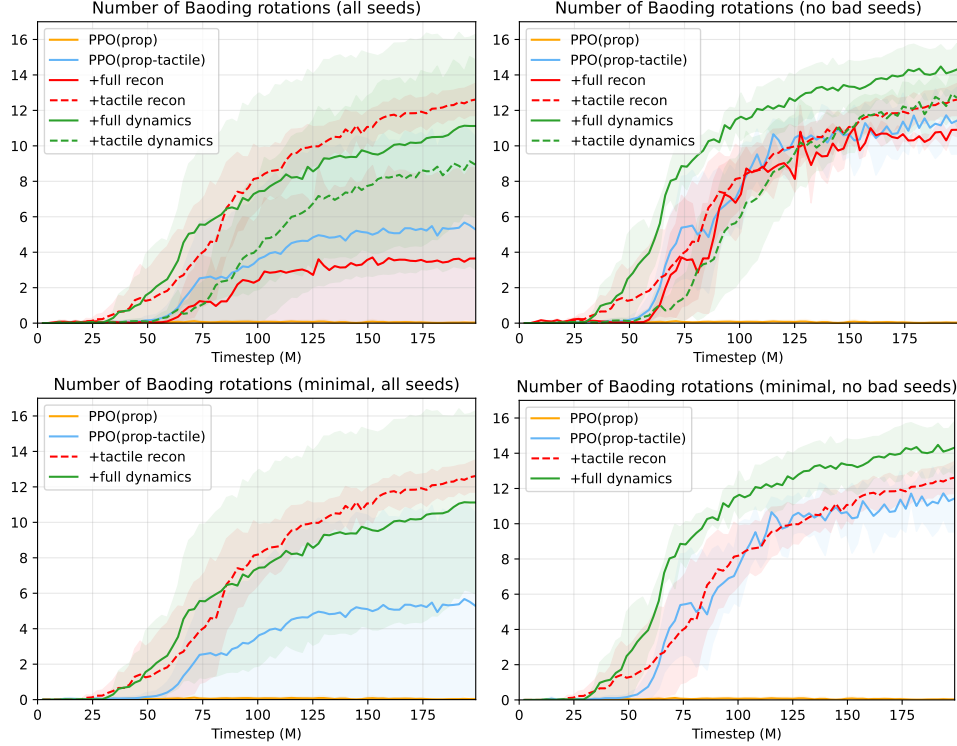
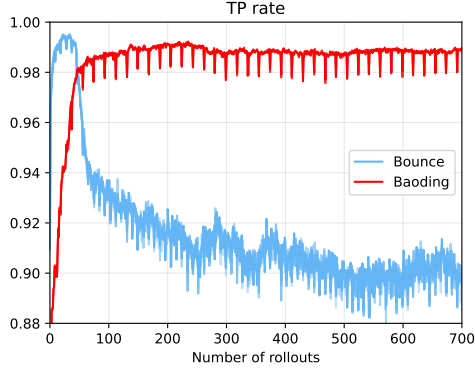


Figure A3: Number of complete Baoding ball rotations i.e., when each ball makes a full rotation in 10 seconds. **Left:** The distribution across all seeds. **Right:** The distribution across seeds that learned at least 1 rotation. **Top:** All experiments. **Bottom:** A subset of experiments.

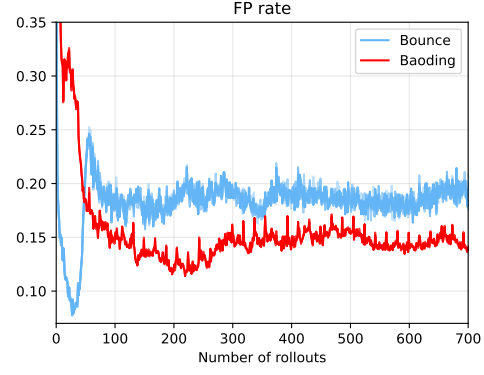
B Future tactile state prediction analysis

To understand how well MLPs can model the forward dynamics of tactile interactions, we provide various classification metrics throughout training (Figure A4) and spatio-temporal rollout visualisations of trained *Bounce* and *Baoding* agents (Figures A5 and A6). We can compute metrics like recall and precision because tactile reconstruction was formulated as classification (rather than regression), but unlike typical supervised learning the performance does not increase monotonically because of the nonstationary data distribution. We perform this analysis on agents trained with the tactile dynamics objective since they were specifically trained for future tactile prediction. This analysis could be done on agents trained with full dynamics, but would require learning a separated tactile decoder. Finally, due to the large proportion of 0s to 1s, we found it necessary to apply a positive weighting in the binary cross entropy loss of $p_c = 10$ across all classes (activation regions) due to tactile data imbalance.

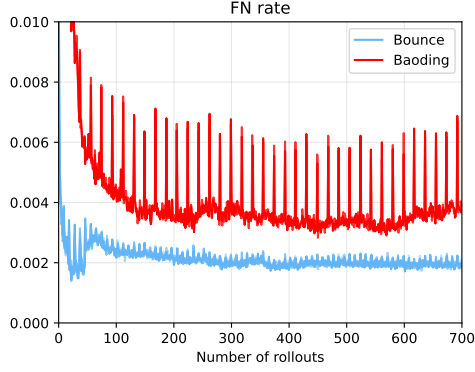
Overall, we were surprised how robust the performance remained n_f timesteps into the future (evidenced by how difficult it is to distinguish between the timesteps in Figure A4). This suggests that the multi-step dynamics objective was very effective at encoding information relevant for future state predictions. Moreover, the rate of missed contacts was $<1\%$ throughout training for both *Bounce* and *Baoding*, which we attribute to the positive weighting applied. Interestingly, despite the agent having access to the 3 last tactile states that would form the first 3 tactile states of the prediction, some of these states were not always perfectly “copied” over (e.g., s_5 , s_{11} , and s_{12} in *Bounce*). This highlights that the states are not merely being ‘memorised’, but being represented in some (imperfect) way. For future work, we believe dynamically updating the positive weighting would be beneficial to reflect the nonstationary training distribution. Additionally, our implementation applied the same weighting to each activation region, but some regions are much more active than others (e.g., compare palm to pinky in Figure A6), and this discrepancy should be accounted for.



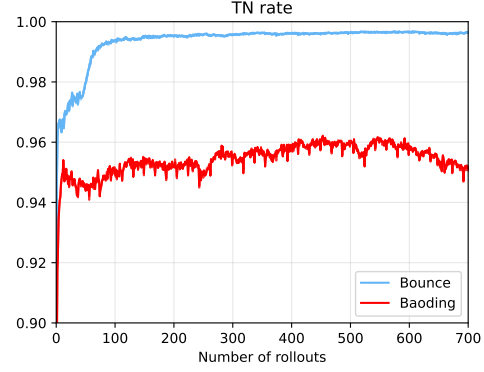
(a) $TPR = \frac{TP}{TP+FN}$ (also known as *recall* or *probability of detection*, proportion of actual contacts that were classified as contacts)



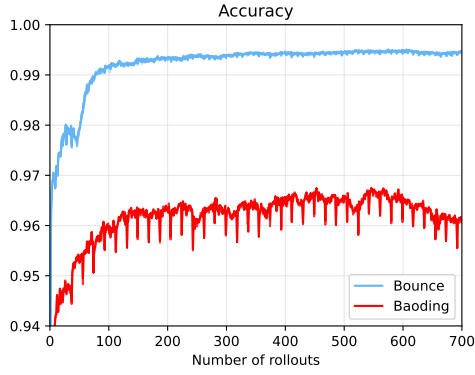
(b) $FPR = \frac{FP}{FP+TN}$ (also known as *probability of false alarm*, proportion of null contacts that were misclassified as contacts)



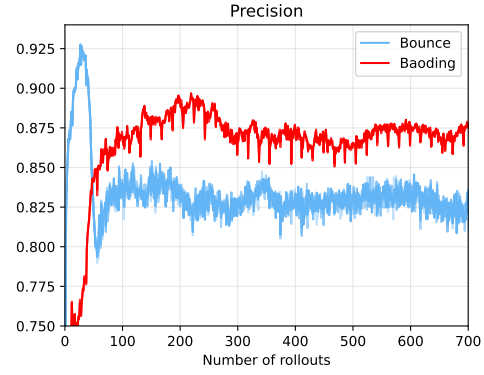
(c) $FNR = \frac{FN}{TP+FN}$ (proportion of actual contacts that were missed)



(d) $TNR = \frac{TN}{FP+TN}$ (proportion of null contacts that were classified as null contacts)



(e) Accuracy $A = \frac{TP+TN}{TP+FP+TN+FN}$



(f) Precision $P = \frac{TP}{TP+FP}$

Figure A4: Classification metrics for future predicted tactile states of *Bounce* and *Baoding* agents trained with tactile dynamics SSL objective for $t = 1, 2, 3, 9$ and $t = 1, 2$ timesteps into the future respectively. The metrics for future timesteps are shown with decreasing opacity but strongly overlap with $t = 1$, making it difficult to distinguish. The metrics were computed on one minibatch after the RL/SSL update.

For *Bounce*, tactile interactions are increasingly sparse (e.g., compare Figures A5 and A6). Thus the metrics we were most interested in were true positive rate (proportion of contacts caught) and false negative rate (proportion of contacts missed). True positive rate (TPR) decreased from $\sim 99\%$ to 90% throughout training, which we attribute to increased difficulty predicting the landing sites of a bouncing ball. The proportion of contacts that were missed (FNR) was surprisingly low throughout training, converging to $\sim 0.2\%$. Fascinatingly, from the no-contact state s_7 , the decoder correctly anticipates contact in the next state (albeit in the wrong locations, however two predictions are just 1 timestep early). This result suggests that some form of object position information is being encoded.

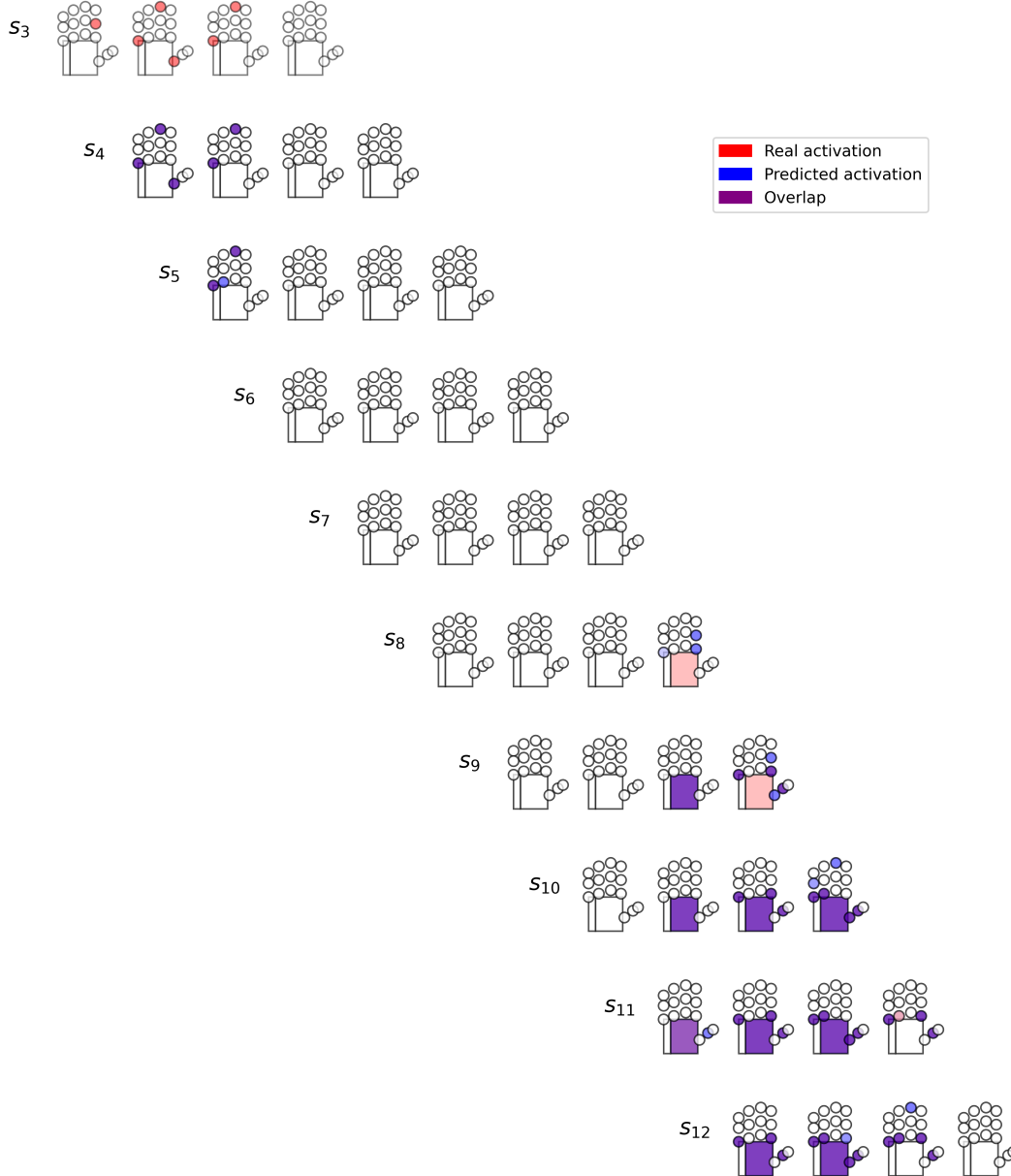


Figure A5: Spatio-temporal visualisation of 1-step predicted future tactile states of a *Bounce* agent trained with the tactile dynamics SSL objective. The state s is comprised of 4 observations, and each column corresponds to the same observation. The sigmoid activation of the tactile prediction is displayed with varying opacity (e.g., less confident is lighter blue).

For *Baoding*, the frequency of tactile interactions remains high throughout learning, thus all metrics are of relevance. The proportion of contacts that were correctly detected was $\sim 99\%$, which is very high. Similarly to *Bounce*, very few true contacts were missed (FNR), and the false positive rate was relatively high at $\sim 15\%$. The accuracy remained at $>96\%$ for majority of training. Also similarly to *Bounce*, some of the false positive predictions (e.g., in states s_{10}, s_{11}) are just one-step too early. Finally, across all metrics in Figure A4, we can see negative performance spikes at fixed intervals throughout learning that are not present for *Bounce*. This is discussed in the next section.

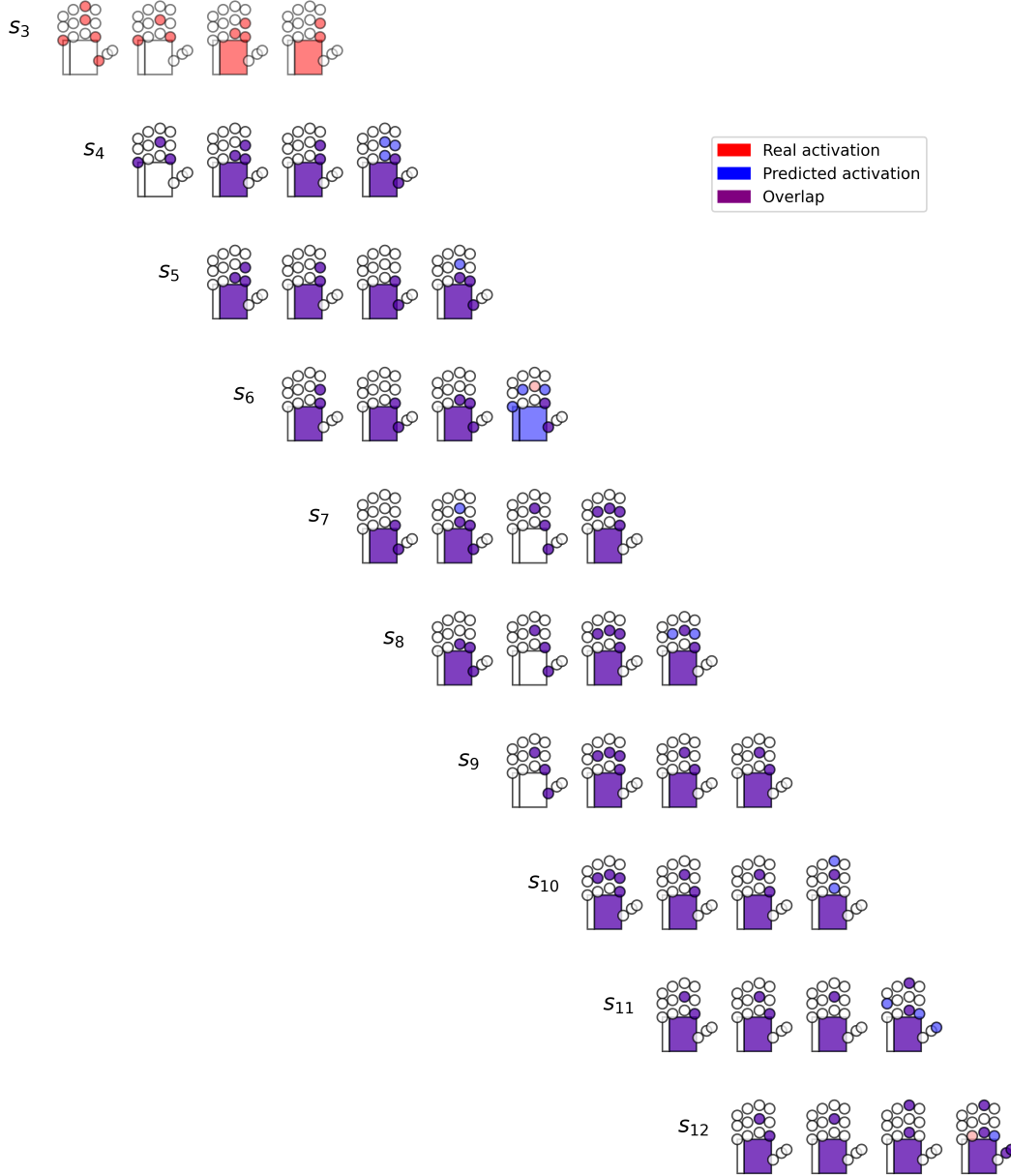


Figure A6: Spatio-temporal visualisation of 1-step predicted future tactile states of a *Baoding* agent trained with the tactile dynamics SSL objective. The state s is comprised of 4 observations, and each column corresponds to the same observation. The sigmoid activation of the tactile prediction is displayed with varying opacity (e.g., less confident is lighter blue).

C Impact of auxiliary memory on self-supervised learning

Figure A7 shows the average minibatch (full) dynamics loss of a *Baoding* agent with varying auxiliary memory size. There are 19 rollouts between each peak, and with a rollout length of 32 steps this corresponds to 608 timesteps \sim 1 episode. We interpret the loss peaks coming from episode resets when the Baoding balls are set above the palm and potentially spend a large portion of one rollout falling into the palm, disrupting SSL as the loss of tactile data is highly out-of-distribution. Increasing the memory size flattens and widens out the peak, potentially leading to smoother gradient updates and the improved agent performance in this task.

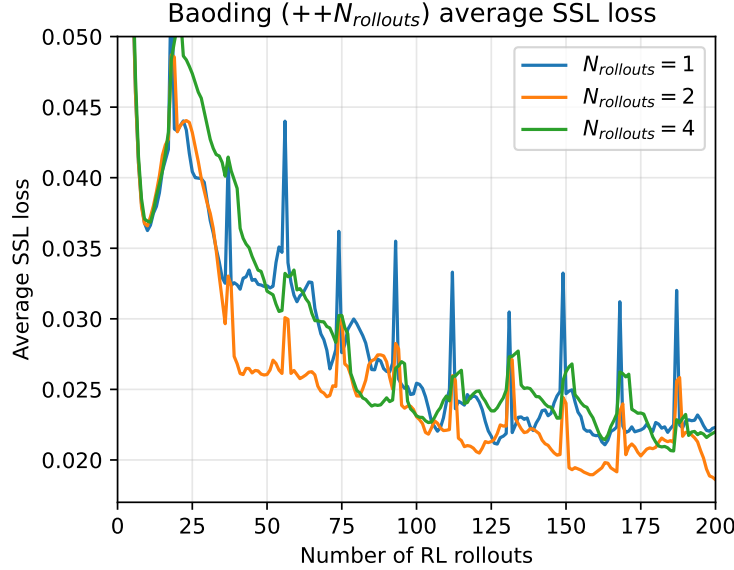


Figure A7: The average minibatch self-supervised dynamics loss of a *Baoding* agent with varying auxiliary memory size.

D MDP

We use a physics simulation frequency of 120 Hz across all environments with a control decimation of 2. This means the agent receives observations and computes actions at 60 Hz, because the same action is applied twice in the physics engine. The default static and dynamic friction for robots and objects was set to 1.0.

D.1 Observations

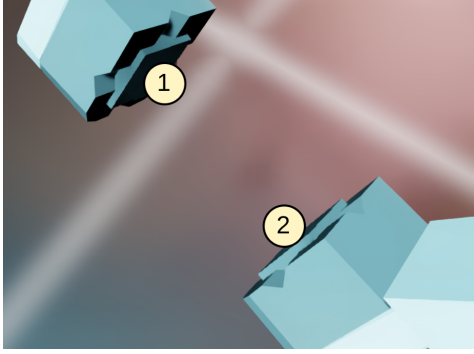
A summary of the different types of observation is shown in Table A1. Each environment uses a stack of observations o_t to form the state s_t (16 for the Franka Find environment, 4 for the Shadow environments). We apply input preprocessing as follows: joint angles θ are normalised between $[-1, 1]$. Joint velocities $\dot{\theta}$ are scaled down by a scalar (0.33 for Franka, 0.2 for Shadow). The norm of each 3-dim normal force vector is clamped between $[0, \text{MAX}]$, and normalised between $[0, 1]$. The value of MAX was chosen to be 20N for Franka and 30N for Shadow.

We retrieve the forces through Isaac Lab’s `ContactSensor` class, which returns the net contact force acting on a given rigid body¹. To mimic real-world tactile sensing and make the task more challenging, we registered two ‘plate-like’ bodies to atop the Franka fingers to act as contact sensors (Figure A8a). With this setup, the sensors would only register forces that resulted from collision with these bodies, which is only possible from the object or other finger, and not the ground. Since the Shadow Hand was fixed in midair we let each link become a sensor, resulting in 17 sensors (Figure A8b).

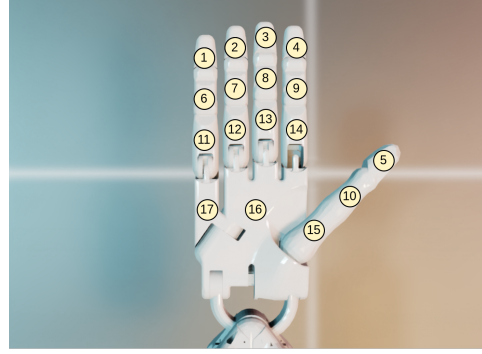
¹https://isaac-sim.github.io/IsaacLab/main/source/overview/core-concepts/sensors/contact_sensor.html

Table A1: Observation spaces across environments.

Symbol	Description	<i>Find</i>	<i>Bounce</i>	<i>Baoding</i>
f	normal forces	2	17	17
a	last action	9	20	20
θ	joint angles	9	24	24
$\dot{\theta}$	joint velocities	9	24	24
w	gripper width	1		
x_{ee}	EE position	3		
q_{ee}	EE quaternion	4		
o_t	timestep observation	37	85	85
s_t	stacked state ($S \times o_t$)	560 ($16o_t$)	340 ($4o_t$)	340 ($4o_t$)



(a) Franka contact sensors



(b) Shadow Hand contact sensors

Figure A8: Contact sensor configuration across robot embodiments.

D.2 Actions

Both robots are joint position controlled, with dimensions 9 and 20 for the Franka and Shadow Hand respectively. The Shadow Hand is underactuated, with coupled distal and proximal joints like in humans (2 wrist + 5 thumb + 3 index + 3 middle + 3 ring + 4 pinky = 20).

D.3 Rewards

The rewards for each environment step are given by the sum of the different terms each multiplied by the given scale. For enhanced value function learning, we track a running mean and variance for normalising the returns and values in the PPO update.

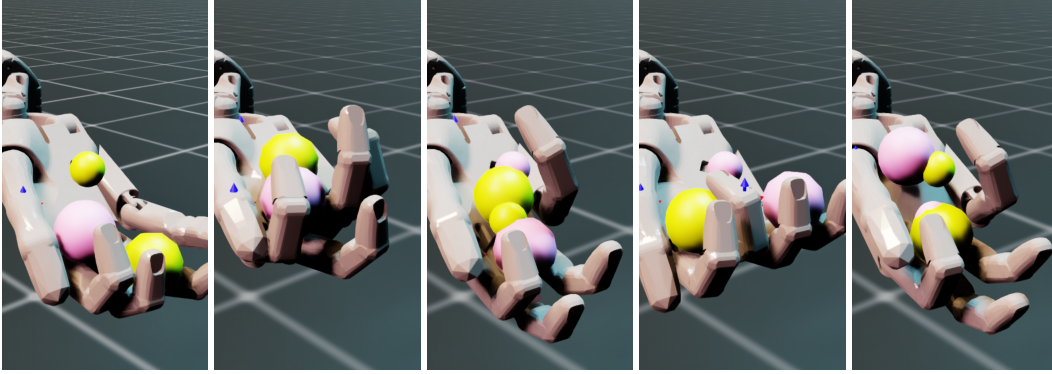
For *Find* there is one reward term r_{dist} that grows as distance between the object and end-effector decreases.

For *Bounce*, the reward is given by $r_{air} + r_{bounce} + r_{fall}$. To aid initial exploration, the agent is rewarded proportionally to number of time steps since last contact, r_{air} . A bounce event is defined as there being contact, then no contact, and then contact again, for which the agent is rewarded with a bonus r_{bounce} . The fall penalty is applied if the object is more than 24 cm from a fixed central position.

For *Baoding* the reward is given by $r_{dist_1} + r_{dist_2} + r_{rotation} + r_{fall}$. Our original approach was to maximise the xy angular velocity of the vector connecting the two balls which worked well, but sometimes the agent came up with creative strategies that no longer resembled the original Baoding task. Thus, we reformulated the reward around two fixed target poses (Figure A9). When the centers of both balls were within 1.0 cm of the given target centers, the targets switched and the agent receives a bonus reward $r_{rotation}$. We also needed to use two dense ball-center-to-target distance rewards r_{dist_1}, r_{dist_2} to aid exploration in the beginning. This approach better constrained the ball positions and stabilised policies for comparing methodologies. The fall penalty is applied if the distance between the balls exceeds 15 cm.

Table A2: Reward components across environments.

Symbol	Description	Equation	Scale	Find	Bounce	Baoding
r_{dist}	distance to target	$1 - \tanh d_{target}/0.1$	1, 0.1	✓		✓✓
r_{air}	time without contact	+1	0.01		✓	
r_{bounce}	successful bounce	+1	10		✓	
$r_{rotation}$	successful rotation	+1 if $d_{1\&2} < 1.0\text{cm}$	10			✓
r_{fall}	fall penalty	-1 if $d > d_{max}$	10		✓	✓

Figure A9: *Baoding*: When both balls are within 1cm of their virtual targets (shown as smaller balls), the targets switch.

D.4 Reset

Episodes can be terminated by a failure state, or truncated by a time limit. The *Find* environment episode length is $T = 300$ timesteps (5s), and *Bounce* and *Baoding* environments are $T = 600$ timesteps (10s). The *Bounce* and *Baoding* episodes can be terminated early if the balls fall out of the hand, measured by a distance. At the beginning of each episode, the Franka joint angles are randomised up to $\pm 7^\circ$ and the Shadow joint angles are randomised upto $\pm 20\%$. The ball in *Find* is randomised to any position on the $20\text{cm} \times 20\text{cm}$ plate. The *Bounce* ball and *Baoding* balls are randomised by $\pm 1\text{ cm}$ and $\pm 0.5\text{cm}$ respectively along the global xyz axis .

E Network architectures

Encoder. The encoder e is a 3-layer MLP with dimensions $s_t \rightarrow 1024 \rightarrow 512 \rightarrow 256 \rightarrow z_t$. Layer normalisation and ELU activations are applied after each layer.

Policy. The policy π is a 3-layer MLP with dimensions $z_t \rightarrow 128 \rightarrow 64 \rightarrow n_{actions} \rightarrow a_t$. ELU activations are applied after the first two layers. The output layer activation is tanh for *Find* and identity for *Bounce* and *Baoding*.

Value function. The value function v is a 3-layer MLP with dimensions $z_t \rightarrow 128 \rightarrow 64 \rightarrow 1 \rightarrow V_t$. ELU activations are applied after the first two layers. The output layer activation is identity.

Reconstruction. The decoder d is a 3-layer MLP. The full reconstruction decoder has dimensions $z_t \rightarrow 512 \rightarrow 512 \rightarrow \text{len}(s_t) \rightarrow \hat{s}_t$. The tactile reconstruction decoder has dimensions $z_t \rightarrow 512 \rightarrow 512 \rightarrow \text{len}(s_t^{tact}) \rightarrow \hat{s}_t^{tact}$. ELU activations are applied after the first two layers. The output layer activation is sigmoid for tactile predictions, and identity for proprioception predictions.

Forward dynamics. The forward model f is a 3-layer MLP with dimensions $(z_t, a_t) \rightarrow 512 \rightarrow 256 \rightarrow 256 \rightarrow \hat{z}_{t+1}$, with ELU activations after the first two layers. The projector is a 2-layer MLP with dimensions $\hat{z}_{t+i} \rightarrow 256 \rightarrow 256 \rightarrow \mathcal{L}_i$ with an ELU activation after the first layer. The target encoder e_T is identical to e , but updated according to Equation 1 with $\alpha = 0.01$.

$$\theta_{e_T} \leftarrow (1 - \alpha)\theta_{e_T} + \alpha\theta_e \quad (1)$$

F Training details

The combined loss L is the summed policy, value, entropy, and auxiliary loss (Equation 2).

$$L(\theta_e, \theta_\pi, \theta_v, \theta_{aux}) = L_\pi(\theta_e, \theta_\pi) + c_v L_v(\theta_e, \theta_v) + c_{ent} L_{ent}(\theta_\pi) + c_{aux} L_{aux}(\theta_e, \theta_{aux}) \quad (2)$$

RL. We use separate Adam optimisers for the policy, value, and encoder. The policy, value, and encoder optimisers share a constant learning rate lr . The gradient norms of the policy, value, and encoder networks are clipped at 1.0.

SSL. The same Adam optimiser is used to optimise the encoder and auxiliary-related networks (e.g., decoder d , forward model f , nonlinear projector p) with constant learning rate lr_{aux} . The gradient norms are *not* clipped: this seemed to degrade performance. The formulations of L_{aux} for the different objectives are shown in Section 3.2.

G Tactile forward dynamics objective

Figure A10 shows how our proposed tactile forward dynamics objective is computed, as described in Section 3.2. The loss optimises the encoder to learn a state representation z_t that can both predict future latent states and reconstruct the future tactile states from these latent states.

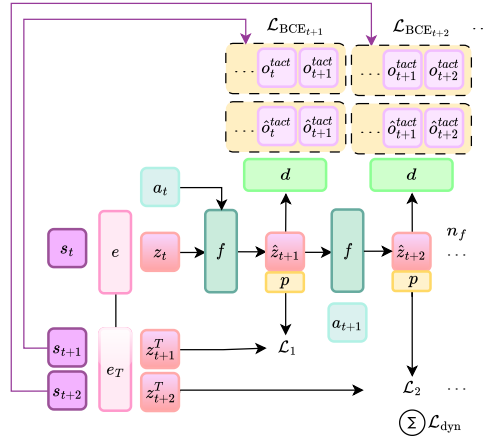


Figure A10: Tactile forward dynamics objective.

H Human capabilities

Find. Replicating the environment in real life and testing with one human subject, the average time to find and grasp the object across 10 trials was 2.1 seconds.

Bounce. The most tennis ball touches using the hand in one minute is 353, and was achieved by Manikandan Thirumaniselvam in India on 4 February 2023². This translates to $353/6 \sim 58$ bounces in 10 seconds. We note the properties of our ball (30g, 70mm diameter) are different to a tennis ball (~ 58 g, 67mm diameter), but would expect to see similar results.

Baoding. The fastest demonstration we could find online achieves 13 rotations in 10 seconds³. We believe the properties of our Baoding balls and the ones in the video are identical (55g, 1.5 inch diameter) since we possess the same ones and modelled our simulated ones off them.

²**Bounce record:** <https://www.guinnessworldrecords.com/world-records/590513-most-tennis-ball-touches-using-the-hand-in-one-minute>, **Bounce video:** <https://www.youtube.com/watch?v=ORiHY0MwT4A>

³**Baoding video:** <https://www.youtube.com/shorts/x-ns-auc098>

I Hyperparameter tuning

We carefully tuned all our experiments to give each agent the best shot. For fairness, we followed the same hyperparameter tuning recipe for each individual experiment (3 environments \times 7 experiments = 21 sweeps). We use the Optuna library [1] with the TPE sampler (5 startup trials) and no pruner. We wait for each sweep to reach 20 complete trials (some hyperparameter combinations lead to policy/value NaNs which are terminated early).

The hyperparameters and possible ranges we tested are provided in Table A3, with the optimised values in Table A4. We did not sweep over the following hyperparameters: discount factor $\gamma = 0.99$, value loss scale $c_v = 0.1$, gradient norm clip 1.0, value clip 0.2, ratio clip 0.2. We provide the mean evaluation returns of each run in the sweep of *Bounce* with full dynamics self-supervision in Figure A11 to demonstrate the sensitivity of our agents to hyperparameters, and illustrate the importance of performing quality hyperparameter tuning for each individual experiment.

Table A3: Tunable hyperparameters and ranges for each experiment.

Hyperparameter	Symbol	Tunable values
Rollout	R	$\{16, 32, 64\}$
Minibatches	mb	$\{4, 8, 16, 32, 64\}$
Learning epochs	le	$\{4, 8, 16, 32\}$
Learning rate	lr	$[10^{-5}, 10^{-3}] \subset \mathbb{R}$
Entropy loss scale	c_{ent}	$\{0, 0.05, 0.1\}$
Auxiliary learning rate	lr_{aux}	$[10^{-5}, 10^{-3}] \subset \mathbb{R}$
Auxiliary loss weight	c_{aux}	$[10^{-3}, 10] \subset \mathbb{R}$
Dynamics sequence length	$n_f + 1$	$\{2, 3, 4, 10\}$
Auxiliary memory size	$N_{rollouts}$	$\{2, 3, 4\}$

Table A4: Tuned hyperparameters for each experiment.

Environment	Experiment	R	mb	le	lr	c_{ent}	lr_{aux}	c_{aux}	n_f	$N_{rollouts}$
<i>Find</i>	PPO(prop)	32	16	8	1.06×10^{-5}	0				
	PPO(prop-tactile)	32	16	8	1.06×10^{-5}	0				
	+full recon	64	64	4	7.39×10^{-5}	0	5.91×10^{-5}	0.0023		
	+tactile recon	64	16	8	1.36×10^{-5}	0.1	2.55×10^{-5}	0.004477		
	+full dynamics	64	64	4	1.15×10^{-5}	0.1	1.55×10^{-4}	0.0062	2	
	+tactile dynamics	64	64	4	2.32×10^{-5}	0	1.57×10^{-4}	0.0024563	4	
	+full dynamics+ $N_{rollouts}$	64	64	4	1.15×10^{-5}	0.1	3.81×10^{-5}	0.1364	3	3
<i>Bounce</i>	PPO(prop)	32	32	4	5.93×10^{-5}	0				
	PPO(prop-tactile)	16	8	4	3.21×10^{-4}	0				
	+full recon	64	16	16	1.88×10^{-4}	0	2.77×10^{-5}	0.05669		
	+tactile recon	64	32	16	4.65×10^{-5}	0	5.13×10^{-5}	0.00384		
	+full dynamics	32	64	4	1.50×10^{-4}	0	4.53×10^{-5}	0.8462	10	
	+tactile dynamics	64	32	8	1.22×10^{-4}	0.05	1.64×10^{-4}	0.23547	10	
	+full dynamics+ $N_{rollouts}$	32	64	4	1.50×10^{-4}	0	1.16×10^{-4}	0.19954	4	2
<i>Baoding</i>	PPO(prop)	32	8	4	9.96×10^{-5}	0.05				
	PPO(prop-tactile)	32	4	8	2.02×10^{-4}	0.05				
	+full recon	16	32	4	3.68×10^{-4}	0	5.18×10^{-5}	0.058866		
	+tactile recon	64	32	8	3.61×10^{-4}	0	1.00×10^{-5}	0.2707		
	+full dynamics	32	16	4	5.47×10^{-4}	0	2.87×10^{-4}	3.686	2	
	+tactile dynamics	32	16	8	2.08×10^{-5}	0.05	1.53×10^{-4}	0.04839	3	
	+full dynamics+ $N_{rollouts}$	32	16	4	5.47×10^{-4}	0	1.67×10^{-5}	1.6349	4	4

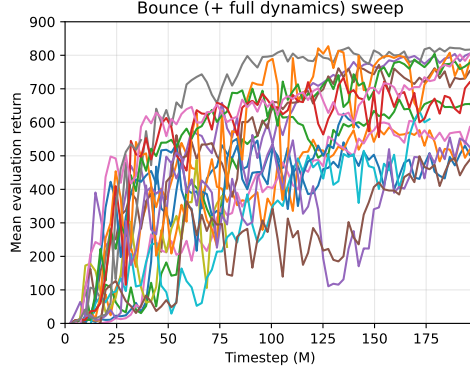


Figure A11: Sweep of *Bounce* (+ full dynamics), demonstrating sensitivity of agent to different hyperparameter combinations.

J Latent trajectory analysis

Figures A12, A13, and A14 show a two dimensional latent representation of a single episode across all environments. Trajectories for RL-only and a subset of self-supervised agents are shown (tactile reconstruction, full dynamics, and tactile dynamics). The 256-dim z_t latent vector at each timestep was reduced using 2-component Principal Component Analysis (PCA). Note that the tactile activations shown are only the sum of activations in the current observation, and does not sum the history.

Baoding. The ring-like trajectory of the RL-only agent illustrates the repeated motion the agent develops. There are two tactile peaks on opposite sides of the ring, indicating symmetry in contact activations between half-rotations. The trajectory of the tactile reconstruction agent is quite different (heart-shaped, diffuse). This shows each rotation is slightly different, and there is now asymmetry between the contact activations of half-rotations. From rendering the policy, the gait is smooth like the dynamics agent but keeps the balls close together like the RL-only agent. The trajectory of the full dynamics agent is again ring-like, but with tighter bounds than the RL-only agent. Like the tactile reconstruction agent, there is contact activation asymmetry between half-rotations. Finally, the tactile dynamics agent trajectory appears to be a blend of the dynamics and tactile reconstruction trajectories.

Bounce. The latent trajectories of the self-supervised agents are highly different to the RL-only agent. From the trajectory with the time colourbar, we can see that the sequential latent states of the RL-only agent are highly discontinuous and far apart (e.g., yellow), and the agent repeats the same motion with high precision. There are two regions with non-zero tactile observations of upto 6 activations, which is understood by the ‘safe’ gait of raising the index and pinky finger to stabilise the ball. The gait changes completely for the self-supervised agents, which predominately uses 1 or 2 contacts. Sequential latent states are still spread out in various regions, but these regions are much more diffuse than in the RL-only.

Find. It is clear the self-supervised agents find the object faster by observing the trajectories colourised by time. Otherwise, the shape of the latent trajectories is not drastically different between RL-only and self-supervised agents. A distinction between 1 and 2 tactile activations appears in the dynamics agent trajectory.

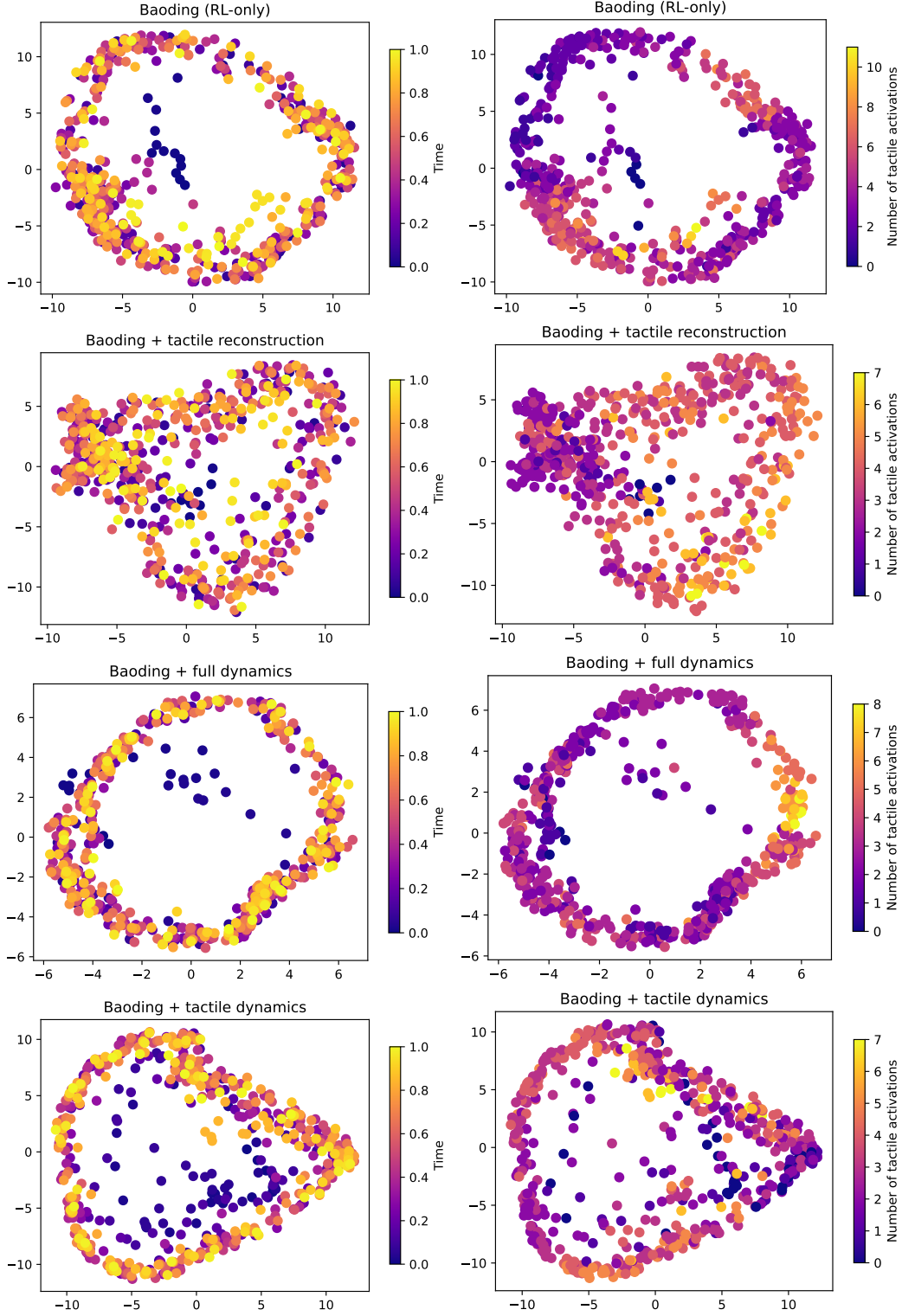


Figure A12: Latent episode trajectory (PCA) of the best *Baoding* agents. **Left:** Samples coloured by time. **Right:** Samples coloured by summed tactile activations of the last tactile observation o_t^{tact} .

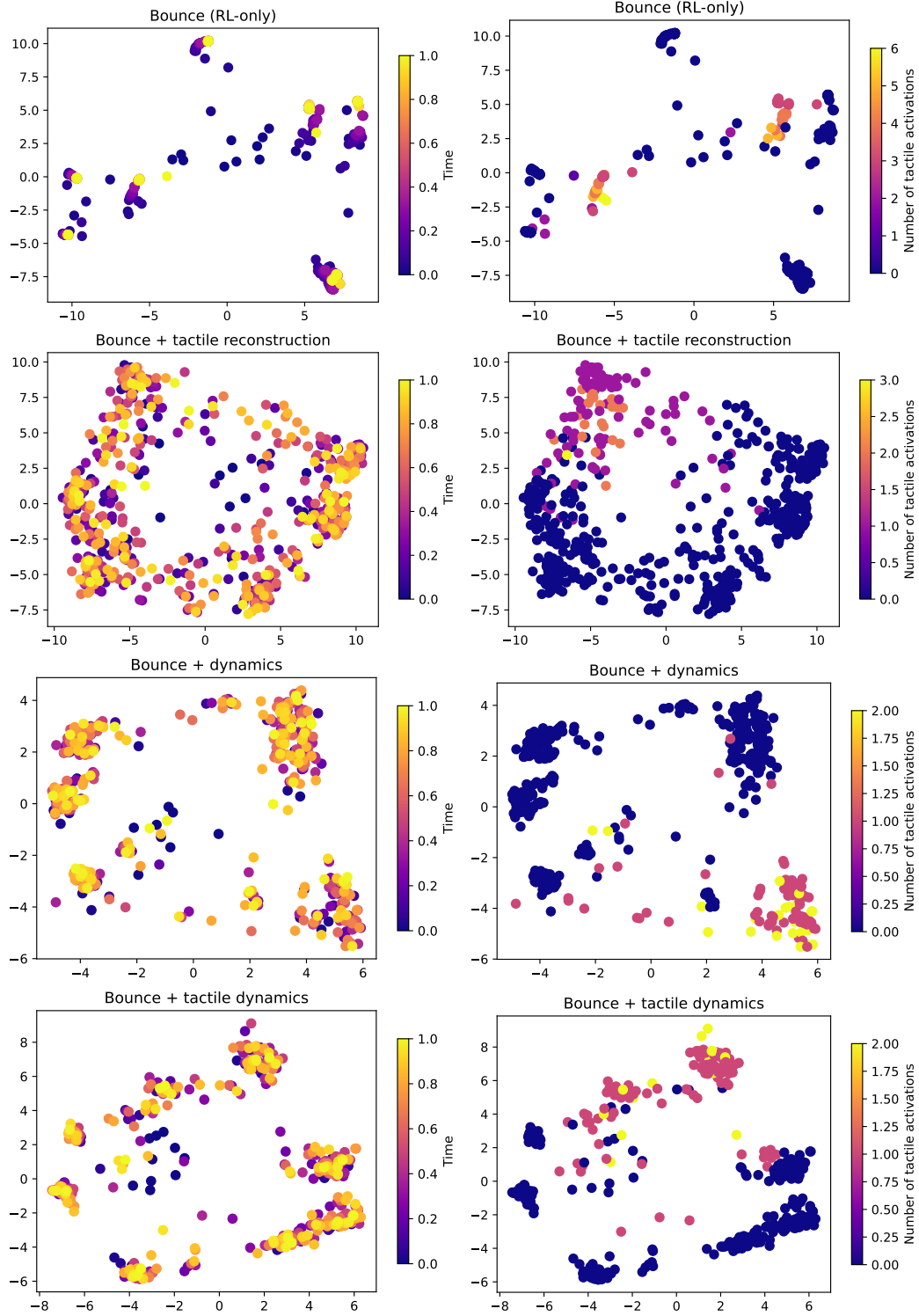


Figure A13: Latent episode trajectory (PCA) of the best *Bounce* agents. **Left:** Samples coloured by time. **Right:** Samples coloured by summed tactile activations of the last tactile observation o_t^{tact} .

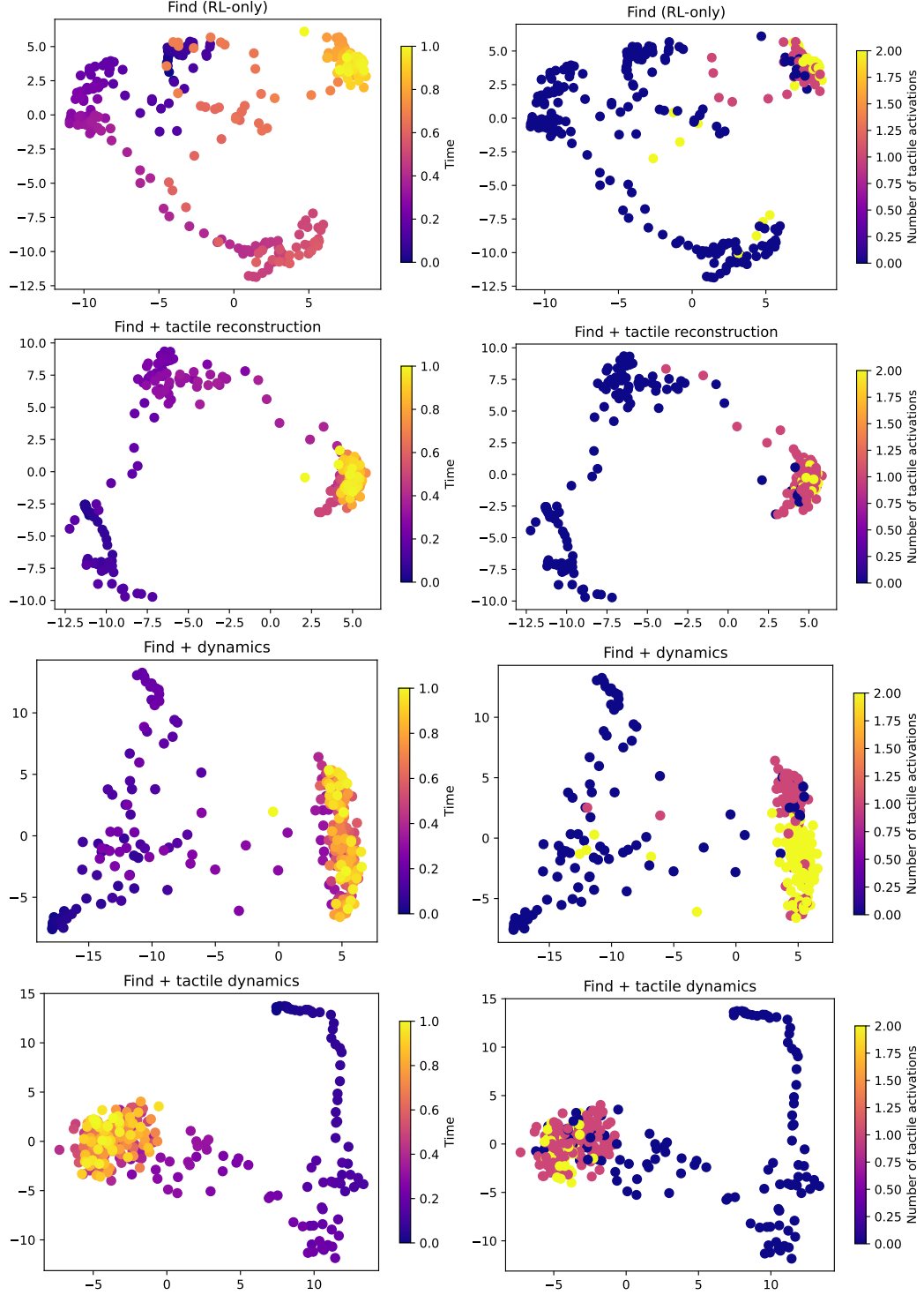


Figure A14: Latent episode trajectory (PCA) of the best *Find* agents. **Left:** Samples coloured by time. **Right:** Samples coloured by summed tactile activations of the last tactile observation o_t^{tact} .

References

- [1] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2019.