From Contact to Return: Curriculum and Predictive Shaping for Humanoid Table Tennis

Hannah Clay
Department of Computer Science
Stanford University
hclay116@stanford.edu

Shane Mion
Department of Computer Science
Stanford University
smion@stanford.edu

Kyle Schmoyer
Department of Computer Science
Stanford University
kyle7@stanford.edu

Abstract

Robotic table tennis is a difficult reinforcement-learning benchmark because it combines high-dimensional humanoid control, fast ball dynamics, sparse task success, and contact-rich post-impact behavior. We study a Booster K1 humanoid in the Isaac Lab `k1_tt` table-tennis task using the RSL-RL implementation of Proximal Policy Optimization (PPO). Our original question was whether a policy that receives a learned prediction of future ball pose outperforms a reactive policy that observes only the current ball state. The final result is more nuanced: our controlled experiments do not conclusively establish that predictive observations alone outperform reactive observations. Instead, the dominant finding is that sparse reward, curriculum design, and contact-to-return conversion determine whether useful learning occurs at all.

The task naturally separates into two stages. First, the robot must acquire contact by moving the paddle close enough to the incoming ball. Second, it must convert that contact into a valid return by controlling paddle velocity, paddle face orientation, net clearance, and opponent-table landing. Early sparse-reward runs produced little useful task signal. We therefore introduced curriculum and reward modifications: widened contact basins, narrowed serve/reset distributions, a first-contact detection fix, touchpoint-axis shaping with larger vertical error weight, and later post-contact shaping for predicted net clearance and table landing. These changes produced intermittent table-success behavior, but not a robust final policy.

The most defensible interpretation is diagnostic rather than state-of-the-art performance. A same-threshold poster comparison found identical peak table-success reward for reactive and predictive 0.35 m curriculum runs, so it should not be used as proof that prediction alone is better. Widened-basin runs reached high hit rates, but hit rate did not imply true table-tennis success. A later strict 0.05 m return-oriented run preserved the true contact threshold and difficult serve distribution and produced a strong temporary window of valid returns, but the behavior was not sustained; final cumulative strict success remained low. Because reward weights and contact thresholds changed across some runs, raw weighted reward magnitudes are treated as within-configuration diagnostics rather than directly comparable final success rates.

Our main contribution is an empirical failure analysis for humanoid table tennis with model-free PPO and auxiliary prediction. Prediction and touchpoint shaping can help define useful intercept-oriented learning signals, but the harder bottleneck is controlling the outgoing ball after contact. Future work should run frozen-checkpoint evaluation under common serve distributions and contact thresholds,

report physical event rates rather than only weighted rewards, and ablate predictive observations separately from predictor-guided reward shaping.

1 Introduction

Robotic table tennis is a compact but difficult benchmark for dynamic manipulation. A successful robot table-tennis player must perceive a fast-moving ball, move its paddle to an appropriate intercept point, coordinate whole-body posture, and produce a controlled post-contact ball trajectory. This makes the task harder than ordinary reaching or locomotion because success depends not only on where the robot's end effector moves, but also on how the paddle-ball collision changes the ball's velocity.

In this project, we investigate humanoid table tennis using deep reinforcement learning. The task is `k1_tt`: a 22-degree-of-freedom Booster K1 humanoid must contact a served ping-pong ball and return it across the table. Our central research question is whether predicting future ball state helps a humanoid table-tennis policy more than reacting only to the current ball state.

This question is motivated by the fast dynamics of table tennis. A purely reactive policy observes the current ball state and must immediately choose an action. A predictive policy, in contrast, can move toward where the ball will be rather than where it currently is. We expected this to matter because the humanoid body and paddle cannot move instantaneously.

However, the project revealed that prediction alone is not the only or even the dominant issue. Early sparse-reward experiments showed that PPO can receive near-zero useful task signal before the agent learns reliable ball contact. As a result, much of the project became an empirical study of reward engineering: which curriculum signals help the policy first learn contact, and which signals help it convert contact into a valid return.

Our main contribution is an analysis of this learning progression. We compare reactive and predictive policy structures where possible, but we do not claim a clean predictive-over-reactive result. Instead, the main finding is that curriculum and reward design dominate performance in the tested setting, and that contact success is not the same as true table success. A policy can often move the paddle near the ball while still failing to send the ball over the net and onto the opponent's table half.

2 Related Work

Robot table tennis has been studied through both skill-learning and full-system robot-control approaches. Earlier work learned striking motions and generalized table-tennis behaviors from experience [Mülling et al., 2013, Büchler et al., 2022]. More recent systems frame robotic table tennis as a high-speed learning problem that requires perception, prediction, planning, and control to work together [D'Ambrosio et al., 2023]. These systems motivate our project because a table-tennis robot must react to a fast-moving ball while also controlling the result of a contact-rich paddle-ball collision.

Our work is based on the TTRL humanoid table-tennis environment [Hu et al., 2025]. Unlike classical modular table-tennis systems, our final controller is trained with model-free reinforcement learning. The learned predictor is used as an auxiliary future-state signal and shaping target, not as a full dynamics model for planning.

The training setup uses Isaac Lab for GPU-parallel simulation [Mittal et al., 2025] and RSL-RL for PPO training [Schwarke et al., 2025, Schulman et al., 2017]. Isaac Gym and Isaac Lab-style simulation are useful for this kind of work because they allow many robot environments to run in parallel on the GPU [Makoviychuk et al., 2021]. This makes high-dimensional humanoid reinforcement learning more practical than single-environment simulation.

The main learning issue in our project is sparse reward. Successful table-tennis returns are rare early in training, so PPO receives little useful task signal before the policy learns to contact the ball. Our curriculum and reward changes are related to curriculum learning [Bengio et al., 2009] and reward shaping [Ng et al., 1999]. However, our results also show a common danger of shaping: proxy rewards such as widened contact can improve hit rate without solving the true return task.

3 Methods

3.1 Environment and Infrastructure

We use Isaac Lab [Mittal et al., 2025], built on NVIDIA Isaac Sim, as the simulator. The robot is the Booster K1 humanoid in the `k1_tt` table-tennis task. Training and evaluation are run through a Modal-based infrastructure with headless Isaac Lab jobs, Weights & Biases logging, and checkpoints stored on a Modal volume. Interactive Isaac Sim GUI rendering was unreliable in the cloud environment because native/noVNC attempts encountered renderer and Vulkan issues, so our final workflow uses headless physics as the main compatibility target.

The main setup is:

- **Simulator:** Isaac Lab / NVIDIA Isaac Sim.
- **Robot:** Booster K1 humanoid, 22 DoF.
- **Task:** Intercept and return a table-tennis ball.
- **Algorithm:** PPO through RSL-RL.
- **Parallelism:** 256–512 environments, depending on run.
- **Training horizon:** typically 750–1200 PPO iterations.
- **Compute:** NVIDIA L40S GPU through Modal.
- **Throughput:** approximately 2,400–6,150 environment steps per second in the key touchpoint-axis runs.
- **Logging:** W&B project path `kyles7-stanford-university/leggedlab`.

The working repo owns Docker, Modal, docs, reports, and experiment commands, while the table-tennis task code lives in the forked TTRL submodule. We used smoke tests to verify headless physics and imports before running experiments.

3.2 MDP and PPO Training

We model the task as a finite-horizon Markov decision process with state s_t, action a_t, reward r_t, and termination signal d_t. The policy outputs continuous humanoid joint-control actions through the TTRL/RSL-RL control stack. The observation contains robot proprioception and ball state; the predictive variant augments this observation with a learned future ball-pose estimate. Episodes terminate according to the task environment's table-tennis and humanoid-stability conditions, including failed or completed ball interactions and robot termination events.

We train policies using the RSL-RL implementation of PPO [Schwarke et al., 2025, Schulman et al., 2017] rather than a custom PPO implementation. PPO optimizes a clipped surrogate objective

$$L^{\text{clip}}(\theta) = \mathbb{E}_t \left[\min \left(\rho_t(\theta)\hat{A}_t, \text{clip}(\rho_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t \right) \right], \tag{1}$$

where $\rho_t(\theta) = \pi_\theta(a_t \mid s_t)/\pi_{\theta_{\text{old}}}(a_t \mid s_t)$ and $\hat{A}_t$ is the advantage estimate. The full training objective also includes value-function and entropy terms,

$$L(\theta) = L^{\text{clip}}(\theta) - c_v L^{\text{value}}(\theta) + c_e \mathcal{H}(\pi_\theta). \tag{2}$$

Runs used 256–512 parallel Isaac Lab environments and approximately 750–1200 PPO iterations depending on the experiment. For the hover-stability sweep, each 512-environment, 900-iteration run represented 11,059,200 simulated transitions. For reproducibility, the final submitted version should report the exact RSL-RL configuration from the training config, including rollout length, control frequency, discount factor, GAE parameter, clip ratio, learning rate, number of epochs, minibatch size, entropy coefficient, value coefficient, policy/value network architecture, and observation/advantage normalization settings. In this draft, we report the settings that were consistently tracked in the experiment notes: environment count, iteration count, seed when available, contact threshold, serve ranges, reward changes, and W&B metrics.

3.3 Reactive and Predictive Policies

We compare two policy variants.

Reactive policy. The reactive policy receives robot proprioception and the current ball state. It must choose actions based on the ball's current position and velocity.

Predictive policy. The predictive policy receives the same information as the reactive policy, plus a predicted future ball pose. This prediction is produced by TTRL's predictor-augmented runner, where a small online MLP predicts future ball position. In the current experiments, prediction can affect both the policy input and the shaping signal, so the results should be interpreted as *predictive shaping* rather than a clean test of predictive observation alone. A stronger future ablation would separately test: (1) current-state observations with nonpredictive shaping, (2) predicted-state observations with nonpredictive shaping, (3) current-state observations with predictive touchpoint shaping, and (4) predicted-state observations with predictive touchpoint shaping.

	Reactive	**Predictive**
Observation	Proprioception + current ball state	Same + predicted future ball pose
Predictor	None	Small online MLP
Training	PPO / RSL-RL	PPO / RSL-RL with predictor runner
Goal	React to current ball	Anticipate future intercept point

Table 1: Reactive and predictive policy comparison.

3.4 Predictor Details and Ablation Caveat

The predictor is a learned future-state module from the TTRL predictor-augmented runner [Hu et al., 2025]. Its role is to estimate a future ball pose that can be used by the policy and by future end-effector shaping terms. The report should state the exact predictor input vector, prediction horizon, output target, architecture, loss function, label-generation procedure, and whether predictor gradients are shared with the policy. Those details should be copied from the TTRL configuration/source before final submission. The current evidence is still useful, but it cannot distinguish whether gains come from extra predictive information, from predictor-guided reward shaping, or from the surrounding curriculum.

3.5 Reward and Curriculum Patches

The baseline task contains sparse table-tennis success signals and locomotion regularizers. We describe the shaped reward as

$$r_t = \sum_i \lambda_i r_i(s_t, a_t, s_{t+1}), \tag{3}$$

where terms include task success, paddle-ball proximity, future touchpoint alignment, humanoid regularization, and post-contact return shaping depending on the profile. Sparse success alone was not enough for early learning, so we applied runtime patches inside the Modal container before training. The most important changes were:

1. **Contact detection fix.** A likely first-contact detection issue was patched so that contact is counted when `contact_score > 0`. This avoids missing the first contact frame.
2. **Wider contact basins.** The original contact threshold was 0.05 m. Curriculum runs used thresholds such as 0.25 m and 0.35 m to expose the agent to earlier contact reward.
3. **Touchpoint-axis shaping.** The future end-effector target was changed from a generic paddle position to the actual paddle touch point. The distance error was axis-weighted with weights $w_x = 1.0$, $w_y = 1.5$, and $w_z = 3.0$ because earlier rollouts often missed vertically even when table-plane position was close.
4. **Reward scaling.** The contact reward was increased, and predictor-guided `reward_future_dis_ee` shaping was strengthened.

5. **Serve and reset narrowing.** Urgent success curricula narrowed ball velocity ranges and robot reset poses. For example, the touchpoint-axis curriculum narrowed x speed from $(-6.5, -5.0)$ to $(-5.6, -5.0)$ m/s, y speed from $(-0.8, 0.4)$ to $(-0.2, 0.2)$ m/s, and z speed from $(1.5, 2.0)$ to $(1.55, 1.85)$ m/s.
6. **Post-contact shaping.** The strict 0.05 m return profile kept the original difficult serve/reset distribution and true 0.05 m threshold, but added return-oriented shaping for predicted opponent-table landing and net clearance.

We use the following contact shaping intuition:

$$\text{contact}(d_t) = \text{clip}\left(\frac{\tau - d_t}{\tau}, 0, 1\right), \tag{4}$$

where d_t is paddle-ball distance and τ is the contact threshold. A larger τ creates a wider basin and gives earlier learning signal, while a smaller τ forces more precise contact.

3.6 Experimental Configurations

We group the experiments into four sets:

- **Touchpoint-axis curriculum runs.** Predictive runs using widened 0.35 m contact basin, narrowed serves/resets, touchpoint-axis shaping, and W&B logging. The main runs used 256 environments for 1000 iterations and 512 environments for 750 iterations.
- **Poster comparison runs.** A reactive 0.35 m curriculum run, a predictive 0.35 m run, and a predictive stricter 0.15 m run, all around 750 iterations. These compare predictor usage under similar curriculum patches.
- **Hover stability sweep.** Three predictor-based PPO runs with 512 environments and 900 iterations tested whether explicitly rewarding steady humanoid base height/orientation improves table-tennis performance.
- **Strict 0.05 m return run.** A later run with 512 environments and 1200 iterations kept the original difficult serve distribution, true 0.05 m contact threshold, and real opponent-table bounce success definition. It added post-contact ballistic landing and net-clearance shaping.

4 Results

4.1 Touchpoint-Axis Curriculum Runs

The first successful predictive curriculum runs were the two touchpoint-axis runs in Table 2. The longer 256-environment run produced the best peak table-success reward, while the 512-environment run produced better throughput and higher contact-rate style metrics.

Metric	256 env, 1000 iter	512 env, 750 iter
Peak table-success reward	**0.0500**	0.0375
Final table-success reward	**0.0167**	0.00625
Peak sparse success rate	**0.368%**	0.315%
Peak / final hit rate	88.64% / 86.72%	**91.17% / 91.17%**
Peak contact reward	**0.4705**	0.4370
Peak future touchpoint reward	14.08	**14.38**
Peak / final FPS	2393 / 2253	**6150 / 5871**

Table 2: Key W&B metrics for the two predictive touchpoint-axis curriculum runs. These runs used a widened 0.35 m contact basin, so they show curriculum-level success rather than final strict performance.

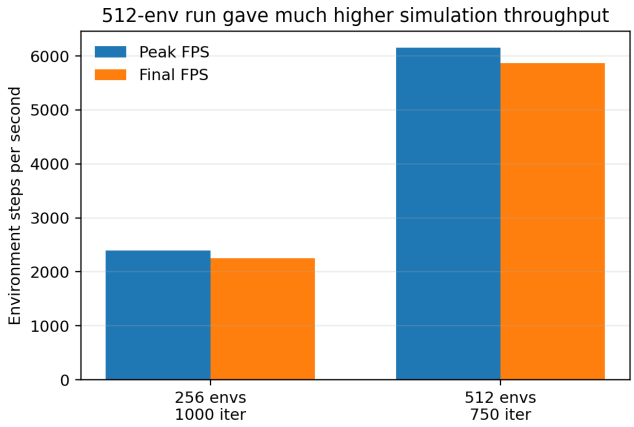

Figure 1: The 512-environment touchpoint-axis run provided much higher environment-step throughput than the 256-environment run.

The strongest safe claim from these runs is that the predictive touchpoint-axis curriculum produced intermittent table-success reward and high contact behavior. It should not be presented as robust table-tennis success under the original 0.05 m threshold.

4.2 Reactive vs. Predictive Poster Comparison

The poster-session comparison added a reactive curriculum baseline and a stricter predictive variant. All runs logged enough W&B data for poster-level interpretation, although the concurrent runs filled the Modal volume because W&B cache files were written to the shared volume. This was fixed by moving `WANDB_DIR` to `/tmp`.

Condition	Threshold	Peak table success	Peak contact	Peak future dis EE
Reactive	0.35 m	0.0375	0.447	**15.22**
Predictive	0.35 m	0.0375	0.437	14.38
Predictive strict	0.15 m	0.0333	**0.638**	14.76

Table 3: Poster comparison runs. Contact reward is not directly comparable across thresholds because the reward formula changes with the contact basin. The predictive 0.35 m run also logged a 91.2% hit rate, while the predictive 0.15 m run logged 57.6%.

The key takeaway is that predictor-based shaping was useful, but the final comparison is confounded by curriculum and reward changes. The reactive baseline should be interpreted carefully rather than treated as a clean final ablation.

4.3 Hover Stability Sweep

The hover stability experiment tested whether explicitly rewarding a steady K1 trunk would improve table-tennis performance. The profile rewarded root height near 0.72 m, upright posture, low vertical velocity, and low angular velocity. Three predictor-based PPO runs used 512 environments and 900 iterations.

Metric	Hover mean	Strict reference	Relative change
Peak table-success reward	0.0644	**0.1900**	-66.1%
Hit rate	58.34%	**68.80%**	-15.2%
Sparse success rate	0.499%	**0.581%**	-14.0%
Peak mean reward	143.8	**164.4**	-12.5%
Latest mean episode length	**272.2**	265.0	+2.7%
Latest termination penalty	**-0.660**	-0.868	24.0% less negative

Table 4: Hover stability improved some survival-related metrics but reduced table-tennis performance.

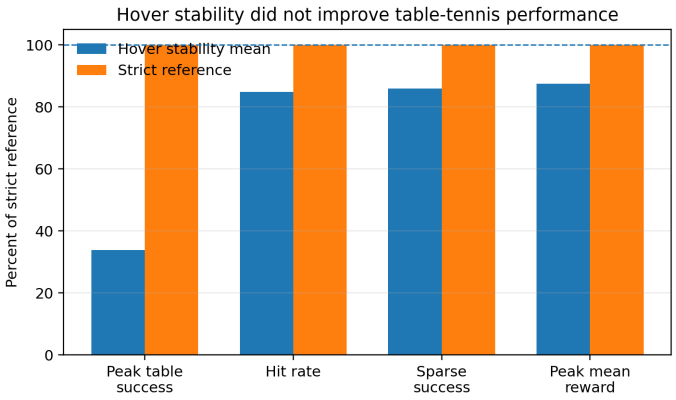

Figure 2: Hover stability metrics normalized against the strict-contact reference. Longer survival did not translate into better table-tennis returns.

The interpretation is that a generic steady-base reward competed with the fast whole-body motion needed to reach and redirect the ball. It kept episodes alive slightly longer, but this did not improve task success.

4.4 Strict 0.05 m Return-Oriented Run

The most important later experiment was the `strict_005_table_return` profile. Unlike the 0.35 m curriculum runs, this run preserved the true 0.05 m paddle contact threshold, the original difficult serve velocity ranges, the upstream reset distribution, interval pushes, and the real opponent-table bounce success definition. The profile changed only reward terms: contact weight increased from 150 to 500, future paddle-touchpoint alignment used weight 100 with axis-weighted error, real table-success weight increased from 100 to 1500, post-contact ballistic landing target used weight 120, and post-contact predicted net clearance used weight 70.

Strict 0.05 m metric	**Result**
Peak windowed table-success reward	**0.500 at iter. 945**
Final windowed table-success reward	0.031
Final cumulative strict hit rate	12.018%
Peak cumulative strict success rate	0.189% at iter. 399
Final cumulative strict success rate	0.088%
Peak / final mean reward	85.01 / 81.42
Final mean episode length	263.13

Table 5: Strict 0.05 m return-oriented run. The peak windowed table-success reward shows that valid returns occurred in a concentrated reporting window, but the cumulative sparse success rate remained low.

This result changes the main story of the project. The 0.35 m curriculum runs showed that the agent could learn contact-like behavior under an easier basin. The strict 0.05 m return run showed that the policy can occasionally produce valid opponent-table returns under the true contact threshold and original difficult serves. However, the behavior was not stable. W&B `Episode_Reward/reward_table_success` is a windowed reward statistic, while `Train/TT_success_rate` is cumulative. In addition, the strict profile increased the table-success reward weight, so the raw reward magnitude is not directly comparable to earlier profiles. Therefore a peak of 0.500 should be described as a strong temporary within-run window, not as a stable 50% success policy or as a directly normalized improvement over other reward profiles.

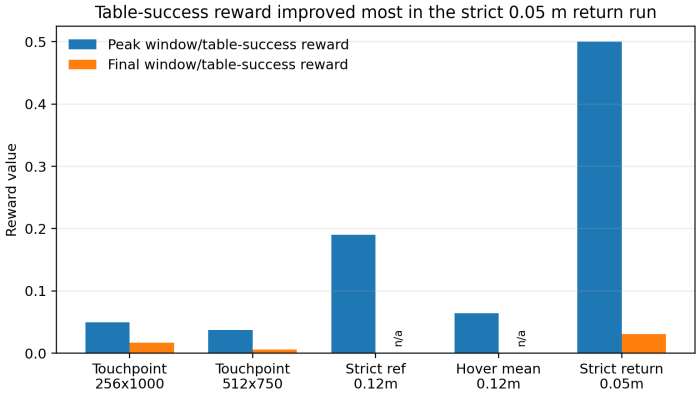

Figure 3: Windowed table-success reward within each configuration. Because reward weights changed across profiles, raw weighted magnitudes should not be interpreted as a controlled cross-run success-rate comparison.

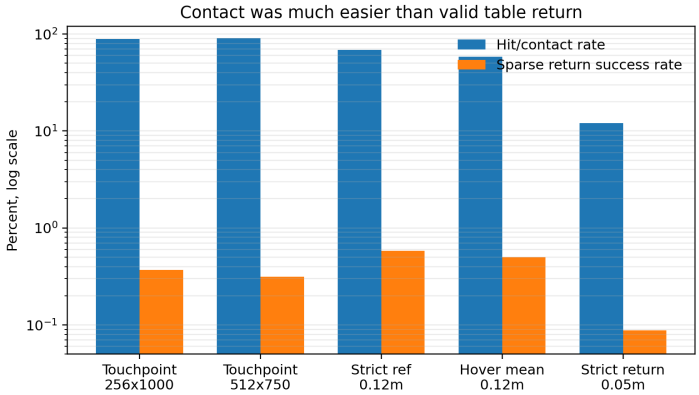

Figure 4: Hit/contact rate is much larger than sparse return success rate. The y-axis is logarithmic because contact and valid return occur at very different frequencies.

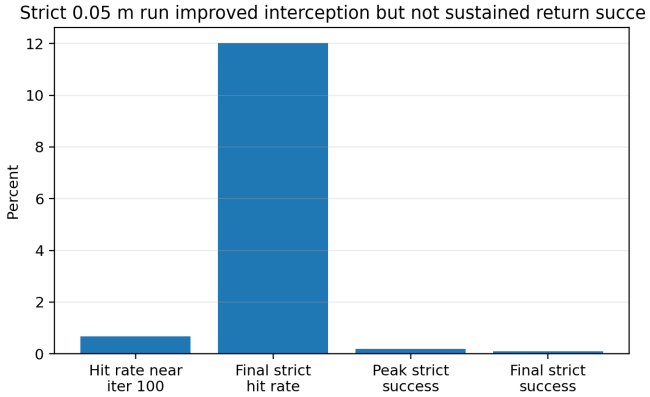

Figure 5: The strict 0.05 m profile improved interception from about 0.67% near iteration 100 to 12.018% final strict hit rate, but final strict return success remained low.

4.5 Recommended Common-Condition Evaluation

The current results are primarily training-log diagnostics. A stronger final evaluation should freeze selected checkpoints and run each for a fixed number of episodes under identical conditions. Table 6 defines the evaluation we would use before making final claims about predictive versus reactive control.

Evaluation item	Fixed setting
Checkpoints	Best and final checkpoint for each condition
Episodes	500–1000 per checkpoint
Serve distribution	Same original difficult serve distribution
Robot initialization	Same reset distribution and disturbances
Contact thresholds	Report both 0.12 m and 0.05 m
Success definition	Paddle contact followed by opponent-table bounce
Metrics	Contact rate, net-clearance rate, landing rate, valid-return rate
Uncertainty	Episode-level confidence intervals or multiple seeds

Table 6: Common-condition frozen-checkpoint evaluation needed for a clean final comparison.

5 Discussion

Our original framing emphasized predictive versus reactive control, but the experiments do not cleanly establish that predictive observations outperform reactive observations. The same-threshold poster comparison produced equal peak table-success reward for reactive and predictive 0.35 m curriculum runs, while later strict and return-oriented profiles changed thresholds and reward terms. The more defensible conclusion is that prediction helped define useful shaping targets, but reward design and curriculum dominated performance in the tested setting.

The results support a two-stage view of humanoid table tennis:

1. **Contact acquisition.** The policy must first learn to move the paddle near the incoming ball. Widened contact basins, narrowed serves, and touchpoint-axis shaping help with this stage.
2. **Return control.** After contact, the policy must control the outgoing ball so it clears the net and lands on the opponent's table. This requires paddle velocity, paddle face orientation, timing, and post-contact trajectory shaping.

The high hit rates in widened-basin runs show that contact acquisition can improve without solving the actual table-tennis objective. Conversely, the strict 0.05 m run shows that real returns can happen under the true contact threshold, but sparse cumulative success remains low. The next bottleneck is therefore conversion: among contacts, how many become valid opponent-table bounces?

The hover stability experiment was also informative. It showed that intuitive humanoid-control objectives can conflict with the task. A steadier base is not automatically useful when the robot must execute fast reaching and redirection. The right stability regularizer may still matter, but a generic hover-style reward was not the next best training direction.

6 Limitations

Several limitations affect the interpretation of these results.

First, the experiments are not a perfectly controlled benchmark. Some runs used different contact thresholds, serve distributions, and reward weights. These changes were necessary to make progress, but they make direct comparisons harder.

Second, contact reward is not directly comparable across thresholds. A 0.35 m contact basin rewards a much easier event than the original 0.05 m threshold. Weighted table-success rewards are also not directly comparable when reward coefficients change across profiles. For this reason, widened-basin hit rate and weighted reward magnitudes should be described as curriculum-level diagnostics, not final task-success rates.

Third, some W&B runs were marked as crashed even though useful training metrics were logged. Modal logs did not show PPO training tracebacks in the final window for the key touchpoint-axis runs, so the likely failure point was shutdown, artifact sync, or app teardown. Still, checkpoint verification and common-threshold evaluation are needed before treating artifacts as final.

Fourth, the hover stability sweep had an experiment-design caveat: the completed three-run sweep was launched before `hover_stability_strict_contact` was corrected to inherit every reward weight from the strict-contact reference. The results are sufficient to reject that hover formulation as the next direction, but not to estimate a precise causal effect size.

Fifth, visual debugging was limited by cloud GUI instability. Most conclusions rely on W&B metrics and logged quantities. Future work should record rollouts and classify failures visually: missing the ball, contacting with poor paddle angle, sending the ball downward, hitting into the net, or overhitting.

7 Conclusion and Future Work

We studied humanoid table tennis using PPO in Isaac Lab. The project began as a comparison between reacting to the current ball state and predicting future ball state, but the current experiments do not prove that predictive observations alone outperform reactive control. The final lesson is broader and more useful: sparse reward, curriculum design, and post-contact dynamics dominate the learning problem.

The touchpoint-axis curriculum runs produced the first intermittent task-success signal under a widened 0.35 m contact basin. The hover stability sweep showed that generic base-stability reward did not improve table-tennis performance. The strict 0.05 m return-oriented run was the strongest evidence that the agent can occasionally produce true returns under the original contact threshold and difficult serves, reaching a peak windowed table-success reward of 0.500. However, final cumulative strict success was still only 0.088%, so the policy is not yet robust.

Future work should focus on common-threshold evaluation and contact-to-return conversion. First, all checkpoints should be evaluated under the same 0.12 m and 0.05 m thresholds. Second, training should use threshold annealing, for example:

$$0.35\ \text{m} \rightarrow 0.25\ \text{m} \rightarrow 0.15\ \text{m} \rightarrow 0.05\ \text{m}. \tag{5}$$

Third, rewards should directly shape post-contact behavior: positive outgoing ball velocity toward the opponent side, predicted net clearance, predicted landing on the table, paddle face orientation at contact, and penalties for downward or sideways returns. The project shows that hitting the ball is only an intermediate milestone. The real task is learning to hit the ball correctly.

8 Contributions

Hannah Clay contributed to training and evaluation runs, plot organization, TensorBoard/W&B workflow, and interpretation of reinforcement learning results.

Shane Mion contributed to Modal/Docker integration, repo hygiene, experiment orchestration, implementation/debugging of reward and curriculum variants, and metric tracking.

Kyle Schmoyer contributed to TTRL environment/assets checks, table/paddle/ball debugging, project framing, report/poster writing, reward-shaping analysis, experimental interpretation, and presentation of the distinction between contact success and true table success.

The final division of work changed from the original plan because the project became more infrastructure- and reward-shaping-heavy than expected. Sparse reward failure and simulator/debugging constraints required more effort on curriculum design, experiment diagnosis, and interpretation of partial but informative runs.

9 AI Tools Disclosure

ChatGPT was used as a writing and revision assistant for this report. It helped synthesize experiment notes, suggest clearer framing, identify overclaiming risks, draft LaTeX revisions, and improve

figure/report formatting. The project design, implementation work, experiment execution, interpretation decisions, and final responsibility for technical accuracy remain with the authors. The authors reviewed and edited generated text before submission.

References

Y. Bengio, J. Louradour, R. Collobert, and J. Weston. Curriculum learning. *Proceedings of the 26th International Conference on Machine Learning*, pages 41–48, 2009.

D. Büchler, S. Guist, R. Calandra, B. Schölkopf, and J. Peters. Learning to play table tennis from scratch using muscular robots. *IEEE Transactions on Robotics*, 38(6):3850–3860, 2022.

D. B. D'Ambrosio, J. Abelian, S. Abeyruwan, M. Ahn, A. Bewley, J. Boyd, K. Choromanski, O. Cortes, E. Coumans, T. Ding, et al. Robotic table tennis: A case study into a high speed learning system. *arXiv preprint arXiv:2309.03315*, 2023.

M. Hu, W. Chen, W. Li, F. Mandali, Z. He, R. Zhang, P. Krisna, K. Christian, L. Benaharon, D. Ma, K. Ramani, and Y. Gu. Towards versatile humanoid table tennis: Unified reinforcement learning with prediction augmentation. *arXiv preprint arXiv:2509.21690*, 2025.

V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, and G. State. Isaac gym: High performance gpu-based physics simulation for robot learning. *arXiv preprint arXiv:2108.10470*, 2021.

M. Mittal et al. Isaac lab: A gpu-accelerated simulation framework for robot learning. *arXiv preprint arXiv:2511.04831*, 2025.

K. Mülling, J. Kober, O. Kroemer, and J. Peters. Learning to select and generalize striking movements in robot table tennis. *The International Journal of Robotics Research*, 32(3):263–279, 2013.

A. Y. Ng, D. Harada, and S. Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Proceedings of the Sixteenth International Conference on Machine Learning*, pages 278–287, 1999.

J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

C. Schwarke, M. Mittal, N. Rudin, D. Hoeller, and M. Hutter. Rsl-rl: A learning library for robotics research. *arXiv preprint arXiv:2509.10771*, 2025.