Robotics: Science and Systems 2026
Sydney, Australia, July 13–17, 2026

KlaskTron: An Open-Source Platform for Physical Adversarial Multi-Agent RL

Aswin Ramachandran, Jona Schulz, Maurus Derungs,
Carlo Angelini, Tobias Meier, Raffaello D'Andrea
Institute for Dynamic Systems and Control (IDSC), ETH Zürich, Switzerland
Email: aramachandra@ethz.ch

***Abstract*—Progress in robot learning, particularly physical multi-agent reinforcement learning (MARL), is currently challenged by a limited availability of accessible, standardized benchmarks. While simulation-based MARL has produced remarkable emergent behaviors from coordinated team play to complex tool use, translating these advances to physical systems remains difficult. A key barrier is infrastructural: physical platforms are often prohibitively expensive, require specialized facilities, or lack open mechanisms and designs for reproducibility. We introduce KlaskTron, an open-source, low-cost (<$2500 USD), desktop-scale robotic testbed for adversarial MARL based on the dynamic tabletop game KLASK. The platform features dual CoreXY gantries with high-torque brushless motors, enabling the high-speed, precise manipulation required for competitive play. Crucially, we provide a complete ecosystem: hardware designs (CAD, BOM), a GPU-native digital twin in NVIDIA Isaac Lab for high-fidelity modeling and simulation, and a validated sim-to-real baseline using a learned neural actuator model. We demonstrate that this baseline enables zero-shot policy transfer, with trained agents exhibiting emergent adversarial behaviors. To ensure full reproducibility of our results, all project assets are released publicly: hardware CAD/BOM and assembly documentation at https://idscethzurich.github.io/klask_hardware/ (https://github.com/IDSCETHZurich/klask_hardware), inference software stack at https://github.com/IDSCETHZurich/klask_software and the simulation and MARL setup at https://github.com/IDSCETHZurich/klask_rl.**

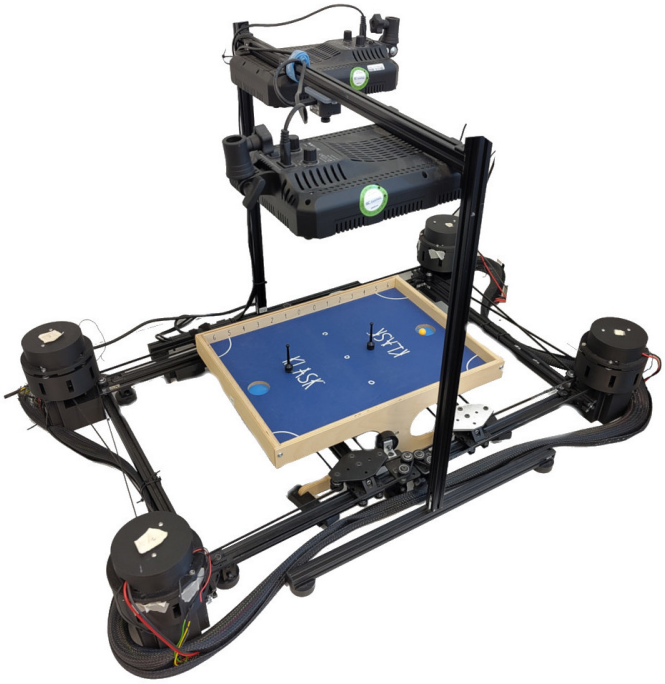

Fig. 1. KlaskTron robot with the 2x 2D gantry for each player

I. Introduction

Robot Learning and Multi-agent reinforcement learning (MARL) have achieved remarkable milestones in digital domains. From the superhuman coordination of AlphaGo [1], OpenAI Five in Dota 2 [2] and AlphaStar in StarCraft II [3], and foundational algorithms like MADDPG [4], agents have demonstrated that sophisticated long-horizon behaviors can emerge from simple reward signals given sufficient competitive pressure. These digital successes suggest a promising future for embodied intelligence. While these principles are well-established in digital environments, a critical challenge remains: the transition from abstract strategic reasoning to the high-stakes constraints of high-speed physical multi-robot play. A key barrier to this transition is infrastructural. Bridging the reality gap requires a benchmark that balances high-fidelity simulation with accessible hardware. While software environments enable millions of interactions per hour, physical MARL is often constrained by a lack of accessible high-fidelity benchmarks. Platforms in this domain are frequently either prohibitively expensive industrial systems [5] or limited to single-agent tasks.

We introduce **KlaskTron** (Fig. 1), a platform designed to bridge this gap. KlaskTron is an open-source, low-cost (<$2500 USD), desktop-scale robotic testbed explicitly built for physical adversarial MARL. It occupies a unique niche: more dynamic than static manipulation benchmarks, more accessible than industrial robotics, and designed for two-agent competitive interaction.

Why KLASK? KLASK instantiates a problem class we term *high-speed, contact-rich adversarial MARL*: a zero-sum, continuous-control task in which two agents must simultaneously attack, defend, and cope with explosive contact dynamics under long-horizon credit assignment. Unlike robotic air hockey or table tennis, KLASK requires no compressed-air infrastructure and fits on a standard desk, enabling fully automatic episode resets and continuous physical training without human intervention. Each ∼31-second episode contains on average 12.4 dynamic contacts (defensive blocks plus offensive strikes).With more than one million units sold

worldwide, the rules are immediately intuitive to a broad audience, lowering the barrier to demonstration and community engagement.

Our contributions are:

1) **Mechanisms & Designs:** We provide complete CAD files, bill of materials (BOM), and assembly instructions for a dual-gantry robotic system costing under $2500 USD with an estimated build time of 20 hours.
2) **Modeling and Simulation:** A GPU-native digital twin component for NVIDIA Isaac Lab enabling massively parallel training, capable of processing 20M steps in $\sim$12 minutes on a single RTX 4090.
3) **A Carefully Engineered Sim-to-Real Baseline:** A reproducible PPO-based reference agent, comprising shaped-reward design and a learned neural actuator model, that achieves zero-shot transfer to hardware. We release trained weights and the full training pipeline. Out-of-the-box reinforcement learning approaches required substantial reward shaping and replay-buffer engineering to produce competent policies; the released baseline reflects this engineering effort and is offered as a reproducible starting point rather than a definitive algorithmic claim.
4) **Full Open-Source Release:** All hardware designs, simulation code, and training pipelines are publicly available to serve as a community-driven research ecosystem.

By providing this complete ecosystem, we establish KlaskTron as a challenging and accessible testbed designed to foster progress at the intersection of MARL, robotics, and the sim-to-real problem.

II. Related Work

We contribute to the achievements of state-of-the-art robotic games with KlaskTron, an accessible platform that complements current systems and facilitates physical adversarial learning.

A. Collaborative and Fleet Benchmarks

Recent initiatives have standardized physical benchmarks for multi-robot coordination. M4Bench [6] establishes a framework for Human-Robot Interaction (HRI) manipulation, while FORMIGA [7] addresses fleet management in field robotics. Similarly, RoboVerse [8] proposes a unified platform for scalable robot learning. Digital benchmarks like PettingZoo [9] and VMAS [10] have standardized soft-body and vector simulations. While these platforms advance cooperative logistics, they do not address the high-frequency, contact-rich adversarial dynamics fundamental to competitive MARL.

B. Single-Agent Dynamic Platforms

Advances in dynamic physical skill learning have largely focused on single-agent settings. The *CyberRunner* benchmark [11] showcased the potential for low-cost, open-hardware platforms to facilitate physical reinforcement learning for dynamic tasks. While it provides a structured framework for evaluating single-agent performance, it is inherently limited by its design to individual tasks.

At the high-performance end of the spectrum, D'Ambrosio et al. [5, 12] established the gold standard for high-speed actuation with their robotic table tennis system, demonstrating human-level rallying. However, such systems rely on industrial-grade hardware inaccessible to most labs. KlaskTron aims to provide a reference baseline and an accessible platform to this class of problems, leveraging the GPU-parallelized simulation paradigm championed by recent benchmarks like ManiSkill3 [13] to enable rapid policy iteration on accessible hardware.

C. Adversarial Tabletop Games

Robotic game-playing has a rich history, from early foosball systems [14] to autonomous pool [15], with early adversarial multi-robot competitions such as RoboFlag [16] laying conceptual groundwork for distributed control under competitive constraints. RoboCup [17] remains the premier event for physical multi-agent sports. The closest functional analog to our work is Robot Air Hockey [18], as surveyed by general game-playing reviews. Recent work by Jankowski et al. [19] has advanced this domain by distilling offline optimal control into reactive policies for high-speed contact planning. KlaskTron complements this control-first literature by providing a learning-first testbed specifically designed for adversarial MARL. This is critical for studying emerging phenomena such as coordinated adversarial attacks [20] and safe multi-agent control [21], which require reproducible physical environments to validate theoretical robustness guarantees.

D. Sim-to-Real Methodology

The reality gap remains a primary bottleneck for physical reinforcement learning [22], as surveyed by Zhao et al. [23]. To demonstrate the feasibility of the KlaskTron platform, we provide a baseline implementation using established techniques in Domain Randomization and System Identification. Crucially, we adopt the approach of learning a neural actuator model to capture complex motor dynamics (friction, latency) that analytical models like MuJoCo [24] miss. This methodology was recently validated by Fey et al. [25] demonstrated that unsupervised actuator networks are essential for bridging the sim-to-real gap in athletic loco-manipulation. KlaskTron serves as an accessible testbed where such emerging sim-to-real methodologies can be benchmarked under the unique distribution shifts presented by high-speed adversarial interaction.

III. The KlaskTron Platform

This section describes the KlaskTron hardware, software ecosystem, and the design principles that guided its development.

A. Design Philosophy

KlaskTron was designed according to three principles for accessibility and reproducibility, following best practices from recent open-source robotics platforms [11, 26]:

1) **Accessibility:** Total cost under $2500 USD using off-the-shelf components and 3D-printed parts.

2) **Reproducibility:** Complete open-source release of all designs, with detailed assembly instructions.
3) **Research Utility:** Sufficient dynamics complexity to serve as a meaningful benchmark for emergent MARL behaviors. KlaskTron isolates specific open problems in physical RL: high-speed contact dynamics, non-linear reward landscapes, and adversarial intent inference.

B. Hardware Architecture

The KlaskTron hardware (Fig. 1) consists of:

Dual CoreXY Gantries: Two identical, opposing CoreXY gantry systems are mounted beneath a standard KLASK game board. The CoreXY architecture is ideal for this application because it keeps the motors stationary, minimizing the inertia of moving components and enabling accelerations (mean 6.25 m/s^2) sufficient to traverse the playing field in under a second, matching the pace of human reaction.

Direct-Drive Brushless DC Motors: The system uses direct-drive brushless DC motors controlled by ODrive S1 drives, providing the torque and acceleration necessary for competitive, human-like play. Integrated encoders enable closed-loop velocity control with a measured tracking RMSE of 0.013 m/s. We initially evaluated NEMA 17 and sensored stepper motors but observed unavoidable stalls, jerk, and slip that prevented the explosive movements competitive play demands; after comparing several BLDC options we selected the ODrive S1 stack for its reliable sensored control and tuning flexibility, which together unlock the full skill range required by the task.

Magnetic Coupling: Following the original KLASK game design, strikers on the board surface are magnetically coupled to carriages beneath the board, enabling non-contact actuation.

Overhead Camera: A single camera operating at 120 Hz provides observations of the game state. This sensing rate was selected to balance cost-effectiveness with the performance required for high-speed interaction. By providing observations at this frequency (50 Hz), the platform requires agents to account for system latency, which encourages the development of robust predictive representations.

Communication: Low-level motor control is handled via direct CAN bus connection, with a ROS node converting high-level velocity commands to motor controller messages.

C. GPU-Native Digital Twin

A critical component of the KlaskTron ecosystem is the **digital twin** implemented in NVIDIA Isaac Lab. This GPU-native simulation environment enables:

- **Massively Parallel Training:** Thousands of independent KLASK environments run simultaneously on a single GPU.
- **Rapid Iteration:** 20 million PPO steps complete in approximately 12 minutes on an RTX 4090.
- **High-Fidelity Physics:** The PhysX engine models board geometry, contact dynamics, and friction with tunable parameters.

The simulation (Fig. 3) models the board, walls, strikers, and ball using primitive shapes. For the baseline experiments, we simplify the game by omitting the magnetic obstacle pieces (biscuits) present in the commercial game.

IV. Methodology: Baseline Learning and Transfer

To establish the utility of KlaskTron as a research platform, we provide a reference sim-to-real baseline. The primary objective of this methodology is to serve as a reference baseline, verifying that the physical environment's complexity is manageable within current reinforcement learning and system identification paradigms. Rather than defining the platform's limits, these methods provide a validated foundation from which the community can build, iterate, and compare novel algorithmic approaches.

The following subsections describe the state estimation, actuator modeling, and training protocols used to achieve initial zero-shot policy transfer.

A. Vision-Based State Estimation

On the physical system, the game state is estimated from camera images using an Extended Kalman Filter (EKF) [27]. The state vector for each tracked object is defined as $\mathbf{x} = [x, y, \dot{x}, \dot{y}]^T$. To handle the high-speed dynamics and sudden transitions during play, we utilize an **adaptive process noise** mechanism, a technique established for the identification of variances in filtering systems [28].

As illustrated in Fig. 4, the process noise covariance Q_k is adjusted dynamically based on the ball's predicted position. In the open areas of the board Q_k is small, reflecting high confidence in the physics-based transition model. Near the Boundaries and Strikers regions Q_k increases significantly due to collisions, causing the filter to rely more heavily on real-time measurements to account for non-linear collision dynamics. This enables rapid convergence to post-collision trajectories.

B. The Sim-to-Real Challenge

Training directly on hardware is impractical due to the sample inefficiency of reinforcement learning, necessitating simulation-based training. The platform's high-speed dynamics expose the reality gap in ways that static or slower manipulation tasks do not: idealized Proportional–Derivative controllers, even when augmented with standard domain randomization, produce control signals that the hardware cannot faithfully execute. The system must account for non-linearities such as friction, the torque–speed envelope of the brushless motors, and the magnetic coupling between the under-board carriage and the surface peg, together with the transient delays inherent in a 50 Hz control loop and the adversarial distribution shifts induced by an unpredictable opponent. By providing a testbed where idealized simulation fails, KlaskTron offers a rigorous environment for validating sophisticated transfer mechanisms such as the learned neural actuator model we describe next.

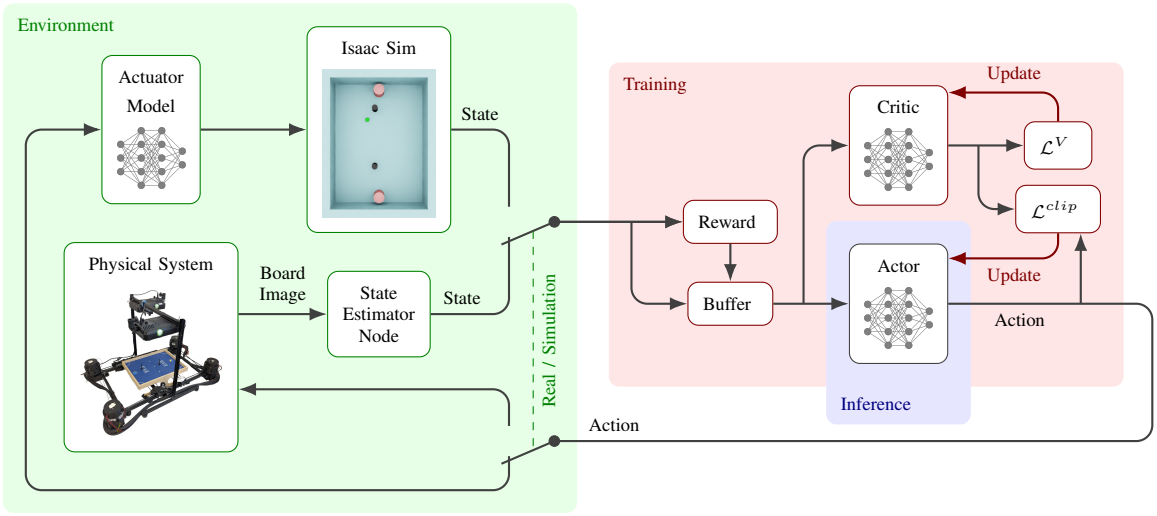

Fig. 2. System architecture of the KlaskTron platform. The environment is interchangeable between the Isaac Lab simulation (for training) and the physical system (for deployment). State estimation in simulation is provided directly by the physics engine; on hardware, an Extended Kalman Filter processes camera observations. The control stack uses PPO during training and the frozen actor network during inference. Actions pass through the learned actuator model in simulation or directly to physical motors on hardware.

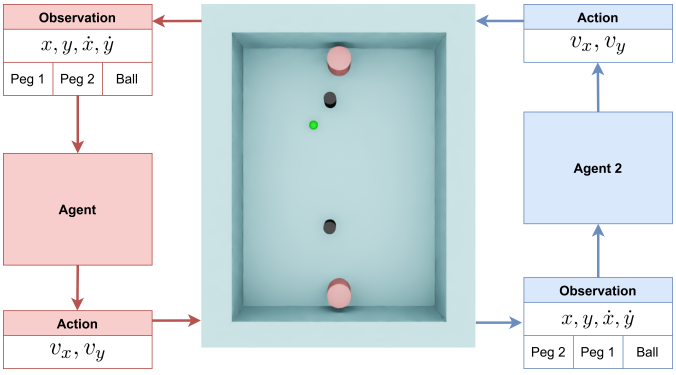

Fig. 3. The Isaac Lab simulation environment for KlaskTron. Each agent observes ball and striker positions/velocities and outputs continuous velocity commands.

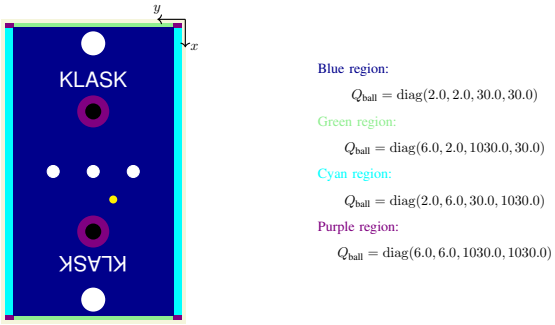

Fig. 4. Adaptive process noise regions on the KLASK board (left) with corresponding covariance matrices (right). Higher values in Q matrices indicate reduced confidence in the process model, causing the EKF to rely more on measurements.

C. Learned Neural Actuator Model

Following prior work on learned actuator dynamics [29], we address the sim-to-real gap by replacing the idealized PD actuator in simulation with a learned model that captures real motor behavior. **Data Collection:** We collected 37,500 time-series samples ($\sim$12.5 minutes at 50 Hz) from the physical system, executing diverse trajectories (circles, diagonals, linear sweeps) with commanded velocities up to ± 0.7 m/s. **Model Architecture:** A three-hidden-layer MLP (64 units each) predicts the next velocity from a history of commands and velocities:

$$v_{t+1} = f_{NN}(v_t, a_{t-1}, u_{t-1}, \dots)$$

The architecture and history length were selected by a trajectory-based ablation across $\{1, 5, 10\}$-step history and $\{32, 64, 128\}$-unit widths trained over 8 random seeds; the 10-step / 3×64 configuration achieved the lowest velocity-prediction RMSE while remaining compact, and is the model used for all results reported in this paper.

Deconstructing the Sim-to-Real Gap: The 10-timestep input history enables the model to implicitly learn system latency and transient response, performing action interpolation without explicit time-delay buffers. By training on trajectories with velocities up to ± 0.7 m/s, the network learned the physical torque–speed envelope, identifying saturation points where commanded accelerations are physically unachievable. We established this model as the primary transfer mechanism by demonstrating successful zero-shot deployment with Domain Randomization disabled, whereas policies relying on Domain Randomization with idealized physics failed to transfer.

Figure 5 validates the model: simulated velocities closely track real system response across diverse trajectories, and the seed-band collapses tightly around the mean prediction.

D. Training Protocol

Out-of-the-box reinforcement learning approaches required substantial reward shaping and replay-buffer engineering to

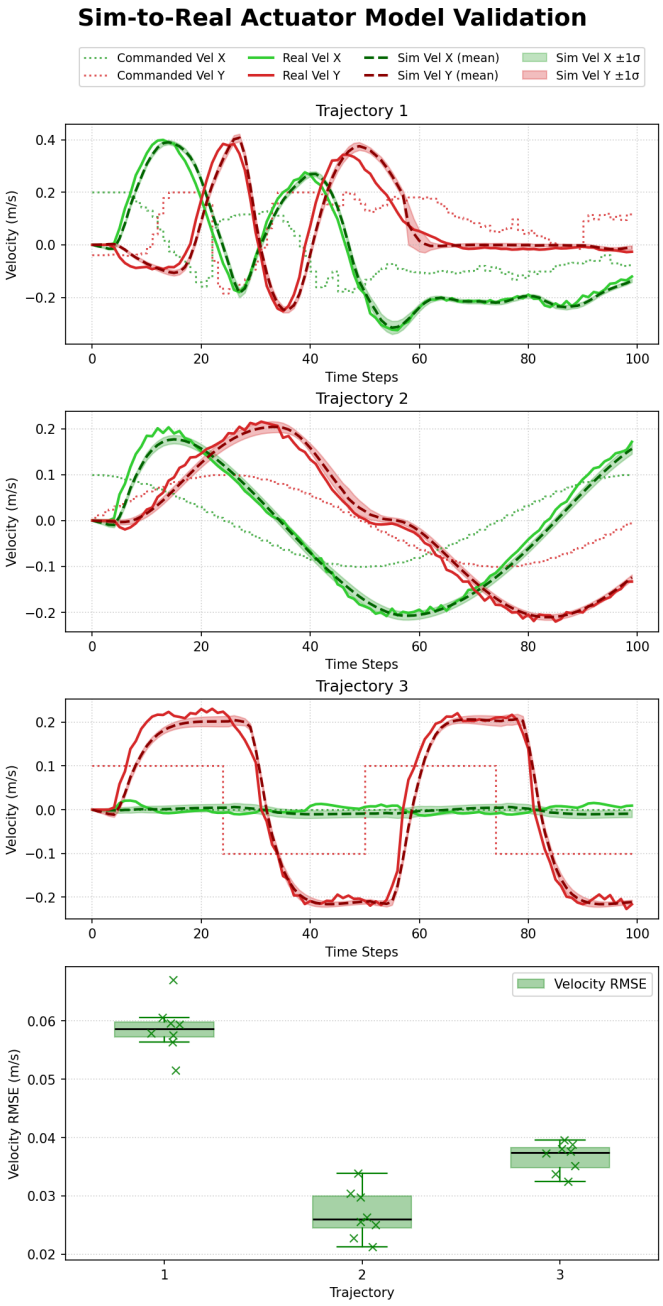

Fig. 5. Validation of the learned actuator model (10-step history, three hidden layers of 64 units). Solid lines show real hardware response; dashed lines show simulated response to identical command sequences. Shaded bands show the spread across 8 training seeds, indicating that prediction quality is robust to initialisation. The model achieves a global validation RMSE of 0.013 m/s, with peak errors of 0.032 m/s (x) and 0.049 m/s (y) on these representative high-frequency trajectories.

produce competent policies on this task; the PPO baseline described below reflects this engineering effort and is offered as a reproducible starting point rather than a definitive algorithmic claim. We summarise the components here; full reward formulae, curriculum schedules, and self-play hyperparameters are provided in Appendix A.

Reward and Curriculum. Sparse game-outcome rewards alone yielded passive policies. We therefore use a time-decaying curriculum that initially guides exploration with dense shaping signals (offensive progress, defensive positioning, ball analysis) and transitions to sparse adversarial incentives (goals scored, conceded, and self-goal penalties).

Self-Play. Adversarial competence is developed via Enhanced Self-Play [2]: the agent trains against a pool of past checkpoints sampled with geometric decay favouring recent policies.

1) MDP Formulation: The KLASK game is modeled as a two-player zero-sum Markov Decision Process (MDP) defined by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{P}, \gamma)$.

State Space ($\mathcal{S}$): The observation vector consists of two components:

- *Standard Observations:* Positions and velocities of the player's striker, the opponent's striker, and the ball.
- *Augmented Features:* Relative angle between the striker and the goal, Euclidean distance from the ball to the goals, and distance from the striker to the ball.

Action Space ($\mathcal{A}$): Each agent outputs continuous planar velocity commands:

$$a_t = (v_x, v_y) \in [-0.4, 0.4]^2 \text{ m/s}$$

Objective: Each agent maximizes expected discounted rewards:

$$\pi^* = \arg\max_{\pi} \mathbb{E}_{\pi} \left[\sum_{t=0}^{\infty} \gamma^t r_t \right] \tag{1}$$

where γ is the discount factor (see Appendix A4).

E. Evaluation Metrics

We report agent quality through three complementary metrics standardised in our benchmark protocol: *scoring rate* (goals per minute of physical play), *episode length* (seconds per goal), and *contact frequency* (defensive blocks plus offensive strikes per episode). For agent-versus-agent comparisons in simulation we additionally use the Elo rating system [2] with a K-factor of 10 and baseline rating of 1200. Detailed Elo derivations and round-robin curricula results are provided in Appendix A.

V. Experiments: Validating the Testbed

We present experiments demonstrating that KlaskTron functions as a reproducible physical benchmark: simulated policies transfer zero-shot to hardware, the platform is reliable across extended evaluation, and the residual gap between agent and human play is well characterised. Supporting algorithmic analyses (curriculum comparison, self-play variants, observation augmentation) are reported in Appendix A.

A. Sim-to-Real Transfer

The utility of the KlaskTron benchmark depends on the feasibility of transferring simulated policies to physical hardware. Our experiments identify a clear dependency between actuator modelling and transfer success.

Actuator Modelling Constraints. Initial attempts using idealised Proportional–Derivative controllers with domain

randomization failed to transfer: the discrepancy between simulated and physical command response produced control signals the hardware could not execute, leading to immediate policy failure.

System Identification and Transfer. Incorporating the learned actuator model (Section IV-C, Fig. 5) into the training pipeline enables successful zero-shot transfer. While domain randomization is a common sim-to-real heuristic, our high-fidelity actuator model allowed transfer even with DR disabled, suggesting that for KlaskTron precise system identification is a more critical factor than broad parameter randomization.

Emergent Behaviours. Transferred policies exhibited differentiated offense/defense (autonomous switching between defensive positioning and offensive strikes), goal-directed play (consistent scoring against stationary and random opponents), and reactive adaptation to ball trajectories and opponent movements.

B. Hardware Failure Modes

The residual 0.5–1.5 cm offensive misses observed in human-robot play are policy and perception artefacts rather than hardware failures. The dominant physical effect is the magnetic coupling between the under-board carriage and the surface peg, which lags by approximately 0.4 cm during direction changes; combined with gantry damping and EKF position uncertainty this produces a non-trivial sub-centimetre offset that the policy must learn to anticipate. Quantitatively, decelerating the peg from its peak velocity of 0.5 m/s to standstill consumes 77 ms and 2 cm, while accelerating it back to the same speed in the opposite direction takes 111 ms and 7 cm. Kinematic constraints intrinsic to the magnetically coupled actuator that even human players experience and that the agent must internalise rather than overcome.

C. System Reliability and Repeatability

Hardware reliability. Across evaluations totalling more than 30 consecutive games we observed no hardware-level failure, where hardware failure is defined as motor connection error, camera dropout, EKF divergence, peg decoupling, or ball departure from the board. Software-level resets occurred and were logged but did not corrupt the protocol.

Software reproducibility. Two students independently retrained the baseline policy on different days using the released training pipeline; both reached comparable competitive levels in physical play, providing initial evidence that the released hardware–software stack constitutes a repeatable substrate for RL evaluation.

Sources of stochasticity. We deliberately distinguish *system reliability* (faithful command execution; velocity-tracking RMSE 0.013 m/s) from *system stochasticity* (magnetic coupling slip and minor mechanical variation) and *policy stochasticity* (action sampling noise). Natural system stochasticity is preserved by design, since the research question is whether learned policies cope with real-world randomness, not eliminate it. Our released benchmark protocol formalises scoring rate, episode length, and contact frequency as the primary reporting metrics.

D. Human-Robot Benchmark Validation

To evaluate the platform's potential as a high-ceiling benchmark we tested the baseline policy against a recreational human opponent over a 10-game series (Table I). The agent achieved a 10% win rate; each $\sim$31-second episode contained on average 12.4 dynamic contacts (defensive blocks plus offensive strikes), confirming that physical adversarial play emerges in the policy. The agent exhibited several emergent strategies, including reactive goal-tending and the use of board geometry for defensive bank shots.

Performance was limited by the offensive action mismatch characterised in Section V-B: the policy often selected the correct velocity vectors but the striker missed the ball by 0.5–1.5 cm. Together with the 31.3 s average episode duration, this confirms substantial research headroom remains on the KlaskTron platform.

TABLE I
10-GAME HUMAN-ROBOT TOURNAMENT STATISTICS. THE AGENT DEMONSTRATES EMERGING DEFENSIVE COMPETENCE BUT SIGNIFICANT OFFENSIVE LIMITATIONS.

Game #	Winner	Duration (s)	Defensive Blocks	Offensive Strikes	Goal-Directed Hits	Possession (%)
1	Human	28	6	4	1	42%
2	Human	35	8	5	2	45%
3	**Robot**	42	12	9	4	51%
4	Human	21	4	3	1	38%
5	Human	30	7	5	2	44%
6	Human	28	6	4	1	43%
7	Human	25	5	3	1	40%
8	Human	32	9	6	2	46%
9	Human	29	6	4	1	43%
10	Human	43	11	7	3	44%
Avg.	*Human (90%)*	*31.3*	*7.4*	*5.0*	*1.8*	*43.6%*

VI. CONCLUSION

We have introduced KlaskTron, an open-source platform that lowers the infrastructural barrier to physical adversarial multi-agent reinforcement learning. The contribution is the platform itself: hardware designs, a GPU-native digital twin, a formalised gameplay protocol with reproducible reliability statistics, and a carefully engineered PPO-based reference baseline released as trained weights and a complete training pipeline. The accompanying experiments characterise the platform rather than advance an algorithmic claim. They quantify hardware reliability over 30^{+} consecutive games, attribute the residual sub-centimetre offensive miss to physical effects the policy must learn (Section V-B), and isolate the actuator-modelling components most critical for zero-shot transfer.

Future work will extend the platform along two axes: increased task complexity through reintroduction of the magnetic biscuit obstacles, yielding a non-convex landscape that demands multi-dimensional strategy, and natural extension to four-agent play via the commercially available round-Klask variant, opening team-based MARL questions on the same hardware. Beyond these, we aim to move past hand-engineered features by exploring autonomous representation learning and to integrate game-theoretic approaches that approximate Nash equilibria in competitive physical environments. Uniquely positioned

between static manipulation tasks and industrial-grade systems, KlaskTron is intended as a community substrate on which future algorithmic work in physical MARL can be developed and compared.

Acknowledgments

The authors thank the IDSC technical staff Dani Wagner and Matthias Mueller for their constant support and help in developing the hardware, and Thomas Bi for his support and consultation on various topics regarding reinforcement learning and hardware development.

References

[1] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot *et al.*, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[2] C. Berner, G. Brockman, B. Chan, V. Cheung, P. Dębiak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme *et al.*, "Dota 2 with large scale deep reinforcement learning," *arXiv preprint arXiv:1912.06680*, 2019.

[3] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev *et al.*, "Grandmaster level in starcraft ii using multi-agent reinforcement learning," *Nature*, vol. 575, no. 7782, pp. 350–354, 2019.

[4] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 30, 2017.

[5] D. B. DAmbrosio, S. Abeyruwan, L. Graesser, A. Iscen, H. B. Amor, A. Bewley, B. J. Reed, K. Reymann, L. Takayama, Y. Tassa *et al.*, "Achieving human level competitive robot table tennis," in *2025 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2025, pp. 74–82.

[6] K. Yoshida, A. Aydemir, A. Andriella, A. Angleraud, R. Pieters, and V. Hautamaki, "A multi-user multi-robot multi-goal multi-device human-robot interaction manipulation benchmark," *Frontiers in Robotics and AI*, vol. 12, p. 1528754, 2025.

[7] B. Yalcinkaya, M. S. Couceiro, S. Soares, and A. Valente, "Formiga: a fleet management framework for sustainable human–robot collaboration in field robotics," *Frontiers in Robotics and AI*, vol. 12, p. 1706910, 2025.

[8] H. Geng, F. Wang, S. Wei, Y. Li, B. Wang, B. An, C. T. Cheng, H. Lou, P. Li, Y.-J. Wang *et al.*, "Roboverse: Towards a unified platform, dataset and benchmark for scalable and generalizable robot learning," *arXiv preprint arXiv:2504.18904*, 2025.

[9] J. Terry, B. Black, N. Grammel, M. Jayakumar, A. Hari, R. Sullivan, L. Santos, C. Dieffendahl, C. Horsch, R. Perez-Vicente *et al.*, "PettingZoo: Gymnasium for multi-agent reinforcement learning," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 34, 2021, pp. 15 032–15 043, datasets and Benchmarks Track.

[10] K. Li, A. Gupta, P. Morales, and R. Allen, "VMAS: A vectorized multi-agent simulator," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 35, 2022, pp. 25 042–25 055, datasets and Benchmarks Track.

[11] T. Bi and R. D'Andrea, "Sample-efficient learning to solve a real-world labyrinth game using data-augmented model-based reinforcement learning," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 7455–7460.

[12] 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.

[13] S. Tao, F. Xiang, A. Shukla, Y. Qin, X. Hinrichsen, X. Yuan, C. Bao, X. Lin, Y. Liu, T.-K. Chan *et al.*, "Maniskill3: Gpu parallelized robot simulation and rendering for generalizable embodied ai," in *7th Robot Learning Workshop: Towards Robots with Human-Level Abilities*, 2025.

[14] T. Weigel, J.-S. Gutmann, and B. Nebel, "KiRo – a table soccer robot ready for the market," in *RoboCup 2005: Robot Soccer World Cup IX*, ser. Lecture Notes in Computer Science, vol. 4020. Springer, 2006, pp. 468–475.

[15] T. Nierhoff, K. Leibrandt, T. Lorenz, and S. Hirche, "Robotic billiards: Understanding humans in order to counter them," *IEEE Transactions on Cybernetics*, vol. 46, no. 8, pp. 1889–1899, 2016.

[16] R. D'Andrea and R. Murray, "The roboflag competition," in *Proceedings of the 2003 American Control Conference, 2003.*, vol. 1, 2003, pp. 650–655 vol.1.

[17] C. Buche, A. Rossi, M. Simões, and U. Visser, Eds., *RoboCup 2023: Robot World Cup XXVI*, ser. Lecture Notes in Computer Science. Cham: Springer Nature Switzerland, 2024, vol. 14140, symposium/Competition Report.

[18] X. Liu, P. Liu, W. Wan, and K. Harada, "Robot reinforcement learning on the constraint manifold," in *Conference on Robot Learning (CoRL)*. PMLR, 2022, pp. 1357–1366.

[19] J. Jankowski, A. Marić, P. Liu, D. Tateo, J. Peters, and S. Calinon, "Distilling contact planning for fast trajectory optimization in robot air hockey," in *Robotics: science and systems: online proceedings*, 2025.

[20] S. Lee, J. Hwang, Y. Jo, and S. Han, "Wolfpack adversarial attack for robust multi-agent reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2025, pp. 33 025–33 056.

[21] S. Zhang, O. So, M. Black, Z. Serlin, and C. Fan, "Solving multi-agent safe optimal control with distributed epigraph form marl," *arXiv preprint arXiv:2504.15425*, 2025.

[22] Y. Chebotar, A. Handa, V. Makoviychuk, M. Macklin, J. Issac, N. Ratliff, and D. Fox, "Closing the sim-to-real

loop: Adapting simulation randomization with real world experience," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 8973–8979.

[23] W. Zhao, J. P. Queralta, and T. Westerlund, "Sim-to-real transfer in deep reinforcement learning for robotics: a survey," in *2020 IEEE symposium series on computational intelligence (SSCI)*. IEEE, 2020, pp. 737–744.

[24] 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 (IROS)*. IEEE, 2012, pp. 5026–5033.

[25] N. Fey, G. B. Margolis, M. Peticco, and P. Agrawal, "Bridging the sim-to-real gap for athletic loco-manipulation," in *7th Robot Learning Workshop: Towards Robots with Human-Level Abilities*, 2025.

[26] J. Moos, C. Derstroff, N. Schröder, and D. Clever, "Learning to play foosball: System and baselines," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 8945–8951.

[27] S. J. Julier and J. K. Uhlmann, "New extension of the kalman filter to nonlinear systems," in *Signal processing, sensor fusion, and target recognition VI*, vol. 3068. SPIE, 1997, pp. 182–193.

[28] R. Mehra, "On the identification of variances and adaptive kalman filtering," *IEEE Transactions on Automatic Control*, vol. 15, no. 2, pp. 175–184, 1970.

[29] J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, and M. Hutter, "Learning agile and dynamic motor skills for legged robots," *Science Robotics*, vol. 4, no. 26, p. eaau5872, 2019.

APPENDIX

A. Training Protocol: Reward, Curriculum, Self-Play, Observations

1) Reward Curriculum: The total reward at time t combines a sparse termination signal with time-decaying shaping rewards:

$$r_{total}(t) = r_{term} + \sum_i R_i \cdot \exp\left(-\frac{t}{N}\right) \cdot r_{shape,i} \tag{2}$$

where N is the decay time constant.

Termination (r_{term}): Goal Scored ($+1$), Goal Conceded (-1), Agent Self-Goal Error (-1).

Shaping (r_{shape}):

- *Offensive Progress:* $\exp(-\|p_B - p_O\|/D)$ (ball to opponent goal)
- *Defensive Positioning:* $-\exp(-\|p_B - p_P\|/D)$ (ball to own goal)
- *Ball Analysis:* $\exp(-\|p_{PS} - p_B\|/C)$ (proximity) and $\|v_B\|$ (speed)

The decay schedule encourages meaningful interaction during initial training while progressively removing the training wheels to prevent reward hacking. Sparse-only reward variants yielded passive policies in our preliminary experiments; the curriculum was selected from a comparison of three variants (collision-only, dense ball-to-goal, sparse-only), with the dense ball-to-goal curriculum reaching the highest Elo in round-robin tournaments (Elo ≈ 1283 vs. ≈ 1240 for sparse and ≈ 1220 for collision-only).

2) Self-Play Configuration: Adversarial competence is developed via Enhanced Self-Play [2]: the agent trains against a pool of past checkpoints sampled with geometric decay, $w_i = 0.9^{(N-i-1)}$, where i indexes checkpoints and $N-1$ is the most recent. The opponent pool updates when the average score difference exceeds 0.7. We compared this against random pool sampling and best-opponent-only sampling: the best-opponent variant proved unstable due to insufficient opponent diversity, while Enhanced Self-Play and random pool sampling produced comparable competent policies.

3) Observation Augmentation: Following findings in large-scale game domains [2], we provide hand-engineered features (striker-to-ball vector, angle to goal) alongside the raw state vector. Augmented observations accelerate long-term convergence relative to raw-state baselines.

Parameter	Value
Discount factor (γ)	0.99
GAE parameter (λ)	0.95
Learning rate	3×10^{-4}
Clip parameter (ϵ)	0.2
Mini-batch size	4096
Entropy coefficient	0.01

TABLE II
PPO HYPERPARAMETERS.

4) PPO Hyperparameters:

B. Domain Randomization

For the best-performing transferred policy, domain randomization was **disabled**. The following ranges were used in ablation studies:

Parameter	Range
Static friction	$[0.2, 0.5]$
Dynamic friction	$[0.3, 0.45]$
Restitution	$[0.95, 0.99]$
Ball mass	$[1.8, 2.2]$ mg

TABLE III
DOMAIN RANDOMIZATION RANGES (DISABLED FOR BEST AGENT).

C. Actuator Model Details

1) Data Collection Protocol: Training data was collected by executing scripted trajectories:

- Circular paths (varying radii and speeds)
- Diagonal sweeps
- Linear velocity ramps
- Random walk sequences

Total: 69,669 samples at 50 Hz ($\sim$23.2 minutes). Commanded velocities ranged ± 0.7 m/s.

2) Model Architecture:

- Input: History of 10 timesteps (commands and velocities)
- Hidden layers: 3 $\times$ 64 units, ReLU activation
- Output: Predicted velocity $\hat{v}_{t+1}$
- Training: MSE loss, Adam optimizer, 1000 epochs

Validation RMSE on held-out trajectories: 0.013 m/s (global average). The architecture and history length were selected by an ablation over $\{1, 5, 10\}$-step history $\times$ $\{32, 64, 128\}$-unit widths trained over 8 random seeds; the 10-step / 3×64 configuration achieved the lowest combined velocity-prediction RMSE while remaining compact.