# End-to-End Autonomous Soccer Shooting of Humanoid Robots via Reinforcement Learning and Spatial Adaptive Training

Luoyun Wang[1], Linqi Ye[2], *Member, IEEE*

***Abstract*— Autonomous shooting at random ball positions is a challenging task for humanoid robots, requiring tight coordination of locomotion, ball positioning, heading alignment, and contact-rich motion control. This paper develops an end-to-end autonomous soccer shooting system for the Unitree R1 humanoid robot deployed in the Unity simulation environment. The system adopts a reinforcement learning (RL) policy that maps a 59-dimensional observation space to 12-dimensional leg joint actions for unified shooting control. To address the spatially uneven shooting success rate across different ball positions, we propose a multi-stage optimization pipeline consisting of full-region random training, weak-region performance diagnosis, targeted incremental training, and global re-evaluation. Evaluated on a paired dataset with 200 identical ball positions, the overall shooting success rate improves from 65.5% to 72.5%. In particular, the identified weak region achieves a substantial performance boost, with the success rate rising from 48.1% to 76.9%. Further system-level comparisons across 40 trials at eight fixed ball positions show that the proposed end-to-end RL system scores 33 goals, whereas a traditional rule-driven baseline only scores 18 goals. Experimental results validate the effectiveness of our method in realizing robust wide-area autonomous shooting and remedying regional performance weaknesses of humanoid soccer robots.**

A video demo is available at the following link: https://linqi-ye.github.io/gewu/videos/r1soccer.mp4

## I. INTRODUCTION

Humanoid robot soccer is a sophisticated composite task that tightly integrates bipedal locomotion, contact-rich motion control, target localization, and sequential decision-making. To perform a successful shooting motion, the robot needs to stably approach the ball while maintaining balance, achieve precise behind-ball pose alignment, and generate transient and effective foot-ball contact to guide the ball toward the goal. Different from isolated bipedal walking or fixed-position kicking, shooting from randomly distributed ball positions imposes more stringent requirements on whole-body dynamic stability and spatial adaptive capability, making it a challenging problem for humanoid robot control.

Reinforcement learning (RL) has emerged as an effective paradigm for acquiring dexterous bipedal soccer skills [1]. Existing research has advanced humanoid shooting capabilities via multiple technical routes, including motion-guided curriculum learning for high-impulse shooting control [2], teacher–student training for robust continuous kicking under sensing noise [3], and RL-based localization and in-motion kicking strategies for RoboCup scenarios [4] and distance-adaptive kicking tasks [5]. Despite these progresses, most existing studies focus separately on kicking quality optimization, perceptual robustness enhancement, or individual specialized kicking skills. In contrast, this work targets the full closed-loop autonomous shooting pipeline, covering ball approaching, behind-ball pose adjustment, and continuous goal-directed shooting. We systematically investigate the shooting performance of the Unitree R1 humanoid robot over a wide range of static ball distributions and further analyze the inherent spatial failure patterns of the shooting task.

In contact-rich robotic control tasks, incorporating motion priors with task-specific rewards has been proven to efficiently enhance exploration efficiency and training stability [6]. Meanwhile, data-driven legged locomotion learning has been widely validated on both bipedal and quadrupedal robotic platforms [7], [8], [9], [10]. Furthermore, residual reinforcement learning [11] and curriculum learning strategies [12], [13] provide feasible technical references for learning residual corrective actions based on prior motion knowledge and adaptively optimizing training data distribution. Inspired by the above works, our framework adopts a periodic gait prior as the basic motion foundation, learns state-dependent residual leg joint actions to compensate for prior motion deficiencies, and implements targeted continued training in diagnosed weak spatial regions after full-field performance evaluation, so as to mitigate regional performance imbalance.

The main contributions of this paper are summarized as two-fold: (1) A complete autonomous shooting framework tailored for the Unitree R1 humanoid robot, which forms an end-to-end control paradigm. This paradigm fully removes the dependence on manually predefined kicking angles and fixed swing trajectories to support adaptive shooting at random ball positions; (2) A spatial adaptive optimization training pipeline, consisting of full-region random training, weak-region performance diagnosis, local targeted continued training, and global re-evaluation, which effectively alleviates the spatial imbalance of shooting success rate and greatly improves the overall and regional shooting robustness of humanoid robots.

## II. PROBLEM FORMULATION

### A. Task Definition

Let the planar positions of the R1 base, ball, and goal center be $\boldsymbol{p}_r = [x_r, z_r]^{\mathrm{T}}$, $\boldsymbol{p}_b = [x_b, z_b]^{\mathrm{T}}$, and $\boldsymbol{p}_g = [x_g, z_g]^{\mathrm{T}}$. The ball is sampled from the effective region $\Omega$, i.e., the red-shaded rectangle in Fig. 1, with bounds

[1]Luoyun Wang is with the School of Future Technology, Shanghai University, Shanghai, China. 1920693846@qq.com

[2]Linqi Ye is with the School of Future Technology, Shanghai University, Shanghai, China. yelinqi@shu.edu.cn

$X \in [-5.5, 5.5]\,\mathrm{m}$ and $Z \in [-4, 4]\,\mathrm{m}$. At the start of each episode, the robot pose is reset, the ball velocities are zeroed out, and the ball is placed following the training or evaluation protocol.

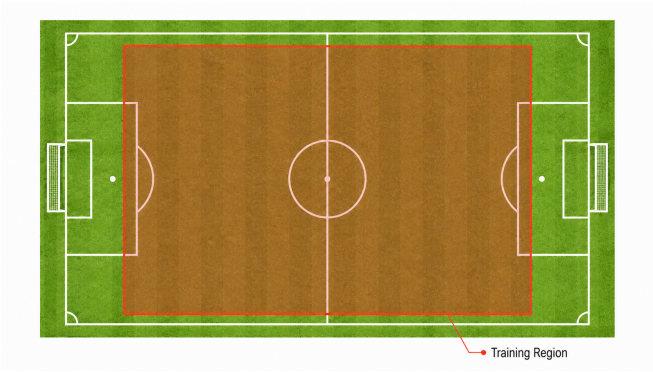



Fig. 1. Unity soccer field and effective ball-sampling region (red rectangle).

The objective is to drive the ball into the goal trigger region before the episode timeout limit, where the ball-to-goal distance threshold serves only as a fallback termination condition. An episode terminates upon scoring a goal, reaching the timeout limit, or triggering a protocol-specific failure.

### B. Markov Decision Process

The task is modeled as $M = (S, A, P, R, \gamma)$, where $S$, $A$, $P$, $R$, and $\gamma$ denote the state, action, transition, reward, and discount factor. At time $t$, policy $\pi_\theta(a_t \mid s_t)$ maps $s_t \in S$ to $a_t \in A$; the environment advances to $s_{t+1}$ and returns $r_t$. We optimize the policy with PPO in ML-Agents [15].

## III. PROPOSED METHOD

### A. Reinforcement Learning Framework

The task layer computes behind-ball targets and state transitions. The policy maps a 59-dimensional observation vector to 12 discrete actions, and the execution layer translates these actions into setpoints for the 12 revolute leg joints via ArticulationBody drives. All non-leg revolute joints maintain their initial reference targets. Physics simulation is handled by Unity, while ML-Agents governs observations, actions, reward signals, and episode lifecycle [14].

A periodic gait prior provides the underlying bilateral rhythmic motion, and the policy outputs state-dependent residual corrections to modulate this baseline motion. Both components jointly drive robot movement during the `MoveToKickPoint` phase. The gait prior is suspended in the `FaceGoal` phase to mitigate motion overshoot, and re-enabled within `KickTrain`, where residuals adapt the whole-body motion according to body configuration and foot-ball interaction states. This architecture follows the general paradigm of learning corrective residuals around a predefined motion prior [11].

### B. State and Observation Space

The 59-dimensional observation space consists of five groups: proprioceptive state, joint state, steering command, soccer task information, and kicking enhancement features

TABLE I

COMPOSITION OF THE 59-DIMENSIONAL OBSERVATION

| **Observation group** | **Contents** | **Dim.** |
|---|---|---|
| Proprioceptive state | Gravity direction, base angular velocity and linear velocity | 9 |
| Joint state | Angles and angular velocities of 12 leg joints | 24 |
| Steering command | Target yaw rate control signal $w_r$, calculated from the yaw error between the current orientation and the target direction | 1 |
| Soccer task information | Local position and distance of the ball (3), local position and distance of the goal (3), local position and distance of the behind-ball point (3), local unit direction from ball to goal (2), direction error to the behind-ball point (1), shooting direction error (1), task state indicators for the unified policy (4) | 17 |
| Kicking enhancement features | Relative position and distance of the right foot with respect to the ball (3), relative position and distance of the left foot with respect to the ball (3), local planar velocity of the ball (2) | 8 |
| **Total** | — | **59** |

(see Table I). All spatial quantities, including positions, orientations, and velocities, are transformed and represented in the robot's local coordinate frame. Specifically, the 4-dimensional task state indicators serve only as auxiliary inputs for the unified policy to characterize the current control context, and do not correspond to independent sub-policies. All observations are fed into a single reinforcement learning policy, which continuously outputs 12-dimensional control actions for the leg joints.

### C. Action Space and Periodic Gait Prior

The policy outputs $a_t = \{a_{t,1}, \ldots, a_{t,12}\}$ for hip pitch/roll/yaw, knee, and ankle pitch/roll on both legs. Exponential smoothing limits inter-step discontinuities:

$$u_t = \alpha u_{t-1} + (1 - \alpha) a_t. \tag{1}$$

We set $\alpha = 0.8$. Smoothed actions are converted to residual targets $q_i^{\mathrm{RL}} = k_i c_t u_{t,i}$ by joint scales $k_i$ and phase scale $c_t$; $c_t = 1$ for approach, $c_t = 1.2$ for kicking, and $k = [10, 20, 30, 30, 30, 10]$ for each leg.

During approach, alternating cosine gait offsets are added. With unilateral period $T_1$, the phase function is

$$\varphi(t) = \frac{1 - \cos(2\pi t / T_1)}{2}. \tag{2}$$

The two legs are activated alternately in adjacent half-cycles. The gait amplitude of a single leg is given by

$$g = (d_h \varphi + d_0) c_g,$$

where $T_1 = 35$, $d_h = 25$, $d_0 = 5$, $c_g$ denotes the gait scaling coefficient, and $\varphi$ is the phase function defined in Eq. (2). The periodic gait is applied to the hip pitch, knee, and ankle pitch joints, with corresponding angular offsets of $-g$, $2g$, and $-g$, respectively.

The periodic gait is fully enabled during the behind-ball positioning phase when the robot approaches the target point behind the ball. When entering the heading alignment phase, $c_g$ is set to 0 to temporarily disable periodic stepping and

avoid forward overshoot. In the kick execution phase, the periodic gait is reactivated, and is superimposed with the joint residual actions output by the reinforcement learning policy to jointly drive the leg joints.

### *D. Behind-Ball Target and Shooting Direction*

To ensure the robot is positioned on the side of the ball facing away from the goal before shooting, this paper constructs a behind-ball target point based on the positions of the ball and the goal. Let $\mathbf{p}_b$, $\mathbf{p}_g$, and $\mathbf{p}_k$ denote the position of the ball, the goal target, and the behind-ball target point respectively, and let $d$ denote the behind-ball distance. The target point is calculated as:

$$p_k = p_b - d\,\frac{p_g - p_b}{\|p_g - p_b\|_2}. \tag{3}$$

The value of $d$ is set to $0.65\,\mathrm{m}$. The robot first moves toward $\mathbf{p}_k$; when the horizontal distance between the robot base and $\mathbf{p}_k$ is less than $0.30\,\mathrm{m}$, the robot adjusts its orientation. The unit shooting direction points from the ball to the goal center. After a shooting attempt, if the ball fails to enter the goal, the system waits until the ball reaches a stable motion state. Once the ball stops or the waiting time reaches the preset upper limit, the behind-ball target point $\mathbf{p}_k$ is recalculated based on the current new position of the ball, and the approaching, alignment and shooting process is executed again.

### *E. Autonomous Shooting Task Workflow*

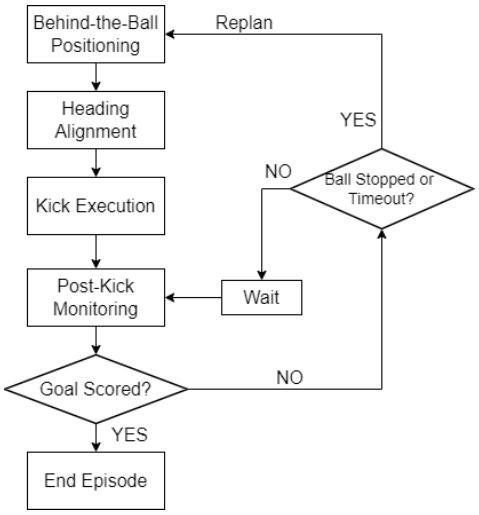



Fig. 2. Task workflow of R1 autonomous shooting.

To characterize the distinct task contexts of the unified policy throughout the complete shooting process, this paper divides the autonomous shooting procedure into four phases: behind-the-ball positioning, heading alignment, kick execution, and post-kick monitoring. All phases share a single unified reinforcement learning policy; the current phase is distinguished solely by task state indicators embedded in the observation vector, and the workflow proceeds according to predefined switching conditions without employing independent sub-policies.

TABLE II

MAIN REWARD WEIGHTS USED IN THE EXPERIMENTAL SETTING

| Reward Item | Symbol | Reward Category | Experimental Weight |
|---|---|---|---|
| Alive reward | $r_{\text{alive}}$ | Base reward $r_{\text{base}}$ | 0.002 |
| Per-step time penalty | $r_{\text{time\_pen}}$ | Base reward $r_{\text{base}}$ | -0.0005 |
| Kick-point progress | $r_{\text{kp}}$ | Stage reward $r_{\text{state}}$ | 1.50 |
| Foot-to-ball progress | $r_{\text{foot}}$ | Stage reward $r_{\text{state}}$ | 0.10 |
| Kicking stability | $r_{\text{bal}}$ | Stage reward $r_{\text{state}}$ | 0.015 |
| First effective touch | $r_{\text{hit}}$ | Kick reward $r_{\text{kick}}$ | 4.00 |
| Ball-to-goal progress | $r_{\text{bg}}$ | Kick reward $r_{\text{kick}}$ | 1.80 |
| Ball velocity toward goal | $r_{\text{vg}}$ | Kick reward $r_{\text{kick}}$ | 0.50 |
| Goal scored | $r_{\text{goal}}$ | Terminal reward $r_{\text{terminal}}$ | +30.0 |
| Fall | $r_{\text{fall}}$ | Terminal reward $r_{\text{terminal}}$ | -5.50 |
| Episode timeout | $r_{\text{timeout}}$ | Terminal reward $r_{\text{terminal}}$ | -1.00 |

The robot first calculates the behind-ball target point based on the positions of the ball and the goal, and moves toward this point. Upon approaching the behind-ball point, the robot enters the heading alignment phase, adjusting its body orientation to align with the shooting direction from the ball to the goal while ensuring the ball remains within a valid shooting window. When the heading error, ball position offset, and body posture all meet the preset thresholds, the robot enters the kick execution phase, where the unified policy continuously outputs leg joint actions to complete the strike.

Subsequently, the system enters the post-kick monitoring phase to continuously track the ball's motion state and determine whether a goal is scored. If the ball enters the goal, the current episode terminates. If no goal is scored and the ball is still in motion with the waiting time below the maximum limit, the system continues waiting and monitoring. If the ball comes to rest or the waiting time reaches the preset limit without a goal, the behind-ball target point is recalculated based on the new position of the ball, and the system returns to the behind-the-ball positioning phase to start the next shooting attempt.

### *F. Reward Design*

The reward function consists of four components: a base reward, a stage reward, a kicking reward, and a terminal reward:

$$r_t = r_{\text{base}} + r_{\text{state}} + r_{\text{kick}} + r_{\text{terminal}}. \tag{4}$$

where $r_{\text{base}}$ is the base reward computed at every control step, including the alive reward, time penalty, and body-posture penalty; $r_{\text{state}}$ is computed according to the current task stage and guides the robot to approach the behind-the-ball point, align its heading, and accomplish the intermediate objective of each stage; $r_{\text{kick}}$ is activated during the kicking interaction to encourage the feet to approach the ball, produce an effective touch, increase the ball velocity toward the goal and the rate of progress toward the goal after contact, while maintaining body stability; and $r_{\text{terminal}}$ is a one-time terminal reward evaluated at the end of an episode, including the goal reward, fall penalty, and timeout penalty. Table II lists the major reward terms and their weights used in the experiments; each term belongs to one of the four reward categories above.

### *G. Full-Region Random Training*

Ball positions are sampled uniformly from $X \in [-5.5, 5.5]\,\mathrm{m}$ and $Z \in [-4, 4]\,\mathrm{m}$, covering near, far, lateral,

and behind-robot locations. Diverse task distributions have also supported complex legged skills [8], [17].

Training first uses near-ball states to learn contact, goal-directed speed, and stability; it then enables the full state machine and full-region sampling. After spatial diagnosis, training continues in the selected weak region. The final policy accumulated about $7.0 \times 10^7$ environment steps.

### H. Weak-Region Diagnosis and Spatially Targeted Training

Partition $\Omega$ into $K$ subregions $\Omega_i$. If region $i$ has $S_i$ successes among $N_i$ trials, its success rate is

$$P_i = \frac{S_i}{N_i}. \tag{5}$$

We split the field at $X = 0\,\mathrm{m}$ and $Z = 0\,\mathrm{m}$, assigning boundaries to the $X \geq 0\,\mathrm{m}$ and $Z \geq 0\,\mathrm{m}$ sides. Baseline diagnosis selected the lower-right region $\Omega_w = \{X < 0\,\mathrm{m}, Z < 0\,\mathrm{m}\}$; subsequent ball sampling was restricted to $\Omega_w$. This concentrates training on poor locations and is related to curriculum methods [12], [13], but selection and switching are manual.

After continued training, the full distribution is restored and the frozen policy is re-evaluated. The procedure addresses spatial imbalance; it is not an automatic region-discovery algorithm.

## IV. EXPERIMENTS AND RESULTS

### A. Unity Simulation Environment

Experiments use Unity 2022.3.62f3c1, ML-Agents 3.0.0, and a $0.01$-s physics step on an Intel Core i7-12700H computer with 16-GB RAM and 4-GB GPU memory. Training used about $7.0 \times 10^7$ steps. Spatial trials allow $30\,\mathrm{s}$. In fixed-position tests, the learned and rule-driven systems allow $25$ and $80\,\mathrm{s}$, respectively. Their scenes advance independently but use identical local coordinates, ordering, and goal criteria. Thus, the comparison evaluates the configured systems, gives the rule baseline a more permissive limit, and does not support an equal-budget speed claim. Real-robot transfer would require additional treatment of the simulation gap, e.g., domain randomization [16].

TABLE III

MAIN EXPERIMENTAL SETTINGS

| Item | Setting |
|---|---|
| Robot / environment | R1 / Unity soccer field |
| Observation / action | 59-D / 12-D continuous |
| Physics step | $0.01\,\mathrm{s}$ |
| Evaluation limits | $30\,\mathrm{s}$ (spatial) / $25\,\mathrm{s}$ (learned fixed) / $80\,\mathrm{s}$ (rule fixed) |
| Training/evaluation region | $X \in [-5.5, 5.5]\,\mathrm{m}$, $Z \in [-4, 4]\,\mathrm{m}$ |
| Software / hardware | Unity 2022.3.62f3c1; ML-Agents 3.0.0; i7-12700H; 16-GB RAM; 4-GB GPU |
| PPO / network | $3 \times 512$; normalized obs.; batch 2048; buffer 20480; lr $3 \times 10^{-4}$ linear; $\varepsilon = 0.2$; $\beta = 0.005$; $\lambda = 0.95$; 3 epochs; $\gamma = 0.995$; horizon 1000 |
| Training steps / repeats | $\approx 7.0 \times 10^7$ / 5 per fixed position |

### B. Training Region and Spatial Partition

The region is split at $X = 0\,\mathrm{m}$ and $Z = 0\,\mathrm{m}$; boundary points join the $X \geq 0\,\mathrm{m}$ and $Z \geq 0\,\mathrm{m}$ sides. A fixed seed yields 53, 52, 44, and 51 samples in the four regions. Baseline diagnosis selects $X < 0\,\mathrm{m}$, $Z < 0\,\mathrm{m}$ as the lower-right weak region.

We compare a full-region model and its weak-region-continued successor. Both frozen policies use the same 200 coordinates in the same order. Because baseline outcomes on these coordinates selected the weak region and the coordinates were then reused, we call them the diagnosis–re-evaluation set: paired outcomes measure change at controlled locations but are not independent holdout evidence. The successor also receives extra training, so this is not an equal-budget one-factor ablation. A single training run and checkpoint provide only preliminary evidence [18].

### C. Full-Region Random-Position Evaluation

Seed 202608 generates $N = 200$ positions. Each trial allows $30\,\mathrm{s}$; a goal is success and a fall or timeout is failure. We report the success rate with a 95% Wilson interval [19], use the two-sided exact McNemar test for paired outcomes [20], and compute time only over successful trials.

TABLE IV

RESULTS BEFORE AND AFTER WEAK-REGION CONTINUED TRAINING ON THE DIAGNOSIS–RE-EVALUATION SET

| Training | Overall | Weak region | Reg. SD | Mean time |
|---|---|---|---|---|
| Full-region only | 65.5% | 48.1% | 11.32 | $9.60\,\mathrm{s}$ |
| Full + weak cont. | 72.5% | 76.9% | 8.14 | $8.52\,\mathrm{s}$ |

Table IV shows 131/200 successes for full-region training (65.5%, 95% CI 58.7%–71.7%) and 145/200 after continued training (72.5%, 95% CI 65.9%–78.2%). The pair counts are 99 both successful, 23 both failed, 32 baseline-only successes, and 46 continued-only successes. McNemar $p = 0.141$; the 7.0-point overall increase is therefore not significant at 0.05.

The selected lower-right region improved from 25/52 (48.1%) to 40/52 (76.9%), with paired $p = 0.0026$. Because the same baseline outcomes selected this region, the $p$-value is descriptive post-selection evidence, not independent confirmation. Upper-right and upper-left rates rose by 3.8 and 15.9 points, while lower-left fell by 19.6 points. The four-region standard deviation decreased from 11.32 to 8.14 points, but the regression indicates cross-region forgetting risk. Falls decreased from 68 to 54; each model had one timeout. Successful-trial means were $9.60$ and $8.52\,\mathrm{s}$, and the continued model was $0.43\,\mathrm{s}$ faster over the 99 commonly successful positions.

The data support improvement on the diagnosed region, not stable generalization to unseen positions. Independent holdout tests, multiple seeds, equal training budgets, and forgetting analysis are required.

### D. Spatial Visualization of Shooting Outcomes

Fig. 4 contains 350/500 successes (70.0%, 95% CI 65.8%–73.9%) [19]. Of 150 failures, 148 were falls and two were

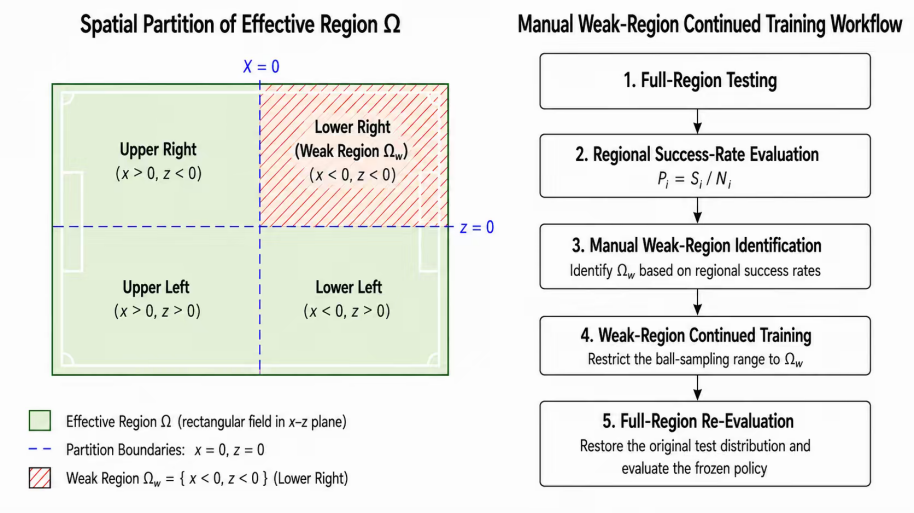



Fig. 3. Spatial partition and manually selected weak-region continued training.

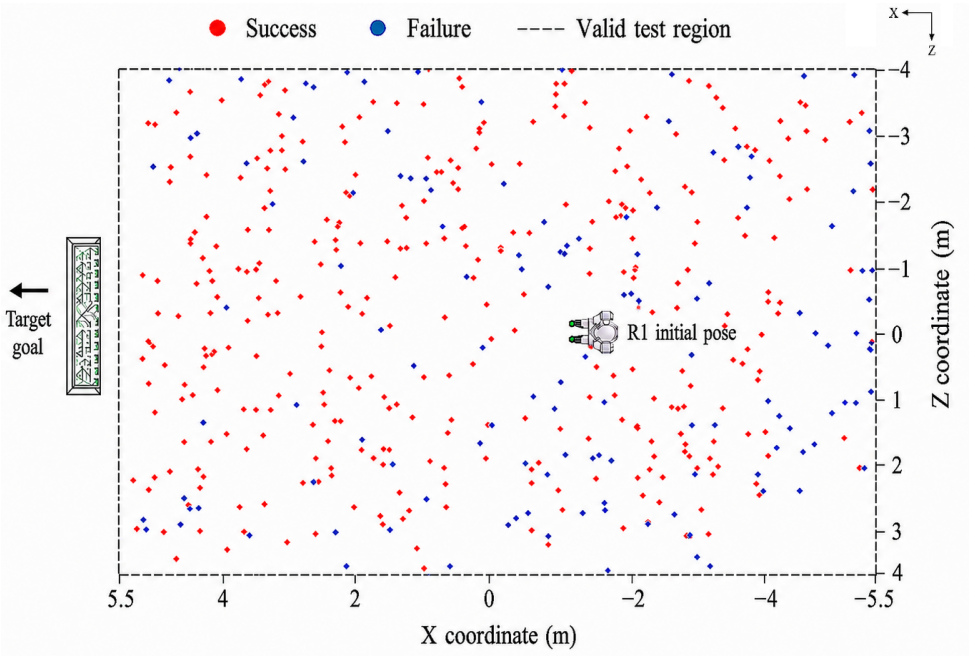
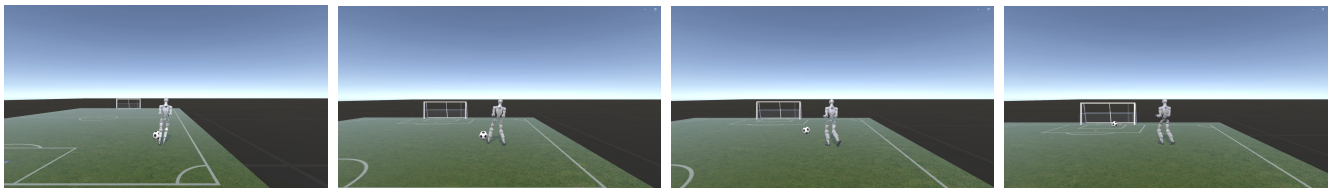



Fig. 4. Outcomes at 500 random positions after weak-region continued training (red: success; blue: failure).



(a) approaching (b) kicking (c) ball launch (d) goal scoring

Fig. 5. Representative R1 shooting sequence in Unity

timeouts, identifying dynamic stability as the principal failure mode.

Successes cover the region, but the rate is 237/315 (75.2%) in front of the initial robot position ($X \geq -1.3\,\mathrm{m}$) and 113/185 (61.1%) behind it. The $Z \in [2, 4]\,\mathrm{m}$ band reaches 61.1%, versus 71.0%–75.0% in the other equal-width bands. Fig. 4 visualizes coverage; direct training comparison uses the paired 200 positions in Section IV-C.

### *E. Learned Versus Rule-Driven Shooting*

Both systems use the same R1 model, field geometry, ball, goal, eight local positions, five rounds, order, and goal

criterion. Scenes are offset by $15\,\mathrm{m}$ along world $Z$ but retain identical local geometry. Our system learns 12 residual leg actions; the baseline uses an existing walking policy plus hand-coded chasing/contact rules and does not learn the kick. Limits are $25\,\mathrm{s}$ and $80\,\mathrm{s}$, so this is a comparison of complete configured systems rather than an equal-time kick ablation.

P1–P8 cover upper-center, upper-left, upper-right, right, left, lower-center, lower-left, and lower-right locations, for 40 trials per method. Labels follow the initial robot heading. Wilson intervals [19] and paired McNemar tests [20] are reported.

TABLE V

EIGHT FIXED BALL POSITIONS AND PER-POSITION OUTCOMES

| Pos. | Region | Base $(X, Z)$ | Rule /5 | Learned /5 |
|---|---|---|---|---|
| P1 | Upper center | $(2.20, 0.00)$ | 4 | 3 |
| P2 | Upper left | $(3.39, 2.23)$ | 5 | 4 |
| P3 | Upper right | $(3.39, -2.59)$ | 4 | 5 |
| P4 | Right | $(-1.06, -3.15)$ | 3 | 5 |
| P5 | Left | $(-1.06, 2.72)$ | 1 | 5 |
| P6 | Lower center | $(-3.39, 0.10)$ | 1 | 4 |
| P7 | Lower left | $(-2.74, 2.54)$ | 0 | 5 |
| P8 | Lower right | $(-2.81, -2.88)$ | 0 | 2 |

TABLE VI

OVERALL SHOOTING PERFORMANCE

| Method | Trials | Goals | Rate | Mean successful time |
|---|---|---|---|---|
| Rule-driven R1 | 40 | 18 | 45.0% | $49.74\,\mathrm{s}$ |
| Learned R1 (ours) | 40 | 33 | 82.5% | $11.31\,\mathrm{s}$ |

### *F. Results and Discussion*

The learned system succeeds in 33/40 trials (82.5%, 95% CI 68.1%–91.3%) and the rule system in 18/40 (45.0%, 95% CI 30.7%–60.2%), a 37.5-point gap. Pair counts are 15 both successful, four both failed, 18 learned-only successes, and three rule-only successes; McNemar $p = 0.0015$. Because the rule system has a longer limit and a different controller, this establishes a difference between the configured systems, not the isolated causal effect of learned kicking under equal time.

Successful-trial times are $11.31 \pm 4.49\,\mathrm{s}$ and $49.74 \pm 15.29\,\mathrm{s}$. Different limits and success sets make these descriptive; we do not claim an equal-budget speed advantage. The learned system has more successes at P3–P8 and scores 11/15 versus 1/15 over the three behind-robot positions. Its 2/5 result at P8 still exposes a lower-right weakness.

## V. CONCLUSION

We developed an R1 random-position shooting system in which behind-ball planning and a state machine organize the task, a periodic gait supplies a motion prior, and learned 12-D residual actions replace fixed kick angles. We also evaluated a full-region, diagnosis, weak-region continuation, and re-evaluation procedure.

On the reused 200-position diagnosis–re-evaluation set, success rose from 65.5% to 72.5% and the selected lower-right region from 48.1% to 76.9%; however, overall $p = 0.141$, the regional result is post-selection, and lower-left performance regressed. In the fixed-position system comparison, learned and rule-driven systems scored 33/40 and 18/40 goals ($p = 0.0015$) under unequal limits. The results support task feasibility and targeted improvement potential. Future work should use independent holdouts, multiple seeds, equal-time/equal-training controls, and module ablations, and should address moving balls, vision, opponents, and sim-to-real transfer.

## REFERENCES


[1] T. Haarnoja *et al.*, "Learning agile soccer skills for a bipedal robot with deep reinforcement learning," *Sci. Robot.*, vol. 9, no. 89, Art. no. eadi8022, 2024.

[2] Y. Zhong, Y. Lu, Y. Lu, T. Tang, H. Mai, Y. Pan, T. Li, L. Chen, J. Wang, Z. Li, P. Lu, and H. Li, "RoboNaldo: Accurate, stable and powerful humanoid soccer shooting via motion-guided curriculum reinforcement learning," arXiv:2606.11092, 2026.

[3] Z. Xu, M. Seo, D. Lee, H. Fu, J. Hu, J. Cui, Y. Jiang, Z. Wang, A. Brund, J. Biswas, and P. Stone, "Learning agile striker skills for humanoid soccer robots from noisy sensory input," arXiv:2512.06571, 2025.

[4] M. Abreu, T. Silva, H. Teixeira, L. P. Reis, and N. Lau, "6D localization and kicking for humanoid robotic soccer," *J. Intell. Robot. Syst.*, vol. 102, Art. no. 30, 2021.

[5] A. Abdolmaleki, D. Simões, N. Lau, L. P. Reis, G. Neumann, S. Sarıel, and D. D. Lee, "Learning a humanoid kick with controlled distance," in *RoboCup 2016: Robot World Cup XX*, LNAI, vol. 9776, 2017, pp. 45–57.

[6] X. B. Peng, P. Abbeel, S. Levine, and M. van de Panne, "DeepMimic: Example-guided deep reinforcement learning of physics-based character skills," *ACM Trans. Graph.*, vol. 37, no. 4, Art. no. 143, pp. 1–14, 2018.

[7] Z. Xie, P. Clary, J. Dao, P. Morais, J. Hurst, and M. van de Panne, "Learning locomotion skills for Cassie: Iterative design and sim-to-real," in *Proc. Conf. Robot Learn.*, PMLR, vol. 100, 2020, pp. 317–329.

[8] N. Rudin, D. Hoeller, P. Reist, and M. Hutter, "Learning to walk in minutes using massively parallel deep reinforcement learning," in *Proc. 5th Conf. Robot Learn.*, PMLR, vol. 164, 2022, pp. 91–100.

[9] X. Da *et al.*, "Learning a contact-adaptive controller for robust, efficient legged locomotion," in *Proc. Conf. Robot Learn.*, PMLR, vol. 155, 2021, pp. 883–894.

[10] S. Ha, P. Xu, Z. Tan, S. Levine, and J. Tan, "Learning to walk in the real world with minimal human effort," in *Proc. Conf. Robot Learn.*, PMLR, vol. 155, 2021, pp. 1110–1120.

[11] T. Johannink *et al.*, "Residual reinforcement learning for robot control," in *Proc. IEEE ICRA*, 2019, pp. 6023–6029.

[12] C. Florensa, D. Held, M. Wulfmeier, M. Zhang, and P. Abbeel, "Reverse curriculum generation for reinforcement learning," in *Proc. 1st Annu. Conf. Robot Learn.*, PMLR, vol. 78, 2017, pp. 482–495.

[13] C. Florensa, D. Held, X. Geng, and P. Abbeel, "Automatic goal generation for reinforcement learning agents," in *Proc. 35th Int. Conf. Mach. Learn.*, PMLR, vol. 80, 2018, pp. 1515–1528.

[14] A. Juliani *et al.*, "Unity: A general platform for intelligent agents," arXiv:1809.02627, 2018.

[15] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," arXiv:1707.06347, 2017.

[16] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," in *Proc. IEEE/RSJ IROS*, 2017, pp. 23–30.

[17] Z. Zhuang *et al.*, "Robot parkour learning," in *Proc. 7th Conf. Robot Learn.*, PMLR, vol. 229, 2023, pp. 73–92.

[18] P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup, and D. Meger, "Deep reinforcement learning that matters," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 3207–3214.

[19] E. B. Wilson, "Probable inference, the law of succession, and statistical inference," *J. Amer. Statist. Assoc.*, vol. 22, no. 158, pp. 209–212, 1927.

[20] Q. McNemar, "Note on the sampling error of the difference between correlated proportions or percentages," *Psychometrika*, vol. 12, no. 2, pp. 153–157, 1947.