Extended Abstract

Motivation Recent quadruped parkour systems have achieved agile locomotion over complex terrain, but they are usually trained as single-agent traversal tasks or with multi-stage pipelines that separate locomotion, perception, and navigation. In contrast, animals often move through constrained environments while reacting to other dynamic agents. This project studies whether competitive self-play can serve as an automatic curriculum for quadruped parkour. We ask whether predator and prey quadruped agents can jointly learn pursuit, evasion, and terrain-aware strategies through one-stage reinforcement learning with survival-driven rewards.

Method We build a simulated Unitree Go2 predator-prey environment with flat and constrained terrains, including walls, narrow passages, platforms, ramps, gaps, and cluttered regions. Both agents are trained simultaneously from scratch. Our main reward design combines a dense *instinct* reward and sparse *survival* rewards. The dense reward encourages the predator to reduce the predator-prey distance and the prey to increase it, while a body-frame forward-motion factor discourages unnatural crab-walking. Sparse rewards handle capture, timeout, safe-zone reaching, and fall or collision termination. We also include standard locomotion regularizers to improve stability and physical plausibility.

Implementation The policy uses proprioception, egocentric 360° LiDAR perception, temporal memory, and estimator modules. A DP3D-style PointNet encoder extracts point-cloud tokens, a transformer fuses perception and proprioception, and a GRU preserves temporal information. We train both a privileged variant, partial-vision-estimator variant, and vision-estimator variant. In the vision setting, the actor does not directly receive privileged opponent state at execution time; instead, it receives onboard proprioception and a detached estimator latent. As an intermediate stage, in partial vision variant, the policy have access to the ground-truth opponent relative information, which are hard-to-estimate. During training, however, the critic and estimator still use privileged simulator supervision, so robustness under noisier real-world sensing remains an important challenge.

Results Our experiments show that our one-stage self-play policy, both privileged and partial-vision variant, can produce stable locomotion and nontrivial competitive behavior. Bad-locomotion terminations decrease substantially during training, with the privileged flat setting reaching about 10% locomotion failure. Same-era win rates remain competitive in the privileged flat and partial vision settings. The full vision variant reveal estimator-related bottlenecks on predicting opponent states, and would require a larger number of environments and longer training, which is beyond our time and compute budget to investigate. Qualitatively, we observe anticipatory pursuit, obstacle-aware evasion, and strategy evolution: early prey policies exploit scene obstacles to make predators overshoot, while later predators adapt by slowing near walls/corners and reacting to the prey's escape direction.

Discussion and Conclusion These results suggest that competitive self-play can induce more complex strategies than merely reward-shaped behaviors, including terrain-aware behaviors beyond static obstacle traversal, and strategy evolution. The opponent becomes a moving part of the environment and creates an adaptive curriculum for pursuit and evasion. However, our current evaluation is limited by idealized simulation, empirical reward tuning, partial dependence on privileged training supervision, and the instability of the full vision-estimator setting. Future work include improving estimator pretraining and curricula, extend safe-zone and multi-agent objectives, and improve sim-to-real robustness through domain randomization. Overall, this project provides initial evidence that pursuit, evasion, locomotion, and terrain use can emerge jointly from competitive self-play training, while the idealized one-staged sim-to-real ready pipeline will require more compute and careful tuning of the training regime.

Survival Instinct: One-Staged RL for Quadruped Parkour Self-Play

Haoyue Xiao
Department of Computer Science
Stanford University
xiao1121@stanford.edu

Shatong Zhu
Department of Electrical Engineering
Stanford University
stzhu@stanford.edu

Edward Lee
Department of Computer Science
Stanford University
edwardnl@stanford.edu

Abstract

Quadruped parkour has achieved agile locomotion over complex terrain, but most existing systems focus on single-agent traversal or multi-stage training pipelines. We instead study quadruped parkour as a predator-prey self-play problem, where two simulated Unitree Go2 agents jointly learn locomotion, pursuit, evasion, and terrain-aware strategy. We build flat and constrained training environments and train both agents from scratch with PPO-style actor-critic learning. Our method uses survival-driven rewards: a dense distance-progress reward encourages predators to close the gap and prey to escape, while sparse rewards handle capture, timeout, safe-zone objectives, and locomotion failures. We also develop a vision-estimator architecture using proprioception, egocentric 360° LiDAR, temporal memory, and supervised estimator modules for opponent-related information. Experiments show that one-stage self-play can produce stable locomotion and nontrivial competitive behaviors in privileged and partial vision-estimator settings. Bad-locomotion terminations decrease during training, same-era win rates remain competitive in several settings, and qualitative rollouts show anticipatory pursuit, obstacle-aware evasion, and corner-based strategy adaptation. These results suggest that competitive self-play can serve as an automatic curriculum for dynamic quadruped parkour, while revealing challenges in full vision-estimator stability, reward robustness, and sim-to-real transfer.

1 Introduction

Legged robots have become increasingly capable of agile locomotion over complex terrain, including stairs, gaps, ramps, narrow passages, and cluttered obstacle courses. Recent reinforcement learning approaches to quadruped parkour have shown that carefully designed rewards, terrain curricula, privileged training, and perception estimators can produce robust traversal behaviors Cheng et al. (2024); Zhuang et al. (2023); Luo et al. (2024). However, most existing parkour settings treat the task as single-agent traversal through static terrain. In these environments, the main challenge comes from the geometry of the terrain itself. In natural settings, however, agile locomotion is often interactive: animals and robots must move through constrained environments while reacting to other dynamic agents.

In this project, we study whether quadruped agents can learn pursuit, evasion, and terrain-aware strategies through one-stage predator–prey self-play with survival-driven reward engineering. We

formulate a two-agent pursuit-evasion task in which two simulated Unitree Go2 quadrupeds are trained in the same arena. The predator aims to catch the prey, while the prey aims to survive by escaping, reaching a safe region, or delaying capture until timeout. This setting couples low-level locomotion with high-level interactive decision making. A successful predator must anticipate the prey's future motion, cut off escape routes, and remain stable near obstacles, while a successful prey must use walls, turns, narrow passages, and safe regions to break pursuit. In this way, the opponent becomes a moving source of difficulty, forcing each agent to adapt as the other improves.

This setting creates several challenges. First, simultaneous predator–prey learning is non-stationary because each agent's opponent changes throughout training. Second, sparse win/loss events such as capture, timeout, and safe-zone reaching are difficult to learn from without dense reward shaping. Third, quadruped agents must preserve physically plausible locomotion while optimizing an adversarial objective, otherwise they may exploit unstable motions or collide with obstacles. Finally, a deployable policy should rely on onboard sensing rather than direct privileged simulator state, which introduces partial observability and makes opponent-state estimation a central part of the problem.

To address these challenges, we build a simulated predator–prey quadruped parkour environment in IsaacLab/IsaacSim. The environment includes flat debugging arenas and constrained terrains with walls, platforms, ramps, rough regions, depressions, and safe zones. We train predator and prey policies from scratch with PPO-style actor–critic learning Schulman et al. (2017). During development, we use a privileged setting to stabilize the learning pipeline and validate the environment, reward, reset, critic, and evaluation logic. We then introduce a vision-estimator setting in which privileged opponent information is removed from the actor and replaced by a recurrent estimator latent computed from egocentric 360° LiDAR and proprioceptive history. This allows us to study both the strategic potential of self-play under privileged information and the additional difficulty introduced by perception-based deployment.

A central technical contribution of our project is the survival-instinct reward design. Purely sparse survival rewards were not sufficient to produce stable chase–evasion behavior: agents could discover degenerate strategies such as predators hovering near the prey without completing capture or prey policies passively waiting for timeout. To address this, we design a dense instinct reward based on the change in predator–prey distance. The predator is rewarded for reducing the gap, while the prey is rewarded for increasing it. We further multiply this progress signal by a body-frame forward-motion factor to discourage crab-walking and promote visually natural pursuit and escape. This dense term is combined with sparse survival rewards for capture, timeout, safe-zone reaching, and locomotion failure, together with small regularizers for smoothness, stability, and contact quality.

Our experiments evaluate whether this one-stage self-play formulation can produce stable locomotion and meaningful competitive behavior. In the privileged setting, agents learn stable pursuit and evasion, with locomotion failure decreasing substantially during training. In the partial vision-estimator setting, removing most privileged actor information still preserves nontrivial locomotion and competitive interaction, although performance drops compared to the privileged policy. The full vision-estimator setting remains the most difficult: noisy early opponent estimates can destabilize policy learning, leading to higher locomotion failure and weaker strategic behavior. These results suggest that the reward and self-play formulation can induce chase–evasion behavior, while the estimator remains a key bottleneck for fully deployable perception-based control.

Qualitatively, the learned policies exhibit behaviors that are not explicitly programmed by the reward. Predators learn to intercept turning prey rather than simply tailgating their current position. Prey agents exploit walls and corners to block direct pursuit and increase the effective path length to capture. Across checkpoints, we observe strategy evolution: earlier predators may collide with walls when prey use sudden turns, while later predators slow down near corners, observe the prey's escape direction, and execute more agile catches. We also use same-era win rates and cross-checkpoint evaluation to diagnose whether self-play improves strategic strength or merely produces transient opponent-specific adaptations.

Overall, this project is an empirical study of survival-driven self-play for quadruped parkour. Rather than treating parkour as a single-agent traversal problem, we study it as an interactive game in which terrain and opponent behavior jointly determine the optimal strategy. Our results suggest that competitive self-play, combined with dense instinct rewards, sparse survival objectives, terrain curricula, and perception-aware policy design, can produce meaningful and interpretable pursue-and-evade behaviors in quadruped agents. At the same time, the gap between privileged and

full vision-estimator performance highlights important future directions in estimator pretraining, perception robustness, reward balance, and sim-to-real transfer.

2 Related Work

2.1 Legged Parkour and Perception-Based Locomotion

Recent reinforcement learning systems have shown that legged robots can learn agile locomotion over difficult terrain when trained with large-scale simulation, carefully shaped rewards, and terrain curricula. Massively parallel RL has made it practical to train robust quadruped locomotion policies efficiently in simulation and transfer them to real robots (Rudin et al., 2022). Several works build on this foundation and extend quadruped locomotion to parkour. Extreme Parkour trains quadrupeds to traverse obstacles such as gaps, hurdles, stairs, and ramps using privileged terrain information and a later perception-based policy (Cheng et al., 2024). Robot Parkour Learning demonstrates that a single vision-based policy can select and execute diverse skills such as climbing, leaping, crawling, and squeezing through obstacles (Zhuang et al., 2023). PIE further improves perception-based parkour with an implicit-explicit estimator that fuses proprioception and egocentric perception in a one-stage learning framework (Luo et al., 2024).

These works provide our project's technical foundation of reward shaping, privileged training, perception estimation, and terrain curricula. However, they mainly focus on single-agent traversal through static terrain. Our setting introduces an adversarial learning agent, creating an adapting environment. Terrain is both an obstacle and a strategic resource: walls, corners, and narrow passages can be used by prey to evade or by predators to intercept. This makes the task both a locomotion problem and a pursuit-evasion game.

2.2 Privileged-to-Perception Training

Real-world legged policies cannot rely on privileged simulator state, so many parkour systems use privileged information during training while learning policies or estimators that use onboard observations at deployment. Extreme Parkour follows a privileged-to-perception pipeline, and PIE shows that explicit estimates and latent implicit features can bridge noisy perception and agile legged motion (Cheng et al., 2024; Luo et al., 2024). Perceptive Humanoid Parkour applies a related idea to humanoid robots, combining perception, skill composition, and policy distillation for long-horizon obstacle traversal (Wu et al., 2026).

Our system follows the same broad motivation but differs in setting and objective. We use privileged simulator information during development to stabilize training, while moving toward actor-side estimators that infer terrain and opponent-related quantities from non-privileged observations. Unlike prior work where perception mainly supports static obstacle traversal, our perception module must also support interactive reasoning about a moving opponent. Our use of egocentric 360° LiDAR and point-cloud tokens is motivated by compact 3D representations for robot control, such as 3D Diffusion Policy, which shows that sparse point-cloud features can provide useful and generalizable visuomotor representations (Ze et al., 2024).

2.3 Self-Play and Embodied Multi-Agent Reinforcement Learning

Self-play has been widely used as an automatic curriculum for competitive reinforcement learning. AlphaZero showed that strong policies can emerge from self-play without human demonstrations in board games such as chess, shogi, and Go (Silver et al., 2018). In robotics, recent work on learning robot soccer from egocentric vision demonstrates that multi-agent RL can produce active perception and competitive embodied behavior in a dynamic, partially observable environment (Tirumala et al., 2025).

Our project brings this idea into quadruped parkour. Instead of training a single robot to traverse a fixed terrain, we train predator and prey quadrupeds together so that each agent becomes part of the other's curriculum. This couples low-level locomotion, terrain use, perception, and strategic interaction. The key difference from prior self-play work is that our agents must solve an agile legged locomotion problem at the same time as a pursuit-evasion game in constrained terrain.

3 Method

3.1 Overview

We study a 1-vs-1 **predator–prey** pursuit game between two Unitree Go2 quadrupeds, trained by self-play in IsaacLab/IsaacSim. Both robots learn simultaneously each with their individual PPO: the predator learns to catch and the prey to evade, so each policy continually best-responds to an improving opponent. Our core goal is a deployable, vision-based policy: at execution time each agent perceives its opponent only through an egocentric LiDAR scan, never through privileged simulator state. We achieve this with a recurrent point-cloud estimator that distills the privileged opponent state from the LiDAR cloud and proprioception, and whose latent is taken by the actor.

3.2 Problem Formulation

We model the task as a two-agent partially observable Markov game with agents $i \in \{\text{pred}, \text{prey}\}$ and fixed roles within an episode. At each timestep t, each agent applies an action a_t^i, the simulator advances, and each agent receives a role-specific reward r_t^i and a partial observation o_t^i. An episode ends when:

- **Catch, predator wins:** the predator reaches the prey within $d_{\text{catch}}{=}0.55\,\text{m}$
- **Escape, prey wins:** the prey reaches a designated safe region $\mathcal{Z}_{\text{safe}}$, a hard-to-reach corner in the map (agents are not told its existence).
- **Timeout, prey wins:** the prey survives to $T = 1000$.
- **Fall, excluded:** an agent's body tilts past a gravity threshold.

The interaction is approximately zero-sum: the predator is rewarded for closing distance and catching, the prey for the opposite.

3.3 Observation and Action Spaces

We consider two agents, predator and prey, indexed by i. Each agent uses the same action and observation parameterization, with role information provided as part of the input.

Action. The policy outputs a 12-dimensional joint-position command

$$a_t^i \in \mathbb{R}^{12}, \tag{1}$$

corresponding to the target positions of the quadruped's leg joints. The targets are tracked by a fixed low-level PD controller. The action space is identical in the privileged and vision settings.

Proprioception. The onboard proprioceptive state is

$$x_t^i = \left[\boldsymbol{v}_t^i,\, \boldsymbol{\omega}_t^i,\, \boldsymbol{g}_t^i,\, \boldsymbol{q}_t^i,\, \dot{\boldsymbol{q}}_t^i\right] \in \mathbb{R}^{33}, \tag{2}$$

where $\boldsymbol{v}_t^i$ and $\boldsymbol{\omega}_t^i$ are the base linear and angular velocities, $\boldsymbol{g}_t^i$ is the gravity direction projected into the body frame, and $\boldsymbol{q}_t^i, \dot{\boldsymbol{q}}_t^i$ are joint positions and velocities. The angular velocity and gravity direction are measured by the IMU, joint states are measured by encoders, and the base linear velocity is provided by the state estimator. We augment x_t^i with a role one-hot vector e^i and a normalized time-to-go $\tau_t = (T - t)/T$. The actor also receives a short proprioceptive history

$$\mathcal{H}_t^i = \left(x_{t-H+1}^i, \ldots, x_t^i\right), \qquad H = 10. \tag{3}$$

We denote the deployable proprioceptive input by

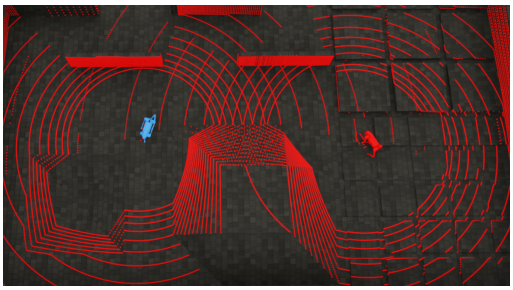
$$p_t^i = \left[x_t^i,\, e^i,\, \tau_t,\, \mathcal{H}_t^i\right]. \tag{4}$$

Privileged information. During training, the critic has access to privileged variables

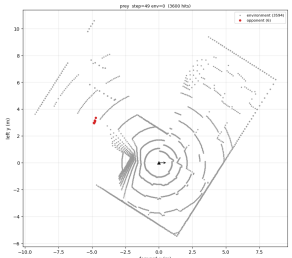
$$\psi_t^i = \left[\boldsymbol{r}_{\text{pos},t}^i,\, \boldsymbol{r}_{\text{vel},t}^i,\, \boldsymbol{m}_{\text{height},t}^i,\, \boldsymbol{m}_{\text{wall},t}^i,\, \eta_t^i\right], \tag{5}$$

where $\boldsymbol{r}_{\text{pos}}^i$ and $\boldsymbol{r}_{\text{vel}}^i$ are the opponent's relative position and velocity, $\boldsymbol{m}_{\text{height}}^i$ is a downward height scan, $\boldsymbol{m}_{\text{wall}}^i$ denotes wall-distance rays, and η^i is the absolute heading. The privileged critic observation is

$$o_{\text{crit},t}^i = \left[p_t^i,\, \psi_t^i\right]. \tag{6}$$



(a) Body-mounted 360° LiDAR ray casting in the simulator.



(b) A 2D visualization of lidar scan result; red dots denotes the opponent (for visualization only, not known by the estimator)

Figure 1: LiDAR observation used by the opponent-state estimator. Each ray is represented by a normalized 3D endpoint together with range and angular information in the egocentric frame.

Actor observations. We train two variants using the same reward and code path. The privilege-info policy receives

$$o^i_{\text{priv},t} = \left[b^i_t,\, \psi^i_t\right], \tag{7}$$

and the pure vision policy removes all privileged blocks and receives only onboard-observable inputs plus an estimator latent:

$$o^i_{\text{vis},t} = \left[b^i_t,\, \text{sg}(z^i_t)\right]. \tag{8}$$

Here z^i_t is produced by the opponent-state estimator described in Sec. 3.6, and $\text{sg}(\cdot)$ denotes stop-gradient. Thus, the actor cannot directly access opponent state, height scans, wall rays, or absolute heading in the vision setting; these quantities are retained only by the critic.

LiDAR observation. Each agent uses a body-mounted 360° LiDAR, simulated by ray casting against the terrain and the opponent mesh. The sensor has 10 vertical beams and 360 horizontal azimuth samples, giving $N = 3600$ rays per scan. For ray k at time t, let $\rho_{t,k}$ be the measured range, $\boldsymbol{u}_k \in \mathbb{R}^3$ be its fixed unit direction in the agent's egocentric yaw-aligned frame, α_k be its horizontal azimuth angle, and β_k be its vertical elevation angle. We encode each ray as

$$\boldsymbol{p}_{t,k} = \left[\underbrace{\frac{\rho_{t,k}\boldsymbol{u}_k}{\rho_{\max}}}_{\text{3D hit point } (x,y,z)},\ \underbrace{\frac{\rho_{t,k}}{\rho_{\max}}}_{\text{range}},\ \underbrace{\sin\alpha_k, \cos\alpha_k}_{\text{azimuth / yaw direction}},\ \underbrace{\beta_k}_{\text{elevation}}\right] \in \mathbb{R}^7. \tag{9}$$

The first three dimensions are the normalized Cartesian coordinates of the ray endpoint in the robot-centric frame. The fourth dimension keeps the raw normalized distance, which makes near and far returns explicit. The sine–cosine pair encodes the horizontal bearing of the ray without an angle discontinuity at 2π, and the final elevation term identifies which vertical beam produced the return. The full LiDAR scan is therefore represented as the unordered point set

$$\mathcal{P}^i_t = \{\boldsymbol{p}^i_{t,k}\}_{k=1}^N, \tag{10}$$

which provides both local geometry and bearing information for opponent-state estimation.

3.4 Policy and Value Networks

Each agent has an independent stochastic actor

$$\pi_{\theta_i}(a^i_t \mid o^i_t) = \mathcal{N}\left(\mu_{\theta_i}(o^i_t), \text{diag}(\sigma^2_{\theta_i})\right), \tag{11}$$

where μ_{θ_i} is an MLP and σ_{θ_i} is a learned diagonal standard deviation. The critic is asymmetric: it consumes the privileged observation $o^i_{\text{crit},t}$ during training but is discarded at deployment.

To reduce interference between reward components, the critic predicts a vector of value heads

$$\boldsymbol{V}_{\phi_i}(o^i_{\text{crit},t}) = \left[V^{\text{sparse}}_{\phi_i}, V^{\text{shape}}_{\phi_i}, V^{\text{reg}}_{\phi_i}\right], \tag{12}$$

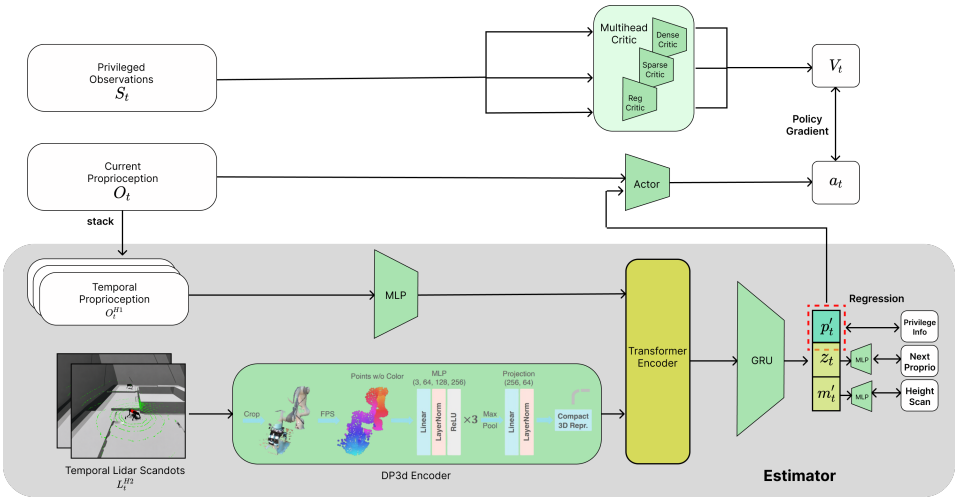


Figure 2: Architecture overview.

3.5 "Survival-Instinct" Rewards

Our core dense shaping term is a gap-progress reward that induces chase–evasion behavior, which we called the "instinct". It's motivated by the animal's instinct to approach their prey/escape from the predator. Let d_t denote the predator–prey distance and define the clipped gap-rate

$$\dot{d}_t = \mathrm{clip}\left(\frac{d_t - d_{t-\Delta t}}{\Delta t}, -c, c\right). \tag{13}$$

To discourage sideways crab-walking, we multiply the reward by a head-first motion factor

$$\alpha_t = \max\left(0, \frac{v^B_{x,t}}{\sqrt{(v^B_{x,t})^2 + (v^B_{y,t})^2 + \varepsilon}}\right), \tag{14}$$

where $v^B_{x,t}$ and $v^B_{y,t}$ are the body-frame planar velocities. The final reward expression is

$$r_{\mathrm{dense},t} = \begin{cases} 1 + \alpha_t \left(\exp(-\lambda \dot{d}_t) - 1\right), & \text{predator}, \\ 1 + \alpha_t \left(\exp(+\lambda \dot{d}_t) - 1\right), & \text{prey}. \end{cases} \tag{15}$$

Thus, the predator is rewarded for reducing the gap, while the prey is rewarded for increasing it; in both cases, the progress term is strongest when the robot moves forward in its own body frame.

Sparse rewards define episode outcomes. A fall or invalid collision terminates the agent when the body tilt exceeds preset thresholds,

$$r_{\mathrm{fall},t} = -\mathbf{1}\left[|g^B_{x,t}| > \tau_x \ \vee \ |g^B_{y,t}| > \tau_y\right]. \tag{16}$$

The predator wins when the planar predator–prey distance enters the capture radius,

$$r^{\mathrm{pred}}_{\mathrm{cap},t} = \mathbf{1}\left[d_{xy}(p_t, q_t) < d_{\mathrm{cap}}\right], \tag{17}$$

whereas the prey receives sparse survival rewards for either reaching the safe zone or surviving until timeout,

$$r^{\mathrm{prey}}_{\mathrm{surv},t} = \mathbf{1}[p^{\mathrm{prey}}_t \in \mathcal{Z}_{\mathrm{safe}}] + \mathbf{1}[t \geq T_{\max}]. \tag{18}$$

We additionally include small-weight locomotion regularizers, please check the Appendix:B.

3.6 Privileged-State Estimator

The estimator maps deployable sensory inputs to a latent representation and supervised opponent-state predictions. For each agent, the estimator f_ϕ takes the LiDAR point set $\mathcal{P}^i_t$ and proprioceptive history $\mathcal{H}^i_t$ as input and produces

$$f_\phi(\mathcal{P}^i_t, \mathcal{H}^i_t) = \left(z^i_t, \hat{\boldsymbol{r}}^i_{\mathrm{pos},t}, \hat{\boldsymbol{r}}^i_{\mathrm{vel},t}, \hat{v}^i_t\right), \tag{19}$$

experiment	num_envs per rank	compute	iterations	num seeds
privileged flat	12000	2× H100	1000	4
privileged complex	10000	3× H100	800	2
vision-estimator flat (partial)	4000	3× H100	500	3
vision-estimator flat (full)	4000	3× H100	200	1

Table 1: All experiments ran in our setup.

where z_t^i is the actor latent, $\hat{\boldsymbol{r}}_{\text{pos}}^i$ and $\hat{\boldsymbol{r}}_{\text{vel}}^i$ are predicted opponent relative position and velocity, and $\hat{v}_t^i$ is a visibility logit indicating whether the opponent can be sensed by the LiDAR.

Architecture. The LiDAR cloud is encoded by a DP3d-style PointNet that first applies transforms on the pointset using MLP, apply $\mathrm{LayerNorm}$, aggregate result using $\mathrm{MaxPool}$, and apply a final projection to get the tokens.

$$e_{\text{lidar},t}^i = \text{Proj}\left(\max_{k=1,\dots,N} \text{MLP}_{\text{pt}}(\boldsymbol{p}_{t,k}^i)\right). \tag{20}$$

The proprioceptive history is encoded by a temporal convolutional network:

$$e_{\text{prop},t}^i = \text{TCN}(\mathcal{H}_t^i). \tag{21}$$

The two tokens are fused by a small Transformer encoder and passed through a GRU:

$$y_t^i = \text{Transformer}\left([e_{\text{lidar},t}^i,\, e_{\text{prop},t}^i]\right), \qquad h_t^i = \text{GRU}(y_t^i, h_{t-1}^i). \tag{22}$$

Linear heads on h_t^i produce z_t^i, $\hat{\boldsymbol{r}}_{\text{pos},t}^i$, $\hat{\boldsymbol{r}}_{\text{vel},t}^i$, and $\hat{v}_t^i$.

Targets and loss. All supervised targets are defined in the agent's egocentric yaw-aligned frame. This avoids requiring absolute heading, which is not observable from an egocentric LiDAR scan. Since opponent position and velocity are meaningful only when the opponent is visible, these regression losses are applied only on visible timesteps. Let $v_t^\star \in \{0, 1\}$ denote the ground-truth LiDAR visibility label. The estimator loss is

$$\mathcal{L}_{\text{est}} = w_b\ \text{BCE}(\hat{v}_t^i, v_t^\star) + \mathbf{1}[v_t^\star]\left(w_p \left\|\hat{\boldsymbol{r}}_{\text{pos},t}^i - \boldsymbol{r}_{\text{pos},t}^{i,\star}\right\|_2^2 + w_v \left\|\hat{\boldsymbol{r}}_{\text{vel},t}^i - \boldsymbol{r}_{\text{vel},t}^{i,\star}\right\|_2^2\right). \tag{23}$$

The actor receives $\text{sg}(z_t^i)$, so policy gradients do not update the estimator. Thus, representation learning is driven only by $\mathcal{L}_{\text{est}}$. We use ground truth visibility at early stages of training to help converging.

4 Experimental Setup

Simulation Platform & Setup. We train two Unitree Go2 quadrupeds in IsaacLab/IsaacSim in a 1-vs-1 predator–prey self-play task. Each agent controls 12 joint-position targets through a low-level PD controller. Episodes terminate when the predator reaches the prey within $0.55\,\text{m}$, when the prey reaches the safe zone, when the episode times out after 20 seconds, or when a robot falls. We use both a flat/walled arena for debugging and a deterministic structured arena with walls, ramps, rough regions, depressions, and a safe zone.

Training uses thousands of parallel Isaac Gym environments on GPU, with multi-GPU scaling through DDP. Successfully ran experiments are listed in Table:1

Training Curriculum. For the complex terrain, we utilizes a difficulty coefficient to control the terrain height levels, such as short wall heights, ramp/pond slopes. The curriculum is promotion-only, and monitored by a few factors like locomotion success rate, prey win rate, and velocity reward magnitude of the two agents.

Algorithm Parameters. Both agents are trained jointly with self-play on their own PPO and estimator. Rollouts contain 32 steps and are reused for 4 PPO epochs with 4 minibatches. We use PPO clip ratio 0.2, discount $\gamma = 0.99$, GAE parameter $\lambda = 0.95$, and learning rate 5×10^{-4}.

5 Results

Throughout the project, we evaluate **privileged-info policy** as a conceptual ground of our survival self-play reward, **partial vision-estimator** which uses estimated privileged state except for the hardest opponent-relative info, and **full vision** which uses all onboard-ready informations. A table of all experiment ran can be found at Table:1.

Note that due to the heavy simulation of full vision policy (only 3000 envs per rank on H100, more than 140s per iteration), a convergent training of full vision policy is beyond our budget, though we believe that this undertrained experiment ($\tilde{2}00$ iters) still means something in our pipeline. We tried our best to make the opponent-estimation work, and please check the Appendix:C for the detail.

We evaluate whether simultaneous self-play with the proposed survival-instinct reward can produce meaningful chase–evasion strategies between two agents and around scene obstacles. We first study a privileged-information policy, where the actor has access to the opponent's relative state. This setting serves as a conceptual grounding: if self-play and the reward are sufficient, the agents should learn nontrivial pursuit and escape behaviors even before introducing perception bottlenecks. We then evaluate vision-based variants that remove privileged opponent information from the actor and replace it with estimator outputs, moving toward a sim-to-real deployable pipeline.

Quantitatively, we report locomotion stability, average velocity, same-era predator–prey win rates, and cross-checkpoint win-rate matrices. Cross-checkpoint evaluation is particularly important in self-play: a policy can appear strong against its current opponent while overfitting to a local strategy. By evaluating newer checkpoints against older opponents, we test whether training improves the global strategic strength of the agents rather than only producing cyclic adaptations. Qualitatively, we visualize the learned policies and inspect whether the agents develop emergent strategies beyond the explicit reward terms.

5.1 Experimental Setup

Table 1 summarizes all experiments used in our evaluation. For experiments with multiple random seeds, we report mean and standard deviation across seeds. For single-seed experiments, we report the raw value without seed variance.

5.2 Quantitative Evaluation

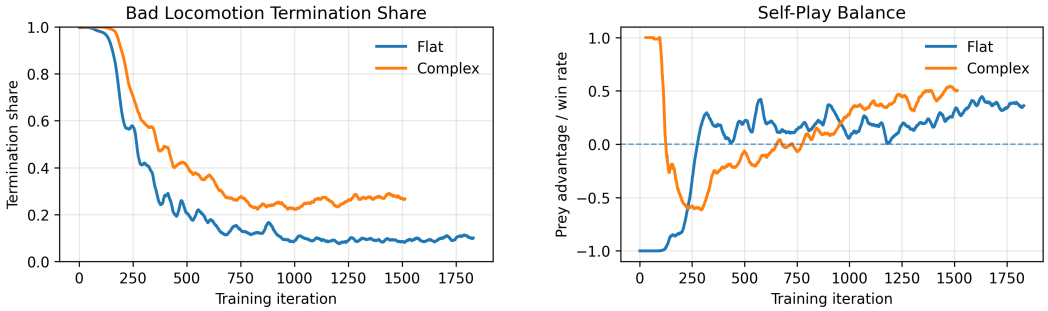
Locomotion stability. We first measure whether each policy remains dynamically stable during self-play. Table 2 reports the locomotion failure rate, defined as the fraction of episodes terminated by falling, invalid collision, or excessive body tilt. The privileged policy learns stable locomotion quickly, reducing the failure rate to around 10% after approximately 1000 iterations on flat terrain and around 21% after approximately 800 iterations on complex terrain. This can also been seen in Figure 3a with two example training runs. The partial vision-estimator policy also improves consistently and reaches below 20% failure rate, indicating that the estimator latent provides enough information for stable locomotion when the hardest-to-estimate opponent relative velocity is omitted.

The full vision-estimator setting remains substantially harder. Since the estimator is initially noisy, the policy receives unstable opponent-state features during early RL updates. In our current experiments, we have not yet reduced the full-vision locomotion failure rate below 40%, due to both the inherent difficulty of the task and insufficient training. We believe this setting requires more parallel environments, longer training, and potentially a stronger estimator pretraining or curriculum before the policy can consistently exploit the latent representation.

Average velocity. The average planar velocity captures whether the agents are merely static or actively engage in chase–evasion. As shown in Table 3, the privileged policy achieves the best chasing–evation velocities. The drop from privileged to vision-estimator policies is expected because the latter must infer opponent information from noisy egocentric sensing. Nevertheless, the partial

Setting	Terrain	Locomotion Failure rate	Iteration
Privileged	Flat	$10.03\% \pm 0.95$	1000
Privileged	Complex	$21\% \pm 0.028$	800
Vision-estimator (partial)	Flat	$26.2\% \pm 0.044$	500
Vision-estimator (full)	Flat	$> 75\%$	200

Table 2: Locomotion stability across training settings.



(a) Bad locomotion termination

(b) Self-play balance

Figure 3: Training curves for terrain comparison.

vision model maintains nontrivial motion speed, suggesting that the learned estimator latent is useful for policy learning.

Same-era win rates. We evaluate predator and prey policies from the same training stage against each other. Figure 3b shows an example of the privileged win rates over time in two example training runs, and Table 4 reports the resulting win rates. In same-era evaluation, a roughly balanced win rate can indicate that self-play maintains competitive pressure between the two roles, while large imbalance may suggest that one role learns faster or that the reward coefficients favor one side. In our experiments, the privileged setting remains balanced with prey attains slight advantage after epoch 1000. For the vision-estimator setting, the predator learns significantly faster than the prey because when both agent moves slowly at the early stage, the predator only needs to learn to move toward the prey, while the undirectional escape route for prey is harder to learn.

Cross-checkpoint win rates. To test whether self-play improves strategic strength over training, we evaluate policies against opponents from different training checkpoints. Figure 4 shows cross-checkpoint prey win-rate matrices for the privileged complex-terrain setting and the partial vision flat-terrain setting.

The privileged complex-terrain matrix shows a clear asymmetric learning process. Early predators are weak: the checkpoint-0 predator loses to almost all prey checkpoints, with prey win rates above 0.8. As predator training progresses, newer predators reliably dominates early prey checkpoints, giving near-zero prey win rates against prey checkpoints 0 and 200. This suggests that the predator quickly learns a generally stronger chasing strategy. However, the prey learns more slowly. Once the prey has sufficiently experienced the terrain and opponent distribution, roughly after the mid-stage checkpoints and clearly by checkpoints 800–1000, it recovers an advantage even against late-stage predators. This indicates that effective evasion requires more training than direct pursuit, likely because the prey must learn both locomotion and map-dependent escape strategies.

The partial vision setting follows the same trend but is less mature. Later predators strongly dominate early prey checkpoints, indicating that pursuit remains learnable even with estimator-based observations. In contrast, prey improvement is delayed: only the latest prey checkpoint becomes consistently competitive. Since this experiment was only trained up to checkpoint 500 (due to the heavy simulation of vision environment), the prey has less time to catch up. Together with the higher locomotion failure rate in the vision setting, this suggests that the prey side is more sensitive to estimator noise and requires longer training or more parallel environments to reach the strategic stability observed in the privileged setting.

Setting	Terrain	Avg. velocity prey	Avg. velocity pred	Notes
Privileged	Flat	0.72 m/s	0.66 m/s	Stable chase–evasion
Privileged	Complex	0.51 m/s	0.42 m/s	Obstacle-aware behavior
Vision-estimator (partial)	Flat	0.56 m/s	0.51 m/s	No privileged opponent velocity
Vision-estimator (full)	Flat	0.28 m/s	0.35 m/s	undertrained

Table 3: Average planar velocity during evaluation, inferred from average velocity reward,

Setting	Terrain	prey win rate
Privileged	Flat	$61.5\% \pm 5.22$
Privileged	Complex	$70.9\% \pm 4.61$
Vision-estimator (partial)	Flat	$56.6\% \pm 4.23$
Vision-estimator (full)	Flat	5%

Table 4: Same-era predator–prey win rates. We exclude the termination due to locomotion failure.

Overall, the quantitative results suggest three conclusions. First, privileged self-play is sufficient to produce stable and competitive chase–evasion behavior under the proposed reward. Second, partial vision-estimator training preserves much of the locomotion stability while removing most privileged actor information. Third, the full vision-estimator setting is the main remaining bottleneck: its early estimator noise causes high locomotion failure, and scaling the number of environments is crucial for reliable convergence, but the heavy simulation and time constraint prevents us from scaling both env nums and iterations.

5.3 Qualitative Evaluation

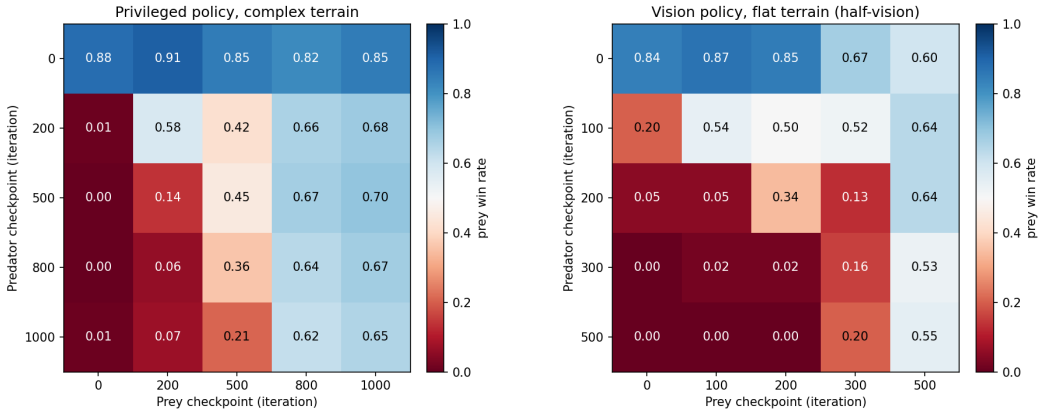
Figure 5 shows representative rollout snippets from both privileged training and partial-vision training. Although the reward only directly specifies gap progress, capture, survival, and locomotion regularization, the learned policies exhibit structured pursuit–evasion behaviors that are not explicitly scripted.

Interception. In Fig. 5(a), the predator does not simply chase the prey's current position. Instead, it cuts toward the prey's future path and produces an interception trajectory. This indicates that the predator learns to exploit the prey's motion trend and choose a shorter capture route, rather than tailgating the prey.

Obstacle-assisted evasion. In Fig. 5(b), the prey uses the short wall as a strategic obstacle. It performs an agile turnaround around the wall, while the predator's direct pursuit path is blocked. This suggests that the prey learns to use terrain geometry as a resource for increasing the effective path length, rather than merely avoiding obstacles.

Strategy evolution. Figures 5(c) and 5(d) illustrate why observing policy behavior across checkpoints is important in self-play. In the earlier-stage rollout in Fig. 5(c), the predator charges at full speed, but the prey suddenly turns using the wall. The predator cannot react quickly enough, hits the wall, and the prey escapes. In the later-stage rollout in Fig. 5(d), the same escape pattern is no longer effective: the predator brakes in advance, observes the prey's corner turn, and then executes a tighter catch. This shows a clear counter-strategy emerging over training.

Overall, these qualitative examples clearly shows that self-play does not only improve low-level locomotion or instantaneous reward maximization, but also drives a competition between evasion strategies and pursuit counter-strategies. The partial-vision examples further suggest that similar behaviors can begin to emerge even when the actor relies on estimator-based observations rather than privileged opponent state, although the resulting strategies are less stable than in the privileged setting.



(a) Cross-play prey win rate for privileged complex terrain.

(b) Cross-play prey win rate for partial vision flat terrain

Figure 4: LiDAR observation used by the opponent-state estimator. Each ray is represented by a normalized 3D endpoint together with range and angular information in the egocentric frame.

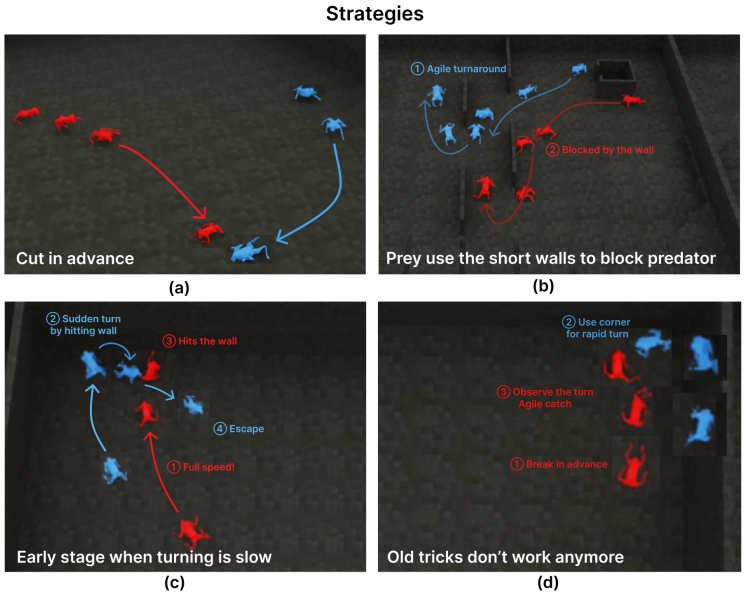


Figure 5: Qualitative examples of emergent predator–prey strategies from privileged and partial-vision rollouts. Red denotes the predator and blue denotes the prey. Faded bodies show temporal traces, and arrows indicate the main motion directions.

6 Discussion

6.1 Interaction Between Terrain and Self-Play

Overall, these results suggest that the main challenge in predator-prey quadruped self-play is not simply learning to move to win, but balancing locomotion, strategy, and opponent adaptation at the same time. The flat-terrain setting provides a useful sanity check because agents can focus on direct pursuit and evasion without major environmental constraints. In contrast the complex terrain setting reveals whether the learned behavior depends on meaningful terrain interaction, since obstacles create opportunities for interception, escape, and failure. The fact that agents exhibit different behaviors across these settings suggests that the task is shaped by both the opponent and the environment, rather than by a fixed locomotion policy alone. This supports our broader motivation

that competitive self-play can turn parkour terrain into a strategic environment, where physical control and decision-making are learned jointly.

6.2 Limitations

Our biggest limitation was the computational cost of training non-privileged/full-vision policies. Unlike the privileged setting, where agents can directly access compact simulator state, the non-privileged setting requires generating LiDAR observations at every step (3600 rays per env and per agent). In particular, ray tracing a point cloud for each agent and each observation is expensive when scaled across thousands of parallel environments. This substantially reduces training throughput and makes convergence much slower.

This bottleneck is exacerbated in our complex terrains with obstacles, where the policy must learn both stable locomtion and perception-based pursuit-evasion behavior under partial observability. As a result, the non-privileged models required significantly more compute and training time than privileged models before meaningful behavior emerges. In practice, this limited how extensively we could tune the vision pipeline and evaluation fully converged non-privileged policies.

6.3 Future Work

Our current results suggest that one-stage predator-prey self-play can produce meaningful pursuit and evasion behavior, but there are still several directions to explore.

First, future work can improve and further train the non-privileged perception pipeline when we have more compute. Our current non-privileged setting is limited by the computational cost of generating LiDAR point clouds at scale, and the resulting policies are harder to train than privileged policies. A more efficient perception pipeline could try lower-resolution LiDAR, cached terrain representations, or other perception observations (e.g. camera images) that preserve the most important opponent and terrain information while reducing simulation overhead.

Second, the environment can be expanded with richer game objectives. Adding safe-zone rewards can induce more complex strategies around reaching or guarding these zones. Future environments could include multiple safe zones, moving goals, limited visibility regions, or resource constraints to provide more variables for long-horizon strategy.

Third, future work could explore adding more agents such as multiple predators or prey to create heard-like settings. In addition to the adversarial behavior, this could encourage cooperative behavior, role specialization, group evasion, and more complex pursuit strategies. These settings would better match natural predator-prey dynamics and make self-play a stronger automatic curriculum for more complex behavior.

Finally, the system can be moved to sim-to-real deployment. Current training uses idealized sensing, terrain, and contact dynamics. Future work can add domain randomization over friction, mass, latency, terrain geometry, sensor noise, and actuation delay to make learned behaviors more physically robust. This would help competitive quadruped self-play transfer from simulation to real robot demonstrations.

7 Conclusion

In this project, we studied predator-prey self-play to learn terrain-aware quadruped parkour behavior. We built a simulated Unitree Go2 environment where a predator and prey learn to pursue, evade, and exploit terrain through PPO-style self-play with dense instinct rewards and sparse survival objectives. Our results show that, with careful reward design, agents can learn stable locomotion and adversarial strategy jointly, producing behaviors such as anticipatory pursuit, obstacle-aware evasion, and strategy around walls and corners.

Our results suggest that competitive self-play can serve as an automatic curriculum for quadruped behavior, where the opponent is an adapting source of difficulty and the terrain is a strategic resource. While our current system remains limited by ideal simulation assumptions and the computational cost of non-privileged perception training, it is a promising avenue for studying interactive, perception-

aware legged locomotion. Future work can build on our framework through richer objectives such as safe zones, multi-agent settings, and bringing our simulations into the real world.

8 Team Contributions

- **Haoyue Xiao:** I led the codebase development and worked on the environment terrain setup, and reward design. I helped set up the estimator and ran privileged experiements. I also helped write the proposal, milestone, poster, and final report, and helped present the poster.
- **Shatong Zhu:** I worked on the self-play MAPPO training framework, checkpointing, and initial evaluation metrics. I also helped write the proposal, milestone, and final report. I also tried running experiments for multi-stage training that went unused.
- **Edward Lee:** I helped with the estimator and perception side of the project and ran and evaluated vision training runs. I also helped write the proposal, milestone, poster, and final report, and helped present the poster.

Changes from Proposal We now believe that a full one-staged full-vision policy is beyond our compute and time budget in this project, so our discussion and results heavily reply on the privileged policy which serve as the conceptual verifier of our survival self-play rewards. Though we think given enough compute, a full vision policy is both possible and interested to explore.

References

Xuxin Cheng, Kexin Shi, Ananye Agarwal, and Deepak Pathak. 2024. Extreme Parkour with Legged Robots. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 11443–11450. doi:10.1109/ICRA57147.2024.10610200

Shixin Luo, Songbo Li, Ruiqi Yu, Zhicheng Wang, Jun Wu, and Qiuguo Zhu. 2024. PIE: Parkour with Implicit-Explicit Learning Framework for Legged Robots. *IEEE Robotics and Automation Letters* 9, 11 (2024), 9986–9993. doi:10.1109/LRA.2024.3459797

Nikita Rudin, David Hoeller, Philipp Reist, and Marco Hutter. 2022. Learning to Walk in Minutes Using Massively Parallel Deep Reinforcement Learning. In *Proceedings of the 5th Conference on Robot Learning (Proceedings of Machine Learning Research, Vol. 164)*, Aleksandra Faust, David Hsu, and Gerhard Neumann (Eds.). PMLR, 91–100. https://proceedings.mlr.press/v164/rudin22a.html

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. *arXiv preprint arXiv:1707.06347* (2017).

David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy Lillicrap, Karen Simonyan, and Demis Hassabis. 2018. A General Reinforcement Learning Algorithm that Masters Chess, Shogi, and Go through Self-Play. *Science* 362, 6419 (2018), 1140–1144. doi:10.1126/science.aar6404

Dhruva Tirumala, Markus Wulfmeier, Ben Moran, Sandy Huang, Jan Humplik, Guy Lever, Tuomas Haarnoja, Leonard Hasenclever, Arunkumar Byravan, Nathan Batchelor, Neil Sreendra, Kushal Patel, Marlon Gwira, Francesco Nori, Martin Riedmiller, and Nicolas Heess. 2025. Learning Robot Soccer from Egocentric Vision with Deep Reinforcement Learning. In *Proceedings of The 8th Conference on Robot Learning (Proceedings of Machine Learning Research, Vol. 270)*, Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard (Eds.). PMLR, 165–184. https://proceedings.mlr.press/v270/tirumala25a.html

Zhen Wu, Xiaoyu Huang, Lujie Yang, Yuanhang Zhang, Koushil Sreenath, Xi Chen, Pieter Abbeel, Rocky Duan, Angjoo Kanazawa, Carmelo Sferrazza, Guanya Shi, and C. Karen Liu. 2026. Perceptive Humanoid Parkour: Chaining Dynamic Human Skills via Motion Matching. *arXiv preprint arXiv:2602.15827* (2026).

Yanjie Ze, Gu Zhang, Kangning Zhang, Chenyuan Hu, Muhan Wang, and Huazhe Xu. 2024. 3D Diffusion Policy: Generalizable Visuomotor Policy Learning via Simple 3D Representations. In *Proceedings of Robotics: Science and Systems (RSS)*.

Ziwen Zhuang, Zipeng Fu, Jianren Wang, Christopher G. Atkeson, S"oren Schwertfeger, Chelsea Finn, and Hang Zhao. 2023. Robot Parkour Learning. In *Proceedings of The 7th Conference on Robot Learning (Proceedings of Machine Learning Research, Vol. 229)*, Jie Tan, Marc Toussaint, and Kourosh Darvish (Eds.). PMLR, 73–92. `https://proceedings.mlr.press/v229/zhuang23a.html`

A Implementation Details

The implementation uses an IsaacLab/IsaacSim backend with two replicated Unitree Go2 articulations per environment. Simulation runs at 200 Hz with a control decimation of 4, so policies act at 50 Hz. Each policy output is interpreted as a normalized 12-dimensional joint-position command and converted to torques by the environment-side PD controller. We disable domain randomization for the main experiments and use deterministic terrain generation so that environment layouts are reproducible across runs.

Each agent owns an independent actor–critic network and optimizer. The actor and critic are both MLPs with hidden dimensions $[512, 256, 128]$ and ELU activations. The Gaussian policy uses a learned diagonal standard deviation initialized to 1.2. The critic is asymmetric and, in the decomposed setting, predicts separate value heads for shaping, sparse-event, and regularization reward groups; these heads are summed when computing PPO advantages. PPO uses 32-step rollouts, 4 epochs, 4 minibatches, clip ratio 0.2, $\gamma = 0.99$, GAE $\lambda = 0.95$, entropy coefficient 0.01, gradient clipping at 1.0, and learning rate 5×10^{-4}.

For the privileged setting, the actor observation contains proprioception, relative opponent state, role encoding, local height scans, wall-distance features, absolute heading encoding, time-to-go, and a 10-step history. For the vision setting, privileged actor inputs are removed: the actor receives only onboard proprioception, role, time-to-go, proprioceptive history, and a detached estimator latent. The estimator reconstructs per-ray point features from LiDAR ranges and fixed ray directions, encodes the point cloud with a PointNet-style network, fuses it with proprioceptive history using a Transformer, and maintains temporal memory with a GRU. Its supervised heads predict egocentric relative opponent position, egocentric relative opponent velocity, and opponent visibility.

B Reward Terms

The per-step reward of agent i is a weighted sum $r_t^i = \sum_k \lambda_k \, r_{k,t}^i$, where the terms r_k are grouped into the three critic heads used by the decomposed-return critic (*shaping*, *sparse event*, *regularization*). Negative λ_k denote penalties; the total per-step reward is clipped to be non-negative (`only_positive_rewards`). Notation: $\boldsymbol{q}, \dot{\boldsymbol{q}}$ joint positions/velocities, $\boldsymbol{\tau}$ joint torques, $\boldsymbol{a}$ action, $\boldsymbol{v}, \boldsymbol{\omega}$ base linear/angular velocity, $\boldsymbol{g}$ projected gravity, $\boldsymbol{q}^{\text{def}}$ the default pose, Δt the control step, and $\mathbf{1}[\cdot]$ the indicator.

Task and event rewards. The two core game-objective terms are defined in Section 3.5 (their closed forms are given there and omitted here):

- **Velocity (gap-rate) shaping** r_{vel}, $\lambda = 7.0$ — see Section 3.5.
- **Catch** (predator), $\lambda = 50.0$ — time-discounted sparse catch reward; see Section 3.5.

The remaining event rewards are simple terminal indicators:

- **Prey-caught penalty** (prey): $r = \mathbf{1}[\text{prey caught}]$, $\lambda = -10.0$.
- **Safe-zone escape** (prey): $r = \mathbf{1}[\text{prey in safe region}]$, $\lambda = 1000.0$.
- **Survival** (prey): $r = \mathbf{1}[\text{episode timeout}]$, $\lambda = 50.0$.

Regularization. Standard locomotion-smoothness and posture penalties (applied to both agents):

Term	Expression r_k	Coefficient λ_k
Vertical velocity	v_z^2	-1.0
Angular velocity	$\|\boldsymbol{\omega}_{xy}\|^2$	-0.01
Orientation	$\|\boldsymbol{g}_{xy}\|^2$	-1.0
Joint acceleration	$\left\|(\dot{\boldsymbol{q}}_{t-1} - \dot{\boldsymbol{q}}_t)/\Delta t\right\|^2$	-2.5×10^{-7}
Collision	$\sum_{b\in\mathcal{B}} \mathbf{1}[\|\boldsymbol{f}_b\| > 0.1]$	-10.0
Action rate	$\|\boldsymbol{a}_t - \boldsymbol{a}_{t-1}\|^2$	-0.01
Torque rate	$\|\boldsymbol{\tau}_t - \boldsymbol{\tau}_{t-1}\|^2$	-1.0×10^{-7}
Torque	$\|\boldsymbol{\tau}\|^2$	-1.0×10^{-5}
Hip deviation	$\sum_{j\in\text{hip}} q_j^2$	-1.5
Joint deviation	$\|\boldsymbol{q} - \boldsymbol{q}^{\text{def}}\|^2$	-0.1
Foot stumble	$\mathbf{1}\left[\exists\,\text{foot} : \|\boldsymbol{f}_{xy}\| > 4\,\|f_z\|\right]$	-1.0

Table 5: Regularization reward terms and coefficients. $\boldsymbol{f}_b$ is the contact force on penalized body $b \in \mathcal{B}$; $\boldsymbol{f}_{xy}, f_z$ are the tangential/normal foot contact forces.

C Notes on Making Opponent Estimation Work

Getting the vision-based opponent estimator to train end-to-end required substantially more engineering than the privileged setting, and we report the attempt for completeness. The first obstacle was perception itself: a standard ray-caster only sees the static terrain, so the moving opponent was invisible to the LiDAR; sensing it required IsaacLab's multi-mesh ray-caster (only available after a simulator upgrade), which tracks the opponent's body mesh per environment but rebuilds (and vertex-merges) that mesh once *per environment* at startup, an $O(\text{num\_envs})$ CPU cost that made large-scale and multi-GPU launches stall for minutes and forced us to serialize initialization and cap the environment count. We then found several correctness pitfalls that had to be fixed before the estimator could learn anything useful: the supervision targets and the LiDAR scan had to live in the same egocentric (yaw-only) frame rather than the world frame (absolute heading is unobservable from an egocentric scan and was dropped); the privileged opponent state had to be removed from *both* the actor observation and the estimator's proprioceptive input to prevent the network from trivially reading off the answer; the recurrence had to use stored-state truncated BPTT so the GRU's hidden trajectory stayed consistent between rollout and update under PPO's shuffled minibatches; and the relative-state heads had to be supervised only when the opponent was actually visible, with an explicit visibility classifier, so that "not seen" did not corrupt the regression. We also had to shape the LiDAR beam pattern to concentrate hits on the opponent and trade off point-cloud subsampling (cheaper, but drops a sparse/distant opponent) against keeping the full cloud (faithful, but compute- and memory-bound). With these fixes the estimator trains and recovers opponent visibility and relative state in simulation; however, the simulator-side init cost and multi-process contention kept vision self-play at a smaller scale than the privileged runs, so the vision results we report are preliminary and the privileged policy remains our primary result.