Wheeled Lab: Modern Sim2Real for Low-cost, Open-source Wheeled Robotics

Tyler Han* **Preet Shah** **Sidharth Rajagopal** **Yanda Bao**
Sanghun Jung **Sidharth Talia** **Gabriel Guo** **Bryan Xu** **Bhaumik Mehta**
Emma Romig **Rosario Scalise** **Byron Boots**

University of Washington

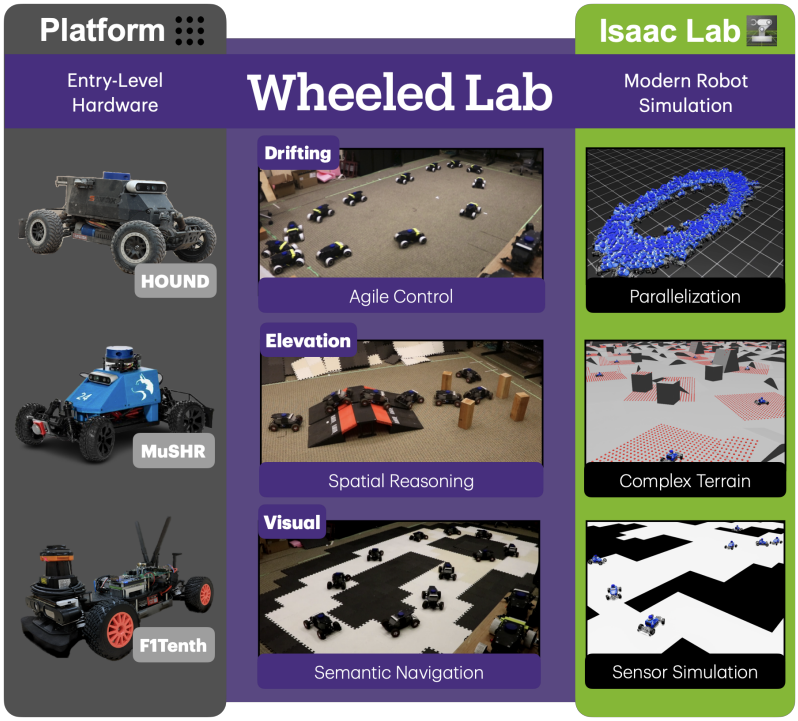


Figure 1: Wheeled Lab bridges popular low-cost open-source wheeled platforms [1, 2, 3] with the research-backed robotics ecosystem Isaac Lab [4] to support and democratize access to cutting-edge tools in modern robotics.

Abstract: Reinforcement Learning (RL) has been pivotal in recent robotics milestones and is poised to play a prominent role in the future. However, these advances can rely on proprietary simulators, expensive hardware, and a daunting range of tools and skills. As a result, broader communities are disconnecting from the state-of-the-art; education curricula are poorly equipped to teach indispensable modern robotics skills involving hardware, deployment, and iterative development. To address this gap between the broader and scientific communities, we contribute Wheeled Lab, an ecosystem which integrates accessible, open-source wheeled robots with Isaac Lab, an open-source robot learning and simulation framework, that is widely adopted in the state-of-the-art. To kickstart research and education, this work demonstrates three state-of-the-art zero-shot policies for small-scale RC cars developed through Wheeled Lab: controlled drifting, elevation traversal, and visual navigation. The full stack, from hardware to software, is low-cost and open-source. Videos and additional materials can be found at: https://uwrobotlearning.github.io/WheeledLab/

*Corresponding author: than123@uw.edu

9th Conference on Robot Learning (CoRL 2025), Seoul, Korea.

1 Introduction

The robotics community has made remarkable strides in recent years, boasting advances that enable quadrupeds to hike mountain trails and perform parkour [5, 6], autonomous drones to race at champion-level ability [7], and robot hands to solve Rubik's Cubes [8]. Shared across these feats is simulated policy learning followed by direct deployment in the real world, termed *Sim2Real*.

However, these remarkable Sim2Real innovations currently target expensive, space-intensive, and high-maintenance systems, including quadrupeds, humanoids, drones, and dexterous manipulators. Such platforms make it difficult for the general population of robotics builders and users to access the state-of-the-art techniques they employ, such as domain randomization and sensor simulation [7, 9, 10, 6].

In contrast, small-scale autonomous RC cars offer low-cost* yet performative platforms that can leverage modern robotics methods, as we demonstrate in this paper. Used by classrooms, national racing competitions, and enthusiasts, platforms like F1Tenth [3] have established themselves as go-to, entry-level platforms for beginning roboticists and researchers alike. Crucially, these platforms still provide sufficient onboard real estate to house components like the battery and compute necessary for modern algorithms, as well as sufficient mobility to exercise complex dynamics and environment interactions like drifting or off-roading [12, 13, 14, 15]. In short, small-scale wheeled robots have high algorithmic potential at relatively low cost and hardware complexity.

A survey of existing ecosystems for low-cost wheeled platforms indicates that current simulation capabilities are isolated and outdated (see Table 1). As a result, education [16] and research opportunities [17] currently limit practitioners to model-based or model-free methods with low-fidelity simulation. With the advent of high-fidelity, open-source simulators like Isaac Lab [4], which are geared toward modern robotic learning, these limited ecosystems no longer need to be the status quo.

This work aims to support education and research by making modern robotics methods available to the broader community. To accomplish this, *we contribute a Sim2Real framework for open-source wheeled robots that we have integrated with a state-of-the-art, research-grounded simulation ecosystem, Isaac Lab.* Modern methods we implement and evaluate include: massive parallelization, domain randomization, sensor simulation, and end-to-end learning. We demonstrate these methods using three policy types trained in simulation and deployed on low-cost, open-source platforms:

1. The **Drifting Policy** (π_{drift}) performs a controlled drift through extensive domain randomization, an approach in modern Sim2Real for tasks with uncertain and unstable dynamics. This is the first work to demonstrate direct Sim2Real transfer for drifting without online fine-tuning.
2. The **Elevation Policy** (π_{elev}) traverses three-dimensional features with spatial reasoning. This demonstration highlights end-to-end training for modern tasks that tightly couple perception and control. Wheeled Lab is the first to provide integration of an accessible elevation-based Sim2Real pipeline with massive parallelization.
3. The **Visual Policy** (π_{vis}) traverses visual features with semantic understanding. This demonstration highlights simulation, end-to-end training, and deployment with cameras, a sensor modality gaining substantial research attention due to its pre-training potential. Wheeled Lab provides a lower-cost alternative for visual Sim2Real compared to existing integrations.

By integrating low-cost, open-source platforms with Isaac Lab, this work addresses a current need in robotics to extend modern principles in Sim2Real to resource-constrained audiences.

*About 3000 USD or less. See [11] for a comparative analysis.

	Sensor Simulation			Agent Physics		Reinforcement Learning			
Ecosystem	Elevation	LiDAR	Camera	Dynamics	Parallelization	Perturbation	Corruption	Community	Platforms
Wheeled Lab	✓	3D	Depth, RGB	16 DOF + Collisions	High	✓	✓	RSL, SB3	HOUND, MuSHR, F1Tenth
AutoDrive [11]	✗	2D	✗	16 DOF	Low	✗	✗	Gym	F1Tenth, Nigel
F1Tenth [3]	✗	2D	✗	Kinematic	✗	✗	✗	Gym	F1Tenth
CARLA [18]	✗	3D	Depth, RGB	16 DOF + Collisions	✗	Steering Only	✓	SB3	✗
Brunnbauer *et al.* [19] (PyBullet)	✗	2D	RGB	12 DOF	✗	✗	✗	✗	F1Tenth
Hamilton *et al.* [20] (Gazebo)	✗	2D	✗	Kinematic	✗	✗	✗	✗	F1Tenth
PIETRA [21] (Chrono)	✓	✗	✗	12 DOF	✗	✗	✗	✗	F1Tenth
RoboTHOR [22]	✗	✗	RGB	Kinematic	✗	✗	✓	AllenAct	Household

Table 1: **Comparisons of existing ecosystems on their capabilities.** Various simulation, learning, and deployment ecosystems have been integrated with accessible wheeled platforms for Sim2Real. However, these ecosystems are noticeably isolated from the research community and, where implemented, have outdated features.

2 Related Work

Considerable prior work in Sim2Real has addressed open-source wheeled robots, including [15, 23, 24, 25, 26, 27, 12, 28]. Some of these works produce one of π_{drift}, π_{elev}, or π_{vis} with partial or full Sim2Real. The individual accomplishments of this research both motivate our choice of demonstrations and help us highlight that *varied state-of-the-art approaches can be coherently integrated under the framework presented here*.

Drifting Policy. Drifting is a challenging task in both optimal control and reinforcement learning (RL) [29, 30, 31]. It is a fundamentally unstable maneuver whose control solutions are traditionally determined by vehicle and ground parameters [32]. In many cases (ours included), researchers are unable to demonstrate a controlled drift for imitation-based methods [27]. These challenges make drifting an appealing task for RL. RL-based approaches have thus far been achievable only through further real-world fine-tuning [27, 12]. *In this work, we show that aggressive domain randomization, perturbation simulation, and massive parallelization can enable direct transfer of a drifting policy.* To reduce the barriers-to-entry, we deliberately distance our methods from techniques such as gain tuning and extensive system identification that require additional domain expertise and tooling.

Elevation Policy. It has become standard practice for locomotion policies on outdoor terrain to be trained with elevation maps [6]. In parallel, autonomous off-road vehicles have also demonstrated the importance of incorporating elevation in model-based methods [14, 33, 34]. Generally, these elevation-based tasks address challenges that arise at the interface of agent morphology and perception, which are often difficult to model. Therefore, lower-cost wheeled platforms have traditionally been oriented toward flat terrain [3, 24], revealing a gap in methodology between the broader community and the state-of-the-art. Notable exceptions include [15, 35, 36] but do not offer as extensive community support for RL-based approaches.

Visual Policy. Improvements in camera and photorealistic scene simulation have advanced policies capable of semantic reasoning, bridging the gap between pre-trained visual models and embodied actions [37, 38]. Demonstrations of small-scale platforms with pre-trained visual models have also been investigated [35]. In general, cameras provide an approachable and interpretable sensor modality for modern robotics education and research. Unfortunately, similar to elevation policies, visual policy pipelines on low-cost mobile platforms are not readily available even though the success of proprietary camera-based Sim2Real platforms suggests a clear demand [24]. *This work demonstrates simulation, end-to-end training, and deployment with cameras on a low-cost platform.*

Isaac Lab Simulation Framework. This work builds on Isaac Lab, an open-source, widely adopted, and rapidly growing simulation framework for robotics research [4, 10] that is heavily supported by industry and academia alike [39, 40, 41, 42, 43, 10]. Isaac Lab has become a primary learning framework of choice for recent robotics research due to its impressive performance and ease-of-use features [44, 6, 45, 46, 47].

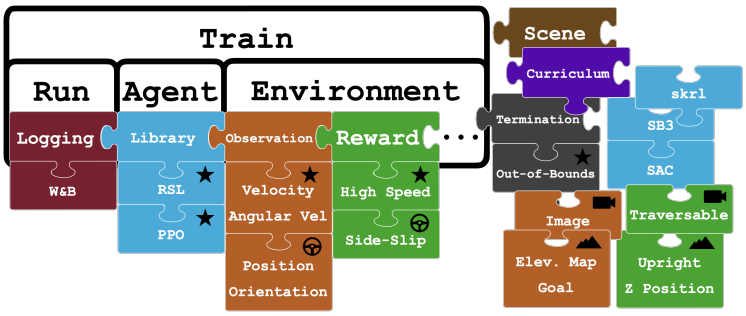


Figure 2: **Modular training framework imagined as the assembly of a puzzle.** Three main components comprise training: `Run`, `Agent`, `Environment`. Outlined puzzle pieces (e.g., `Observation`, `Reward`) are sub-components whose behaviors are defined by the pieces below them. For instance, `High Speed` is a reward setting in all of our RL environments, as denoted by the star (★) icon. Wheel (🛞), camera (📹), and mountain (⛰) icons indicate specific design choices for π_{drift}, π_{vis}, and π_{elev}, respectively. Components shown are non-exhaustive.

3 Implementation

Wheeled Lab bridges low-cost wheeled platforms with Isaac Lab (see Figure 1). We extend Isaac Lab by designing infrastructure and assets which enforce reproducible experimentation. Some of the highest-level abstractions towards this effort are illustrated in Figure 2 (e.g., Scene, Observation, Reward), highlighting the inclusion of run parameters, architecture, and hyperparameters in addition to existing environment configurations. This design helps ensure reproducible, accessible, and portable experimentation settings and results, which is critical for practitioners to build upon existing work.

This work presents reliable and empirically validated baseline configurations of these abstractions, which are amenable specifically to low-cost wheeled platforms. As Sim2Real remains an iterative process, especially in low-cost settings, our implementations consider accessibility factors throughout the entire development process: limited onboard compute and sensing, visual diversity, system identification feasibility, access to evaluation terrain, access to open-source packages, and access to training resources. Further implementation details for each task can be found in Appendix B.

4 Experiments

Figure 3: Several autonomy platforms shared among students in a robotics class.

While we contribute the novelties in applied RL as discussed in Section 1, our experiments aim to highlight the disparity in open-source methodologies between the broader and scientific communities. Thus, we compare policies trained through Wheeled Lab to baselines which reflect what training ecosystems currently provide to low-cost wheeled platforms, as shown in Table 1. This includes ablations over techniques such as domain randomization, perturbations, parallelization capacity, and architectures. $\overline{(\cdot)}$ denotes the baseline version of a policy (e.g., $\overline{\pi}_{\text{drift}}$). Across all policies, we use Proximal Policy Optimization (PPO) [48] as implemented in the RSL RL library to train the agent in simulation [49].

We implement, deploy, and evaluate three state-of-the-art policy types: π_{drift}, π_{elev}, and π_{vis} on the HOUND and MuSHR platforms. Settings for drifting and elevation tasks are shown in Table 2. All other configuration settings not mentioned (e.g., rewards, actuator parameters, articulations) can be assumed to be the same between π and $\overline{\pi}$.

Our experiments employ the HOUND [1] for π_{drift} and the MuSHR [2] for π_{elev} and π_{vis}. The HOUND's estimated total price is about 3000 USD for a complete autonomy package with sensing (i.e., LiDAR, RGB) and compute (i.e., Jetson Orin NX). The MuSHR is estimated at 930 USD. A single NVIDIA RTX 3080 GPU is used for training. Isaac Lab also offers cloud-based training for broader audiences without personal GPU access.

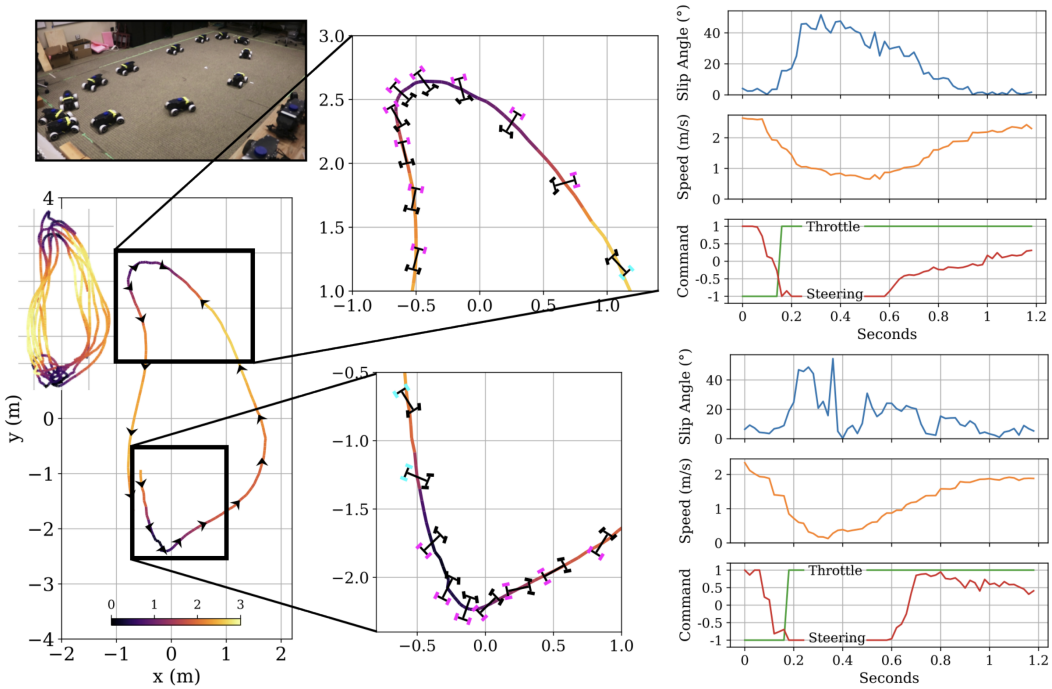


Figure 4: **A control strategy arises for drifting through turns. Left:** (Overlay: colored trajectories over multiple runs.) One run is spotlighted for this visualization. Arrows indicate the direction of travel, and the colorbar denotes speeds (in m/s). **Middle:** Magnification of drifted turns with markers visualizing vehicle orientation, steering (front wheel direction), and throttle (rear wheels; blue: -1, pink: $+1$). **Right:** Slip angle, speed, steering, and throttle are plotted for each turn. To initiate the drift, the platform quickly cuts the throttle to destabilize the rear wheels and then steers inwards sharply to throw them outwards. With the platform now facing the track center, it throttles through the remainder of the turn while counter-steering to control its residual angular momentum from the initial maneuver. The entire sequence occurs in just over 1 second. Visualization inspired by [31].

4.1 Drifting

Background. Drifting is characterized by a vehicle's *side-slip angle*, i.e., the angle between the heading and the velocity, as it maneuvers through a turn. In high-performance driving, drifting allows the vehicle to take much sharper turns at higher speeds than would be otherwise allowable by the no-slip turn radius [50]. The total mass, center-of-mass, friction properties of the surface and the wheels, actuator properties, state estimation, steering linkages, and suspension are all relevant factors in the narrow dynamics of a controlled drift.

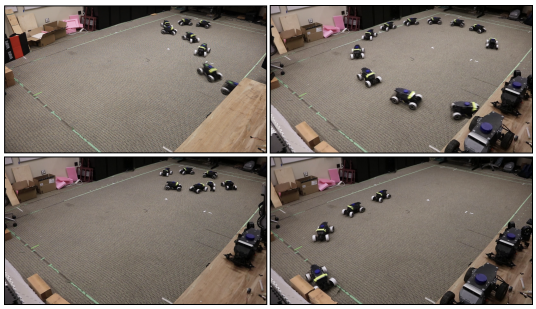


Figure 5: **Captured trajectories from the baseline drift policy in real experiments**. The baseline is unable to complete a valid turn without crashing or spinning out.

The sensitivity of these dynamics becomes especially obvious when we (researchers) attempt the task manually despite the deceptively simple action space (see website for videos). When the maneuver is not executed correctly, the vehicle spins uncontrollably off track, losing traction (also known as a *spin out*) and usually requiring a reset.

Task. For our evaluation, the task involves maintaining speed while minimizing the cross-track distance from an oval racing line without spinning out.

Setup. State estimation is provided primarily by onboard Visual-Inertial-Odometry (VIO). We use a motion capture system for evaluation and occasional (1 hz) integrator-drift correction due to space

limitations of our indoor testing environment. Successful VIO-only runs not used for data collection are shown on our website. To enable the platform to be physically capable of drifting, tape is wrapped around the platform's rubber wheels to reduce friction and converted the base to a rear-wheel drive.

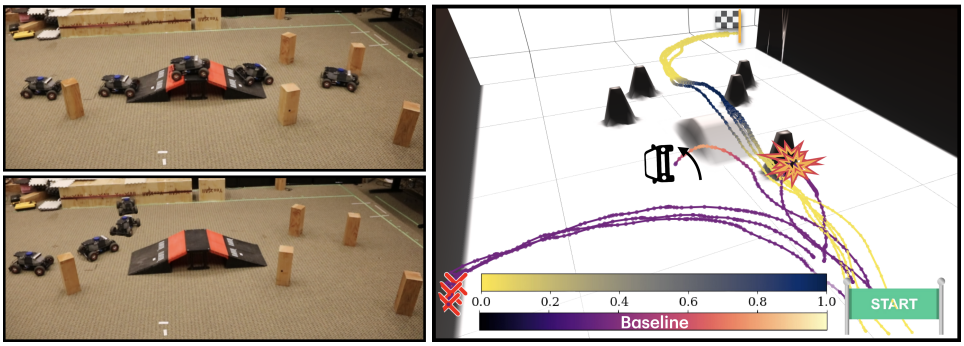
Results. We find that the baseline is unable to make a stable turn, much less complete a full lap (Figure 5). However, it occasionally attempts to counter-steer despite showing no indication during training. Baseline training runs consistently fail to discover the drifting mode. Motor cogging, noisy state estimates, constant slipping, and steering biases cause significant covariate shift.

Policy	Corruption	Perturbation	DR	# Envs	Epochs
$\overline{\pi}_{\text{drift}}$	✓	✗	✗	64	5000
π_{drift}	✓	✓	✓	1024	5000
$\overline{\pi}_{\text{elev}}$	✗	✗	✗	128	1000
π_{elev}	✓	✓	✓	1024	1000

Table 2: **Training settings**. Baseline policies π_{drift} and π_{elev} (liberally) reflect capabilities currently available to low-cost, open-source wheeled platforms.

On the other hand, π_{drift} is able to complete full laps. In addition, we observe evidence of high robustness in π_{drift} not yet seen in previous literature [12, 31]. That is, when the platform does spin out, instead of attempting to regain control by cutting the throttle, the policy maintains its angular momentum for a full spin (or more) before returning to the track. A visualization of the results (Figure 4) gives insight into the policy's precise control. The maximum (controlled) slip angle is $58°$, and the average speed is about 1.6 m/s. Appendix B.1 provides further details about the design and implementation of π_{drift}.

4.2 Elevation



(a) Captured trajectories of elevation policy in real experiments.

(b) 3D view of global elevation map.

Figure 6: Comparison of policy π_{elev} and baseline $\overline{\pi}_{\text{elev}}$ in simulation and real-world elevation tasks. Fig. 6a shows real-world execution: π_{elev}(**top**) successfully avoids obstacles using local elevation, while $\overline{\pi}_{\text{elev}}$ (**bottom**) avoids elevation features entirely. Fig. 6b visualizes the elevation map along with additional experiment result annotations. Trajectories from π_{elev} (yellow/blue) and $\overline{\pi}_{\text{elev}}$ (purple/white) are overlaid. Policies only access their local 2.5×2.5 m maps during evaluation.

Background. On unstructured and uneven terrain, an elevation map is essential for identifying traversable areas in the scene. However, traversability is strongly dependent on agent morphology and dynamic state. Along with the ground geometry, the following affect how to safely traverse uneven terrain: the platform's ground clearance, center-of-mass height, wheel size, maximum torque, suspension, and momentum. Poor traversal can result in catastrophic failures, such as rollovers, and motor (or engine) stalling.

Task. To show that the trained elevation-based policy is capable of geometric reasoning, we construct a scene with two types of elevation features that contribute to the overall elevation map, i.e., walls and ramps. Crucially, the incorporation of ramps sets the task apart from sole obstacle avoidance since ramps are features with height but are still traversable with the correct approach strategy. The starting and goal positions are placed on opposite sides of the testing area, with the elevation features placed in between. The platform must safely traverse the scene to arrive at the goal position by avoiding obstacles and ascending ramps when locally feasible.

Setup. States provided to the policy are goal-relative position, orientation, and a local (2.5×2.5 meters) body-centric elevation map. A local map is sampled from a prior global elevation map. Although the platform is capable of creating onboard maps [1], a configurable indoor environment requires higher mapping resolution than is available for this evaluation.

Results. We find that the baseline primarily deviates from any elevation features to approach the goal. When the baseline does ascend the ramp, it quickly falls off the sides. However, π_{elev} can both traverse the ramp safely and navigate through subsequent obstacles to reach the goal. Like $\overline{\pi}_{\text{drift}}$, the lower number of simulated agents gives the baseline policy significantly fewer opportunities to explore correct trajectories in a narrow range of safe states.

4.3 Visual

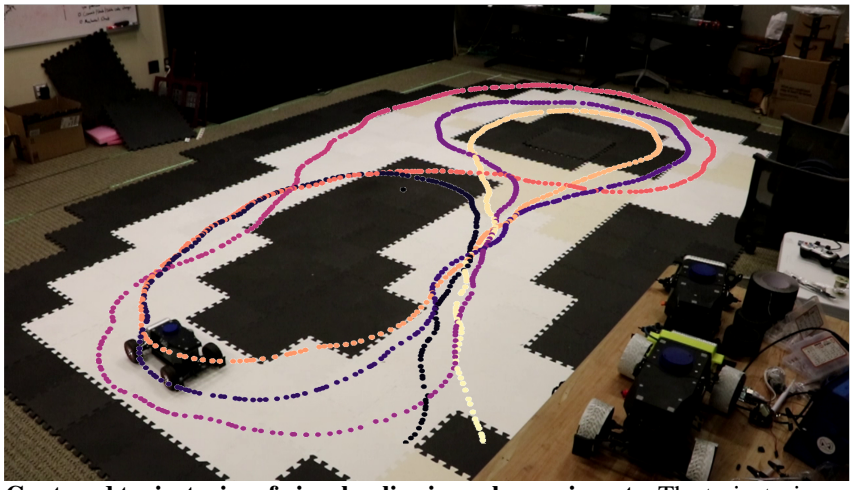


Figure 7: **Captured trajectories of visual policy in real experiments**. The trajectories are obtained from the motion capture system. We include videos of each setting and trial in the supplementary.

Background. Visual navigation traditionally relies heavily on a semantic understanding of the environment through visual feedback [51]. Despite its importance, embedding information such as traversability through this modality has its unique challenges [52]. First, visual observations (e.g., RGB images) from robots are represented in high-dimensional tensors, requiring exponentially more training data. Also, small changes in environments that result from illumination, weather, or time of day can result in catastrophic failure of machine learning models [53].

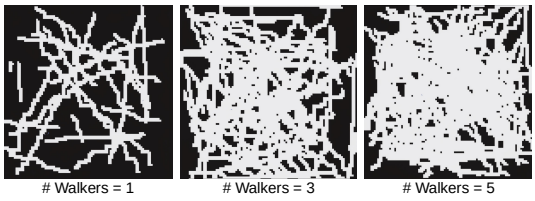


Figure 8: **Examples of randomly generated environments to train π_{vis}.** Black and white represent penalizing and safe regions, respectively. We adjust the number of walkers to train the policy across different difficulties. Using a smaller number of walkers will make the training environments more challenging.

These challenges can be effectively addressed through the simulation of visual information. In addition to generating large amounts of task-relevant data, domain randomization can help improve generalization during transfer.

Task. We construct our task scene using black and white foam tiles (see Fig. 7). These tiles are inexpensive, safe, and widely available. Surrounded by black tiles, a "figure-8" path is laid using white tiles. The platform must remain on the white path while avoiding the surrounding black tiles. A dynamic version of this task is also used in evaluation; black "barricade" tiles are dynamically removed and placed to demonstrate robustness and real-time navigation reasoning.

We generate simulation environments with black and white areas, as we show in Fig. 8 to train the visual policy. Each example in the figure represents a sub-environment, and our entire map is com-

posed of sub-environments. This allows us to assign different difficulties for each sub-environment. See Appendix B.3 for additional details about map generation.

Evaluation. We measure success or failure over five trials. We define success as the robot finishing at least one lap without staying in one place for more than 5 seconds.

Setup. Our input states include grayscale observation (resolution of 40×60), linear and angular velocity, and last action. Linear and angular velocities are obtained from the motion capture system. We design experiments to demonstrate the effectiveness of image augmentations (i.e., color jittering and Gaussian blur) and different architectures (i.e., MLP vs CNN) in Sim2Real generalization.

	MLP		CNN	
	No Aug.	With Aug.	No Aug.	With Aug.
# of Success / Trial	0 / 5	3 / 5	0 / 5	1 / 5

Table 3: **Visual policy results.** Results demonstrate the effectiveness of image augmentations for Sim2Real generalization. While we expect CNN to be more capable of learning, MLP presents better generalization capability. We include videos of each run in the supplementary materials.

Results. For quantitative evaluation, we run experiments five times and count the number of successes for each setting. As reported in Table 3, image augmentation is essential in Sim2Real generalization. The policies without augmentation fail at generalization. Further, despite the better learning capability of CNNs, they result in worse generalization to the real world; we also found that CNN training often collapses to a trivial solution, with the reward designs remaining the same as MLP training. Such results imply that CNNs may overfit to simulated data and require more careful design in rewards and loss functions for stable training.

In the supplementary materials, we qualitatively evaluate the results using recorded videos. Although both policies with augmentation predict meaningful actions, the MLP-based policy generates smoother and precise actions. In contrast, the CNN-based policy often generates noisy actions, as shown in fast steering changes and audible motor cogging.

5 Conclusion

Drawing from our results in Section 4, observe that there is an alarming gap in techniques between the scientific and broader communities. Methods and modalities (e.g., parallelization, corruption, elevation), almost native in popular implementations and frameworks for platforms like humanoids and quadrupeds, are scattered throughout different implementations and research efforts for wheeled platforms.

We present this work, Wheeled Lab, to act as a bridge between the broader community and state-of-the-art research in robot learning through low-cost, open-source wheeled platforms. Atop Isaac Lab, Wheeled Lab provides a modular and self-contained design framework to lower the barrier-to-entry while being fully integrated with Isaac Lab's extensive research community and ecosystem.

To kickstart this integration, we demonstrate the entire Sim2Real pipeline on three state-of-the-art policy types. Each policy and task is motivated by their fundamental relevance to general robot learning and Sim2Real. *Drifting* is a complex control problem that highlights accurate, randomized, and massive physical simulation. Through Wheeled Lab, we demonstrate the first zero-shot transfer of a drifting policy thus far in the literature. *Elevation traversal* defines safe behavior at the interface between perception and control, highlighting physical scene generation and massive interaction. *Visual navigation* integrates a sensor modality with substantial research attention and pre-training potential while showcasing challenges in data diversity and transferability.

Wheeled Lab is entirely low-cost and open-source. Please refer to the project website (`https://uwrobotlearning.github.io/WheeledLab/`) for experiment videos and more information.

6 Limitations

Although an objective of Wheeled Lab is to support education, this work has not yet conducted a study to evaluate how comprehensible the pipeline is to robotics students. Like F1Tenth, we aim to construct an open curriculum around Wheeled Lab to help introduce difficult or interdisciplinary topics that may be sources of confusion. This study will help reveal what steps in the Sim2Real process might prove most challenging or unfamiliar.

While iterative testing for unique conditions can serve pedagogical purposes, the appeal of building, training, and deploying a platform that "just works" can be inspirational for beginner roboticists. However, the policies presented here in their current open-source state must be met with patience and an intent to iterate. Thus, custom low-cost wheeled platforms invite yet another frontier of modern robot learning: robust adaptation [54]. For instance, the development of a drifting pipeline that adequately adapts to friction on deployment can help lower the barrier-to-entry even further.

Towards Accessible Sim2Real

The development of Wheeled Lab is constrained by both platform *and development* to low-cost yet easily obtainable resources. Mechanical hysteresis and tolerances of hobby-grade components, limited onboard compute and sensing, visual diversity, methodological simplicity, access to evaluation terrain, access to open-source packages, and access to training resources are all considered in the development of Wheeled Lab. Sim2Real accessibility is not only a challenging constraint on robot hardware but on the entire development process.

In accessibility efforts, experts are perhaps paradoxically limited by their expertise, making viewing problems from the perspective of a novice or non-traditional practitioner challenging or impossible. We find that broader audiences tend to benefit significantly from approachable infrastructure, familiar abstractions, modularity, and straightforward, reproducible, yet instructive tasks and experiments. Wheeled Lab has fortunately garnered interest from the broader community in support of these objectives, receiving contributions and feedback from the perspective of non-scientific audiences. Meanwhile, we expect that experts are better positioned to contribute fundamental concepts with vision. Wheeled Lab is an essential step towards aligning these communities and fostering a future of robot learning where all roboticists can equip themselves with indispensable skills in hardware, deployment, and iterative development.

References

[1] S. Talia, M. Schmittle, A. Lambert, A. Spitzer, C. Mavrogiannis, and S. S. Srinivasa. Demonstrating HOUND: A Low-cost Research Platform for High-speed Off-road Underactuated Nonholonomic Driving, July 2024. URL http://arxiv.org/abs/2311.11199. arXiv:2311.11199 [cs].

[2] S. S. Srinivasa, P. Lancaster, J. Michalove, M. Schmittle, C. Summers, M. Rockett, R. Scalise, J. R. Smith, S. Choudhury, C. Mavrogiannis, and F. Sadeghi. MuSHR: A Low-Cost, Open-Source Robotic Racecar for Education and Research, Dec. 2023. URL http://arxiv.org/abs/1908.08031. arXiv:1908.08031 [cs].

[3] M. O'Kelly, H. Zheng, D. Karthik, and R. Mangharam. F1TENTH: An Open-source Evaluation Environment for Continuous Control and Reinforcement Learning. In *Proceedings of the NeurIPS 2019 Competition and Demonstration Track*, pages 77–89. PMLR, Aug. 2020. URL https://proceedings.mlr.press/v123/o-kelly20a.html. ISSN: 2640-3498.

[4] M. Mittal, C. Yu, Q. Yu, J. Liu, N. Rudin, D. Hoeller, J. L. Yuan, R. Singh, Y. Guo, H. Mazhar, A. Mandlekar, B. Babich, G. State, M. Hutter, and A. Garg. Orbit: A Unified Simulation Framework for Interactive Robot Learning Environments. *IEEE Robotics and Automation Letters*, 8(6):3740–3747, June 2023. ISSN 2377-3766, 2377-3774. doi:10.1109/LRA.2023.3270034. URL http://arxiv.org/abs/2301.04195. arXiv:2301.04195 [cs].

[5] T. Miki, J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter. Learning robust perceptive locomotion for quadrupedal robots in the wild. *Science Robotics*, 7(62):eabk2822, Jan. 2022. doi:10.1126/scirobotics.abk2822. URL https://www.science.org/doi/abs/10.1126/scirobotics.abk2822. Publisher: American Association for the Advancement of Science.

[6] D. Hoeller, N. Rudin, D. Sako, and M. Hutter. ANYmal parkour: Learning agile navigation for quadrupedal robots. *Science Robotics*, 9(88):eadi7566, Mar. 2024. doi:10.1126/scirobotics.adi7566. URL https://www.science.org/doi/full/10.1126/scirobotics.adi7566. Publisher: American Association for the Advancement of Science.

[7] E. Kaufmann, L. Bauersfeld, A. Loquercio, M. Müller, V. Koltun, and D. Scaramuzza. Champion-level drone racing using deep reinforcement learning. *Nature*, 620(7976):982–987, Aug. 2023. ISSN 1476-4687. doi:10.1038/s41586-023-06419-4. URL https://www.nature.com/articles/s41586-023-06419-4. Publisher: Nature Publishing Group.

[8] OpenAI, I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas, J. Schneider, N. Tezak, J. Tworek, P. Welinder, L. Weng, Q. Yuan, W. Zaremba, and L. Zhang. Solving Rubik's Cube with a Robot Hand, Oct. 2019. URL http://arxiv.org/abs/1910.07113. arXiv:1910.07113 [cs].

[9] K. Huang, R. Rana, A. Spitzer, G. Shi, and B. Boots. DATT: Deep Adaptive Trajectory Tracking for Quadrotor Control, Dec. 2023. URL http://arxiv.org/abs/2310.09053. arXiv:2310.09053 [cs].

[10] Q. Liao, B. Zhang, X. Huang, X. Huang, Z. Li, and K. Sreenath. Berkeley Humanoid: A Research Platform for Learning-based Control, July 2024. URL http://arxiv.org/abs/2407.21781. arXiv:2407.21781 [cs].

[11] T. V. Samak, C. V. Samak, S. Kandhasamy, V. Krovi, and M. Xie. AutoDRIVE: A Comprehensive, Flexible and Integrated Digital Twin Ecosystem for Enhancing Autonomous Driving Research and Education. *Robotics*, 12(3):77, May 2023. ISSN 2218-6581. doi:10.3390/

robotics12030077. URL http://arxiv.org/abs/2212.05241. arXiv:2212.05241 [cs].

[12] G. Williams, N. Wagener, B. Goldfain, P. Drews, J. M. Rehg, B. Boots, and E. A. Theodorou. Information theoretic MPC for model-based reinforcement learning. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1714–1721, May 2017. doi:10.1109/ICRA.2017.7989202. URL https://ieeexplore.ieee.org/document/7989202/?arnumber=7989202.

[13] T. Han, A. Liu, A. Li, A. Spitzer, G. Shi, and B. Boots. Model Predictive Control for Aggressive Driving Over Uneven Terrain, June 2024. URL http://arxiv.org/abs/2311.12284. arXiv:2311.12284 [cs].

[14] T. Han, S. Talia, R. Panicker, P. Shah, N. Jawale, and B. Boots. Dynamics Models in the Aggressive Off-Road Driving Regime, May 2024. URL http://arxiv.org/abs/2405.16487. arXiv:2405.16487 [cs].

[15] A. Datar, C. Pan, M. Nazeri, and X. Xiao. Toward Wheeled Mobility on Vertically Challenging Terrain: Platforms, Datasets, and Algorithms. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 16322–16329, May 2024. doi:10.1109/ICRA57147.2024.10610079. URL https://ieeexplore.ieee.org/document/10610079.

[16] E. Horváth, G. Ignéczi, N. Markó, R. Krecht, and M. Unger. Teaching Aspects of ROS 2 and Autonomous Vehicles. *Engineering Proceedings*, 79(1):49, 2024. ISSN 2673-4591. doi:10.3390/engproc2024079049. URL https://www.mdpi.com/2673-4591/79/1/49. Number: 1 Publisher: Multidisciplinary Digital Publishing Institute.

[17] B. D. Evans, R. Trumpp, M. Caccamo, F. Jahncke, J. Betz, H. W. Jordaan, and H. A. Engelbrecht. Unifying F1TENTH Autonomous Racing: Survey, Methods and Benchmarks, Apr. 2024. URL http://arxiv.org/abs/2402.18558. arXiv:2402.18558 [cs].

[18] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun. CARLA: An Open Urban Driving Simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16. PMLR, Oct. 2017. URL https://proceedings.mlr.press/v78/dosovitskiy17a.html. ISSN: 2640-3498.

[19] A. Brunnbauer, L. Berducci, A. Brandstätter, M. Lechner, R. Hasani, D. Rus, and R. Grosu. Latent Imagination Facilitates Zero-Shot Transfer in Autonomous Racing, Feb. 2022. URL http://arxiv.org/abs/2103.04909. arXiv:2103.04909 [cs].

[20] N. Hamilton, P. Musau, D. M. Lopez, and T. T. Johnson. Zero-Shot Policy Transfer in Autonomous Racing: Reinforcement Learning vs Imitation Learning. In *2022 IEEE International Conference on Assured Autonomy (ICAA)*, pages 11–20, Mar. 2022. doi:10.1109/ICAA52185.2022.00011. URL https://ieeexplore.ieee.org/document/9763640/?arnumber=9763640.

[21] X. Cai, J. Queeney, T. Xu, A. Datar, C. Pan, M. Miller, A. Flather, P. R. Osteen, N. Roy, X. Xiao, and J. P. How. PIETRA: Physics-Informed Evidential Learning for Traversing Out-of-Distribution Terrain. *IEEE Robotics and Automation Letters*, pages 1–8, 2025. ISSN 2377-3766. doi:10.1109/LRA.2025.3527285. URL https://ieeexplore.ieee.org/document/10833878/?arnumber=10833878. Conference Name: IEEE Robotics and Automation Letters.

[22] M. Deitke, W. Han, A. Herrasti, A. Kembhavi, E. Kolve, R. Mottaghi, J. Salvador, D. Schwenk, E. VanderBilt, M. Wallingford, L. Weihs, M. Yatskar, and A. Farhadi. RoboTHOR: An Open Simulation-to-Real Embodied AI Platform. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3161–3171, Seattle, WA, USA, June 2020. IEEE. ISBN 978-1-72817-168-5. doi:10.1109/CVPR42600.2020.00323. URL https://ieeexplore.ieee.org/document/9157346/.

[23] S. Salimpour, J. Peña-Queralta, D. Paez-Granados, J. Heikkonen, and T. Westerlund. Sim-to-Real Transfer for Mobile Robots with Reinforcement Learning: from NVIDIA Isaac Sim to Gazebo and Real ROS 2 Robots, Jan. 2025. URL http://arxiv.org/abs/2501.02902. arXiv:2501.02902 [cs].

[24] B. Balaji, S. Mallya, S. Genc, S. Gupta, L. Dirac, V. Khare, G. Roy, T. Sun, Y. Tao, B. Townsend, E. Calleja, S. Muralidhara, and D. Karuppasamy. DeepRacer: Autonomous Racing Platform for Experimentation with Sim2Real Reinforcement Learning. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2746–2754, May 2020. doi:10.1109/ICRA40945.2020.9197465. URL https://ieeexplore.ieee.org/abstract/document/9197465?casa_token=P1_UodV4wbEAAAAA:6EV0TBoDKvtpQailXF-0QMLLebuYMajCFlRjbSzrKQB__Q2khTqumEjWZKAM4C3Xrg7WgicvRg. ISSN: 2577-087X.

[25] C. Samak, T. Samak, and V. Krovi. Towards Sim2Real Transfer of Autonomy Algorithms using AutoDRIVE Ecosystem. *IFAC-PapersOnLine*, 56(3):277–282, Jan. 2023. ISSN 2405-8963. doi:10.1016/j.ifacol.2023.12.037. URL https://www.sciencedirect.com/science/article/pii/S2405896323023704.

[26] W. Xiao, H. Xue, T. Tao, D. Kalaria, J. M. Dolan, and G. Shi. AnyCar to Anywhere: Learning Universal Dynamics Model for Agile and Adaptive Mobility, Sept. 2024. URL http://arxiv.org/abs/2409.15783. arXiv:2409.15783 [cs].

[27] M. Cutler and J. P. How. Autonomous drifting using simulation-aided reinforcement learning. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5442–5448, Stockholm, Sweden, May 2016. IEEE. ISBN 978-1-4673-8026-3. doi:10.1109/ICRA.2016.7487756. URL http://ieeexplore.ieee.org/document/7487756/.

[28] B. D. Evans, H. W. Jordaan, and H. A. Engelbrecht. Comparing deep reinforcement learning architectures for autonomous racing. *Machine Learning with Applications*, 14:100496, Dec. 2023. ISSN 26668270. doi:10.1016/j.mlwa.2023.100496. URL https://linkinghub.elsevier.com/retrieve/pii/S266682702300049X.

[29] J. Gonzales, F. Zhang, K. Li, and F. Borrelli. Autonomous drifting with onboard sensors. In *Advanced Vehicle Control*. CRC Press, 2016. ISBN 978-1-315-26528-5. Num Pages: 6.

[30] P. Cai, X. Mei, L. Tai, Y. Sun, and M. Liu. High-Speed Autonomous Drifting With Deep Reinforcement Learning. *IEEE Robotics and Automation Letters*, 5(2):1247–1254, Apr. 2020. ISSN 2377-3766. doi:10.1109/LRA.2020.2967299. URL https://ieeexplore.ieee.org/document/8961997/?arnumber=8961997. Conference Name: IEEE Robotics and Automation Letters.

[31] F. Djeumou, M. Thompson, M. Suminaka, and J. Subosits. Reference-Free Formula Drift with Reinforcement Learning: From Driving Data to Tire Energy-Inspired, Real-World Policies, Oct. 2024. URL http://arxiv.org/abs/2410.20990. arXiv:2410.20990 [cs].

[32] G. Chen, X. Zhao, Z. Gao, and M. Hua. Dynamic Drifting Control for General Path Tracking of Autonomous Vehicles. *IEEE Transactions on Intelligent Vehicles*, 8(3):2527–2537, Mar. 2023. ISSN 2379-8904. doi:10.1109/TIV.2023.3235007. URL https://ieeexplore.ieee.org/document/10011537/?arnumber=10011537. Conference Name: IEEE Transactions on Intelligent Vehicles.

[33] J. Frey, M. Patel, D. Atha, J. Nubert, D. Fan, A. Agha, C. Padgett, P. Spieler, M. Hutter, and S. Khattak. RoadRunner – Learning Traversability Estimation for Autonomous Off-road Driving, Aug. 2024. URL http://arxiv.org/abs/2402.19341. arXiv:2402.19341 [cs].

[34] J. Gibson, B. Vlahov, D. Fan, P. Spieler, D. Pastor, A.-a. Agha-mohammadi, and E. A. Theodorou. A Multi-step Dynamics Modeling Framework For Autonomous Driving In Multiple Environments, May 2023. URL http://arxiv.org/abs/2305.02241. arXiv:2305.02241 [cs].

[35] K. Stachowicz, D. Shah, A. Bhorkar, I. Kostrikov, and S. Levine. FastRLAP: A System for Learning High-Speed Driving via Deep RL and Autonomous Practicing. In *Proceedings of The 7th Conference on Robot Learning*, pages 3100–3111. PMLR, Dec. 2023. URL https://proceedings.mlr.press/v229/stachowicz23a.html. ISSN: 2640-3498.

[36] T. Xu, C. Pan, and X. Xiao. Reinforcement Learning for Wheeled Mobility on Vertically Challenging Terrain. In *2024 IEEE International Symposium on Safety Security Rescue Robotics (SSRR)*, pages 125–130, Nov. 2024. doi:10.1109/SSRR62954.2024.10770034. URL https://ieeexplore.ieee.org/document/10770034/?arnumber=10770034. ISSN: 2475-8426.

[37] K. Kang, S. Belkhale, G. Kahn, P. Abbeel, and S. Levine. Generalization through Simulation: Integrating Simulated and Real Data into Deep Reinforcement Learning for Vision-Based Autonomous Flight. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 6008–6014, May 2019. doi:10.1109/ICRA.2019.8793735. URL https://ieeexplore.ieee.org/document/8793735/?arnumber=8793735. ISSN: 2577-087X.

[38] Z. Yuan, T. Wei, S. Cheng, G. Zhang, Y. Chen, and H. Xu. Learning to Manipulate Anywhere: A Visual Generalizable Framework For Reinforcement Learning, Oct. 2024. URL http://arxiv.org/abs/2407.15815. arXiv:2407.15815 [cs].

[39] X. Technologies. 1X Technologies | Safe, Intelligent Humanoids. URL https://www.1x.tech/.

[40] Crossing the Sim2Real Gap With NVIDIA Isaac Lab, . URL https://agilityrobotics.com/content/crossing-sim2real-gap-with-isaaclab.

[41] Get Started with Reinforcement Learning for Spot, . URL https://support.bostondynamics.com/s/article/Get-Started-with-Reinforcement-Learning-for-Spot-49966.

[42] NVIDIA Isaac Lab Blog, . URL https://fieldai.com/news/field-ai-nvidia-partnership.

[43] MenteeBot, . URL https://menteebot.com/blog/#shopping-companion-2024.

[44] Y. J. Ma, W. Liang, G. Wang, D.-A. Huang, O. Bastani, D. Jayaraman, Y. Zhu, L. Fan, and A. Anandkumar. Eureka: Human-Level Reward Design via Coding Large Language Models, Apr. 2024. URL http://arxiv.org/abs/2310.12931. arXiv:2310.12931 [cs].

[45] S. Tao, F. Xiang, A. Shukla, Y. Qin, X. Hinrichsen, X. Yuan, C. Bao, X. Lin, Y. Liu, T.-k. Chan, Y. Gao, X. Li, T. Mu, N. Xiao, A. Gurha, Z. Huang, R. Calandra, R. Chen, S. Luo, and H. Su. ManiSkill3: GPU Parallelized Robotics Simulation and Rendering for Generalizable Embodied AI, Oct. 2024. URL http://arxiv.org/abs/2410.00425. arXiv:2410.00425 [cs].

[46] Y. Yang, G. Shi, C. Lin, X. Meng, R. Scalise, M. G. Castro, W. Yu, T. Zhang, D. Zhao, J. Tan, and B. Boots. Agile Continuous Jumping in Discontinuous Terrains, Sept. 2024. URL http://arxiv.org/abs/2409.10923. arXiv:2409.10923 [cs].

[47] T. Han, Y. Bao, B. Mehta, G. Guo, A. Vishwakarma, E. Kang, S. Jung, R. Scalise, J. Zhou, B. Xu, and B. Boots. Model Predictive Adversarial Imitation Learning for Planning from Observation, July 2025. URL http://arxiv.org/abs/2507.21533. arXiv:2507.21533 [cs].

[48] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal Policy Optimization Algorithms, Aug. 2017. URL http://arxiv.org/abs/1707.06347. arXiv:1707.06347.

[49] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter. Learning quadrupedal locomotion over challenging terrain. *Science Robotics*, 5(47):eabc5986, Oct. 2020. doi:10.1126/scirobotics.abc5986. URL https://www.science.org/doi/10.1126/scirobotics.abc5986. Publisher: American Association for the Advancement of Science.

[50] M. Acosta, S. Kanarachos, and M. E. Fitzpatrick. A Hybrid Hierarchical Rally Driver Model for Autonomous Vehicle Agile Maneuvering on Loose Surfaces:. In *Proceedings of the 14th International Conference on Informatics in Control, Automation and Robotics*, pages 216–225, Madrid, Spain, 2017. SCITEPRESS - Science and Technology Publications. ISBN 978-989-758-263-9 978-989-758-264-6. doi:10.5220/0006393002160225. URL http://www.scitepress.org/DigitalLibrary/Link.aspx?doi=10.5220/0006393002160225.

[51] S. Jung, J. Lee, X. Meng, B. Boots, and A. Lambert. V-STRONG: Visual Self-Supervised Traversability Learning for Off-road Navigation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1766–1773, May 2024. doi:10.1109/ICRA57147.2024.10611227. URL https://ieeexplore.ieee.org/abstract/document/10611227.

[52] Y. D. V. Yasuda, L. E. G. Martins, and F. A. M. Cappabianco. Autonomous Visual Navigation for Mobile Robots: A Systematic Literature Review. *ACM Computing Surveys*, 53(1):1–34, Jan. 2021. ISSN 0360-0300, 1557-7341. doi:10.1145/3368961. URL https://dl.acm.org/doi/10.1145/3368961.

[53] S. Choi, S. Jung, H. Yun, J. T. Kim, S. Kim, and J. Choo. RobustNet: Improving Domain Generalization in Urban-Scene Segmentation via Instance Selective Whitening. pages 11580–11590, 2021. URL https://openaccess.thecvf.com/content/CVPR2021/html/Choi_RobustNet_Improving_Domain_Generalization_in_Urban-Scene_Segmentation_via_Instance_Selective_CVPR_2021_paper.html.

[54] K. Bousmalis, A. Irpan, P. Wohlhart, Y. Bai, M. Kelcey, M. Kalakrishnan, L. Downs, J. Ibarz, P. Pastor, K. Konolige, S. Levine, and V. Vanhoucke. Using Simulation and Domain Adaptation to Improve Efficiency of Deep Robotic Grasping. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4243–4250, May 2018. doi:10.1109/ICRA.2018.8460875. URL https://ieeexplore.ieee.org/document/8460875/?arnumber=8460875. ISSN: 2577-087X.

A Supplementary Material

Videos and more information can be found on the project website:

https://uwrobotlearning.github.io/WheeledLab/

B Design and Implementation Details

The typical ranges on friction, mechanical tolerances, damages, etc. for low-cost platforms are far from ideal. However, iterating and developing on individual platforms in unique environments is practically and pedagogically critical, especially for real-world robotics. We hope that this work allows practitioners to focus their efforts on precisely this development cycle rather than infrastructure. The following implementation details should be regarded as a starting point and potential reference for troubleshooting inevitable issues.

Isaac Lab environment configurations primarily include parameter settings and reward functions, and can be quickly parsed while efficiently communicating environment behavior. We highlight some notable details in this section.

B.1 Drifting

We found that drifting was an exemplary task to demonstrate the physical Sim2Real gap. Much of the development cycle was focused on closing this gap, as supported by countless other seminal works [8, 6].

B.1.1 Actuators

In many related works for agile tasks, data is typically collected to assess actuator response and estimate gains. However, this process can be unfamiliar to beginning roboticists or non-traditional enthusiasts. *Instead, all actuator parameters are extracted exactly from their technical specification sheet online, and gains are domain-randomized across environments.* Achieving the task in this manner helps to convey that, while domain expertise can certainly improve performance (domain randomized policies are known to find conservative solutions [10]), it is not strictly necessary through modern methods.

It is worth noting that a few "Sim2Real2Sim" (or Sim2Sim) cycles were spent narrowing the randomized gain ranges for the throttle actuator settings. Due to neural network policy tendencies to exhibit "bang-bang" (on-off) behavior, gain values can become extremely important to the actual forces exerted on the vehicle.

Establishing and improving pipelines for this process can be an accessible yet effective future contribution for use by potential practitioners.

B.1.2 Friction

Initially, the domain randomization over friction was used to cover a wide range (0.2 to 0.8) of potential friction values between the tape-covered tires and the carpeted ground. However, resulting policies would perform poorly and inconsistently on deployment. A long thread of Sim2Sim cycles was used to search for friction parameters but still resulting policies were not reliably transferring.

In the end, a cheap (less than 5 USD) spring scale was used to estimate the coefficient of friction, measured by dragging the vehicle with the scale along the carpet. This value (about 0.4) was then used as the midpoint of the friction randomization during training.

The project website also contains videos of deployments on different surfaces (e.g. finished concrete). Lower friction values appear to cause the vehicle to take wider turns (due to lack of turning friction) and spin out quickly during drift maneuvers. Enabling the platform to drift over multi-

ple surfaces is also of fundamental interest to the robotics community as it is a form of domain adaptation.

B.1.3 Articulations

While we initially used the MuSHR [2] articulations defined by the open-source URDF file, we found that its modeling inaccuracies affected the precise control expectations of the platform. As a result, we improved the articulations: (1) by moving the steering linkages from the wheel axis to the actual steering joint and (2) by articulating the suspension joints of the vehicle.

Steering inaccuracies were noticeable due to the gap between turning radii in real and sim. The moment arm on the linkage can also adversely affect the tracking of the steering joint during high-velocity maneuvers. This can even be observed in simulation when steering joints behave strangely asymmetrically with unsuitable actuator settings.

Suspension helps close the gap on load shifting during turns. With a suspension travel of 4 cm, these shifts have consequences in the total friction and "thrust" produced while throttling through turns. Interestingly, suspension articulation also helped to stabilize the simulation. At a wheel speed of 3 m/s, the rotation rate of the wheels reaches 60 rad/s, occasionally resulting in collision penetrations during simulation which cause unrealistic "jumping" through drifted turns. Damped suspension joints helped to stabilize these collisions without requiring smaller simulation time steps which would significantly lengthen training times.

B.1.4 Rewards

Six rewards were designed for this task:

i.) **Cross-Track Distance**. The lateral distance of the vehicle from an oval track line. This is defined without time-dependent reference trajectories as often done in model-based approaches or even in deep RL methods [30]. This is highly weighted otherwise, the agent tends to take larger, safer turns.

ii.) **Velocity**. Reward for high speeds.

iii.) **Side-Slip**. Only stable side-slip angles were rewarded. Stable side-slip angles are manageable through the steering limits (0-30°).

iv.) **Progress**. Reward for making progress along an arbitrary direction (counter-clockwise, in this work).

v.) **Turn-Energy**. Shaping term for increasing velocity rewards, specifically when inside turn regions.

vi.) **Turn-Left-Go-Right**. Shaping term for encouraging exploration to discover an alternate turning mode (e.g., drifting) by counter-steering. Strictly positive when the angular velocity and the steering command are opposite to each other. 0 otherwise.

Penalties were also given for going out-of-bounds.

B.2 Elevation

B.2.1 Training

Training elevation in simulation proved to be a challenging task. Training would often collapse into one of: (1) seeking goals while avoiding obstacles or (2) climbing ramps. Striking a balance between the two often required re-designing the scene or task configuration in simulation.

As observed for model-based methods [14], evaluating uneven terrain traversal for vehicles can be challenging due to the reasonable alternate mode of avoiding risky terrain altogether, which may often be faster. However, the primary goal of this work is to better enable the broader community to

engage with state-of-the-art methods in elevation-based robotics. Thus, the reward and scene were constructed to incentivize the agent to actively seek out risky terrain while penalizing failures.

Supported by machine learning literature, noise injection becomes less effective for generalization at higher dimensions. For states such as elevation maps, perceived noise is likely more structured than the additive Gaussians injected through observation corruption. While debugging, we found that the policy would improve if the elevation features were brought closer in shape to the features seen in the simulation. To resolve this without "training on test", the scene was augmented with slopes of various gradations and heights.

B.2.2 Suspension

As mentioned in Appendix B.1, suspension joints were added to the open-source MuSHR articulations. This is critical for elevation where maintaining momentum up a ramp helps training stability. Without suspension articulation, climbing ramps have an abrupt and destabilizing impact at higher speeds.

B.2.3 Rewards

i.) **Goal Velocity**. Rewards velocity projected onto the goal heading vector.

ii.) **Z-Position**. Rewards larger z-positions (gained through ascending ramps).

iii.) **Falling Penalty**. Penalizes negative body-frame z velocities.

iv.) **Roll on elevation**. Penalizes roll angles while on elevation features.

B.3 Visual

While visual information is essential in modern robotics to solve complex real-world problems, simulating such information requires state-of-the-art rendering techniques to close the Sim2Real gap. Despite such computationally heavy efforts, machine learning models trained with simulated data often fail at real-world generalization due to diverse types of environmental changes, e.g., illuminations, weather, etc. Instead, in this work, we "simplify" visual information from both simulation and real by compressing RGB pixels to grayscale, closing the Sim2Real gap. Still, we observed that such compressed visual information exhibits a non-trivial amount of Sim2Real gap. This section describes how we tackle this problem by providing details of our environment, training augmentations, architectures, and reward designs.

B.3.1 Visual map generation

We designed the problem by encoding traversability into color information. Black tiles are non-traversable regions while white indicates safe to travel. We divided the whole map into multiple sub-environments that are individually configurable. Then, we randomized white paths over the black area in each sub-environment. Specifically, each sub-environment comprises 9 cells, and we sample a start point in each cell and generate paths toward random endpoints in the sub-environment. We made the number of random paths from a single start point configurable, allowing the users to adjust the difficulty of the training (see Fig. 8). This can later be used for curriculum learning by assigning different difficulties for sub-environments. We tested the generated map with more than 2,000 agents in parallel, running seamlessly with a single Nvidia RTX 3080 GPU. The algorithm overview for sub-environment generation is presented in Alg. 1.

B.3.2 Model Architectures

We adopt a multi-layer perceptron (MLP) as our basic policy network. For ablations, we add a simple 3-layer convolutional neural network (CNN) that serves as an image feature extractor before the policy networks. The extracted features are concatenated to the rest of the observation terms, such as linear/angular velocities and last action, and fed into the MLP policy networks. We observe

Algorithm 1 Visual Policy Sub-environment Generation

Input:
$(H_{\text{env}}, W_{\text{env}}) \leftarrow$ # of rows and # of columns for each sub environment
$(H_{\text{cell}}, W_{\text{cell}}) \leftarrow$ # of rows and # of columns for each cell
$N_{\text{walkers}} \leftarrow$ # of walkers for path generation

Pseudo-code:
▷ Initialize traversability map; 0 - not traversable / 1 - traversable
$\mathbf{T} \in \{0, 1\}^{H_{\text{env}} \times W_{\text{env}}} \leftarrow \mathbf{0}$

▷ Sample start points from each cell
$\mathbf{P}^{\text{start}} \leftarrow [\,]$ ▷ Initialize an array to save start points
▷ Sample random start points for each cell
for row $r = 0, 1, \ldots, H_{\text{env}}/H_{\text{cell}}$ **do**
  for col $c = 0, 1, \ldots, W_{\text{env}}/W_{\text{cell}}$ **do**
    $p_{\text{row}} \leftarrow \text{UniformSampling}(0, H_{\text{cell}}) + r \cdot H_{\text{cell}}$
    $p_{\text{col}} \leftarrow \text{UniformSampling}(0, W_{\text{cell}}) + c \cdot W_{\text{cell}}$
    $\mathbf{P}^{\text{start}} \mathrel{+}= [(p_{\text{row}}, p_{\text{col}})]$ ▷ Append the start point to $\mathbf{P}^{\text{start}}$
  end for
end for

for $(p_{\text{row}}, p_{\text{col}})$ in $\mathbf{P}^{\text{start}}$ **do**
  for iter$= 0, 1, \ldots, N_{\text{walkers}}$ **do**
    ▷ Sample random end points
    Do
      $q_{\text{row}} \leftarrow \text{UniformSampling}(0, H_{\text{env}})$
      $q_{\text{col}} \leftarrow \text{UniformSampling}(0, W_{\text{env}})$
    do While $\mathbf{T}(q_{\text{row}}, q_{\text{col}}) == 1$
    ▷ Generate random paths and update traversability map
    $\text{GenerateRandomPaths}(\mathbf{T},\ (p_{\text{row}}, p_{\text{col}}),\ (q_{\text{row}}, q_{\text{col}}))$
  end for
end for
Return: $\mathbf{T}$

that inducing CNN takes longer training time to learn optimal actions and often collapses into a trivial solution - agents decide to spin in a circle, only maximizing the velocity rewards.

B.3.3 Image Augmentations

To close the sim-to-real gap, we apply image augmentations for policy training. This includes aggressive color jittering (*i.e.,* brightness, contrast, saturation, and hue adjustments) and Gaussian blur. As demonstrated in both qualitative and quantitative evaluations, image augmentations play a crucial role in closing sim-to-real gaps. Further augmentations or randomization can be easily applied to our code design.

B.3.4 Rewards

Two rewards were designed for this task

i.) **Velocity Reward**. Rewards higher velocities

ii.) **Traversability Reward and Penalty**. Rewards being on white regions, and penalizes it for being on black regions

B.3.5 Terminal Conditions

We experimented with different terminations. In addition to the standard time-out and out-of-bounds terminal conditions, we added a terminal condition when the robot is on black. However, this often resulted in a lack of exploration, and policies that produced "jerky" or undesirable actions, such as idling at the initial location.