2024 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)
July 15-19, 2024. Boston, MA, USA

Rough Terrain Path Tracking of an Ackermann Steered Platform using Hybrid Deep Reinforcement Learning

Dhruv Mehta, Ameya Salvi, Venkat Krovi

Abstract— **Accurate path tracking of wheeled Uncrewed Ground Vehicles (UGVs) in off-road environments faces numerous challenges stemming from the diversity of operational conditions. Traditional model-based controllers for Ackermann-steered vehicles feature good (skid-free) path-tracking performance on flat ground, but performance degrades with increasingly uneven terrain and faster traversal speeds. This paper introduces a novel approach, a Hybrid Deep Reinforcement Learning (HDRL) controller, leveraging the strengths of a Linear Quadratic Regulator (LQR) and a Deep Reinforcement Learning (DRL) controller, for the enhanced path tracking of Ackermann-steered UGVs. The DRL controller primarily compensates for uncertainty in terrain conditions and unknown vehicle parameters but can be computationally expensive to train. The LQR controller guides the DRL controller during the initial training phases, ensuring a more stable performance and achieving higher rewards in early iterations. In doing so, this hybrid methodology offers promise to overcome the limitations of model-based controllers and the sample-inefficient nature of conventional DRL approaches. Preliminary results showcased in the manuscript show promise for the HDRL controller, demonstrating better performance than model-free DRL and conventional feedback controllers.**

I. INTRODUCTION

Uncrewed Ground Vehicles (UGVs) are gaining popularity for their use in unstructured off-road military, agriculture, mining, and planetary exploration applications [1]. Navigation in off-road environments introduces more complex challenges than on-road scenarios – such UGVs must traverse increasingly challenging and diverse terrain conditions like steep slopes, ditches, rocks, and vegetation. Motion-planning and subsequent tracking control in such unstructured terrains requires increasing model-fidelity tied closely to the vehicle architecture.

The selected vehicle-architecture plays a significant role in determining mobility, maneuverability and traversability on off-road terrain. Conventionally, skid-steered vehicles have been preferred for off-road terrain navigation due to their high maneuverability and ability to make zero-radius turn. However, the unpredictability of the roll/skid tire-terrain interactions requires complex system modeling and significant power consumption. In contrast, an Ackermann-steered vehicle is potentially more agile, and energy-efficient. However, formulating precise motion control at higher speeds is challenging due to complex dynamical excitation from rapidly changing terrain conditions.

In the past, model-based control approaches have been typically used with increasing levels of fidelity for more precise control. [2] demonstrates using an LQR controller for path tracking, emphasizing lateral stability. However, uncertainties in terrain conditions and the absence of wheel-terrain interaction models pose a challenge for LQR controllers to be deployed on rough terrains. Model Predictive Control (MPC) based approaches in [3] show promising results but are dependent on model fidelity and require consistent re-planning for increasing generalizability, thereby increasing its computational cost.

In recent years, model-free Deep Reinforcement Learning (DRL) has gained popularity for locomotion control. However, few papers tackle DRL applied to UGVs in off-road environments and Ackermann-steered architecture. [4] demonstrate impressive results by using an end-to-end DRL controller for the path following of an excavator on rough terrain, but requires millions of steps to train a good policy.

This paper presents a HDRL-based controller for unknown off-road terrain path tracking controller. The LQR controller eliminates the need for high-fidelity model-based controllers and reduces training time. The DRL controller is trained to compensate for uncertainty in terrain conditions and unknown vehicle parameters. Our main contributions are as follows:

1) Implementing a hybrid framework that merges the steering output from an LQR controller with a DRL controller providing correctional steering angle and vehicle velocity. This combination ensures good path tracking on simulated off-road terrain, designed for an Ackermann-steered robot.
2) Introducing a novel dense reward formulation to achieve stable and accurate path tracking, minimizing errors in the process.
3) Development of a robust policy through training on a single racetrack with diverse terrain conditions. This policy is successfully deployed on unseen racetrack paths, showcasing increased generalizability.

II. RELATED WORK

Various studies have delved into deploying path planning and path tracking of autonomous vehicles and wheeled humanoid locomotion, particularly exploring the effectiveness of hybrid and residual reinforcement learning approaches. The HDRL framework in this paper shares its fundamental concept with residual RL from [5], where the controller is decomposed into a conventional model based controller that outputs actions and the DRL controller that learns the residual actions arising due to modeling uncertainties used in the feedback.Both the control signals are then superimposed to produce the final output.

Similarly, [6] establishes a baseline control policy bootstrapped on apriori expert knowledge extracted from driver data for learning for lateral and longitudinal accelerations and then is further augmented with a DRL controller to provide corrective accelerations. [7] combines a pure pursuit controller with a DRL approach, successfully reducing lap times for an f1tenth autonomous racing vehicle. HDRL has shown promise for highly nonlinear wheeled humanoid systems as well, In [8] a combination of LQR and DRL controller is used for accurately controlling its motion with varying parameters.

We extend upon the foundational concepts introduced in [5] and [8], which utilize a residual reinforcement learning framework akin to the HDRL framework. However, to the best of the authors' knowledge, the path tracking of an Ackermann-steered platform on rough terrain using HDRL still needs to be explored. Our primary contribution is the novel integration of an LQR controller, leveraging a discrete kinematic bicycle model in conjunction with a Proximal Policy Optimization (PPO)–based Deep Reinforcement Learning (DRL) controller. This integration presents an approach to addressing the challenges associated with path tracking on uneven surfaces, thereby enhancing the maneuverability and stability of such platforms. The outputs from both the controllers are superimposed. Our proposed system generates steering angle outputs and wheel joint velocities for tracking a path in simulated off-road terrain. The DRL controller learns to adapt to external disturbances from diverse terrain conditions. The LQR controller guides the DRL controller to learn correctional steering angle in the initial training phases that an end-to-end controller may not achieve.

III. Methodology

A. Mathematical Model and Feedback Controller Design

1) Mathematical Model: In this section, we present the mathematical modeling of the commonly used linearized kinematic bicycle model shown in Fig.1 as an abstraction of an Ackermann-steered vehicle. The model illustrates the vehicle turning about and instant center with radius R with L being the wheelbase, δ the steering angle, β as is the slip angle and θ as the vehicle heading angle in the yaw plane.

Equation 3 represents the state space formulation of the error dynamics of the bicycle model with e as the crosstrack error, $\dot{e}$ the rate of change of the crosstrack error, e_θ as the error in the heading angle and $\dot{e_\theta}$ as the rate of change of error in the heading angle.

$$x = \begin{bmatrix} e & \dot{e} & e_\theta & \dot{e_\theta} \end{bmatrix} \in \mathbb{R}^4 \, , u = \begin{bmatrix} \delta \end{bmatrix} \in \mathbb{R}^1 \tag{1}$$

The discretized state-space equation for the kinematic bicycle model is given by:

$$X_{k+1} = AX_k + BU_k \tag{2}$$

Where A and B are the state transition and input matrices, respectively.

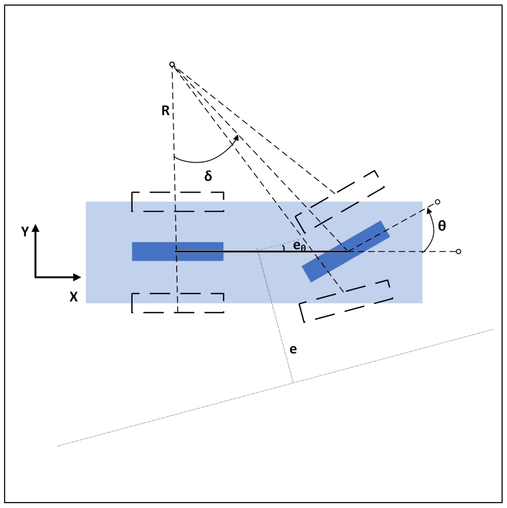


Fig. 1: Kinematic vehicle bicycle model

The linearization of matrices A and B is carried by first ordered taylor series expansion as shown in [9].

$$A = \begin{bmatrix} 1 & \Delta t & 0 & 0 \\ 0 & 0 & v & 0 \\ 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 1 \end{bmatrix} B = \begin{bmatrix} 0 \\ 0 \\ \frac{v}{L} \\ 0 \end{bmatrix} \tag{3}$$

2) LQR Controller Design: This section outlines the formulation of Linear Quadratic Regulator (LQR) control for vehicle path tracking. The LQR controller tracks the desired path by getting information of the closest waypoint and the corresponding heading angle. The goal of the LQR controller is to minimize a cost function as shown in Equation 4, that penalizes deviations from the desired path and the control effort in accordance to assigned weighing coefficients Q and R respectively.

$$J = \int_0^\infty (X^\top Q X + U^\top R U) dt \tag{4}$$

For the LQR controller, weighting matrices are $Q = \text{diag}(10, 100, 100, 1)$ for cross-track error, rate of change of cross-track error, and yaw error, respectively, and $R = [1]$ for an optimized control strategy.

B. Hybrid Reinforcement Learning Framework

The HDRL controller integrates output from the LQR controller and the DRL controller. Within the scope of this paper, the HDRL framework transmits steering angle commands to the robot. Furthermore, the DRL controller concurrently generates wheel joint velocities at each time step acting as a longitudinal controller as shown in Fig. 2.

$$\delta = \delta_{LQR} + \delta_{\Pi_\theta} \tag{5}$$

Where δ_{LQR} is the steering angle output from the LQR controller, δ_{Π_θ} is the correctional steering angle output from the DRL controller and ω_{wheel} are the rear wheel joint velocities since the platform is a rear-wheel drive system. The algorithm chosen for the DRL component of the HDRL

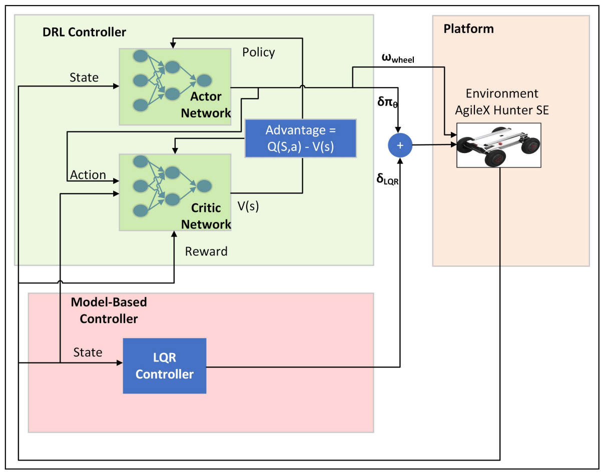


Fig. 2: Hybrid Deep Reinforcement Learning Framework

controller is Proximal Policy Optimization (PPO) that is explained in more detail as shown below.

1) PPO: The PPO algorithm is an actor-critic approach that is widely used in DRL to learn stable and high-performing policies. It employs a surrogate objective function that constrains the distance between the new and old policies during training to prevent large policy updates that can lead to instability and poor performance [10], [11].

PPO employs a clip parameter that limits the magnitude of policy updates during training. The algorithm computes a probability ratio between the new and old policies and then clips the ratio to a certain range to avoid drastic policy changes that can lead to instability. The clip parameter is a hyperparameter that can be tuned to balance stability and performance.

2) Hyperparameters: : The PPO algorithm's hyperparameters are critical to its performance, and key additional hyperparameters were carefully chosen for this study to ensure optimal performance. We choose clipping ratio ϵ as 0.2, horizon length of 128, mini-batch size of 512, 5 epochs and 32 parallelized environments. Discount factor γ is 0.99 and the Generelized Advantage Estimate (GAE) λ is 0.95. The learning rate is adaptive and the entropy coefficient is 0.001.

3) State and Action Space: The action space for our HDRL network is defined as a range of velocities for the rear wheel joint and the steering angle, denoted by [(δ_{LQR} + $\delta_{\Pi\theta}$), ω_{wheel}] $\in \mathbb{R}^{2\times1}$, where ω_{wheel} and δ represent the rear wheel joint velocities and steering angle, respectively. To enable effective learning, our network requires observations that provide relevant information about the robot's state and location. These observations guide the reward formulation strategy and guides the robot to track the desired path. The observations for PPO consist of the following parameters:

- Euclidean distance e to the closest waypoint as the cross track error and heading angle error e_{θ}
- Robot poses [x y γ] where x and y denote the robot's position, while γ is the robot's heading angle and roll angle α
- Robot speed [v] w.r.t body frame of reference.

The total observation space for the HDRL controller includes [e $\dot{e}$ e_{θ} $\dot{e_{\theta}}$ x y γ v α] $\in \mathbb{R}^{9\times1}$, where each parameter provides critical information about the robot's current location, orientation and desired location. By leveraging these observations, our network can learn to make informed decisions about the appropriate action to take at each time step.

4) Reward Formulation: In order to optimize an agent's learning progress and encourage exploration of the environment, a dense reward formulation strategy is employed, taking into account the complexity of the task. Sparse rewards, although simpler to formulate, may hinder learning progress and discourage exploration. The reward function, which guides the agent towards the target location while adhering to stability constraints, is formalized as follows:

$$R_t = R_e \times R_{\Theta} \times R_v + R_{Termination} \tag{6}$$

Where R_t is the total cumulative reward after the agent takes an action. R_e is the cross track error reward, R_{Θ} is the yaw error reward, R_v is the velocity reward and $R_{Termination}$ is the reward due to early termination. Each reward objective is normalized $\in$ [0 , 1]. The reward objectives are provided below:

Cross track error reward: Cross track error reward encourages the robot to move closer to the nearest waypoint and discourages moving away from the waypoint.

$$R_e = e^{-w_1 \times d_t} \tag{7}$$

Where w_1 is the weight and d_t is the euclidean distance from the closest waypoint.

Yaw error reward: Yaw error reward encourages robot to align with the desired heading angle and discourages higher heading angle error.

$$R_{\Theta} = e^{-w_2 \times \Theta} \tag{8}$$

Where w_2 is the weight and Θ is the heading angle error from the closest waypoint.

Velocity reward: The velocity reward incentivizes the agent to maintain a speed above the minimum threshold. Without this reward, the agent tends to output the minimum speed to avoid the accumulation of low rewards over time.

$$R_v = w_3 \times v \tag{9}$$

Early termination reward: This rewards adds a negative penalty to the total reward function to prevent the agent from undergoing resets before it reaches the maximum episode lengths of 1000 time steps.

$$R_{Termination} = \begin{cases} -1 & \text{if episodes} <= 1000 \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

Where w_3 is the weight and v is the normalized speed of the vehicle in the body frame of reference. The controller's primary objective is to minimize cross-track and heading angle errors. To achieve this, we have formulated a primary task reward with an exponential nature. Recognizing the importance of minimizing both errors simultaneously to avoid bias toward maximizing either objective, we have adjusted the yaw error reward by multiplying it with the cross-track error reward.

The secondary objective of the controller is to maximize the robot's speed while ensuring that cross-track and heading-angle errors are not compromised. To achieve this, we have formulated a speed reward of a linear nature and adjusted it by multiplying it with the cross-track and yaw error rewards.

IV. Simulation Experiments

A. AgileX Hunter SE Robot

Our candidate UGV is a mid-scaled platform AgileX Hunter SE as shown in Fig. 3 where an ongoing operationalization is in progress to test the zero-shot transfer of the HDRL framework on the real robot for future work. Additional results and hardware testing can be found at https://youtu.be/dryFZX5XkX8.

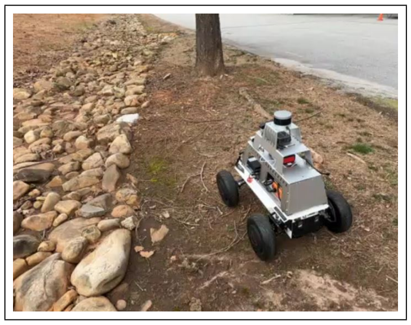


Fig. 3: Agilex HunterSE Robot

It is a rear-wheel drive Ackermann-steered robot equipped with LIDAR, camera, GNSS and IMU. Furthermore, it has suspensions at the rear wheels that provides additional stabilization. However, for simplification suspensions have not been modeled in simulation. The robot weighs 54.14 kilograms and has a wheelbase of 0.608 meters, a track width of 0.554 meters, and a maximum speed of 3 meters per second.

B. Terrain Features

This study employs NVIDIA's Isaac Sim as the simulator for training and evaluating controllers in simulation on rough terrain [12]. This choice is based on its capability to integrate rough terrain features with diverse conditions while providing high visual and physics fidelity.

The designed rough terrain encompasses a range of features, including varying slopes and stairs with gradient $\in [0^{\circ}, 30^{\circ}]$ as 30° is the maximum climbing capacity of the robot. As shown in Fig. 4, the comprehensive set of features is deliberately crafted to introduce diverse difficulty levels for effectively training and evaluating the controllers.

C. Training and Evaluation Setup

For training and evaluation purposes, the paths of the f1tenth racetrack centerlines are utilized with friction of 0.7 [13]. These paths are characterized by convoluted segments,

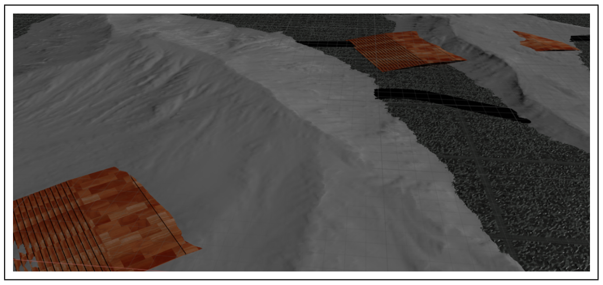


Fig. 4: Simulated Terrain

which introduce complexities that enhance the robustness of the training process. Specifically, the Austin racetrack is designated as the training path, while the Brands Hatch and Silverstone racetracks are employed to evaluate the performance and generelizability of the HDRL algorithm as shown in Fig. 5. For training, a single workstation was used consisting of an NVIDIA RTX A6000 GPU and 256 GB RAM.

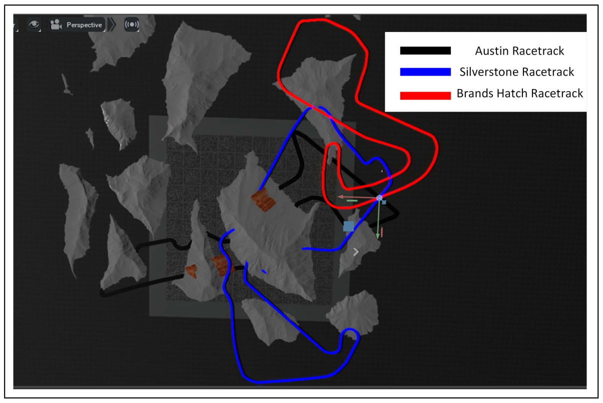


Fig. 5: Training and Evaluation Paths

In the HDRL framework, PPO is selected as the DRL algorithm due to its ability to mitigate significant policy deviations from the previous iteration. However, the PPO algorithm is an on-policy algorithm, which limits its sample efficiency. To overcome this limitation and enhance sample efficiency, multiple robots were deployed in parallel to collect data. Specifically, 32 robots were initialized at the origin of the training track. We use `rl_games` and omni-isaacgymenvs library to setup the parallelized training. The simulation is rendered at 60 Hz and the control commands to the robot are sent at 10 Hz to replicate the real-world scenario.

V. Results and Discussion

To assess the training performance of the HDRL controller, the HDRL controller is compared against an end-to-end PPO-based DRL controller, as illustrated in Fig. 6. The mean cumulative reward, depicted on a logarithmic scale, represents the average total reward across all agents at each time step. We examined both the off-policy algorithm Soft Actor-Critic (SAC) and the on-policy PPO. PPO outperformed SAC and demonstrated robustness to hyperparameter tuning, which motivated our selection of PPO for both HDRL and end-to-end DRL frameworks.

The HDRL controller undergoes training for 300 iterations, while the DRL controller is trained for 200 iterations. The total wall times for these training sessions are 75.66 minutes and 87.48 minutes, respectively. Notably, the HDRL controller exhibits a higher initial reward and converges to a higher reward value compared to the DRL controller, which appears to be trapped in a local optimum and converges to a suboptimal policy. Analysis of our evaluation outputs reveals that the DRL controller tends to traverse at minimal speed to mitigate error accumulation, resulting in convergence to a significantly lower reward value. In contrast, the HDRL controller generates safer policies from the outset and converges to an optimal policy faster.

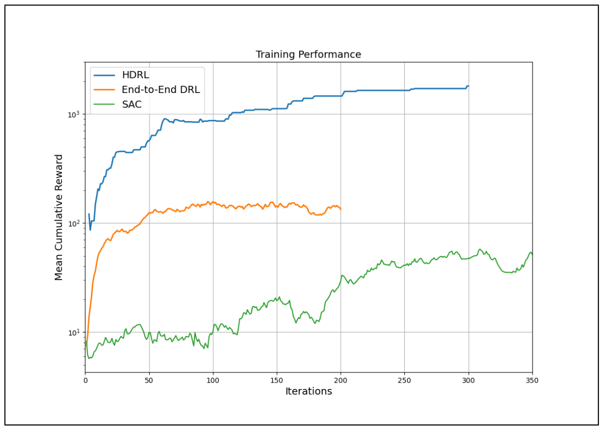


Fig. 6: Training Performance

To evaluate the generalizability and robustness of the HDRL controller, we benchmark it against several baselines using path-tracking outputs, cross-track errors, and yaw error metrics on unseen Brands Hatch and Silverstone racetracks. Specifically, the following baselines are considered. A tuned LQR controller, as described in Section III-A, has comparable rear-wheel joint velocity outputs to those of the HDRL controller while generating steering commands and an end-to-end PPO-based DRL controller that produces both rear wheel joint velocities and steering angles.

A. Crosstrack and Yaw Error Evaluation

The HDRL controller demonstrates robustness and generalizability across various tracks, including the Austin, Brands Hatch and Silverstone circuits, despite not being exposed to them during its training phase. Across all three tracks, the HDRL controller consistently outperforms both the LQR and DRL controllers in terms of minimizing cross-track error.

During evaluation, it became apparent that terrain slopes and sharp corners influence all the controllers. However, the DRL controller is the most affected, followed by the LQR controller. In contrast, the HDRL controller minimizes cross-track and yaw errors across varying terrain slopes. As depicted in Fig 7, the DRL controller effectively minimizes the yaw error, but with increasing elevation angle, it struggles to minimize the cross-track error. The LQR controller also minimizes the yaw error but is less effective at minimizing the cross-track error. In contrast, the HDRL controller compensates for varying slopes by introducing corrective steering angle outputs, enhancing overall performance.

While the HDRL and LQR controllers successfully traverse the path without resets, the DRL controller struggles, especially at steep slopes and corners. This difficulty stems from its tendency to converge towards suboptimal policies, leading to suboptimal speed control and slipping on steeper slopes and sharp corners. To quantify the results, key evaluation metrics are defined in Table I as mean squared error of both the crosstrack and yaw error across all the three tracks.

TABLE I: Evaluation Metrics

Error Type	Controller	Austin	Silverstone	Brands Hatch
Crosstrack	LQR	0.098	0.013	0.042
	HDRL	0.013	$4.95e^{-05}$	0.0795
	DRL	0.022	0.905	1.686
Yaw	LQR	$8.02e^{-05}$	$1.19e^{-05}$	$4.34e^{-06}$
	HDRL	$2.88e^{-07}$	$5.98e^{-06}$	$7.85e^{-06}$
	DRL	$2.29e^{-05}$	0.00022	0.0013

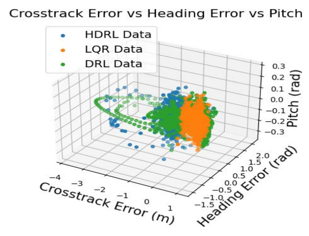
Disturbance rejection evaluations were done for all three controllers on the challenging Brands Hatch track. Disturbances were introduced every 10 seconds as random linear and angular velocities ranging $\in$ [-3 , 3] m/s and [-3 , 3] rad/s, respectively. As observed in Fig. 8, the HDRL controller effectively compensates for disturbances, ultimately minimizing cross-track and yaw errors. However, the LQR controller, once disturbed with a cross-track error more significant than 1 meter, only minimizes the yaw error. The DRL controller compensates for the disturbance but tends to overshoot the cross-track error, eventually minimizing only the yaw error. The data further indicates that the HDRL controller outperforms the LQR and DRL controllers in rough terrain path tracking scenarios, exhibiting lower cross-track and yaw errors, thereby enhancing performance and resilience against disturbances.

VI. Conclusion

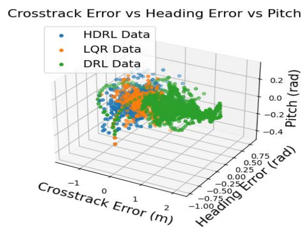
This study showcases the significant promise of the HDRL framework for improved rough terrain path tracking of an Ackermann-steered platform. It demonstrates robustness against external disturbances and showcases the ability to generalize across diverse conditions. Furthermore, it reduces the total training time needed to develop an optimal policy by leveraging DRL and LQR controllers' complementary strengths and addressing their individual limitations. Moreover, the HDRL framework alleviates the necessity for high-fidelity nonlinear model-based controllers, which can be challenging to both formulate and deploy in scenarios with high-dimensional state and action spaces.

Acknowledgment

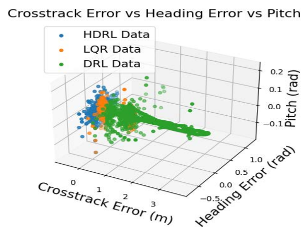
This work was supported in part by the US National Science Foundation under NSF IIS-1925500 and NSF CNS-1939058



(a) Austin Racetrack

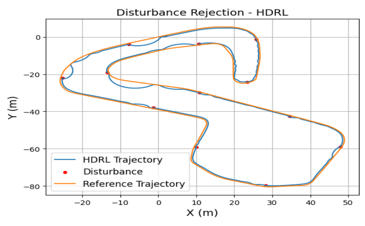


(b) Silverstone Racetrack

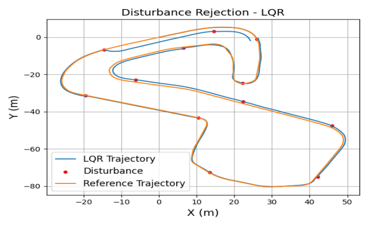


(c) BrandsHatch Racetrack

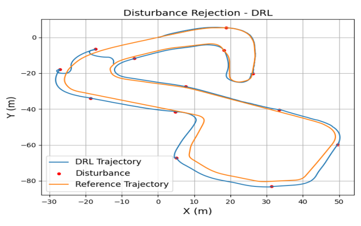
Fig. 7: Crosstrack Error vs Yaw Error vs Pitch Visualization



(a) HDRL Disturbance Rejection

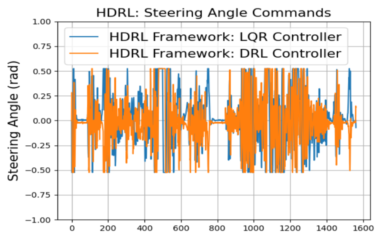


(b) LQR Disturbance Rejection

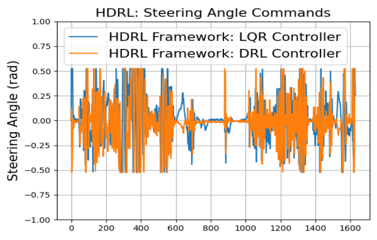


(c) DRL Disturbance Rejection

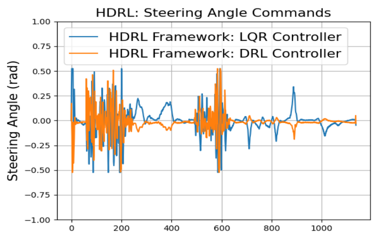
Fig. 8: Disturbance Rejection Visualization



(a) Austin Racetrack



(b) Silverstone Racetrack



(c) BrandsHatch Racetrack

Fig. 9: HDRL Steering Commands Visualization

References

[1] D. Mehta, K. Kosaraju, and V. Krovi, "Actively articulated wheeled architectures for autonomous ground vehicles - opportunities and challenges," *SAE Technical Paper*, 2023. [Online]. Available: https://doi.org/10.4271/2023-01-0109

[2] X. Chen, H. Peng, W. Wang, L. Zhang, Q. An, and S. Cheng, "Path tracking coordinated control strategy for autonomous four in-wheel-motor independent-drive vehicles with consideration of lateral stability," *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering*, vol. 235, pp. 1023–1036, 2020.

[3] S. Yu, C. Shen, and T. Ersal, "Nonlinear model predictive planning and control for high-speed autonomous vehicles on 3d terrains," *IFAC-PapersOnLine*, vol. 54, no. 20, pp. 412–417, 2021, modeling, Estimation and Control Conference MECC 2021.

[4] V. Wiberg, E. Wallin, T. Nordfjell, and M. Servin, "Control of rough terrain vehicles using deep reinforcement learning," *IEEE Robotics and Automation Letters*, vol. 7, no. 1, pp. 390–397, 2022.

[5] T. Johannink, S. Bahl, A. Nair, J. Luo, A. Kumar, M. Loskyll, J. A. Ojea, E. Solowjow, and S. Levine, "Residual reinforcement learning for robot control," *2019 International Conference on Robotics and Automation (ICRA)*, pp. 6023–6029, 2018. [Online]. Available: https://api.semanticscholar.org/CorpusID:54464184

[6] X. Hou, J. Zhang, C. He, Y. Ji, J. Zhang, and J. Han, "Autonomous driving at the handling limit using residual reinforcement learning," *Advanced Engineering Informatics*, vol. 54, p. 101754, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1474034622002129

[7] R. Trumpp, D. Hoornaert, and M. Caccamo, "Residual policy learning for vehicle control of autonomous racing cars," *2023 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1–6, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:256846509

[8] D. Baek, A. Purushottam, and J. Ramos, "Hybrid lmc: Hybrid learning and model-based control for wheeled humanoid robot via ensemble deep reinforcement learning," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022, pp. 9347–9354.

[9] A. Sakai, D. Ingram, J. Dinius, K. Chawla, A. Raffin, and A. Paques, "PythonRobotics: a Python code collection of robotics algorithms," 2018. [Online]. Available: https://arxiv.org/abs/1808.10703

[10] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *ArXiv*, vol. abs/1707.06347, 2017. [Online]. Available: https://api.semanticscholar.org/CorpusID:28695052

[11] A. Salvi, J. Coleman, J. Buzhardt, V. Krovi, and P. Tallapragada, "Stabilization of vertical motion of a vehicle on bumpy terrain using deep reinforcement learning*," *IFAC-PapersOnLine*, vol. 55, no. 37, pp. 276–281, 2022, 2nd Modeling, Estimation and Control Conference MECC 2022.

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

[13] J. Betz, H. Zheng, A. Liniger, U. Rosolia, P. Karle, M. Behl, V. Krovi, and R. Mangharam, "Autonomous vehicles on the edge: A survey on autonomous vehicle racing," *IEEE Open J. Intell. Transp. Syst.*, vol. 3, pp. 458–488, 2022.