Towards Off-road Autonomous Driving

Zhouchonghao Wu

CMU-RI-TR-25-55

June 30, 2025

The Robotics Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA

Thesis Committee:
Jeff Schneider, *chair*
Guanya Shi
Samuel Triest

Submitted in partial fulfillment of the requirements for the degree of Master of Science in Robotics.

To my family and friends, and to the robots I made along the way.

Abstract

Off-road autonomous driving poses significant challenges such as navigating unmapped, variable terrain and handling uncertain, diverse dynamics. Addressing these challenges requires effective long-horizon planning and adaptable control. Many existing methods employ either Model Predictive Control (MPC) or Reinforcement Learning (RL). MPC methods rely on dense sampling and accurate dynamics models, making them computationally expensive and unsuitable for real-time long-horizon planning. In contrast, RL methods are computationally efficient at deployment but struggle with exploration in obstacle-dense and unpredictable terrains. To overcome these limitations, we propose a hierarchical autonomy pipeline consisting of a low-frequency global planner and a high-frequency local RL controller. To address the exploration challenges in RL, we introduce a teacher-student learning paradigm that enables end-to-end training of an RL policy capable of real-time control in complex environments. We present a novel policy gradient method that extends Proximal Policy Optimization (PPO), incorporating off-policy trajectories for teacher supervision and on-policy trajectories for student exploration. Our method is evaluated in a realistic off-road simulation environment and demonstrates superior performance compared to baseline RL and imitation learning approaches. It is further deployed on a high-performance real-world vehicle, showcasing its practical applicability.

Acknowledgments

First and foremost, I would like to express my deepest gratitude to my advisor, Professor Jeff Schneider, for his invaluable mentorship, guidance, and encouragement throughout my graduate studies. His insights and support have been instrumental in shaping both my academic and personal growth.

I would like to extend my appreciation to the engineers at the National Robotics Engineering Center (NREC) for their generous support and technical assistance with our hardware experiments. Their expertise and commitment greatly contributed to the success of this project.

I am grateful to Professor Katerina Fragkiadaki and her research team for the opportunity to collaborate and for their valuable contributions to our work. Their perspectives enriched the direction and scope of the research.

I am sincerely thankful to my project teammates–Vedant, Raymond, and Anoushka–for their collaboration, dedication, and shared effort throughout the course of this work. Working together has been both productive and fulfilling.

Finally, I wish to thank my family, lab mates, and friends for their unwavering support, encouragement, and patience throughout this journey. Their presence has been a constant source of motivation and strength.

Contents

When this dissertation is viewed as a PDF, the page header is a link to this Table of Contents.

List of Figures

List of Tables

Chapter 1

Introduction

Autonomous ground vehicles have advanced significantly in recent years, with applications such as delivery robots [11] and self-driving taxis [2]. While great progress has been made in autonomous driving in structured, urban environments, navigation in off-road terrain remains a major challenge. Autonomous driving in off-road scenarios presents distinct challenges compared to structured on-road scenarios. These challenges arise from the complex and uncertain dynamics between off-road terrains and vehicles, the presence of diverse surface conditions that demand different driving strategies, and the need for online planning to adapt to variable conditions.

Modeling vehicle-terrain interaction dynamics is challenging due to factors such as loose particles, uneven surfaces, and slopes, all of which affect traction and vehicle stability. The challenge is further intensified by the wide variety of terrains and obstacles a vehicle may encounter, such as sand, gravel, rock formations, and vegetation, each of which requires unique traversal techniques. To address these challenges, a versatile control policy is necessary to avoid obstacles, traverse through ditches and adapt to different terrain types. Moreover, in off-road scenarios, the significant variance in traversal difficulty across different trajectory segments means that a globally optimal trajectory may not necessarily be locally optimal. This necessitates long-term trajectory planning in addition to real-time adaptability. Thus, autonomous off-road navigation is a complex problem that demands both adaptive, multi-skill control and long-horizon planning.

Existing approaches to off-road autonomy rely on sampling-based Model Predictive

Control (MPC) methods [7, 37]. These methods require dense rollout sampling and a highly accurate vehicle dynamics model to effectively navigate obstacles and traverse diverse terrains. However, both dense sampling and accurate dynamics modeling are computationally intensive, making these methods impractical for real-time execution of globally optimal trajectory planning and terrain navigation.

Reinforcement learning (RL) methods are also used for tackling complex and high-dimensional sequential decision-making tasks that are often challenging for traditional control methods. RL models typically utilize neural networks with a limited number of layers, enabling rapid inference while minimizing the computational load on the vehicle. Its application in off-road driving has typically targeted handling complex terrain interactions without addressing the planning component [15] or focused on simplified driving environments [9, 38]. Despite their advantages in computational speed, existing RL-based methods prove insufficient for the planning component of the off-road autonomy task.

In this thesis, we investigate hybrid approaches to off-road autonomy by exploring how to combine the strengths of both MPC and RL methods. In particular, we focus on leveraging demonstrations generated by MPC-based techniques to accelerate the training of RL controllers and enable real-time long-horizon off-road autonomy. We introduce Teacher Action Distillation with Policy Optimization (**TADPO**), a novel extension of Proximal Policy Optimization (PPO) that enables concurrent learning from fixed expert demonstrations and on-policy environment interactions to tackle long-horizon planning and hard exploration problems. We devise a training regime using TADPO to distill the behavior of a computationally expensive but high performance controller into an RL control policy to achieve similar performance in real time for off-road autonomy. We develop a hierarchical, end-to-end autonomy pipeline capable of real-time, long-range navigation in complex off-road terrains by integrating a TADPO-based controller, which leverages detailed on-board sensor data, with a global planner that utilizes a coarse global map.

Chapter 2

Related Work

2.1 Model Predictive Control

Model Predictive Path Integral (MPPI) [39] and Cross-Entropy Method (CEM) [18] are families of MPC methods conventionally employed in control tasks with complex dynamics. MPPI, in particular, has been applied in the off-road autonomy domain in previous works [7, 22, 35]. These methods typically focus on either the high-level planning or low-level control, but not both. Although the sampling methods vary between these techniques, they all require sampling a large number of trajectories to select a feasible action sequence that minimizes a cost function. While they are effective for generating control action in complex, nonlinear systems, the dense sampling required renders real-time operation of long-horizon planning intractable. Some attempts like RL+MPPI [31] and TD-MPC [8] have been made to improve sampling efficiency by learning a state-dependent control action distribution and learning a terminal value function, thereby reducing required number of samples and planning horizon.

2.2 Reinforcement Learning

Proximal Policy Optimization (PPO) and Soft Actor-Critic (SAC) are common modern RL techniques for solving complex robotics tasks. PPO [34] is a on-policy RL

framework that performs stable policy learning by limiting policy updates through a clipped surrogate objective function. SAC [6] is an off-policy RL algorithm that optimizes a stochastic policy and value function, enabling efficient and stable learning for continuous control tasks. Both methods have been successfully applied to robotics tasks involving visual inputs and continuous action spaces, including dexterous manipulators, bipedal robots, and unmanned aerial and ground vehicles [1, 6, 27, 30, 36]. Despite their successes, these algorithms encounter unique challenges in the proposed off-road driving problem, including navigating diverse terrains and long-horizon planning.

Some works have explored end-to-end RL methods for off-road [9, 15, 38] and on-road [4, 12, 17] driving. Previous works in the off-road domain are limited, as vehicles often focus solely on immediate obstacle avoidance without incorporating longer-term planning capabilities, or are tested in unrealistic simulations. Meanwhile, on-road driving research typically focuses on the unpredictable behavior of other road users, rather than addressing the variability of terrain encountered in off-road environments. An issue among RL methods presented in these works is their inability to explore efficiently, rendering them ineffective in obstacle-rich environments where simulation is computationally expensive and dynamics are highly complex. As a result, exploration in these scenarios is challenging without external guidance.

2.3 Imitation Learning and Learning from Demonstration

DAgger [33] is an Imitation Learning (IL) method that supervises policy learning by allowing queries to a teacher policy during training. Initially, the student policy is initialized by behavior cloning (BC) using a teacher policy. Subsequently, the student policy is enhanced by penalizing the discrepancy between the actions predicted by the student and teacher at each encountered state. Implicit Q-learning [20] extends Q-learning and actor-critic methods as an offline reinforcement learning technique that estimates Q-values without directly optimizing a policy. This approach enables the agent to implicitly select actions that maximize the value function.

There have been attempts to incorporate demonstrations in PPO, though with

notable limitations. PPO+D [23] extends PPO by incorporating a single off-policy trajectory into the training process. This approach modifies the PPO replay buffer to include three components: D_r for successful trajectories, D_v for failure trajectories and D for the currently sampled trajectories. When sampling from D_v, the paper employs value-based sampling, which becomes impractical for tasks that involve large replay buffers with visual inputs. In off-road driving, navigating diverse terrains requires a broad range of skills, which substantially increases the size of the teacher demonstration and the failure replay buffers in PPO+D. This necessity renders PPO+D unsuitable for the task.

There have also been a few works for learning policies from demonstrations through a teacher-student framework in autonomous driving. Peng et al. [29] uses SAC, an off-policy method, to choose actions between a teacher and student policy to solve simple tasks including lane following and obstacle avoidance. Some teacher-student paradigms, such as those using Deep Q-network [10], focus on discrete action spaces. Other methods [16, 24] use SAC to update a student policy, but in complex planning and control tasks, these methods exhibit low stability during training [13].

Chapter 3

Background

We model the control problem of an off-road autonomous vehicle as a Markov Decision Process (MDP), represented by the tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, r, \gamma)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $P(s'|s, a)$ is the transition dynamics function, $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, and $\gamma \in [0, 1)$ is the discount factor. The objective is to identify an optimal policy π^* such that

$$\pi^* = \arg\max_{\pi} \mathbb{E}_{\pi} \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \right] \tag{3.1}$$

where a policy is a mapping from state s_t to a probability distribution of actions $p(a_t)$. Policy gradient methods are a common family of methods that directly optimize the expected return by adjusting the parameters of a stochastic policy in the direction of the gradient of the objective. Value functions, such as the state-value function $V^{\pi}(s)$ or the action-value function $Q^{\pi}(s, a)$, estimate expected future rewards.

3.1 Proximal Policy Optimization

Proximal Policy Optimization (PPO) [34] is an on-policy algorithm in the Policy Gradient family. Given a θ-parameterized policy π_θ and a set of trajectories (commonly referred to as a "rollout") collected by it, PPO employs an actor-critic architecture where the actor learns the policy and the critic estimates the value function to guide

policy improvement, maximizing a clipped surrogate objective to update the policy as in Equation (3.5).

$$L^{\text{CLIP}}(\theta) = \mathbb{E}_t \left[\min \left(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t \right) \right] \tag{3.2}$$

$$L^{\text{VF}}(\theta) = \mathbb{E}_t \left[(V_{\pi_{\theta_{\text{old}}}}(s_t) - R_t)^2 \right] \tag{3.3}$$

$$L^{\text{entropy}}(\theta) = \mathbb{E}_t \left[-H[\pi_\theta(\cdot|s_t)] \right] \tag{3.4}$$

$$L^{\text{PPO}}(\theta) = L^{\text{CLIP}}(\theta) - c_1 L^{\text{VF}}(\theta) + c_2 L^{\text{entropy}}(\theta) \tag{3.5}$$

where $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}$ is the ratio of the probability of the action a_t under the current policy (π_θ) to the policy used to collect the rollout ($\pi_{\theta_{\text{old}}}$), $\hat{A}_t = \sum_{i=t}^{t+T} (\gamma\lambda)^{i-t}\delta_i$ is the estimate of the advantage, with $\delta_t = R_t + \gamma V_{\pi_{\theta_{\text{old}}}}(s_{t+1}) - V_{\pi_{\theta_{\text{old}}}}(s_t)$, $R_t = \sum_{i=t}^{t+T} \gamma^{i-t} r(s_i, a_i) + \gamma^{T-t+1} V(s_{T+1})$ is the discounted return, and T is the number of transitions, $H[\pi_\theta(\cdot|s_t)]$ is the entropy of the action distribution induced by the policy given the state s_t, $V_{\pi_{\theta_{\text{old}}}}(s_t)$ is the expected return of the policy at state s_t, and c_1, c_2 are constants.

The advantage estimator $\hat{A}_t$ quantifies the relative benefit of executing action a_t over the expected return $V_{\pi_{\theta_{\text{old}}}}(s_t)$ of $\pi_{\theta_{\text{old}}}$ at s_t. This formulation implies that the policy gradient update derived from Equation (3.5) remains sound only when $V_{\pi_{\theta_{\text{old}}}}$ maintains a faithful approximation of the policy's expected returns. This insight forms a critical foundation for the formulation of Equation (4.4) in our proposed method.

For tasks involving complex long-horizon planning, PPO faces significant challenges in exploration, often resulting in an inability to learn effective policies [23]. The introduction of undirected randomness in the actor's exploration strategy often leads to inefficient sampling, as random actions frequently fail to align with task objectives. This misdirected exploration can impede policy improvement, particularly in complex environments that require targeted exploration. While distilling planning knowledge from a teacher expert could potentially address this issue, PPO's on-policy nature precludes the direct utilization of off-policy data such as expert demonstrations. This limitation significantly constrains PPO's applicability in domains where effective exploration is challenging.

3.2 Model Predictive Control

Model Predictive Control (MPC) is a control framework that employs sampling or optimization techniques to minimize a cost function, rendering it effective for generating control actions in complex, nonlinear systems.

The optimal action a^* is determined by:

$$a^* = \arg \min_{a_0 \ldots a_h \in \mathcal{A}} \sum_{i=0}^{h} C(\hat{s}_i, a_i) \tag{3.6}$$

where C represents the cost function, $\hat{s}_i$ and a_i denote the model-predicted state and sampled action at step i, respectively, and h is the planning horizon. Prominent techniques for selecting optimal actions are Cross-Entropy Method (CEM) [18] and Model Predictive Path Integral (MPPI) [39]. CEM is an iterative, sampling-based optimization technique that progressively refines a probability distribution over control parameters. In contrast, MPPI is a sampling-based method that leverages importance-weighted optimization to generate control outputs. MPPI has gained prominence in recent literature due to its high degree of parallelizability and rapid convergence properties [7, 21, 22, 25, 31].

Chapter 4

Methods

4.1 TADPO: Teacher Action Distillation with Policy Optimization

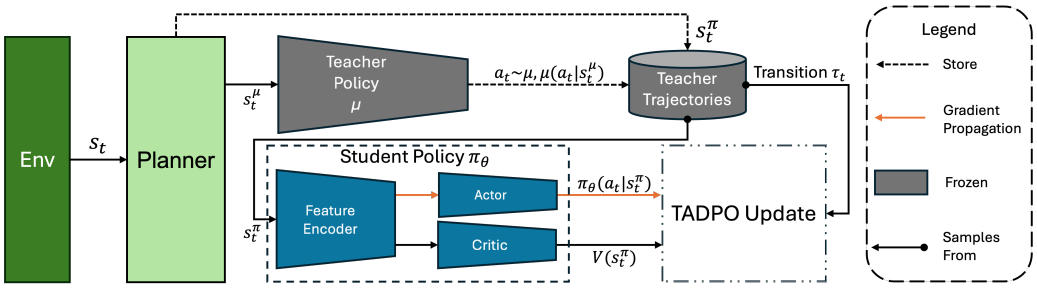

Figure 4.1: Teacher Action Distillation Rollout and Update Process. Teacher demonstration buffer is frozen while training the student policy. The student policy would performs TADPO update with a probability p. This process performs a gradient update solely on the actor and the feature encoder of the policy, using the critic to estimate the advantage of the teacher rollout over the student for any environment state.

We propose a novel method to train a θ-parameterized policy π_θ by extending PPO to incorporate demonstrations from a teacher policy μ. This extension allows π_θ to be trained concurrently using its own on-policy rollouts and demonstrations by the teacher policy.

4.1.1 Teacher Action Distillation Policy Gradient

Given a pre-trained teacher policy μ, we define a loss function L^{TAD} in Equation (4.1) to train a student policy π_θ. This loss is computed solely on teacher rollouts, where actions at each time step t are sampled from the teacher policy, i.e., $a_t \sim \mu$.

$$L^{\text{TAD}}(\theta) = L^{\mu}(\theta) + c_2 L^{\text{entropy}}(\theta) \tag{4.1}$$

$$L^{\mu}(\theta) = \mathbb{E}_{a_t \sim \mu}\big[\max(0, \min(\rho_t(\theta), 1+\epsilon_\mu)\hat{\Delta}_t)\big] \tag{4.2}$$

$$\rho_t(\theta) = \frac{\pi_\theta(a_t \mid s_t^\pi)}{\mu(a_t \mid s_t^\mu)} \tag{4.3}$$

$$\hat{\Delta}_t = R(a_t, s_t) - V_{\pi_{\theta_{\text{old}}}}(s_t^\pi) \tag{4.4}$$

where L^{entropy} is defined in Equation (3.4) and ϵ_μ is a hyperparameter. It is important to note that μ and π_θ can be defined to operate on distinct observation spaces, denoted by s_t^μ and s_t^π respectively, despite being derived from the same underlying environment state s_t.

In Equation (4.3), $\rho_t(\theta)$ is defined as the ratio of probability of a_t under π_θ to the probability under μ. This concept is analogous to the probability ratio r_t in PPO. In Equation (4.4), $\hat{\Delta}$ estimates the advantage of the achieved return of the teacher rollout at state s_t over the expected return of the student. We observe that applying the policy gradient update from L_μ to the student policy exclusively when $\hat{\Delta} > 0$ (i.e., when the teacher outperforms the student's expectations) results in stable updates and higher reward attainment by the student.

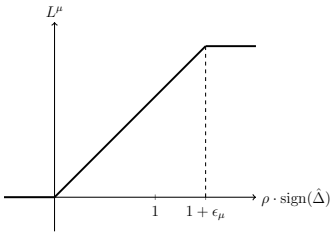

Figure 4.2: A single timestep of the teacher distillation loss function L^μ as a function of $\rho \cdot \text{sign}(\hat{\Delta})$. Analogous to PPO, this modified clipping mechanism facilitates stable training and convergence to superior reward outcomes.

In Equation (4.2), L_μ is clipped as illustrated in Figure 4.2. This clipping mechanism prevents policy gradient updates when ρ_t exceeds $1 + \epsilon_\mu$, effectively halting updates when the probability ratio of action a_t under π_θ relative to μ surpasses this threshold, indicating that the student policy has already captured the desired behavior.

Thus, L^{TAD} in Equation (4.1) ensures that the policy gradient propagates only when two conditions are met: (i) the teacher rollout's return exceeds the student policy's expected return, and (ii) the student policy's probability of executing action a_t is not substantially higher than that of the teacher policy. Analogous to PPO, L^{entropy} in Equation (4.1) modulates the student policy's exploration.

4.1.2 Training Procedure

TADPO involves learning a student policy π_θ thorugh concurrent utilization of teacher demonstrations and student rollouts. Transitions are collected from both the pre-trained teacher policy μ and the student policy π_θ being trained, and stored in separate teacher and student buffers, respectively. With probability p, the algorithm samples transitions from the teacher's buffer and performs a TADPO update, which likely involves distilling knowledge from the teacher. Otherwise, it samples from the student's buffer and performs a standard PPO update. As shown in Algorithm 1, this alternating process continues for multiple iterations and epochs, allowing the student to learn from both its own experiences and the teacher's expertise. In our implementation, $\hat{\Delta}_t$ is normalized to have unit standard deviation within each mini-batch.

As shown in Figure 4.1, during the TADPO update, gradient propagation occurs exclusively through the actor and feature encoder components of the student policy π_θ, while the critic remains frozen, ensuring the value function maintains independent state-value estimates based solely on the student's experiences.

4.2 End-to-end Off-road Autonomy in Simulation

We employ a hierarchical architecture to achieve end-to-end off-road autonomy. Given a final goal $\mathbf{p}_g$, a global planner (implemented using A*) generates sparse (80m)

Algorithm 1 Training procedure for TADPO combining PPO and imitation learning.

procedure TADPO(μ, π, p)
  Input: Teacher μ, Student π, Sampling prob. p
  Return: Student policy params θ
  $\mathcal{B}_\mu \leftarrow N_\mu$ transitions: $\{\tau_{t_{a_t \sim \mu}} = (s_t^\mu, a_t, R_t, \mu(a_t|s_t^\mu))\}$
  for iter = 1 to I **do**
    $\mathcal{B}_\pi \leftarrow N_\pi$ transitions:
    $\{\tau_{t_{a_t \sim \pi_{\theta_{\text{old}}}}} = (s_t^\pi, a_t, R_t, \pi_{\theta_{\text{old}}}(a_t|s_t^\pi))\}$
    for epoch = 1 to K **do**
      while $\mathcal{B}_\pi \neq \emptyset$ **do**
        Sample $r \sim \mathcal{U}(0, 1)$
        if $r > p$ **then**
          Sample n transitions $\tau \sim \mathcal{B}_\pi$ w/o replacement
          $\theta \leftarrow$ PPOUpdate(τ)
        else
          Sample n transitions $\tau \sim \mathcal{B}_\mu$ w/o replacement
          $\theta \leftarrow$ TADPOUpdate(τ)
        end if
      end while
      Reset $\mathcal{B}_\mu$, $\mathcal{B}_\pi$
    end for
  end for
  return θ
end procedure

waypoints utilizing a coarse global map. These waypoints are tracked by an RL controller trained using TADPO. As the globally planned sparse waypoints may be suboptimal and fail to account for all obstacles, the RL controller must incorporate long-horizon planning capabilities to effectively track these waypoints. A detailed exposition of the implementation and code specifics is provided in Appendix L.

4.2.1 Training

An MPPI controller is designed to interpolate the sparse waypoints in accordance with [7], with the same cost function. This controller generates dense (16m) waypoints that are subsequently used for training the teacher policy. A comprehensive description of the MPPI controller is provided in Appendix E. This controller generates waypoints that result in a high success rate when substantial computational resources are available. However, its performance deteriorates significantly under real-time computational constraints, as shown in Table 6.1. These computationally expensive yet successful trajectories are precomputed and stored, thus removing the need for online controller execution during the teacher training process.

The teacher policy μ is trained via PPO using dense waypoints generated by the MPPI controller. Hence, this policy learns to perform dense waypoint tracking without requiring the planning capabilities to manuevers through ditches and avoid obstacles.

The student policy π_θ is then trained to distill the teacher behavior via the TADPO training procedure in 4.1.2 while operating solely with sparse waypoints provided by the global planner. Hence, training with sparse waypoints, requires π_θ to learn sophisticated planning capabilities to manuever through ditches and avoid obstacles.

Both the teacher and the student policies adhere to the reward function in 4.2.3. The corresponding observation and action spaces for these policies are explained in 4.2.4.

4.2.2 Deployment

During deployment, we employ an A* global planner that generates sparse waypoints for navigation. These waypoints are subsequently tracked by the policy π_θ, establishing an end-to-end framework for off-road autonomous navigation. This hierarchical

approach enables efficient long-range navigation while allowing the policy to handle local terrain traversal and obstacle avoidance. More details about the global planner can be found in Appendix C.

4.2.3 Reward Function

The reward function for training both teacher and student policies comprises of five essential components: a progress reward that incentivizes goal-directed movement by measuring the reduction in distance to the waypoint from the vehicle's position, penalties for collisions and vehicle damage to ensure safety, a jerk penalty to discourage abrupt acceleration changes, and a success reward for waypoint achievement. Detailed specifications of the reward function are provided in Appendix J.

4.2.4 Observation and Action Spaces

We combine proprioceptive states with visual input for both the teacher and the student policies. The proprioceptive observation includes the vehicle's normalized speed, roll, pitch, and encodings of the current and next waypoint are provided, with the teacher using densely planned waypoints and the student using sparsely planned waypoints, as detailed in 5.1.1 and Appendix F.

The visual observations consist of top-down views for planning and a forward camera for obstacle avoidance. The top-down views are fully-observed. While the teacher policy uses a local top-down image, the student policy uses a wider-area image with a lower spatial resolution to planning. Both utilize a stack of 3 historical frames for the final observation. The controller directly outputs throttle and steering commands. More information about the observation and action spaces can be found in Appendix F and Appendix G respectively.

4.3 End-to-end Off-road Autonomy in Real World

We deploy our policy in a real-world vehicle. The autonomy stack is largely similar to that used in simulation. Several specific adaptations are made to the task setup, however:

- Due to limitations of the control systems available on the vehicle, we use velocity control for both throttle and steering.
- To limit complexity and reduce observability problems, we employ a limited observation space. The observation space includes linear forward velocity, yaw rate, a forward-facing 224x224 RGB image with 60 degrees field-of-view, and goal-conditioning representations from the previous section.
- Because of the changed observation space, we reduce the sparse and dense waypoint distances from 80m/6m to 15m/4m respectively to allow for deployment without a requisite mapping software stack.

To enable sim-to-real transfer on policies trained for this task, we used pretrained feature encoders to reduce distribution shift between the simulation and real-world visual inputs. Our training process, as shown in Figure 4.4, consists of three stages. First, we train a state-only PPO policy using dense waypoints to establish a strong baseline. Next, we generate a repository of trajectories by executing the teacher policy with added noise; during this phase, the teacher continues to use state-only observations and dense waypoints, while we simultaneously collect the student's visual observations and sparse waypoints. Using this data, we pretrain a feature encoder that combines DINOv2 [28] with a VAE architecture. Finally, we freeze the pretrained encoder and use it to train the student policy with TADPO+PPO.

Figure 4.5 shows the network architecture for the pretrained encoder. DINOv2 encodes the image into a set of tokens, which are then reduced in dimensionality and combined with proprioceptive states to produce a VAE latent state. Supervision is applied in two ways. First, the state embedding and goal conditioning are used to predict rewards, damage, and trajectory termination, while only the goal conditioning is passed to the decoder to facilitate future experiments with frame stacking and recurrence. Second, a separate dynamics MLP predicts the next state embedding, and an MSE loss is applied to enforce temporal consistency.

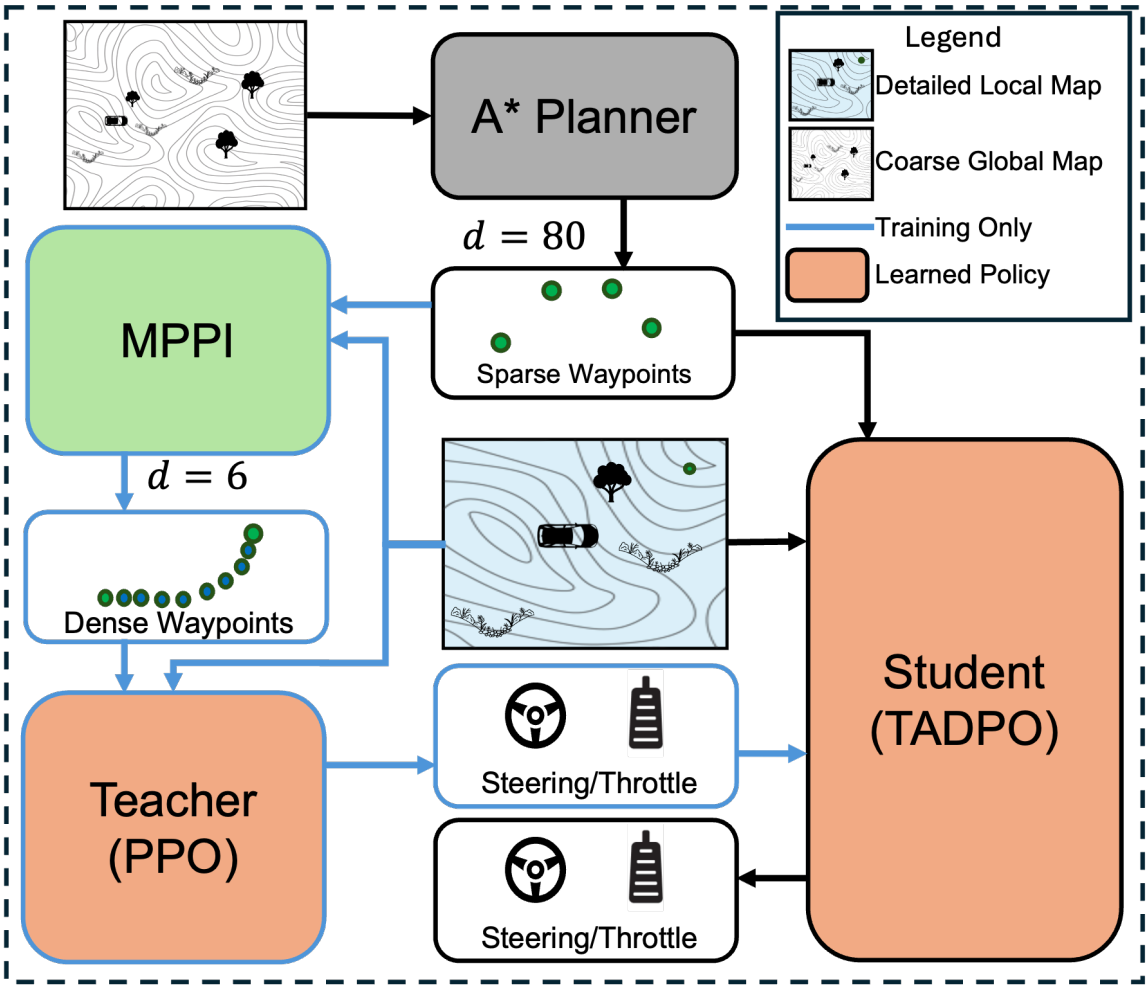

Figure 4.3: Illustration of the proposed hierarchical autonomy pipeline. During training, the proposed approach uses an MPPI-based controller to interpolate sparse waypoints provided by the global planner and tracks the interpolated dense waypoints using a teacher policy, thereby creating demonstration trajectories for training of the TADPO policy. During deployment, the trained TADPO policy uses only sparse waypoints and the detailed local map. d denotes waypoint distances in meters.

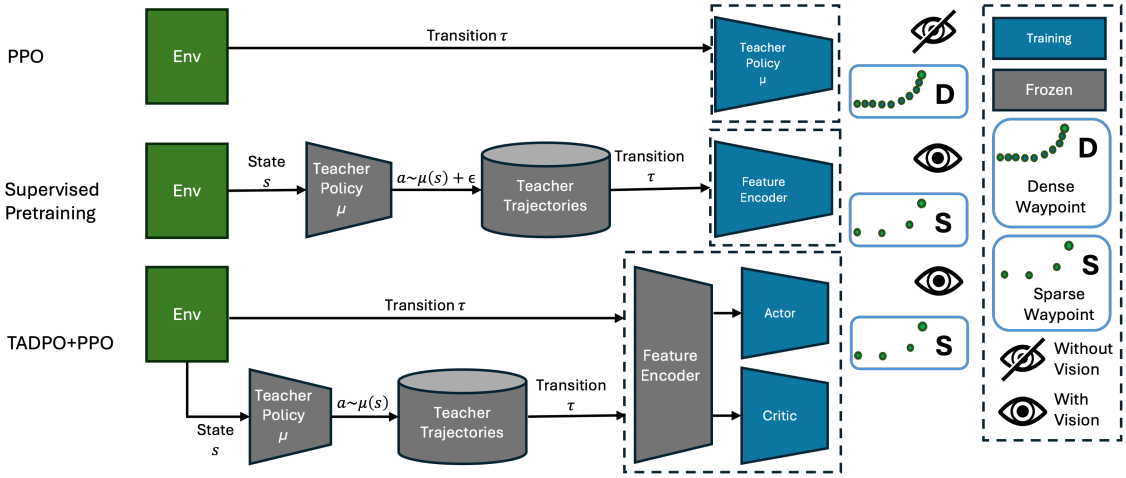

Figure 4.4: Deployed policy pretraining scheme. Training consists of three stages: (1) a state-only PPO policy is trained with dense waypoints; (2) noisy teacher rollouts are used to collect student visual observations and sparse waypoints, enabling pretraining of a DINOv2+VAE feature encoder; (3) the frozen encoder is used to train the student policy with TADPO+PPO.

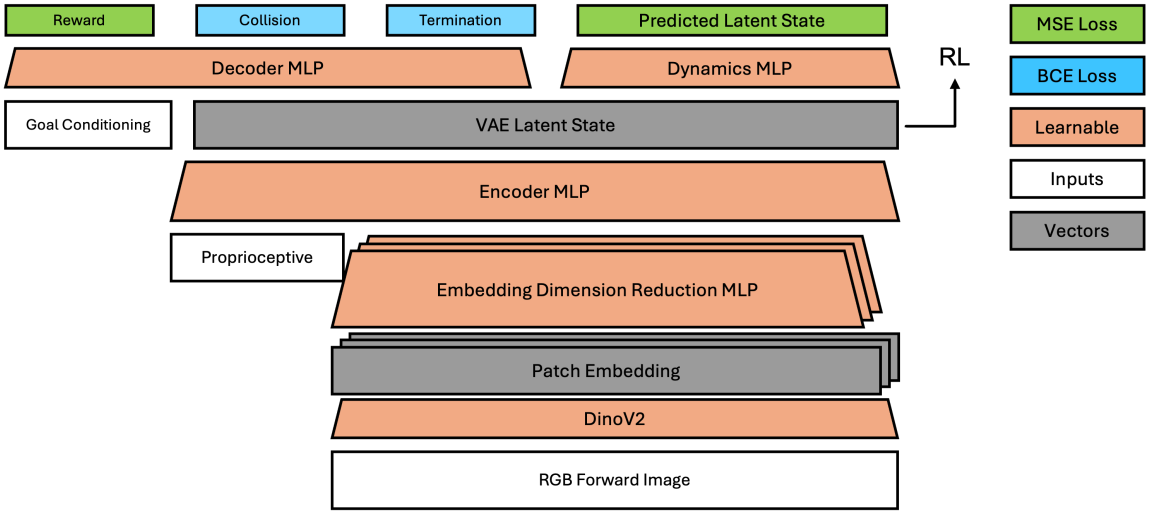

Figure 4.5: Feature encoder architecture and losses applied.

Chapter 5

Experiments

5.1 Experiment Setup in Simulation

5.1.1 Training, Demonstration, and Testing Datasets

We utilize `BeamNG` [3] as our simulation platform for the offroad vehicle. Additional details about the simulator are provided in Appendix K.

We train both the teacher and student policies on an extensive desert terrain map within the simulator. For each start-goal pair, sparse waypoints at 80 m intervals are generated using a coarse global map and an A* global planner. We choose a fixed set of start-goal pairs for teacher training and demonstration trajectories. Using an MPPI controller equipped with detailed local semantic segmentation and depth information, we generate dense waypoints at 6 m intervals between these sparse waypoints. A separate set of trajectories is created for policy evaluation. For methods where expert demonstrations are required, we use the dense waypoints generated by the MPPI controller.

The selected trajectories encompass diverse off-road challenges, as shown in Figure 5.1, and can be categorized into terrains featuring: (i) obstacles, (ii) extreme slopes, or (iii) hybrid. The obstacle-rich terrains include both natural (boulders, trees) and artificial (parked trailers, fences) obstacles, while extreme slopes comprise ditches and sandy cliffs.

For teacher demonstrations, we collect 15 trajectories each for categories (i) and

(ii), and 20 for category (iii). The evaluation set consists of 8 trajectories each for categories (i) and (ii), and 15 for category (iii). Further, we established 6 hybrid evaluation trajectories in grassland (4) and asphalt (2) terrain to demonstrate the generalizability of the RL controller. These terrains are very distinct from the training and evaluation sets in terms of the visual inputs and the terrain-vehicle dynamics. A comparison of these environments can be seen in Figure 5.1c.

5.1.2 Evaluation Metrics

We evaluate policy performance across test trajectories. For policies that produce an action distribution, the mode of the distribution is chosen as the selected action. All baseline methods operate in conjunction with an A* planner that provides sparse waypoints as input to the respective policies. For each episode, we define the following metrics, normalized to the range [0,1]:
(i) Success Rate (`sr`): Binary indicator where $\texttt{sr} = 1$ if the vehicle reaches within radius r of the goal position, and $\texttt{sr} = 0$ otherwise.
(ii) Completion Percentage (`cp`): Measures the maximum progress toward the goal position, normalized by the initial goal distance.
(iii)Mean Speed (`ms`): Average vehicle speed during the episode.
Further details regarding the metrics are provided in Appendix H.

5.2 Experiment Setup in Deployment

We conduct preliminary testing of our policy on SaberCat, a vehicle built by National Robotics Engineering Consortium. The test is conducted at Gascola Slag Dump over dirt roads cut through a forest terrain. Due to time constraints, we only perform experiments in 2 tasks: (i) trail following and (ii) obstacle avoidance. (i) involves traversing an uneven terrain covered in dirt and mud while navigating to a predetermined waypoint. (ii) requires the autonomous robot to avoid traffic barrel obstacles. Figure 6.1 shows the vehicle on a test course.

(a) Training set examples.

(b) Test set examples.

(c) Comparison of Desert (left), Grassland (middle), and Asphalt Terrains (right).

Figure 5.1: (a) shows example trajectories in the train set. Blue dots are waypoints as observed by the teacher, and red dots are waypoints as observed by the student. (b) shows example trajectories from the test set. Both cover a diverse set of terrain that include obstacles, ditches, and cliffs. (c) shows the difference between Desert, Grassland, and Asphalt environments used to demonstrate generalizability of the RL controller. Training is done only in desert terrain, and generalizability tests are done in grassland and asphalt terrains.

Chapter 6

Results and Discussions

6.1 Experiments in Simulation

6.1.1 MPC Baselines

In Table 6.1, MPC (Nonreal-time) baselines demonstrate the performance of CEM, MPPI and RL+MPPI controllers with a long planning horizon h and high number of samples N. These controllers calculate the next waypoints while the simulation is paused, enabling them to determine the next action before resuming the simulation. The results show that, given a sufficient number of samples and an extended planning horizon, these controllers achieve similar levels of performance.

MPC (Real-time) baselines demonstrate the performance of these controllers when operating under real-time constraints with a limited computational budget. We observe that the time of inference is more sensitive to h than N, which reduces long-horizon understanding and, in turn, degrades real-time performance. In contrast to MPPI, CEM employs a more iterative approach to sampling and evaluating action sequences, which results in a higher computational cost. RL+MPPI improves upon MPPI by incorporating a learned terminal value function and a state-dependent action distribution, reducing the N and h requirement. However, all of these methods experience a substantial decline in performance when executed with a reduced computational budget for real-time inference.

Hyperparamters are provided in Appendix D, and details about MPPI implemen-

tation are provided in Appendix E.

	Controller	Extreme Slopes			Obstacles			Hybrid			
		sr	cp	ms	sr	cp	ms	sr	cp	ms	ti
	MPPI + Teacher	0.88	0.96	5.83	1.00	1.00	5.91	0.94	0.96	5.69	2.02
MPC (Nonreal-time)	CEM + PID	0.88	0.96	5.51	1.00	1.00	5.16	0.87	0.94	5.13	3.47
	MPPI + PID	0.88	0.96	5.39	1.00	1.00	5.87	0.87	0.94	5.43	2.02
	RL+MPPI + PID	0.88	0.96	5.26	1.00	1.00	5.88	0.87	0.94	5.40	1.77
MPC (Real-time)	CEM + PID	0.38	0.49	**5.52**	0.25	0.38	5.16	0.27	0.43	5.13	0.13
	MPPI + PID	0.38	0.57	5.43	0.25	0.48	**5.48**	0.27	0.46	5.54	0.12
	RL+MPPI + PID	0.38	0.61	5.32	0.25	0.50	5.46	0.27	0.52	**5.63**	0.12
	TADPO†	**0.75**	**0.87**	4.99	**0.85**	**0.96**	5.26	**0.67**	**0.88**	5.30	**0.002**
RL/IL (Real-time)	DAgger	0.00	0.58	1.96	0.00	0.83	1.62	0.00	0.79	1.68	0.002
	PPO	0.00	0.14	0.38	0.00	0.25	0.49	0.00	0.37	0.40	0.002
	PPO+BC	0.00	0.25	0.94	0.00	0.40	0.78	0.00	0.32	0.84	0.002
	SAC	0.00	0.01	1.71	0.00	0.16	1.64	0.00	0.24	1.61	0.002
	SAC+Teacher	0.00	0.50	1.21	0.00	0.29	1.24	0.00	0.58	1.24	0.002
	IQL	0.25	0.49	4.85	0.13	0.71	5.01	0.07	0.76	5.03	0.002
	TADPO†	**0.75**	**0.87**	**4.99**	**0.85**	**0.96**	**5.26**	**0.67**	**0.88**	**5.30**	0.002

Table 6.1: Our method (†) compared with baselines, where `sr` denotes Success Rate, `cp` denotes Completion Percentage, `ms` denotes Mean Speed, and `ti` is the Time of Inference for one control step. (Real-time) denotes allotting a limited compute budget for the main control loop necessary for real-time deployment. For RL and IL methods, the model architecture is kept constant to ensure fair comparison.

6.1.2 RL and IL Baselines

We evaluate TADPO against established RL and imitation learning techniques, adapted as necessary for our domain. Here, we discuss their implementation details and analyze their limitations in the context of our problem. When applicable, all policies are supplied with the same teacher policy μ trained in 4.2.1. The model architecture is held constant across all the policies to ensure a fair comparison. The training reward curves for these baselines are presented in Appendix B.

DAgger is used to distill the behavior of the teacher through supervised learning. In long-horizon planning tasks, DAgger suffers from compounding errors. As the policy accumulates errors and deviates from expert trajectories, it encounters previously unseen states, leading to significant performance degradation compared to the teacher policy.

The **PPO** policy is trained analogously to the teacher policy μ described in 4.2.1, but using only sparse waypoints. As explained in 3.1, PPO faces challenges in effective exploration and struggles to differentiate between various terrain types and obstacles, resulting in a suboptimal, overly cautious navigation strategy.

The **PPO + BC** policy aims to distill teacher behavior by adding a KL divergence loss to the PPO loss function as $L^{\text{KL}} = L^{\text{PPO}} - \beta \text{KL}[\pi(a_t|s_t^\pi), \mu(a_t|s_t^\mu)]$, introducing a term that aligns the policy π with the teacher policy μ across all encountered states. While this method provides strong supervision, it encounters challenges similar to DAgger when the student queries the expert from out-of-distribution states. Moreover, the unconstrained updates from the KL divergence term lead to training instability, resulting in convergence to a suboptimal policy.

The **SAC** policy struggles because its entropy maximization leads to excessive exploration of irrelevant states reducing focus on task-specific objectives, making SAC less effective in environments requiring targeted exploration and adaptation across multiple distinct tasks.

The **SAC + Teacher** policy incorporates teacher demonstrations into the SAC framework by pre-populating a portion of the replay buffer. We maintain consistency with TADPO by using an equivalent buffer size and setting the teacher trajectory ratio to $p = 0.5$. However, SAC's performance degrades in multi-task problems [40].

The **IQL** policy is trained using teacher demonstrations to reinforce behavior by learning the Q-values associated with those actions. Although IQL demonstrates some success in navigating steep slopes, its overall performance lags behind TADPO, as it is known to encounter difficulties in multi-task problems, such as off-road autonomy [14].

6.1.3 TADPO

The TADPO policy's success rate (`sr`) and completion percentage (`cp`) notably exceed those of other real-time baseline methods. Furthermore, the policy achieves a comparably high mean speed (`ms`) across all test trajectory sets. These metrics highlight the policy's ability to effectively learn and navigate a wide range of off-road terrains, where rapid adaptive maneuvers and long-horizon planning are crucial for success.

Through ablation studies, we find that setting $\epsilon_\mu = 0.5$ and using a constant $p = 0.5$ yields the best performance for the algorithm, which has been used for comparison with the baselines. Additional ablation studies with different hyperparameter configurations are provided in Appendix A.

6.1.4 Generalizability

Controller	Grassland			Asphalt		
	`sr`	`cp`	`ms`	`sr`	`cp`	`ms`
MPPI + Teacher	0.75	0.90	5.54	1.00	1.00	6.18
TADPO	0.50	0.84	5.70	1.00	1.00	5.99

Table 6.2: Our methods' performance in out-of-distribution terrains, where `sr` denotes Success Rate, `cp` denotes Completion Percentage, and `ms` denotes Mean Speed.

We evaluate the generalization performance of both the teacher and student policies across multiple environments, as summarized in Table 6.2. Generalization tests were not conducted for baseline methods, as they failed to achieve satisfactory performance even in the in-domain (training) environments. Both the teacher and student policies demonstrated strong performance in Grassland and Asphalt terrains, with particularly high `sr`, `cp`, and `ms` metrics in the Asphalt setting.

Qualitative analysis in the Grassland terrain revealed that the MPPI+Teacher policy failed due to a collision with a rock obscured by tall grass—a scenario not encountered in the Desert environment. Similarly, TADPO experienced failures in Grassland, not only due to the same rock collision but also from an additional collision with vegetation. It is important to note that whether vegetation causes damage is determined by the simulation map design; the vehicle itself cannot infer such potential collisions using on-board sensors alone.

6.2 Experiments in Real World

We tested the vehicle on 8 trail following and 3 obstacle avoidance courses. It achieved robust performance over all courses. The vehicle was able to reach an average speed

Figure 6.1: A visualization of SaberCat traversing an obstacle avoidance course.

of 3.5m/s. Figure 6.1 shows the SaberCat vehicle traversing an obstacle avoidance course.

Chapter 7

Conclusion

This thesis investigated hybrid control strategies that combine the strengths of MPC and RL to address the challenges of long-horizon planning and hard exploration in off-road environments. Through a series of experiments, we demonstrated the effectiveness of this approach in enabling real-time, long-range navigation in complex, obstacle-dense, and diverse terrains. By integrating a TADPO-trained policy with a sparse global planner, we developed a complete end-to-end autonomy pipeline capable of operating under challenging conditions.

Looking ahead, we identify two promising research directions for extending this work in off-road autonomy and robotics more broadly. First, as decision horizons increase and control policies must operate with increasingly sparse waypoint guidance, the amount of training data required to learn such behavior may scale unfavorably. Incorporating hierarchical reinforcement learning could significantly reduce this sample complexity by decomposing tasks into manageable sub-goals. Furthermore, recent advances in goal-conditioned reinforcement learning offer additional opportunities to improve data efficiency and generalization in long-horizon settings.

Second, performance-aware learning—where the system compares demonstrations against the current policy—opens the door to enhancing real-world robot performance. Techniques inspired by Real2Sim2Real and lifelong learning can help bridge the gap between heterogeneous demonstration sources and simulated trajectories, enabling continual adaptation and robustness. While aligning diverse demonstrations with simulation-based learning remains an open challenge, the methodology proposed in

this thesis offers a compelling foundation for more adaptive and robust policy learning in real-world scenarios.

Appendix A

Hyperparameter Ablations

We experiment with various hyperparameters in the TADPO implementation, using the best-performing values in our results.

We ablate on the ratio of teacher and student ratio p with values 0.5, 0.7 and 0.3. and ϵ_μ values 0.5, 0.3, 0.7.

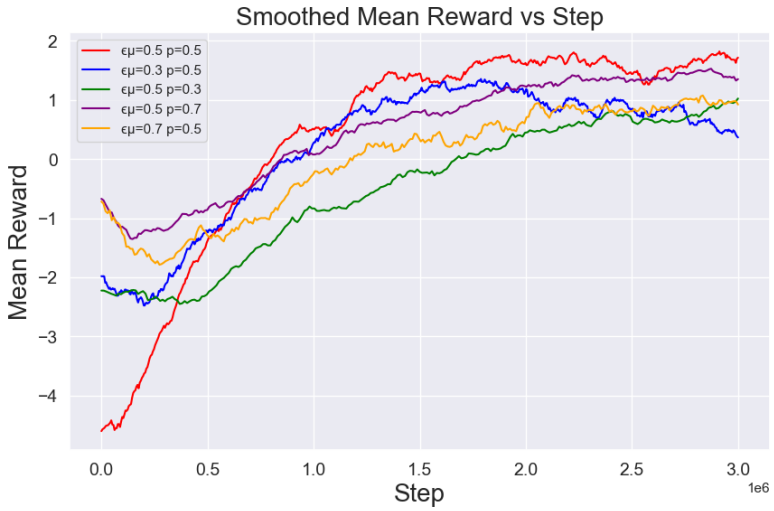

Figure A.1: Smoothed mean reward

We find $p = 0.5$ and $\epsilon_\mu = 0.5$ to be optimal for our task.

Appendix B

RL and Imitation Learning Baseline Training Curves

We experimented with various RL and Imitation Learning baselines. We provide their training curves below.

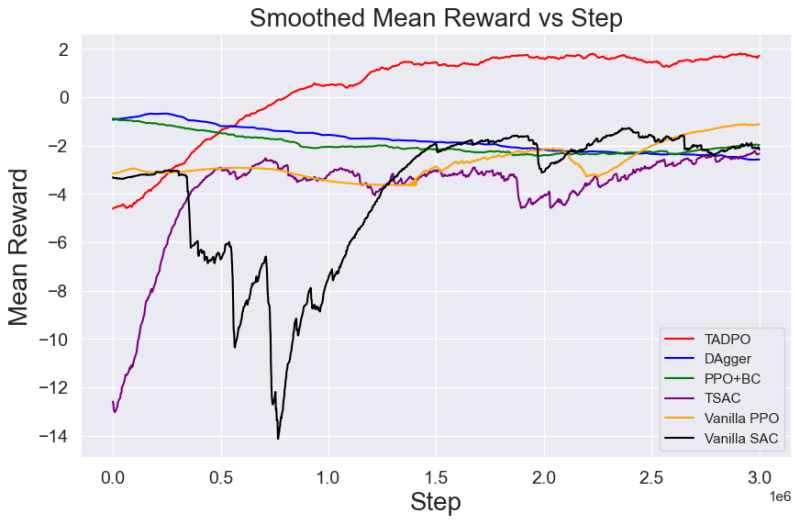

Figure B.1: Smoothed mean reward comparison for baselines

Appendix C

Coarse Global Planner Implementation Details

A coarse global planner is used to provide sparse waypoints to guide the vehicle. A* algorithm is used to generate the waypoints. The algorithm takes a coarse global depth map at 1m resolution. Traversability is determined by the immediate change in altitude between adjacent depth measurements, where if the adjacent depth measures differ by more than 1m, the edge is deemed untraversable. Because of the sparsity of the input data, we expect the output of the A* algorithm to be noisy and expect the underlying controller to compensate for the inaccuracy with onboard sensors.

Appendix D

MPC Baseline Hyperparameters

Planner	N	h	Number of iterations
MPPI	1×10^7	30	1
CEM	1×10^6	30	3
RL+MPPI	1×10^5	20	1
MPPI*	1×10^5	4	1
CEM*	5×10^4	4	3
RL+MPPI*	1×10^5	4	1

Table D.1: Comparison of Common Hyperparameters for Different Methods. * denotes alloting a limited compute budget for real-time deployment.

For training of RL+MPPI, the following hyperparameters are used.

Hyperparameters	Value
Replay Buffer Size	1×10^5
Batch Size	256
Discount Factor (γ)	0.99
Critic Learning Rate	1×10^{-5}
Actor Learning Rate	2×10^{-5}
Learning Rate of α	1×10^{-5}
Model Learning Rate	1×10^{-4}
Expected Entropy ($\mathcal{H}$)	$\mathcal{H} = -4$
Learning Rate for Target	0.005
Maximum Episode Length	1000
Training Iteration Number	1×10^6

Table D.2: RL+MPPI: RL Module Hyperparameters

Appendix E

MPPI Implementation Details

The MPPI Planner for this problem is designed in accordance with [7], employing the same model dynamics and cost function. The key difference lies in the waypoint sampling method: while [7] uses a fixed time-based sampling approach, our approach utilizes a fixed distance-based sampling method, where distance is a hyperparameter. The predicted trajectory rollout given by the bicycle kinematics model [19]. We use this fixed distance-based sampling to provide distance-based waypoints to the PPO policy. The algorithm takes in the final goal position (g) and outputs the waypoints w_i for $i \in 1, .., h$ where h is the planning horizon. It is important to note that MPPI uses a depth map to compute rollover and toppling costs, and an annotation map to calculate segmentation costs. The cost is calculated at time t for all sampled $w_{j,i}$ where $j \in \{1, ..., N\}$ where N is the number of samples.

The cost function is:

1. Goal cost: $u_1 * ||w_{j,i} - g||$
2. Rollover Cost: u_2* Same as [7]
3. Toppling Cost: u_3* Same as [7]
4. Segmentation Cost: $u_4 * sw_j$
5. Smoothness Cost: $u_5 * s_t^2$

and the weights u_i for $i \in \{1, 5\}$ are re-tuned for optimal performance for our vehicle, being 1, 10, 10, 100, 0.8 respectively. Segmentation weights sw_j for $j \in 1, ...5$ are obstacles = 1, rocks = 0.8, dirt = 0, sand = 0.2 and else = 0.

Appendix F

Observation Space

The non-visual inputs to the teacher and student policies are the same except for the observed waypoints.

At a time step t, given the environment state s_t, the vehicle has position $\mathbf{p}_t$, velocity $\mathbf{v}_t$, acceleration $\mathbf{a}_t$, roll θ_t, pitch ϕ_t, and yaw ψ_t in Tait-Bryant format in the world frame $\mathbf{O}$.

For a planning horizon h, waypoint distance d, the MPPI planner generates $\mathbf{w}_{i,t} \in \mathbb{R}$ for $i \in \{1, ..., h\}$, where $d_2(\mathbf{w}_{i,t}, \mathbf{w}_{i+1,t}) = d$ for $i \in \{1, ..., h-1\}$ and $\mathbf{w}_{i,t}$ are in the vehicle frame. We also set $\mathbf{w}_{k,t} = \mathbf{w}_{h,t} \forall k > h$. The vehicle's relative yaw to waypoint $\mathbf{w}_{i,t}$ is $\beta_i = \arctan2(\mathbf{w}_{i,t}^0, \mathbf{w}_{i,t}{}^1)$, where $\mathbf{w}_{i,t}^j$ is the j-th element of $\mathbf{w}_{i,t}$ using 0-based indexing. The global A* planner plans with $d = 80$ from the starting position to goal position using a coarse global map. The local planner plans between waypoints generated by the global planner using detailed local maps.

At time step t, the environment maintains an index i_t which indicates the next waypoint the vehicle should traverse through. If the vehicle is within some switching threshold distance r_{switch}, $i_{t+1} = i_t + 1$; otherwise, $i_{t+1} = i_t$. When $i_{t+1} > h$, we consider the traversal of the planned route successful.

We define the signed waypoint distance for time step t and waypoint at index i as

$$d_t^i = \begin{cases} d_2(\vec{p_t}, \vec{w}_{i,t}) & \text{if } \frac{\pi}{2} \leq \beta_t \leq \frac{3\pi}{2}, \\ -d_2(\vec{p_t}, \vec{w}_{i,t}) & \text{otherwise.} \end{cases} \tag{F.1}$$

The non-visual inputs to the policies then are $O_t = (d_t^{i_t}, d_t^{i_t+1}, \beta_{i_t}, \beta_{i_t+1}, \frac{|\vec{v_t}|}{v_{\max}}, \theta_r, \theta_p)$ where $v_{\max}$ is an arbitrary maximum speed of the vehicle being driven.

All observations made by the teacher and student are stacked with that generated by s_{t-1} and s_{t-2}.

Also, we define $C_t^{\text{td}}(\texttt{rad}, \texttt{res}, \texttt{chan})$ to be a visual observation generated by a top-down camera at time t. The camera field-of-view is maintained so that when viewing flat ground at the level of the vehicle's center of mass, it would be able to observe a square with each side measuring $2 \cdot \texttt{rad}$ m. The camera's resolution is set to $(\texttt{res}, \texttt{res})$. `chan` could be either `rgbd`, in which case a color image is stacked with a depth image, or `depth`, in which case only a depth image is presented. The camera uses a perspective camera model with a z-position of 240 m above the vehicle.

Additionally, we define C_t^{f} to be the observation of a front-facing camera at time t. The camera generates a 64×64 image in `rgbd`. It is positioned $(0.0, -1.5, 2.0)$ m offset from the center of mass of the vehicle. It has a field-of-view of 84°.

F.1 Teacher Policy

The teacher policy's observation at time t is

$$s_t^{\mu} = (O_t^{\mu}, C_t^{\text{td}}(15, 64, \texttt{rgbd}), C_t^{\text{f}})$$

with O_t^{μ} generated by $r_{\text{switch}} = 3, d = 6$.

F.2 Student Policy

The student policy's observation at time t is

$$s_t^{\pi} = (O_t^{\pi}, C_t^{\text{td}}(30, 64, \texttt{rgbd}), C_t^{\text{td}}(90, 64, \texttt{depth}), C_t^{\text{f}})$$

with O_t^{π} generated by $r_{\text{switch}} = 3, d = 80$.

Appendix G

Action Space

The action space used is defined to be (τ_t, s_t) where τ_t is throttle and s_t is the steering at time instance t. s_t ranges from -1 (full right turn) to +1 (full left turn) and τ_t controls the gas pedal, with +1 for full forward acceleration and -1 for full reverse. Gear shifts are managed by the simulator's controller, and brakes are applied when the vehicle's direction opposes τ_t. Otherwise, τ_t controls the engine, moving the vehicle forward for positive values and backward for negative values.

Appendix H

Evaluation Metrics

For an episode, with vehicle position $\mathbf{p}_t$ and speed v_t at time $t \in \{1...T\}$, goal position $\mathbf{p}_g$, acceptance radius r_{accept}, control period τ, the evaluation metrics are given as follows:

$$\texttt{sr} = \begin{cases} 1 & d_2(\mathbf{p}_T, \mathbf{p}_g) < r_{\text{accept}} \\ 0 & \text{otherwise} \end{cases} \tag{H.1}$$

$$\texttt{cp} = 1 - \min_{t \in \{1...T\}} d_2(\mathbf{p}_t, \mathbf{p}_g) \tag{H.2}$$

$$\texttt{ms} = \frac{\sum_{i=1}^{T-1} d_2(\mathbf{p}_t, \mathbf{p}_{t+1})}{\tau \cdot T} \tag{H.3}$$

Appendix I

Hyperparameters

I.1 Teacher Policy (PPO)

Hyperparameters	Value
Learning Rate	3e-4
Discount Factor (γ)	0.99
GAE Parameter (λ)	0.95
Clip Range (ϵ)	0.2
Number of Epochs	10
Mini-batch Size	256
Number of Steps per Update	2048
Value Function Coefficient (λ_v)	0.5
Entropy Coefficient (λ_e)	0.001
MLP Network Architecture	[128,64,64]
CNN Feature Extractor	NatureCNN [26]
CNN Latent Space	256

Table I.1: Hyperparameters for Teacher and Student Training

I.2 Student Policy (TADPO)

The PPO part of the student is trained using the same hyperparameters as in I.1. Hyperparameters for the TADPOupdate are as follows.

Hyperparameters	Value
Update ratio (ϵ_μ)	0.5
Teacher policy ratio (p)	0.5
Learning Rate	3e-4
Discount Factor (γ)	0.99
Clip Range (ϵ)	0.2
Number of Epochs	20
Mini-batch Size	256
Number of Steps per Update	2048
Value Function Coefficient (λ_v)	0.5
Entropy Coefficient (λ_e)	0.001
MLP Network Architecture	[128,64,64]
CNN Feature Extractor	NatureCNN [26]
CNN Latent Space	256
Teacher Demonstration Buffer Size	1e5

Table I.2: Hyperparameters for Teacher and Student Training

Appendix J

Rewards

The reward function is designed to encourage progress towards the desired waypoint at t, while penalizing collisions, excessive jerk, and vehicle damage. Additionally, a success reward is granted upon reaching the final waypoint.

1. Progress: $c_1 * (||\vec{p_{t-1}} - \vec{w_{i,t}}|| - ||\vec{p_t} - \vec{w_{i,t}}||)$
2. Collision: $\begin{cases} c_2 & \text{if dam} > \text{dam}_{\text{thresh}} \\ 0 & \text{otherwise} \end{cases}$
3. Damage: $c_3 * \texttt{dam}$
4. Jerk: $c_4 * (||a_t - a_{t-1}||/dt)$
5. Success: $\begin{cases} c_5 & \text{if } ||\vec{p_{t-1}} - \vec{w_{i,t}}|| < w_{\text{thresh}} \\ 0 & \text{otherwise} \end{cases}$

where c_i for $i \in \{1, \ldots, 5\}$ are scaling factors for the rewards, with values of 1, -2, -1, -0.003, and 1, respectively. The progress reward reflects the distance the vehicle travels toward the goal, with the maximum reward between two waypoints being equal to the distance between them.

These rewards are significantly sparse for exploration in the off-road navigation problem that involve navigating diverse terrains and obstacles.

Appendix K

Simulator

We use BeamNG.tech [3] as the simulator for training and evaluating our algorithms. BeamNG.tech offers a highly realistic simulation environment, featuring advanced vehicle dynamics and damage modeling. BeamNG.tech offers detailed vehicle dynamics and damage modeling, allowing us to test our algorithms in a realistic environment that closely mirrors real-world conditions.

We use `etk800` as our vehicle. The car has dimensions of 2 meters in width, 4.7 meters in length, and 1.4 meters in height. It features a simulated internal combustion engine, an automatic transmission, and an artificially imposed speed limit of $30m/s$.

Appendix L

Algorithm Implementations

We use existing software packages for the implementations of the baselines. We use existing implementation of MPPI by https://github.com/UM-ARM-Lab/pytorch_mppi and CEM by https://github.com/LemonPi/pytorch_cem for our planners. We use the official TD-MPC implementation by [8]. We use the DAgger implementation included in `imitation` by [5]. We also use the official implementation of IQL [20]. For SAC and PPO, we use Stable Baselines 3 (`SB3`) by [32]. We implement our algorithm and other baseline algorithms based on the `SB3` framework. We publish the source code for our method at https://github.com/tadpo-algorithm/tadpo.

Bibliography

[1] Miguel Abreu, Nuno Lau, Armando Sousa, and Luis Paulo Reis. Learning low level skills from scratch for humanoid robot soccer using deep reinforcement learning. In *2019 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, pages 1–8, 2019. doi: 10.1109/ICARSC.2019.8733632. 2.2

[2] Mustafa Baniodeh, Kratarth Goel, Scott Ettinger, Carlos Fuertes, Ari Seff, Tim Shen, Cole Gulino, Chenjie Yang, Ghassen Jerfel, Dokook Choe, Rui Wang, Vinutha Kallem, Sergio Casas, Rami Al-Rfou, Benjamin Sapp, and Dragomir Anguelov. Scaling laws of motion forecasting and planning – a technical report, 2025. URL https://arxiv.org/abs/2506.08228. 1

[3] BeamNG GmbH. BeamNG.tech. URL https://www.beamng.tech/. 5.1.1, K

[4] Abdur R. Fayjie, Sabir Hossain, Doukhi Oualid, and Deok-Jin Lee. Driverless car: Autonomous driving using deep reinforcement learning in urban environment. In *2018 15th International Conference on Ubiquitous Robots (UR)*, pages 896–901, 2018. doi: 10.1109/URAI.2018.8441797. 2.2

[5] Adam Gleave, Mohammad Taufeeque, Juan Rocamonde, Erik Jenner, Steven H. Wang, Sam Toyer, Maximilian Ernestus, Nora Belrose, Scott Emmons, and Stuart Russell. imitation: Clean imitation learning implementations. arXiv:2211.11972v1 [cs.LG], 2022. URL https://arxiv.org/abs/2211.11972. L

[6] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor, 2018. URL https://arxiv.org/abs/1801.01290. 2.2

[7] Tyler Han, Alex Liu, Anqi Li, Alex Spitzer, Guanya Shi, and Byron Boots. Model predictive control for aggressive driving over uneven terrain, 2024. URL https://arxiv.org/abs/2311.12284. 1, 2.1, 3.2, 4.2.1, E, 2, 3

[8] Nicklas Hansen, Xiaolong Wang, and Hao Su. Temporal difference learning for model predictive control, 2022. URL https://arxiv.org/abs/2203.04955. 2.1, L

[9] Crockett Hensley and Matthew Marshall. Off-road navigation with end-to-end

imitation learning for continuously parameterized control. In *SoutheastCon 2022*, pages 591–597, 2022. doi: 10.1109/SoutheastCon48659.2022.9763997. 1, 2.2

[10] Todd Hester, Matej Vecerik, Olivier Pietquin, Marc Lanctot, Tom Schaul, Bilal Piot, Dan Horgan, John Quan, Andrew Sendonaris, Gabriel Dulac-Arnold, Ian Osband, John Agapiou, Joel Z. Leibo, and Audrunas Gruslys. Deep q-learning from demonstrations, 2017. URL https://arxiv.org/abs/1704.03732. 2.3

[11] Mokter Hossain. Autonomous delivery robots: A literature review. *IEEE Engineering Management Review*, 51(4):77–89, 2023. doi: 10.1109/EMR.2023.3304848. 1

[12] David Isele, Reza Rahimi, Akansel Cosgun, Kaushik Subramanian, and Kikuo Fujimura. Navigating occluded intersections with autonomous vehicles using deep reinforcement learning, 2018. URL https://arxiv.org/abs/1705.01196. 2.2

[13] Suresh S. Muknahallipatna James W. Mock. A comparison of ppo, td3 and sac reinforcement algorithms for quadruped walking gait generation, 2023. URL 10.4236/jilsa.2023.151003. 2.3

[14] Michael Janner, Yilun Du, Joshua B. Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis, 2022. URL https://arxiv.org/abs/2205.09991. 6.1.2

[15] Dvij Kalaria, Shreya Sharma, Sarthak Bhagat, Haoru Xue, and John M. Dolan. Wroom: An autonomous driving approach for off-road navigation, 2024. URL https://arxiv.org/abs/2404.08855. 1, 2.2

[16] Bingyi Kang, Zequn Jie, and Jiashi Feng. Policy optimization with demonstrations. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 2469–2478. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/kang18a.html. 2.3

[17] Alex Kendall, Jeffrey Hawke, David Janz, Przemyslaw Mazur, Daniele Reda, John-Mark Allen, Vinh-Dieu Lam, Alex Bewley, and Amar Shah. Learning to drive in a day, 2018. URL https://arxiv.org/abs/1807.00412. 2.2

[18] Marin Kobilarov. Cross-entropy motion planning. *The International Journal of Robotics Research*, 31(7):855–871, 2012. doi: 10.1177/0278364912444543. URL https://doi.org/10.1177/0278364912444543. 2.1, 3.2

[19] Jason Kong, Mark Pfeiffer, Georg Schildbach, and Francesco Borrelli. Kinematic and dynamic vehicle models for autonomous driving control design. In *2015 IEEE Intelligent Vehicles Symposium (IV)*, pages 1094–1099, 2015. doi: 10.1109/IVS.2015.7225830. E

[20] Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning, 2021. URL https://arxiv.org/abs/2110.06169. 2.3, L

[21] Hojin Lee, Junsung Kwon, and Cheolhyeon Kwon. Learning-based uncertainty-aware navigation in 3d off-road terrains, 2022. URL https://arxiv.org/abs/2209.09177. 3.2

[22] Hojin Lee, Taekyung Kim, Jungwi Mun, and Wonsuk Lee. Learning terrain-aware kinodynamic model for autonomous off-road rally driving with model predictive path integral control. *IEEE Robotics and Automation Letters*, 8(11): 7663–7670, November 2023. ISSN 2377-3774. doi: 10.1109/lra.2023.3318190. URL http://dx.doi.org/10.1109/LRA.2023.3318190. 2.1, 3.2

[23] Gabriele Libardi and Gianni De Fabritiis. Guided exploration with proximal policy optimization using a single demonstration, 2021. URL https://arxiv.org/abs/2007.03328. 2.3, 3.1

[24] Jesus Bujalance Martin, Raphael Chekroun, and Fabien Moutarde. Learning from demonstrations with sacr2: Soft actor-critic with reward relabeling, 2021. URL https://arxiv.org/abs/2110.14464. 2.3

[25] Michal Minarik, Robert Penicka, Vojtech Vonasek, and Martin Saska. Model predictive path integral control for agile unmanned aerial vehicles, 2024. URL https://arxiv.org/abs/2407.09812. 3.2

[26] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning, 2013. URL https://arxiv.org/abs/1312.5602. ??, ??

[27] OpenAI, Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafal Jozefowicz, Bob McGrew, Jakub Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, Jonas Schneider, Szymon Sidor, Josh Tobin, Peter Welinder, Lilian Weng, and Wojciech Zaremba. Learning dexterous in-hand manipulation, 2019. URL https://arxiv.org/abs/1808.00177. 2.2

[28] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision, 2024. URL https://arxiv.org/abs/2304.07193. 4.3

[29] Zhenghao Peng, Quanyi Li, Chunxiao Liu, and Bolei Zhou. Safe driving via expert guided policy optimization. In Aleksandra Faust, David Hsu, and Gerhard

Neumann, editors, *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pages 1554–1563. PMLR, 08–11 Nov 2022. URL https://proceedings.mlr.press/v164/peng22a.html. 2.3

[30] Chenyang Qi, Chengfu Wu, Lei Lei, Xiaolu Li, and Peiyan Cong. Uav path planning based on the improved ppo algorithm. In *2022 Asia Conference on Advanced Robotics, Automation, and Control Engineering (ARACE)*, pages 193–199, 2022. doi: 10.1109/ARACE56528.2022.00040. 2.2

[31] Yue Qu, Hongqing Chu, Shuhua Gao, Jun Guan, Haoqi Yan, Liming Xiao, Shengbo Eben Li, and Jingliang Duan. Rl-driven mppi: Accelerating online control laws calculation with offline policy. *IEEE Transactions on Intelligent Vehicles*, 9(2):3605–3616, 2024. doi: 10.1109/TIV.2023.3348134. 2.1, 3.2

[32] Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22(268):1–8, 2021. URL http://jmlr.org/papers/v22/20-1364.html. L

[33] Stephane Ross, Geoffrey J. Gordon, and J. Andrew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning, 2011. URL https://arxiv.org/abs/1011.0686. 2.3

[34] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL https://arxiv.org/abs/1707.06347. 2.2, 3.1

[35] David Silver, James Bagnell, and Anthony Stentz. High performance outdoor navigation from overhead data using imitation learning. 06 2008. doi: 10.15607/RSS.2008.IV.034. 2.1

[36] Hamid Taheri, Seyed Rasoul Hosseini, and Mohammad Ali Nekoui. Deep reinforcement learning with enhanced ppo for safe mobile robot navigation, 2024. URL https://arxiv.org/abs/2405.16266. 2.2

[37] Qifan Tan, Cheng Qiu, Jing Huang, Yue Yin, Xinyu Zhang, and Huaping Liu. Path tracking control strategy for off-road 4ws4wd vehicle based on robust model predictive control. *Robotics and Autonomous Systems*, 158:104267, 2022. ISSN 0921-8890. doi: https://doi.org/10.1016/j.robot.2022.104267. URL https://www.sciencedirect.com/science/article/pii/S0921889022001567. 1

[38] Yiquan Wang, Jingguo Wang, Yu Yang, Zhaodong Li, and Xijun Zhao. An end-to-end deep reinforcement learning model based on proximal policy optimization algorithm for autonomous driving of off-road vehicle. In Wenxing Fu, Mancang Gu, and Yifeng Niu, editors, *Proceedings of 2022 International Conference on Autonomous Unmanned Systems (ICAUS 2022)*, pages 2692–2704, Singapore,

2023. Springer Nature Singapore. ISBN 978-981-99-0479-2. 1, 2.2

[39] Grady Williams, Andrew Aldrich, and Evangelos Theodorou. Model predictive path integral control using covariance variable importance sampling, 2015. URL https://arxiv.org/abs/1509.01149. 2.1, 3.2

[40] Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, Chelsea Finn, Stanford University, Uc Berkeley, and Robotics at Google. Multi-task reinforcement learning without interference. 2019. URL https://api.semanticscholar.org/CorpusID:233305688. 6.1.2