CS 378 Geometric Foundations
Final Project – Full Report

Jacob Villanueva
Department of Computer Science
University of Texas at Austin

Arnav Bagad
Department of Computer Science
University of Texas at Austin

*M*ay 3, 2025

Abstract

In this paper, we study how to improve decision making for autonomous driving when encountering uncertainty due to sensor noise and unpredictable behavior of the surrounding vehicles. We used the HighwayEnv simulator to evaluate a baseline control method and implement a set of extensions grounded in reinforcement learning. Our contributions include a noise aware PPO agent that integrates confidence scores, a rule based intention augmentation derived from vehicle lateral velocity, and a hybrid controller that overrides unsafe PPO actions using rule based logic. We also explore skill abstraction using parameterized high level actions to increase sample efficiency. The experiments show that the enhancements we implemented can significantly improve robustness, safety, crash rates, and produce smoother driving trajectories under noisy and ambiguous settings.

1 Introduction

As real time perception and control becomes increasingly more important for autonomous driving, systems need to have the ability to make safe decisions under times of uncertainty, however this is challenging. In a real driving environment, an agent has to account for many factors outside of the noisy sensory input it receives, this includes irrational or erratic behavior of other vehicles. Through these sources of uncertainty, decision policies can become unsafe and not robust if they aren't accounting for the information properly.

In this paper we investigate improvements on current robust autonomous driving decision making utilizing noisy estimations for reinforcement learning through a Markov Decision Process (RL-MDP) framework. Looking at prior work in robust planning and safe policy optimization, we extend a baseline method with several improvements that specifically address two types of uncertainty. The first being the unpredictability of surrounding vehicles and the second being noise in the tracking of velocity and position of other vehicles.

We took a look at four potential improvements. The first is we train a noise injected PPO agent that incorporates uncertainty into its state representation. Secondly we use a rule based predictor for surrounding vehicles intention that is derived from the vehicle's lateral velocity. The third idea we implemented was a hybrid control module that prevents unsafe actions via a rule based safety override system. Lastly we experiment with skill based abstraction to structure policy learning at a higher semantic level.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

2 Related Work

2.1 Handling Behavioral Uncertainty (Scenario 1)

When working with autonomous agents, we have to deal with the unoptimal driving behavior of surrounding drivers. We looked at several works that have proposed several strategies in modifying the observation space with behavioral or intent labels. Yildirim et al. [3] introduced a prediction augmented DQN architecture that demonstrates how including inferred lane change intentions improved decision making in traffic filled situations. They specifically use labels for lane change behavior with lane keep, right lane change, and left lane change. They also use a regression model to predict the time till lane change for other vehicles. This paper inspired us to take a look into implementing a rule based intention wrapper that helps us predict if we think the cars around us will change lanes at the current moment.

Another strategy we saw was to abstract the low level actions and convert them into high level actions, essentially creating macros for the car to follow. Wang et al. [13] suggested this in their paper where they incorporate high level behaviors such as overtaking and lane following. By doing this the decisions we make can be done at a higher level which can improve results when handling uncertainty in car behavior. To account for generalization in decision making we found literature that discusses hybrid controls for applying safety rules. Kimura et al.[11] explores this specifically where they applied deterministic safety rules when their policy decided on riskier actions. They specifically look at using a model predictive control (MPC) which focuses on avoiding collision, providing bounds for the car, and penalizing heavily for risky decisions. Their methodology influenced how we approached our own override system.

2.2 Robustness to Sensor Noise (Scenario 2)

To ensure there is robustness in our policy we cannot assume that tracking of other vehicles will be perfect which in a simulator it would be. Leurent et al. [1] present two robust planning algorithms that estimate the worst case outcomes by modeling uncertainty in system dynamics. Although we don't follow their entire methodology we explore robustness through noisy observations and learned policies. We found that research has been focusing on improving generalization of situations to avoid overfitting and assuming perfect data. Injecting a type of noise to our observed states will provide a simulation to sensors having noise in their data readings. This will enable an agent to be more reliable in the real world and act more cautious.

Finally, all our experiments are conducted in the `highway-env` simulator [16]. Its flexible configuration allows us to test robustness across high-density traffic, variable reward shaping, and noisy observations.

3 Method

3.1 Environment and Baseline PPO

We use the highway-env simulator with two scenarios: highway-fast-v0 and roundabout-v0. Each episode runs for a fixed duration (e.g. 40 s on the highway, 60 s in the roundabout) and agents choose among five discrete meta-actions: {Left, Right, Accelerate, Decelerate, Idle}.

Our base learner is a Proximal Policy Optimization (PPO). We train an end-to-end MLP policy with two hidden layers of 256 units each. The key hyperparameters are:

- learning rate: 5×10^{-4}
- discount factor: $\gamma = 0.99$
- minibatch size: 64
- steps per update: 1024

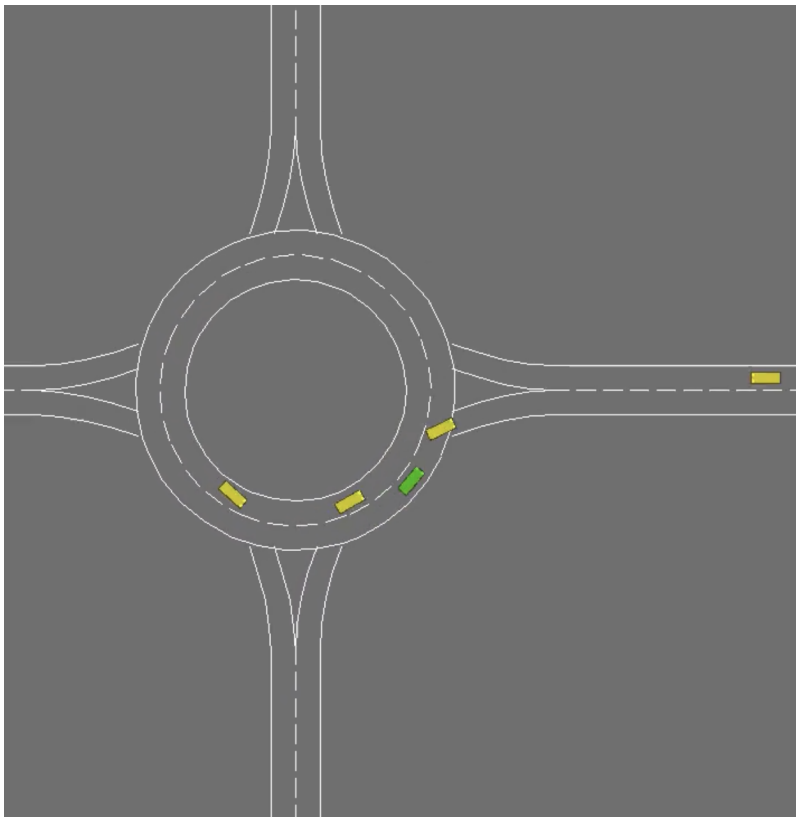

Figure 1: Highway env: Roundabout v_0

- total timesteps: 10,000

PPO is a model-free reinforcement learning algorithm. Its goal is to maximize the expected reward without having drastic changes in the policy at every step. At each step it computes how much better the chosen actions were compared to the old policy (the "advantage" $\hat{A}_t$), and then adjusts the policy to improve those advantages, but only up to a small threshold. PPO does this through the clipped objective:

$$L^{\mathrm{CLIP}}(\theta) = \mathbb{E}_t\big[\min\big(r_t(\theta)\,\hat{A}_t,\ \mathrm{clip}\big(r_t(\theta),\,1-\epsilon,\,1+\epsilon\big)\,\hat{A}_t\big)\big],$$

where

$$r_t(\theta) = \frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\mathrm{old}}}(a_t \mid s_t)}$$

is the change in probability of action a_t, and ϵ (often 0.2) is the maximum allowed shift. This clipping makes training more stable by preventing very large policy updates at once.

3.2 Provided Dynamics Model

Based on the provided code we model the system dynamics using a learned discrete-time approximation of the form

$$x_{t+1} = x_t + \Delta t \cdot \left(A(x_t, u_t)\, x_t + B(x_t, u_t)\, u_t\right),$$

where $x_t \in \mathbb{R}^n$ is the current state, $u_t \in \mathbb{R}^m$ is the current action, and Δt is the time step. The matrices $A(x_t, u_t) \in \mathbb{R}^{n\times n}$ and $B(x_t, u_t) \in \mathbb{R}^{n\times m}$ are generated by two neural networks that take the concatenated pair $[x_t; u_t]$ as input. This dynamics model is utilized with a Cross-Entropy Method planner to predict its actions.

3.3 Noise Injection and Confidence

To simulate sensor uncertainty, we add Gaussian noise to the observed positions and velocities. Specifically, after each environment step we affect the kinematic matrix (columns for x, y, v_x, v_y) with $\mathcal{N}(0, \sigma^2)$ noise. In our case we use $\sigma = 0.2$. We then compute a per-vehicle *confidence* score

$$c_i = \exp\big(-\mathrm{Var}(x, y, v_x, v_y)\big)\,,$$

and append it as an extra feature in the observation vector. This lets the policy learn to trust or distrust noisy inputs.

3.4 Intent Prediction

Inspired by [3], we decided to infer a simple lane change intent for each vehicle from its lateral speed:

$$\text{intent} = \begin{cases} +1 & v_y > 0.3 \\ -1 & v_y < -0.3 \\ 0 & \text{otherwise} \end{cases}$$

This intent bit is concatenated to each vehicle's feature vector, giving the policy early warning of nearby cars' planned maneuvers given by this rule based predictor. We hope this will provide safer handling around other vehicles and decrease the car crash rate and increase our rewards.

3.5 Safety Override

To be a defensive driver against risky maneuvers, we place a lightweight safety filter around our trained PPO policy, inspired by [11]. Let a_t be the action proposed by PPO at time t, and let $d_i(t)$ be the longitudinal distance between the ego vehicle and vehicle i. We define a safety buffer D which enforces

$$\text{if } a_t \in \{\text{LaneLeft, LaneRight}\} \quad \text{and} \quad \exists\, i:\ d_i(t) < D, \quad \text{then override } a_t \leftarrow a_{\text{safe}},$$

where a_{safe} is a conservative fallback action (e.g., remain in lane or idle).

By intervening only when $d_i(t)$ falls below the threshold D, this override preserves most of the learned policy's behavior while preventing dangerously close overtakes. We apply this filter only during evaluation, so the PPO agent trains without artificial constraints but is always held to a safe standard when deployed.

3.6 Skill Abstraction

Rather than acting on low-level primitives at every time step, we also explore a higher-level decision layer in which the agent selects from a small, interpretable set of macros. Let

$$S = \{0, 1, 2, 3, 4\}$$

be the set of macros, where each macro $s \in S$ corresponds to a fixed sequence of primitive actions

$$A_s = \big[a_{s,1},\, a_{s,2},\, \ldots,\, a_{s,K}\big].$$

For example:

$$A_{\text{OVERTAKE\_LEFT}} = [\text{Left, Accelerate, Accelerate, Right}], \quad A_{\text{FOLLOW}} = [\text{Idle, Idle, } \ldots].$$

During training, PPO learns a policy $\pi(s \mid o_t)$ over these skills rather than raw actions. At each decision point, the agent samples $s_t \sim \pi(\cdot \mid o_t)$ and then executes the entire sequence A_{s_t} over the next K steps.

To ensure we are not overgeneralizing and falling into risky traps if at any step within A_s the proximity to another vehicle falls below our buffer D, our safe policy interrupts the remaining primitive actions in A_s and substitutes a safer maneuver (e.g., Idle or maintain lane). By combining these two methods we can hope to have our vehicle take safer options while having preplanned moves through macros.

3.7 Training and Evaluation Pipeline

Our workflow has two clear stages. First, during training we launch multiple (e.g. four) parallel copies of the environment in a runner, train PPO for a fixed budget of timesteps, and stream training metrics to TensorBoard. Second, during evaluation we use a single environment wrapped in both a video recorder and an episode statistics logger. Here, every policy call passes through our safety or skill wrappers, and we run a small set of episodes (e.g. ten) per variant. For each episode we log total return, crashes, lane-change counts, and the agent's average confidence in its noisy observations. This separation lets us iterate quickly on policy improvements while ensuring that all final results are collected in a consistent, reproducible setup.

All of these components listed above are implemented in Python using Gymnasium, Stable-Baselines3, and our custom wrapper classes. The complete code and runnable notebooks are available on our GitHub repository for full transparency and ease of replication.

4 Experiments / Results

We trained robust autonomous driving agents using an optimized Proximal Policy Optimization (PPO) algorithm with modular wrappers designed for noise injection, intention modeling, skill abstraction, and safety overrides. The training pipeline was built on top of the HighwayEnv simulation suite and run on two environments: highway-fast-v0 and roundabout-v0. For each configuration, we trained agents for 10,000 timesteps and evaluated them across 10 episodes, logging metrics including average reward, crash rate, and average confidence. The modular wrappers allowed us to isolate the contribution of each component—for instance, adding noise to simulate sensor imperfections, or enforcing rule-based control when confidence was low. The results (Tables 1 & 2) show that safety and skill abstraction helped reduce crash rates significantly under noise, while preserving or recovering reward in most settings. Confidence scores provided additional insight into how uncertain the model was in each configuration.

Table 1: Performance of PPO variants on `highway-fast-v0` (10000 timesteps, 10 eval episodes).

Variant	Avg Reward	Crash Rate	Avg Confidence
1. PPO Baseline	21.54 $\pm$0.72	0.00	N/A
2. Noise	16.97 $\pm$7.68	0.30	0.736
3. Noise + Intent	20.79 $\pm$2.79	0.20	0.04
4. Noise + Safety	8.78 $\pm$6.39	0.90	0.737
5. Noise + Safety + Skills	21.28 $\pm$0.90	0.00	0.731

Table 2: Performance of PPO variants on `roundabout-v0` (10000 timesteps, 10 eval episodes).

Variant	Avg Reward	Crash Rate	Avg Confidence
1. Provided Baseline	9.17	0.00	N/A
2. PPO Baseline	16.88 $\pm$9.78	0.30	N/A
3. Noise	11.43 $\pm$9.79	0.40	0.806
4. Noise + Intent	20.46 $\pm$7.00	0.10	-0.617
5. Noise + Safety	7.28 $\pm$9.14	0.50	0.816
6. Noise + Safety + Skills	8.45 $\pm$9.53	0.50	0.007

5 Discussion

Throughout our experiments, we went through extensive trial and error with reward functions, environment configurations, and PPO hyperparameters in search of a robust, generalizable driving

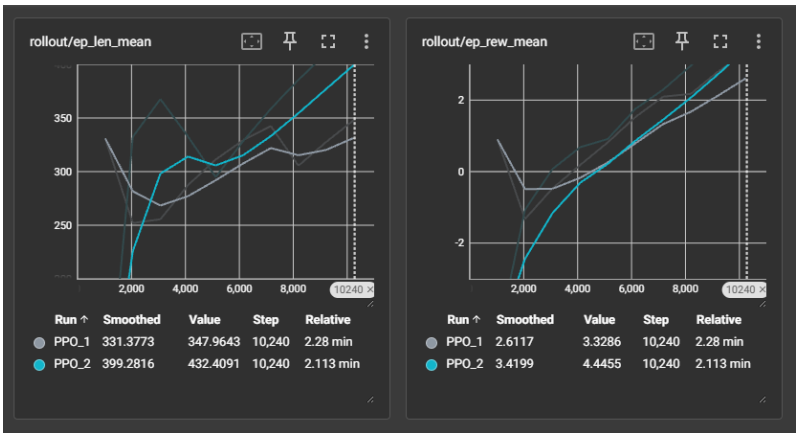

Figure 2: Roundabout v_0 : $Noise + Intent$

policy.

We experimented with tuning learning rates, batch sizes, discount factors, and network architectures, and found that while PPO could learn effectively in clean settings, its performance became unstable when noise or complex interactions (like merging in roundabouts) were introduced. Modifying the reward function to better emphasize safety and lane discipline led to modest gains, but required careful balancing to avoid unintended incentives (e.g., overly conservative driving). Similarly, adjusting environment configuration files, such as increasing traffic density or making vehicles more aggressive, exposed edge cases that our models initially failed to handle. Although skill abstraction and safety override reduced crash rates under noisy conditions, they sometimes suppressed reward by overcorrecting.

These results suggest our modular design has potential, but more tuning and policy-architecture refinement is needed. In particular, we believe improved confidence estimation, better skill sequencing, and hybrid rule-learning approaches could push performance further for these categories as we saw with intent.

6 Conclusion

In this project, we explored how to enhance autonomous driving robustness through modular policy design in noisy and uncertain environments. By integrating confidence-aware noise handling, intention augmentation, safety overrides, and skill-based abstraction into a PPO framework, we demonstrated that safety and stability can be improved without sacrificing reward in some cases.

Our results across two distinct driving environments showed that these components, especially when combined, meaningfully reduce crash rates and enable smoother driving. This displays our progress towards improving the two scenarios depicted in the project. However, our experiments also revealed the sensitivity of performance to reward shaping, hyperparameter tuning, and environment configuration. Future work should explore more adaptive confidence estimation, dynamic skill switching, and policy refinement techniques to further close the gap between safe and high performing autonomous control.

7 Github

Github Link: https://github.com/ja-pavi/CS378-Final-PPO/tree/main

References

[1] Edouard Leurent, Yann Blanco, Denis Efimov, and Odalric-Ambrym Maillard. Approximate robust control of uncertain dynamical systems. arXiv preprint arXiv:1903.00220, 2019. 7

[2] Llorca, D. F., Salinas, C., Jiménez, M. et al.: 'Two-camera based accurate vehicle speed measurement using average speed at a fixed point', Proc. of IEEE Intell. Transp. Sys. Conf. (ITSC), 2016, pp. 2533–2538

[3] M. Yildirim, S. Mozaffari, L. McCutcheon, M. Dianati, A. Tamaddoni-Nezhad and S. Fallah, "Prediction Based Decision Making for Autonomous Highway Driving," 2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC), Macau, China, 2022, pp. 138-145, doi: 10.1109/ITSC55140.2022.9922398.

[4] Ginzburg, C., Raphael, A., Weinshall, D.: 'A Cheap System for Vehicle Speed Detection', arXiv:1501.06751, 2015

[5] Liu, Y., Lian, Z., Ding, J. et al.: 'Multiple Objects Tracking Based Vehicle Speed Analysis with Gaussian Filter from Drone Video', Intell. Sci. and Big Data Eng. Vis. Data Eng. (IScIDE), 2019, pp. 362–373

[6] Maduro, C., Batista, K., Peixoto, P. et al.: 'Estimation of vehicle velocity and traffic intensity using rectified images'. Proc. Int. Conf. on Image Processing, 2008

[7] Lee, J., Roh, S., Shin, J. et al.: 'Image-Based Learning to Measure the Space Mean Speed on a Stretch of Road without the Need to Tag Images with Labels', Sensors, 2019, 19, 1227

[8] Alefs, B., Schreiber, D.: 'Accurate Speed Measurement from Vehicle Trajectories using AdaBoost Detection and Robust Template Tracking'. Proc. IEEE Int. Transp. Syst. Conf. (ITSC)), 2007

[9] Bouziady, A. E., Thami, R. O. H., Ghogho, M. et al.: 'Vehicle speed estimation using extracted SURF features from stereo images', Proc. IEEE Int. Conf. Intell. Sys. and Comp. Vis. (ISCV), 2018

[10] K. Zheng, H. Yang, S. Liu, K. Zhang and L. Lei, "A Behavior Decision Method Based on Reinforcement Learning for Autonomous Driving," in IEEE Internet of Things Journal, vol. 9, no. 24, pp. 25386-25394, 15 Dec.15, 2022, doi: 10.1109/JIOT.2022.3196639.

[11] Hikaru Kimura, Masaki Takahashi, Kazuhiro Nishiwaki, Masahiro Iezawa, Decision-Making Based on Reinforcement Learning and Model Predictive Control Considering Space Generation for Highway On-Ramp Merging, IFAC-PapersOnLine, Volume 55, Issue 27, 2022, Pages 241-246, ISSN 2405-8963, https://doi.org/10.1016/j.ifacol.2022.10.519.

[12] J. Zhao, Y. Zhao, W. Li and C. Zeng, "End-to-End Autonomous Driving Algorithm Based on PPO and Its Implementation," 2024 IEEE 13th Data Driven Control and Learning Systems Conference (DDCLS), Kaifeng, China, 2024, pp. 1852-1858, doi: 10.1109/DDCLS61622.2024.10606596.

[13] Letian Wang, Jie Liu, Hao Shao, Wenshuo Wang, Ruobing Chen, Yu Liu, Steven L. Waslander (2023). Efficient Reinforcement Learning for Autonomous Driving with Parameterized Skills and Priors. In Robotics: Science and Systems.

[14] Yazid, Imam Rachmawati, Ema. (2023). Autonomous driving system using proximal policy optimization in deep reinforcement learning. IAES International Journal of Artificial Intelligence (IJ-AI). 12. 422. 10.11591/ijai.v12.i1.pp422-431.

[15] Tammewar, A., Chaudhari, N., Saini, B., Venkatesh, D., Dharahas, G., Vora, D., Patil, S., Kotecha, K., Alfarhood, S. (2023). Improving the Performance of Autonomous Driving through Deep Reinforcement Learning. Sustainability, 15(18), 13799. https://doi.org/10.3390/su151813799

[16] authors: - family-names: "Leurent" given-names: "Edouard" title: "An Environment for Autonomous Driving Decision-Making" version: 1.4 date-released: 2018-05-01 url: "https://github.com/eleurent/highway-env"