Autonomous Driving Via Context-aware Multi-sensor Perception and Enhanced Inverse Reinforcement Learning

Yu Shen$^1$, Weizi Li$^2$, and Ming C. Lin$^1$

https://gamma.umd.edu/researchdirections/autonomousdriving/eirl/

Abstract—Despite significant advancements, collision-free navigation in autonomous driving is still challenging. On one hand, the perception module needs to interpret multimodal unstructured data and produce structured data of the environment. On the other hand, the navigation module needs to balance the use of machine learning and motion planning in order to achieve efficient and effective control of the vehicle. We propose a novel framework combining context-aware multi-sensor perception and enhanced inverse reinforcement learning (EIRL) for autonomous driving. Our perception module not only achieves the highest "mean Average Precision" (mAP) scores across all test cases in the KITTI dataset, but also suppresses up to 15% false positives, compared to other recent methods. The EIRL module implements several attributes including non-uniform prior for features, model parameter reuse for continuous training, and learning from accidents. These attributes help reduce the number of collisions of the vehicle up to 41%, increase the training efficiency by 2.5x, and obtain higher test scores up to two orders of magnitude. Overall, our method can enable the vehicle to drive 10x further than other methods, while achieving collision avoidance over both static and dynamic obstacles.

I. INTRODUCTION

Autonomous vehicles (AVs) have the potential to contribute to a more efficient and safer transportation system by alleviating traffic congestion and reducing the number of accidents. In general, autonomous driving can be realized via either an end-to-end approach or a mediated-perception approach [1]. An end-to-end approach meaning that raw sensor data are directly mapped to control commands (e.g., steering angles), which usually results in a succinct training pipeline at the cost of model interpretability. A mediated-perception approach, on the other hand, decouples perception and navigation, thus offering better model interpretability and enhanced driving safety. However, a mediated-perception approach commonly adopts a planning algorithm for navigating an AV—a process that can be computationally expensive because of the requirement of holistic environment information for achieving global optimality and planning in high-dimensional state space.

We propose a novel mediated-perception approach for autonomous driving. Our approach consists of context-aware multi-sensor perception and enhanced inverse reinforcement learning (EIRL). The perception module interprets unstructured information (i.e., images and point clouds) of an environment using multiple sensors with different viewpoints, extract context-aware information, and produces structured information, such as the size and position of an object in the environment. EIRL then takes the structured information along with expert trajectories as input to learn a control policy for autonomous driving. The pipeline of our approach is illustrated in Fig. [1]

Fig. 1. System pipeline. At each time step, the vehicle/simulator generates unstructured data such as images and point clouds. These data are processed by the perception module to produce structured data, which are then used by the EIRL module to learn a control policy for autonomous driving.

Our approach has several advantages. First of all, as a mediated-perception approach, instead of using a planning algorithm for navigating an AV at all time steps, we only use a planning algorithm to generate expert demonstrations for EIRL to imitate. Once learned, the control policy can operate in real time with a small number of features. This design choice greatly reduces the computational overhead of using a planning algorithm. Second, EIRL complements the original IRL [2] with the flexibility to impose a non-uniform prior on important features for a specific problem, for example, collision for autonomous driving. Third, since our approach is based on reinforcement learning, compared to using supervised learning for imitation, EIRL is more robust in rare situations and can generalize better in new environments [2]. Lastly, similar to ADAPS [3], alternative safe trajectories are generated during the analysis of an accident. This approach enables the learning algorithm to learn from accidents, which is crucial as collecting accident data from the real world is impractical. In summary, our contributions are as follows:

- A novel autonomous driving training pipeline, i.e., context-aware multi-view, multi-sensor perception and enhanced inverse reinforcement learning, where the extracted features from one is used to guide the sampling prior of the other (Sec. III-A).
- An improved multi-view, multi-sensor 3D perception
algorithm that can suppress up to 15% false positives compared to AVOD [4] by utilizing context-aware semantic information (Sec. III-B).

- An enhanced version of IRL, i.e., EIRL, which imposes a non-uniform prior on extracted features, reuses learned parameters for continuous training, and learn from accidents to achieve up to 10x further safe driving (Sec. III-C);

- A modular autonomous driving training platform, containing virtual sensors and scenes, data collectors, car controllers, and various algorithms for end-to-end learning and perception-based planning (Sec. III-D).

The effectiveness and efficiency of our approach are demonstrated in a variety of experiments. Overall, our method can enable the AV to drive safely 10x further than the other methods. More specifically, our perception module not only outperforms other state-of-the-art methods in “mean Average Precision” (mAP) scores over all cases of the KITTI dataset [5], but also suppresses up to 15% false positives compared to AVOD [4]. The attribute non-uniform prior can assist in reducing the number of collisions of the AV up to 41%; the use of learned model parameters for continuous training can result in 2.5x faster training; and learning from accidents can help achieve higher test scores up to two orders of magnitude. In addition to quantitative results, we have qualitatively demonstrated that our method can steer the AV to avoid both static and dynamic obstacles, even in the presence of a narrow passage (see supplementary video).

II. RELATED WORK

A. Autonomous Driving

Various methods have been proposed to solve the perception, planning, and control problems for autonomous driving [6], [7]. Examples of the end-to-end approach include end-to-end reinforcement learning [8] and end-to-end imitation learning [3], [9], [10], [11]. These approaches usually require a large amount of training data in order to be robust in rare cases, such as pre-accident scenarios. In addition, the use of deep neural networks in these approaches to directly map raw sensor data to control commands can lead to low model interpretability.

Examples of the mediated-perception approach include perception plus motion planning [12] and perception plus learning-based planning [9]. Because of the decomposition of perception and navigation, these approaches enjoy better model interpretability, hence improved driving safety. Recently, Li et al. propose ADAPS [3], an end-to-end imitation learning framework that enables an AV to learn from accidents. Compared to ADAPS, our work proposes a new architecture that can not only fuse data from different modalities but also generalize better, as it relies on reinforcement learning rather than using supervised learning to imitate the expert’s behaviors.

B. 3D Perception

There are multiple classes of 3D perception algorithms, e.g., 3D perception from monocular images, point clouds, or multi-source data. To provide a few examples, 3DVP [13] takes RGB images as input, aligns them with 3D CAD models, and then trains a model to generate 3D voxel patterns for 2D detection and 3D pose estimation. PointsNet [14] uses point clouds as input for 3D object classification and semantic segmentation. AVOD [4] uses a region proposal network (RPN) to learn image features for efficient generation of high-quality proposals [15], [16]. Our method, compared to AVOD [4], can fuse multiple RGB images and point clouds to detect 3D bounding boxes of vehicles via the extracted information such as segmentation map and depth map, thus providing more accurate and fine-grained detection.

C. Inverse Reinforcement Learning (IRL)

As an effective technique for imitation learning, IRL involves two steps: 1) learning a reward function from experts’ demonstrations and 2) using the acquired reward function for reinforcement learning to learn a control policy [2]. To provide some examples, Sharifzadeh et al. [17] apply Deep Q-Networks to extract a reward function in large state space. You et al. [18] use deep neural networks to approximate the latent reward function of the expert and then apply deep Q-learning to obtain the control policy.

Compared to the original IRL [2], our approach allows the use of a non-uniform prior, instead of a uniform prior, on the features. This can prevent important features from receiving trivial weights during the learning process. In addition, our approach uses the parameters of a pre-trained policy as the start point for continuous training, which can improve training efficiency while maintaining the model performance.

III. APPROACH

A. Framework Overview

Our architecture combines context-aware multi-sensor perception and enhanced inverse reinforcement learning (EIRL). The perception module takes multiple sensors’ data as input and produces structured data, which are then used by the EIRL module to learn a control policy.

We assume that the AV can obtain a global map from an external service, and compute its position and rotation with on-board GPS and Inertial Measurement Unit (IMU). We further assume that the AV knows beforehand a few sparse checkpoints on its path to the goal, and the task of the AV is converted to reach the checkpoints consecutively.

Our perception module can produce semantic-rich structured data to learn a reward function by utilizing the context-aware semantic information. Features with clear semantic interpretations can help construct non-linear features such as “whether there is a car in front within 3 meters”, resulting in more flexible decision-making. This is particularly useful considering that IRL restricts the reward function to be a linear combination of the features.

The EIRL training process, shown in Fig. 2, consists of an offline step and an online step. In the offline step, we
use a planning algorithm as the expert to generate driving trajectories and compute the expert’s feature expectation. In the online step, we learn the feature weights under a non-uniform prior given the expert’s policy and the learned policy at the current iteration. Then, we construct a reward function using the learned feature weights. By further adopting the notion of learning from accidents [3], we use the resulting additional training data along with the newly constructed reward function for RL to update the learned policy. Inspired by transfer learning [19], the model parameters from the previous iteration are used as a starting point in the updating process.

**B. Context-aware Multi-sensor 3D Perception**

We simulate three RGB cameras for the front-view, left-view, and right-view, respectively, as well as a 360-degree Lidar of the AV. We combine one RGB image and the point cloud on each side to detect nearby vehicles. Then, we merge the results from all sides to obtain an overall view. The module’s architecture of one view is shown in Fig. 3 and the module works as follows.

First, it generates the front-view data, which contain segmentation and depth maps, and bird’s-eye-view image from the point cloud. By having the calibration data between the Lidar and camera, we can then align the front-view data with the RGB images. In the second step, we combine and feed the aligned front-view data with the RGB images into the feature extractor to obtain the front-view feature map. We apply the same procedure for the bird’s-eye-view image to obtain the bird’s-eye-view feature map. Then, we use the region proposal network (RPN)[20] and detection network from AVOD [4] to obtain the perception results for the front-view. Similarly, we obtain the perception results for the left-view and right-view. Finally, we merge all perception results together using the extrinsic calibration data of the three cameras.

In contrast to AVOD, which uses the point cloud to generate only the bird’s-eye-view image, we develop an efficient and effective technique to extract additional 2D information from the point cloud for 3D object detection:

- We map the point cloud onto the image space using the calibration data between the Lidar and camera. We then dilate the projected points to obtain the depth map.
- We process the “ground” and “wall” by first identifying the corresponding points in the point cloud, then mapping the points onto the image space to obtain segmentation maps. Specifically, for finding “ground”, we search for the plane that has the most points nearby in close proximity to an initial guessed “ground” from the IMU data. For finding “wall”, we search for planes that are perpendicular to the “ground” and have enough points nearby. Both “ground” and “wall” are contextual information that are critical for collision avoidance in autonomous driving. An example of finding the “ground” is shown in Fig. 4.
- We create the bird’s-eye-view image according to MV3D [16]. The resulting image contains height, intensity, and density information in different channels. The parameters used in training are from AVOD [4].

Our method can improve perception efficiency and performance as it extracts context-aware semantic information from the point cloud, i.e., segmentation map and depth map. This semantic information can be used to construct useful features, e.g., a car is unlikely to appear on a wall, or the further one of the two cars (of the same size) should not have a larger bounding box.

The output of the perception module contains 3D bounding boxes of nearby vehicles (dynamic obstacles). Our approach can also be extended to detect static obstacles when needed.

**C. Enhanced Inverse Reinforcement Learning (EIRL)**

The original IRL achieves imitation learning by first computing the expert’s feature expectation $\hat{\mu}(\pi_E) = \frac{1}{m} \sum_{i=1}^{m} \sum_{t=0}^{\infty} \gamma^t \phi(s_t^i)$, given $m$ trajectories $\{(s_0^i, \hat{a}_0^i), \ldots, (s_t^i, \hat{a}_t^i)\}$ from the expert’s policy $\pi_E$, the discount factor $\gamma$, and the feature vector $\phi(\cdot)$. Then, IRL learns $w (w \in \mathbb{R}^k$ and $\|w\|_1 \leq 1$) given $\pi_E$ and $\pi$ at the current iteration, and synthesizes a reward function $R(s) = w \cdot \phi(s)$, where $s$ is the state of the environment. Next, the policy $\pi$ is re-learned using RL. This iterative process continues until $\|\mu(\pi) - \hat{\mu}(\pi_E)\|_2 \leq \epsilon$.

The features used by EIRL are based on structured data from the perception module and the simulated environment, which include bounding boxes of nearby vehicles and static obstacles in the scene.

One limitation of IRL is that it imposes a uniform prior on all features, causing small weights to be possibly assigned to some crucial features during the learning process. For example, in the context of driving, we find that the feature collision can receive a small weight, since any collision behavior of the AV will terminate a training episode. This limitation of IRL can lead to subpar performance of the task learning to drive (see Sec. IV-B.2). To give an example, the expert can drive the car safely (without any collision)
around a simulated city, while a randomly initialized policy can hardly drive the car far without collision. In this case, the feature expectation of the expert is calculated using extended “global” trajectories, while the feature expectation of the learned policy is calculated using short “local” trajectories. This discrepancy will cause some important features to receive small weights during the minimization process of the feature expectations in IRL. A concrete example of this phenomenon is shown in Table I.

A concrete example of this phenomenon is shown in Table I.

### TABLE I

LIMITATION OF IRL. IRL CAN LEARN FEATURE WEIGHTS \( w \) BY MINIMIZING THE DIFFERENCE OF FEATURE EXPECTATION BETWEEN THE EXPERT’S POLICY \( \mu(\pi_E) \) AND A RANDOM POLICY \( \mu(\pi_R) \). WHILE BOTH POLICIES HAVE SMALL EXPECTATIONS FOR THE FEATURE \( \text{collision} \), THE ACTUAL TRAJECTORY FROM THE EXPERT’S POLICY CAN BE MUCH LONGER THAN THE TRAJECTORY FOR THE RANDOM POLICY. AS A RESULT, IRL ASSIGNS A NEGLIGIBLE WEIGHT FOR \( \text{collision} \) (I.E., \(-5.34 \times 10^{-6}\)), COMPARED TO THE WEIGHTS OF OTHER FEATURES.

<table>
<thead>
<tr>
<th>( \text{left dist} )</th>
<th>( \text{front dist} )</th>
<th>( \text{right dist} )</th>
<th>( \text{collision} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \mu(\pi_E) )</td>
<td>1.3914</td>
<td>369.50</td>
<td>25.84</td>
</tr>
<tr>
<td>( \mu(\pi_R) )</td>
<td>40.31</td>
<td>216.06</td>
<td>142.87</td>
</tr>
<tr>
<td>( w )</td>
<td>0.119</td>
<td>0.184</td>
<td>-0.124</td>
</tr>
</tbody>
</table>

To alleviate the aforementioned limitation of IRL, we propose EIRL—an approach that can incorporate a non-uniform prior on the features by allowing users to specify the weights of certain features for ensuring essential properties of a task, e.g., collision for driving. EIRL will then learn the weights of the rest of the features automatically. Formally, denoting the empirically-set weights as \( w_m = [w_0, w_1, \ldots, w_k] \) and the features to be learned as \( w_l = [w_{k+1}, w_{k+2}, \ldots, w_n] \), we have the overall weights \( w = [w_m, w_l] \). Because IRL requires \( \|w\|_1 \leq 1 \), we set \( \|w_m\|_1 = v < 1 \) and compute the rest of the features under the constraint \( \|w_l\|_1 \leq 1 - v \). Additionally, IRL retrains \( \pi \) at each iteration, which process can be inefficient. In EIRL, we improve the training efficiency by using the learned model parameters \( \theta(i) \) of \( \pi(i) \) at the \( i \)-th iteration as the initial parameters for training \( \pi(i+1) \) at the \( (i+1) \)-th iteration. The whole learning process stops when \( \|\mu(\pi) - \hat{\mu}(\pi_E)\|_2 < \epsilon \). The complete EIRL algorithm is shown in Algorithm 1.

Inspired by ADAPS [3], we adopt learning from accident to improve the training efficiency. Specifically, when the car crashes during the online training process, we will backtrack for certain frames and let the expert drive for certain frames to avoid collision. Both the crash and collision-free data will be (re-)used to train the policy.

Essentially, by imposing a non-uniform prior on extracted features, we can introduce certain expert experience to the learning process directly. In addition, although the synthesized reward function is different at each iteration, they may embed useful information to share with each other. Thus, inspired by transfer learning [19], we reuse learned parameters for continuous training for faster convergence. Lastly, the process of learning from accident can result in both negative examples and positive examples in learning, leading to better performance and faster convergence.

### D. Autonomous Driving Training Platform

Our platform is developed using the Unity game engine [21] and consists of the following components.

- Virtual sensors. Our platform supports simulated RGB camera, depth camera, Lidar, gyroscope, and GPS.
- Portable learning algorithms. Our platform supports end-to-end learning, perception plus traditional plan-
Algorithm 1: Enhanced Inverse Reinforcement Learning (EIRL)

Result: policy $\pi^{(i)}$

Initialization: Set $i = 0$ and $\epsilon = 0.1$;
Randomly set the model parameters $\theta^{(0)}$ for $\pi^{(0)}$;
Compute $\mu^{(0)}$

Empirically set $w_m$ such that $\|w_m\|_1 \equiv v < 1$

Compute $\epsilon^{(0)} = \max_{w_l:\|w_l\|_2 \leq 1} w_l^T(\mu(\pi_E) - \mu^{(0)})$;
($\mu_l$ is the feature expectation computed using $w_l$);
Let $w_l^{(1)}$ store the above maximum value and set $w^{(1)} = [w_m \ w_l^{(1)}]$;

while $\epsilon^{(i)} > \epsilon$ do

Set $i = i + 1$;
Compute the reward function $R = ((w^{(i)})^T \phi)$;
Using $R$ and $\theta^{(i-1)}$ in RL to compute an optimal policy $\pi^{(i)}$
Compute $\mu^{(x(i))}$
Compute $\epsilon^{(i)} = \max_{w_l:\|w_l\|_2 \leq 1} \min_{j\in\{0, \ldots, i\}} w_l^T(\mu(\pi_E) - \mu_l(\pi^{(i)}))$;
Let $w_l^{(i+1)}$ store the above maximum value and set $w^{(i+1)} = [w_m \ w_l^{(i+1)}]$
end

ning, and perception plus learning-based planning. These algorithms can be switched and replaced at ease.

- Communication and control module. Our platform supports cross-platform communication with sockets. This feature enables remote control of simulated AVs using input from either a planning algorithm or user.

- Data collection module. Our platform can operate various types of data, including sensor data, calibration data, and environment data in any format. For example, our platform can generate data that are compatible with the KITTI dataset [5] for 3D object detection.

IV. Experiments and Results

In this section, we detail our experiments and results. All experiments are conducted using an Intel(R) Xeon(TM) W-2123 CPU, an Nvidia GTX 1080 GPU, and 32G RAM.

A. Context-aware Multi-sensor 3D Perception

We evaluate our perception module using the KITTI dataset [5] and mAP (mean Average Precision) as the metric. We treat one detected 3D object as a true positive if and only if there exists one matched ground-truth 3D object, and their 3D bounding box IoU (Intersection over Union) is over 70%. We show the mAP scores of our evaluation using the easy, moderate, and hard cases of KITTI in Table II. Our method outperforms the other state-of-the-art methods by receiving the highest scores across all cases.

Next, we showcase that our method can suppress certain false positives because of the use of the segmentation map and depth map. Case 1 in Fig. 5(a) shows a false positive detected by AVOD [4] as a vehicle, which is suppressed by our method. Case 2 in Fig. 5(a) shows another false positive detected by AVOD: as we only used the “car” category of KITTI for training, the mini-bus should not be detected. Our method suppresses this false positive by successfully differentiating the two types of vehicles. Overall, we can suppress up to 15% false positives from AVOD using the same score threshold.

B. Enhanced Inverse Reinforcement Learning (EIRL)

1) Overall Performance: We compare EIRL to the original IRL, general RL, and end-to-end imitation learning (IM). We use the method by Bojarski et al. [24] as the IM, and deep Q-learning with [164, 150] hidden units in each of the 2 dense layers and 20% dropout rate in all RL-based methods. (We also experimented with more complex network architectures up to 6 hidden layers, but found [164, 150] with 2 hidden layers works best.) We train all methods for 24 hours. The features used in both the original IRL and EIRL include depth information, obstacle type, distance to the goal, direction to the goal, distance difference to the goal compared with the previous frame, and collision.

Our evaluation criterion is how far can the AV travel under a fixed number of steps. The AV will stop if it finishes 1000 steps or is in collision. Since the AV is assumed to know beforehand a series of checkpoints on its path to the goal, we compute the score based on the number of checkpoints reached by the AV:

$$s_{\text{final}} = (n_{\text{reached}} + (1 - \frac{\text{dist}_{\text{next}}}{\text{dist}_{\text{last,next}}})) \times s_{\text{unit}},$$

(1)

where $s_{\text{final}}$ is the final score, $n_{\text{reached}}$ is the number of checkpoints reached, $\text{dist}_{\text{next}}$ is the distance between the last position on the car’s trajectory and the next checkpoint on the car’s route, $\text{dist}_{\text{last,next}}$ is the distance between the last reached checkpoint to the next checkpoint, and $s_{\text{unit}}$ is the unit score by reaching a checkpoint, which is set to 100. Note that a negative score may appear if $\text{dist}_{\text{next}} > \text{dist}_{\text{last,next}}$, which happens when the car drives away from

<table>
<thead>
<tr>
<th>Model</th>
<th>Easy (mAP)</th>
<th>Moderate (mAP)</th>
<th>Hard (mAP)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MV3D [16]</td>
<td>71.09</td>
<td>62.55</td>
<td>55.12</td>
</tr>
<tr>
<td>VoxNet [22]</td>
<td>77.47</td>
<td>65.11</td>
<td>57.73</td>
</tr>
<tr>
<td>F-PointNet [23]</td>
<td>81.20</td>
<td>70.39</td>
<td>62.19</td>
</tr>
<tr>
<td>AVOD-FPN [4]</td>
<td>81.94</td>
<td>71.88</td>
<td>66.38</td>
</tr>
<tr>
<td>Ours</td>
<td>82.16</td>
<td>73.22</td>
<td>67.38</td>
</tr>
</tbody>
</table>

TABLE II

Comparison of 3D Detection Models. Our method achieves the highest mAP scores across all cases in the KITTI dataset [5].
the next checkpoint. We use three scenes shown in Fig. 6 for evaluation:

- Scene 1: open space with only moving vehicles;
- Scene 2: city street with only static obstacles;
- Scene 3: city street with moving vehicles and static obstacles.

We record the final scores $s_{final, 2}$ and $s_{final, 3}$ from Scene 2 and 3; and average trajectory length $l_{final, 1}$, $l_{final, 2}$, $l_{final, 3}$ from Scene 1, 2, and 3. Because $s_{final, 1}$ is computed based on the goal position and there is no explicit goal in Scene 1, we do not compute $s_{final, 1}$. All data are obtained by running each learning method for 100 times. Our method achieves the highest scores and can enable the AV to drive safely 10x further (and longer) than the other methods. The full results are shown in Table III.

![Fig. 6. Screenshots of the three scenes used in our evaluation. From left to right: Scene 1, Scene 2, Scene 3.](image)

**2) Attribute Effectiveness:** The first attribute of our approach is the non-uniform prior for features. To test this attribute, we empirically set the weight for collision to be $-0.8$. We count the number of collisions from the original IRL and IRL+non-uniform prior (EIRL) by running both approaches for 10,000 steps. As shown in Table IV, EIRL can reduce the number of collisions up to 41%.

![Fig. 7. By using previous model parameters for continuous training, we can achieve comparable score (model performance) 2.5x faster.](image)

**TABLE III**

<table>
<thead>
<tr>
<th>Method</th>
<th>$s_{final, 2}$</th>
<th>$s_{final, 3}$</th>
<th>$l_{final, 1}$</th>
<th>$l_{final, 2}$</th>
<th>$l_{final, 3}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>IM</td>
<td>77.8</td>
<td>60.1</td>
<td>105.6 m</td>
<td>53.7 m</td>
<td>44.7 m</td>
</tr>
<tr>
<td>RL</td>
<td>99.8</td>
<td>59.1</td>
<td>72.9 m</td>
<td>59.7 m</td>
<td>39.7 m</td>
</tr>
<tr>
<td>IRL</td>
<td>110.7</td>
<td>59.7</td>
<td>228.8 m</td>
<td>60.4 m</td>
<td>33.2 m</td>
</tr>
<tr>
<td>Ours</td>
<td><strong>205.8</strong></td>
<td><strong>177.3</strong></td>
<td><strong>276.3 m</strong></td>
<td><strong>748.4 m</strong></td>
<td><strong>324.2 m</strong></td>
</tr>
</tbody>
</table>

**TABLE IV**

<table>
<thead>
<tr>
<th>Model</th>
<th>Scene 1</th>
<th>Scene 2</th>
<th>Scene 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>IRL</td>
<td>35</td>
<td>58</td>
<td>111</td>
</tr>
<tr>
<td>IRL+non-uniform prior</td>
<td>33</td>
<td>41</td>
<td>93</td>
</tr>
</tbody>
</table>

The second attribute of our approach is learned model parameters for continuous training, which aims to improve training efficiency. From the results shown in Fig. 7 we can see that by having this attribute, we can achieve comparable model performance at 2.5x faster.

The last attribute of our approach is learning from accidents. We use ORCA [25] as the expert algorithm to generate alternative safe trajectories during the analysis of an accident. These trajectories are then used to generate additional training data for our algorithm. In Fig. 8 we show that the learning algorithm with this attribute added can achieve much higher scores up to two orders of magnitude in near collision scenarios under the same number of epochs.

![Fig. 8. Having the additional training data by learning from accidents, the learning algorithm achieves higher scores up to two orders of magnitude in near collision scenarios under the same number of epochs.](image)

**C. Driving Cases**

Our method can enable safe autonomous driving (without collision) in scenes with both static and dynamic obstacles. We show the results of certain driving cases achieved by our method, as shown in the supplementary video.

**V. CONCLUSION AND FUTURE WORK**

We propose a framework integrating context-aware multi-sensor perception and enhanced inverse reinforcement learning for autonomous driving. We evaluate our approach using a variety of experiments, over the entire algorithm and each individual component. As shown in all comparison results, our method outperforms the other state-of-the-art methods on all experiments.

There are some limitations of this work. First, the inference efficiency of the perception module and the EIRL module can be improved. Second, our approach inherits limitations of IRL. One of them is to encode multiple expert trajectories into a single feature expectation, which can be subject to information loss.

Future directions of this work are abundant. We plan to develop a more efficient perception module by leveraging the sparsity embedded in the input data [26]. We also hope to alleviate the information loss during the expert’s feature expectation computation. Lastly, We plan to test our approach in dense virtual traffic [27] that is estimated and reconstructed using real-world traffic data [28], [29], [30].

**REFERENCES**


