Inverse Reinforcement Learning with Hybrid-weight Trust-region Optimization and Curriculum Learning for Autonomous Maneuvering

Yu Shen[1], Weizi Li[2], and Ming C. Lin[1]

Abstract— **Despite significant advancements, collision-free navigation in autonomous driving is still challenging, considering the navigation module needs to balance learning and planning to achieve efficient and effective control of the vehicle. We propose a novel framework of inverse reinforcement learning with hybrid-weight trust-region optimization and curriculum learning (IRL-HC) for autonomous maneuvering. Our method can incorporate both expert demonstration (from real driving) and domain knowledge (hard constraints such as collision avoidance, goal reaching, etc. encoded in reward functions) to learn an effective control policy. The hybrid-weight trust-region optimization is used to determine the difficulty of the task curriculum for fast incremental curriculum learning and improve the efficiency of inverse reinforcement learning by hybrid weight tuning of different sets of hyperparameters. IRL-HC is also compatible with domain-dependent techniques such as *learn-from-accident*, which can further boost performance. Overall, IRL-HC can reduce the number of collisions *up to 48%*, increase the training efficiency *by 2.8x*, and enable the vehicle to drive *10x* further compared to other methods.**

I. Introduction

Autonomous driving generally can be realized via either an end-to-end system or a mediated-perception approach [1]. The former takes in raw sensor data and directly output control commands (e.g., steering angles), which usually results in a succinct training pipeline at the cost of model interpretability. The later decouples perception and navigation, thus offering better model interpretability with enhanced driving safety. However, a mediated-perception approach commonly adopts a planning algorithm for navigating a self-driving car, which can be computationally expensive, given the requirement of an holistic environment information for achieving global optimality and planning in high-dimensional state space.

We propose an efficient and novel mediated-perception framework for autonomous driving that exploits context-aware multi-sensor perception for inverse reinforcement learning with hybrid-weight trust-region optimization and curriculum learning (IRL-HC). The perception module interprets unstructured information (i.e., images and point clouds) of an environment using multiple sensors, extracts context-aware information, and produces structured information (e.g., the shape and position of an object). IRL-HC then takes the structured information along with expert trajectories as input to learn a control policy for autonomous vehicles (AV).

[1]Yu Shen and Ming C. Lin are with the Department of Computer Science, University of Maryland at College Park {yushen,lin}@cs.umd.edu

[2]Weizi Li is with the Department of Computer Science, University of Memphis wli@memphis.edu

Our novel IRL-HC contains two main elements: (a) hybrid-weight trust-region optimization and (b) curriculum learning. Hybrid-weight trust-region optimization addresses a fundamental limitation of the original IRL [2]: a *uniform prior* is imposed on all features, which can lead to subpar imitation performance of the learner to the expert *even their feature expectations converge and match* [3]. In the context of autonomous driving, since some hard constraints like collision avoidance for safety and goal-oriented navigation to destination are most critical over other considerations, this limitation due to uniform priors in weight tuning can lead to frequent collisions. Our method alleviates this problem by imposing a *non-uniform prior* on task features and updating the features' weights using hybrid-weight *trust-region optimization*, which is progressively updated to automate weight-tuning optimization. This design of *hybrid-weight* optimization enables the use of both expert demonstrations and domain knowledge for learning an effective policy that takes into account of both experts (e.g., driver) and desired constraints (e.g., reaching the goal without collision). The second element of IRL-HC is curriculum learning, which has been shown effective in improving RL agents' performance [4]. Our key insight is that the trust-region is naturally linked to the estimation of the task learning progress—which can then be used to determine the difficulty of the task curriculum that is built on the maximum step size for the learner to progress on the expert's trajectory.

In summary, we introduce a novel, efficient IRL framework consisting of hybrid-weight trust-region optimization and curriculum learning (IRL-HC), supported by a mediated-perception module that provides context-aware multi-sensor perception, as shown in Fig. 1. It offers several advantages:

- Hybrid-weight trust-region optimization improves upon IRL [2] by imposing non-uniform priors on task-critical features, e.g., collision avoidance and goal-seeking, incorporating *both* expert demonstration and domain knowledge to automate weight tuning effectively;
- Curriculum learning retains important lessons through *increasingly difficult* RL to task learning, thus improving overall training efficiency and performance;
- Curriculum learning also utilizes the hybrid-weight trust-region optimization to *assess curriculum difficulty*;
- IRL-HC is further compatible with domain-dependent techniques, such as learn-from-accident [5], which generate safe trajectories, further boosting the overall performance in autonomous driving.

The effectiveness and efficiency of IRL-HC are demon-

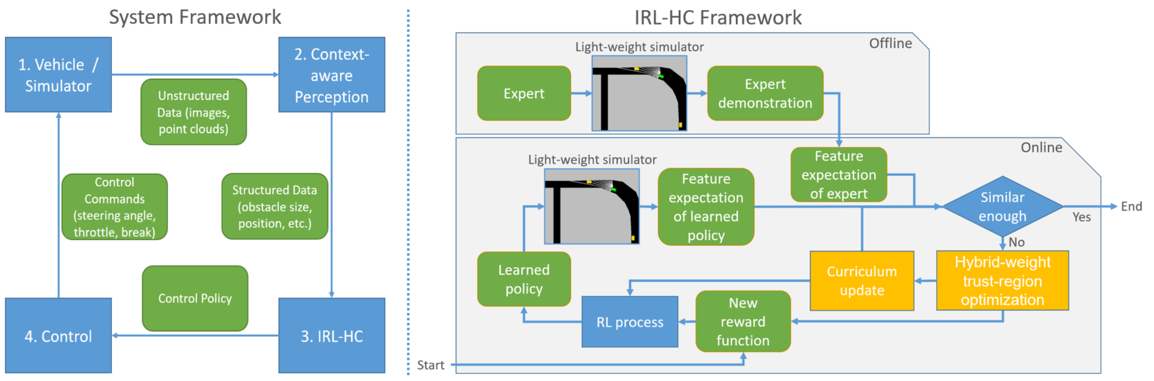

System Framework
1. Vehicle / Simulator
2. Context-aware Perception
Unstructured Data (images, point clouds)
Control Commands (steering angle, throttle, break)
Structured Data (obstacle size, position, etc.)
Control Policy
4. Control
3. IRL-HC
IRL-HC Framework
Offline
Light-weight simulator
Expert
Expert demonstration
Online
Light-weight simulator
Feature expectation of learned policy
Feature expectation of expert
Similar enough
Yes
End
No
Learned policy
Curriculum update
Hybrid-weight trust-region optimization
RL process
New reward function
Start

Fig. 1. **System Pipeline (LEFT)**. 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 IRL-HC module to learn a control policy for autonomous driving. **IRL-HC training process (RIGHT).** We compute the expert trajectories offline. Next we use hybrid-weight trust-region optimization to obtain a new reward function, which is then used to compute the feature expectation of a learned policy online. The curriculum difficulty will also be updated progressively, according to the trust-region updating rules that assess the learned task difficulty.

strated in a variety of experiments. To show IRL-HC can work without perfect perception (i.e., use ground-truth data), we run IRL-HC in the autonomous system described in Sec. III-A. Overall, our method can enable the vehicle to drive safely up to **10x** further than existing SOTA methods, assist in reducing the number of collisions up to **48%**, and **2.8x** faster training. In addition to the statistical results, we show that IRL-HC can steer the vehicle to avoid both static and dynamic obstacles, even in the presence of a narrow passage (see project website).

II. Related Work

Various methods [6], [7] have been proposed to address the perception, planning, and control of an **autonomous vehicle** (**AV**). Examples of the end-to-end approach include end-to-end reinforcement learning [8] and end-to-end imitation learning [5], [9], [10], [11], [12]. 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 [13] 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 [5], 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 generalize better as it is based on RL rather than supervised learning for imitating the expert.

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 RL to learn a control policy [2]. To provide some examples, Sharifzadeh et al. [14] apply Deep Q-Networks to extract a reward function in large state space. You et al. [15] use deep neural networks to approximate the latent reward function of the expert and then apply deep Q-learning to obtain the control policy.

Maximum entropy IRL [16] provides a probabilistic approach based on the entropy concept. Hierarchical guidance [17] leverages the hierarchical structure of the underlying problem to integrate different modes of expert interactions. Guided Cost Learning [18] uses a nonlinear cost function and guided sampling strategy. GAIL [19] draws the analogy between imitation learning and generative adversarial networks and develops an effective model-free imitation learning for complex, high-dimensional tasks. Maximum entropy deep IRL [20] combines neural networks and traditional IRL. Another work [21] proposes a scalable IRL algorithm based on an adversarial reward learning process. Brown et al. [22] study the suboptimal demonstrations in IRL. Some recent studies [3], [23] explore the feature design in IRL to better meet human's intention. Compared to other IRL-based methods, our approach incorporates both expert demonstrations and domain knowledge to design an effective initial reward function for obtaining an effective policy via hybrid-weight trust-region optimization and curriculum learning. Our framework can also be used with feature design [3], [23] at ease.

Curriculum learning for RL has gained much attention recently [4], [24]. Early studies use curricula for tasks such as grammar learning [25] and robotics control problems [26]. Some well-known work such as AlphaGo [27] implicitly use curricula to guide training. Narvekar et al. [28] propose a meta-MDP to select tasks for the learning agent. Recently Song et al. [29] propose a three-stage curriculum RL to train an autonomous racing agent.

III. Approach

A. *Framework Overview*

Our framework combines context-aware multi-sensor perception and inverse reinforcement learning with hybrid-weight trust-region optimization and curriculum learning (IRL-HC). The perception module takes multiple sensors input and produces structured data, which are then used by IRL-HC to learn a control policy, see Fig. 1 (LEFT).

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 waypoints on its path to the goal, and the task of the AV is converted to reach the waypoints 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 IRL-HC training process, shown on the right of Fig. 1, 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. In the online step, we learn the feature weights with hybrid-weight trust-region optimization given the expert's policy and the learned policy at the current iteration and non-uniform prior. Then, we construct a reward function using the learned feature weights. The difficulty of curriculum will be updated according to the trust-region updating rule. By further adopting the notion of *learn from accidents* [5], 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 [30], the model parameters from the previous iteration are used as a starting point in continue training.

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 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. Next, 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. We use the region proposal network (RPN) [31] and detection network from AVOD [32] 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. The output of the perception module contains 3D bounding boxes and types of nearby dynamic obstacles. Our approach can also be extended to detect static obstacles when needed. The context-aware, multi-sensor 3D perception detects different types of information to produce features used by IRL.

C. IRL with Hybrid-weight Trust-region Optimization

We first review original IRL, point out some issues with IRL, then present our solution by extending the *trust-region optimization* to our scenario. To the best of knowledge, our method is the first one that can take advantages of both demo (expert data) and domain knowledge (reward function), leading to overall better performance.

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 $\{s_0^{(i)}, s_1^{(i)}, \dots\}_{i=1}^{m}$ from the expert's policy π_E, the discount factor γ, and the feature vector $\phi(\cdot)$. Then, IRL learns w ($w \in \mathbb{R}^k$ and $\|w\|_1 \leq 1$) given π at the current iteration and π_E, and synthesizes a reward function $R(s) = w \cdot \phi(s)$, where s is the state of the environment. Next, the policy π is re-learned using RL. This iterative process continues until $\|\mu(\pi) - \hat{\mu}(\pi_E)\|_2 \leq \epsilon$. The final policy is selected among all learned policies from all iterations.

The features used by IRL-HC are based on structured data from the perception module, which include bounding boxes of nearby vehicles and static obstacles in the scene. See detailed feature list in Sec. IV.

One fundamental 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 as a result of any collision behavior of the AV will terminate a training episode. This limitation of IRL can lead to subpar task performance (see Sec. IV-C). To give an example, the expert can drive the car safely (without any collision), 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 is likely to cause some important features to receive small weights during the minimization process of the feature expectations in IRL. A concrete example of this phenomenon from our experiments is shown in Table I.

TABLE I

Limitations of IRL. IRL LEARNS FEATURE WEIGHTS w BY MINIMIZING THE DIFFERENCE OF FEATURE EXPECTATION BETWEEN THE EXPERT'S POLICY $\mu(\pi_E)$ AND A RANDOM POLICY $\mu(\pi_0)$. WHILE BOTH POLICIES HAVE SMALL EXPECTATIONS FOR THE FEATURE *collision*, THE ACTUAL TRAJECTORY FROM THE EXPERT'S POLICY CAN BE MUCH LONGER THAN THE TRAJECTORY FROM A RANDOM POLICY. AS A RESULT, IRL ASSIGNS A NEGLIGIBLE WEIGHT TO *collision* (I.E., $-5.34e-06$).

feature	left dist.	front dist.	right dist.	...	**collision**
weight	0.113	0.184	-0.124	...	-0.00000534
$\mu(\pi_E)$	139.14	369.50	25.84	...	0
$\mu(\pi_0)$	40.31	216.06	142.87	...	0.005

To alleviate the aforementioned limitation of IRL, we propose an approach that not only incorporates 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, but also uses a hybrid weight tuning to update the policy so that the vehicle can adapt to different environments while maintaining desired behaviors. Formally, the overall weights $w = [w_m, w_l]$ consist of empirically-initialized weights $w_m = [w_1, \ldots, w_k]$ and the weights to be learned from scratch $w_l = [w_{k+1}, w_{k+2}, \ldots, w_n]$, where n is the total number of features. To achieve an optimal policy in diverse environments, we use trust-region optimization [33] to automatically tune w_m by avoiding violent exploratory behaviors. Specifically, in the ith iteration, we first try to update ϵ and w by solving the following quasi-convex optimization program:

$$\epsilon^{(i)} = \max_{\substack{w^{(i+1)}:\|w^{(i+1)}\|_2 \le 1 \\ \|w_m^{(i+1)} - w_m^{(i)}\|_2 \le \Delta}} \{ \min_{j \in \{0,\ldots,i\}} (w^{(i+1)})^T (\mu(\pi_E) - \mu(\pi^{(j)})) \}, \tag{1}$$

where Δ is the trust-region radius ($\Delta = \Delta^{(i)}$). The ratio $\epsilon^{(i)}/\epsilon^{(i-1)}$ is used to determine the acceptance of the newly updated w. Since the predicted upper bound of $\epsilon^{(i)}/\epsilon^{(i-1)}$ is $\frac{n}{\sqrt{n^2+(1-\gamma)^2\epsilon^2}}$ [2], we set the acceptance condition to

$$\frac{\epsilon^{(i)}}{\epsilon^{(i-1)}} \le \alpha \frac{n}{\sqrt{n^2 + (1-\gamma)^2 \epsilon^{*2}}}, \tag{2}$$

where $\alpha < 1$ is the threshold parameter. If the condition is failed, we drop the newly found w, and update ϵ and w by solving Eq. 1 with $\Delta = 0$. As the last step of one iteration, we update Δ as follows:

$$\Delta^{(i+1)} = \begin{cases} \min(c_u \Delta^{(i)}, b_u) & \text{Eq. 2 met} \\ \max(c_l \Delta^{(i)}, b_l) & \text{Eq. 2 failed p times} \\ \Delta^{(i)} & \text{otherwise} \end{cases} \tag{3}$$

where b_u and b_l are the upper- and lower-bound of the trust-region radius, and $c_u > 1$ and $c_l < 1$ are the coefficients.

Additionally, IRL retrains π at each iteration, such a process can be inefficient. In IRL-HC, we improve the training efficiency by using the learned model parameters $\theta^{(i)}$ of $\pi^{(i)}$ at the ith iteration as the initial parameters for training $\pi^{(i+1)}$ at the $(i+1)$th iteration. The whole learning process stops when $\|\mu(\pi) - \hat{\mu}(\pi_E)\|_2 < \epsilon$.

In many cases, we can design a reward function to incorporate domain knowledge. For example, in autonomous driving we want the vehicle to reach the goal without collision; an example reward function could be $r = w_1 g - w_2 c$, where g is the "goal-reaching" flag, c is the collision flag, and w_1 and w_2 are positive coefficients. In complex scenarios such as our city environment, however, such a naive reward function does not work well: sometimes learned policy will let the car stop there forever to avoid collision, or run in a circle in a very small area (keep running without collision but cannot achieve the goal). In this case, we can learn from expert demonstrations to complete the task.

We can represent any reward function as a sum of its linear/nonlinear components. To set the initial reward function for IRL-HC, we add each component of the reward function into the feature vector $\phi(\cdot)$ and add the corresponding coefficient into empirically-initialized weights w_m. For these components, since they already contain prior information, we do not want to aggressively update them (in fact in the beginning of IRL, aggressively updating feature components can deteriorate the learning performance because the initial policy is unstable)—this is achieved via hybrid-weight trust-region optimization.

Algorithm 1: Inverse Reinforcement Learning with Hybrid-weight Trust-region Optimization and Curriculum Learning (IRL-HC)

```
Result: policy $\pi^{(i)}$
Initialization: Calculate $\mu(\pi_E)$ with expert trajectories;
Set $i = 0$, set $\epsilon, \gamma, \alpha, b_u, b_l, c_u, c_l, p, e_u, N_u$;
Randomly set the model parameters $\theta^{(0)}$ for $\pi^{(0)}$;
Compute $\mu(\pi^{(0)})$;
Set $w_m^{(0)}$ such that $\|w_m^{(0)}\|_2 < 1$ (initial reward weights), $w_l^{(0)} = \mathbf{0}$;
Compute $\epsilon^{(0)} = (w^{(0)})^T(\mu(\pi_E) - \mu(\pi^{(0)}))$, where $w^{(0)} = [w_m^{(0)} w_l^{(0)}]$;
Set $\Delta^{(0)}, n_s^{(0)}$;
while $\epsilon^{(i)} > \epsilon$ do
    Set $i = i + 1$;
    Compute the reward function $R = ((w^{(i-1)})^T \phi)$;
    Using $R$, $\theta^{(i-1)}$, and $n_s^{(i-1)}$ in RL to compute an optimal policy $\pi^{(i)}$;
    Compute $\mu(\pi^{(i)})$;
    Solve Optimization 1 with $\Delta = \Delta^{(i-1)}$, and get solution $\epsilon^{(i)}$ at $w^{(i)}$;
    if Eq. 2 is True then
        Accept $\epsilon^{(i)}$ and $w^{(i)}$;
    else
        Reject, solve Eq. 1 with $\Delta = 0$, and update $\epsilon^{(i)}$ and $w^{(i)}$;
    end
    Set $\Delta^{(i)}$ with Eq. 3 (trust-region update);
    Set $n_s^{(i)}$ with Eq. 5 (curriculum difficulty update);
end
```

Key Insight: Our method is different from *manually setting initial weights* for IRL. Notice the IRL update the feature weights by solving an optimization without the previous feature weights (although the previous feature weights will influence the feature expectation of the learned policy in the optimization implicitly, the uncertainty of RL process will reduce such relation). This means the updated weights can be quite different from the previous weights. *Adding the trust-region mechanism can bound the weights during the entire IRL process while allowing tuning within a safe region.* In addition, our method is different from simple *reweighting mechanism* since ours contains an optimization under different constraints for two sets of parameters.

D. Curriculum Learning with Trust-region

Another problem of IRL is efficiency, since IRL contains multiple RL training process. We use Sequence Curriculum [4] in IRL-HC to help train IRL, aiming to improve the performance as well as the efficiency. Observing that the trust-region implicitly reflects the current learning progress, we use the trust-region updating criteria (Eq. 2) to determine when to increase the curriculum difficulty. Specifically, we use the maximum step number of each trajectory n_s as an curriculum difficulty variable (large n_s leads to an increase of the state space, thus increasing the learning difficulty). Meanwhile, we also monitor the safe step number of learned policy at each iteration to show the effectiveness of the policy learned so far:

$$n_{lp} \geq \beta \min(n_s^{(i)}, n_E) \tag{4}$$

where $\beta < 1$ is the threshold parameter, n_{lp} is the safe step number of the trajectory generated by the learned policy, n_E is the safe step number of the trajectory generated by the expert. n_s will be initialized with a small number and then increased with the following formula after each round of hybrid-weight trust-region optimization:

$$n_s^{(i+1)} = \begin{cases} \min(e_u n_s^{(i)}, N_u) & \text{Eq. 2 and Eq. 4 met} \\ n_s^{(i)} & \text{otherwise} \end{cases} \tag{5}$$

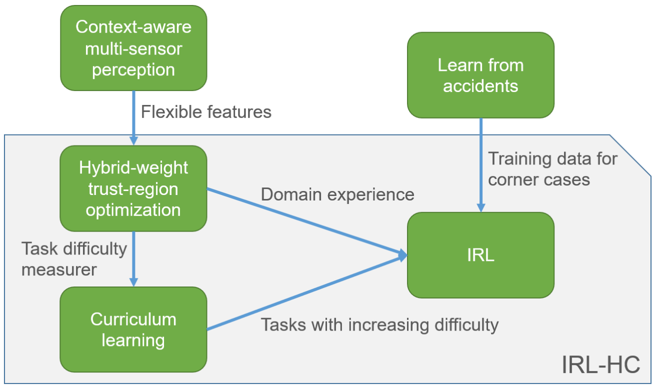
where $e_u > 1$ is the coefficient and N_u is the upper-bound of the n_s. Notice the curriculum is designed to be gradually harder, so n_s will not decrease, which is different than the trust-region radius. After the update of n_s, the RL process will use n_s and the new reward function given by hybrid-weight trust-region optimization to update the control policy. The IRL-HC algorithm is shown in Algorithm 1. In addition, *as the task difficulty gradually increases, we reuse learned parameters in continue training. This approach enables the transfer learning between tasks in curriculum learning and improves the performance of the learning agent.*

E. Learn from Accidents

IRL-HC is also compatible with other domain-dependent techniques. Inspired by ADAPS [5], we adopt *learn 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. Lastly, the process of *learn from accident* can result in both negative examples and positive examples in learning, leading to better performance and faster convergence.

F. Framework Design

We have introduced the key components of our framework and here we provide an integrated interpretation as well as module relation (see Fig. 2). Context-aware multi-sensor perception converts raw sensor data to structured information, providing flexibility in choosing features to be used in the hybrid-weight trust-region optimization. Next, *domain experience (via non-uniform prior) is incorporated by the optimization program into IRL for improving learning performance, and provides difficulty measurements for curriculum development. Curriculum learning then generates tasks with increasing difficulty to further accelerate the convergence rate of the learning agent.* Lastly, learn from accident provides data for corner cases, balancing the overall training data distribution.



Context-aware multi-sensor perception
Learn from accidents
Flexible features
Hybrid-weight trust-region optimization
Training data for corner cases
Domain experience
IRL
Task difficulty measurer
Curriculum learning
Tasks with increasing difficulty
IRL-HC

Fig. 2. **Module relation of our framework.** The perception module converts sensor data to various features, which serve as input to the hybrid-weight trust-region optimization. Subsequently, curriculum learning is adopted to generate tasks with increased difficulty to train the learning agent. The learn from accident process further provides training data of corner cases to balance the data distribution, leading to overall better performance.

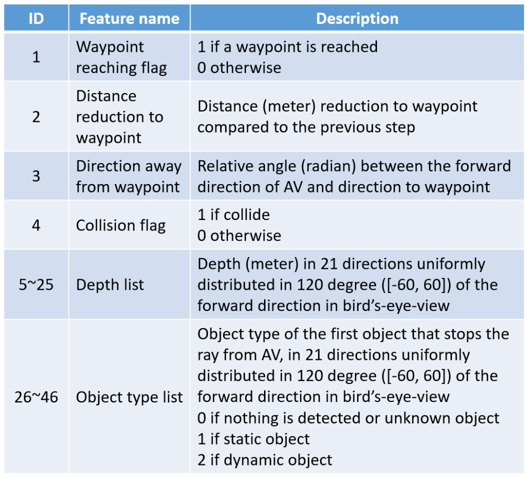
IV. Experiments and Results

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

A. Overall Performance

We compare IRL-HC to the original IRL, Generative Adversarial Imitation Learning (GAIL) [19], adversarial inverse reinforcement learning (AIRL) [21], and end-to-end imitation learning (IM) [34]. We use deep Q-learning with [164,150] hidden units in each of the 2 dense layers and 20% dropout rate in all RL-based methods except for GAIL, where we use TRPO [35] as proposed in the original paper. We also experiment with Q-learning using more complex network architectures up to 6 hidden layers, but found [164,150] with 2 hidden layers works the best. The features used in IRL-based methods are listed in Fig. 3.

The action space contains 25 discrete {rotational speed, acceleration} action pairs using 5 levels of rotational speed (steering angle) and 5 levels of acceleration (throttle value and break value). We set policy similarity threshold $\epsilon = 0.1$ and discount factor $\gamma = 0.99$ for both IRL and IRL-HC. We collect 20,000 steps of the expert trajectory and train



ID	Feature name	Description
1	Waypoint reaching flag	1 if a waypoint is reached 0 otherwise
2	Distance reduction to waypoint	Distance (meter) reduction to waypoint compared to the previous step
3	Direction away from waypoint	Relative angle (radian) between the forward direction of AV and direction to waypoint
4	Collision flag	1 if collide 0 otherwise
5~25	Depth list	Depth (meter) in 21 directions uniformly distributed in 120 degree ([-60, 60]) of the forward direction in bird's-eye-view
26~46	Object type list	Object type of the first object that stops the ray from AV, in 21 directions uniformly distributed in 120 degree ([-60, 60]) of the forward direction in bird's-eye-view 0 if nothing is detected or unknown object 1 if static object 2 if dynamic object

Fig. 3. **Features used in IRL and IRL-HC (46 in total).**

all methods for 24 hours (we also observe that training 12 more hours will not improve the performance). For IRL-HC, we set the trust-region acceptance coefficient $\alpha = 0.9$, trust-region increasing and decreasing coefficients $c_u = 1.1, c_l = 0.9$, upper- and lower-bound $b_u = 0.05, b_l = 0.001$, and consecutive rejection number $p = 5$. We empirically initialize weights for the first 4 features in Fig. 3 with values [0.4, 0.01, -0.1, -0.8] (represents the initial reward function), and let IRL-HC learn the weights of depth- and object type-related features. The perception network is trained using 10,000 frames of simulated data.

Our evaluation criterion is *the distance travelled by the AV under a fixed number of steps without any collision*. The AV will stop if it finishes $1,000$ steps or is in collision. Since the AV is assumed to know beforehand a series of waypoints on its path to the goal, we compute the score based on the number of waypoints reached by the AV:

$$s_{final} = (n_{reached} + (1 - \frac{dist_{next}}{dist_{last\_next}})) \times s_{unit}, \quad (6)$$

where s_{final} is the final score, $n_{reached}$ is the number of waypoints reached, $dist_{next}$ is the distance between the last position on the car's trajectory and the next waypoint on the car's route, $dist_{last\_next}$ is the distance between the last reached waypoint to the next waypoint, and s_{unit} is the unit score by reaching a waypoint, which is set to 100. Note that a negative score may appear if $dist_{next} > dist_{last\_next}$, which happens when the car drives away from the next waypoint. We use three scenes shown in Fig. 4 for evaluation (test field about 160m x 160m):

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

B. Comparison with Other SOTA Methods

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} is computed based on the waypoint position and there is no explicit

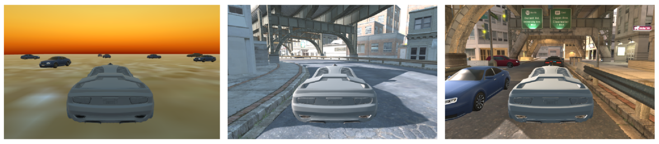


Fig. 4. Screenshots of the three scenes used in our evaluation. From left to right: Scene 1, Scene 2, Scene 3.

waypoint in Scene 1, we do not compute $s_{final,1}$. All data are obtained by learning 3 times and testing 100 times with randomly initialized scene configurations such as the start position and direction of the AV, and the start position, direction and speed of obstacle vehicles. 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 II.

TABLE II
Performance of IRL-HC vs. other SOTA methods, USING THE SCORE DEFINED IN EQ. 6 AND SAFE-TRAJECTORY LENGTH. OUR METHOD NOT ONLY ACHIEVES THE HIGHEST SCORES BUT ALSO ENABLES THE AV TO DRIVE SAFELY 10X FURTHER THAN THE OTHER METHODS.

Method	$s_{final,2}$	$s_{final,3}$	$l_{final,1}$	$l_{final,2}$	$l_{final,3}$
IM [34]	77.4	60.1	105.6 m	53.7 m	44.7 m
IRL [2]	110.7	59.7	228.8 m	69.4 m	33.2 m
GAIL [19]	103.0	52.8	49.1 m	69.9 m	35.1 m
AIRL [21]	119.9	83.6	74.1 m	73.6 m	50.7 m
Ours	**203.2**	**179.6**	**279.1** m	**733.5** m	**335.9** m

We also compare performance between methods using reward functions vs. expert demonstration in Table III. The reward function for RL is the same as the initial reward function for our method. IRL with only expert data cannot learn well either, while our method with both expert data and reward functions performs the best.

TABLE III
Overall performance comparison between methods using reward function vs. expert data. RL WITH REWARD FUNCTION (R) OR IRL WITH ONLY EXPERT DATA (E) DOES NOT PERFORM WELL ON THE TASK, WHILE OURS WITH BOTH (E+R) PERFORMS THE BEST.

Method	$s_{final,2}$	$s_{final,3}$	$l_{final,1}$	$l_{final,2}$	$l_{final,3}$
RL (R)	99.8	59.1	72.9 m	59.7 m	39.7 m
IRL (E)	110.7	59.7	228.8 m	69.4 m	33.2 m
Ours (E+R)	**203.2**	**179.6**	**279.1** m	**733.5** m	**335.9** m

The analysis of the results shown in Table II and Table III is the following. RL relies on a good reward function constructed using domain knowledge, which can be difficult to obtain when addressing complex control tasks. IM does not generalize well to new environments since it "copies" the expert trajectory. IRL, GAIL, or AIRL only considers the expert trajectory while our method is able to combine expert trajectory and domain knowledge for learning. The hybrid-weight trust-region optimization enables our method to adapt to different environments. The curriculum learning further enables the learning agent to behave in gradually more difficult tasks while the reuse of the model parameters helps improve training efficiency. In addition, learn from accidents provides the RL agent with more training data for corner cases. Each above-mentioned element has contributed to the performance boost of the learning agent, thereby resulting in considerably better overall performance than other methods shown in Table II. Note that all these analyses are not restricted to the autonomous driving task but likely to be relevant for other robot learning tasks.

C. Ablation Study

Here, we first show the effectiveness of individual components of our framework in Table IV. H, HC, LfA stand for "hybrid-weight trust-region optimization," "hybrid-weight trust-region optimization and curriculum learning," and "learn from accidents," respectively. Then, we show the combination of all components leads to the best performance. Note that the experiment of "IRL + curriculum learning" is omitted since the curriculum design depends on the trust-region optimization.

TABLE IV

Ablation study of individual components. THE COMBINATION OF ALL COMPONENTS LEADS TO THE BEST OVERALL PERFORMANCE.

Method	$s_{final,2}$	$s_{final,3}$	$l_{final,1}$	$l_{final,2}$	$l_{final,3}$
IRL [2]	110.7	59.7	228.8 m	69.4 m	33.2 m
IRL + H	130.7	103.4	235.6 m	140.4 m	133.9 m
IRL + HC	174.9	137.4	253.8 m	305.6 m	202.8 m
IRL + LfA	124.8	84.6	241.4 m	103.6 m	61.5 m
Ours (IRL+ALL)	**203.2**	**179.6**	**279.1** m	**733.5** m	**335.9** m

Next, we show the effectiveness of the main attributes, i.e., hybrid-weight trust-region optimization and curriculum learning, using the number of collisions encountered by the vehicle. To be specific, we count the number of collisions from the original IRL, IRL+H, and IRL+HC by running all approaches for 10,000 steps. As shown in Table V, IRL+HC can reduce the number of collisions up to 48%.

TABLE V

The number of collisions in different scenes OVER 10,000 STEPS: ORIGINAL IRL VS. IRL+H VS. IRL+HC. IRL+HC (OUR APPROACH) CAN REDUCE THE NUMBER OF COLLISIONS UP TO 48%.

Model	Scene 1	Scene 2	Scene 3
IRL	35	58	111
IRL+H	33	41	93
IRL+HC	**25**	**30**	**78**

Except for performance, curriculum learning in our approach also aims to improve training efficiency. From the results shown in Fig. 5, we can see that by having this attribute, we can achieve comparable model performance at *2.8x faster*.

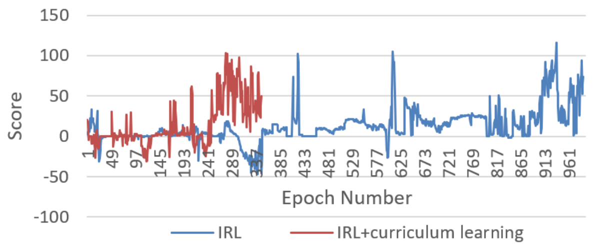


Fig. 5. Using curriculum learning, we can achieve comparable score (model performance) **2.8x** faster.

D. Demo Driving Cases

Our method can enable safe autonomous driving by avoiding both static and dynamic obstacles. We show the results of some driving cases achieved by our method in Fig. 6. The full demo video can be found on the project website.

In Fig. 6(a), we show that our method can steer the AV to make a left turn around the static obstacle while maintaining safe driving; In Fig. 6(b), we show that our method can lead the AV to avoid both static and dynamic obstacles. In particular, the AV (in green) steers to the right to avoid another vehicle coming from the opposite direction while moving away from the static obstacle; Lastly, we show that our approach can avoid multiple dynamic obstacles in Fig. 6(c), where the AV (in green) is able to make a left turn to pass a narrow space between two other vehicles.

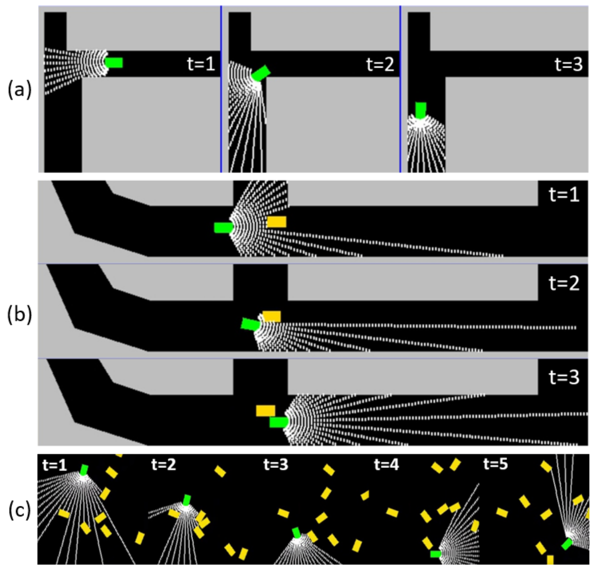


Fig. 6. (a) **Static obstacle avoidance.** Our method can lead the AV to make a left turn to avoid a static obstacle while maintaining safe driving. (b) **Static and dynamic obstacle avoidance.** The AV (in green) can avoid another vehicle (in yellow) coming from the opposite direction by steering away from the static obstacle. (c) **Collision avoidance with multiple dynamic obstacles.** Our method can direct the AV (in green) to avoid all nearby vehicles even when a narrow passage is presented.

V. Conclusion and Future Work

We propose a framework that utilizes context-aware multi-sensor perception to enable inverse reinforcement learning with hybrid-weight trust-region optimization and curriculum learning (IRL-HC) for autonomous driving. The hybrid-weight trust-region optimization empowers the network to incorporate both expert demonstrations and domain knowledge into learning. The curriculum learning enables the vehicle to perform well gradually on a series of increasingly difficult tasks. The process of learn from accidents further provides our learning agent with training data on edge cases to improve its task performance. We evaluate our approach using a variety of experiments, over the entire algorithm, and perform ablation study on the contribution of each individual component. As shown in all comparisons, our method outperforms the state-of-the-art methods on all measures.

Limitations and Future Work: First, we observe that the training results are relatively sensitive to the initial conditions. Second, our approach inherits the limitations of IRL, for example, information loss as a result of encoding expert trajectories into a single feature expectation. In the future, we hope to alleviate the information loss issue. Finally, we plan to test our approach in dense virtual traffic [36] reconstructed using real-world traffic data [37], [38], [39].

References

[1] S. Ullman, "Against direct perception," *Behavioral and Brain Sciences*, vol. 3, no. 3, pp. 373–381, 1980.

[2] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *Proceedings of the twenty-first international conference on Machine learning*, 2004, p. 1.

[3] A. Bobu, A. Bajcsy, J. F. Fisac, S. Deglurkar, and A. D. Dragan, "Quantifying hypothesis space misspecification in learning from human–robot demonstrations and physical corrections," *IEEE Transactions on Robotics*, vol. 36, no. 3, pp. 835–854, 2020.

[4] S. Narvekar, B. Peng, M. Leonetti, J. Sinapov, M. E. Taylor, and P. Stone, "Curriculum learning for reinforcement learning domains: A framework and survey," *arXiv preprint arXiv:2003.04960*, 2020.

[5] W. Li, D. Wolinski, and M. C. Lin, "ADAPS: Autonomous driving via principled simulations," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2019, pp. 7625–7631.

[6] W. Schwarting, J. Alonso-Mora, and D. Rus, "Planning and decision-making for autonomous vehicles," *Annual Review of Control, Robotics, and Autonomous Systems*, 2018.

[7] Q. Chao, H. Bi, W. Li, T. Mao, Z. Wang, M. C. Lin, and Z. Deng, "A survey on visual traffic simulation: Models, evaluations, and applications in autonomous driving," *Computer Graphics Forum*, vol. 39, no. 1, pp. 287–308, 2019.

[8] A. E. Sallab, M. Abdou, E. Perot, and S. Yogamani, "Deep reinforcement learning framework for autonomous driving," *Electronic Imaging*, vol. 2017, no. 19, pp. 70–76, 2017.

[9] C. Chen, A. Seff, A. Kornhauser, and J. Xiao, "Deepdriving: Learning affordance for direct perception in autonomous driving," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 2722–2730.

[10] M. Pfeiffer, M. Schaeuble, J. Nieto, R. Siegwart, and C. Cadena, "From perception to decision: A data-driven approach to end-to-end motion planning for autonomous ground robots," in *2017 ieee international conference on robotics and automation (icra)*. IEEE, 2017, pp. 1527–1533.

[11] F. Codevilla, M. Miiller, A. López, V. Koltun, and A. Dosovitskiy, "End-to-end driving via conditional imitation learning," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 1–9.

[12] Y. Shen, L. Zheng, M. Shu, W. Li, T. Goldstein, and M. C. Lin, "Gradient-free adversarial training against image corruption for learning-based steering," in *Thirty-fifth Conference on Neural Information Processing Systems (NeurIPS)*, 2021.

[13] M. Werling, S. Kammel, J. Ziegler, and L. Gröll, "Optimal trajectories for time-critical street scenarios using discretized terminal manifolds," *The International Journal of Robotics Research*, vol. 31, no. 3, pp. 346–359, 2012.

[14] S. Sharifzadeh, I. Chiotellis, R. Triebel, and D. Cremers, "Learning to drive using inverse reinforcement learning and deep q-networks," *arXiv preprint arXiv:1612.03653*, 2016.

[15] C. You, J. Lu, D. Filev, and P. Tsiotras, "Advanced planning for autonomous vehicles using reinforcement learning and deep inverse reinforcement learning," *Robotics and Autonomous Systems*, vol. 114, pp. 1–18, 2019.

[16] B. D. Ziebart, A. L. Maas, J. A. Bagnell, A. K. Dey, *et al.*, "Maximum entropy inverse reinforcement learning." in *Aaai*, vol. 8. Chicago, IL, USA, 2008, pp. 1433–1438.

[17] H. Le, N. Jiang, A. Agarwal, M. Dudík, Y. Yue, and H. Daumé, "Hierarchical imitation and reinforcement learning," in *International conference on machine learning*. PMLR, 2018, pp. 2917–2926.

[18] C. Finn, S. Levine, and P. Abbeel, "Guided cost learning: Deep inverse optimal control via policy optimization," in *International conference on machine learning*. PMLR, 2016, pp. 49–58.

[19] J. Ho and S. Ermon, "Generative adversarial imitation learning," *Advances in neural information processing systems*, vol. 29, pp. 4565–4573, 2016.

[20] M. Wulfmeier, P. Ondruska, and I. Posner, "Maximum entropy deep inverse reinforcement learning," *arXiv preprint arXiv:1507.04888*, 2015.

[21] J. Fu, K. Luo, and S. Levine, "Learning robust rewards with adversarial inverse reinforcement learning," *The International Conference on Learning Representations (ICLR)*, 2018.

[22] D. Brown, W. Goo, P. Nagarajan, and S. Niekum, "Extrapolating beyond suboptimal demonstrations via inverse reinforcement learning from observations," in *International conference on machine learning*. PMLR, 2019, pp. 783–792.

[23] A. Bobu, M. Wiggert, C. Tomlin, and A. D. Dragan, "Inducing structure in reward learning by learning features," *arXiv preprint arXiv:2201.07082*, 2022.

[24] X. Wang, Y. Chen, and W. Zhu, "A survey on curriculum learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[25] J. L. Elman, "Learning and development in neural networks: The importance of starting small," *Cognition*, vol. 48, no. 1, pp. 71–99, 1993.

[26] T. D. Sanger, "Neural network learning control of robot manipulators using gradually increasing task difficulty," *IEEE transactions on Robotics and Automation*, vol. 10, no. 3, pp. 323–333, 1994.

[27] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, *et al.*, "Mastering the game of go with deep neural networks and tree search," *nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[28] S. Narvekar, J. Sinapov, and P. Stone, "Autonomous task sequencing for customized curriculum design in reinforcement learning." in *IJCAI*, 2017, pp. 2536–2542.

[29] Y. Song, H. Lin, E. Kaufmann, P. Dürr, and D. Scaramuzza, "Autonomous overtaking in gran turismo sport using curriculum reinforcement learning," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 9403–9409.

[30] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, H. Zhu, H. Xiong, and Q. He, "A comprehensive survey on transfer learning," *Proceedings of the IEEE*, 2020.

[31] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015, pp. 91–99.

[32] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. L. Waslander, "Joint 3d proposal generation and object detection from view aggregation," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 1–8.

[33] A. R. Conn, K. Scheinberg, and L. N. Vicente, *Introduction to derivative-free optimization*. SIAM, 2009.

[34] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, *et al.*, "End to end learning for self-driving cars," *arXiv preprint arXiv:1604.07316*, 2016.

[35] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *International conference on machine learning*. PMLR, 2015, pp. 1889–1897.

[36] D. Wilkie, J. Sewall, W. Li, and M. C. Lin, "Virtualized traffic at metropolitan scales," *Frontiers in Robotics and AI*, vol. 2, p. 11, 2015.

[37] W. Li, D. Nie, D. Wilkie, and M. C. Lin, "Citywide estimation of traffic dynamics via sparse GPS traces," *IEEE Intelligent Transportation Systems Magazine*, vol. 9, no. 3, pp. 100–113, 2017.

[38] W. Li, D. Wolinski, and M. C. Lin, "City-scale traffic animation using statistical learning and metamodel-based optimization," *ACM Trans. Graph.*, vol. 36, no. 6, pp. 200:1–200:12, Nov. 2017.

[39] L. Lin, W. Li, H. Bi, and L. Qin, "Vehicle trajectory prediction using LSTMs with spatial-temporal attention mechanisms," *IEEE Intelligent Transportation Systems Magazine*, 2021.