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

Yu Shen, Weizi Li, and Ming C. Lin

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

Contents

Motivation and Background

Autonomous driving

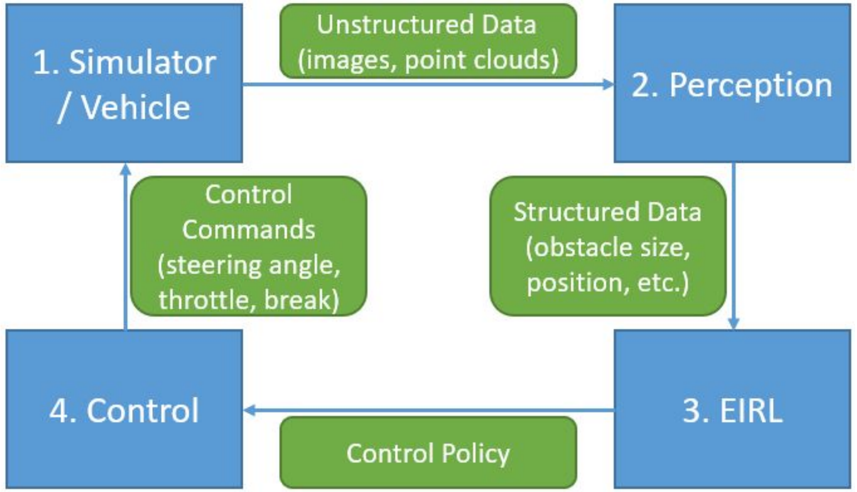
- A technology that can make vehicles drive safely to the goal without human.
- Improve safety, efficiency, etc.

Our Contributions

- **A complete architecture**, including context-aware multi-view, multi-sensor perception and enhanced inverse reinforcement learning (EIRL), that has both high compatibility and efficiency
- **A context-aware perception algorithm** based on AVOD to extract task-relevant information for multi-view, multi-sensor 3D perception
- **A enhanced IRL algorithm (EIRL)** that can utilize non-uniform prior, reuse the model parameters for continuous training, and adapt the "learning from accidents" concept by using expert demonstration & additional simulation data
- **An autonomous driving platform** that encompasses realistic scenes in Unity, various functions like virtual sensors and data collectors, and various architectures such as end-to-end learning and perception-based planning

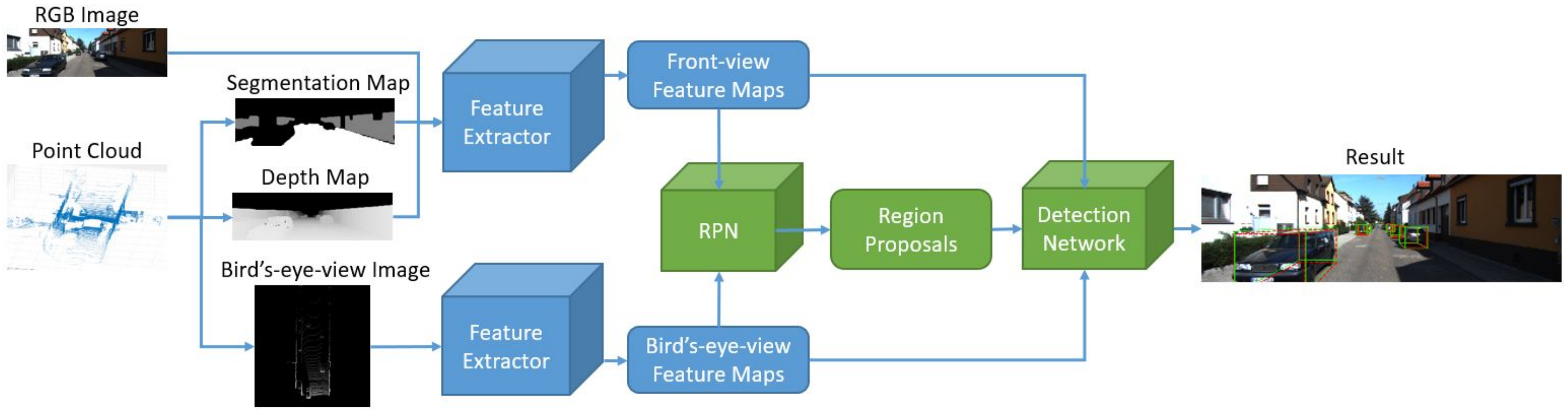
Overall Pipeline

- At each time step, the 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 to control the vehicle.


1. Simulator / Vehicle
Unstructured Data (images, point clouds)
2. Perception
Structured Data (obstacle size, position, etc.)
3. EIRL
Control Policy
4. Control
Control Commands (steering angle, throttle, break)

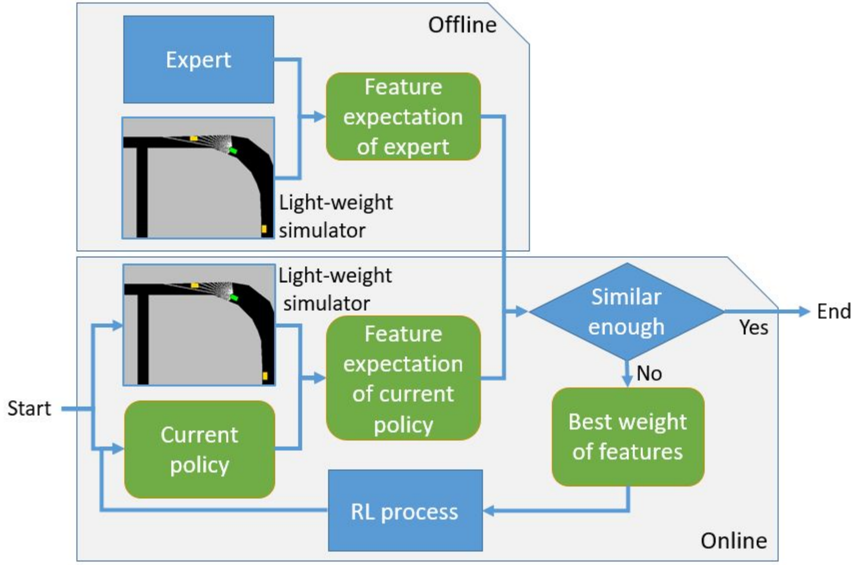
Perception Pipeline

- Extract the *context-aware* segmentation map, depth map and bird's-eye-view image from the point cloud, pack RGB images with segmentation map & depth map as front-view information;
- Extract features from both front view and bird's-eye-view information (in blue);
- 3D objects are detected using Region Proposal Network (RPN) & detection network (in green).



EIRL Pipeline

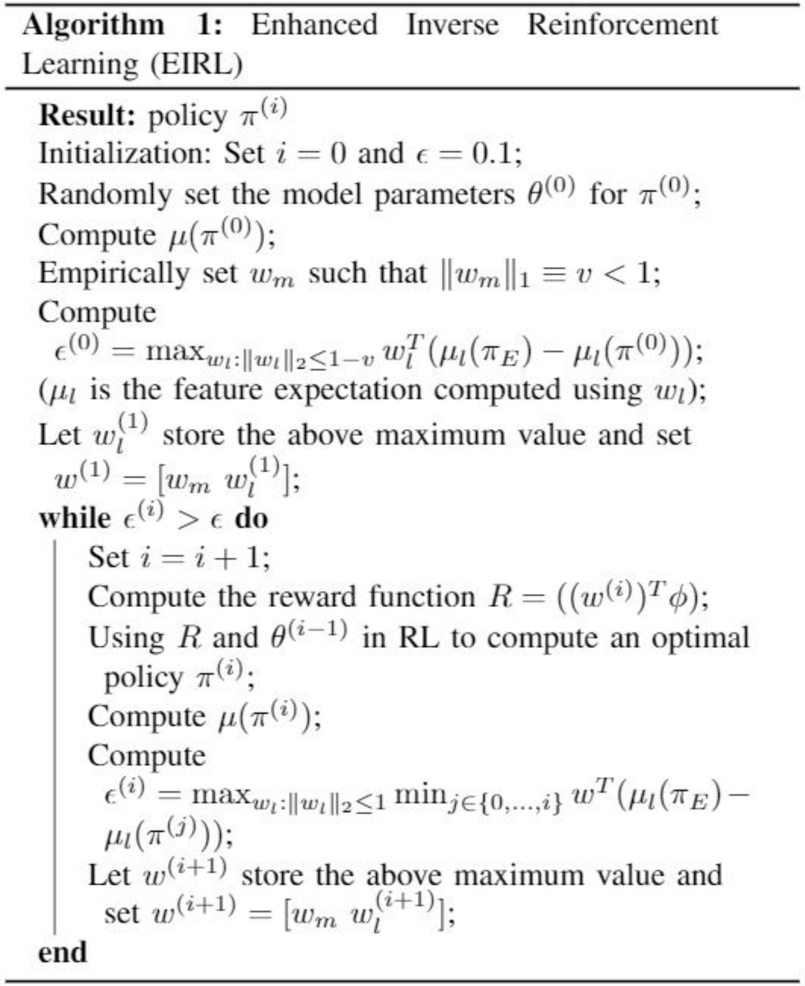
- Offline: Collect experts' trajectories, compute the feature expectation.
- Online: Compute the feature expectation of the current policy. If similar with the expert policy feature expectation then stop; or, else calculate a new reward and learn a new policy, iteratively.



EIRL Algorithm

- Non-uniform prior
- Reused model parameters
- Learning from accidents

Variable	Description
π	Policy
θ	Model parameters
μ	Feature expectation
w_m	Manually set weights
w_l	Weights to be learnt
$\phi(s)$	Features of a state

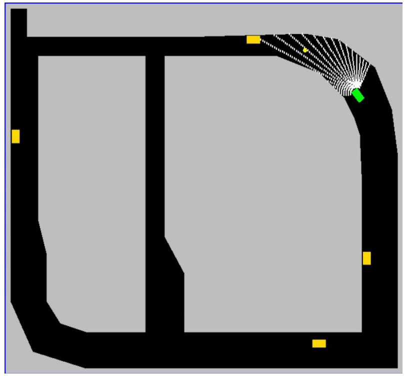


Algorithm 1: Enhanced Inverse Reinforcement Learning (EIRL)

```
Result: policy $\pi^{(i)}$
Initialization: Set $i = 0$ and $\epsilon = 0.1$;
Randomly set the model parameters $\theta^{(0)}$ for $\pi^{(0)}$;
Compute $\mu(\pi^{(0)})$;
Empirically set $w_m$ such that $\|w_m\|_1 \equiv v < 1$;
Compute
  $\epsilon^{(0)} = \max_{w_l:\|w_l\|_2 \leq 1-v} w_l^T(\mu_l(\pi_E) - \mu_l(\pi^{(0)}))$;
($\mu_l$ 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 $\mu(\pi^{(i)})$;
    Compute
      $\epsilon^{(i)} = \max_{w_l:\|w_l\|_2 \leq 1} \min_{j \in \{0,\dots,i\}} w^T(\mu_l(\pi_E) -$
      $\mu_l(\pi^{(j)}))$;
    Let $w^{(i+1)}$ store the above maximum value and
      set $w^{(i+1)} = [w_m \; w_l^{(i+1)}]$;
end
```

Autonomous Driving Platform

- Virtual Sensors: RGB camera, depth camera, Lidar, gyroscope, and GPS.
- Portable Architectures: End-to-end learning, perception plus motion planning, and perception plus learning-based planning.
- Collecting Module: Sensor data, calibration data and environment data, with different format, e.g., the KITTI dataset for the 3D object detection task.
- Others: Communication module, control module, etc.



Capital
50 North
Durant Rd.
University Ave.
1 MILE
89 East
Logan Ave.
Clearwater
1 MILE
Statistics

Audio:
Level: -74.8 dB
Clipping: 0.0%
DSP load: 0.2%
Stream load: 0.0%
Graphics:
79.6 FPS (12.6ms)
CPU: main 12.6ms render thread 7.4ms
Batches: 2434 Saved by batching: 1295
Tris: 3.7M Verts: 3.4M
Screen: 1021x543 - 6.3 MB
SetPass calls: 2130 Shadow casters: 885
Visible skinned meshes: 0 Animations: 0

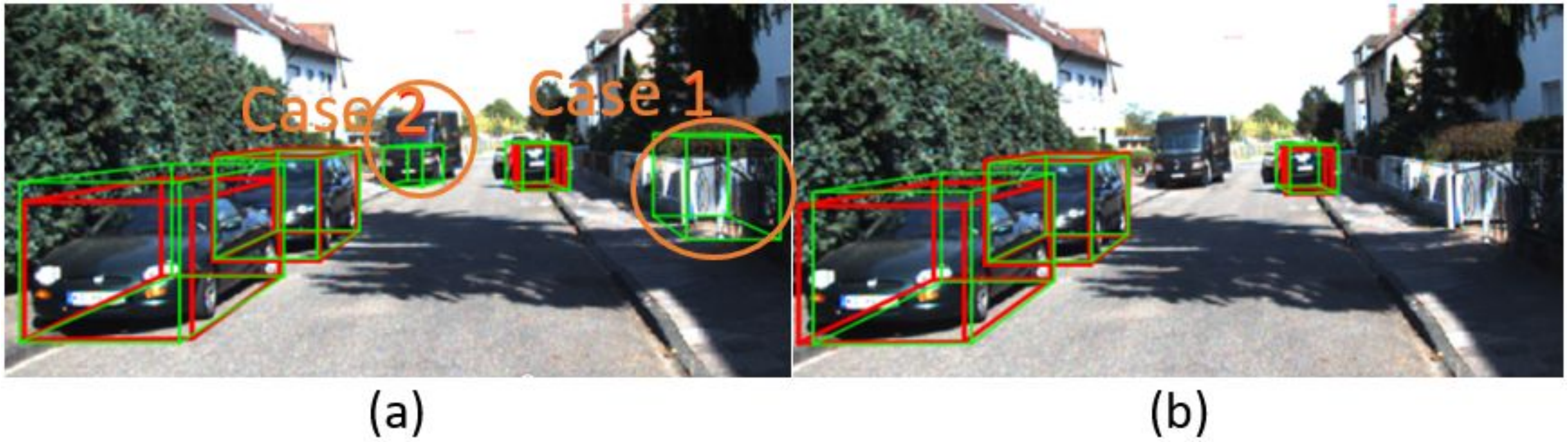
Perception Result Comparison

- Comparison of 3D Detection Models using mean Average Precision (mAP) as the metric. Our method achieves the highest scores across all difficulty levels of the detection task using KITTI. Overall, we can suppress up to 15% false positives compared with AVOD.

Model	Easy (mAP)	Moderate (mAP)	Hard (mAP)
MV3D	71.09	62.35	55.12
VoxelNet	77.47	65.11	57.73
F-PointNet	81.20	70.39	62.19
AVOD-FPN	81.94	71.88	66.38
Ours	**82.16**	**73.22**	**67.38**

Perception Case Analysis

- Detection results of (a) AVOD-FPN and (b) our method. The red bounding boxes denote ground truth while the green ones denote the detection results. Our method can suppress the false positives---case 1 and case 2---in (a) due to the use of the segmentation map and depth map.



EIRL Test Scenes

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

EIRL Result Comparison

- l_{final} shows safe trajectory length (in meters)
- s_{final} shows how many checkpoints the AV can achieve (number * 100)
- Our method not only achieves the **highest scores** but also enables the AV to *drive safely* **10x** **further** than the other methods.

Method	$l_{final,1}$	$s_{final,2}$	$l_{final,2}$	$s_{final,3}$	$l_{final,3}$
IM	105.6	77.4	53.7	60.1	44.7
RL	72.9	99.8	59.7	59.1	39.7
IRL	228.8	110.7	69.4	59.7	33.2
Ours	**276.3**	**205.8**	**748.4**	**177.3**	**324.2**

EIRL Feature Effectiveness

- *Non-uniform prior* can reduce the number of collision.

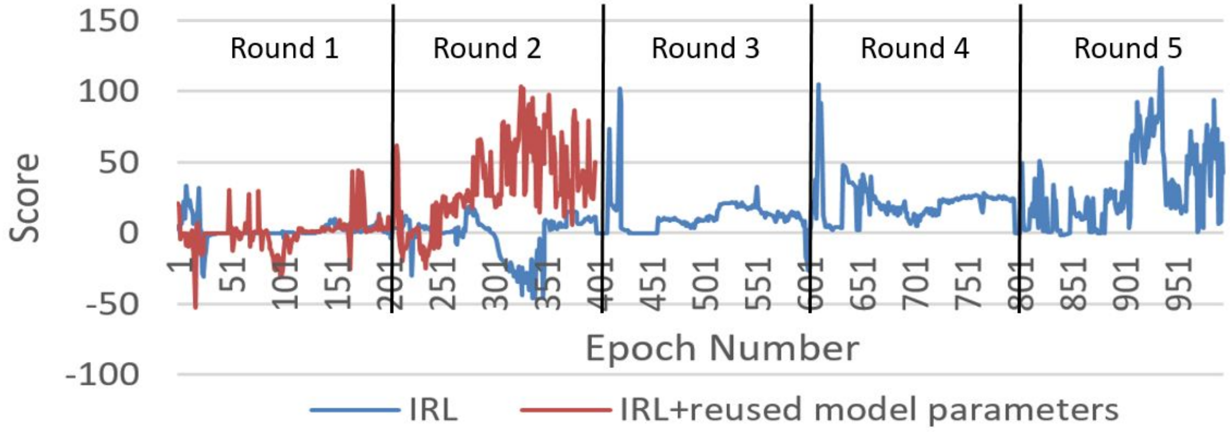
Number of collisions in three different scenes within 10,000 steps: IRL vs. EIRL (IRL+non-uniform prior). Our EIRL can reduce the number of collisions ***up to 41%***.

Model	Scene 1	Scene 2	Scene 3
IRL	35	58	111
IRL+non-uniform prior	**33**	**41**	**93**

EIRL Feature Effectiveness

- *“Reused model parameters”* can improve the training efficiency.

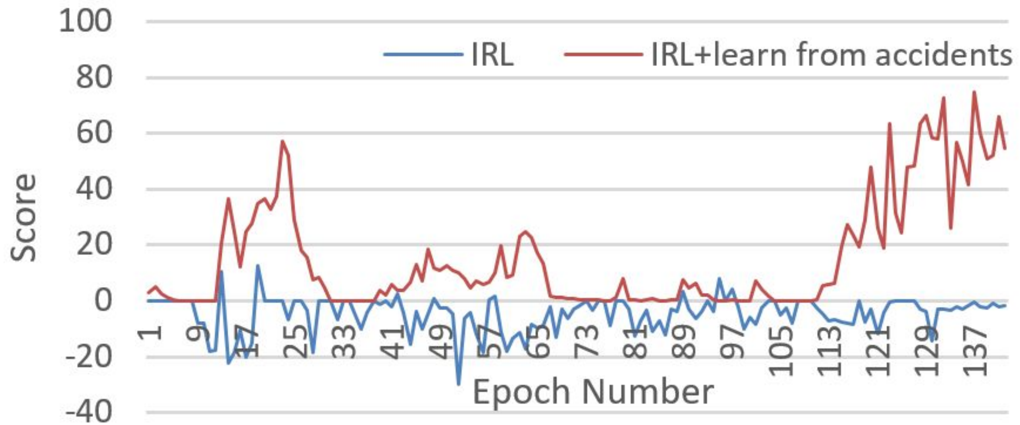
By adding this feature, we can achieve the same score after 2 rounds of training -- otherwise achieved after 5 rounds of training -- resulting in **<u>2.5x</u> speedup**



EIRL Feature Effectiveness

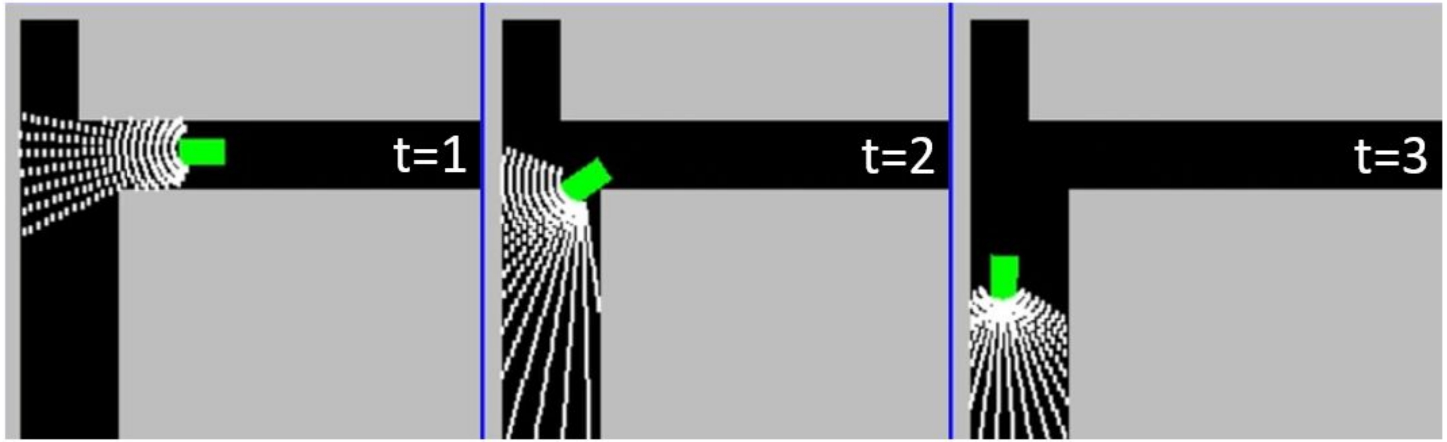
- *"Learning from accidents"* can also improve the training efficiency.

Having additional training data from this feature, the learning algorithm achieves higher scores up to *two orders of magnitude* in near collision scenarios under the same number of epochs.



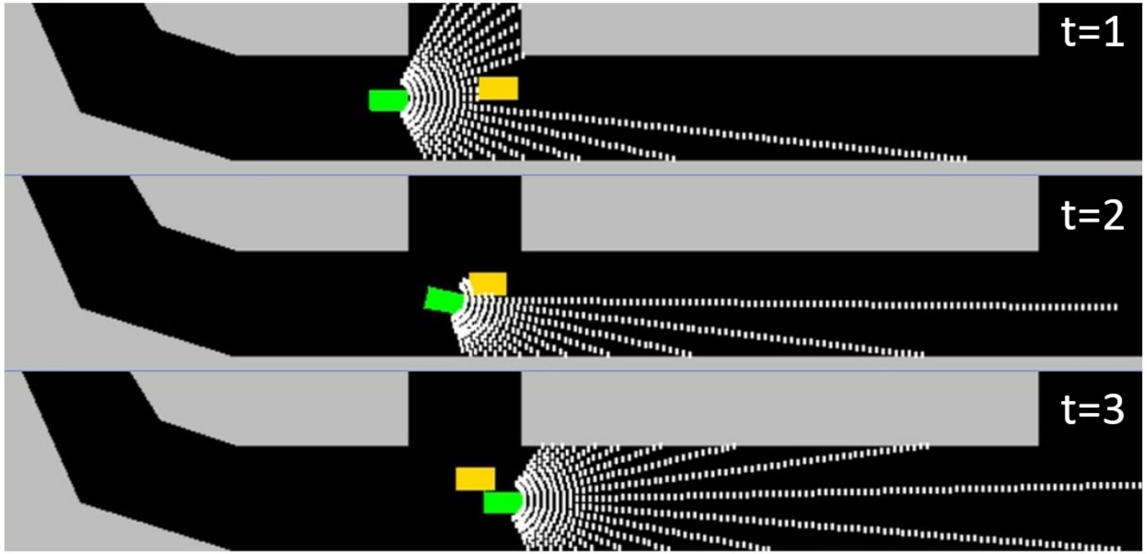
EIRL Case Analysis

- *Static obstacle avoidance*. Our method can cause the car to make a left turn for collision avoidance and resume safe driving.



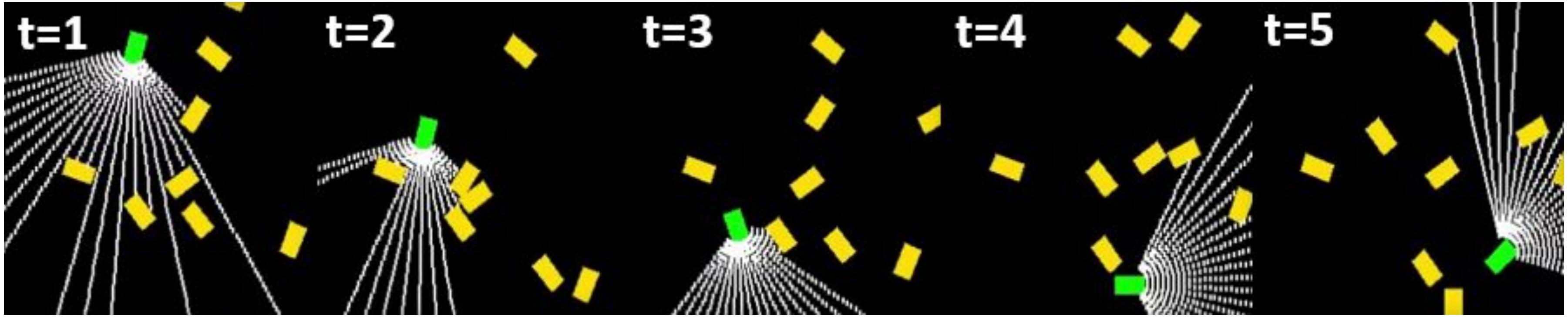
EIRL Case Analysis

- *Static and dynamic obstacle avoidance*. Our car (denoted in green) can avoid another car (denoted in yellow) coming from the opposite direction while steering away from the static obstacles.



EIRL Case Analysis

- *Collision avoidance with multiple dynamic obstacles*. Our method can direct the car (denoted in green) to avoid all nearby cars even when a narrow passage is presented.



Video Demo: Collision Avoidance in Open Space

Video Demo: Collision Avoidance on City Streets

Video Demo: Collision Avoidance on City Streets with Other Cars

Video Demo: Demo in Unity

Limitations and Future Work

- Perception: Although our perception module achieves high detection accuracy, the inference speed can be improved.

- Planning: Our approach inherits the limitations of IRL. One of them is encoding expert trajectories into one feature expectation. This process is subject to information loss.

Thank you!