Adversarial Inverse Reinforcement Learning for Decision Making in Autonomous Driving

Pin Wang\textsuperscript{1}, Dapeng Liu\textsuperscript{2,3}, Jiayu Chen\textsuperscript{4}, and Ching-Yao Chan\textsuperscript{1}

Abstract—Generative Adversarial Imitation Learning (GAIL) is an efficient way to learn sequential control strategies from demonstration. Adversarial Inverse Reinforcement Learning (AIRL) is similar to GAIL but also learns a reward function at the same time and has better training stability. In previous work, however, AIRL has mostly been demonstrated on robotic control in artificial environments. In this paper, we apply AIRL to a practical and challenging problem—the decision-making in autonomous driving, and also augment AIRL with a semantic reward to improve its performance. We use four metrics to evaluate its learning performance in a simulated driving environment. Results show that the vehicle agent can learn decent decision-making behaviors from scratch, and can reach a level of performance comparable with that of an expert. Additionally, the comparison with GAIL shows that AIRL converges faster, achieves better and more stable performance than GAIL.

I. INTRODUCTION

The application of Reinforcement Learning (RL) in robotics has been very fruitful in recent years. The application tasks range from flying inverted helicopter [1] to robot soccer [2] and other robotic manipulations [3][4]. Though the results are encouraging, one significant barrier of applying RL to real-world problems is the required definition of the reward function, which is typically unavailable or infeasible to design in practice.

Inverse Reinforcement Learning (IRL) [5] aims to tackle such problems by learning the reward function from expert demonstrations, thus avoiding reward function engineering and making good use of the collected data. Maximum entropy IRL [6][7] is commonly used, where trajectories are assumed to follow a Boltzmann distribution based on a cost function. However, because of the expansive reinforcement learning procedure in the inner loop, it has limited application in problems involving high-dimensional state and action spaces [8].

Generative Adversarial Imitation Learning (GAIL) [8] overcomes these challenges by learning a policy against a discriminator that tries to distinguish learnt actions from expert actions. It is very efficient because the policy generator and discriminator can be iteratively updated in a single loop. Guided Cost Learning (GCL) [3] tackles the problem from another direction, where importance sampling is used to estimate the partition function in the maximum entropy IRL formulation. Finn et al. [9] shows that GAIL with a properly reparameterized discriminator is mathematically equivalent to GCL with some tweaks in importance sampling. Adversarial Inverse Reinforcement Learning (AIRL) [10] combines GAIL and GCL formulation and learns the cost functions together with the policy in an adversarial way. Compared to AIRL, GAIL does not recover a cost function and may suffer from training instability issues. In the implementation of both GAIL and AIRL, the discriminator takes individual station-action pairs instead of the whole trajectories as input to reduce the high variance problem that exists in GCL. While the AIRL formulation is appealing, it has only been applied to robotic controls in OpenAI Gym environments to our knowledge.

Autonomous driving is a complicated problem as it involves extensive interactions with other vehicles in a dynamically changing environment, and this is particularly true for the decision-making task that needs to monitor the environment and decides maneuvering commands to the control module. In this paper, we apply AIRL to learn the challenging decision-making behavior in a simulated environment where each vehicle is interacting with all other vehicles in its surroundings. Our study, different from RL based studies, makes good use of demonstration data and, different from IRL based studies, learns both a reward function and a policy.

The remainder of the paper is organized as follows. A literature review is given in Section II. Section III introduces the methodology, followed by the description of the decision-making task in Section IV and experiment in Section V. Conclusion and discussions are given in the last section.

II. RELATED WORK

The applications of adversarial learning based algorithms are mostly verified with control tasks in OpenAI Gym [3][8][10]. For example, GAIL was tested by 9 control tasks which were either classic controls in OpenAI Gym or 3D controls in MuJoCo [8]. In AIRL, the experiments for testing the algorithm were simple tabular MDP, 2D point mass navigating, and a running ant [10]. In GCL, the algorithm was verified in both simulated and physical robotic arms, but the experimenting environment was stationary and there was no interaction with the environment [3]. Among

---

1 P. Wang and C. Chan are with California PATH, University of California, Berkeley, Richmond, CA, 94804, USA. \{pin.wang, cychan\}@berkeley.edu.
2,3 D. Liu is with Zenunity AB and Chalmers University of Technology, Gothenburg, Sweden. dapeng.liu@zenunity.com
4 J. Chen is with College of Engineering, Peking University, Beijing, China. 1600011060@pku.edu.cn
these algorithms, AIRL has shown robust and superior performance than GAIL and GCL methods for applications where the dynamics of environment are changed. Singh et al. [4] apply adversarial learning to real-world robotic control tasks like placing books and arranging objects without reward crafting, and the learning is based on a fraction of state space with labels instead of entire expert demonstration trajectories.

The decision-making task in our use case of lane changes is much more complex than the OpenAI Gym-based experiments implemented in the original papers of [3][8][10]. First, the decision-making task involves more complicated factors as the goal is not just to change to the target lane, it also involves other criteria such as safety (e.g. not crash into other objects during the changing), efficiency (e.g. not be too cautious to take much time in changing), and comfort (e.g. not bring unsmooth travelling experiences. In contrast, the goal in the implemented OpenAI gyms in [3] and [10] is clear and simple, as to reaching a goal position or velocity. Second, different from the stationary background environment in the OpenAI gyms, the driving environment we are experimenting with is dynamically changing which creates diverse driving situations and complicates the rewarding mechanism, making the learning much more challenging. Last but not the least, the decision-making task itself is a complicated problem as it is entangled with vehicle control. A decision impacts the control action at the next step and the control action in return affects the selection of the subsequent decision. The coupled effect makes the decision-making problem challenging.

In autonomous driving, many recent studies have applied RL to the vehicle control task [11][12][13]. Some studies began to apply it to decision making [14][15]. Only a few studies have applied adversarial learning to driving. One work was done by Kuefler et al. [16]. They applied GAIL to learn lane-keeping task in a driving simulator, with a recurrent neural network as the policy network. The task is relatively simple as lane keeping requires less interactions with road users. To our best knowledge, no prior work has applied AIRL to practical tasks in autonomous driving that needs to deal with interactions with surrounding vehicles. Our work is to explore its feasibility for handling the challenging decision-making task.

III. METHODOLOGY

AIRL is directly related with maximum entropy IRL [7] and Guided Cost Learning [3]. It uses a special form of the discriminator different from that used in GAIL, and recovers a cost function and a policy simultaneously as that in GCL but in an adversarial way. In this section, we give the details of AIRL and the augmented AIRL that we adapted in this paper.

A. Preliminaries

Maximum entropy inverse reinforcement learning: Maximum entropy Inverse Reinforcement Learning models the distribution of the demonstrated behaviors with a Boltzmann distribution [6]

\[ p_\theta(\tau) = \frac{1}{Z} \exp(-c_\theta(\tau)) , \]

where \( \tau = (s_0, a_0, ..., s_T, a_T) \) is a behavior sequence from the demonstrated data; \( c_\theta(\tau) = \sum_t c_\theta(s_t, a_t) \) is the unknown cost function parameterized by \( \theta \) that we want to learn, and \( Z \) is the partition function which is the integral of over all possible trajectories from the environment and the estimation of this term lays the barrier of applying Maximum entropy IRL to high-dimensional or continuous problems.

The goal of IRL is to find the cost function by solving a maximum likelihood problem

\[ \max_{\theta} \mathbb{E}_{\tau \sim D}[\log p_\theta(\tau)] , \]

where \( D \) is the dataset of demonstrations. Among research which focus on solving the estimation of \( Z \) [6][17][18], Finn et al. [3] proposed an importance sampling based method, Guided Cost Learning, to estimate \( Z \) for high-dimensional and continuous problems.

Guided cost learning: Guided Cost Learning introduces a new sampling distribution \( q(\tau) \) to generate samples for the estimation of \( Z \). It is verified that \( q(\tau) \), at optimality, is proportional to the Maximum entropy IRL based distribution: \( q(\tau) \propto \exp(-c_\theta(\tau)) \) \( \Rightarrow \exp(-c_\theta(\tau)) \), therefore, the objective of the cost learning can be reformulated as

\[
\mathcal{L}_{\text{cost}}(\theta) = \mathbb{E}_{\tau \sim p}[ -\log p_\theta(\tau) ] - \mathbb{E}_{\tau \sim p}[c_\theta(\tau)] + \log(Z) \\
= \mathbb{E}_{\tau \sim p}[c_\theta(\tau)] + \log(\mathbb{E}_{\tau \sim q}\frac{\exp(c_\theta(\tau))}{q(\tau)}).
\]

The update of the importance sampling distribution is a policy optimization procedure with the goal to minimizing the KL divergence between \( q(\tau) \) and the Boltzmann distribution \( \frac{1}{Z} \exp(-c_\theta(\tau)) \), which results in an objective of minimizing the learned cost function and maximizing the distribution’s entropy

\[
\mathcal{L}_{\text{sample}}(q) = \mathbb{E}_{\tau \sim q}[c_\theta(\tau)] - \mathbb{E}_{\tau \sim q}[\log q(\tau)].
\]

The cost learning step and policy optimization step alternate until convergence is reached.

B. Adversarial Inverse Reinforcement Learning

The form of discriminator: To be in consistent with the previous section, we still use \( p(\tau) \) to denote the true distribution of the demonstration, and \( q(\tau) \) is the generator’s density. Based on the mathematical proof in [9], a new form of discriminator can be designed as in [3] where \( p(\tau) \) is estimated by the Maximum entropy IRL distribution.

\[
D_\theta(\tau) = \frac{\frac{1}{Z} \exp(-c_\theta(\tau))}{\frac{1}{Z} \exp(-c_\theta(\tau)) + q(\tau)}
\] (2)

The optimal solution of this form of discriminator is independent of the generator, which improves the training stability that we will show later in our training results.
The use of the full trajectory in the cost calculation could result in high variance. Instead, a revised version of the discriminator based on individual state-action pairs is used in [10] to remedy this issue. By subtracting a constant $\log Z$ from the cost function, the discriminator can be changed to the following form as in [5] where $p(\tau)$ is equivalently replaced with $\pi(a|s)$.

$$D_\theta(s, a) = \frac{\exp(f_\theta(s, a))}{\exp(f_\theta(s, a)) + \pi(a|s)} \tag{3}$$

**Objectives of Discriminator and Generator:** The objective of the discriminator is to distinguish whether the state-action pair is from the expert or is generated. This corresponds to the sum of two terms as

$$\mathcal{L}(D) = \mathbb{E}_{(s,a) \sim p}[\log D(s, a)] + \mathbb{E}_{(s,a) \sim G}[\log(1 - D(s, a))]. \tag{4}$$

The generator’s task to produce samples to fool the discriminator, i.e. to minimize its log probability of being classified as generated samples. This signal alone is not enough to train the generator when the discriminator fast learns to distinguish between generated data and expert data. Therefore, another part, interpreted as the discriminator’s confusion [9], is added to the generator’s loss as

$$\mathcal{L}(G) = \mathbb{E}_{(s,a) \sim p}[\log D(s, a)] + \mathbb{E}_{(s,a) \sim G}[\log(1 - D(s, a))]. \tag{5}$$

To convert it into a RL formulation, we obtain the following form as the generator’s objective

$$G = \mathbb{E}_{(s,a) \sim G}[log D(s, a) - \log(1 - D(s, a))]. \tag{6}$$

The reward function can be extracted as in [7] and is used in our policy optimization.

$$R = \log(D(s, a)) - \log(1 - D(s, a)) \tag{7}$$

**Optimization:** The optimization of the discriminator and generator is similar to the idea of GCL in which the cost learning interleaves with the policy optimization procedure.

The optimization of discriminator is based on [4] as a binary logistic regression problem. The updated discriminator is fed to the reward function [7] for policy optimization. In GCL, the policy is optimized based on guided policy search [19], a model-based method for unknown dynamics in which it iteratively fits a time-varying linear dynamic model to the generated samples [3]. As the dynamic driving environment is too complicated to learn for the driving task, we instead use a model-free policy optimization method, Trust Region Policy Optimization (TRPO) [20], to update the policy by using the updated reward function.

**C. Augmented AIRL**

In AIRL, the reward function is formulated purely in the view of the adversarial learning theory and learns the intrinsic rewarding mechanism embedded in the demonstrated behaviors. If we can apply some domain knowledge and augment the learned reward function with a semantic but sparse reward signal, it should provide the agent some informative guidance and assist it to learn fast. Based on this insight, we augment the reward function with intuitively defined semantic rewards.

**Semantic Reward:** The semantic reward should be easily defined and obtained. In our study case, a clear goal is to successfully complete the lane change and not to collide with other objects. Additionally, we encourage the agent to initiate the lane-change as soon as possible once it receives the lane-change command, i.e. select the closest gap and commit lane change immediately, which corresponds to action 2 according to our definition (the definition of the action space is referred to Section IV along with further explanations of the decision-making task). Therefore, we define the semantic reward as follows.

$$r_{sem} = \begin{cases} 
100 & \text{if success} \\
-300 & \text{if crash} \\
3 & \text{if action = 2} \\
0 & \text{otherwise} 
\end{cases} \tag{8}$$

We add the semantic term to the generator reward function as follows.

$$R(s, a) = \log(D_\theta(s, a)) - \log(1 - D_\theta(s, a)) + r_{sem}(s, a) \tag{9}$$

This treatment is equivalent to adding $r_{sem}(s, a)$ to $c_\theta(s, a)$ for policy updates as it can be proved that

$$R(s, a) = \log(D_\theta(s, a)) - \log(1 - D_\theta(s, a)) + r_{sem}(s, a) = \log \frac{D_\theta(s, a)}{1 - D_\theta(s, a)} + r_{sem}(s, a)$$

$$= \log \frac{e^{c_\theta(s, a)}}{\pi(a|s)} + r_{sem}(s, a)$$

$$= c_\theta(s, a) + r_{sem}(s, a) - \log \pi(a|s).$$

Actually, when using this form of reward to optimize the policy, we are learning an entropy-regularized policy which maximizes the accumulated reward and also captures the most stochasticity of the policy. Such features make the policy more robust to system noises. Thus the learned reward function characterizes the demonstrated behavior and can be potentially applied to other environments, and the learned policy can be used to generate similar decision-making strategies as demonstrated by experts.

The core of the training procedure of the augmented AIRL method is in Algorithm 1.

**IV. Decision-making Task**

Decision-making is a broad term and the definition of the task may vary in different applications. In our study, the decision-making task is defined to select an appropriate gap on the target lane and decide whether to merge into it at the
current step, when a lane change requirement (e.g. "changing to the left/right lane") is issued by a separate module at the strategic level. In other words, the task of our focus includes a longitudinal decision – the selection of a target gap, and a lateral decision – whether to commit the lane change right now. After the decision is made, a low-level controller will calculate corresponding control commands to execute the decision.

To apply AIRL to learn the decision making behavior, we need to design reasonable action space and state space, and make their distributions representative and distinguishable. Details about the action space and state space are described in the next subsections.

### A. Action Space

The longitudinal decision corresponds to the target gap selection. Fig. [1] demonstrates a typical driving situation with candidate gaps that the ego vehicle may choose when it is to execute the command of left turn. Gap 0, gap 1, gap 2 and gap 3 refer to the front gap, alongside gap, back gap and the current-lane gap, respectively. The decision on the lateral movement is defined as a binary decision task, i.e. whether to execute the lateral movement of the lane-change behavior at the current step.

In combination, there are 8 action pairs. When practical situations are taken into account, some combinations are not preferable. Take the action pair (0, 1) for example, if gap 0 is selected, the lateral decision should not be 1 (i.e. not to take the action of “committing lane change right now”) as it has to adjust its longitudinal movement first to position itself at a proper location. Similarly, the combination of (2, 1) and (3, 1) should also be avoided in order to reduce the agent’s unnecessary exploration.

To make it simple, we use an one-dimensional vector to represent the five possible action tuples. The mapping is 
\[[(0, 0), (1, 0), (1, 1), (2, 0), (3, 0)] \rightarrow [0, 1, 2, 3, 4] \]

### B. State Space

In our daily driving, normally we only pay attention to certain parts of the scene that relates to our driving task rather than to all the information in the observable field of view. This subconscious act helps lower our burden of scene understanding and makes us focus more on the most irrelevant information. Similar idea should be applied to the learning agent. Instead of providing all the available information that can be gathered by sensors, we choose to use observations that are only relevant to the decision-making task. That is, the state space in our study only includes features from relevant vehicles, i.e. vehicles \( V_0, V_1, V_2, V_3, V_4 \) from gap 0 to gap 3, as shown in Fig. [1]. In this study, we use observations and states interchangeably, and do not consider hidden states.

For each relevant vehicle, we gather vehicle kinematics with what we can derive with on-board sensors, which includes vehicle speed, acceleration, position, lane id, vehicle id, etc. For the ego vehicle, we also gather similar information plus a target lane id. In total, there are 44 features.

### C. Expert Data

The demonstrations can be data collected from real-world driving or from a well-developed simulator. Since the training involves data generation by current policy, for consistency, we use a driving simulator to collect the demonstration data. The training can be similarly carried out with real-world driving data from physical systems when available.

Generating expert behaviour is not a straightforward task. If rules can be formulated to produce reasonable behaviours in all random traffic scenarios, there is little need for learning methods as rule-based solutions will be adequate. Instead of defining explicit rules for all possible situations, in this work we generate expert data by imposing criteria that will lead to safe, efficient, and smooth driving maneuvers.

The primary concern is safety, namely, no crash is introduced by the lane-change vehicle’s behaviour. The safety in simulation is ensured, on one hand, by exploiting the natures of behaviour models such as Intelligent Driver Model [21], and on the other hand, by implementing an enforced safety threshold. Another point is efficiency, i.e., arriving at the target lane as fast as possible. By calculating a dummy time point of the completion of a lane-change task based on the current kinematics of both the lane-changing vehicle and

---

**Algorithm 1 Augmented AIRL**

1. Collect expert lane-change sequences \( \tau_E \)
2. Initialize discriminator \( D_\theta \), decision-making policy \( \pi_\phi \)
3. for iteration \( i = 1 \) to \( N \) do
4. Collect decision-making sequences \( \tau_i = (s_0, a_0, ..., a_{T-1}, a_{T-1}) \) with current policy \( \pi_{\phi_i} \)
5. Train \( D_\theta \) based on \( \tau_i \) with data drawn equally from \( \tau_E \) and \( \tau_i \) for \( m \) iterations
6. Update reward function \( R \) using \( \tau_i \) with \( D_\theta \) and semantic reward \( r_{sem} \)
7. Train policy \( \pi_\phi \) w.r.t. \( R \) using TRPO for \( n \) iterations
8. end for

---

![Fig. 1. Illustration of candidate gaps and vehicles in a lane-change scenario](image)
interactive vehicles, we can simulate behaviors that pursue driving efficiency. Comfort is also considered by minimizing the jerk for both the lane-changing vehicle and vehicles that the lane-changing vehicle interacts with.

With these practical considerations implemented in the simulator, the expert data we gather from simulation can represent reasonable driving behaviors under diverse driving situations.

V. EXPERIMENT

In our experiment, we evaluate the performance of AIRL and augmented AIRL on the decision-making task and compare the results with GAIL and expert data in simulated highway driving scenarios.

A. Simulation Environment

The simulation environment is a highway segment of 500 meters long and three-lane wide on each direction. The traffic is moderately dense in which the headways among adjacent vehicles are about 1-3 seconds, creating diverse challenging situations. Vehicles are generated randomly on all the lanes and they can perform car-following behaviors modeled with an adapted Intelligent Driver Model [22]. The ego vehicle initially enters on the middle lane and receives left or right lane-change command after entering the road for around 50 meters. Once it receives the command, the ego vehicle starts the decision-making function modeled by the aforementioned adversarial algorithms. After a decision is made, a low-level controller, consisting of a PID [21] based lateral control and sliding-mode [23] based longitudinal control, is activated to execute the decision. In our study, we allow the agent to perform the decision-making at a frequency of 10 Hz and we do not involve a planning model for near future trajectory planning. A demonstration of the simulated environment is shown in Fig. 2.

The simulator embraces many features. For example, other vehicles can yield to or overpass the lane-changing vehicle, and the ego vehicle can also adapt its own speed to merge into gaps before or behind it. These features enrich the diversity of vehicle behaviors, and make the driving environment very much like a real-world setting. With this simulator, we can learn interactive decision-making behaviors.

B. Evaluation Metrics

The selection of proper evaluation metrics is important to assess the performance of the algorithms. Simply using the accumulated reward and episode length (as in [3] and [10]) is not enough for evaluating the complicated lane-change behavior. In our study, we propose to use four metrics. One is the successful lane change ratio calculated with the number of successful lane change cases and the total number of the lane change cases. Another metric is the decision-making steps that a lane change process consumes. It is counted from the moment the lane change command is issued to the time when the lane change process finishes. Also, we gather the changing-lane steps which is counted from the start of lateral movement to the completion of the lane change process. The last metric is the total reward accumulated during the lane-changing procedure. The reward value is obtained from a reward function that contains multiple components that we defined in the simulator. It is used as a criterion for evaluating the performance of different algorithms.

C. Training Results

The neural network architecture of the discriminator and generator for the three algorithms (i.e. GAIL, AIRL, AugAIRL) are the same: two hidden layers of 100 units each for the generator, and two hidden layers of 512 units each for the discriminators. We use the same amount of expert data (5000 lane-change episodes) in the three trainings. Fig. 3 depicts the training curves of the four metrics over around 15,000 iterations. Table I gives the exact results of the three algorithms from the last saved models.

As shown in Fig. 3, the metric curves from the three adversarial based methods all show convergence. In particular, the AIRL based methods demonstrate better performance and converge faster than GAIL in all four metrics. It is also noticeable that the training by both AIRL methods are more stable than that by GAIL, proving the effectiveness of the special form of the discriminator.

More interesting results can be obtained from exploring the curves of AIRLs. From the successful ratio curves we can observe that the agent completes lane changes with a quite high successful rate, approaching to the expert performance of 1.0, which is very appealing since the agent learns the decision-making strategy from scratch.

The values of the decision-making steps and lane-changing steps of AIRL methods indicate that the lane change behavior at convergence stays quite stable and consumes similar maneuvering time to that of the expert. Additionally, from these two curves, we can infer that at the beginning of the training, the agent is quite conservative and hesitates to commit lane change as the decision steps are high and the actual lane-changing steps are low. As training goes on, the agent begins to explore and imitates the demonstrated behaviors.

From the reward curves, we can observe that the accumulated rewards at convergence, particularly for the augmented AIRL, are around 20 and quite close to the expert value of

![Fig. 2. Demonstrations of decision-making in lane change situations](image-url)
TABLE I
Training results in four evaluation metrics from the last saved models

| Algorithms    | Total Reward | Success Ratio | Decision Steps | Changing Steps | Discrim. Loss |
|---------------|--------------|---------------|----------------|----------------|---------------|
|               | Mean         | Std           | Mean           | Std            | Mean          | Std            | Mean          | Std            | Mean          | Std            |
| GAIL          | 15.67        | 5.78          | 0.955          | 0.071          | 77.44         | 12.82         | 55.57         | 3.6            | –             | –             |
| AIRL          | 17.58        | 4.58          | 0.99           | 0.01           | 67.54         | 8.41          | 52.53         | 3.56           | 0.694         | 0.032         |
| AIRL+SPARSE   | 18.87        | 2.81          | 0.99           | 0.01           | 64.30         | 5.66          | 56.31         | 3.36           | 0.625         | 0.035         |
| Expert        | 24.21        | –             | 1.00           | –              | 67.74         | –             | 58.18         | –              | –             | –              |

Fig. 3. Training results of GAIL, AIRL and Augmented AIRL, in comparison with experts in four metrics over 15k iterations

24.

The training performance of the augmented AIRL is slightly better than AIRL when we zoom in to look into the details of the data in Table I that shows the results of the last saved model at 15,000 iterations. The values of total rewards, success ratio, and changing steps from the augmented AIRL are much more closer to the experts’ performance, and their standard deviations are smaller than AIRL as well as GAIL. In general the improvement is consistent. The next could be to explore other forms of semantic reward to provide more intuitive guidance to the agent.

D. Validation Results

We conduct testing on 5 saved checkpoints during training. For each checkpoint, we run 50 episodes and average their values. Fig. 4 shows the testing results of the four metrics with expert data plotted in dashed lines.

As shown in the Fig. 4 the testing results are generally consistent with the training results that AIRL methods perform better and are more stable than GAIL. Additionally, we can see more clearly that at testing time the augmented AIRL method shows more satisfactory results. Its total reward is relatively higher, its success ratio approaches 1.0 faster, and its decision-making steps and lane-changing steps are lower which means it completes the task faster.

VI. Conclusions and Discussions

In this paper, we applied Adversarial Inverse Reinforcement Learning (AIRL) methods to the challenging decision-making task in autonomous driving, and augmented the learned reward with a semantic reward term to improve learning. The results were compared with that of the Generative Adversarial Imitation Learning (GAIL) and the expert data. Comparison shows that the performance of AIRL and augmented AIRL, in terms of the four metrics (i.e. total reward, success ratio, decision-making steps and lane-changing steps) is better than that of GAIL, and is in comparable level with that of the experts. This verifies the effectiveness of the AIRL method, and the slightly but consistently improved performance of augmented AIRL also proves effectiveness of adding the semantic reward term. The results also indicate the feasibility and capability of learning both a reward function and a policy simultaneously from the demonstrated behaviors.

Designing different forms of semantic reward and adding it to different parts in the learning framework are worth trying to gain more significant improvement in performance. Also, we will extend the current work to multiple driving style learning where the demonstrations include multiple driving behaviors.
REFERENCES

[1] A. Y. Ng, A. Coates, M. Diel, V. Ganapathi, J. Schulte, B. Tse, E. Berger, and E. Liang, “Autonomous inverted helicopter flight via reinforcement learning,” in Experimental robotics IX. Springer, 2006, pp. 363–372.

[2] M. Riedmiller, T. Gabel, R. Hafner, and S. Lange, “Reinforcement learning for robot soccer,” Autonomous Robots, vol. 27, no. 1, pp. 55–73, 2009.

[3] C. Finn, S. Levine, and P. Abbeel, “Guided cost learning; Deep inverse optimal control via policy optimization,” in International Conference on Machine Learning, 2016, pp. 49–58.

[4] A. Singh, L. Yang, K. Hartikainen, C. Finn, and S. Levine, “End-to-end robotic reinforcement learning without reward engineering,” arXiv preprint arXiv:1904.07854, 2019.

[5] A. Y. Ng, S. J. Russell et al., “Algorithms for inverse reinforcement learning.” in ICML, vol. 1, 2000, p. 2.

[6] B. D. Ziebart, A. Maas, J. A. Bagnell, and A. K. Dey, “Maximum entropy inverse reinforcement learning,” 2008.

[7] B. D. Ziebart, “Modeling purposeful adaptive behavior with the principle of maximum causal entropy,” Ph.D. dissertation, figshare, 2010.

[8] J. Ho and S. Ermon, “Generative adversarial imitation learning,” in Advances in neural information processing systems, 2016, pp. 4565–4573.

[9] C. Finn, P. Christiano, P. Abbeel, and S. Levine, “A connection between generative adversarial networks, inverse reinforcement learning, and energy-based models,” arXiv preprint arXiv:1611.03852, 2016.

[10] J. Fu, K. Luo, and S. Levine, “Learning robust rewards with adversarial inverse reinforcement learning,” arXiv preprint arXiv:1710.11248, 2017.

[11] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, “Continuous control with deep reinforcement learning,” arXiv preprint arXiv:1509.02971, 2015.

[12] M. Kuderer, S. Gulati, and W. Burgard, “Learning driving styles for autonomous vehicles from demonstration,” in 2015 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2015, pp. 2641–2646.

[13] P. Wang and C.-Y. Chan, “Automated driving maneuvers under inter-active environment based on deep reinforcement learning,” Tech. Rep., 2019.

[14] S. Shalev-Shwartz, S. Shammah, and A. Shashua, “Safe, multi-agent, reinforcement learning for autonomous driving,” arXiv preprint arXiv:1610.03295, 2016.

[15] P. Wang, C.-Y. Chan, and A. de La Fortelle, “A reinforcement learning based approach for automated lane change maneuvers,” in 2018 IEEE Intelligent Vehicles Symposium (IV). IEEE, 2018, pp. 1379–1384.

[16] A. Kuefler, J. Morton, T. Wheeler, and M. Kochenderfer, “Imitating driver behavior with generative adversarial networks,” in 2017 IEEE Intelligent Vehicles Symposium (IV). IEEE, 2017, pp. 204–211.

[17] S. Levine and V. Koltun, “Continuous inverse optimal control with locally optimal examples,” arXiv preprint arXiv:1206.4617, 2012.

[18] D.-A. Huang and K. M. Kitani, “Action-reaction: Forecasting the dynamics of human interaction,” in European Conference on Computer Vision. Springer, 2014, pp. 489–504.

[19] S. Levine and V. Koltun, “Guided policy search,” in International Conference on Machine Learning, 2013, pp. 1–9.

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

[21] K. J. Åström and T. Hägglund, PID controllers: theory, design, and tuning. Instrument society of America Research Triangle Park, NC, 1995, vol. 2.

[22] M. Treiber, A. Hennecke, and D. Helbing, “Congested traffic states in empirical observations and microscopic simulations,” Physical review E, vol. 62, no. 2, p. 1805, 2000.

[23] C. Edwards and S. Spurgeon, Sliding mode control: theory and applications. Crc Press, 1998.