Preprints of the 21st IFAC World Congress (Virtual)
Berlin, Germany, July 12-17, 2020

Motion Planning at Intersections with Event-driven Recurrent Q-Learning

Xinze Jin * **Qing-Shan Jia** * **Dongchun Ren** ** **Yu Bai** **

* *CFINS, Dept. of Automation, BNRist, Tsinghua Univ. (email: jxz18@mails.tsinghua.edu.cn, jiaqs@tsinghua.edu.cn)*
** *Meituan-Dianping Group (email: {rendongchun, baiyu12}@meituan.com)*

Abstract: Autonomous driving at intersection has great potential on control for smart cities to relieve the energy consumption and transportation congestion. However, it remains challenging to find promising behavior sequence in multi-agent environment with uncertain participation of obstacles. This work develops Event-driven Recurrent Q-Learning (ERQL) to focus on the motion planning task towards intersection scenarios to conclude a sample path with safety and efficiency. We elaborate the definition of events to capture the environment structure and introduce recurrency to process sequence model. Besides, we incorporate collision-avoidance into the event-driven framework and design a mechanism to extract recurrent feature from replay buffer in Q-learning framework. Simulation results show that the developed off-line learning procedure can adapt to on-line decision making towards uncertain agent behaviors.

Keywords: motion planning, reinforcement learning, event-driven, recurrency

1. INTRODUCTION

Autonomous driving systems are built upon a modular pipeline consisting of perception, localization, decision making and motion planning, see Levinson et al. (2011) and Geiger et al. (2012). Among them, decision making module deals with the list of surrounding objects and traffic participants extracted from upstream and plan a dynamic trajectory with high uncertainty in environment and target. Motion planning for autonomous vehicles is challenging due to the influence of surrounding traffic participants and the conflicting criteria between efficiency and safety.

Autonomous vehicles are currently being tested in various scenarios. Leading companies like Waymo, Momenta, etc. have deployed business in highways and routine urban roads. However, motion planning towards intersection scenarios remains follow-up study due to the high flow density, see Fig. 1, which is responsible for the energy consumption and transportation congestion. Besides, motion planning at intersections may apply to typical businesses like unmanned delivery, see Kimchi et al. (2017). Current last-mile-logistics has great demands but high labor costs at the same time. Thus, motion planning for autonomous vehicles has potential prospects at intersection scenarios.

Krishnan et al. (2018) breaks down an intersection management into the different conundrums involved in motion planning and current approaches to solve them. The common crux is that the environment changes and perception noise will decrease the possibility of collision-free trajectories at the intersection when confronting obstacles with unexpected actions. It is necessary to consider the occurrence of environment uncertainty and model inaccuracy in finding computation-efficient paths.

Fig. 1. An intersection scenario (Camara et al. (2018))

This problem is usually challenging due to the following difficulties. *First*, the scale of search space towards the optimization problem may cause the curse of dimensionality when the scenario becomes complex. *Second*, the reputedly high computational costs under dynamic programming may cause the on-line decision making highly time-consuming. *Third*, the uncertain environment and inaccurate model may introduce the trade-off between fast passage and collision avoidance when given an off-line policy.

In order to address the above challenges, we make the following major contributions. *First*, we introduce event-based optimization to relieve the waste of exploration budget on the basis of guaranteed performance. *Second*, we extract recurrent feature from replay buffer in reinforcement learning to process the sequence model with partial observability. *Third*, we show the considerable adaptabil-

[1] Qing-Shan Jia is the corresponding author.

ity of the developed method to couple with uncertain agent behaviors in complex intersection scenario through simulation results.

The rest of this paper is organized as follows. We briefly review related literature in section II, formulate the problem in section III, introduce the proposed mechanism in section IV, present the numerical results in section V, and briefly conclude in section VI.

2. LITERATURE REVIEW

The motion planning in traffic has attracted much attention in recent years to ensure collision-free navigation. We briefly review the related literatures in this section from two mainstream aspects as rule-based and learning-based.

Classical rule-based decision-making systems can process rule information directly, but are limited in light of ambiguous and noisy sensor data. Van Den Berg et al. (2011) adjusts velocity vector to specify one-step interaction rules for the current geometric configuration. It is short-sighted in time since the approach does not consider the evolution of future states. Aoude et al. (2013) propagates the dynamics forward of obstacles. Camara et al. (2018) uses empirical data to measure behavior. While it may be computationally prohibitive in anticipating motions to account for real-time evaluation.

With the rise of artificial intelligence, reinforcement learning (Sutton and Barto (1998)) shows great potential in a number of domains based on its capability of iterating decision policies from data automatically. Isele et al. (2018) learns active behaviors for autonomous vehicles to navigate in the case of occlusions. Chen et al. (2017) and Everett et al. (2018) use value network to encode the estimated target for finding a collision-free velocity vector with the given joint configuration. Learning-based methods can effectively pretrain the online procedure during the offline computation but only learn mapping from current states.

While real scenarios often feature incomplete and noisy state in current episode resulting from partial observability, the ego agent needs to hedge against the uncertainty in enduring intentions of nearby agents. Bai et al. (2015) presents an intention-aware online planning scheme to utilize the information from systematic states. Osipychev et al. (2015) proposes a proactive collision avoidance system to ensure the safety constraints. Despite these, it is still ambiguous with the gap in sequential decision between collision-free and faster alternatives. Thus, it remains open to find an applied mechanism to coordinate the off-line learning procedure with on-line decision making.

3. PROBLEM FORMULATION

Multi-agent motion planning can be formulated as Partially Observable Markov Decision Process (POMDP) under the assumption that the agent cannot communicate to each other, as the full state of the system cannot be provided to the agent or even determined. POMDP better captures the system dynamic by explicitly acknowledging that the underlying system state are only partial glimpses of the agent.

3.1 System Dynamic

State Space The state vector can be divided into $s = \left(s^o, s^h\right)$, where s^o denotes the observable part, and s^h denotes the hidden part.

The observable state is constructed by concatenating the individual state of ego agent and nearby agents, denoted by s_0^o and $\tilde{s_i^o}$ respectively, in which $\tilde{s_i^o}, i = 1, 2, \ldots, N$ is the state for the ith nearby agent. Take the intended goal position and preferred speed as hidden state s^h.

We simplify the size of system to make the problem tractable. Take the position and velocity vectors in plane as state space. Define x, y as the axis of an agent, and v, ϕ as the magnitude and heading of velocity. Let $s_0^o = (x, y, v, \phi) \in \mathbb{R}^4$, $\tilde{s_i^o} = (x_i, y_i) \in \mathbb{R}^2$. Then the observable state space can be denoted as

$$s^o = (s_0^o, \tilde{s_1^o}, \ldots, \tilde{s_N^o}). \tag{1}$$

Action Space The action space is divided into the set of permissible magnitude and heading value of acceleration. The ego agent followed kinematic constraints derived from second order alignment can better fit the downstream control module. Denote the action set as

$$a = (\eta^1, \eta^2, \ldots, \eta^m) \times (\varphi^1, \varphi^2, \ldots, \varphi^n), \tag{2}$$

in which $\eta^j, j = 1, 2, \ldots, m$ is the discrete magnitude ranges from $[-5, 3](m/s^2)$ and $\varphi^k, k = 1, 2, \ldots, n$ is the discrete heading angle ranges from $[-\pi/12, \pi/12](rad)$. The action space contains the control sequence from deceleration to acceleration and turning from left to right.

State Transition The objective function here is to minimize the expected time T for ego agent to pass the intersection by developing a policy $\pi : s^o \mapsto a$ given the observed sample path. Then it may derive into an optimization problem:

$$\begin{aligned} \underset{\pi}{\operatorname{argmin}} \;& \mathbb{E}\left[T|s^o(t=0)\right] \\ \text{s.t.} \;& x(t+\Delta t) = x(t) + v(t)\cos\phi(t)\Delta t \\ & y(t+\Delta t) = y(t) + v(t)\sin\phi(t)\Delta t \\ & v(t+\Delta t) = v(t) + \eta(t)\Delta t \\ & \phi(t+\Delta t) = \phi(t) + \varphi(t), \end{aligned} \tag{3}$$

where (3) describes the kinematic constraints of ego agent, which will not be influenced by the state distribution of nearby agents. We set the frequency as 20Hz, which means the time interval Δt is 0.05s.

3.2 Q-Learning

Reinforcement learning is concerned with learning control policies for agents interacting with unknown environments. Q-Learning (Volodymyr et al. (2015)) is a model-free algorithm for estimating the long-term expected return of executing an action from a given state.

We can learn estimates for the optimal value of each action, defined as the expected sum of future rewards when taking that action and following the optimal policy thereafter. Taking the observed reward and the max Q-value over all actions a' in the resulting state s' into account, Q-values are learned iteratively with a scalar step size α in the way as

$$Q(s,a) := Q(s,a) + \alpha \left(r + \gamma \max_{a'} Q\left(s', a'\right) - Q(s,a) \right). \tag{4}$$

In the case of Deep Q-Learning (DQN), the model is a neural network parameterized by weights and biases collectively denoted as θ, which naturally generalizes beyond the states and actions it has been trained on. Thus we can take the neural network as a nonlinear approximator to learn the parameterized value function $Q(S_t, A_t; \theta_t)$.

After taking action A_t in state S_t and observing the immediate reward R_{t+1} and resulting state S_{t+1}, DQN update the parameters θ through gradient descent as

$$\theta_{t+1} = \theta_t + \alpha \left(Y_t^{\mathrm{Q}} - Q\left(S_t, A_t; \theta_t\right) \right) \nabla_{\theta_t} Q\left(S_t, A_t; \theta_t\right). \tag{5}$$

DQN uses the same values both to select and to evaluate an action, which makes it more likely to select overestimated values. To prevent this, we refer Double-DQN (Hasselt et al. (2015)) to decouple the selection from the evaluation, which can reduce the observed overestimations. In this way, the target Y_t^Q is defined as

$$Y_t^{\mathrm{Q}} \equiv R_{t+1} + \gamma Q\left(S_{t+1}, \underset{a}{\operatorname{argmax}} Q\left(S_{t+1}, a; \theta_t\right); \theta_t' \right), \tag{6}$$

in which θ_t are the weights of online network to estimate according to the current values, while θ_t' are the weights of target network to fairly evaluate the value of this policy.

The reward function R_t at here is specified to award the agent for passing the intersection, and penalize the agent for colliding with the other agents. As it relates to the proposed methodology, we will describe it further in section 5.2.

3.3 Sequence Model

Deep Q-Learning has no explicit mechanisms for deciphering the underlying state of the POMDP and is only effective if the observations are reflective of underlying system states.

RNN is a natural generalization of feedforward neural networks to sequences (Sutskever et al. (2014)). Hausknecht and Stone (2015) investigate the effects by replacing the first fully-connected layer with a recurrent layer in the Deep Q-Network. Recurrency is a feasible alternative to stacking a history of sequences in input layer, which could better adapt at evaluation time if the quality of observations changes.

We take the sample path from several prior frames into consideration to take advantage of the capability of recurrent network in predicting sequence patterns. In the next section, we will discuss the bootstrapped updates of sequence modeling in-depth.

In the setting of motion planning, the state space and action space do not affect the environment in relevant way. This work refers dueling-DQN (Wang et al. (2015)) into network design, which may help to learn which states are valuable, without having to learn the effect of each action for each state. Coordinating with the sequence modeling by recurrent network, we take shape of the network architecture with one LSTM and two fully-connected layer, see Fig. 2 for the designed network architecture.

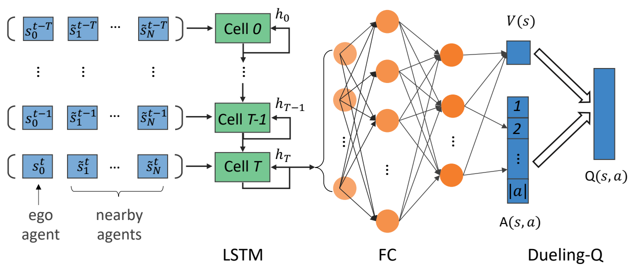


Fig. 2. Network architecture

4. METHODOLOGY

In Section 3, we derived the multi-agent motion planning into POMDP and formulated it by recurrent Q-learning. However, when the numbers of agents increase and the topology of scenarios become complex, the size of search space will grow exponentially. Some basic property of the problem may help to prune the searched space. Besides, the concept of recurrency in sequence modeling is confusing to combine with the experience of Q-learning as for the randomness in sampling from replay buffer. Some designed mechanism may help to extract the feature in fixed length. In this section, we will discuss the improvements in detail.

4.1 Event-based Optimization

Many practical systems have event-driven features, and the dynamic evolution of system states are triggered by discrete events (Cao (2007)). Event-based Optimization (EBO) can effectively mitigate the curse of dimensionality and capture the structure of particular plant in Discrete Event Dynamic System.

EBO provides a unified framework for problems in which decisions can be made only when certain events occur (Xia et al. (2014)). Most existing studies focus on memoryless policies, which make decisions only based on the current observable events. Jia (2011) extend studies on finite-stage EBOs and convert infinite-stage EBO to POMDP.

As for the motion planning in intersection scenario, the agent has no need to determine a state transition everywhere in the whole state space, which may cause the learning process difficult to access optimal sample path. Thus, we define several sets of state transitions with certain common properties. We named them as distance-driven, boundary-driven and collision-driven, which capture the information revealed in a physical event without expanding the state space. How these events trigger the action are described below, see Fig. 3 for details.

- *distance-driven*: When the ego agent arrives to a region scope in designated distance with nearby agents, the action will be concluded from the network output, which means both magnitude and heading will be determined and rectified. The alternative actions are the same as those in the original state space.
- *boundary-driven*: When the ego agent tends to reach the boundary of the road, only the heading will reset to opposite direction, which can soft correct to feasible area. The action space here only aggregates to the dimension of heading.

- *collision-driven*: When the ego agent reaches the last region defined in distance-driven events, an avoidance policy will be triggered as emergent action. The action is within the safe set through a geometric projection check when the event occurs.

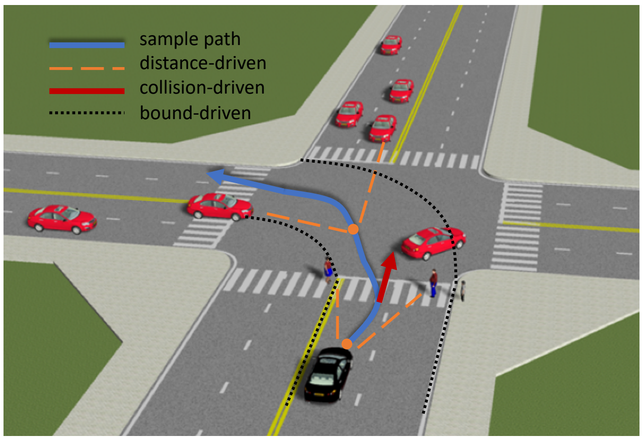


Fig. 3. Description of event-trigger

The controllable events defined above meet social norms as human judgements and execute in a logical timing order. To be mentioned, the event-based mechanism is deployed in both training and evaluation procedure. The potential collision risk is considered and the policy is ensured to avoiding collision except in some extremely limited scenario. Thus, we may take the average speed along with the success rate as the measurement criteria in passing the intersection.

4.2 Random Hybrid Episode

Hausknecht and Stone (2015) introduce recurrent layer to Q-learning architecture, using bootstrapped sequential updates or bootstrapped random updates to combine them together. In Bootstrapped Random Updates, episodes are selected randomly from the replay memory and updates begin at random points in the episode but proceed for only unroll backward call. Sequential updates have the advantage of carrying the LSTM's hidden state forward from the beginning of the episode but violate random sampling policy in DQN.

In order to better fit the bootstrapped updates of recurrent layer and replay buffer mechanism in Q-learning. We propose a method named Random Hybrid Episode (RHE), which is motivated by the concept of Random Hybrid Stroke (RHS) in writer identification task. According to Zhang et al. (2016), each RHS is a randomly sampled short sequence from the whole online data with both pen-down (real stroke) and pen-up (imaginary stroke) information. The added dimension of stroke helps to unify the characters with different lengths.

In the proposed RHE method, we add an extra dimension in input states to mark the transition from one or different episode in experience memory. When a particular event occurs, we define the connection between the same episode as real episode, and the connection starts from the end of one episode to the beginning of the next episode as imaginary episode. The flag we set is binary, see Fig. 4 for details.

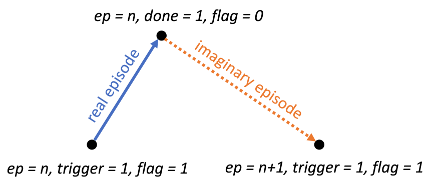

ep = n, done = 1, flag = 0
real episode
imaginary episode
ep = n, trigger = 1, flag = 1
ep = n+1, trigger = 1, flag = 1

Fig. 4. Definition of random hybrid episode

In this way, we simply distract the feature of RHE in replay buffer and derive the bootstrapped updates to a fixed-length vector as the input of LSTM layer. Then the Recurrent Deep Q-networks can better approximate actual Q-values from sequences of observations, leading to better policies in partially observed environments. See Fig. 5 for the complete workflow.

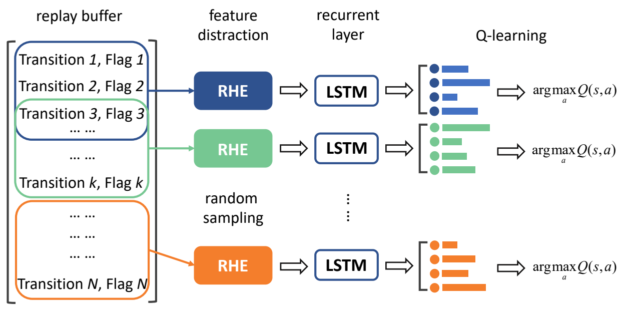


Fig. 5. Workflow of ERQL

5. NUMERICAL RESULTS

In this section, we take the intersection scenario with different kinds of obstacles into account, in which the auxiliary cars and pedestrians may occur with uncertain norms. To verify the efficiency of our method, we analysis from the following aspects.

5.1 Approach

The learning procedure, outlined in Algorithm 1, generalizes our idea to deal with the problem formulation in Section 3. We make a few remarks about the value network. The systematic state taken as input parameterizes a vector range from $[0, 1]$ to normalize the effect in different dimensions. Specifically, we concatenate the binary flag (line 11) with state vector to incorporate RHE feature during the training process, while the flag is fixed to 1 (line 15) in test process as we can regard that the evaluation is in a complete episode.

In the following experiment, we consider the scenario similar as Fig. 3. The black car represents the ego agent, while the red vehicles and the pedestrians wandering in crosswalk are regarded as obstacles.

The ego agent aims to turn left in the intersection, and the process could be divided into two stages. Before crossing the crosswalk, the pedestrians walking from different sides are potential conflicts. After arriving in the center region of intersection, the prdestrians in the crosswalk no longer threaten the ego agents. Then we consider about vehicles from the side of west and north respectively. The obstacles come from two directions in both stages, so we

Algorithm 1 Event-driven Recurrent Q-Learning

```
Input: joint state $s$ and prior experience $E$
Output: off-policy action $a$ for ego agent
1: initialize the value network $V(\cdot;\theta_0)$
2: fulfill the replay buffer $B \leftarrow E$
3: for $ep = 1, 2, \dots, N$ do
4:     initialize the environment
5:     while not passed intersection do
6:         derive the transition $T$ from current $s$
7:         if $T \in$ typical event then
8:             emerge recurrent sequence to update $B$
9:             distract RHE feature $\leftarrow randSubset(B)$
10:        end if
11:        update $\theta_{ep+1} \leftarrow backprop\,(V\,(\cdot;\theta_{ep}))$
12:        update $s$ based on system dynamic
13:    end while
14: end for
15: return $a \leftarrow V(\cdot;\theta_N)$
```

regulate the dimension of input states to switch in smooth. When the selected nearby agents encompass the potential adjacent conflicts, their input states will be substituted by subsequent obstacles from the same direction as they are ensured to be risk-free.

At least four obstacles are concluded in the case, it can also be used under other settings to show the performance of proposed method without loss of generality.

5.2 Performance Evaluation

The maximum speed of ego agent is limited as $10m/s$ during the crossing. We consider average speed and event proportion in final state into reward function with proper weights allocation (0.5:0.5 at here), which represents the efficiency and quality of sample path respectively. In particular, average speed is correspond with the time to pass intersection, while event proportion will be adaptive adjusted to lessen the frequency of boundary-driven events. A punishment -1 will be given if the ego agent collide with nearby agents and the reward in middle process is set to be sparse.

Due to the uncertainty in environment initialization, the policy is evaluated by the mean performance in test set. We conduct 1000 cases towards different initial states and time-variant actions under uncertainty. We additionally run random seeds to test generalization. See Fig. 6 for the performance of different methods in learning procedure. Among them, non-RHE method (EQL) cannot accomplish optimal performance due to the short-sight in subsequent sequences, non-EBO method (RQL) learned with a slow slope due to the computation cost in exploration, while ERQL is found to be quite robust through the confidence interval of learning curve.

In Fig. 7(a), we present the iteration steps in each episode before and after event-trigger. It is shown that the search space with EBO reduced by an order of magnitude than non-EBO method. We make a trade-off between exploration adequacy and computation cost on the promise of performance evaluation. The utilization of event space not only matches the norms in reality, but also saves the computation budget in training procedure. It is shown in

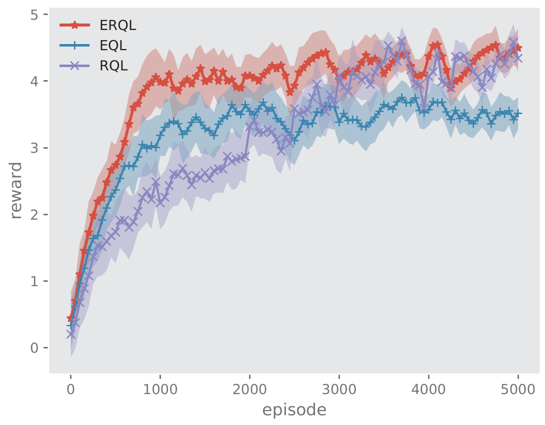
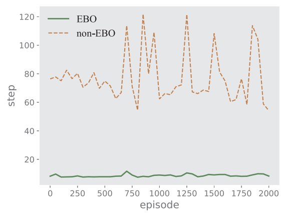
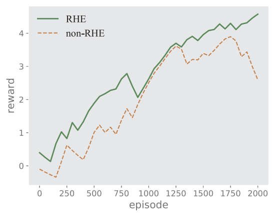


Fig. 6. Performance evaluation



(a) Computation budget



(b) Bootstrap update

Fig. 7. Mechanism improvement

Fig. 7(b) that the RHE mechanism can improve the efficiency of bootstrapped update. Learning procedure with RHE always accomplish a more stable performance than non-RHE method in unroll timesteps, and the latter always yield oscillation. It is ambiguous for the original bootstrapped update mechanism when proceeding for more than one backward call, while the proposed random hybrid episodes could sufficiently utilize the sequence information in experience memory with no limit.

5.3 Method Comparison

This part takes the rule-based and learning-based methods as baseline to evaluate the effect of ERQL. The rule-based strategy relies on a hand-engineered time to collision strategy to decide when to cross. The learning-based method adopt the orthodox Q-learning framework with the same system dynamic and training rounds as ERQL.

Table 1. Method Comparison

Method	average speed	success rate
Rule-based	7.69 m/s	78.5%
Learning-based	7.43 m/s	93.0%
ERQL	8.12 m/s	98.2%

In Table 1, we present the effect between these methods in different characterization. We can see that the proposed ERQL takes the lead in the items of average speed and success rate. Rule-based method has a better speed towards learning-based method but a lower success rate for the reason that it measures with prioritized pass order but easy to confused by imperfect observations.

The rule-base method is always a delicate designed numerical method, which can obtain the local optimal solution on the basis of a simple part. However, when the nearby agents move with agnostic, the rule-based method may stuck in dilemma unless making on-policy decision, which is time-consuming. The learning-based method is a pre-evolutionary version, which could apply to various problems and scenarios. However, a general one with sightless large-scale computing does not excavate the structure of transition and the recurrency of sequence, which may be hard to reach a better solution.

According to the simulation results, the proposed ERQL method helps to solve the problem of motion planning with uncertain environment, which shows considerable improvement in fast passage and collision avoidance.

6. CONCLUSION

This work develops Event-driven Recurrent Q-Learning (ERQL) to focus on motion planning at intersections. In the approach, we incorporate system structure into an event-based framework and introduce recurrency to couple with the partial observability. The designed mechanism shows superiority in saving computation budget and extracting sequence features. However, there are some issues remained for future study. First, the utilization of the open dataset could help to describe the norm of uncertainty. Second, policy-gradient algorithms could be developed to search for continuous action space. Third, the efficiency of migration ability to other scenarios subjects to subsequent experiment verification.

ACKNOWLEDGEMENTS

This work is partially supported by the National Natural Science Foundation of China (Grant No. 61673229).

REFERENCES

Aoude, G.S., Luders, B.D., Joseph, J.M., Roy, N., and How, J.P. (2013). Probabilistically safe motion planning to avoid dynamic obstacles with uncertain motion patterns. *Autonomous Robots*, 35(1), 51–76.

Bai, H., Cai, S., Ye, N., Hsu, D., and Lee, W.S. (2015). Intention-aware online pomdp planning for autonomous driving in a crowd. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 454–460.

Camara, F., Romano, R., Markkula, G., Madigan, R., Merat, N., and Fox, C. (2018). Empirical game theory of pedestrian interaction for autonomous vehicles. In *Proceedings of Measuring Behavior 2018*.

Cao, X.R. (2007). Event-based optimization of markov systems. In *Stochastic Learning and Optimization*, 387–454.

Chen, Y.F., Liu, M., Everett, M., and How, J.P. (2017). Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 285–292.

Everett, M., Chen, Y.F., and How, J.P. (2018). Motion planning among dynamic, decision-making agents with deep reinforcement learning. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 3052–3059.

Geiger, A., Lenz, P., and Urtasun, R. (2012). Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 3354–3361.

Hasselt, H.V., Guez, A., and Silver, D. (2015). Deep reinforcement learning with double q-learning. *Computer Science*.

Hausknecht, M. and Stone, P. (2015). Deep recurrent q-learning for partially observable mdps. In *2015 AAAI Fall Symposium Series*.

Isele, D., Rahimi, R., Cosgun, A., Subramanian, K., and Fujimura, K. (2018). Navigating occluded intersections with autonomous vehicles using deep reinforcement learning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2034–2039. IEEE.

Jia, Q.S. (2011). On solving event-based optimization with average reward over infinite stages. *IEEE Transactions on Automatic Control*, 56(12), 2912–2917.

Kimchi, G., Buchmueller, D., Green, S.A., Beckman, B.C., Isaacs, S., Navot, A., Hensel, F., Bar-Zeev, A., and Rault, S.S.J.M. (2017). Unmanned aerial vehicle delivery system. US Patent 9,573,684.

Krishnan, S., Aadithya, R.G., Ramakrishnan, R., Arvindh, V., and Sivanathan, K. (2018). A look at motion planning for avs at an intersection. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, 333–340.

Levinson, J., Askeland, J., Becker, J., Dolson, J., Held, D., Kammel, S., Kolter, J.Z., Langer, D., Pink, O., Pratt, V., et al. (2011). Towards fully autonomous driving: Systems and algorithms. In *2011 IEEE Intelligent Vehicles Symposium (IV)*, 163–168.

Osipychev, D., Tran, D., Sheng, W., Chowdhary, G., and Zeng, R. (2015). Proactive mdp-based collision avoidance algorithm for autonomous cars. In *2015 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER)*, 983–988.

Sutskever, I., Vinyals, O., and Le, Q. (2014). Sequence to sequence learning with neural networks. *Advances in Neural Information Processing Systems*.

Sutton, R.S. and Barto, A.G. (1998). Reinforcement learning: An introduction. *IEEE Transactions on Neural Networks*, 9(5), 1054–1054.

Van Den Berg, J., Guy, S.J., Lin, M., and Manocha, D. (2011). Reciprocal n-body collision avoidance. In *Robotics Research*, 3–19.

Volodymyr, M., Koray, K., David, S., Rusu, A.A., Joel, V., Bellemare, M.G., Alex, G., Martin, R., Fidjeland, A.K., and Georg, O. (2015). Human-level control through deep reinforcement learning. *Nature*, 518(7540), 529.

Wang, Z., Schaul, T., Hessel, M., Van Hasselt, H., Lanctot, M., and De Freitas, N. (2015). Dueling network architectures for deep reinforcement learning. *arXiv preprint arXiv:1511.06581*.

Xia, L., Jia, Q.S., and Cao, X.R. (2014). A tutorial on event-based optimization—a new optimization framework. *Discrete Event Dynamic Systems*, 24(2), 103–132.

Zhang, X.Y., Xie, G.S., Liu, C.L., and Bengio, Y. (2016). End-to-end online writer identification with recurrent neural network. *IEEE Transactions on Human-Machine Systems*, 47(2), 285–292.