An Autonomous Driving Experience Platform with Learning-Based Functions

Dong Li*†, Dongbin Zhao*†, Qichao Zhang*†, Yuanheng Zhu*†
*The State Key Laboratory of Management and Control for Complex Systems,
Institute of Automation, Chinese Academy of Sciences. Beijing, China
†University of Chinese Academy of Sciences, Beijing, China
E-mail: lidong2014@ia.ac.cn; dongbin.zhao@ia.ac.cn; zhangqichao2013@163.com; yuanheng.zhu@ia.ac.cn

***Abstract*—This paper presents an autonomous driving experience platform for the purpose of improving the social acceptance of autonomous driving technologies. The platform includes two modules: a dangerous object detection (DOD) module and a lane keeping assist (LKA) module. The DOD module first employs an object detection convolutional neural network to locate the vehicle on the image, then the dangerous level is determined based on the corresponding location and geometrical relationship. The LKA module, which is based on reinforcement learning inputs the physical sensor measurements and outputs the steering command to control the vehicle in the lane center. The experiments validate the effectiveness of the proposed methods. The platform is exhibited in 2018 Beijing science and technology week[1].**

I. Introduction

Recently, autonomous driving has received a lot of attention around the world. Thanks to the state-of-the-art perception and control methods, many autonomous driving companies have emerged. As the autonomous driving technologies gradually transfer from academy research to the common life, there are also many concerns, e.g. reliability and cost. These concerns may lower the social acceptance. However, the social acceptance [1] can be improved by demonstrating the customer-oriented features and the corresponding performances. To this end, we present an autonomous driving experience platform as shown in Fig. 1. Along with the platform, two main aspects in the autonomous driving, i.e. perception and control are covered by proposing a deep learning (DL) based dangerous object detection (DOD) approach and a reinforcement learning (RL) based lane keeping assist (LKA) approach, respectively.

As one of the important ingredients in autonomous driving perception module, DOD aims to warn the driver with clear warnings, such as voice warning or visual warning to avoid accidents. Based on the type of sensors used, the DOD methods can be divided into two categories, i.e. the physical sensor based methods, and the visual sensor based methods. The physical sensors mainly include the millimeter-wave radar [2] and lidar [3], which are typically expensive and hard to be large-scale deployed. In contrast, the on-board camera is a relatively cheap device and has been available on most vehicles. The visual information it provides is richer which includes the vehicles, pedestrians, and traffic signs, etc. Thanks to the development of DL, many convolutional neural network (CNN) based approaches have achieved remarkable performance in the vision-based tasks [4] [5] [6]. This motivates the DOD approach with the image stream input. These methods can be divided into two branches. In the first category, the problem is formulated in a two-stage framework [7] which includes an object detection stage and a dangerous level assignment stage. The underlying object is detected by using the pre-designed feature detectors [8] [9] or the CNNs [4] [10]. Then the dangerous level is determined according to the different distance, which is measured by RGB-D camera [11] or radar [12]. In the second category, the DOD process is conducted in an end-to-end manner [13]. By exerting the correlation between the object detection task and distance prediction task, a multi-task learning CNN is employed to predict the dangerous level. However, the above approaches deal with the problem in a static manner in which the temporal information between the consecutive images is ignored. In this paper, we work in the first category and a temporal filter is employed to tackle the false detection or missed

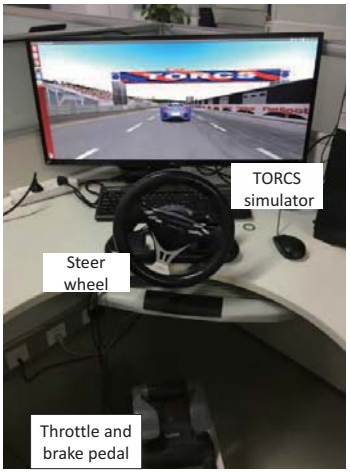


Fig. 1. The TORCS-based autonomous driving experience platform.

This work is supported by the Beijing Science and Technology Plan under Grant No. Z181100008818075 and No. Z181100004618003, the National Natural Science Foundation of China (NSFC) under Grants No. 61573353, No. 61533017, No. 61603382, and No. 61803371, the National Key Research and Development Plan under Grant No. 2016YFB0101000.

[1]Details in https://github.com/dongleecsu/ExperiencePlatform

978-1-5386-9276-9/18/$31.00

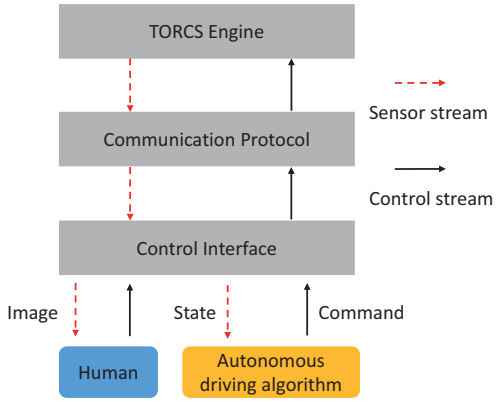


Fig. 2. The autonomous driving experience platform system diagram.

detection cases. In this paper, we focus on the dangerous vehicle detection.

The LKA system aims to control the vehicle running along the lane center. The popular approaches include linear quadratic regulator [14] and model predictive control [15]. But these methods require the system model which is intractable to be accurately estimated. This may harm the control performance. Recently, as a category of data-driven methods, RL has achieved remarkable performance in many control tasks [16] [17] [18]. In the LKA problem, the control action, i.e. steering angle is a continuous variable. So we formulate the control problem in the policy gradient framework.

In this paper, we propose an autonomous driving experience platform for the purpose of improving the social acceptance of the autonomous driving technologies. Along with the platform, a DL-based dangerous vehicle detection approach is proposed, in which the front vehicles are first detected and the dangerous levels are determined based on the geometrical relationship. The RL-based lane keeping assistance approach is developed. The developed experience platform is exhibited in 2018 Beijing science and technology week.

The remainder of this paper is organized as follows. The experience platform is introduced in section II. The DOD and LKA approaches are detailed in section III and IV, respectively. In section V, we show the experimental validations. The conclusion and future work are presented in section VI.

II. Autonomous Driving Experience Platform

In this section, we give a detailed description of the autonomous driving experience platform from three aspects: the platform ingredients, the sensors, and the functions.

A. Platform Ingredients

First, an autonomous driving simulator is needed to provide the basic framework. As one of the most popular open source racing simulators, the open racing car simulator (TORCS) [19] has been tested and applied in many artificial intelligence (AI) researches. It provides real-time multi-view driving scene rendering and various vehicle models. TORCS has been employed as the official platform in the 2009 simulated car racing championship [20]. Therefore, we take the TORCS as the basic framework in the experience platform. The platform diagram is shown in Fig. 2 whose working flow is as follows. First, the sensor measurements are collected from the TORCS engine to describe the current driving state. The underlying sensors include the camera, range meter, and speed meter, etc. Then the measurements are sent to the control interface via the communication protocol. The control interface rearranges the measurements and gives the human participator and the autonomous driving algorithm driver-view image and state variable, respectively. After the human participator and the autonomous driving algorithm make the decision, the control command, i.e. steering command is sent to TORCS engine to finish the one-step simulation.

TABLE I
The Autonomous Driving Experience Platform Sensors List

Sensor	Notation	Value	Unit
Range meter	$range$	[0, 200]	m
Angle sensor	$angle$	$[-\pi, \pi]$	rad
Speed meter	v	[0, 200]	km/h
Engine rotation	rot	[0, 10000]	rpm
Odometer	len	[0, inf]	m
Timer	t_{lap}	[0, inf]	s
Camera	$image$	[0, 255]	-

B. Sensors

Since the origin TORCS only provides the track, vehicle, and basic simulator engine, we develop a set of sensors to support perception and control algorithm implementation in the experience platform. A list of implemented sensors are shown in Table I. There are seven developed sensors in total. The first six are low-dimensional sensors that cover the traffic information in the distance, speed, and angle aspects. In order to give a better driving experience, the camera is implemented to collect high-resolution image 1600×900.

C. Functions

The experience platform includes two main functions, i.e. DOD and LKA. Here we introduce how these two functions work. The underlying algorithms will be detailed in the next two sections.

In the DOD function, the human participator is allowed to drive the host vehicle freely in the track. There are a few pre-programmed traffic vehicles running ahead or behind the host vehicle. The aim is to detect these front traffic vehicles and assign the dangerous level accordingly. Therefore, the DOD module accepts the driver-view image of the host vehicle, i.e. the same image seen by the human participator and predicts the bounding box of the vehicle. Then the distance-based dangerous level estimator outputs the corresponding dangerous level. An additional temporal filter gives the final dangerous level detection results, which are rendered on the screen to give

a visual warning. Four kinds of color including red, orange, yellow, and green are used to render the different dangerous levels. Red indicates the most dangerous vehicle while green indicates the safest vehicle.

The LKA function aims to show the stable control performance of the RL-based autonomous driving controller. The human participator is asked to drive in the lane center as closely as possible. Simultaneously, the RL controller accepts the physical sensor measurements and controls the other vehicle in the next lane. In order to measure the performance, a few metrics are employed including the lane departure count, departure distance, departure time, and a heuristic score which is defined as

$$score = cos(\theta) - sin(\theta) - d/(w) \tag{1}$$

where $\theta \in [-\pi, \pi]$ is the angle between vehicle heading and lane orientation, $d \in [0, +\infty)$ is the distance to lane center, and w is the half lane width.

III. Dangerous Object Detection

The objective of the DOD module is to detect the front vehicles and determine the dangerous level of each vehicle based on the visual input. It includes three parts: a DL-based vehicle detector, a geometry-based distance estimator, and a temporal filter. The vehicle detector first exploits the state-of-the-art object detection method to predict the vehicle location, which is represented by a bounding box. The coordinates of the bounding box center are fed into the distance estimator to predict the relative distance to the detected vehicle. Then the dangerous level is determined based on the distance. Finally, the temporal filter utilizes the correlation information between consecutive frames to output the filtered dangerous vehicle results. The details are introduced as follows.

A. Vehicle Detection

Different from the object detection task in the popular MS COCO challenge which prefers detection accuracy rather than speed, the object detection task here considers both the accuracy and speed since the detection task must be finished on the fly as the human participator drives the host vehicle. To this end, we employ the YOLO [10] object detection neural network, which is able to achieve high detection accuracy while keeping it fast, e.g. 35 frames per second.

The YOLO-based vehicle detection neural network works in an end-to-end manner where the object localization and classification are fused into a single CNN. First, the input image is divided into a few grids. In each grid, the network predicts several bounding boxes and the corresponding confidence scores. The bounding box describes where the detector thinks the vehicle is and the score describes how likely the vehicle is in the bounding box. The confidence score is formulated as $P(obj) * \text{IOU}_{pred}^{truth}$ where $P(obj)$ is the probability an object occurs in the bounding box and $\text{IOU}_{pred}^{truth}$ is the intersection over union of the bounding box and the ground-truth. The loss function, which is used to optimize the bounding box regressor and the classifier is formulated as follows:

$$\mathcal{L} = \sum_i ||(t_i, t_i^*)||_2^2 + ||(p_i, p_i^*)||_2^2 \tag{2}$$

where t_i and t_i^* are the predicted and ground-truth five-element vector containing the bounding box coordinates, height, width, and confidence score. p_i and p_i^* are the predicted class probability. i iterates over all grids and bounding boxes in which the network thinks it contains a vehicle, and $||\cdot||_2^2$ is the square of l-2 norm. Since we intend to detect the vehicle in an image, the classifier predicts two classes: the vehicle and the background.

B. Dangerous Level Estimator

Since the distance to the front vehicle d_v is a direct metric measuring the dangerous level, the dangerous level estimator first estimates the distance to the detected front vehicle, then determines the dangerous level according to the distance. Actually, in the image space, the y-axis coordinate and the distance to the camera are related. We utilize a nine-order polynomial to fit y-axis coordinate of the bounding box center and the distance. After estimating the distance, the dangerous levels are determined as follows

- dangerous, marked red if $d_v < 10$ m;
- relatively dangerous, marked orange if $10 < d_v < 20$ m;
- relatively safe, marked yellow if $20 < d_v < 30$ m;
- safe, marked green if $d_v > 30$ m.

C. Temporal Filter

Although the object detection CNNs have shown promising performance, it works on the static image. For the video-based object detection problem considered, the consecutive frames usually show a strong correlation. Since an object cannot appear or disappear abruptly, a temporal filter is used to improve the detection results. As a post-processing method, it deals with the missed detection and the false detection cases.

For the first case, the temporal filter aims to infer the missed bounding box based on a sliding window over frames. The horizon is considered as 5 in this paper. To this end, we utilize the following bounding box features: the coordinates of the bounding box center (x, y) and the corresponding width and height (w, h). In the temporal sliding window, the difference of these features, e.g. $\Delta x_{t-i} = |x_{t-i} - x_{t-i-1}|$ changes slowly. In order to recover the missed detection at time t, a four-order polynomial is fitted for each feature changes, namely, $\Delta x_t, \Delta y_t, \Delta w_t, \Delta h_t$. Then the missed detection is recovered based on the previous bounding box and the feature changes. In the false detection case, the network falsely detects a bounding box on the background. This case is easy to detect when the feature differences change abruptly. To this end, we maintain an average of each feature difference in the sliding window. If one of these feature differences is much bigger than a threshold, the corresponding bounding box is discarded.

IV. Lane Keeping Assist

The LKA module aims to show the stability of the model-free autonomous driving controller by comparing with the human participator. To tackle the difficulty in vehicle dynamics modeling, we employ the RL-based model-free method to tackle the LKA problem. The goal of this module is to control the vehicle in the lane center with the physical sensor measurements like range meter, angle meter, and speed, etc as its input.

A. Reinforcement Learning Controller

In the LKA problem, the controller determines and conducts a steering command based on the current state, and then the TORCS engine runs the one-step simulation and returns the next state by which the controller generates the next steering command. This process is a sequential decision-making process which is suitable to employ the RL. In the RL framework, the decision-making process is described by the Markov decision process (MDP), which is composed of a five-tuple $(s_t, a_t, r_t, P(s_{t+1}|s_t, a_t), \gamma)$. Here $s_t \in \mathcal{S}$ is the state at time t, $a_t \in \mathcal{A}$ is the action command, $r_t = r(s_t, a_t)$ is the reward signal judging the undertaken action, $P(s_{t+1}|s_t, a_t)$ is the system dynamics which is unknown here, and $\gamma \in [0, 1)$ is the discount rate. A policy $\pi(s_t, a_t)$ casts the probability of selecting a_t at s_t. The objective is to maximize the expected long-term cumulative reward

$$J = \mathbb{E}_{s,a\sim\pi,r}[\sum_{k=1}^{T} \gamma^{k-1} r_k]. \tag{3}$$

where $\mathbb{E}_x[\cdot]$ is the expectation with respect to the variable x, and T is the terminal time step. For a specific state-action pair (s, a), the reward-to-go following the policy $\pi(s, a)$ is formulated as the value function

$$Q^\pi(s, a) = \mathbb{E}_\pi[\sum_{k=1}^{T} \gamma^{k-1} r_k | s_1 = s, a_1 = a]. \tag{4}$$

In the LKA problem, the state s and the steering action a are both continuous variables, which means this is a continuous state-action control problem. Therefore, we employ the popular continuous action RL algorithm deterministic policy gradient (DPG) [21]. A deterministic policy $\mu(s)$ is maintained to generate the action. As in most policy gradient algorithms, the optimal policy $\mu^*(s)$ in DPG is approximated in the actor-critic framework, in which two neural networks are concerned. An actor network $\mu(s; \mathbf{w}^\mu)$ is used to parameterize the policy, and a critic network $Q(s, a; \mathbf{w}^Q)$ is used to parameterize the value function. $\mathbf{w}^\mu$ and $\mathbf{w}^Q$ are the weights of the actor and critic network, respectively. Since the critic aims to approximate the Q value function in (4), the critic optimization is a regression problem, in which the network weights are optimized by minimizing the square error between the approximation $Q(s, a; \mathbf{w}^Q)$ and the ground-truth $Q(s, a)$. Since the true Q-function is unknown, it is substituted by a bootstrapped target:

$$y_t = r_t + \gamma Q^\mu(s_{t+1}, \mu(s_{t+1}; \mathbf{w}^{\mu-}); \mathbf{w}^{Q-}) \tag{5}$$

where $\mathbf{w}^{\mu-}$ and $\mathbf{w}^{Q-}$ are network weights of the target actor and target critic, and the updating speeds are much slower to keep the optimization process stable [22]. Therefore, the critic loss function is formulated as

$$\mathcal{L}(\mathbf{w}^Q) = \mathbb{E}_{s_t,a_t,r_t}[(Q^\mu(s_t, a_t; \mathbf{w}^Q) - y_t)^2]. \tag{6}$$

The actor aims to search the policy $\mu(s; \mathbf{w}^\mu)$ in the policy space to maximize the objective function defined in (3). By following the gradient ascent algorithm, the weights are updated by following the deterministic policy gradient theorem:

$$\nabla_{\mathbf{w}^\mu} J = \mathbb{E}_{s_t}[\nabla_a Q(s, a; \mathbf{w}^Q)|_{s=s_t, a=\mu(s_t)} \nabla_{\mathbf{w}^\mu} \mu(s; \mathbf{w}^\mu)|_{s=s_t}]. \tag{7}$$

B. Learning System

1) State: In order to control the vehicle in the lane center, two kinds of features are needed: the static features, e.g. the vehicle position and pose in the lane coordinate and the dynamic feature, e.g. the vehicle speed. To this end, we use the following sensor measurements to form the state variable. The range meter is used to measure the distances to lane markings which are converted into the distance to lane center d. The angle sensor measures the angle between the vehicle heading direction and the lane orientation θ. The speed meter measures the host vehicle speed along the lane orientation v_x and vertical to lane orientation v_y. Therefore, the state at time t is $s_t = [d_t, \theta_t, v_{x,t}, v_{y,t}]$. Note these sensor measurements are normalized into $[-1, 1]$ to merge the values with different units in one vector. Additionally, a Gaussian noise $\mathcal{N}(0, 0.03^2)$ is added to the normalized state vector to mimic the sensor disturbance in real.

2) Action: The action in the LKA task is the continuous steering command. In RL, one of the most important problems is to balance the exploration and exploitation to ensure the agent learn the optimal policy based on the diverse experiences. The deterministic policy $\mu(s)$ outputs a certain action which is conducted with probability one and does not introduce any exploration by itself. We add a Gaussian noise $\alpha\mathcal{N}(0, 0.03^2)$ with a small probability ϵ on the deterministic action to trigger exploration. The noise coefficient α linearly decays in the learning process from initial value 1 and finally fixes on a small constant 0.1.

3) Reward: The reward signal r_t describes the goodness of the taken action a_t at the state s_t. In the LKA task, the goodness of the issued action can be described from two aspects. First, the controller needs to control the vehicle in the lane center as closely as possible. So the distance to lane center d_t and the angle between vehicle heading and lane orientation θ_t are taken to judge the deviation from the position and pose aspects, respectively. Second, the controller needs to avoid the too big steering command, which will cause poor driving experience. Based on the above two aspects, the reward signal is defined as follows

$$r_t = -\lambda_1 d_t^2 - \lambda_2 \theta_t^2 - \lambda_3 a_t^2 + 0.2 \tag{8}$$

where the coefficient λ_i is used to weigh the influence from the different aspects.

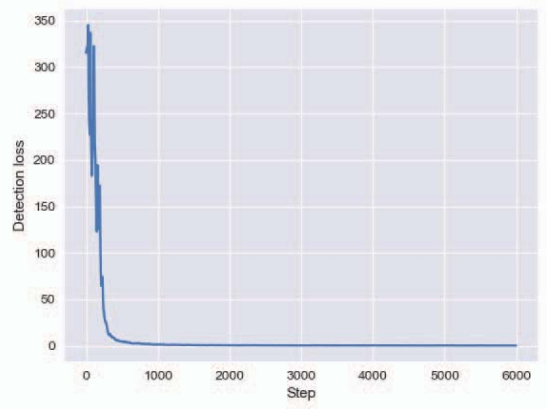


Fig. 3. The object detection neural network training loss.

4) Network Architecture: In the actor network, two fully connected layers with 150 and 100 neurons in each are used to extract the hidden features. Then the last layer with 1 neuron outputs the steering action. In the critic network, the state vector is first transferred by two fully connected layers with 150 and 100 neurons in each, and the action is transferred by a hidden layer with 100 neurons. Then these two outputs are merged and passed to the last hidden layer which has 100 neurons. Finally, the output layer linearly transfers the features to a scalar Q value. Since the steering command is in $[-1, 1]$, the activation function in the output layer of the actor is chosen as tanh function. The activation functions in all hidden layers are $\text{relu}(x) = \max(0, x)$.

V. Experiments

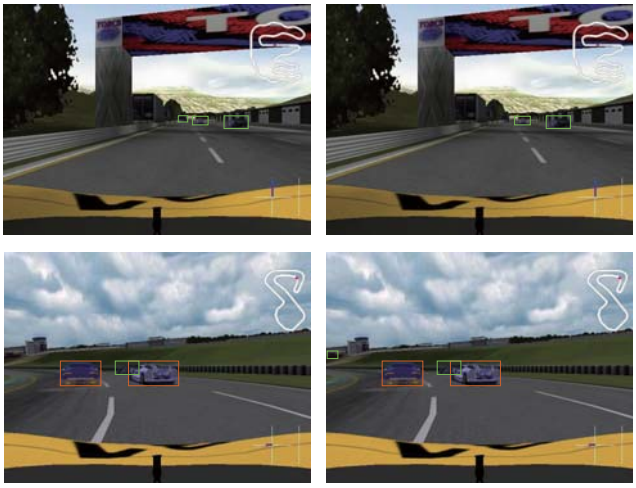
A. Dangerous Object Detection Results

To train the vehicle detection network, several pre-programmed AI vehicles are set as the front traffic vehicles. The host vehicle is controlled by the human to collect the driver-view images with better diversity. Moreover, five tracks serve as the data collection tracks so as to improve the data diversity. Finally, there are 6000 images collected and labeled, in which 5000 images are the training data and the rest are the test data.

To deal with over-fitting, we train the vehicle detection network based on the 32-layer pre-trained YOLO-v3 model. The stochastic gradient descent optimizer with momentum 0.9 is used to update the weights, and the learning rate is 0.001. The l-2 norm regularization with weight decay $5e^{-4}$ is used to keep the network from over-fitting. The total training steps are 6000. The training loss is shown in Fig. 3 where it gradually decreases and converges finally. The detection precision and recall metrics on the test dataset is shown in Table II. Moreover, the results of using temporal filter are also shown in Table II. Although the detection network itself is accurate enough, the temporal filter is able to improve the performance when the vehicle is very small or the image is blurred. For example, the detection comparison with and without temporal filter is shown in Fig. 4.

TABLE II
The Detection Performance with and without Temporal Filter

Detector	Precision	Recall
With temporal filter	0.943	0.945
Without temporal filter	0.927	0.937



(a) With temporal filter (b) Without temporal filter

Fig. 4. The vehicle detection results with and without temporal filter. The green bounding box represents safe since the distance is over 30 m. The orange bounding box means relatively dangerous since the distance is less than 20 m.

B. Lane Keeping Assist Results

In the LKA experiments, we want to answer the question that how good the RL-based controller is for the LKA task. The state, action, and reward are defined as in section IV. The reward coefficients are selected as $\lambda_1 = 0.3$, $\lambda_2 = 1$, and $\lambda_3 = 0.03$. The discount rate $\gamma = 0.99$ and the total training step is set to $6e^5$. The optimizer is Adam, and the learning rates are $1e^{-3}$ and $1e^{-4}$ for the actor and critic network, respectively. Note in the experiment, we set the throttle pedal pressure as a constant to keep the vehicle speed vary in $[50, 70]$ km/h. The training track is chosen as *g-track-3* in TORCS. The curve of the cumulative reward in an episode defined in (3) is shown in Fig. 5. It takes about 10 minutes to learn a feasible policy that is able to control the vehicle in the lane. From the Fig. 5, we can see that the agent tries to explore in the state-action space and drives poorly at the beginning, then it gradually converges to an LKA controller.

In order to validate the generalization performance of the RL controller, we use the controller trained on the track g-track-3 to run on the unknown track *g-track-2*. The steering command, distance to lane center, and yaw angle history in the unknown track are shown in Fig. 6. We can tell from it that the RL controller can not only generalize to the unknown track but also maintain the good performance that is able to control the vehicle in the track center as closely as possible.

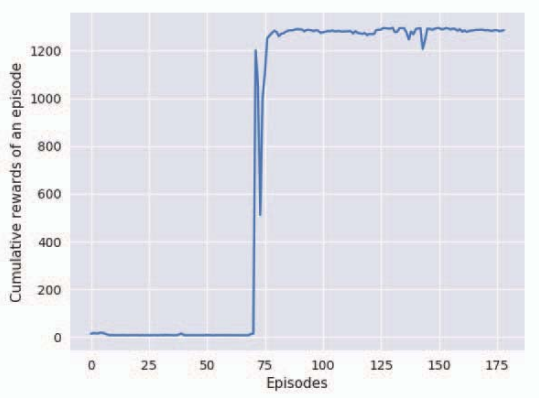


Fig. 5. The cumulative reward of an episode in the training phrase in track g-track-3.

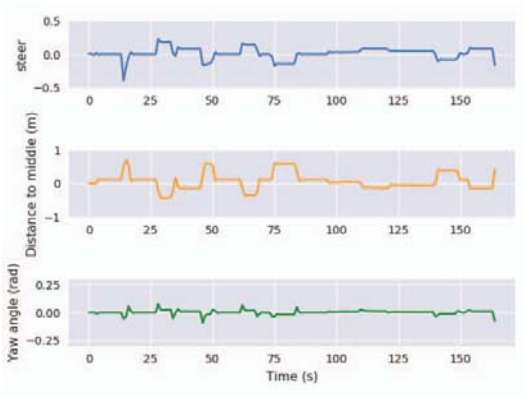


Fig. 6. The steer command, distance to lane center, and yaw angle in the unknown track g-track-2.

C. Autonomous Driving Experience Platform Exhibition

In order to generalize the autonomous driving techniques and test the stability of the developed platform, we attend 2018 Beijing science and technology week from 19 $\sim$ 26 May 2018. There are over 120 thousand viewers participated in the exhibition. The developed experience platform is one of the most popular projects and is reported by the local media[2].

VI. Conclusion

In this paper, we present an autonomous driving experience platform. Along with the platform, two functions which cover the perception and control aspects of the autonomous driving are proposed, i.e. dangerous object detection and lane keeping assist. In the dangerous object detection module, first the front vehicles are located by a CNN, and then the relative distance is estimated and used to determine the dangerous level. To reduce the false positive and false negative detections, a temporal filter is designed to improve the precision and recall. In the lane keeping assist module, the problem is formulated in the RL framework, and a policy gradient controller is designed to learn the control policy. The experiments validate the effectiveness of the proposed methods. The platform is exhibited in the 2018 Beijing science and technology week and reported by the local media. The future work intends to add more functions to the platform, like lane keeping warning and adaptive cruise control.

[2]http://epaper.bjnews.com.cn/html/2018-05/20/content_720544.htm

References

[1] J. Wei, J. M. Snider, J. Kim, J. M. Dolan, R. Rajkumar, and B. Litkouhi, "Towards a viable autonomous driving research platform," *2013 IEEE Intelligent Vehicles Symposium*, pp. 763–770, 2013.

[2] S. J. Park, T. Y. Kim, S. M. Kang *et al.*, "A novel signal processing technique for vehicle detection radar," *IEEE MTT-S International Microwave Symposium Digest*, vol. 1, pp. 607–610 vol.1, 2003.

[3] T. Ogawa, H. Sakai, Y. Suzuki, K. Takagi, and K. Morikawa, "Pedestrian detection and tracking using in-vehicle lidar for automotive application," *IEEE Intelligent Vehicles Symposium*, pp. 734–739, 2011.

[4] S. Ren, K. He *et al.*, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, pp. 1137–1149, 2015.

[5] D. Zhao, Y. Chen, and L. Lv, "Deep reinforcement learning with visual attention for vehicle classification," *IEEE Transactions on Cognitive and Developmental Systems*, vol. 9, pp. 356–367, 2017.

[6] L. Lv, D. Zhao, and Q. Deng, "A semi-supervised predictive sparse decomposition based on task-driven dictionary learning," *Cognitive Computation*, vol. 9, pp. 115–124, 2016.

[7] Z. Shan, Q. Zhu, and D. Zhao, "Vehicle collision risk estimation based on RGB-D camera for urban road," *Multimedia Systems*, vol. 23, pp. 119–127, 2014.

[8] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," *IEEE Computer Vision and Pattern Recognition*, vol. 1, pp. 511–518, 2001.

[9] R. Lienhart and J. Maydt, "An extended set of Haar-like features for rapid object detection," *International Conference on Image Processing*, vol. 1, pp. 900–903 vol.1, 2002.

[10] J. Redmon, S. K. Divvala, R. B. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 779–788, 2016.

[11] C. Kerl, J. Sturm, and D. Cremers, "Dense visual slam for RGB-D cameras," *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2100–2106, 2013.

[12] H. Cho, Y.-W. Seo, B. V. K. V. Kumar, and R. Rajkumar, "A multi-sensor fusion system for moving object detection and tracking in urban driving environments," *IEEE International Conference on Robotics and Automation*, pp. 1836–1843, 2014.

[13] Y. Chen, D. Zhao, L. Lv, and Q. Zhang, "Multi-task learning for dangerous object detection in autonomous driving," *Information Sciences*, vol. 432, pp. 559–571, 2018.

[14] K. Yi, S. Lee, and Y. D. Kwon, "An investigation of intelligent cruise control laws for passenger vehicles," *Proceedings of the Institution of Mechanical Engineers*, vol. 215, no. 2, pp. 159–169, 2001.

[15] T. Keviczky, P. Falcone, F. Borrelli, J. Asgari, and D. Hrovat, "Predictive control approach to autonomous vehicle steering," in *2006 American Control Conference*, June 2006.

[16] Y. Zhu, R. Mottaghi *et al.*, "Target-driven visual navigation in indoor scenes using deep reinforcement learning," *IEEE International Conference on Robotics and Automation*, pp. 3357–3364, 2017.

[17] D. Zhao, Z. Xia, and Q. Zhang, "Model-free optimal control based intelligent cruise control with hardware-in-the-loop demonstration," *IEEE Computational Intelligence Magazine*, vol. 12, no. 2, pp. 56–69, 2017.

[18] Y. Zhu, D. Zhao, and Z. Zhong, "Adaptive optimal control of heterogeneous cacc system with uncertain dynamics," *IEEE Transactions on Control Systems Technology*, pp. 1–8, 2018.

[19] B. Wymann, E. Espié, C. Guionneau *et al.*, "TORCS, The Open Racing Car Simulator," http://www.torcs.org, 2014.

[20] D. Loiacono, P. L. Lanzi *et al.*, "The 2009 simulated car racing championship," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 2, pp. 131–147, 2010.

[21] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. A. Riedmiller, "Deterministic policy gradient algorithms," in *ICML*, 2014.

[22] V. Mnih, K. Kavukcuoglu, D. Silver *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, pp. 529–533, 2015.