Title	A multiple-goal reinforcement learning method for complex vehicle overtaking maneuvers
Author(s)	Ngai, DCK; Yung, NHC
Citation	Ieee Transactions On Intelligent Transportation Systems, 2011, v. 12 n. 2, p. 509-522
Issued Date	2011
URL	http://hdl.handle.net/10722/137287
Rights	Creative Commons: Attribution 3.0 Hong Kong License

A Multiple-Goal Reinforcement Learning Method for Complex Vehicle Overtaking Maneuvers

Daniel Chi Kit Ngai and Nelson Hon Ching Yung, *Senior Member, IEEE*

***Abstract*—In this paper, we present a learning method to solve the vehicle overtaking problem, which demands a multitude of abilities from the agent to tackle multiple criteria. To handle this problem, we propose to adopt a multiple-goal reinforcement learning (MGRL) framework as the basis of our solution. By considering seven different goals, either Q-learning (QL) or double-action QL is employed to determine action decisions based on whether the other vehicles interact with the agent for that particular goal. Furthermore, a fusion function is proposed according to the importance of each goal before arriving to an overall but consistent action decision. This offers a powerful approach for dealing with demanding situations such as overtaking, particularly when a number of other vehicles are within the proximity of the agent and are traveling at different and varying speeds. A large number of overtaking cases have been simulated to demonstrate its effectiveness. From the results, it can be concluded that the proposed method is capable of the following: 1) making correct action decisions for overtaking; 2) avoiding collisions with other vehicles; 3) reaching the target at reasonable time; 4) keeping almost steady speed; and 5) maintaining almost steady heading angle. In addition, it should also be noted that the proposed method performs lane keeping well when not overtaking and lane changing effectively when overtaking is in progress.**

***Index Terms*—Artificial intelligence, learning control systems.**

I. Introduction

THE ULTIMATE goal of the autonomous vehicle (AV) challenge [1]–[4] is the development of driverless cars that can automatically perform a multitude of tasks (e.g., lane following, lane changing [5], [6], and overtaking [7]–[11]) without human intervention.

In principle, overtaking is the act of driving around a slower moving leading vehicle in the path of the host vehicle (agent). It can be treated as a three-phase problem [7], which consists of the agent performing the following tasks: 1) diverting from its original lane; 2) driving pass the leading vehicle in the adjacent faster lane; and 3) returning to its original lane afterward. Compared with other AV research, the problem of overtaking has been somewhat neglected. In [7], Shamir proposed an algorithm to compute the optimal trajectory for overtaking a leading vehicle in a straight road. Naranjo *et al.* [12]–[14] proposed a fuzzy control system that essentially changes the driving mode according to the three phases. Both methods focus on the straight road with one leading vehicle only and are not guaranteed to work if the road has curvature and is occupied by multiple vehicles.

In this paper, we present a learning method to solve the aforementioned vehicle-overtaking problem. We propose to adopt a multiple-goal reinforcement learning (RL) framework as the basis of our solution. RL [15] aims to find an appropriate mapping from situations to actions in which a certain reward is maximized. The vehicle that is responsible for making a decision is called an agent in RL. By considering seven different goals, either Q-learning (QL) or double-action QL (DAQL) [16] is employed for determining action decisions based on whether the other vehicles interact with the agent for that particular goal. QL is a popular RL algorithm, whereas DAQL evolves from QL and uses an autoregressive (AR) prediction model that actively considers the responses of nearby vehicles. While other research concentrates on vehicle control to achieve the goal, the proposed method derives navigation decisions to achieve the same, if not better, results. We believe that vehicle control on throttle, brake, and steering can be easily obtained by the use of proper vehicle dynamic models once correct selection of action to be performed in a particular situation is made. Together with a fusion function, the proposed method offers a powerful approach for dealing with demanding situations such as overtaking. From the simulations, it can be concluded that the proposed method is capable of the following: 1) making correct action decisions for overtaking; 2) avoiding collisions with other vehicles; 3) reaching the target at a reasonable time; 4) keeping a steady speed; and 5) maintaining an almost steady heading angle. In addition, it should also be noted that the proposed method performs lane keeping well when not overtaking and lane changing effectively when overtaking is in progress.

This paper is organized as follows: an overview of the proposed multiple-goal RL (MGRL) framework is given in Section II. Following that, Section III describes the implementation of the RL agent. Section IV presents the details of the method proposed. Section V outlines the simulation environment and the results obtained under different simulation cases. Section VI evaluates the proposed method under different simulation cases. Finally, the conclusion is given in Section VII.

Manuscript received March 1, 2008; revised November 24, 2008, September 17, 2009, June 11, 2010, October 12, 2010, and November 2, 2010; accepted December 12, 2010. Date of publication February 7, 2011; date of current version June 6, 2011. This work was supported in part by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China, under Project HKU7194/06E and in part by the Postgraduate Studentship of the University of Hong Kong. The Associate Editor for this paper was M. Brackstone.

D. C. K. Ngai is with ASM Assembly Automation Ltd., Kwai Chung, Hong Kong (e-mail: ckdaniel@hotmail.com).

N. H. C. Yung is with the Department of Electrical and Electronic Engineering, University of Hong Kong, Pokfulam, Hong Kong (e-mail: nyung@eee.hku.hk).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TITS.2011.2106158

II. Multiple-Goal Reinforcement Learning

Depending on the characteristics of the environment, MGRL can be broadly categorized into three types: 1) static known; 2) static unknown; and 3) dynamically changing. A dynamically changing environment can be defined as an environment with movable obstacles, and a movable target that their current positions can be detected by the sensors of the agent, but their future movements are not known to the agent. Autonomous road vehicle navigation is an example of applications that act on a dynamically changing environment. In addition, autonomous road vehicle navigation is a multiple-goal problem involving a number of the following tasks at any one time: target seeking, collision avoidance, lane following, lane changing, and safety distance keeping. We believe that this problem can be better solved by solutions that explicitly consider these goals simultaneously. The MGRL method proposed in this paper is one such solution.

To deal with a dynamically changing environment, it is believed that the motion of the obstacles must be considered. Such consideration must include two aspects: 1) the influence of obstacles' and/or target's motion on the agent's decision-making process and 2) obstacle motion prediction. While there exist methods that adopt the aforementioned view, however, they assume that obstacle motion or behavior is either known as *a priori* [17], [18] or well defined, e.g., at constant speed and known steering angle [19], [20]. Unfortunately, neither of the assumptions is true in real-world navigation.

Given these limitations, it is motivated in this paper to explicitly consider the "obstacle influence" and "obstacle motion prediction" aspects. In doing so, navigation is no longer treated as a control problem that merely transforms sensor inputs to mechanical outputs. Rather, it is considered as an intelligent decision-making process that deals with higher level strategies called goals with the use of planning, learning, prediction, and the consideration of environment response. Although the goals required in different applications may significantly vary, the navigation problem can now be viewed as a generalized multiple-goal problem. For example, the overtaking problem of the road requires the agent to find a collision-free path while simultaneously obeying the lane following and changing rules. Due to the fact that the road is used by other vehicles that are controlled by other intelligent decision makers (human or otherwise), we need to learn how their decisions will affect the agent's decision and predict their future actions, before we can appropriately respond in the short term, as well as achieve its goals in the long run. As such, we are motivated to develop a generalized methodology that deals with the multiple-goal requirements in a dynamically changing environment by considering the obstacles' influence and motion explicitly throughout.

RL is essentially the learning of the mapping of situations (states) to actions. Traditionally, this is done by using the Markov decision process model. However, it describes a single-agent environment in which there is no other decision-making agent in the environment. By viewing the real world as an environment that contains other multiple decision-making agents, it is therefore necessary to consider the decisions made by each agent. As such, the double-action concept is introduced to capture this extra piece of information. The DAQL method is thus developed based on this inspiration on top of the foundation of QL. DAQL has explicit consideration on the actions performed (or "influence") by other parties in the environment and thus allows a more efficient use of the reinforcement leaning method in a dynamically changing environment [21].

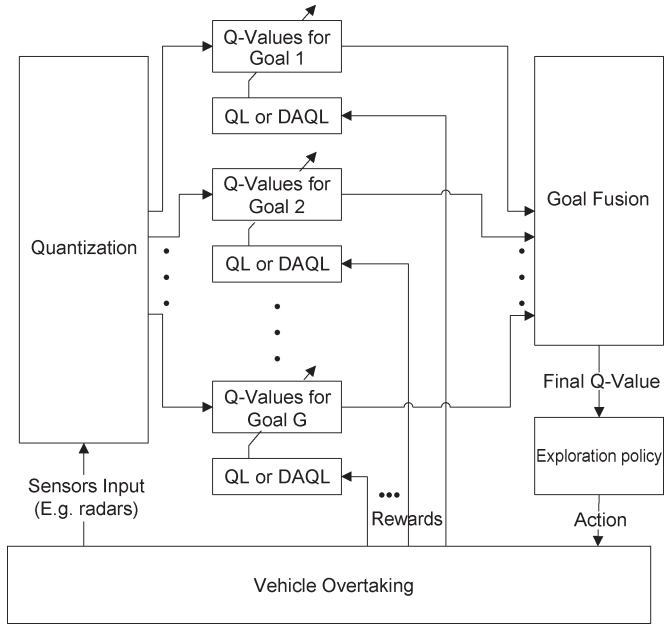

Fig. 1. Conceptual diagram of MGRL.

Within the context of the proposed method, it is assumed that sensors are employed to capture environmental information, which is then quantized into discrete states. From these descriptions of the environment, QL or DAQL is used to enable the agent to learn and decide on actions for a particular goal that will bring a maximum reward in the long run, and a goal fusion function is employed to give an overall consistent output action, given that individual actions are derived from individual goals. Eventually, an exploration policy is applied afterward for the RL algorithm to learn through exploring the environment.

Fig. 1 shows the MGRL model with totally G goals, of which conventional QL and DAQL can be picked for learning in any one case, depending on the nature of the environment. Normally, QL is sufficient for cases where there are no environmental responses or they can be ignored, and DAQL is used otherwise. For example, for a normal navigation problem [22], [23] that consists of two goals, i.e., target seeking and collision avoidance, if other vehicles in the environment or the target is nonstationary, then they have to be dealt with by DAQL, whereas, if they are stationary, then QL will suffice. In the case of vehicle overtaking, we proposed to consider seven goals: 1) target seeking; 2) collision avoidance; 3) lane following; 4) choosing slow lane if *not* overtaking; 5) choosing fast lane if overtaking; 6) keeping steady speed; and 7) keeping steady heading angle. Clearly, some of these goals are contradictory, e.g., goals 5 and 6, whereas some are consistent, e.g., goals 3 and 6. Therefore, a safe and smooth overtaking action requires these goals to be appropriately fused.

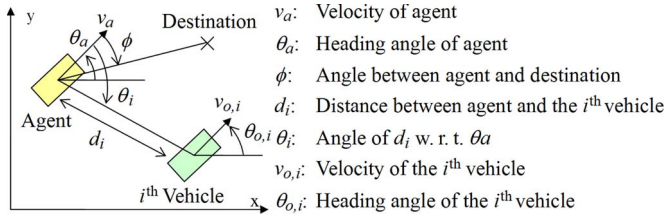

Fig. 2. Control variables of agent and the ith vehicle.

III. Realization of the Multiple-Goal Reinforcement Learning Agent

A. Geometrical Relations Between the Agent and Other Vehicles

1) Quantization of Sensor Inputs: Assume that the environment consists of one MGRL vehicle (agent) and N other vehicles. The agent is the vehicle that is to perform an overtaking action. The control variables of the agent and the ith vehicle at time t are depicted in Fig. 2. We assume that the vehicle operates in an environment where intervehicle communication is not allowed. It is further assumed that the relative distances between the agent and other vehicles can be sensed by distance sensors on the agent [6], [24]. We further assume that the sensors have a minimum and a maximum detectable distance of $d_{s,\min}$ and $d_{s,\max}$, respectively. As not all vehicles are within this range, M vehicles in the environment can be detected, where $M \leq N$. Without losing generality, the information gathered by the agent is quantized into discrete states. In situations where high precision is required, the number of states can be further increased. The location of the ith vehicle is denoted by distance $d_i \in D_o$ from the agent, where $D_o = [d_{s,\min}, d_{s,\max}] \subset \Re$, and angle $\theta_i \in \Theta$, where $\Theta = [0, 2\pi] \subset \Re$. The two parameters d_i and θ_i are quantized into N_d and N_θ number of discrete states, respectively, as follows:

$$\widetilde{d}_i = \begin{cases} \lfloor ((N_d - 1) \times d_i) / d_{s,\max} \rfloor, & \text{if } d_i < d_{s,\max} \\ N_d - 1, & \text{otherwise} \end{cases} \tag{1}$$

$$\widetilde{\theta}_i = \begin{cases} \left\lfloor \frac{\theta_i + \pi/N_\theta}{2\pi/N_\theta} \right\rfloor, & \text{for } 0 \leq \theta_i < (2N_\theta - 1)\pi/N_\theta \\ 0, & \text{for } (2N_\theta - 1)\pi/N_\theta \leq \theta_i < 2\pi. \end{cases} \tag{2}$$

The state set for vehicle location is defined as $s_i \in S_i$, where $S_i = \{(\widetilde{d}_i, \widetilde{\theta}_i) | \widetilde{d}_i \in D_q \text{ and } \widetilde{\theta}_i \in \Theta_{q1}\}$, $\Theta_{\text{q1}} = \{j \mid j = 0, 1, \ldots, N_\theta - 1\}$, and $D_q = \{k | k = 0, 1, \ldots, N_d - 1\}$. There are altogether $N_d \times N_\theta$ states for S_i. Vehicles are thus located in a quantized polar coordinate with respect to the agent.

2) Quantization of Sensor Inputs: We assume that the agent's heading angle is $\phi \in \Theta$ w.r.t. the target. The state set for the relative location of the target is represented by the angle $\widetilde{\phi} \in \Theta_{q2}$ and $\Theta_{q2} = \{j \mid j = 0, 1, \ldots, N_\phi - 1\}$, and quantization is achieved as follows:

$$\widetilde{\phi} = a \quad \text{if} \quad \sigma_{a+1} < \phi \leq \sigma_a \tag{3}$$

where

$$\sigma_a = \begin{cases} \pi/2^a, & \text{for } a = 0, \ldots, N_\phi/2 - 1 \\ -\pi/2^{15-a}, & \text{for } a = N_\phi/2, \ldots, N_\phi - 1. \end{cases} \tag{4}$$

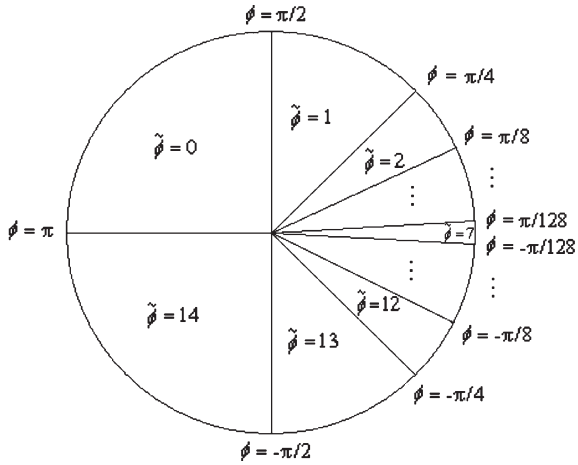

Fig. 3. Quantization of heading angle ϕ when $N_\phi = 15$.

$\widetilde{\phi}$ is quantized in such a way that there are more states in front of the agent rather than behind it so that the agent can more sensitively respond to vehicles moving in front of it when performing overtaking. Therefore, instead of evenly quantizing $\widetilde{\phi}$, N_ϕ states are used to quantize $\widetilde{\phi}$ unevenly, as shown in Fig. 3. Therefore, we have more states to represent vehicles in front of the agent, because this information is more important to the agent when performing overtaking.

3) Quantization of Lane Information: We assumed that the road has an arbitrary number of lanes while we use a road with three lanes for demonstration in this paper. We further assumed that lane width d_{lane} is derived from lane marking that can be detected onboard the agent [25]. Furthermore, the center line of the lane and the agent's relative position to the line can be calculated, as shown in Fig. 4(a). The agent's location relative to the nearest lane is represented by the perpendicular distance (d_{LF}) between the lane center line and agent's center. The difference between its heading angle $(\theta_a, 0 \leq \theta_a \leq 2\pi)$ and the angle of the center line on the nearest lane $(\theta_l, 0 \leq \theta_l \leq \pi)$ is denoted as θ_{LF}, as shown in Fig. 4(b). d_{LF} and θ_{LF} are denoted by $d_{\text{LF}} \in D_{\text{LF}}$, where $D_{\text{LF}} = [-d_{\text{lane}}/2, d_{\text{lane}}/2) \subset \Re$, and $\theta_{\text{LF}} \in \Theta$, where $\Theta = [0, \pi] \subset \Re$. The state set for the agent's location is $s_{\text{LF}} \in S_{\text{LF}}$, where $S_{\text{LF}} = \{(\widetilde{d}_{\text{LF}}, \widetilde{\theta}_{\text{LF}}) | \widetilde{d}_{\text{LF}} \in D_{\text{LF},q} \text{ and } \widetilde{\theta}_{\text{LF}} \in \Theta_{\text{LF},q}\}$, $\Theta_{\text{LF},q} = \{j \mid j = 0, 1, \ldots, N_{\theta LF} - 1\}$, and $D_{\text{LF},q} = \{k | k = 0, 1, \ldots, N_{\text{dLF}} - 1\}$. We use an equally sized partition to quantize the distance and angle of the agent's location relative to the center lane. Therefore, $N_{\theta LF}$ and N_{dLF} discrete states are used to represent θ_{LF} and d_{LF}, respectively, and quantization is achieved as follows:

$$\widetilde{d}_{\text{LF}} = \left\lfloor \frac{d_{\text{LF}} + d_{\text{lane}}/2}{d_{\text{lane}}/N_{\text{dLF}}} \right\rfloor \tag{5}$$

$$\widetilde{\theta}_{\text{LF}} = \left\lfloor \frac{\theta_{\text{LF}}}{\pi/(N_{\theta\text{LF}} - 1)} \right\rfloor. \tag{6}$$

Furthermore, the perpendicular distance between the agent and the center line of the rightmost or the leftmost lane is denoted by d_{SL} or d_{FL}, where $D_{\text{SL}} = [-d_{\text{lane}}/2, 5d_{\text{lane}}/2) \subset \Re$, and $D_{\text{FL}} = [-d_{\text{lane}}/2, 5d_{\text{lane}}/2) \subset \Re$. The state sets for

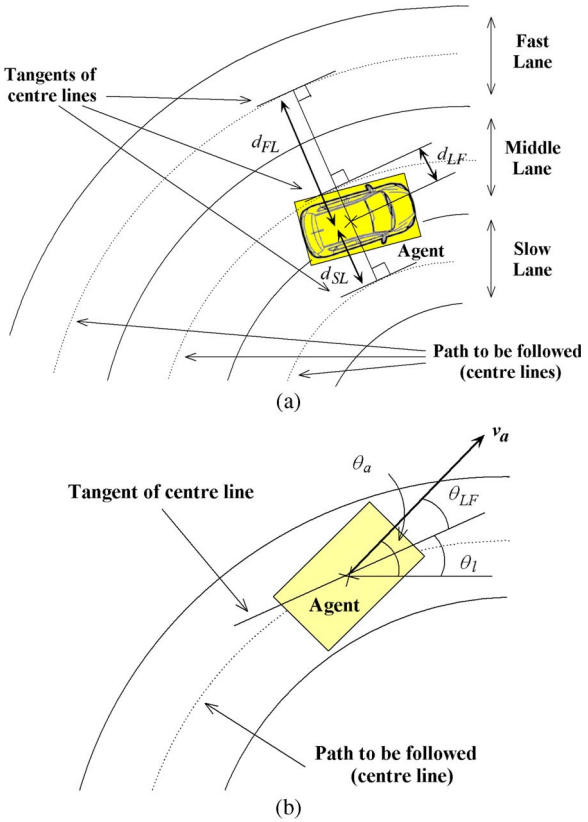

Fig. 4. Relation between agent and lane. (a) Distance relation. (b) Angle relation.

the agent's location with respect to the slow and fast lanes are denoted as $s_{\mathrm{SL}} \in S_{\mathrm{SL}}$ and $s_{\mathrm{FL}} \in S_{\mathrm{FL}}$, respectively, where $S_{\mathrm{SL}} = \{(\widetilde{d}_{\mathrm{SL}}, \widetilde{\theta}_{\mathrm{LF}}) | \widetilde{d}_{\mathrm{SL}} \in D_{\mathrm{SL},q}$ and $\widetilde{\theta}_{\mathrm{LF}} \in \Theta_{\mathrm{LF},q}\}$, $S_{\mathrm{FL}} = \{(\widetilde{d}_{\mathrm{FL}}, \widetilde{\theta}_{\mathrm{LF}}) | \widetilde{d}_{\mathrm{FL}} \in D_{\mathrm{FL},q}$ and $\widetilde{\theta}_{\mathrm{LF}} \in \Theta_{\mathrm{LF},q}\}$, $D_{\mathrm{FL},q} = \{k | k = 0, 1, \ldots, 3N_{\mathrm{dLF}} - 1\}$, and $D_{\mathrm{SL},q} = \{k | k = 0, 1, \ldots, 3N_{\mathrm{dLF}} - 1\}$, and quantization is achieved as follows:

$$\widetilde{d}_{\mathrm{SL}} = \left\lfloor \frac{d_{\mathrm{SL}} + d_{\mathrm{lane}}/2}{d_{\mathrm{lane}}/(3N_{\mathrm{dLF}} - 1)} \right\rfloor \tag{7}$$

$$\widetilde{d}_{\mathrm{FL}} = \left\lfloor \frac{d_{\mathrm{FL}} + d_{\mathrm{lane}}/2}{d_{\mathrm{lane}}/(3N_{\mathrm{dLF}} - 1)} \right\rfloor. \tag{8}$$

Since there are N_{dLF} states for the distance between the agent and the nearest lane, the number of states for the agent in one of the three lanes (fast, middle, and slow) will be three times of N_{dLF}, i.e., $3N_{\mathrm{dLF}}$.

4) Actions of the Agent and Other Vehicles: We assume that the agent's reference heading angle $(\Delta\theta_a)$ is bounded by $\theta_{s,\max}$ in each time step. The output actions are therefore given by $a \in A$, where $A = \{(|\Delta\vec{v}_a|, \Delta\theta_a) | |\vec{v}_a| \in C_a$ and $\Delta\theta_a \in \Theta_a\}$, $C_a = \{m \times c_{a,\max}/2 | m = -(N_v - 1)2, \ldots, (N_v - 1)/2\}$, $\Theta_a = \{2n\theta_{s,\max}/(N_a - 1) - \theta_{s,\max} | n = 0, 1, \ldots, N_a - 1\}$, $\Delta\vec{v}_a$ is the change in velocity of the agent, and $c_{a,\max}$ is the maximum acceleration of the agent. We use N_v and N_a discrete values to represent $\Delta\vec{v}_a$ and $\Delta\theta_a$, respectively. There are altogether $N_v \times N_a$ actions. For DAQL, we assume that vehicles have velocity $|\vec{v}_o| \in \Re^+$ and heading angle $\theta_o \in \Theta$. They are quantized to $a_i^2 \in A_o$, where $A_o = \{(\widetilde{v}_o, \widetilde{\theta}_o) | \widetilde{v}_o \in V_q$ and $\widetilde{\theta}_o \in \Theta_{q3}\}$, $V_q = \{l | l = 0, 1, \ldots, N_{v0} - 1\}$, and $\Theta_{q3} = \{j \quad | j = 0, 1, \ldots, N_{\theta o} - 1\}$. Quantization is achieved as follows:

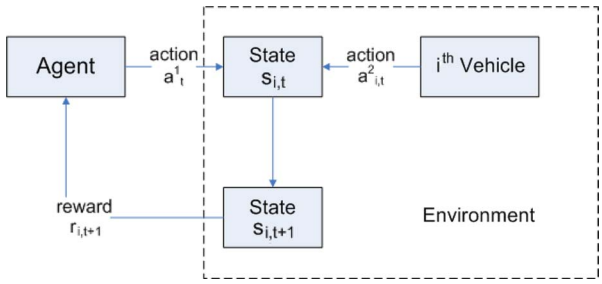

Fig. 5. General concept of DAQL.

$$\widetilde{v}_o = \begin{cases} \left((N_{vo} - 1) \times \left\lfloor |\vec{v}_o| \right) / v_{o,\max} \right\rfloor, & \text{for } 0 \le v_o < v_{o,\max} \\ N_{vo} - 1, & \text{for } v_o \ge v_{o,\max} \end{cases} \tag{9}$$

$$\widetilde{\theta}_o = \left\lfloor \frac{\theta_o}{2\pi/N_{\theta o}} \right\rfloor \tag{10}$$

where $v_{o,\max}$ is the maximum velocity of the vehicle. Since $|\vec{v}_o|$ and θ_o are quantized into N_{vo} and $N_{\theta o}$ states, respectively, there are, altogether, $(N_{\theta o} - 1) \times N_{vo} + 1$ actions for each vehicle, as observed by the agent. (For $|\vec{v}_o| = 0$, the vehicle is at rest, and there is one action only.)

B. Seven RL Goals

By definition, RL is the learning of decision making through the agent's interaction with the environment. An agent is responsible for the action decision process when it traverses across different environmental states. By making observations of the environment, it captures the current state, which then enables an action decision to be made accordingly. The action results in a reward received to update the value function that guides it to select future actions, which further maximizes future rewards.

QL by Watkins [26] is a simple model-free RL approach that allows the agent to learn to optimally act in the Markovian domain. However, due to its simplicity, it is not very effective in handling a dynamically changing environment, which requires the consideration of actions of other agents/obstacles in the environment [27]. For instance, Team QL [28] considered the actions of all the agents in a team and focused on the fully cooperative game in which all agents try to maximize a single reward function together. For agents that do not share the same reward function, Claus and Boutilier [29] proposed the use of joint action learners (JALs). JALs learn the value of their own actions in conjunction with those of other agents. Their results showed that, by taking into account the actions of another agent, JALs perform somewhat better than traditional QL. However, JALs crucially depend on the strategy adopted by the other agents and assume that other agents maintain the same strategy throughout, which may not be valid. Hu and Wellman proposed Nash QL [30] that focused on a general sum game that the agents are not necessarily working cooperatively. Nash equilibrium is used for the agent to adopt a strategy that is the best response to

TABLE I
SUMMARY OF THE SEVEN GOALS

	QL / DAQL	Q Value	State	Reward Function
Collision Avoidance	DAQL	Q_{CA}	$s_i = (\widetilde{d}_i, \widetilde{\theta}_i)$	$r_{CA,i,t} = \begin{cases} (d_i - d_i')/(v_{a,\max}T) & \text{if } d_i \text{ lies inside H} \\ 0 & \text{if } d_i \text{ lies outside H} \end{cases}$ (11)
Target Seeking	QL	Q_{TS}	$\widetilde{\phi}_t$	$r_{TS,t} = (\lvert\theta_T'\rvert - \lvert\theta_T\rvert)/\theta_{s,\max}$ (12)
Lane Following	QL	Q_{LF}	$s_{LF} = (\widetilde{d}_{LF}, \widetilde{\theta}_{LF})$	$r_{LF,t} = 0.5\times(\lvert d'_{LF}\rvert - \lvert d_{LF}\rvert)/(v_{a.\max}T) + 0.5\times(\lvert\theta'_{LF}\rvert - \lvert\theta_{LF}\rvert)/(\theta_{s,\max})$ (13)
Slow Lane	QL	Q_{SL}	$s_{SL} = (\widetilde{d}_{SL}, \widetilde{\theta}_{LF})$	$r_{SL,t} = \begin{cases} \dfrac{\lvert d'_{SL}\rvert - \lvert d_{SL}\rvert}{v_{a,\max}T} & \text{if } -\dfrac{d_{lane}}{2} \le d_{SL} \le (n-\dfrac{1}{2})d_{lane} \\ -1 & \text{otherwise} \end{cases}$ (14)
Fast Lane	QL	Q_{FL}	$s_{FL} = (\widetilde{d}_{FL}, \widetilde{\theta}_{LF})$	$r_{FL,t} = \begin{cases} \dfrac{\lvert d'_{FL}\rvert - \lvert d_{FL}\rvert}{v_{a,\max}T} & \text{if } -\dfrac{d_{lane}}{2} \le d_{FL} \le (n-\dfrac{1}{2})d_{lane} \\ -1 & \text{otherwise} \end{cases}$ (15)
Steady Speed	QL	Q_{SS}	$s_{SS,t} = \widetilde{v}_a$	$r_{SS,t} = (v_a - v_a')/a$ (16)
Steady Steering Angle	QL	Q_{SA}	$s_{SA,t} = \Delta\widetilde{\theta}_a$	$r_{SS,t} = 1 - \lvert\Delta\theta_a - \Delta\theta'_a\rvert/\theta_{s,\max}$ (17)

the other's strategy. This approach requires the agent to learn others' Q-value by assuming that the agent can observe others' rewards, which is hard in practice. DAQL, as proposed in [23], is an improved version of QL by simultaneously considering the agent's own action and other agents' actions. Instead of assuming that the rewards of other agents can be observed, it uses a probabilistic approach to predict their actions so that they may cooperatively, competitively, or independently work. Fig. 5 shows the general concept of DAQL.

1) Collision Avoidance: The function of collision avoidance is to enable the agent to avoid collision with the other vehicles or obstacles. Given multiple mobile vehicles in the environment, DAQL is most applicable here. The DAQL update rule shown in (18) is used to update the Q-values $(q_i(s_{i,t}, a_t^1, a_{i,t}^2))$ of each ith vehicle, which represent the action values in different states

$$q_i\left(s_{i,t}, a_t^1, a_t^2\right) \leftarrow q_i\left(s_{i,t}, a_t^1, a_t^2\right) + \alpha\left[r_{\mathrm{CA},i,t+1} + \gamma \max_{a_{t+1}^1} q_i \times \left(s_{i,t+1}, a_{t+1}^1, a_{t+1}^2\right) - q_i\left(s_{i,t}, a_t^1, a_t^2\right)\right] \quad (18)$$

where $s_{i,t}$ and $s_{i,t+1}$ are the input states, a_t^1 and a_{t+1}^1 are the actions of the agent, and a_t^2 and a_{t+1}^2 are the actions of the vehicles in t and $t+1$, respectively. α and γ are the weighting parameter and discount rate, respectively, which both range from 0 to 1. Define d_i as the distance between the agent and the ith vehicle at the current time step and d_i' as the distance in the previous time step; the reward function adopted by DAQL is defined in (11) in Table I, where T is the sampling time, and therefore, the reward is normalized by $v_{a,\max}T$, where it is the maximum distance that the agent can travel in one time step. H is a hexagonal area surrounding the agent, as shown in Fig. 6, which is defined by d_{width}, d_{length}, d_{f}, and d_{b}. d_{width} and d_{length} are the width and length of the agent, respectively. The hexagonal shape allows the agent to move away from other vehicles adopting a linear path apart from a curved path when a circular shape is used. d_f is defined by

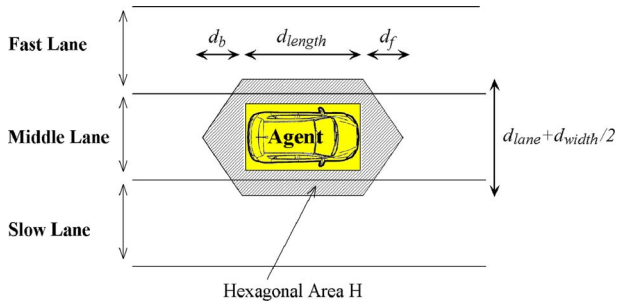

Fig. 6. H surrounds the agent for the calculation of the reinforcement signal.

$$d_f = v_a^2/2a + v_a/2 + 2d_{\mathrm{length}} - v_{o,i}^2/2a \quad (19)$$

and it is the minimum distance that the agent should keep between a vehicle moving in front of it such that the two vehicles will keep a distance of d_{length} apart if the front vehicle suddenly stops and the agent responds and brakes after a reaction time of 0.5s. d_b is defined similarly in

$$d_b = v_{o,i}^2/2a + v_{o,i}/2 + 2d_{\mathrm{length}} - v_a^2/2a \quad (20)$$

but it deals with vehicles behind the agent.

In this case, a negative reward is given either when a collision occurs or when the distance between the agent and other vehicles, which lies inside area H, decreases. When $r_{\mathrm{CA},i,t}$ is available, the agent uses the DAQL update rule to learn collision avoidance with each vehicle in the environment. Given the vehicle's actions in two time steps [$(t-2)$ and $(t-1)$], the

agent updates its Q-values $(q_i(s_{i,t}, a_t^1, a_{i,t}^2))$ at t. Since there are multiple numbers of vehicles in the environment, we elect to use the parallel QL concept [23], [31], [32], in which all M obstacles share a single set of Q-values; therefore, the Q-values are updated M times in one time step.

Apart from learning, the agent also needs to determine its own action in the current time step. Given the state information of the current time step, the agent can use it together with the Q-values to determine an action that is most appropriate for the navigation task. That is, the agent needs to determine an action a_t^1, given $q_i(s_{i,t}, a_t^1, a_{i,t}^2)$. However, since $a_{i,t}^2$ is not known at t, it has to be predicted, which can be independently treated from RL. To predict $a_{i,t}^2$, the AR model is adopted. We assume that accelerations of obstacles are slowly changing in the time interval T between two time steps. A first-order AR model [33], [34] is used to model the acceleration $a_i(t)$

$$a_i(t) = B_{i,t} a_i(t-1) + e(t) \tag{21}$$

where $e(t)$ is the prediction error, and $B_{i,t}$ is a time-dependent coefficient and is adaptively estimated according to the new distance measurements. The acceleration is thus approximated by a combination of velocity and position representations, i.e.,

$$\begin{aligned} a_i(t) &= \frac{1}{T}\left[v_i(t) - v_i(t-1)\right] \\ &= \frac{1}{T^2}\left\{\left[r_i(t) - r_i(t-1)\right] - \left[r_i(t-1) - r_i(t-2)\right]\right\} \\ &= \frac{1}{T^2}\left[r_i(t) - 2r_i(t-1) - r_i(t-2)\right] \end{aligned} \tag{22}$$

where $v_i(t)$ and $r_i(t)$ are the velocity and position of the ith vehicle at t, respectively. Substituting (21) into (22) gives a third-order AR model, i.e.,

$$\begin{aligned} r_i(t) - (2 + B_{i,t}) r_i(t-1) + (2B_{i,t} + 1) r_i(t-2) \\ - B_{i,t} r_i(t-3) = e(t). \end{aligned} \tag{23}$$

The next position of the ith vehicle at $(t+1)$ can be predicted by the following equation if the coefficient $B_{i,t}$ is known:

$$\hat{r}_i(t+1) = r_i(t) + v_k(t)T + \hat{B}_{i,t} a_i(t) T^2 \tag{24}$$

where $\hat{B}_{i,t}$ is time dependent and is updated as follows [35]:

$$\hat{B}_{i,t} = \Delta_{i,t} R_{i,t}^{-1} \tag{25}$$

$$\Delta_{i,t} = \lambda \Delta_{i,t-1} + a_i(t) a_i^T(t-1) \tag{26}$$

$$R_{i,t} = \lambda R_{i,t-1} + a_i(t-1) a_i^T(t-1) \tag{27}$$

where $0 < \lambda \leq 1$ is a weighting factor close to 1. Since $a_i(t)$, $\Delta_{i,t}$, $R_{i,t}$, and λ are all known, $\hat{B}_{i,t}$ can be predicted, and thus, $\hat{r}_i\ (t+1)$ can be predicted, from which the action performed by the ith vehicle at t can be predicted. Thus, probability $p_{a_{i,t}^2}$ of the ith vehicle performing the action $a_{i,t}^2$ can be determined. A probability of 1 is given to the predicted action, and 0 is given to all other actions. On the other hand, if the evenly distributed probability model is used, $p_{a_{i,t}^2}$ is equal to $1/N_0$, and N_0 is the number of actions observable by the agent. To incorporate the predicted $a_{i,t}^2$, the corresponding Q-value can be acquired as follows:

$$q_i\left(s_{i,t}, a_t^1\right) = \sum_{a_t^2} p_{a_t^2} q_i\left(s_{i,t}, a_t^1, a_{i,t}^2\right). \tag{28}$$

The expected value of the overall Q-value is obtained by summing the product of the Q-value of the vehicle when it takes action $a_{i,t}^2$ with its probability of occurrence. Finally, the summation of the Q-values from all the detected vehicles is the overall Q-value set for the entire vehicle population, i.e.,

$$Q_{\mathrm{CA}}\left(a_i^1\right) = \sum_t q_i\left(s_{i,t}, a_t^1\right). \tag{29}$$

2) Target Seeking: The function of target seeking is to enable the agent to reach the target. The reward function is designed such that the agent receives the maximum reward if it is moving toward the target using the line-of-sight path. For convenience, the target is assumed to be stationary, which suffices for QL. If the target is nonstationary, actions performed by the moving target may be considered as in the case of the vehicle, which the same DAQL formulation applies [22]. We use $\mathrm{QL}(Q, s_t, s_{t+1}, r_{t+1})$ to represent the learning of Q-value with the input state s_t and reward function r_{t+1}. The general QL update rule is given as follows:

$$\begin{aligned} \mathrm{QL}(Q, s_t, s_{t+1}, r_{t+1}) : Q\left(s_t, a_t^1\right) \leftarrow Q\left(s_t, a_t^1\right) \\ + \alpha\left[r_{t+1} + \gamma \max_{a_{t+1}^1} Q\left(s_{t+1}, a_{t+1}^1\right) - Q\left(s_t, a_t^1\right)\right]. \end{aligned} \tag{30}$$

The QL algorithm applied for target seeking is therefore $\mathrm{QL}(Q_{\mathrm{TS}}, \widetilde{\phi}_t, \widetilde{\phi}_{t+1}, r_{\mathrm{DS},t+1})$, where $\widetilde{\phi}_t$ and $\widetilde{\phi}_{t+1}$ are the input states. $r_{\mathrm{TS},t+1}$ describes the reward received by the agent generated from the reward function at time $t+1$. We define θ_T as $\theta_T = \theta_a - \phi$ and $0 \leq \theta_T \leq \pi$. Furthermore, θ_T is the angle measured in the current time step, and θ'_T is the angle measured in the previous time step. The normalized reward function for target seeking is defined in (12) in Table I, where $-1 \leq r_{\mathrm{TS},t} \leq 1$. The reward function is based on the idea that higher reward will be given if the agent is moving within line of sight toward the target. The reward proportionally decreases if the vehicle is heading away from the target.

3) Lane Following: The function of lane following is to enable the agent to closely follow the lane. This can be indicated by d_{LF} and θ_{LF}. If the agent is properly performing lane following, d_{LF} and θ_{LF} are close to zero. The basic idea behind then is to minimize the distance d_{LF} and align the agent parallel to the lane according to the reinforcement signal received. A reward is received if the agent moves toward the center of the lane; otherwise, a punishment is received. Because the lane-following goal is static in nature, QL is sufficient for this purpose.

The QL update rule used in the lane-following goal is $\mathrm{QL}(Q_{\mathrm{LF}}, s_{\mathrm{LF},t}, s_{\mathrm{LF},t+1}, r_{\mathrm{LF},t+1})$. Define d'_{LF} as the distance in the previous time step, i.e., the reward function that determines the reinforcement signal is defined by (13) in Table I.

A higher reward will be given if the agent is moving tangential to the lane and with a high speed.

4) Changing to Slow Lane and Fast Lane: Different from lane following, lane changing results in decreasing d_{LF} toward a specific lane (fast/slow). In this paper, there are two lane-change goals: 1) Change to the slow lane, and 2) change to the fast lane. The fast-lane goal is primarily designed for the agent to overtake from a slow lane using the fast lane but not *vice versa*; the slow-lane goal allows the agent to move back to the slow lane after overtaking. QL is used for the agent to learn to achieve the two goals. The QL update rule applied is $\mathrm{QL}(Q_{\mathrm{SL}}, s_{\mathrm{SL},t}, s_{\mathrm{SL},t+1}, r_{\mathrm{SL},t+1})$ and $\mathrm{QL}(Q_{\mathrm{FL}}, s_{\mathrm{FL},t}, s_{\mathrm{FL},t+1}, r_{\mathrm{FL},t+1})$, respectively.

Define d_{SL} as the distance between the agent and the slow lane, as shown in Fig. 4(a), and d'_{SL} as the distance in the previous time step; the reward function that determines the reinforcement signal of the slow lane goal is given by (14) in Table I, where n is the number of lanes in the road.

Similarly, define d_{FL} as the distance between the agent and the fast lane and d'_{FL} as the distance in the previous time step; the reward function that determines the reinforcement signal of the fast-lane goal is given by (15) in Table I. Therefore, higher reward will be given if the distance between the agent and fast lane is decreasing.

5) Steady Speed: The purpose of the steady-speed goal is to enable the agent to move at the highest available speed while keeping acceleration/deceleration as little as possible. A QL algorithm is assigned to handle the goal and can be described as $\mathrm{QL}(Q_{\mathrm{SS}}, s_{\mathrm{SS},t}, s_{\mathrm{SS},t+1}, r_{\mathrm{SS},t+1})$, where $s_{\mathrm{SS},t}$ is the state that refers to the discrete speed of the agent (v_a) adopted in the previous time step. Define v_a as the agent speed in the current time step and v'_a as the speed in the previous time step; the reward function that determines the reinforcement signal is defined by (16) in Table I, where a is the maximum acceleration/deceleration rate of the agent. The reward function is designed such that a reward is assigned to the vehicle if it is accelerating. A penalty is given if it is decelerating.

6) Steady Heading Angle: The purpose of the steady-heading-angle goal is to enable the agent to move with as little change in heading angle as possible. A QL algorithm is assigned to handle the goal and can be described as $\mathrm{QL}(Q_{\mathrm{SA}}, s_{\mathrm{SA},t}, s_{\mathrm{SA},t+1}, r_{\mathrm{SA},t+1})$, where $s_{\mathrm{SA},t}$ is the state that refers to the 15 discrete reference heading angle of the agent $(\Delta\theta_a)$ adopted in the previous time step. The reward function is proportional to the change in heading angle $(\Delta\theta_a - \Delta\theta'_a)$. A reward of 1 will be given to the vehicle if the change in heading angle is zero. The reward decreases as the change increases. A reward of 0 is given if the change is equal to the heading angle limit $(\theta_{s,\max})$, as given by (17) in Table I.

IV. Autonomous Navigation Among Moving Vehicles

The proposed MGRL approach considers seven goals in total to solve the overtaking task. To achieve an overall consistence action decision based on the individual action decisions, a goal fusion mechanism is applied to merge the results of the goals. The weighted sum method is used to combine different goals, and three modes are used to adjust the weights. The three navigation modes are given as follows: 1) Change to the slow lane; 2) change to the fast lane; and 3) stay in the current lane. When the change to slow-lane mode is adopted, the agent switches off the lane following and fast-lane goal for it to smoothly move toward slow lane. When the change to fast-lane mode is adopted, the agent switches off lane following and slow lane for it to smoothly move toward fast lane. Finally, when the agent adopted the stay at the current lane mode, the agent switches off slow- and fast-lane mode and stays in the current lane.

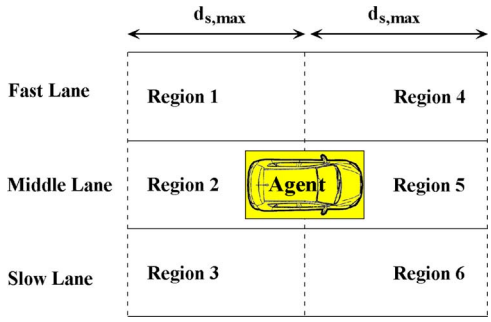

Fig. 7. Agent and its nearby regions.

TABLE II
Definition of $B_{\mathrm{Fast},j}$ and B_{Slow} for Lane Changing

	Region j	1	2	3	4	5	6
If vehicle exists	$B_{Fast,j}$	b_1	b_2	1	b_4	1	1
	$B_{Slow,j}$	1	b_2	b_3	1	0	0

To determine which navigation mode to be used, first, we divide the surrounding area of the agent into six regions relative to the agent and aligned with the highway lanes, as shown in Fig. 7. Each region j is assigned with two Boolean values $B_{\mathrm{Fast},j}$ and $B_{\mathrm{Slow},j}$ to describe the lane-changing rules in real-life road networks (see Table II). A "1" means that a change to the fast/slow lane is allowed, and a "0" means that the change is not allowed if there are vehicles in the corresponding region. Changing to slow/fast lane is always allowed if there are no vehicles nearby. We further define b_j as a Boolean value, of which "0" indicates that the vehicles in region j has caused a negative q-value according to the collision avoidance goal $(q_i(s_{i,t}, a_t^1))$ in (28) and "1" otherwise. If there are vehicles that could cause potential collisions (negative q-value of the collision avoidance goal) in a certain region, then lane changing to that region is not allowed. A rule is designed such that overtaking using the slow lane is not allowed, and therefore, a value of "0" is given to $B_{\mathrm{Slow},5}$ and $B_{\mathrm{Slow},6}$. The proposed method only requires the identification of fast/slow lane and nearby regions with respect to the current lane and, therefore, can be applied to roads with an arbitrary number of lanes.

Therefore, a change to the fast/slow lane is only allowed if the rules in all the regions allow the change, i.e., if B_{CF} or B_{CS} is true, where $B_{\mathrm{CF}} = \prod_{j=1}^{6} B_{\mathrm{Fast},j}$.

The agent then selects to use one of the three modes according to the following rule. The agent changes to a fast lane if b_5

TABLE III
THREE NAVIGATION MODES AND CORRESPONDING VALUES OF β

	β_{CA}	β_{TS}	β_{LF}	β_{SL}	β_{FL}	β_{SS}	β_{SA}
Change to Slow Lane	1	1	0	1	0	1	1
Change to Fast Lane	1	1	0	0	1	1	1
Keep Current Lane	1	1	1	0	0	1	1

is false and B_{CF} is true, and it changes to a slow lane if B_{CS} is true; else, it stays at the current lane. Therefore, overtaking will be performed if the agent determined from the Q-value that a collision is expected to happen if it continues to move straight forward and changing to fast lane is safe. Other goals such as collision avoidance are always turned on to make sure that the agent avoids collisions all the time.

Once the mode to be used has been determined, we can combine the Q-values from different goals according to the following method:

$$Q_{\mathrm{final}}\left(a_t^1\right) = Q_N\left(a_t^1\right)\mathbf{W} \tag{31}$$

where $Q_N(a_t^1)$ is a vector of normalized Q-values from different goals, as shown in the following, given that the current state for each goal is known:

$$Q_N\left(a_t^1\right) = \left[\frac{Q_{\mathrm{CA}}\left(a_t^1\right)}{\sum_{a^1}|Q_{\mathrm{CA}}(a^1)|}\frac{Q_{\mathrm{TS}}\left(a_t^1\right)}{\sum_{a^1}|Q_{\mathrm{TS}}(a^1)|}\frac{Q_{\mathrm{LF}}\left(a_t^1\right)}{\sum_{a^1}|Q_{\mathrm{LF}}(a^1)|}\right.$$
$$\left.\times\frac{Q_{\mathrm{SL}}\left(a_t^1\right)}{\sum_{a^1}|Q_{\mathrm{SL}}(a^1)|}\frac{Q_{\mathrm{FL}}\left(a_t^1\right)}{\sum_{a^1}|Q_{\mathrm{FL}}(a^1)|}\frac{Q_{\mathrm{SS}}\left(a_t^1\right)}{\sum_{a^1}|Q_{\mathrm{SS}}(a^1)|}\frac{Q_{\mathrm{SA}}\left(a_t^1\right)}{\sum_{a^1}|Q_{\mathrm{SA}}(a^1)|}\right]. \tag{32}$$

$\mathbf{W}$ is the weighting vector containing the weighting parameters for each goal as given: $\mathbf{W} = [\beta_{\mathrm{CA}}\ \beta_{\mathrm{TS}}\ \beta_{\mathrm{LF}}\ \beta_{\mathrm{SL}}\ \beta_{\mathrm{FL}}\ \beta_{\mathrm{SS}}\ \beta_{\mathrm{SA}}]^T$, where β in $\mathbf{W}$ are the parameters that vary between 0 and 1.

In QL or DAQL, each Q-value reflects the desirability of the agent to perform its associated action. Simply, the larger the Q-value is, the more desirable it is. In the case of a multiple-goal scenario, the Q-values determined from different goals are combined so that the one with the highest combined score carries the most desirable action. In this paper, we employed a weighted summation method to combine the different sets of Q-values having different weights to illustrate the different importance of these goals.

The three navigation modes allow the agent to adjust the set of weighting parameters (β values) according to the nearby situations. The relation between the three navigation modes and the weighting parameters (β values) is given in Table III. The goal associated with a higher value of β means that it has higher importance when considering the entire task as a whole and *vice versa*. It should be noted that some of these goals are contradictory to each other, e.g., lane following, slow lane, and fast lane. For instance, when the agent plans to overtake the leading vehicle, it needs to move to a faster lane, where β_{SL} is close to 0, and β_{FL} is close to 1.

Once the final Q-value has been calculated, the final decision of the agent is made by using the ε-greedy policy to allow proper exploration, i.e.,

$$a_t^1 = \begin{cases} \arg\max\limits_{a_t^1} Q_{\mathrm{final}}\left(a_t^1\right), & \text{with probability } 1-\varepsilon \\ \text{random}, & \text{with probability } \varepsilon. \end{cases} \tag{33}$$

V. RESULTS AND DISCUSSIONS

A. Simulation Environment

In the simulation, we assumed an agent/vehicle dimension of 1.7 m × 4.25 m ($d_{\mathrm{width}} \times d_{\mathrm{length}}$) with a maximum speed ($v_{a,\max}$) of 30 m/s and a maximum acceleration and deceleration of ± 3 m/s^2. A sensor simulator has been implemented to evaluate distances between the agent and other vehicles. It can produce either accurate or erratic distance measurements from $d_{s,\min} = 1$ m to $d_{s,\max} = 100$ m at T interval (typically 0.1 s) to simulate practical sensor limitations [36], [37]. The lane width in the simulation environment was assumed to be $d_{\mathrm{lane}} = 3.4$ m. The other parameters were set as follows: α is set to 0.6 for faster update of Q-values; γ is set to 0.9 for DAQL and 0.1 for QL; and ε is set to 0.2. Without lost of generality, we use the following values for quantizing the states gathered by the agent. In situations where high precision is required, the number of states can be further increased. N_d and N_θ are set to 15 and 16, respectively, for gathering vehicle information. N_ϕ of 15 is used for quantizing the heading angle. $N_{\theta LF}$ and N_{dLF} are set to 16 and 17, respectively, for the gathering of lane information. Finally, N_v and N_a of 5 and 15 are used, respectively, to represent the action of the agent, whereas N_{vo} and $N_{\theta o}$ of 6 and 16 are used, respectively, to represent the actions of other vehicles. We use roughly 15–17 partitions for spatial-related parameters and 5–6 partitions for speed-related parameters. More partitions are assigned to spatial-related parameters for better collision avoidance performance. Relatively fewer partitions are assigned to speed-related parameters to reduce the total number of state.

To learn to solve the overtaking problem, the agent was first trained in a test environment (see Fig. 25) for 1000 episodes. During training, the environment, which consisted of other moving vehicles, changed because of vehicle movements. One episode is defined as either the agent has reached the target (end of the road) or a maximum of 500 steps has been reached, where one time step represents the time interval of T. Exploration and learning were enabled during training but disabled afterward. To study the agent's learning property, we let it alternatively switch between exploration and no exploration every ten episodes. The average number of collisions per ten episodes is shown in Fig. 8. It shows that collisions mainly occurred in the first 400 episodes. Initially, the number of collision was small, because the agent was also learning other goals such as lane following, which means that the agent does not necessarily follow the lane; therefore, the chance of causing collision was small. Later, the agent learned the ability to follow lanes, and thus, the number of collision increases. After sufficient learning after about 400 episodes, the agent was able to complete rest of the episodes without collisions.

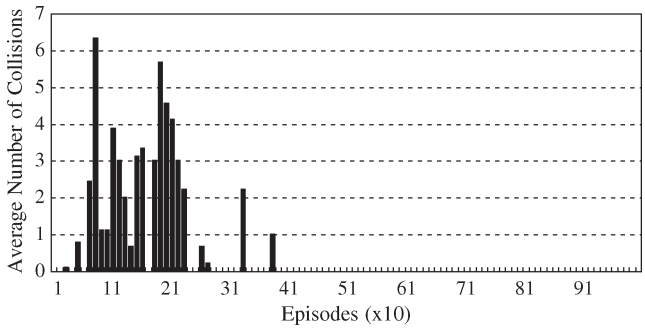

Fig. 8. Learning rate of the agent.

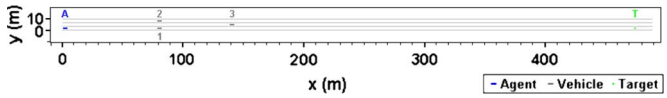

Fig. 9. Original locations of the agent and other vehicles.

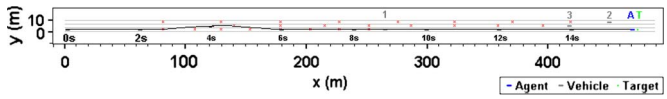

Fig. 10. Path and historical locations of the agent and other vehicles.

When applying the theoretical discrete action output on a realistic vehicle, the change in heading angle is smoothened to have more gradual changes to meet the vehicle dynamic.

B. Case 1: Straight Road With Other Vehicles Moving at Constant Speed

With reference to Fig. 9, the agent was originally located at (2.1, 1.7) and traveled at a maximum speed of 30 m/s toward the target at (474, 1.7). There were three straight lanes in the environment, and there were three vehicles. Vehicles 1, 2, and 3 began at (82, 1.7), (141, 5.1), and (82, 8.5) and traveled at constant speeds of 6, 18, and 24 m/s, respectively. The aim of this simulation is to demonstrate how the agent overtakes a slow-moving vehicle. The paths taken by the agent and other three vehicles are shown in Fig. 10. The historical positions of all the vehicles are depicted as red crosses in 1-s intervals.

The simulation result shows that the agent began overtaking after $t \approx 2$ s when it was just behind vehicle 1. It moved to the middle and faster lane at $t \approx 4$ s, at which time, it was almost in parallel with and nearest to vehicle 1 and some distance behind vehicle 2. At this point, the agent also commenced the action of returning back to its original lane. At $t \approx 6$ s, the agent returned to the slow lane, just ahead of vehicle 1. As vehicle 1 moved slower than the agent, the agent maintained its own speed until the target was reached. This completed the overtaking maneuver without causing a collision or any abrupt changes in speed and heading angle. In these instances, both vehicles 2 and 3 were some distance or lane away and had little impact on the overtaking decision. Throughout the journey, the agent traveled at maximum speed, as shown in Fig. 11. From Fig. 12, it can be seen that the agent started overtaking (a gentle left turn) at $t = 2.4$ s and completed the turn at $t = 4.4$ s. It then

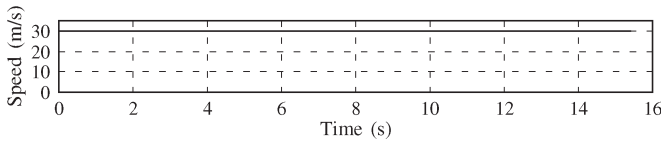

Fig. 11. Speed profile of the agent.

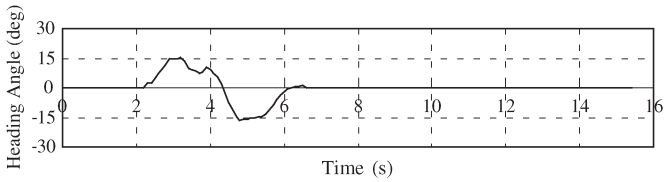

Fig. 12. Heading angle profile of the agent.

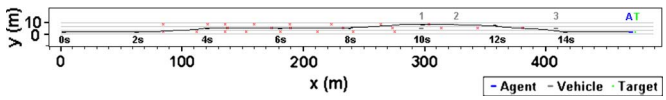

Fig. 13. Path and historical locations of the agent and other vehicles.

began to move back to the slow lane as part of the overtaking action and completed the whole action at $t = 6.4$ s.

C. Case 2: Straight Road With Other Vehicles Moving at Variable Speed

In this case, the road configuration is the same as Case 1, except that the three other vehicles were traveling at variable speed that range between 6 and 24 m/s, with acceleration/deceleration of ± 2 m/s^2. As shown in Fig. 13, this variable speed per vehicle is depicted as unequal distances traveled between the 2-s intervals. In performing the overtaking action, the agent had to consider the distances of other vehicles in its overtaking path, and in this case, when it first overtook vehicle 1 at $t = 2.4$ s, the situation was very similar to Case 1. If vehicle 2 traveled faster or vehicle 1 traveled slower at this point, the agent would probably have returned to the slow lane in its next move. However, vehicle 2 was slowly traveling, and so was vehicle 3; as overtaking from the slow lane is not permitted, it triggered the agent to move to the fast lane at $t = 7.2$ s. At this point, the agent was almost next to vehicle 2, with vehicle 3 some distance behind it in the same lane. As the target was at the end of the slow lane, the target-seeking goal became more important. As a result, it turned right and overtook vehicle 2 before reaching the target. This completed the overtaking maneuver without causing collision or any abrupt changes in speed and heading angle. In this case, it is probable that the agent could slightly slow down and let vehicles 1 and 3 pass first before doing the same thing as shown in Fig. 14 at 3.4 and 7.6 s. From Fig. 15, it can be seen from the heading angle profile that the overtaking began at about $t = 2.4$ s, turned left again at $t = 7.8$ s, and commenced a right turn at $t = 11$ s.

D. Case 3: Curve Road With Other Vehicles Moving at Variable Speed

In this case, a curved road with three lanes was considered. As shown in Fig. 16, the target at (245, 226) was at the end of

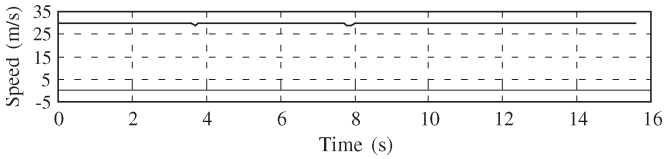

Fig. 14. Speed profile of the agent.

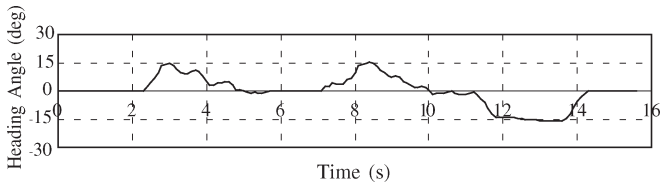

Fig. 15. Heading angle profile of the agent.

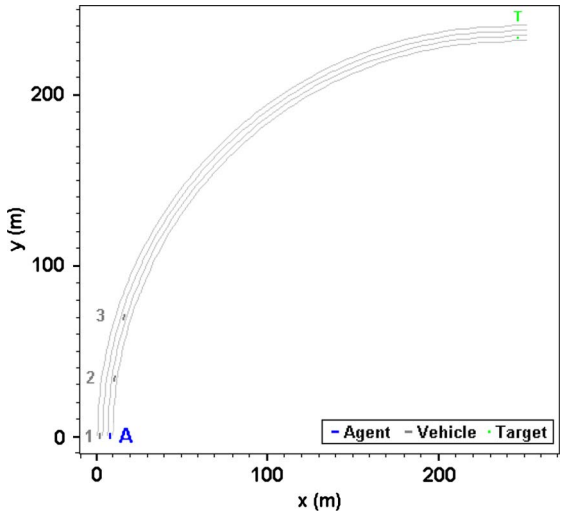

Fig. 16. Original locations of the agent and other vehicles.

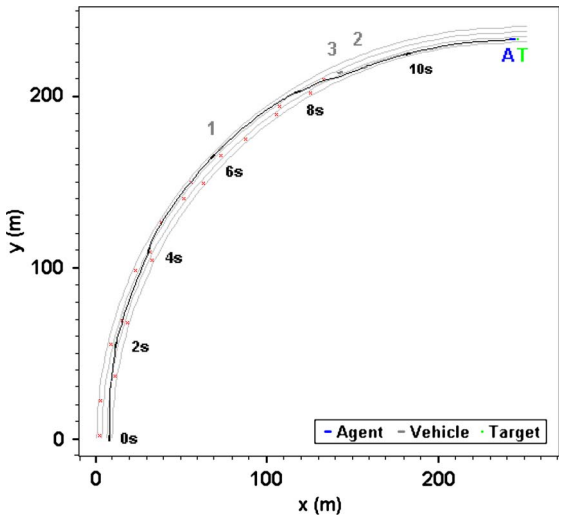

Fig. 17. Path and historical locations of the agent and other vehicles.

the slow lane and the agent, and vehicles 1 (fast lane), 2 (slow lane), and 3 (middle lane) were initially located at (8.5, 0), (1.7, 0), (10, 33), and (14, 69), respectively. The maximum speed of the agent was 30 m/s, whereas vehicles 1, 2, and 3 were traveling at variable speed as in Case 2. Fig. 17 shows

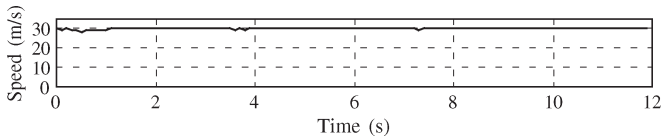

Fig. 18. Speed profile of the agent.

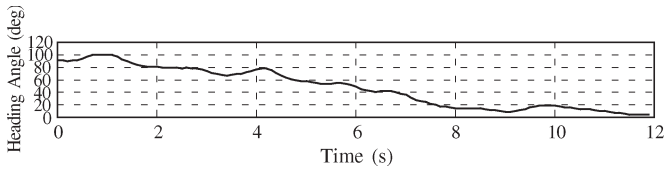

Fig. 19. Heading angle profile of the agent.

the paths of the agent and vehicles and their historical positions in 1-s intervals. As can be seen in Fig. 17, the agent overtook vehicle 2 rather quickly, because it was traveling at relatively low speed. Since the agent was initially located very close to vehicle 2, the agent needed to adjust its speed to ensure there was enough space for it to overtake, thus resulting in the agent slowing down. At about $t = 2$ s, the agent was in the middle lane, some distance behind vehicle 3 and slightly before vehicle 2 in the slow lane, whereas vehicle 1 was the slowest on the fast lane. As the agent caught up with vehicle 3 at about $t = 4$ s, vehicle 2 was almost next to it. Due to collision avoidance and overtaking must be in the fast lane, the agent moved into the fast lane, because vehicle 1 was some distance behind; however, vehicle 3 was too close in front. The agent stayed in the fast lane until roughly $t = 7$ s, when it was almost next to vehicle 3; it began moving back to the middle lane and slow lane to reach the target at the slow lane. This completed the overtaking maneuver, causing neither a collision nor any abrupt changes in speed and heading angle. From Fig. 18, the agent slowed to avoid running into vehicle 2 initially and did that again when it moved to the fast lane from the middle and moved to the middle lane from the fast lane. From Fig. 19, as the agent has to regularly keep changing its heading angle over the curvature, its angle profile is no longer constant, which is the case here. It appears that the additional lane-changing moves during the overtaking action are evident but not excessive.

E. Case 4: A Complex Overtaking Maneuver

In this case, a more complicated road configuration was used with eight other vehicles involved. Again, these vehicles were traveling at variable speed as in Case 2 and at different initial positions, as shown in Fig. 20. The entire path of the agent and its historical locations are shown in Fig. 21, with the final locations of the other vehicles shown as a reference. From this set of data, four time instants were further investigated, as shown in Fig. 21(a)–(d). Fig. 22 shows the location of the agent with respect to the slow lane.

First, Fig. 21(a) shows the agent's and other vehicles' positions at $t = 7.5$ s. This was when the agent overtook vehicle 5 by moving from the slow lane to the middle lane. The actual maneuver was about 6.2 s from Fig. 23, and the agent ended up in front of vehicle 4 but behind vehicle 6. Second, Fig. 21(b)

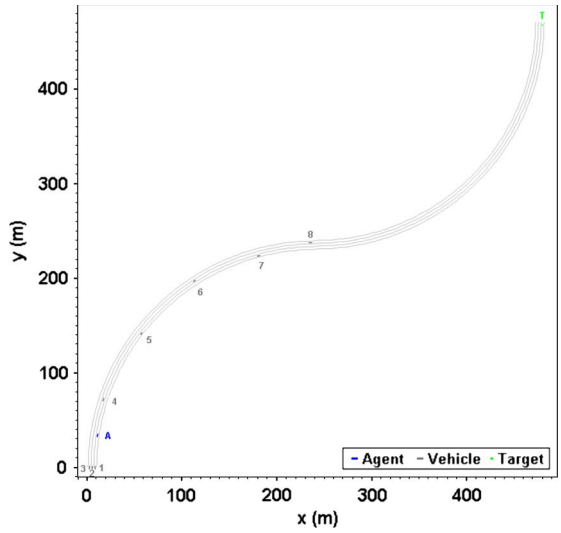

Fig. 20. Original locations of the agent and other vehicles.

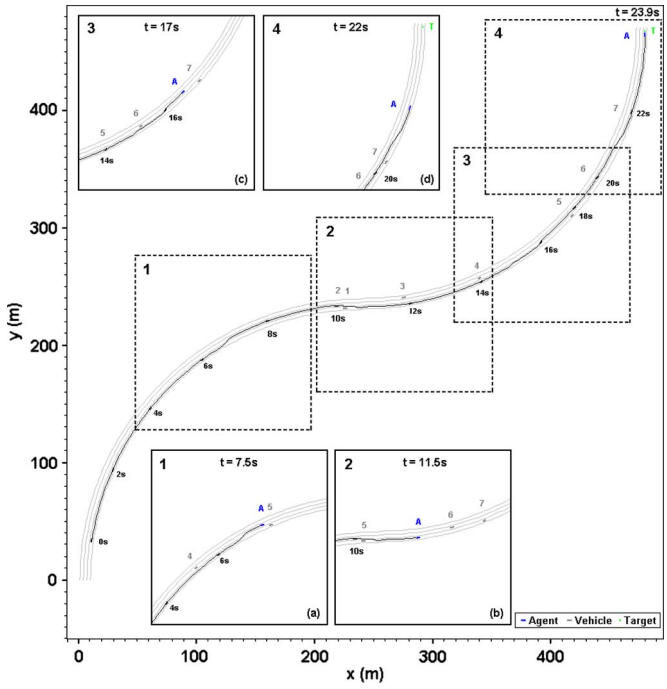

Fig. 21. Path and historical locations of agent and other vehicles.

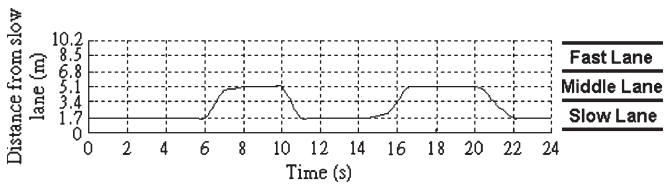

Fig. 22. Location of the agent with respect to the lanes on the road.

shows that the agent completed the overtaking maneuver by moving back to the slow lane, ahead of vehicle 5, at $t = 11.6$ s. If vehicle 6 was slower and closer to the agent, it would probably move to the fast lane. It should be noted that, after $t = 11.6$ s, the agent remained in the slow lane until $t = 17$ s. This was because vehicles 7 (slow lane) and 6 (middle lane) were near each other for a while. At present, the proposed

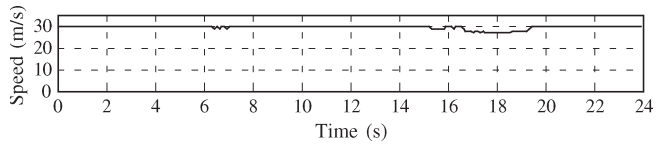

Fig. 23. Speed profile of the agent.

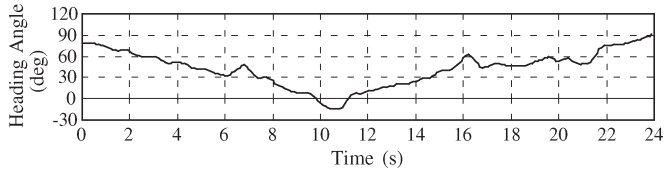

Fig. 24. Heading angle profile of the agent.

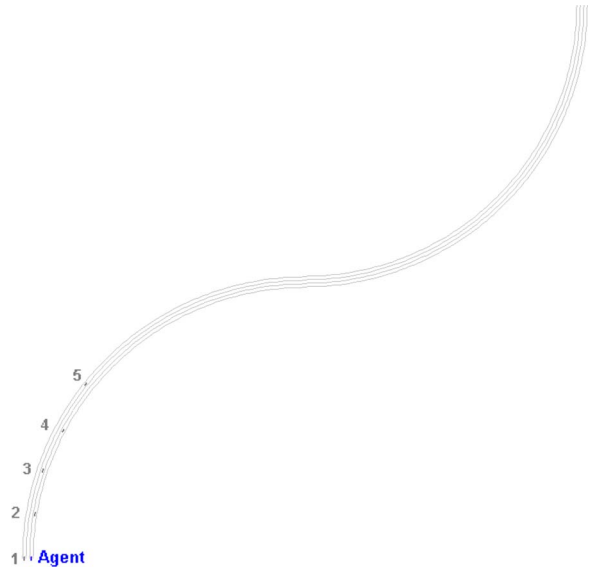

Fig. 25. Evaluation environment.

method is not able to overtake across three lanes; therefore, the agent slowed its speed, as can be seen in Fig. 23, between $t = 15{-}17$ s and waited for an opportunity.

Third, at $t = 17$ s, as shown in Fig. 21(c), the agent detected that vehicles 6 and 7 had sufficient distance between them so that it could overtake them without causing a collision. It should be noted that the agent gradually speeded up to achieve a safe maneuver. Fourth, as shown in Fig. 21(d), at $t = 22$ s, the agent completed the maneuver by moving back to the slow lane when the target was reached. Similar to all the cases simulated, the agent completed a series of complex overtaking maneuvering without causing any collisions. The heading angle profile given in Fig. 24 does not indicate any abrupt changes, but the speed profile in Fig. 23 indicates a drop in the agent's speed due to vehicles 6 and 7 blocking any overtaking actions at the time.

VI. Empirical Performance Evaluation

This section attempts to more specifically evaluate the agent's performance. The number of vehicles ranges from 0 to 5. Their initial locations are showing in Fig. 25. The maximum speed of the agent was 30 m/s, and other vehicles traveled at variable speed between 6 and 24 m/s. In theory, the shortest path

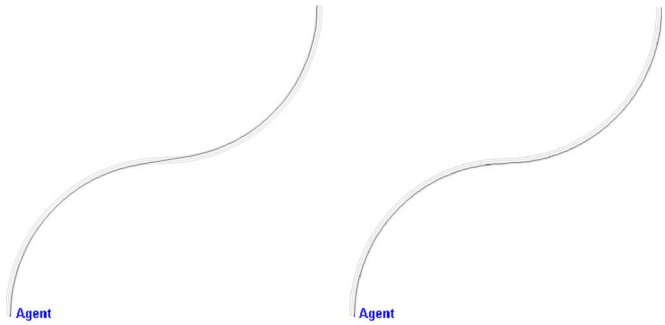

Fig. 26. Shortest path available. (a) Shortest path. (b) Path following slow lane.

is $p_s = 727$ m, as shown in Fig. 26(a). On the other hand, the path following the slow lane is 738 m, as shown in Fig. 26(b). During learning, the agent is placed in an environment that is the same as in Fig. 25, which has five vehicles, and with vehicles moving with various speed. To evaluate path performance, the agent performed 1000 episodes in each situation. One episode is defined as the agent reaching the end of the road. The length of the actual path is p_a, and the relative error between the actual path and the shortest path length $(p_a - p_s)/p_s$ is denoted by E_r.

To measure the quality of the path adopted by the agent, three indexes are used [34], [38].

1) Safety index (SI) $\bar{c}$, which represents the percentage of simulation episodes for the simulation task in which the agent successfully reached the target without collision. It is given by

$$\bar{c} = m/k \tag{34}$$

where m is the total simulation episodes without a collision, and k represents the total number of simulation episodes.

2) Steering smoothness index (SSI) $\bar{\omega}$, which represents the average absolute value of the heading angle of the agent over the simulation episodes. It is given by

$$\bar{\omega} = \left(\sum_{i=1}^{k} |\Delta \bar{\theta}_i| \right) / k \tag{35}$$

where $\Delta\bar{\theta}_i$ stands for the average heading angle in the ith simulation episode. The metric of $\bar{\omega}$ is in radians.

3) Velocity smoothness index (VSI) $\bar{a}$, which represents the average value of the velocity change of the agent over the simulation episodes. It is given by

$$\bar{a} = \left(\sum_{i=1}^{k} |\Delta \bar{v}_i| \right) / k \tag{36}$$

where $\Delta\bar{v}_i$ stands for the average change of the velocity in the ith simulation episode. The metric of $\bar{a}$ is expressed in meters per second.

The result of the empirical evaluation on the agent using the evenly distributed probability model is summarized in Table IV.

TABLE IV
PERFORMANCE OF THE AGENT USING THE EVENLY DISTRIBUTED PROBABILITY MODEL

No. of Vehicles	p_a (m)	p_a-p_s (m)	E_r	Time (s)	SSI (rad)	VSI (m/s)	SI
0	738.00	24.70	0.54%	23.60	0.0132	0.0000	1
1	747.17	33.86	0.74%	24.35	0.0219	0.0915	1
2	757.77	44.47	0.98%	24.68	0.0262	0.1030	1
3	757.79	44.48	0.98%	24.70	0.0263	0.1059	1
4	755.40	42.10	0.92%	24.87	0.0278	0.1314	1
5	737.14	23.84	0.52%	34.21	0.0296	0.7741	0.99

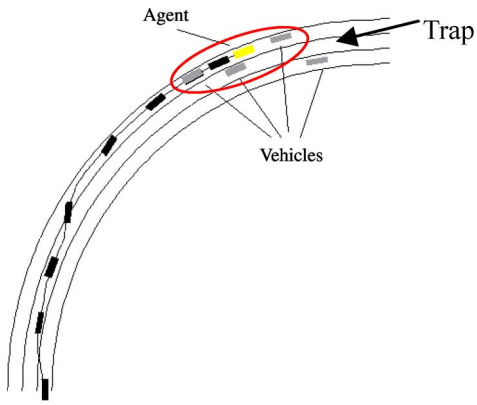

Fig. 27. Formation of "trap."

The evenly distributed probability model assumes that other vehicles have uniform probability of performing different actions. From Table IV, it can be seen that the agent initially adopted a path that is longer than the shortest path as the number of vehicles increases. However, as the number of vehicles is larger than 3, the actual path length decreases. This is because, as there are more and more vehicles on the road, the agent overtakes other vehicles using the fast lane more often, which results in a shorter path. The travel time, SSI, and VSI generally increase as number of vehicles increases. However, when there are five vehicles on the road, the travel time and VSI sharply increase. This is due to the formation of "trap," as shown in Fig. 27. This resulted in the agent having to abruptly and frequently slow down to avoid collision. Collision may be unavoidable in a "trap" situation if the other vehicles do not adjust their speed accordingly, and the SI in the environment with five obstacles is thus 0.99. However, it should be stressed that such situation will unlikely happen in the real world.

The result of the empirical evaluation on the agent using the AR model is summarized in Table V. When compared with Table III, it can be observed that, although the general result of using the AR model has slightly better path length, travel time, SSI, and VSI than using the evenly distributed probability model [23], the former has a lower SI in the corresponding situations, which is obviously not desirable. This can be explained as the AR prediction model allows the agent to adopt a more aggressive behavior and carry out collision avoidance behavior only if required. Therefore, the path length can be shorter and smoother. However, if the predicted results are incorrect, it may cause collisions. On the other hand, the

TABLE V
PERFORMANCE OF THE AGENT USING THE AR MODEL

No. of Vehicles	p_a (m)	p_a-p_s (m)	E_r	Time (s)	SSI (rad)	VSI (m/s)	SI
0	738.00	24.70	0.54%	23.60	0.0132	0.0000	1.000
1	745.86	32.56	0.71%	24.09	0.0211	0.0654	0.993
2	756.53	43.23	0.95%	24.55	0.0267	0.0957	0.988
3	756.36	43.06	0.95%	24.60	0.0271	0.0976	0.986
4	755.56	42.25	0.93%	24.50	0.0272	0.0947	0.977
5	736.98	23.68	0.52%	32.21	0.0272	0.5699	0.946

TABLE VI
ROBUSTNESS TO SENSOR NOISE

Sensor Noise Rate	SI
0	1.000
0.1	0.995
0.2	0.890
0.3	0.875
0.4	0.864
0.5	0.858

evenly distributed probability model gives a more conservative prediction, resulting in more conservative actions.

We have also carried out simulation to investigate how the proposed method tolerates inaccuracy in sensor measurements. The output of the sensor simulator was deliberately corrupted by a Gaussian noise function that has a mean μ of $\mu = d_i$ and standard deviation σ of $n \times \mu$, where $n = 0, 0.1, 0.2, 0.3, 0.4, 0.5$, and 0.6 [24] and is called the sensor noise rate. The initial location of the vehicles is the same as in Fig. 25. Vehicles are moving at variable speed, as in the previous case. For each set of n, the agent was trained for 1000 episodes. After training, the agent was evaluated in the same environment 1000 times with different n's. Table VI depicts the simulation summary. It shows that the SI decreases by 14.2% for $n = 0.5$ when compared with the environment with no noise.

VII. CONCLUSION

In this paper, we have presented a novel MGRL method that can be used in general for making action decisions in AV navigation. A number of overtaking cases were simulated to demonstrate the effectiveness of the proposed method. By considering seven different goals, either QL or DAQL is employed to determine individual action decisions based on whether the other vehicle interacts with the agent for that particular goal. Furthermore, a fusion function is proposed to weigh the importance of each goal before arriving at a combined but consistent action decision. This offers a powerful approach for dealing with demanding actions such as overtaking, particularly when a number of other vehicles are within the proximity of the agent and are traveling at variable speed. From the results of the four simulation cases, it can be concluded that the proposed method is capable of the following: 1) making correct action decisions for overtaking; 2) avoiding collisions with other vehicles; 3) reaching the target in reasonable time; 4) keeping almost steady speed; and 5) maintaining an almost steady heading angle. In addition, it should also be noted that the proposed method performs lane keeping very well when not overtaking and lane changing effectively when overtaking is in progress. When prediction is concerned, a more conservative approach may be the correct choice. In terms of sensor noise, a larger safety margin should be given to maintain an SI of 1 in all cases.

REFERENCES

[1] A. Broggi, *Automatic Vehicle Guidance: The Experience of the ARGO Autonomous Vehicle*. Singapore: World Scientific, 1999.

[2] D. Driankov and A. Saffiotti, *Fuzzy Logic Techniques for Autonomous Vehicle Navigation*. Heidelberg, Germany: Physica-Verlag, 2001.

[3] J. P. How, B. Bethke, A. Frank, D. Dale, and J. Vian, "Real-time indoor autonomous vehicle test environment," *IEEE Control Syst. Mag.*, vol. 28, no. 2, pp. 51–64, Apr. 2008.

[4] C. Urmson and W. Whittaker, "Self-driving cars and the urban challenge," *IEEE Intell. Syst.*, vol. 23, no. 2, pp. 66–68, Mar./Apr. 2008.

[5] J. Feng, J. Ruan, and Y. Li, "Study on intelligent vehicle lane change path planning and control simulation," in *Proc. IEEE Int. Conf. Inf. Acquisition*, 2006, pp. 683–688.

[6] C. Hatipoglu, U. Ozguner, and K. A. Redmill, "Automated lane change controller design," *IEEE Trans. Intell. Transp. Syst.*, vol. 4, no. 1, pp. 13–22, Mar. 2003.

[7] H. Jula, E. B. Kosmatopoulos, and P. A. Ioannou, "Collision avoidance analysis for lane changing and merging," *IEEE Trans. Veh. Technol.*, vol. 49, no. 6, pp. 2295–2308, Nov. 2000.

[8] A. Kanaris, E. B. Kosmatopoulos, and P. A. Ioannou, "Strategies and spacing requirements for lane changing and merging in automated highway systems," *IEEE Trans. Veh. Technol.*, vol. 50, no. 6, pp. 1568–1581, Nov. 2001.

[9] T. Shamir, "How should an autonomous vehicle overtake a slower moving vehicle: Design and analysis of an optimal trajectory," *IEEE Trans. Autom. Control*, vol. 49, no. 4, pp. 607–610, Apr. 2004.

[10] F. H. Wang, M. Ying, and R. Q. Yang, "Conflict-probability-estimation-based overtaking for intelligent vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 10, no. 2, pp. 366–370, Jun. 2009.

[11] Y. Zhu, D. Comaniciu, M. Pellkofer, and T. Koehler, "Reliable detection of overtaking vehicles using robust information fusion," *IEEE Trans. Intell. Transp. Syst.*, vol. 7, no. 4, pp. 401–414, Dec. 2006.

[12] J. E. Naranjo, C. Gonzalez, R. Garcia, T. de Pedro, and M. A. Sotelo, "Using fuzzy logic in automated vehicle control," *IEEE Intell. Syst.*, vol. 22, no. 1, pp. 36–45, Jan./Feb. 2007.

[13] R. Garcia, T. de Pedro, J. E. Naranjo, J. Reviejo, and C. Gonzalez, "Frontal and lateral control for unmanned vehicles in urban tracks," in *Proc. IEEE Intell. Vehicle Symp.*, 2002, vol. 2, pp. 583–588.

[14] J. E. Naranjo, C. Gonzalez, R. Garcia, and T. de Pedro, "Lane-change fuzzy control in autonomous vehicles for the overtaking maneuver," *IEEE Trans. Intell. Transp. Syst.*, vol. 9, no. 3, pp. 438–450, Sep. 2008.

[15] R. S. Sutton and A. G. Barto, *Reinforcement Learning—An Introduction*. Cambridge, MA: MIT Press, 1998.

[16] D. C. K. Ngai and N. H. C. Yung, "Performance evaluation of double action Q-learning in moving obstacle avoidance problem," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, Waikoloa, HI, Oct. 2005, pp. 865–870.

[17] P. Fiorini and Z. Shiller, "Motion planning in dynamic environments using velocity obstacles," *Int. J. Robot. Res.*, vol. 17, no. 7, pp. 760–772, Jul. 1998.

[18] Z. Shiller, F. Large, and S. Sekhavat, "Motion planning in dynamic environments: Obstacles moving along arbitrary trajectories," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2001, vol. 4, pp. 3716–3721.

[19] C. C. Chang and K. T. Song, "Environment prediction for a mobile robot in a dynamic environment," *IEEE Trans. Robot. Autom.*, vol. 13, no. 6, pp. 862–872, Dec. 1997.

[20] S. S. Ge and Y. J. Cui, "Dynamic motion planning for mobile robots using potential field method," *Auton. Robots*, vol. 13, no. 3, pp. 207–222, Nov. 2002.

[21] D. C. K. Ngai, "Reinforcement-learning-based autonomous vehicle navigation in a dynamically changing environment," Ph.D. dissertation, Dept. Elect. Electron. Eng., Univ. Hong Kong, Hong Kong, 2008.

[22] D. C. K. Ngai and N. H. C. Yung, "Fast-maneuvering target seeking based on double-action Q-learning," in *Proc. 5th Int. Conf. MLDM Pattern Recog.*, Leipzig, Germany, Jul. 18–20, 2007, pp. 653–666.

[23] D. C. K. Ngai and N. H. C. Yung, "Automated vehicle overtaking based on a multiple-goal reinforcement learning framework," in *Proc. IEEE ITSC*, Seattle, WA, Sep. 2007, pp. 818–823.
[24] C. Ye, N. H. C. Yung, and D. W. Wang, "A fuzzy controller with supervised learning assisted reinforcement learning algorithm for obstacle avoidance," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 33, no. 1, pp. 17–27, Feb. 2003.
[25] Q. Li, N. N. Zheng, and H. Cheng, "Springrobot: A prototype autonomous vehicle and its algorithms for lane detection," *IEEE Trans. Intell. Transp. Syst.*, vol. 5, no. 4, pp. 300–308, Dec. 2004.
[26] C. J. C. H. Watkins and P. Dayan, "Technical Note: Q-learning," *Mach. Learn.*, vol. 8, no. 3, pp. 279–292, May 1992.
[27] M. L. Littman, "Value-function reinforcement learning in Markov games," *Cogn. Syst. Res.*, vol. 2, no. 1, pp. 55–66, Apr. 2001.
[28] C. Boutilier, "Planning, learning and coordination in multi-agent decision processes," in *Proc. 6th Conf. TARK*, Renesse, The Netherlands, 1996, pp. 195–201.
[29] C. Claus and C. Boutilier, "The dynamics of reinforcement learning in cooperative multiagent systems," in *Proc. 15th Nat. Conf. Artif. Intell.*, 1998, pp. 746–752.
[30] J. L. Hu and M. P. Wellman, "Nash Q-learning for general-sum stochastic games," *J. Mach. Learn. Res.*, vol. 4, pp. 1039–1069, Aug. 15, 2004.
[31] G. Laurent and E. Piat, "Parallel Q-learning for a block-pushing problem," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2001, vol. 1, pp. 286–291.
[32] G. J. Laurent and E. Piat, "Learning mixed behaviours with parallel Q-learning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2002, vol. 1, pp. 1002–1007.
[33] N. Kehtarnavaz and S. Li, "A collision-free navigation scheme in the presence of moving obstacles," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recog.*, Jun. 1988, pp. 808–813.
[34] C. Ye, "Behavior-based fuzzy navigation of mobile vehicle in unknown and dynamically changing environment," Ph.D. dissertation, Dept. Elect. Electron. Eng., Univ. Hong Kong, Hong Kong, 1999.
[35] M. J. Shensa, "Recursive least-squares lattice algorithms—A geometrical approach," *IEEE Trans. Autom. Control*, vol. AC-26, no. 3, pp. 695–702, Jun. 1981.
[36] C. Hartzstein, "76 GHz radar sensor for second generation ACC," in *Proc. 11th Int. Symp. ATA EL—New Technologies for Advanced Driver Assistance Systems*, Siena, Italy, Oct. 2002.
[37] H. Rohling and M. M. Meinecke, "Waveform design principles for automotive radar systems," in *Proc. IEEE CIE Int. Conf. Radar*, Beijing, China, Oct. 2001, pp. 1–4.
[38] J. Yen and N. Pfluger, "A fuzzy-logic based extension to Payton and Rosenblatts command fusion method for mobile robot navigation," *IEEE Trans. Syst., Man, Cybern.*, vol. 25, no. 6, pp. 971–978, Jun. 1995.

Daniel Chi Kit Ngai received the B.Eng. degree in information engineering and the Ph.D. degree in electrical and electronic engineering from the University of Hong Kong, Hong Kong, China, in 2003 and 2008, respectively.

He is currently a Senior Computer Vision Engineer with ASM Assembly Automation Ltd., Kwai Chung, Hong Kong. His research interests include artificial intelligence, learning systems, intelligent transportation systems, and computer vision.

Nelson Hon Ching Yung (M'85–SM'96) received the B.Sc. and Ph.D. degrees from the University of Newcastle-Upon-Tyne, Tyne, U.K.

From 1985 to 1990, he was a Lecturer with the University of Newcastle-Upon-Tyne. From 1990 to 1993, he was a Senior Research Scientist with the Department of Defence, Australia. In late 1993, he joined the Department of Electrical and Electronic Engineering, University of Hong Kong (HKU), Pokfulam, Hong Kong, as an Associate Professor. He is the Founding Director of the Laboratory for Intelligent Transportation Systems Research, HKU. He has coauthored six books and book chapters and has published more than 160 journal and conference proceeding papers in the areas of digital image processing, parallel algorithms, visual traffic surveillance, autonomous vehicle navigation, and learning algorithms. He acts as an expert in many law enforcement projects and in the courts of law in Hong Kong. He was a Guest Editor for the *SPIE Journal of Electronic Imaging*. He also serves as a reviewer for a number of IEEE, Institution of Engineering and Technology, and International Society for Optical Engineers journals.

Dr. Yung was member of the Advisory Panel of the Intelligent Transportation Systems (ITS) Strategy Review, Transport Department, HKSAR; Regional Secretary of the IEEE Asia-Pacific region; Council Member and Chairman of Standards Committee of ITS-HK; and Chair of Computer Division, International Institute for Critical Infrastructures. He is a Chartered Electrical Engineer. He is a member of the Hong Kong Institution of Engineers and the Institution of Electrical Engineers. His biography has been published in *Who's Who in the World* (Marquis, USA) since 1998.