Decision making for Autonomous cars with random traffic flow

1st Charan Sai Guntupalli
Computer Science Department
Central Michigan University
Mount Pleasant, Michigan
guntu1c@cmich.edu

2nd Yasaswi Sai Niharika Avula
Computer Science Department
Central Michigan University
Mount Pleasant, Michigan
avula1y@cmic.edu

3rd Raasi Naidu Talluri
Computer Science Department
Central Michigan University
Mount Pleasant, Michigan
tallu1r@cmich.edu

***Abstract*—Decision-making strategy is compared to the human brain and is extremely important in autonomous vehicles. Such systems must be up to date, safe and efficient. In this research, A Markov's probabilistic decision-making strategy is implemented to automatically derive the ideal maneuver in a Random-Traffic scenario without utilizing any human data. The decision-making issues in a Random-Traffic environment are formulated as the MDP by defining basic elements that include states, actions, basic models, Transition and reward models. They are defined by using a complete prediction model of the surrounding cars, Pedestrians/Cyclists, objects and Animals. A dynamic programming method was used to generate an optimal policy, which was then tested in a two-dimensional simulation environment named DeepCars. Results constitute as, at a given Random-traffic scenario, the self-driving car maintained safety and efficiency with the proposed policy.This study outlines a framework for developing self-driving car driving policies to employ in Random-Traffic scenarios that do not rely on human driving data or hand-modeled regulations.**

I. Introduction

As science and development have evolved, autonomous driving technology has increasingly been the center of study because of its broad application potential in both military and civilian domains. Highly intelligent autonomous military vehicles can help soldiers with a number of activities such as information gathering, firefighting, and monitoring [1], while autonomous vehicles in the civilian sector offer a lot of promise for decreasing traffic accidents and alleviating traffic congestion [2].

A self-driving car is a sophisticated intelligent system that integrates environmental awareness, path planning, decision-making, and motion control technologies [3]. As the "brain" of autonomous vehicles, decision-making systems are critical for their safe and efficient operation, and how to design high-intelligence and reliable decision systems is increasingly becoming a focus of research in the field of autonomous driving.

The Decision-making process is expressed in terms of generating human-level safe and reasonable driving behaviors while taking into account surrounding environmental information, the motion of other traffic participants, and ego vehicle state estimation; the generated driving behaviors are then taken into account by the motion control system to achieve efficient autonomous vehicle operation [4]. There are two sorts of decision-making processes: traditional approaches and learning-based strategies [5].

Typically, autonomous cars operate in a complex, dynamic environment while collaborating with other road users. Learning-based methods are used to achieve better decision-making for autonomous vehicles [6]; in addition, with the emergency of new powerful computational technologies, learning-based approaches have gained huge popularity and development in the field of autonomous vehicles [7]. Classical methods are not always effective in such driving environments due to poor robustness, so learning-based methods are used to achieve better decision-making for autonomous vehicles.

Because of the extremely dynamic, stochastic, and unpredictable character of the traffic environment, this study focuses on decision-making, which is a difficult problem to solve. Since the 2007 Urban Challenge, the finite state machine (FSM) has been the most frequent technique for self-driving automobile decision-making. It creates policies based on rules by manually modeling finite situations in the traffic environment. These scenarios and rules make the decision system clear, but they must be designed by an experienced expert to get high performance. This method was employed in the behavioral layer of the winner of the 2008 Urban Challenge Boss [8]. It planned ahead of time to identify three high-level behaviors: lane driving, intersection handling, and objective choosing. Each of them is associated with a number of low-level behaviors. When the automobile operated in different conditions, the decision system determined these actions based on rules that were pre-programmed. To solve stuckness, the runner-up [9] employed FSM to transition between 13 driving circumstances and invoke exception behaviors. Furthermore, Odin [10] used a hierarchical FSM system to come up with acceptable actions in the present situation. The system can tell the difference between a junction, a parking lot, and a regular road. Similarly, Furda et al. [11] proposed a multiple-criteria decision-making strategy that divided the work into two stages: the first stage used Petri nets to find plausible moves, and the second stage employed a multivariate utility function to choose the most appropriate one. In some instances, the FSM is simple and effective. It cannot, however, be used in a dynamic traffic situation since it does not explicitly account for environmental risks. Furthermore, it necessitates manually

categorizing instances and modeling regulations, which makes it impossible to make decisions in uncommon circumstances.

This paper provides a probabilistic decision-making strategy that may be used in a dynamic traffic situation while also taking into account safety, efficiency, and comfort. By specifying the environment state space and agent action space, the driving problem was originally articulated as a Markov decision process (MDP). Then, using a prediction model of nearby automobiles, it created a state transition model and a reward model. Using the value iteration approach of dynamic programming, the best policy was then automatically inferred (DP). The simulation results suggest that the predetermined objective can be met.

Markov decision processes(MDPS) Model essence is that the state of the environment that affects the immediate reward obtained by the agent, as well as the probabilities of future state transitions [12]. The agent's objective is to select actions to maximize a long-term measure of total reward. model based on Dynamic Programming (DP) and Linear Programming(LP), which can generate efficient predictions of incoming vehicles, Animals(deers, Kangaroos, bears as such), Pedestrians/cyclists. We are going to employ Deep reinforcement Learning(RL) Technique with continuous actions to gather environmental information and switch from one state to another swiftly to bring the benefits of MDPS(maximize total reward). Decision making platform was built to validate the results based on defined policy and previous experience which is gained from training the model with all possible scenes and scenarios in both urban and suburban environments.

II. Background

A. History of Self Driving Cars:

Wings are to flight, as brains are to think. In a similar vein to the previous assertion, The self-driving vehicles primary goal is to have a human brain equivalent leading the vehicle.

The Union of Concerned Scientists states that self-driving cars are "cars or trucks in which human drivers are never required to take control to operate the vehicle safely. Also known as autonomous or 'driver-less' cars, they combine sensors and software to control, navigate, and drive the vehicle."

The origins of self-driving cars can be traced back to Leonardo da Vinci's invention of a cart that could move without being pushed or dragged in the 1500s. The cart was propelled by high-tension springs, and the steering was preprogrammed so it could follow a specified path. The world's first robot is also a term used to describe this gadget. In 1925, inventor Francis Houdina drove a radio-controlled automobile around Manhattan streets with no one at the wheel. General Motors then unveiled the first self-driving car model at the 1939 World's Fair. It was a self-driving electric car powered by magnetized metal spikes buried in the road and guided by radio-controlled electromagnetic fields. In the year 1958, this concept came to fruition. The Japanese expanded on this concept in 1977 with a camera system that sent data to a computer for image processing. The world's first self-driving passenger vehicle, capable of speeds of up to 20 miles per hour, was tested as a result of this. In the 1980s [13], Carnegie Mellon University's Navlab and ALV initiatives introduced the first self-sufficient and totally autonomous vehicles. The autonomous vehicle sector was well established by the early 2000s. DARPA, [14] the research branch of the US Department of Defense, organized a series of competitions to speed up the development of self-driving cars [15]. They sponsored a tournament in 2004 in which automobiles were challenged to self-navigate 150 miles of desert road. The route was not completed by any car. By the mid-2010s, [16] major automakers such as Ford, Mercedes-Benz, and BMW, as well as ride-hailing services such as Uber, have thrown their hats into the self-driving ring.

Autonomy in vehicles is often categorized into six levels, according to a system developed by SAE International, They are generally translated as:**??**

- Level 0 - no automation,
- Level 1 - hands-on/shared control,
- Level 2 - hands-off,
- Level 3 - eyes off,
- Level 4 - mind off, and
- Level 5 - steering wheel optional.

B. The technologies that self-driving cars employ these days:

1) Hybrid navigation-: There are different systems that help the self-driving car control the car, including the car navigation system, the location system, the electronic map, the map matching, the global path planning, the environment perception, the laser perception, the radar perception, the visual perception, the vehicle control, the perception of vehicle speed and direction, and the vehicle control method [17].

2) Homogenization-: In order for autonomous vehicles to comprehend their environment, they must employ a variety of methodologies, each with its own set of digital data (e.g. radar, GPS, motion sensors and computer vision) [18]

3) Vehicle communication systems-: Vehicle-to-vehicle communication systems use vehicles and roadside units as communicating nodes in a peer-to-peer network, exchanging data. [19]

4) Re-programmable-: Autonomous vehicles contain software systems that drive them, That makes the Decision for them, which means that updating the software by reprogramming or modifying it can improve the owner's benefits (e.g. update in better distinguishing Kangaroo as well as deer while driving) Smart autonomous vehicles can produce certain updates and install them based on machine learning (e.g. new navigation maps or new intersection computer systems). [3]

5) Modularity-: Autonomous vehicles are more modular because they are made up of multiple modules, which will be discussed using a Layered Modular Architecture in the following sections.

- The device layer is the first layer of this design, and it contains logical capability and physical machinery.
- The network layer sits on top of the device layer. Physical transport and logical transmission are two independent elements of this layer.

- The service layer contains the apps and features that assist the autonomous vehicle (and its owners) in extracting, creating, storing, and consuming content about their own driving history, traffic congestion, roadways, and parking capabilities.
- The contents layer is the model's final layer. The sounds, graphics, and movies are all contained on this layer.

C. The following are some potential technological roadblocks for self-driving cars:

- A car's computer, as well as a communication system between vehicles, could be compromised [20].
- Large-animal avoidance necessitates recognition and tracking, and Volvo discovered that software designed for caribou, deer [21], and elk was inadequate when it came to kangaroos
- Driving necessitates numerous complicated social interactions [22], which are still difficult for robots to master such as other drivers, cyclists, and animals.
- Weather conditions and road conditions in remote areas. Accident Liability, as well as the development of failsafe Radar Interference.
- To function successfully, autonomous vehicles may require high-definition maps. In the event that these maps become outdated, they must be able to revert to reasonable behavior.
- For automated cars to work optimally, current road infrastructure may need to be updated.
- The need for novel simulation-based approaches, such as digital twins and agent-based traffic simulation, to validate Automated Driving [23].

D. Algorithms to drive in autonomous cars:

Most of the current self-driving cars make use of multiple algorithms to drive. Furthermore, most approaches use supervised learning to train a model to drive the car autonomously. Autonomous driving is a multi-agent setting where the host vehicle must apply sophisticated negotiation skills with other road users when overtaking, giving way, merging, taking left and right turns, and pushing ahead in unstructured urban roadways. Many automobile manufacturers, such as Toyota, Tesla, Ford, Audi, Waymo, Mercedes-Benz, General Motors, and so on, are developing their own autonomous cars and achieving tremendous progress. Meanwhile, automotive researchers are paying attention to overcoming the essential technologies to build automated cars with full automation. Since there are many possible scenarios, manually tackling all possible cases will likely yield a too simplistic policy. Moreover, one must balance between unexpected behavior [24] of other drivers/pedestrians [25] and at the same time not be too defensive so that normal traffic flow [26] is maintained. Reinforcement learning is considered to be a strong AI paradigm that can be used to teach machines through interaction with the environment and learning from their mistakes. Despite its perceived utility, it has not yet been successfully applied in automotive applications. Reinforcement learning [27] is considered a promising direction for driving policy learning. Deep reinforcement learning [28] has received considerable attention after the outstanding performance of AlphaGo. However, training autonomous driving vehicles [29] with reinforcement learning in a real environment involves non-affordable trial-and-error. It is more desirable to the first train in a virtual environment and then transfers to the real environment.

Four significant modules [17] are contained in autonomous vehicles, which are perception, decision-making [30], planning, and control. Perception indicates that autonomous vehicles [31] know information about the driving environments based on the functions of a variety of sensors, such as radar, lidar, a global positioning system (GPS), etc. The Decision-making controller manages the driving behaviors of the vehicles, and these behaviors include acceleration, braking, lane-changing, lane-keep, and so on. The planning function helps the automated cars find reasonable running trajectories from one point to another. Finally, the control module would command the onboard powertrain components to operate accurately to finish the driving maneuvers and follow the planning path. According to the intelligent degrees of these mentioned modules, the AD is classified into six levels, from L0 to L5.

Decision-making strategy is regarded by the human brain and is extremely important in autonomous vehicles. This policy is often generated by the manual rules based on human driving experiences or imitated manipulation learned from supervised learning approaches.

Deep reinforcement learning (DRL) techniques are taken as a powerful tool to deal with long sequential decision-making problems. Many attempts have been implemented to study DRL-based autonomous driving topics in recent years. For example, Duan et al. built a hierarchical structure to learn the decision-making policy via the reinforcement learning (RL) method. The pro of this work is independent of the historical labeled driving data.

One of the main technologies for self-driving cars is decision-making. It's difficult to model policies for various driving scenarios such as in cities, towns and suburbs. A probabilistic decision-making system based on the Markov decision process (MDP) is suggested in this study to automatically derive the ideal maneuver in a random traffic scenario. The MDP addresses decision-making in a random traffic environment by specifying basic aspects such as states, actions, and basic models.

III. Methods

A. Important components used in Self driving cars are :

- Cameras : It's ideal for identifying things like highway lane lines, speed signs, and traffic signals. Some developers believe that with improved machine vision, they will be able to utilize cameras to recognize and navigate through whatever they observe.
- Lidars : It also employs a suite of sensors known as Light Detection and Ranging, or LIDAR, to increase navigation accuracy. A LIDAR operates by lighting a target with

pulsed laser light and detecting the reflected pulses with a sensor to determine distance. The target's digital 3-D representations are created using differences in laser return timings and wavelengths. A LIDAR has a precision of up to 2.5 cm. Multiple LIDAR modules positioned throughout the vehicle's body aid in the creation of an accurate picture of the surrounding area and the avoidance of blind spots. Collision avoidance is further aided by the use of LIDAR and RADAR.

- Machine Learning : ML is already being applied to numerous areas of the technology used in sophisticated driver-assistance systems, despite the fact that autonomous cars are still in the development and testing stages (ADAS). It also appears to be a factor in future developments.
- GPS : GPS may be used by self-driving automobiles to geolocate their physical positions in space, with numerical coordinates (e.g. latitude, longitude) indicating their physical locations. They may also use real-time GPS coordinates in conjunction with other digital map data to navigate (e.g. via Google Maps). The GPS data often fluctuates within a five-meter radius.
- Radars : The core set of sensors in most autonomous cars are cameras, LiDAR (light detection and ranging), and RADAR (radio detection and ranging), which offer imaging, detection, ranging, tracking, and sensing of the driving position for a flawless ride. RADAR relies on radio waves. The time taken by the radio waves to return from the obstacles to the device is used for calculating the distance, angle, and velocity of the obstacle in the surroundings of the autonomous vehicle.

B. Algorithms used in Self Driving Cars :

a Reinforcement Learning : Reinforcement learning is used to make intelligent sequential decisions and continuous decisions in order to reach the final objective. The reinforcement learning method has four components [32].

- Policy: Policy is also known as the decision function, pi:S→A is A mapping from state to action.
- Reward function:The reward function is the reward earned as a result of the Agent's interaction with the environment, and it is used to adjust the strategy [33]. The goal of reinforcement learning is for the Agent's accumulated reward value to be as high as possible.
- Value Function: In the long run, the value function considers a state or a state-action pair. The cumulative return from the side can be reflected by the value function, also known as the evaluation function.
- Environment: The environment is a representation of the situation of the outside world. The environment will provide the Agent the next state and a reward when the Agent performs an activity in a certain state [34]. In an unfamiliar environment, the agent interacts with the environment to obtain the current state, and then performs actions based on policies. At this point, the environment will revert to its previous and present states, as well as the reward value associated with the action done. Following that, the agent will assess the action to be conducted in light of the current reward value.
- The Deep Reinforcement Learning (DRL) technique combines deep learning with reinforcement learning by processing state information collected from the environment with deep learning neural networks and reinforcement learning. Learn how to make judgments while driving.

b Q Learning - :The value function is overestimated by the Q-Learning method [35]. When the strategy is bad, the value estimate is excessively high and diverges in the Actor-critic framework . As the network is trained and iterated, this overestimation spreads and has a detrimental influence on the approach. Two value networks are employed to tackle this problem in this paper's algorithm. The two value networks allow action selection and Q value update to be separated. The strategy and value update are inextricably linked in the original Q-Learning algorithm. When a strategy fails, the value estimate diverges owing to overestimation, causing the strategy evaluation and update to fail due to the value assessment's inaccuracy.

c Double Q- learning:The Double Q-learning [36] technique tackles the above-mentioned value function overestimation problem by using the smaller of two network estimations, which is skewed towards underestimation bias and difficult to spread via training.

d TD3:This employs a dual-delay depth deterministic strategy gradient update approach to decrease variance, which updates the strategy at a frequency lower than the update Q function. After numerous updates, the policy network remains the same until the value error is minimal enough.

C. Experimental Design

DP and RL are computational strategies for addressing decision-making problems in order to attain a desired objective while interacting with the outside environment. The model of the system's behavior is required for DP approaches, but RL is a model-free approach that improves the produced policy while interacting with the environment. To mathematically describe discrete stochastic environments, RL employs the Markov Decision Process (MDP) [37]. Because MDP operates in discrete time, states and actions in RL are often discrete as well, resulting in a sequential decision-making dilemma. The aim is to maximize the accumulated long-term return throughout the course of the interaction with the environment, and rewards also give a useful gauge of the agent's performance. Although recent improvements in RL enable us to apply to continuous issues, we use the discrete MDP to describe the autonomous driving decision-making problem and train RL algorithms to attain a positive performance [38].

To solve the difficulty of high-level decision making for an autonomous car, we combine the standard reinforcement learning strategy known as Q-learning with the Markov decision making technique in this work.

D. Pre-processing:

The primary concept behind reinforcement learning (RL) and dynamic programming (DP) is to use the observed states and rewards obtained from the environment to regulate an agent or a process while interacting with it . DP and RL are computational strategies for addressing decision-making problems in order to attain a desired objective while interacting with the outside environment. The model of the system's behavior is required for DP approaches, but RL is a model-free approach that improves the produced policy while interacting with the environment. To formally describe discrete stochastic environments, RL employs the Markov Decision Process (MDP) . Because MDP operates in discrete time, RL states and actions are often discrete, resulting in a sequential decision-making dilemma. The aim is to maximize the accumulated long-term return throughout the course of the interaction with the environment, and rewards give a useful indicator of the agent's success. Although recent improvements in RL [39] allow us to apply to continuous issues, we use the discrete MDP to describe the autonomous driving decision-making problem and train RL algorithms to produce good results.

A Hierarchical Architecture for Decision-Making in Autonomous Driving using Deep RL Q-learning analyzes how good taking an action could be at a given state (s, a). A memory table Q[s, a] is constructed in Q-learning to store the Q-values for all conceivable combinations of states and actions. The reward R and new states are determined by sampling an action from the current state, after which the next action a with the highest Q(s0, a0) from the memory table is selected. Taking an action in a specific state has a Q value, as shown in eq. 1.

Q(s, a) = R(s, a, s0) + max a0 Qk(s 0 , a0) $\rightarrow$ (1)

Next state and action are represented by s0 and a0, respectively. The memory and processing requirements for action-value function Q will be too high if the combinations of state and actions are too big or if states and actions are continuous. Deep Q-Network (DQN) [40], which approximates the action-value function Q, is used to overcome this issue (s, a).

Two networks, and, are built and trained in this study [41], one for extracting Q values and one for include all updates in the training. Finally, we synchronize and temporarily fix the Q value targets to prevent the target function from changing abruptly. Eq. 2 will be used to compute the loss.

Li(i) = E s,a,s0 ,rD r+max a0 Q(s 0 , a0 ; i Q(s, a; i))^2 $\rightarrow$ (2)

The transitions (s, a, s0, and r) are taken from the experience replay D. We employ the experience replay as a buffer, sampling a mini-batch of samples from it to train the deep neural network. The data is closer to i.i.d and more independent of each other since we are randomly sampling from the replay buffer, which makes the training more stable. Furthermore, E denotes the probability distribution's expectation. The input and output of the model become more stable to train when using the experience replay and target network, and the network acts more like a supervised learning algorithm.

E. Post-processing:

We also added a real-time validation step to standard DQN to capture the best learned model during training. We establish two periods for this purpose, during which the validation phase flag is set and late network weights are recorded and exploited during validation. The agent's performance is reviewed over a number of episodes depending on the time, and the attained mean reward is compared to the most recent maximum value. By verifying on unknown scenarios, the agent is able to record the best learned model. By defining two alternative periods with differing numbers of episodes, the training may be completed faster while also recording a more generalized model.

We also implemented the Double DQN (DDQN) method to increase DQN performance by dealing with overoptimistic value predictions by using two Qnetworks. In Double Q-learning, eq. 3 is used to estimate the target value.

Q(s, a) DoubleQ = R(s, a, s0)+Q(s 0 , argmaxQ(s 0 , a,); 0) $\rightarrow$ (3)

where, and 0 are two Qnetwork parameters, one for determining the greedy policy and the other for determining its value.

F. Results Analyzed

We intend to convert a three-dimensional world into a more simpler two-dimensional simulation environment and create a grid of occupancy for the surrounding actors around the ego vehicle. To make this feasible, we decided to create our own gaming environment using the pygame (a free and open source python programming language package).

IV. Results

We tried implementing open-source autonomous driving simulators like Airsim, Carla and Deepracer. But we couldn't achieve the best possible programmable results because of the trouble we faced in accessing and connecting to the Open-source APIs. So we implemented our model in DeepCars which is a self-driving car simulator environment accessed via pygame platform.

Initially, we are going to implement the Q-learning algorithm. We are planning to feed the occupancy grid of the environment into the agent which makes the use of classical RL impossible for this case.As a result, we will construct our MDP model using the DQN and DDQN methods. Unlike the FSM technique, which determines behavior based on rules, the MDP does not require much experience to create the entire decision system, but it does require some knowledge to design the per-step reward, which is linked to the driving goals. Here, The original DQN algorithm was updated and a real-time validation phase was introduced. This allows us to evaluate the most current model during training and select the best-trained candidates.

This can be accomplished by testing the most recent model on a large number of game episodes, say 100, and keeping track of the agent's performance. Finally, we keep track of the MDP model that performed the best at the end of training.

We used the value iteration method to find the best policy after developing the MDP model. A reward of -1000 is granted to the ego automobile when it transfers to a bad terminal state. When it enters a favorable terminal state a reward of 1000 is granted.

A state is markov if and only if P[S$_t$ + 1|S_t] = $P[S_t + 1|s_1, ..., S_t]$.

For a state St to be markov, The probability of the next state S(t+1) being s be solely determined by the current state St = st and not by the previous states S1=s1, S2=s2, and so on.

MDP state transition probability and reward

$P_s^a s' = P[S_t + 1 = s'|S_t = s, A_t = a]$,

$R_s^a = E[R_t + 1|S_t = s, A_t = a]$

(S, P, R, A, gamma), where S are the states, P is the state-transition probability, Rs is the reward, and gamma is the discount factor. A is the set of actions. It is essentially MRP with actions. Introduction to actions elicits a notion of control over the Markov Process, i.e., before, the state transition probability and the state rewards were more or less random. However, now the rewards and the next state also depend on what action the agent picks.

the state value is directly proportional to the vehicle speed. This is because when the ego car is beyond the other car, it can reach the good terminal state to get the 1000 reward(it can be seen that the MDP defines per step reward and uses the DP method to choose behavior in each state to get the maximal expected total reward in the future.). Because future rewards decay exponentially with respect to steps taken from the current state, the ego car arrives faster, the greater return it gets, and that is why states with a greater speed have a higher value. However, with the speed increasing, the agent is more concerned about preventing collision than being faster. As a result, a small acceleration or even a negative one is taken in this case.

For the control of the agent in DeepCars environment. The observed Markov state for the classical RL is defined as the following: S = ego lane ID x0 x1 ... xn in which the first element is the ego lane ID and xi indicates the distance to the closest car or obstacle in lane i.e Note that the lane number and grid numbers start from 0. For instance the state vector S = [2 6 3 8 0 8]. Also note that the line-of-sight of the vehicle is 9 (from 0 to 8). This means that the furthest car that the agent can percept can be 9 grids (number of rectangles) away. This constraint in fact simulates the sensor range that are implemented on the vehicle.

The main objective is to train the agent to avoid making collisions with other vehicles/random obstacles in the environment. Thus, we define a simple reward function (s, a, s0) = +1 s0 6= sT 1 s0 = sT Where sT indicates the terminal state that the agent makes a collision.

The ability to decompose the autonomous driving problem into a Markov's probabilistic Decision making architecture and leverage AI power to solve each layer separately strengthens the assertion that it is possible to meet the random traffic flow with higher reliability score.

V. Conclusion

In our paper, we looked at the challenges of autonomous driving in Random-Traffic scenarios. We studied the hierarchical architecture of ADAS systems, focusing on the decision-making layer, where a deep reinforcement learning algorithm is used to generate a series of high-level decision instructions. This system defines driving responsibilities as MDPs and includes a complicated motion prediction model for the surrounding vehicle, in which continuous space forecasts and discrete space MDP planning are combined using the probability technique. The optimum strategy is developed analytically and applied in a two-dimensional simulation environment in the specified Random-Traffic scenario. The findings show that it behaves rationally in order to achieve safe and efficient driving, such as braking when the front car slows down, overtaking the front car while traveling slowly, and staying in its lane rather than changing lanes when any obstacle approaches swiftly near the vehicle. The agent uses the occupancy grid in the environment as a state representation and creates lane change instructions to avoid colliding with other vehicles, Pedestrians or animals. In terms of traffic density and other actors' lane placements, the environment is probabilistic, which both evaluates the agent's performance and simulates the fusion layer output that may occur in real-world test situations. Finally, in order to use the decision-making model in practice, a more detailed vehicle dynamic model and a traffic model must be included; additionally, to improve policy generalization and reduce computation costs in large-scale decision problems, the approximation function should be used to represent the value function in the continuous state space.

References

[1] H.-y. Chen and Y. Zhang, "An overview of research on military unmanned ground vehicles," *Acta Armamentarii*, vol. 35, no. 10, pp. 1696–1706, 2014.

[2] Z. Li, B. Wang, J. Gong, T. Gao, C. Lu, and G. Wang, "Development and evaluation of two learning-based personalized driver models for pure pursuit path-tracking behaviors," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, pp. 79–84, IEEE, 2018.

[3] Q. Liu, X. Li, S. Yuan, and Z. Li, "Decision-making technology for autonomous vehicles: Learning-based methods, applications and future outlook," in *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pp. 30–37, IEEE, 2021.

[4] Z. Li, C. Lu, Y. Yi, and J. Gong, "A hierarchical framework for interactive behaviour prediction of heterogeneous traffic participants based on graph neural network," *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[5] Z. Li, J. Gong, C. Lu, and Y. Yi, "Interactive behavior prediction for heterogeneous traffic participants in the urban road: A graph-neural-network-based multitask learning framework," *IEEE/ASME Transactions on Mechatronics*, vol. 26, no. 3, pp. 1339–1349, 2021.

[6] F. Leon and M. Gavrilescu, "A review of tracking, prediction and decision making methods for autonomous driving," *arXiv preprint arXiv:1909.07707*, 2019.

[7] J. Fayyad, M. A. Jaradat, D. Gruyer, and H. Najjaran, "Deep learning sensor fusion for autonomous vehicle perception and localization: A review," *Sensors*, vol. 20, no. 15, p. 4220, 2020.

[8] C. Urmson, J. Anhalt, D. Bagnell, C. Baker, R. Bittner, M. Clark, J. Dolan, D. Duggins, T. Galatali, C. Geyer, *et al.*, "Autonomous driving in urban environments: Boss and the urban challenge," *Journal of field Robotics*, vol. 25, no. 8, pp. 425–466, 2008.

[9] M. Montemerlo, J. Becker, S. Bhat, H. Dahlkamp, D. Dolgov, S. Ettinger, D. Haehnel, T. Hilden, G. Hoffmann, B. Huhnke, *et al.*, "Junior: The stanford entry in the urban challenge," *Journal of field Robotics*, vol. 25, no. 9, pp. 569–597, 2008.

[10] A. Bacha, C. Bauman, R. Faruque, M. Fleming, C. Terwelp, C. Reinholtz, D. Hong, A. Wicks, T. Alberi, D. Anderson, *et al.*, "Odin: Team victortango's entry in the darpa urban challenge," *Journal of field Robotics*, vol. 25, no. 8, pp. 467–492, 2008.

[11] A. Furda and L. Vlacic, "Enabling safe autonomous driving in real-world city traffic using multiple criteria decision making," *IEEE Intelligent Transportation Systems Magazine*, vol. 3, no. 1, pp. 4–17, 2011.

[12] M. L. Littman, "Markov decision processes," 2001.

[13] T. Kanade, C. Thorpe, and W. Whittaker, "Autonomous land vehicle project at cmu," in *Proceedings of the 1986 ACM fourteenth annual conference on Computer science*, pp. 71–80, 1986.

[14] M. Buehler, K. Iagnemma, and S. Singh, *The DARPA urban challenge: autonomous vehicles in city traffic*, vol. 56. springer, 2009.

[15] I. Rudas, "J., haidegger, t., takacs, á., bosl d.,(2018)"highly automated vehicles and self-driving cars"," *IEEE ROBOTICS & AUTOMATION MAGAZINE*, vol. 25, no. 4, pp. 106–112.

[16] J. Deng, "W., dong, r. socher, l," in *J., Li,, K., Li,, and, L., Fei, Fei., Imagenet:, A large-scale, hierarchical, image, database., In, IEEE, Conference, on Computer, Vision, and Pattern, Recogni, tion,(CVPR),, pages*, pp. 248–255, 2009.

[17] J. Zhao, B. Liang, and Q. Chen, "The key technology toward the self-driving car," *International Journal of Intelligent Unmanned Systems*, 2018.

[18] C. Little, "The intelligent vehicle initiative: advancing" human-centered" smart vehicles," *Public Roads*, vol. 61, no. 2, pp. 18–25, 1997.

[19] D. Boehmlaender, S. Hasirlioglu, V. Yano, C. Lauerer, T. Brandmeier, and A. Zimmer, "Advantages in crash severity prediction using vehicle to vehicle communication," in *2015 IEEE International Conference on Dependable Systems and Networks Workshops*, pp. 112–117, IEEE, 2015.

[20] P. E. Ross, "A cloud-connected car is a hackable car, worries microsoft," *IEEE Spectrum*, vol. 11, 2014.

[21] N. Zhou, "Volvo admits its self-driving cars are confused by kangaroos," *The Guardian*, vol. 1, 2017.

[22] W. Song, G. Xiong, and H. Chen, "Intention-aware autonomous driving decision-making in an uncontrolled intersection.," *Mathematical Problems in Engineering*, 2016.

[23] S. Hallerbach, Y. Xia, U. Eberle, and F. Koester, "Simulation-based identification of critical scenarios for cooperative and automated vehicles," *SAE International Journal of Connected and Automated Vehicles*, vol. 1, no. 2018-01-1066, pp. 93–106, 2018.

[24] T. Liu, B. Tian, Y. Ai, L. Chen, F. Liu, D. Cao, N. Bian, and F.-Y. Wang, "Dynamic states prediction in autonomous vehicles: Comparison of three different methods," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pp. 3750–3755, IEEE, 2019.

[25] A. Rasouli and J. K. Tsotsos, "Autonomous vehicles that interact with pedestrians: A survey of theory and practice," *IEEE transactions on intelligent transportation systems*, vol. 21, no. 3, pp. 900–918, 2019.

[26] J. Liao, T. Liu, X. Tang, X. Mu, B. Huang, and D. Cao, "Decision-making strategy on highway for autonomous vehicles using deep reinforcement learning," *IEEE Access*, vol. 8, pp. 177804–177814, 2020.

[27] L. Busoniu, R. Babuska, B. De Schutter, and D. Ernst, *Reinforcement learning and dynamic programming using function approximators*. CRC press, 2017.

[28] Y. Bicer, M. Moghadam, C. Sahin, B. Eroglu, and N. K. Üre, "Vision-based uav guidance for autonomous landing with deep neural networks," in *AIAA Scitech 2019 Forum*, p. 0140, 2019.

[29] A. Broggi, M. Bertozzi, A. Fascioli, C. G. L. Bianco, and A. Piazzi, "The argo autonomous vehicle's vision and control systems," *International Journal of Intelligent Control and Systems*, vol. 3, no. 4, pp. 409–441, 1999.

[30] C.-J. Hoel, K. Driggs-Campbell, K. Wolff, L. Laine, and M. J. Kochenderfer, "Combining planning and deep reinforcement learning in tactical decision making for autonomous driving," *IEEE transactions on intelligent vehicles*, vol. 5, no. 2, pp. 294–305, 2019.

[31] C. Yang, Y. Shi, L. Li, and X. Wang, "Efficient mode transition control for parallel hybrid electric vehicle with adaptive dual-loop control framework," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 2, pp. 1519–1532, 2019.

[32] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International conference on machine learning*, pp. 1861–1870, PMLR, 2018.

[33] T. Haarnoja, H. Tang, P. Abbeel, and S. Levine, "Reinforcement learning with deep energy-based policies," in *International Conference on Machine Learning*, pp. 1352–1361, PMLR, 2017.

[34] B. D. Ziebart, A. L. Maas, J. A. Bagnell, A. K. Dey, *et al.*, "Maximum entropy inverse reinforcement learning.," in *Aaai*, vol. 8, pp. 1433–1438, Chicago, IL, USA, 2008.

[35] H. Hasselt, "Double q-learning," *Advances in neural information processing systems*, vol. 23, 2010.

[36] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 30, 2016.

[37] G. Li, S. E. Li, B. Cheng, and P. Green, "Estimation of driving style in naturalistic highway traffic using maneuver transition probabilities," *Transportation Research Part C: Emerging Technologies*, vol. 74, pp. 113–125, 2017.

[38] B. Paden, M. Čáp, S. Z. Yong, D. Yershov, and E. Frazzoli, "A survey of motion planning and control techniques for self-driving urban vehicles," *IEEE Transactions on intelligent vehicles*, vol. 1, no. 1, pp. 33–55, 2016.

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

[40] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[41] J. Chen, Z. Wang, and M. Tomizuka, "Deep hierarchical reinforcement learning for autonomous driving with distinct behaviors," in *2018 IEEE intelligent vehicles symposium (IV)*, pp. 1239–1244, IEEE, 2018.