[1] **J. Praveenchandar**
[2] **Saju Raj T**
[3] **Geetha Ponnaian**
[4] **T. Magesh**
[5] **S. Vinoth Kumar**

Autonomous Vehicle Traffic Accident Prevention using Mobile-Integrated Deep Reinforcement Learning Technique

Abstract: - When it concerns autonomous traffic management, the most effective decision-making reinforcement learning methods are often utilized for vehicle control. Surprisingly demanding circumstances, however, aggravate the collisions and, as a consequence, the chain collisions. In order to potentially offer guidance on eliminating and decreasing the danger of chain collision malfunctions, we first evaluate the main types of chain collisions and the chain events typically proceed. In an emergency, this study proposes mobile-integrated deep reinforcement learning (DRL) for autonomous vehicles to control collisions. Three essential influencing substances are completely taken into consideration and ultimately achieved by the offered strategy: accuracy, efficiency, and passenger comfort. Following this, we investigate the safety performance currently employed in security-driving solutions by interpreting the chain collision avoidance problem as a Markov Decision Process problem and offering a decision-making strategy based on mobile-integrated reinforcement learning. All of the analysis's findings have the objective of aid academics and policymakers to appreciate the positive aspects of a more reliable autonomous traffic infrastructure and to smooth out the way for the actual adoption of a driverless traffic scenario.

Keywords: Deep Reinforcement Learning; Autonomous vehicle safety; Markov decision process; Safety analysis

I. INTRODUCTION

Amongst the most important AI applications, autonomous driving offers enormous potential to enhance mobility, security, productivity, power use, convenience, and scientific organizations' interest. In recent times, both of those organizations have displayed an enormous amount of interest in this area of study. A car is required to set up a trustworthy, safe, and managed driving policy preceding it can provide entirely autonomous driving functions. It might not be viable to manually generate a controller that operates well in every scenario, even with substantial expert domain knowledge [1]. Due to the constantly shifting character of driving instances and the consequences of other drivers, safe decision-making in automated vehicles represents some of the most crucial unresolved problems Many researchers have felt encouraged to employ reinforcement learning (RL) in autonomous driving for planning and decision-making because of its recent successes. Our technique combines the beneficial aspects of both rule-based and learning-based risk approaches, in contrast to the previous studies. To learn how to forecast safety further into the future and figure out whether or not the future states will cause undesirable actions, the additional safety component—known as the dynamically-learned safety module—incorporates a hypothetical look ahead into the development stage of reinforcement instruction [2].

A technique for synchronization control has been provided in the present investigation to help autonomous vehicles in approaching crossings lacking traffic signals. The outcomes of the simulation demonstrate that the framework of control that was previously proposed might give both performance and safety. Self-driving car actions frequently make use of reinforcement learning (RL) and deep reinforcement learning (DRL). When attempting to generate a real-time decision-making controller, the amalgamation of transfer learning (TL) and reinforcement learning (RL)—also nicknamed DRL—remains an appealing research field.

[1] *Corresponding Author: Assistant Professor, Department of Computer Science and Engineering, Karunya Institute of Technology and Sciences, Coimbatore -641114. Email: praveenjpc@gmail.com

[2] Assistant Professor (SG), Department of Computer Science and Engineering, Vel Tech Rangarajan Dr. Sagunthala R&D Institute of Science and Technology, Avadi, Chennai, India. Email: drsajurajt@veltech.edu.in

[3] Assistant Professor (SG), Department of Computer Science and Engineering, Saveetha School of Engineering, Saveetha Institute of Medical and Technical Sciences, (SIMATS), Thandalam, Chennai, India. Email: geetha.saju@gmail.com

[4] Department of EEE, R.M.K Engineering College, Kavaraipettai, Tamil Nadu, India. Email: tmh.eee@rmkec.ac.in

[5] Associate Professor, Department of Computer Science and Engineering, School of Computing, Vel Tech Rangarajan Dr. Sagunthala R&D Institute of Science and Technology, Avadi-600062. Email: profsvinoth@gmail.com

The main contributions of this paper: are 1) presenting a survey of the recent advances of mobile-integrated deep reinforcement learning and 2) introducing a framework for end-to-end autonomous driving using mobile-integrated deep reinforcement learning to the automotive community [3]. In recognition of their outstanding performance, reinforcement learning (RL) approaches have been applied to highway traffic control tasks in recent years and have attracted an abundance of attention. Through engagement with the traffic environment, the agent in an RL-based VSL understands approximately the dynamics of traffic and, with enough training derived from state-action-reward outcomes, can formulate an optimal control policy. Although the state-action-reward arrangements in the present situation differ from those in the original scenario, implementing the ideal policy from the original scenario directly to the new scenario can end up in lower control performance [4].

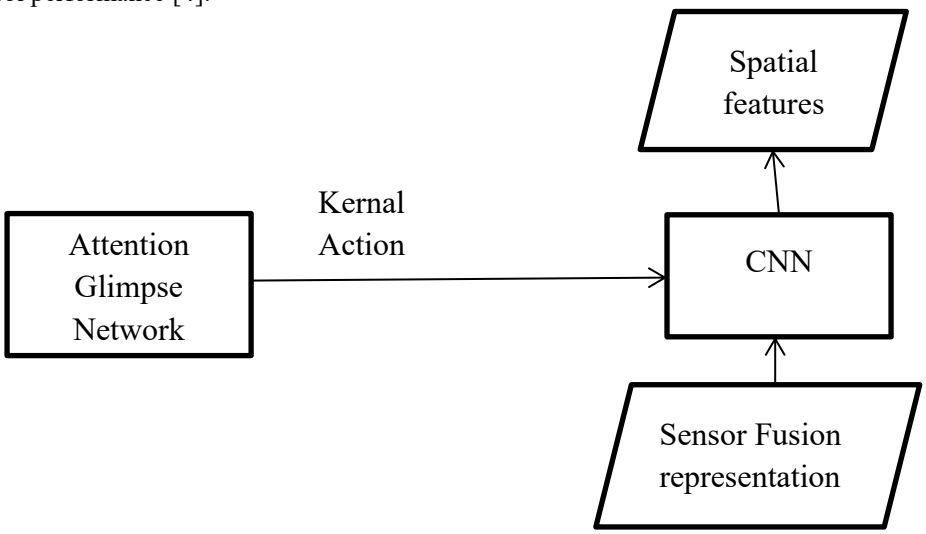


Figure. 1.1. DRL framework block diagram

A prospective way of addressing the issues that arise regarding traffic management system cyber security and road safety is reinforcement learning. Reinforcement learning provides immense potential in terms of developing intelligent autonomous driving systems that may assess traffic patterns, recognize dangerous situations, and engage in proactive steps to prevent incidents. The DRL framework's tidy block diagram for the self-driving system is displayed in Figure 1.1. Furthermore, an evaluation released during comprehensive road traffic management substances prompted by AI and IoT, demonstrates the potential of these breakthroughs to improve traffic management [5].

II. RELATED WORKS

[6] Despite RL has the capability of making intelligent choices, its visual potential is inadequate, hence inhibits it from completely overcoming the perception problem in actual-life situations. For the purpose of provide the agent an increased learning ability so that it can have the potential to address the perception problem of complex systems with greater efficacy, deep reinforcement learning (DRL) merges the ability to reflect of deep learning together the capacity for choice of reinforcement learning. Two neural networks have been generated by the DDPG algorithm, one for the policy network and the alternate one for the value network.

[7] Modern VRP heuristics are unable to rival our RL procedure, therefore representing taking another step forward towards employing RL to alleviate the VRP in real-world scenarios. Our strategy is relatively easy to put into motion in the truth during the time we can fix instances of similar measurements without requiring retraining for each new instance. In this case, a vehicle with a processor can solve its own VRP by employing the trained model by adhering to a set of specified arithmetic operations.

[8] In this research, the PCC challenge for a CAV platoon was originally discovered. For the objective of planning the reference frequencies of all vehicles in the platoon, theoretical methods are recommended. To develop a dispersed optimal state-feedback controller, the RL methodology is applied. Based on the simulation results, each vehicle's advancement, velocity, and acceleration can be managed by the obtained controller to shorten journey times while still preserving the optimal routes for each vehicle.

[9] We propose an RL-based adaptive vehicle trajectory control algorithm for different risk levels to solve this problem, as the existing methods are not optimized and tested for different risk levels. This algorithm can judge the change in risk levels in real-time and switch to the corresponding vehicle control model during the actual driving process. Furthermore, the equipment surpasses a single model when managing variable risk levels when the automobile is confronted by complex and adjustable traffic circumstances.

[10] For connected autonomous vehicles, this article recommended employing deep reinforcement learning (DRL) for enhancing the adaptive cruise control system. By integrating different techniques, the recommended approach strives for enhanced comfort, safety (which involves vehicle stability), and traffic efficiency and meticulously harmonizes jerk, headway, and longitudinal descent. The recommended approach yields considerably better overall performance when contrasted with cooperative ACC schemes and characteristic ACC schemes. Importantly, the DRL methodology produces upgraded traffic flow efficiency in both highway and urban scenarios by surpassing conventional CACC and ACC procedures in headway performance by 36% worldwide and by 47% during the fastest vehicle's speed variation phases.

[11] Multi-agent DRL in Traffic Safety Challenges is the main subject of this article. The basic principles of reinforcement learning and deep learning were initially covered. Immediately following that, it proceeded by presenting a handful of the significant phrases and algorithms that are used in these domains. It also demonstrated DRL and how it resolved the dimensionality challenge. We also met regarding many different cooperative driving strategies and how they change road conditions.

[12] The present study introduces a deep reinforcement learning-based routing protocol to reduce the option of link delays as well as boost the energy consumption and transmission efficiency of vehicle ad hoc networks. This technique is capable of reducing the potential of communication breakdown between automobile nodes by introducing a time projection approach to vehicular informal networks. To choose multihop routing, the algorithm also employs deep reinforcement modeling tactics, which might boost transmission efficiency and cut down transmission loss in automobile network routing.

III. METHODS AND MATERIALS

The space of possible responses could either be separate or ongoing depending on the difficulty location; this differentiation has an important influence on the strategies that deserve to be implemented. We will be speaking about a pair of algorithms in this section: a strategy that operates on discrete actions (DQN) and a different algorithm that succeeds with continuous actions (DDAC).

3.1 Deep Q Networks (DQN)

It is easier to formulate the Q-function as a table whenever the states are discrete. As the number of states comes up, this formulation grows more challenging, and when the states are continuous, it becomes impossible. In this instance, the parameterized function of the state of affairs and behavior, or Q(s,a,w), is utilized to formulate the Q-function. The next stage is figuring out which setting for the parameter w is ideal. Considering this formulation, a Deep Neural Network (DNN) can be effectively applied to approximate the Q-function. This DNN's purpose is to minimize the Qvalues' Mean Square Error (MSE) in the following ways:

$$m(x) = F\left[\left(s + \delta\, arg\max_{b'} R_u(t', b', x) - R_u(t, b, x)\right)^2\right] \quad (1)$$

$$K(x) = \max_x m(x) \quad (2)$$

Concerning its parameters, the aforementioned product is differentiable from starting point to the end, implying that exists. Gradient-based approaches, such as conjugate gradients (CG), stochastic gradient descent (SGD), etc., make it remarkably straightforward to solve the optimization problem. Deep Q-Networks is the short form of the algorithm (DQN).

3.2 Deep Deterministic Actor Critic (DDAC)

The Q-function and the critic function are the same. The policy gradient approaches are used by the algorithms to train both functions. Both functions can be taught using two neural networks within the deep learning framework; $R(t, b, x)$ and $\rho(t, v)$, as the overall objective maintains a differentiable when compared to the weights of the policy and the Q-function. Consequently, as in DQN, the gradient of the Q-function (the critic) is obtained: $\frac{\delta m(x)}{\delta x}$, using the chain rule gradient of policy function is received as

$$\frac{\delta K}{\delta v} = \frac{\delta R}{\delta b}\Big|_{b=\rho(t,v)} \frac{\delta \rho(t,v)}{\delta v} \quad (3)$$

3.3 Deep Attention Reinforcement Learning

A CNN whose services learn features from data is implemented in the DQN model to acquire special properties. There's a possibility that those features do not contribute properly to the overall improvement aim. Not all of the high-dimensional sensory information must be implemented to finish the identification tasks, much like the human recognition process requires only a limited amount of information. The concept that just a little of the CNN-extracted information is put to use in the classification process constitutes an example that attention models make efforts to replicate. The first learning phase and this particular one finished concurrently [13].

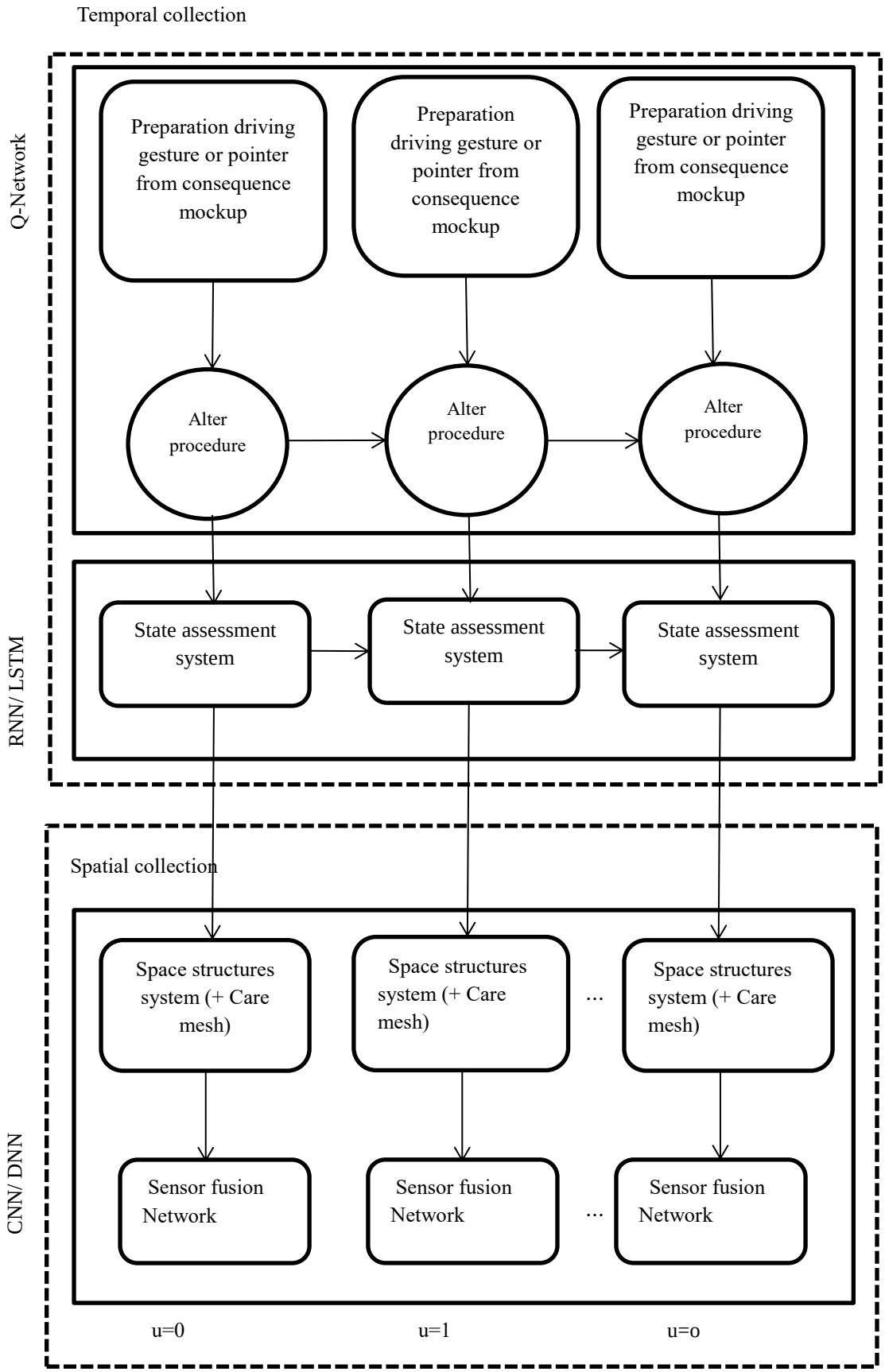


Figure. 3.1. Architecture of Deep Reinforcement Learning

3.4 Deep Reinforcement Learning

The quickest way to maintain obtained estimates (of values, guidelines, or models, for illustration) is to utilize tabular representations, whereby each state action tandem has a discrete estimate attributed to it. The total amount of state-action pair values that are required to be collected expands enormously with each incremental feature tracked in the state when estimates are represented discretely. In the literature, this particular problem frequently gets referred to as the "curse of dimensionality," an appellation that Bellman initially employed. In primitive contexts, this is rarely a problem, but in everyday applications, memory and/or evaluating restraints might render it a non-solvable issue. This is possible to learn over an extended state-action space, but it might require too long to gather useful policies. Persistent mode and/or action areas operate in many real-world domains; in numerous scenarios, these can be transformed. The DRL architecture is illustrated in Figure 3.1, thereby offering an explanation.

The $R(t, b)$ function can be viewed virtually as a deep neural network that, given the input state, predicts the value of each response. Because of this, understanding what to do requires conducting one forward pass through the network. On top of that, for superior testing efficiency, the agent's experiences are preserved in a repeat memory (experience replay), and then sample selections are derived to perform Qlearning updates. The correlation between the remaining samples is severed by this random selection. Reinforcement learning agents might remember and implement past experiences through experience replay, which includes observed transitions that are properly sampled from memory to update the network shortly after having been retained for a while, constantly in a queue. In both traditional Q-learning and DQN, the max operator chooses and examines an action that returns excessively pessimistic value predictions employing the same values. It has been shown that executing this technique results in significantly greater scores on several types of games in conjunction with crafting more accurate value estimates. As an outcome, the DRQN can identify information including object velocity by integrating data from various video frames. When trained on Atari games and examined on rushing games, it came to light that DRQN generalizes its policies with greater efficiency than DQN. Furthermore, DRQN revealed its aptitude for extrapolating its policies in situations of full observations [14].

IV. IMPLEMENTATION AND RESULTS

We will go over the practical use of our DRL-based decision-making algorithm, Algorithm 1, for autonomous highway driving in this section. We will begin with the vehicle dynamical concept that had been used for training first before we go into more details concerning the training environment and examine the policy that was offered.

4.1 Dynamics of the vehicle

A point-mass framework that is computationally advantageous is implemented to depict each vehicle. We propose a discrete-time double integrator for lengthwise coefficients of motion.

$$y(u+1) = y(u) + w_y(u)\Delta u \quad (4)$$

$$w_y(u+1) = w_y(u) + b_y(u)\Delta u \quad (5)$$

where $y \in \mathbb{S}$ is the longitudinal position, $w_y \in \mathbb{S}$ is a vehicle's longitudinal velocity, and u is the time index. Furthermore, Δu reflects the sampling duration. We assume the straightforward kinematic framework postulates for the transverse motion.

$$z(u+1) = z(u) + w_z(u)\Delta u \quad (6)$$

where the car's lateral spot is denoted by $z \in \mathbb{S}$. The outside power inputs $b_y(u)$ and $w_z(u)$ in above equations denote the vehicle's longitudinal boost and lateral velocity, appropriately. We believe that $b_y(u)$ can be split into four values, $b_y = \{b_1, 0, -b_1, -b_2\}$, with $b_1 = 2m/s^2$and $b_2 = 4m/s^2$, and we presume it diverges from nominal propulsion to a powerful brake. Hard braking of the $b_y = -b_2$ is utilized primarily in an urgent matter. The lateral velocity $w_z(u)$ supplies a reference lane for the vehicle, we feel like a lane transformation action needed 5 seconds to finish, with a possibility of halting the lane change movement whilst on each sampling occasion. In this project, recording at 1 Hz is utilised.

4.2 Simulation Atmosphere

It is a three-lane circular loop that was envisioned to appear as a never-ending length of undivided street. Almost $\{1, \ldots, O_U\}$ numbers of motor cars are selected at random within 250 yards of the ego car at the commencement of an episode. In this instance, we chose ϑ-greedy policy, whilst the traffic vehicles use an

IDM controller in conjunction with an arrangement of actuators. Additionally, traffic vehicles pose the flexibility to change channels at random. The system parameters, notably maximum velocity, are decided randomly for the traffic vehicles.

4.3 Decision Making for Ego Vehicle

We deploy a reward function ρ that entails of already established driving aims for the ego vehicle for the purpose to educate the policy π. It is created in anticipation of

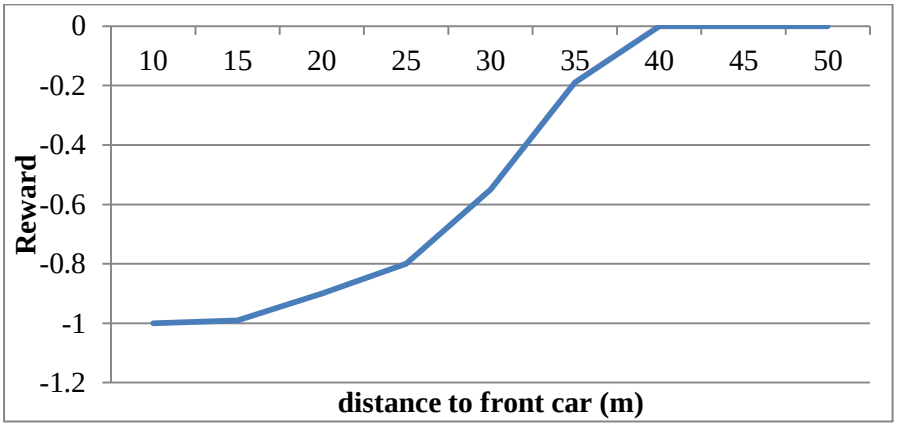
- The preferred speed while considering the movement of vehicles
- With regard to traffic circumstances, the expected lane and lane imbalance.
- Proportional velocity-based gap to the automobile in front of you

$$s_w = e^{-\frac{\left(w_{e_y}-w_{des}\right)^2}{10}} - 1 \quad (7)$$

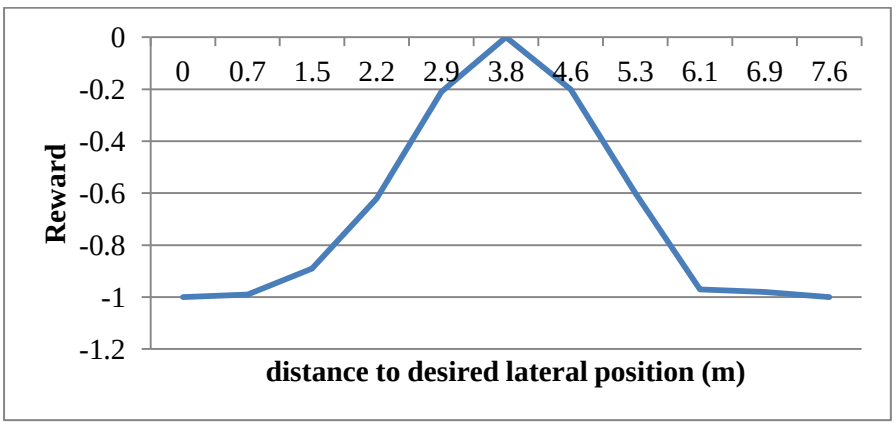
$$s_z = e^{-\frac{\left(f_{e_z}-z_{des}\right)^2}{10}} - 1 \quad (8)$$

$$r_y = \begin{cases} e^{-\frac{\left(f_{lead}-f_{safe}\right)^2}{10 f_{safe}}} - 1 & if\ e_y < f_{safe} \\ 0 & otherwise \end{cases} \quad (9)$$

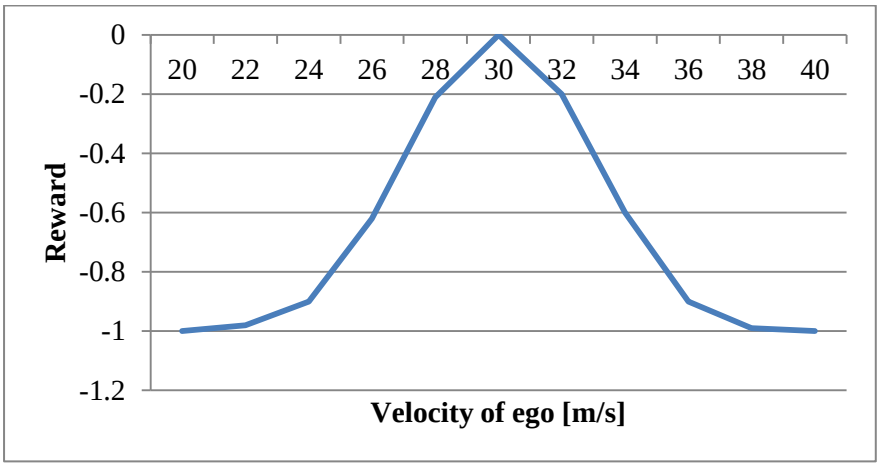
where ego velocity, lateral position, and continuous distance to the lead vehicle are portrayed by the factors w_{e_y}, f_{e_z}, and f_{lead}, respectively. In a similar vein, w_{des}, z_{des}, and f_{safe} represent for the anticipated speed, lane position, as well as the secure longitudinal distance, correspondingly, relative to the originating vehicle. Figure 4.1 conveys a representative depiction of the compensation functions (7)-(9), it is calculated presuming $w_{des} = 30\ m/s$ which can be obtained in the main lane i.e., $z_{des} = 3.8m$ assuming a minimum of safe distance $f_{safe} = 40m$. The expected values might change based on the specifics and the present situation of the traffic. The height of the peak in Figure 4.1 c) will be updated based on fast and slow moving traffic depending on the traffic situation. In this work we punished the ego transport if it doesnt preserve a brief period headway of at a minimum 1.3 seconds.



a) Reward for desired relative distance



b) Reward for desired lateral position



c) Reward for desired ego speed

Figure. 4.1. Sub-goals are equally weighted in the ultimate reward calculation, with the ego car's payout determined by traffic conditions

Every one of 100 episodes during learning, we analyze the (partially) programmed DRL actuator. The cumulative payment per decision completed during the training time frame is shown in Figure 4.2. The agent requires over two hundred episodes to converge. The DRL agent has been instructed over 10,000 occasions in total. Each episode comprising 200 samples or until a collision, whichever arrives sooner. For the first 7000 episodes, exploration is always decreased from 1 to 0.2, and it remains at the same value for the remaining parts of the educational procedure. A deep neural network with two submerged layers and 100 totally linked leaky ReLUs in each, the Q-network. Using the Adam optimizer, we train the network at a preset rate of learning of 1e − 4. It was found that the safety controller was significant in grasping the pertinent policy for the dangerous driving task. The mean and confidence bound for training over 200 training iterations of Algorithm 1 combined with and without safety controllers are presented in Figure 4.2. Without an evident safety check, training a typical DDQN agent failed to teach them an adequate policy and consistently lead to a tragedy. On the other hand, DDQN was capable of to get closer to an excellent policy when it passed an explicit safety check. On the basis of equations (7) through (9), the maximum possible reward for an agent is zero for any given decision; nonetheless, our educated DDQN agent with prevention check normally gets compensation per decision that are typically −0.025.

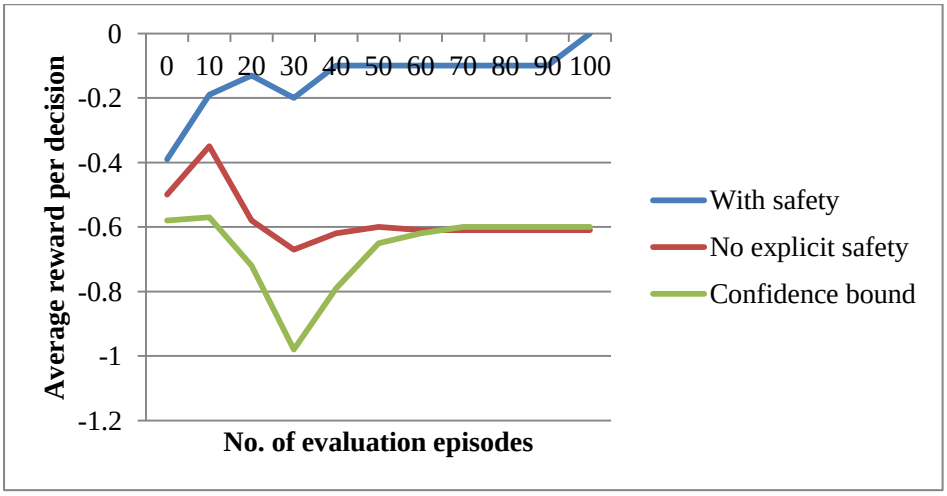


Figure. 4.2. Algorithm 1's average learning curve with reliability bound for both the limited horizon safety assessment and its absence of it.

Table 1. Illustration of Average Reward per Decision According to the Number of Evaluation Episodes

Number of episodes	With safety	No explicit safety	Confidence bound
0	-0.39	-0.5	-0.58
10	-0.19	-0.35	-0.57

20	-0.13	-0.58	-0.72
30	-0.2	-0.67	-0.98
40	-0.1	-0.62	-0.79
50	-0.1	-0.6	-0.65
60	-0.1	-0.61	-0.62
70	-0.1	-0.61	-0.6
80	-0.1	-0.61	-0.6
90	-0.1	-0.61	-0.6
100	0	-0.61	-0.6

We evaluate our programmed DDQN agent in Figure 4.3 to determine average velocity as traffic density increases. When the measured time to collision (TC) rises above the aggregate acceleration (TA), the revised safety controller from (4) is juxtaposed with this. In Figure 4.3, this is referred to as IDM. It must have pointed out that the IDM controller from (4) can't be used to begin lane changes. To solve this, we combine IDM with the decision-making process for lane changes. Figure 4.3, clearly demonstrates the benefits of RL for advanced-level decision processing when compared to model-based techniques. With the increase in traffic density, both the trained DDQN agent and the model-based lane change controller meet to IDM controller. Table 1 describes the illustration of average reward per decision according to the number of evaluation episodes**.** This will occur as switching lanes in a location with greater traffic is neither hazardous nor advantageous. The prioritized experience reply (PER) is simplified in Algorithm 1 by introducing two explicit buffers, Buf_T and Buf_D, to capture safe and non-safe transitions. When compared with the highway driving example, implementing two explicit buffers delivers an arguably more effective technique over PER.

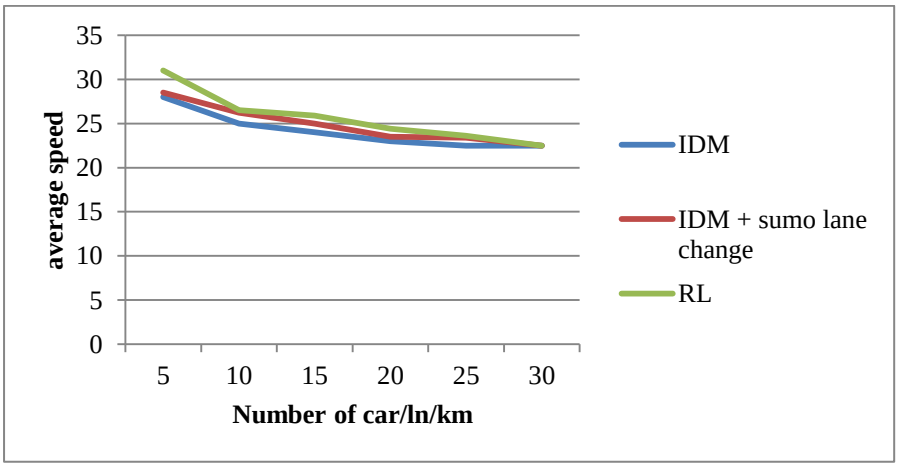


Figure. 4.3. Average speed for a trained RL agent and an initial IDM Controller with Lane Shifting

Algorithm 1: A secure decision maker for robotic highway traveling determined by DRL

Step 1: Initialize: $Buf_T, Buf_D, R(\vartheta)$ and network target $\hat{R}(\hat{\vartheta})$ with $\hat{\vartheta} = \vartheta$

Step 2: for episode =1, O_e, do

Step 3: Initialize: $\{1, \cdots, O_U\}$ cars randomly, get affordance indicator t_0

Step 4: for samples u=1, O_U, or collision, do

Step 5: Using ∋ pick the random action b_u, else $b_u = arg\ \max_b R((t_u, b, \vartheta_u))$

Step 6: For ego car: If b_u is not safe. Then store the value in $(T_u, b_u, *, s_{col})$ in Buf_D and replace b_u by safe action c_t

Step 7: Apply action, observe t_{u+1} and obtain $s_{u+1} = \sigma(T_u, T_{u+1}, b_u)$

Step 8: if collision then

Step 9: Store transition $(T_u, b_u, *, s_{col})$ in collision buffer Buf_D

Step 10: else
Step 11: Store transition $(T_u, b_u, T_{u+1}, s_{u+1})$ in collision buffer Buf_T
Step 12: end if
Step 13 Sample random minibatch $(T_k, b_k, T_{k+1}, s_{k+1})$ from Buf_T and Buf_D
Step 14 Set

$$z_k = \begin{cases} s_{k+1}\ if\ sample\ is\ from\ Buf_D \\ s_{k+1} + \delta\hat{R}\left(t_{k+1}, arg\max_b R(t_{k+1}, b, \vartheta_u), \hat{\vartheta}_u\right)\ if\ sample\ is\ from\ Buf_T \end{cases}$$

Step 15: Achieve gradient descent on $\left(y_k - R(T_k, b_k, \vartheta_u)\right)^2$ with respect to ϑ.
Step 16: Every O_D episodes set $\hat{R} = R$
Step 17: end for
Step 18: end for

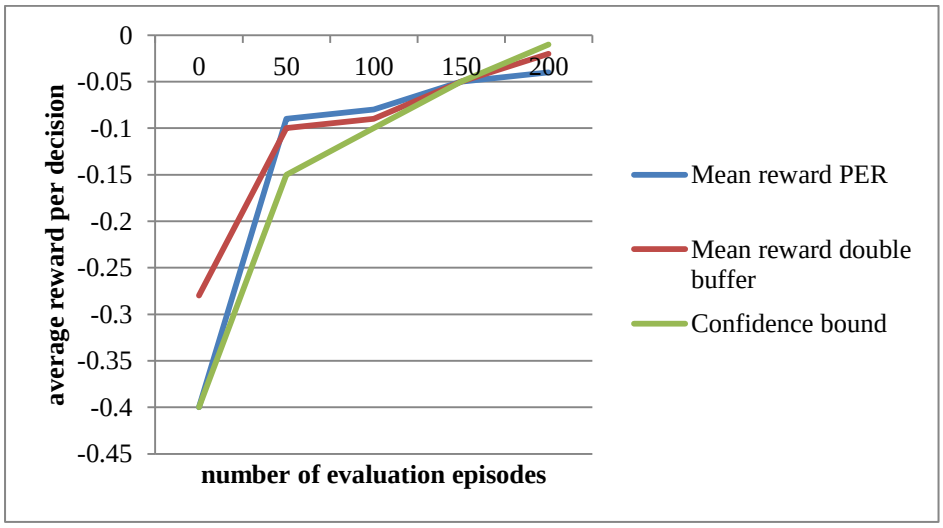


Figure. 4.4. Algorithm 1's Mean Learning Curve with Optimism Bound and Prioritized Experience Response

4.4 Continuous Adaptation

In the implementation stage, we exchange the picked up policy π for the ϑ-greedy policy, or $\rho_\varepsilon\}$ in Algorithm 1 line 5. Buffer Buf_D is updated with fresh information everytime the control decision provided by DDQN misses the shorthorizon stability check. Q-network could be modified by employing a learning rate slightly lower than the one which was used right through training (lines 13 through 16). The ongoing adjustment result around 30K episodes can be observed, all of which was created by applying a moving average filter to average the data set over 10K episodes. Figure 4.4 illustrates the algorithm's mean learning curve for each of the double buffer and the PER. Filtering drives the mean numbers of safety alarms to increase over the period of the first 10,000 episodes and still are constant in the lack of adaptation, instead continuous adaptation drives it to monotonically collapse to a smaller value. The average amount of protection prompt never converges to zero, even with continuous adaptation. This might happen related to the following factors: 1) Rigid and static safety legislation; 2) Utilization of value estimate, where an educated NN can choose to take an action that does not seem risky. In our next project, we hope to deal with the aforementioned issue [15].

V. CONCLUSION

We repeat in our decision that a mobile-integrated deep reinforcement learning-based managing technique for eliminating multiple automobile accidents has been presented in this study. We established extensive taxonomies based on the strategies and solutions previously in effect. We supply an AI-enabled conceptual framework to assist us in elucidating our intended topic. The most serious possibilities of link or several vehicle breaks, which include violent shifts in lanes and recessions are all taken aspirations in this review. We deployed RL-based decision-making approaches to reduce chain events. We offered the challenge of using sensors for collecting vehicle state information on both the participant and the ego. We incorporated the agent vehicle with the PPO, SAC, and DDPG algorithms and constructed a Markov decision process for describing

the collision avoidance approach and reward function based on those variables. Real-world testing or multiple collision-avoidance simulations will be required to assuring the safety of autonomous vehicles. It passes into tremendous length on how to evaluate coaching results for safety efficiency through incorporating uncertainty analysis throughout both single and multi-agent operation settings. Three approaches are utilized in the uncertainty forecasting process to convey various signs on emulated test data in the appropriate way. Simulation results from single- and multi-vehicle training suggest the technique has the advantage of correctly judging a chain collision-avoidance system's driving efficiency.

REFERENCES

[1] Yu, C., Wang, X., Xu, X., Zhang, M., Ge, H., Ren, J., ... & Tan, G. (2019). Distributed multiagent coordinated learning for autonomous driving in highways based on dynamic coordination graphs. *Ieee transactions on intelligent transportation systems*, *21*(2), 735-748.

[2] Baheri, A., Nageshrao, S., Tseng, H. E., Kolmanovsky, I., Girard, A., & Filev, D. (2020, October). Deep reinforcement learning with enhanced safety for autonomous highway driving. In *2020 IEEE Intelligent Vehicles Symposium (IV)* (pp. 1550-1555). IEEE.

[3] Shu, H., Liu, T., Mu, X., & Cao, D. (2021). Driving tasks transfer using deep reinforcement learning for decision-making of autonomous vehicles in unsignalized intersection. *IEEE Transactions on Vehicular Technology*, *71*(1), 41-52.

[4] Ke, Z., Li, Z., Cao, Z., & Liu, P. (2020). Enhancing transferability of deep reinforcement learning-based variable speed limit control using transfer learning. *IEEE Transactions on Intelligent Transportation Systems*, *22*(7), 4684-4695.

[5] SATAPATHY, K., CHO, S. B., PRUSTY, M. R., & MOHANTY, S. N. (2024). Enhancing Road Safety and Cybersecurity in Traffic Management Systems: Leveraging the Potential of Reinforcement Learning.

[6] Yu, C., Ni, A., Luo, J., Wang, J., Zhang, C., Chen, Q., & Tu, Y. (2022). A novel dynamic lane-changing trajectory planning model for automated vehicles based on reinforcement learning. *Journal of advanced transportation*, *2022*.

[7] Nazari, M., Oroojlooy, A., Snyder, L., & Takác, M. (2018). Reinforcement learning for solving the vehicle routing problem. *Advances in neural information processing systems*, *31*.

[8] Gao, W., Odekunle, A., Chen, Y., & Jiang, Z. P. (2019). Predictive cruise control of connected and autonomous vehicles via reinforcement learning. *IET Control Theory & Applications*, *13*(17), 2849-2855.

[9] He, Y., Liu, Y., Yang, L., & Qu, X. (2023). Deep adaptive control: Deep reinforcement learning-based adaptive vehicle trajectory control algorithms for different risk levels. *IEEE Transactions on Intelligent Vehicles*.

[10] Selvaraj, D. C., Hegde, S., Amati, N., Deflorio, F., & Chiasserini, C. F. (2023). A Deep Reinforcement Learning Approach for Efficient, Safe and Comfortable Driving. *Applied Sciences*, *13*(9), 5272.

[11] Neill, D. J. (2021). Using Deep Reinforcement Learning to increase Traffic Safety in Urban areas whilst maintaining Traffic Flow and Efficiency.

[12] Ye, S., Xu, L., & Li, X. (2021). Vehicle-mounted self-organizing network routing algorithm based on deep reinforcement learning. *Wireless Communications and Mobile Computing*, *2021*, 1-9.

[13] Sallab, A. E., Abdou, M., Perot, E., & Yogamani, S. (2017). Deep reinforcement learning framework for autonomous driving. *arXiv preprint arXiv:1704.02532*.

[14] Kiran, B. R., Sobh, I., Talpaert, V., Mannion, P., Al Sallab, A. A., Yogamani, S., & Pérez, P. (2021). Deep reinforcement learning for autonomous driving: A survey. *IEEE Transactions on Intelligent Transportation Systems*, *23*(6), 4909-4926.

[15] Nageshrao, S., Tseng, H. E., & Filev, D. (2019, October). Autonomous highway driving using deep reinforcement learning. In *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)* (pp. 2326-2331). IEEE.