


## Article

# Deep-Reinforcement-Learning-Based Collision Avoidance of Autonomous Driving System for Vulnerable Road User Safety

Haochong Chen, Xincheng Cao, Levent Guvenc  and Bilin Aksun-Guvenc \*

Automated Driving Laboratory, Ohio State University, Columbus, OH 43212, USA; chen.9286@osu.edu (H.C.); cao.1375@osu.edu (X.C.); guvenc.1@osu.edu (L.G.)

\* Correspondence: aksunguvenc.1@osu.edu

**Abstract:** The application of autonomous driving system (ADS) technology can significantly reduce potential accidents involving vulnerable road users (VRUs) due to driver error. This paper proposes a novel hierarchical deep reinforcement learning (DRL) framework for high-performance collision avoidance, which enables the automated driving agent to perform collision avoidance maneuvers while maintaining appropriate speeds and acceptable social distancing. The novelty of the DRL method proposed here is its ability to accommodate dynamic obstacle avoidance, which is necessary as pedestrians are moving dynamically in their interactions with nearby ADSs. This is an improvement over existing DRL frameworks that have only been developed and demonstrated for stationary obstacle avoidance problems. The hybrid A\* path searching algorithm is first applied to calculate a pre-defined path marked by waypoints, and a low-level path-following controller is used under cases where no VRUs are detected. Upon detection of any VRUs, however, a high-level DRL collision avoidance controller is activated to prompt the vehicle to either decelerate or change its trajectory to prevent potential collisions. The CARLA simulator is used to train the proposed DRL collision avoidance controller, and virtual raw sensor data are utilized to enhance the realism of the simulations. The model-in-the-loop (MIL) methodology is utilized to assess the efficacy of the proposed DRL ADS routine. In comparison to the traditional DRL end-to-end approach, which combines high-level decision making with low-level control, the proposed hierarchical DRL agents demonstrate superior performance.

**Keywords:** autonomous driving system; deep reinforcement learning; collision avoidance



**Citation:** Chen, H.; Cao, X.; Guvenc, L.; Aksun-Guvenc, B. Deep-Reinforcement-Learning-Based Collision Avoidance of Autonomous Driving System for Vulnerable Road User Safety. *Electronics* **2024**, *13*, 1952. <https://doi.org/10.3390/electronics13101952>

Academic Editor: Yolanda Blanco Fernández

Received: 15 April 2024

Revised: 13 May 2024

Accepted: 14 May 2024

Published: 16 May 2024



**Copyright:** © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

At present, rapid urbanization and technological development have led to a steady rise in privately owned vehicles. This widespread adoption of private cars makes our lives more convenient by letting us travel easily between our workplaces and homes. However, this convenience is shadowed by growing concerns that are mostly due to the accompanying rapid rise in vehicular accidents. According to the World Health Organization's Global Status Report on Road Safety, over 50 million individuals get injured and approximately 1.3 million lives are lost in car accidents globally each year [1]. A significant portion of these accidents, estimated at around 75%, are directly attributable to driver errors, including drowsy driving, driving under the influence (DUI), and distracted driving [2]. This statistic highlights the critical need for interventions aimed at mitigating human error on the roads. How to effectively reduce traffic accidents has become a challenge that modern cities must confront.

The ADS benefits from powerful and robust autonomous driving algorithms, which can significantly reduce car accidents caused by human mistakes. The Society of Automotive Engineers (SAE) categorized autonomous driving into six levels, ranging from level 0, no automation, to level 5, full automation [3]. The high levels of autonomous driving, especially levels 4 and 5, have the potential to significantly reduce accidents caused by

human error. This includes algorithms to improve the safety of VRUs in their interactions with autonomous vehicles (AVs). Therefore, the development of AV collision avoidance algorithms for VRU safety has become a popular and promising area of research.

Extensive research has already been conducted in this field [4–7]. Currently, most path planning and collision avoidance research follows two major directions. The first is the optimization-based approach, which conceptualizes path planning and collision avoidance as an optimization problem with specific constraints and aims to plan an optimal collision-free trajectory. Guvenc et al. considered the collision avoidance problem as one of waypoint optimization and applied the Elastic Band algorithm to iteratively generate a new trajectory for autonomous vehicles [8]. Wang et al. further applied this algorithm in a low-speed electric shuttle and modified the parameter to make the ego-vehicle maintain a socially acceptable distance with VRUs [9]. Morsali et al. proposed a Support Vector Machine (SVM)-based spatial-temporal planning method, which, combined with the A\* path search algorithm and SVM-based heuristics, can efficiently identify optimal collision-free paths in complex traffic situations [10]. Zhu treated collision avoidance path planning as a quintic splines optimization problem and applied look-up tables to enhance computational efficiency [11]. Chen et al. introduced an enhanced spatio-temporal obstacle avoidance algorithm that benefited from a 3D spatio-temporal grid map to achieve better efficiency and performance compared to the traditional hybrid A\* algorithm [12]. Meanwhile, a large number of similar studies have been performed to tackle the same problem [13–16]. However, the optimization-based approach often cannot guarantee satisfactory real-time performance due to computational complexity, and, sometimes, it does not take control feasibility into consideration. Flow-based path planning and collision avoidance has also been the focus of recent research, like the unmanned air mobility study in reference [17], with a simulated annealing and parallel computing solution, and path planning for shared autonomous vehicles in a dynamic traffic network in reference [18].

The second approach is the machine learning method, which considers autonomous driving and collision avoidance tasks as Markov Decision Processes (MDPs) and utilizes the reinforcement learning method to find optimal solutions. Kendall et al. pioneered the application of the DRL framework in autonomous driving, an innovation proposing an end-to-end model structure for autonomous driving [19]. Yurtsever et al. proposed an innovative hybrid DRL framework to develop ADS [20]. Peng et al. developed an end-to-end ADS utilizing a Dueling Double Deep Q-Network (DDDQN) framework. The Open Racing Car Simulator (TORCS) is utilized to validate the method's efficiency and effectiveness [21]. Jaritz et al. introduced an Asynchronous-Actor-Critic-based method for autonomous driving that maps the RGB image from the front camera to driving actions and using a realistic rally game environment for training. The approach demonstrates faster convergence and robust performance compared to other DRL-based end-to-end methods, indicating its potential for practical applications in autonomous vehicles [22]. In order to handle critical pre-accident scenarios in emergency situations, Merola et al. proposed a Deep Q-Network (DQN)-based approach to design an ADS and to train the system to execute emergency maneuvers to minimize or avoid damage [23]. Cao et al. introduced a hierarchical reinforcement and imitation learning (H-REIL) approach for autonomous driving to handle near-accident scenarios. By integrating a low-level imitation learning controller with a high-level reinforcement learning controller, their approach demonstrated the capability of balancing safety and efficiency [24]. Additionally, a substantial amount of research is currently ongoing within machine learning approaches [25–29]. Many studies combine DRL with traditional approaches, intending to leverage DRL to enhance the performance of conventional methods [30,31]. However, a notable disadvantage of the machine-learning-based approach is the instability in model performance under normal traffic conditions due to the absence of hard-coded safety protocols. Moreover, the overall performance of the machine-learning-based approach heavily relies on the quality of the training data.

This paper introduces a novel, hybrid, hierarchical DRL framework aimed at developing high-efficiency collision avoidance controllers for autonomous driving. This innovative controller integrates a PID-based pure pursuit path-tracking controller with a DRL-based collision avoidance agent. In normal, collision-free situations, a low-level path-following controller is employed for precise navigation. Conversely, when vulnerable road users are detected near the vehicle, a high-level DRL collision avoidance controller is activated to help the vehicle avoid the potential collision by decelerating or changing the trajectory. The major contribution of the research is mainly in the integration of traditional optimization-based path-following methods with machine-learning-based collision-avoidance algorithms. This proposed hybrid approach aims to enhance both path tracking and collision avoidance performance, thereby establishing solid groundwork for future research in hybrid control strategies for autonomous vehicles.

## 2. Methodology

### 2.1. Vehicle Model

A simplified vehicle dynamic model is used in this paper to guarantee meaningful training results. This model is a longitudinal dynamic augmented linear single-track lateral model. The geometry of the longitudinal vehicle model is illustrated in Figure 1. The symbols in Figure 1 mean (1)  $F_a$  = aerodynamic drag caused by headwind with velocity  $V_{wind}$ ; (2)  $F_r$  = rolling resistance; (3)  $\theta$  = road grade; (4)  $Mg$  = vehicle gravitational force; (5)  $F_g$  = force component of vehicle gravitational force along the road grade; and (6)  $F_x$  = longitudinal tire force. The model's input–output structure is illustrated in Figure 2. It uses throttle and brake pedal actions together with headwind velocity and road grade as inputs and longitudinal velocity as the output. It is worth noting that for the sake of simplicity, the headwind and road grade are assumed to be non-existent/zero. The detailed structure of this longitudinal model is illustrated in Figure 3, where Figure 3a demonstrates the calculation procedure from longitudinal forces to the longitudinal vehicle velocity, and Figure 3b demonstrates the calculation procedure from the input engine and brake torques to the longitudinal tire force. Table 1 lists the parameters that appear in Figure 3.

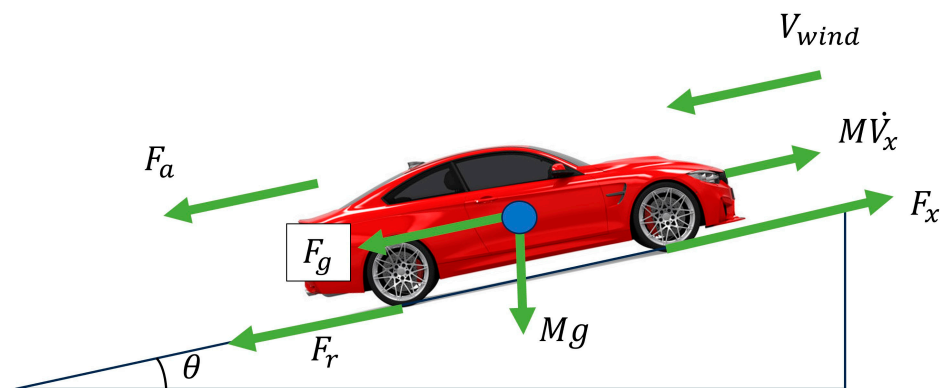


Figure 1. Longitudinal vehicle model geometry.

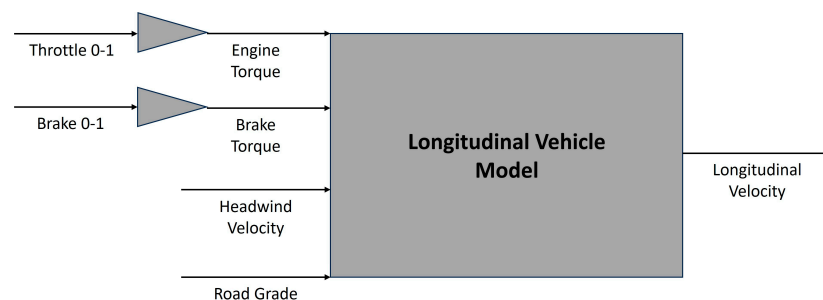
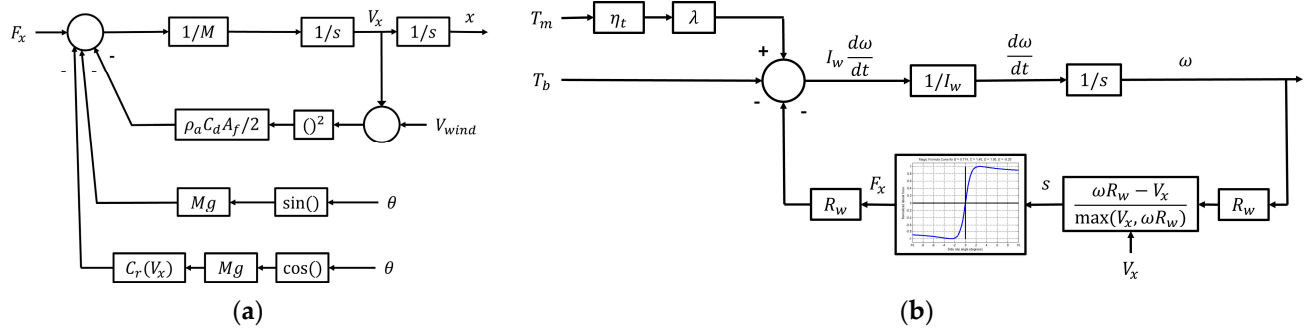


Figure 2. Longitudinal vehicle model input–output structure.



**Figure 3.** Detailed components in the longitudinal vehicle model. (a) Longitudinal forces to longitudinal vehicle velocity calculation. (b) Input torques to longitudinal tire force calculation.

**Table 1.** Longitudinal model parameters.

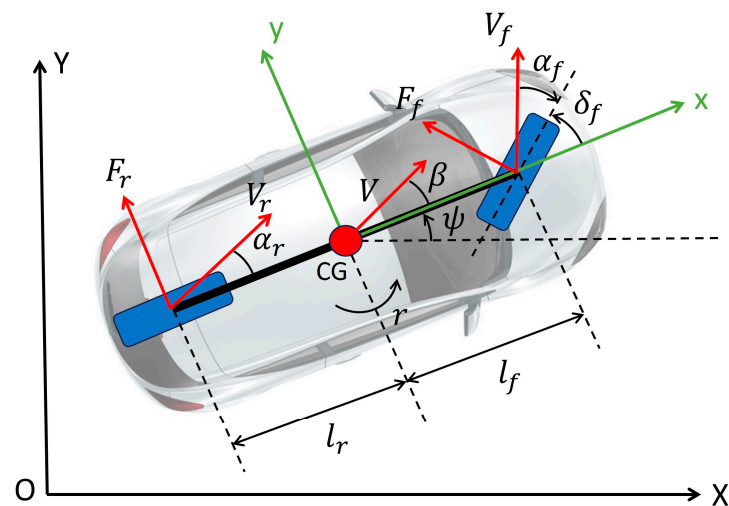
Symbol	Parameter
$F_x$	Longitudinal tire force
$M$	Vehicle mass
$V_x$	Vehicle longitudinal velocity
$x$	Vehicle longitudinal position
$\rho_a$	Air density
$C_d$	Air drag coefficient
$A_f$	Vehicle cross-sectional area
$V_{wind}$	Headwind velocity
$\theta$	Road grade
$C_r$	Rolling resistance coefficient
$T_m$	Motor torque
$T_b$	Brake torque
$\eta_t$	Transmission efficiency
$\lambda$	Gear ratio
$I_w$	Wheel moment of inertia
$\omega$	Wheel angular velocity
$R_w$	Wheel radius
$s$	Longitudinal tire slip

The lateral single-track model geometry is displayed in Figure 4, and the state-space form representation of the simplified linear single-track model is illustrated in Equation (1) [8]. The parameters used in this lateral model are listed in Table 2. The inputs are the front and rear wheel steer angle as well as the vehicle yaw disturbance, and the outputs are the vehicle side-slip angle and the vehicle yaw rate. In this case, we assumed a front-wheel-steer vehicle and no external yaw disturbance; thus, the input is simply the front wheel steer angle. It is also worth noting that in this simplified linear model, vehicle's longitudinal speed is treated as being constant.

$$\begin{bmatrix} \dot{\beta} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} \frac{-C_f - C_r}{MV} & -1 + \frac{C_r l_r - C_f l_f}{MV^2} \\ \frac{C_r l_r - C_f l_f}{I_z} & \frac{-C_f l_f^2 - C_r l_r^2}{I_z V} \end{bmatrix} \begin{bmatrix} \beta \\ r \end{bmatrix} + \begin{bmatrix} \frac{C_f}{MV} & \frac{C_r}{MV} \\ \frac{C_f l_f}{I_z} & \frac{C_r l_r}{I_z} \end{bmatrix} \begin{bmatrix} \delta_f \\ \delta_r \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{I_z} \end{bmatrix} M_{zd} \quad (1)$$

The structure of the overall full vehicle model is displayed in Figure 5. This model uses throttle, brake, and steering actions as its inputs and outputs vehicle X and Y positions as well as the vehicle yaw angle. As mentioned in the previous paragraph, the lateral portion of the model assumes constant speed. As a result, the lateral component must be implemented as a time-varying model, where its parameters can change at each time step.

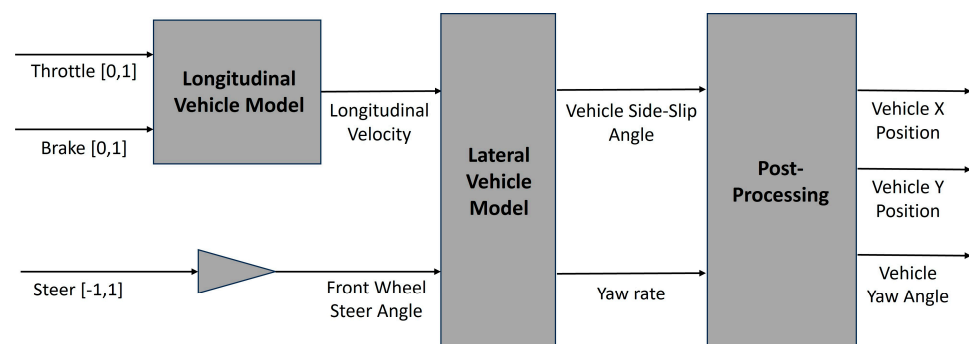




**Figure 4.** Lateral single-track vehicle model geometry.

**Table 2.** Lateral model parameters.

Symbol	Parameter
$X, Y$	Earth-fixed frame coordinate
$x, y$	Vehicle-fixed frame coordinate
$V$	Vehicle center-of-gravity (CG) velocity
$M$	Vehicle mass
$I_z$	Vehicle yaw moment of inertia
$\beta$	Vehicle side-slip angle
$\psi$	Vehicle yaw angle
$r$	Vehicle yaw rate
$M_{zd}$	Yaw disturbance moment
$\delta_f, \delta_r$	Front and rear wheel steer angle
$F_f, F_r$	Front and rear lateral tire force
$V_f, V_r$	Front and rear axle velocity
$\alpha_f, \alpha_r$	Front and rear tire slip angle
$l_f, l_r$	Distance between vehicle CG and front and rear axle
$C_f, C_r$	Front and rear tire cornering stiffness



**Figure 5.** Full vehicle model structure.

## 2.2. Autonomous Driving System Design

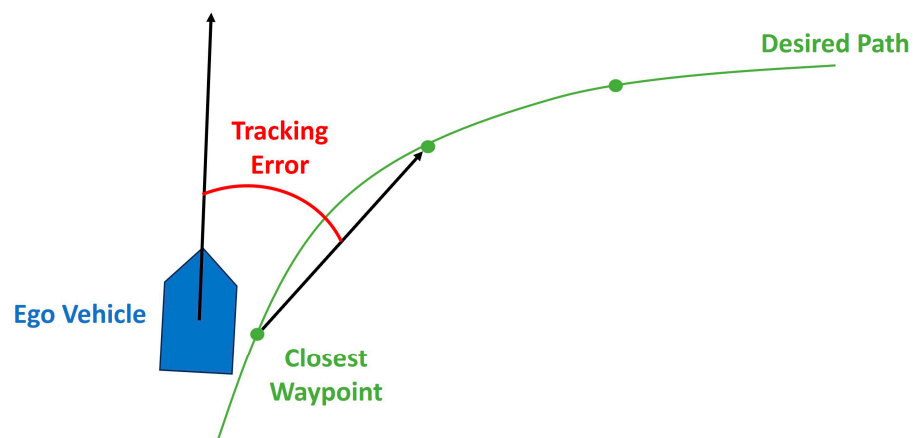
In this paper, an innovative, hybrid, hierarchical, deep-reinforcement-learning-based controller is proposed for autonomous vehicles for tracking a pre-determined path and performing an evasive collision avoidance maneuver when necessary. The architecture of this hybrid controller is designed to leverage the strengths of both the PID pure pursuit controller for precise path-following and the DRL-based collision avoidance controller for safe and efficient collision avoidance. The detailed design of this hybrid control system is presented in the following.

### 2.2.1. PID Pure Pursuit Controller Design

A PID pure pursuit path-tracking controller is utilized to perform precise path-following under normal traffic conditions. The proposed PID controller consists of a longitudinal PID controller and a lateral PID controller. The longitudinal PID controller is primarily responsible for controlling the vehicle's speed. The difference in speed between the current speed of the vehicle and the desired speed, demonstrated in Equation (2), is used as the input to generate a Speed command. Meanwhile, the lateral PID controller is mainly used for controlling the vehicle's steering. The difference in the angle between the vehicle's current moving direction and the desired path direction, given in Equation (3), is taken as the input to generate a vehicle steering command. Figure 6 demonstrates the method for calculating the angle difference input to the lateral PID controller. The controller gains ( $K_p$ ,  $K_i$ ,  $K_d$ ) of both the longitudinal and lateral PID controllers have been manually turned to obtain reasonable values according to the vehicle model.

$$\varepsilon_{\text{longitudinal}} = v_{\text{vehicle}} - v_{\text{desired}} \quad (2)$$

$$\varepsilon_{\text{lateral}} = \theta_{\text{vehicle}} - \theta_{\text{path}} \quad (3)$$



**Figure 6.** Lateral PID controller path-tracking error.

### 2.2.2. Markov Decision Process and Deep Reinforcement Learning

The collision avoidance of an autonomous vehicle is a dynamic and continuous decision-making process. Initially, the AV must identify the positions, velocities, and trajectories of nearby road users. Subsequently, based on the vehicle's own location and velocity, the AV must make prompt and accurate decisions to control the vehicle and dodge other road users safely. This indicates that all decisions made by the AV are correlated with the ego-vehicle's current state as well as the states of nearby road users. This process is very similar to an MDP. Given this analogy, the task of collision avoidance can be treated as a classic MDP, wherein MDPs are employed to model and address the uncertainties inherent in traffic environments. To tackle this complex MDP, the deep reinforcement learning method is utilized to design an autonomous driving system, which can make optimal decisions that aim to maximize expected rewards and enhance the safety and efficiency of the collision avoidance procedure. The fundamental elements of an MDP consist of states ( $S$ ), actions ( $A$ ), transition probabilities ( $P$ ), and rewards ( $r$ ). The setup of the MDP used in this paper is demonstrated as follows.

- **State space ( $S$ ):** The state space contains a collection of states that represent the current traffic environment's information. Each state within the state space consists of four essential components. Firstly, an occupancy grid represented by a 2D array is utilized to map the surrounding obstacles relative to the vehicle. The occupancy grid can identify other road users as obstacles within a specified range and assign a weight to each grid based on the importance of nearby road users. A relatively higher weight

will be assigned to vulnerable road users (VRUs). In this paper, the detection ranges are defined as 20 m forward, 5 m backward, and 7 m to the sides. Figures 7 and 8 demonstrate the occupancy grid for a vehicle with zero yaw angle orientation and non-zero yaw angle orientation, respectively. In the figure, the cross symbols are used to indicate the proposed occupancy grid: white crosses for collision-free areas, black for potential collisions with the pedestrian, red for the vehicle's geometric center, and blue for the vehicle's vertical and horizontal coordinates. In practical implementations, the occupancy grid is typically derived through the fusion of data from multiple sensors, such as the lidar, camera, and radar. However, the purpose of this study is to validate the proposed hybrid controller. Thus, to simplify the data collection procedure, the occupancy grid array data are extracted and processed from the simulation environment through frame transformation techniques, enabling precise collision detection for each array point with obstacles. The second component contains the ego-vehicle's data, such as its location, orientation, and velocity. The third component contains information regarding the pre-calculated path, including target-tracking waypoints. Lastly, the fourth component contains obstacle information, including the vehicle time-to-collision-zone (TTZ\_v), the pedestrian time-to-collision-zone (TTZ\_p), and the difference between vehicle and pedestrian time-to-collision-zones (TTZ\_diff).

- Action space ( $A$ ): The action space contains a collection of discrete actions available to the ego-vehicle in response to varying traffic scenarios. The action is defined as a control command tuple consisting of steering, throttle, and brake commands. The configuration of the action space is designed according to the specific requirements of different test cases.
- Transition model ( $P$ ): The transition model is the key component of traffic simulation designed to simulate next states based on the execution of a given action at the current state and the transition probability. This paper divides the transition model into two major parts. Firstly, a SIMULINK vehicle model simulates the motion of the ego-vehicle. At the same time, the CARLA simulator is used to simulate the traffic environment and the motion of other road users. The details of the vehicle model have been discussed in previous sections, while the details of the traffic simulator are elaborated within the case study section.
- Reward ( $r$ ): A reward function is used to calculate the immediate reward for each step based on the transition from the current state to next state after executing specific actions. The details of the reward function's design are elaborated within the case study section.

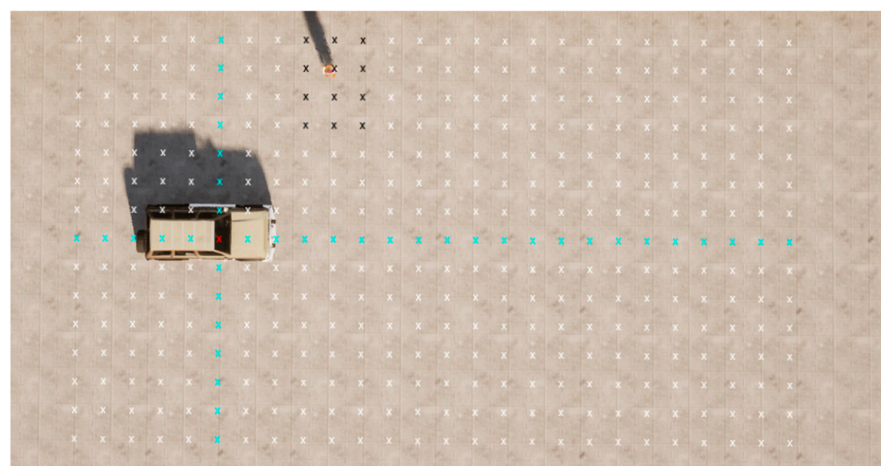


Figure 7. Occupancy grid 2D array with zero yaw angle.



**Figure 8.** Occupancy grid 2D array with 30° yaw angle.

To design a high-performance collision avoidance system, the Double Deep Q-Network (DDQN) approach is utilized to address the previously outlined MDP and seek an optimal policy that maximizes cumulative rewards. The DDQN is an enhanced version of the DQN, which itself is a breakthrough in the field of reinforcement learning. DQN, developed by DeepMind Technologies in 2013, successfully combined Q-learning, a popular reinforcement learning algorithm, with deep neural networks to handle high-dimensional state spaces [32–34]. DQN’s architecture contains many important innovations, such as the replay buffer and two distinct neural networks, the Q-network and the target-network. However, one of the limitations of the original DQN model is its tendency to overestimate action values due to the maximization step in the Q-learning update. This overestimation can lead to suboptimal policy and unstable learning. Hasselt et al. proposed DDQN in 2015, which successfully addressed this issue by decoupling the action selection from the target Q-value generation during the learning process [34]. Like its predecessor, DDQN leverages the strength of both the neural network and Q-learning, and it is designed to address complex MDP problems. DDQN stands out from conventional collision avoidance algorithms, such as the optimization-based approach or the supervised learning method. Unlike the optimization-based approach, DDQN can handle high-dimensional inputs, like the aforementioned occupancy grid array. Moreover, in contrast to supervised learning methods, which depend on pre-labeled training datasets, DDQN learns directly from interactions with the environment, making it exceptionally suitable for training in autonomous driving tasks. Thus, DDQN is expected to be a superior alternative for collision avoidance compared to traditional approaches. Table 3 presents the comparison between traditional optimization-based approaches and the proposed DDQN framework to demonstrate the benefits of choosing DDQN.

**Table 3.** Comparison to traditional optimization-based approaches.

Approaches	Pros	Cons
Elastic Band [9]	1. Easy to implement. 2. Can avoid getting stuck at local minimum. 3. Flexibility in local path modification.	1. Path shape may be irregular, especially in complex environment. 2. Computational complexity may increase with number of obstacles. 3. Path may be control-infeasible.
Potential-Field-related	1. Easy to implement. 2. Can achieve real-time performance. 3. Path easy to visualize and understand.	1. Sometimes stuck at local minimum, especially in complex environment. 2. Oscillations may occur around obstacles. 3. Path may be control-infeasible.
SVM-based optimization [10]	1. Path planning in spatial–temporal region. 2. Can find optimal, efficient, and control-feasible path.	1. Sometimes stuck at local minimum, especially in complex environment. 2. Oscillations may occur around obstacles. 3. Path may be control-infeasible.
Other optimization-based method [11,35]	1. Can generate control-feasible and optimal (either time- or fuel-efficient) path. 2. Can adapt to different traffic scenarios.	1. Computationally inefficient and may not achieve real-time performance. 2. Performance of the optimization might be sensitive to the tuning of parameters.
Proposed DDQN-based ADS	1. Learning ability. 2. Model can achieve real-time performance. 3. Can adapt to different traffic scenarios.	1. Training requires good computational resources. 2. Performance of the model depends on training data quality.

Compared to other DRL algorithms, DDQN has its unique advantages. Currently, there are two primary approaches in the DRL field: policy-based and value-based methods. Policy-based methods, such as Proximal Policy Optimization (PPO), focus on directly optimizing the policy that dictates the agent's actions, aiming to improve the expected long-term rewards. These methods are particularly effective in handling high-dimensional or continuous action spaces, and they are known for their robustness and stability during training [36]. However, such an approach can be less training-efficient than value-based methods like DQN and DDQN, requiring more interactions with the environment to achieve similar performance because it must discard old data after updating the policy. On the other hand, value-based methods, such as DDQN, focus on estimating the values of actions from each state, thereby optimizing a policy indirectly by selecting actions that maximize these estimated values. Because value-based methods are usually off-policy, they can reuse the previously generated data to enhance the training efficiency, and they have been used in collision avoidance research [37–40]. Moreover, another popular approach is the Deep Deterministic Policy Gradient (DDPG). DDPG blends both policy-based and value-based elements. It uses a policy network to determine actions and a value network to evaluate them, and it is suitable for environments with continuous action spaces. However, DDPG is very complex to implement in the CARLA setting, and the performance of DDPG is very sensitive to hyperparameter settings, which require extensive computational resources. Thus, in this paper, DDQN is utilized as the DRL framework to develop ADS collision avoidance.

The purpose of DDQN is to learn from interactions with the environmental and to let the neural network model approximate the optimal action–value function. Unlike DQN, which directly employs the  $Q$  function to determine the maximum future rewards, DDQN introduces a crucial modification to mitigate the overestimation of action values. In DDQN, the loss function is:

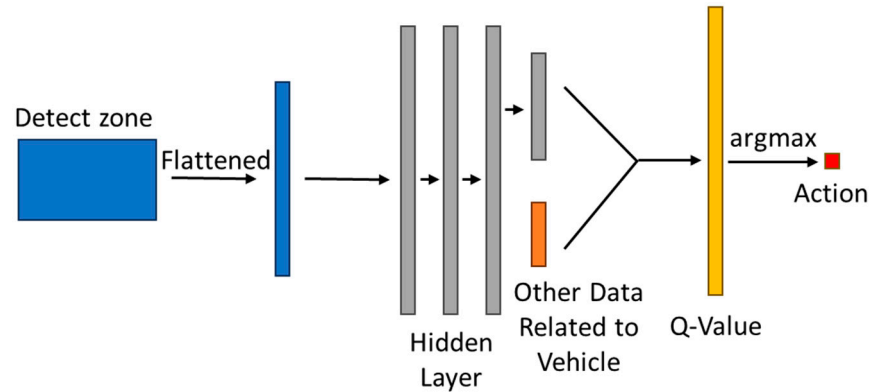
$$L_i(\theta) = \mathbb{E}_{(s,a,r)} \left[ \left( r + \gamma \max_{a_{t+1}} Q_{\theta_i^-} \left( s_{t+1}, \arg \max_{a_{t+1}} Q_{\theta_i} (s_{t+1}, a_{t+1}) \right) - Q_{\theta_i} (s_t, a_t) \right)^2 \right] \quad (4)$$

where  $i$  represents the iteration number,  $s$  represents the state,  $a$  represents the action,  $r$  represents the immediate reward,  $\theta_i$  represents the  $Q$ -network, and  $\theta_i^-$  represents the target-network. In DDQN,  $\max_{a_{t+1}} Q_{\theta_i^-} \left( s_{t+1}, \arg \max_{a_{t+1}} Q_{\theta_i} (s_{t+1}, a_{t+1}) \right)$  is used to replace the  $\max_{a_{t+1}} Q_{\theta_i^-} (s_{t+1}, a_{t+1})$ , which decouples the selection of the action from the evaluation of that action's value and can effectively reduce the overestimation bias of the  $Q$ -value.

The architecture of the neural network employed in the DDQN is shown in Figure 9. This network consists of four fully connected hidden layers: three layers each containing 128 units, followed by a final layer with 32 units. The process begins with the occupancy grid being flattened and fed into the first, second, and third 128-unit hidden layers in sequence. Following the third layer, additional sensor data, including information about the ego-vehicle, path-tracking waypoints, and other road users, are combined with the output from the third layer and fed into the fourth hidden layer, which contains 32 units. This design strategy is intentional, aiming to preserve important information that might otherwise be lost in the initial processing layers, thereby ensuring that important details are considered in the network's final output. In addition, the detailed flowchart of the DDQN algorithm is shown in Algorithm 1. The DDQN algorithm proposed in this study employs a dual neural network architecture to optimize decision making. The online neural network  $Q$  is tasked with generating optimal actions for the given state. Meanwhile, the target neural network  $\hat{Q}$  is utilized for performing gradient descent, with its parameters being updated only after a predefined number of steps. This staged update procedure is designed to stabilize the training process by mitigating rapid fluctuations in learning targets. The simulation steps generated by the SIMULINK vehicle model and the CARLA environment are stored in a replay buffer, which enables the breaking of correlations between consecutive



training samples. This approach significantly reduces the variance in model training and enhances the efficiency of data utilization by ensuring a more uniform and comprehensive sampling from the replay buffer during training sessions. In this paper, the implementation details for the training of the DDQN are as follows. 1. The learning rate is set at 0.01. 2. The initial probability of selecting an action randomly is 1, indicating that exploration starts at its maximum. 3. The decay period for the probability of randomly selecting an action is 200,000 steps. 4. The final probability of selecting an action randomly is reduced to 0.05, balancing exploration with exploitation. 5. The reward discount factor is set at 0.9, influencing how future rewards are valued relative to immediate rewards. 6. Each episode is limited to a maximum of 4000 steps to constrain the training duration per episode.



**Figure 9.** DDQN framework neural network structure.

---

**Algorithm 1.** DDQN

---

```

1: Initialize replay memory  $D$ 
2: Initialize target network  $\hat{Q}$  and Online Network  $Q$  with random weights  $\theta$ 
3: for each episode do
4:   Initialize traffic environment
5:   for  $t = 1$  to  $T$  do
6:     With probability  $\epsilon$  select a random action  $a_t$ 
7:     Otherwise select  $a_t = \max_a Q^*(s_t, a; \theta)$ 
8:     Execute  $a_t$  in CARLA and extract reward  $r_t$  and next state  $s_{t+1}$ 
9:     Store transition  $(s_t, a_t, r_t, s_{t+1})$  in  $D$ 
10:    if  $t \bmod \text{training frequency} == 0$  then
11:      Sample random minibatch of transitions  $(s_j, a_j, r_j, s_{j+1})$  from  $D$ 
12:      Set  $y_j = r_j + \gamma \max_{a_{j+1}} \hat{Q}(s_{j+1}, \arg\max_{a_{j+1}} Q(s_j, a_{j+1}; \theta); \theta)$ 
13:      for non-terminal  $s_{j+1}$ 
14:        or  $y_j = r_j$  for terminal  $s_{j+1}$ 
15:        Perform a gradient descent step to update  $\theta$ 
16:      Every  $N$  steps reset  $\hat{Q} = Q$ 
17:    end if
18:    Set  $s_{t+1} = s_t$ 
19:  end for
20: end for

```

---

### 2.3. Vehicle-in-Virtual-Environment (VVE)

ADAS testing involving pedestrians poses a significant safety concern. As a result, a safer and more efficient approach is required to reliably implement the proposed pedestrian collision avoidance experiment. Vehicle-in-virtual-environment (VVE) is an ideal alternative for this purpose. VVE allows for the motion synchronization between the real test vehicle operating in a safe, open space and a virtual vehicle operating in a highly detailed and realistic 3D virtual environment. It also allows for the motion synchronization between real and virtual pedestrians, where the virtual pedestrian operates in the same environment as the virtual vehicle. This enables the pedestrian collision avoidance experiment to be carried out as depicted in Figure 10. Preliminary results of vehicle motion synchronization and vehicle-to-pedestrian (V2P) connectivity using the VVE method are discussed in detail in [41].



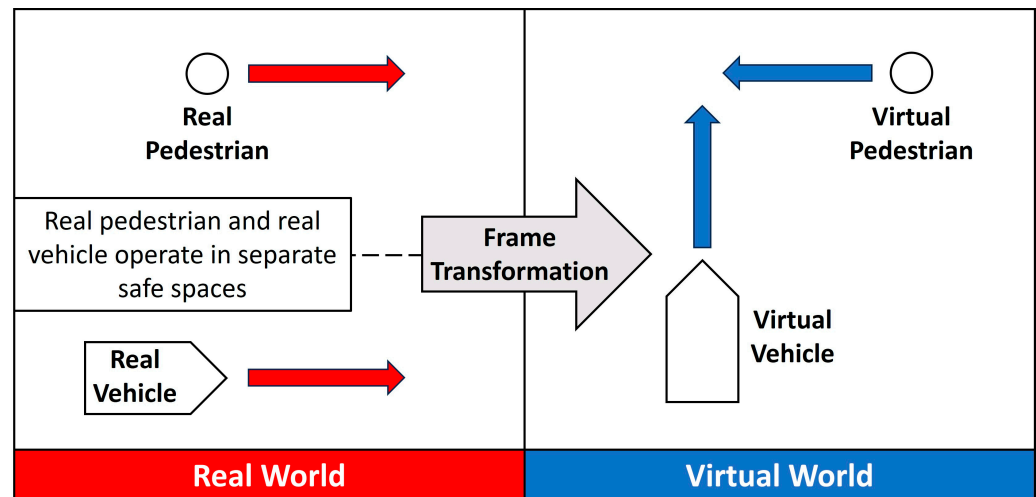


Figure 10. One possible pedestrian collision avoidance test setup using the VVE method.

### 3. Case Study

To demonstrate the capabilities of the proposed routine, two traffic scenarios of pedestrian collision avoidance are introduced for model-in-the-loop evaluation implementation. The test is conducted in the CARLA realistic animation environment with the SIMULINK vehicle dynamic model being used to facilitate the model-in-the-loop test configuration.

#### 3.1. Scenario 1

The first traffic scenario is illustrated in Figure 11. In this scenario, the pedestrian enters the crosswalk as the vehicle approaches. The crosswalk hence becomes the potential collision zone, and the vehicle must execute a gradual slow-down and stop before it reaches the edge of the zone. It should be noted that the DRL module action space will only include throttle and brake actions, as only a longitudinal motion maneuver needs to be performed. It should be additionally noted that for this scenario, several exemplary slowing-down speed profiles are generated for the DRL module to learn from so that the routine does not rely on random action selection as much. This significantly reduces the time needed to train the model.

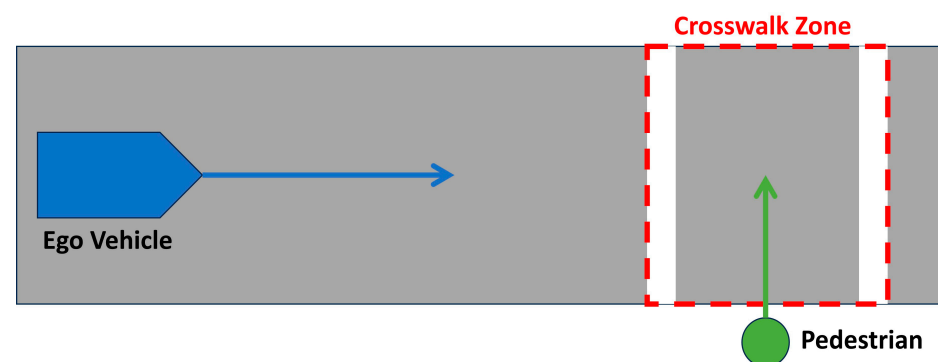
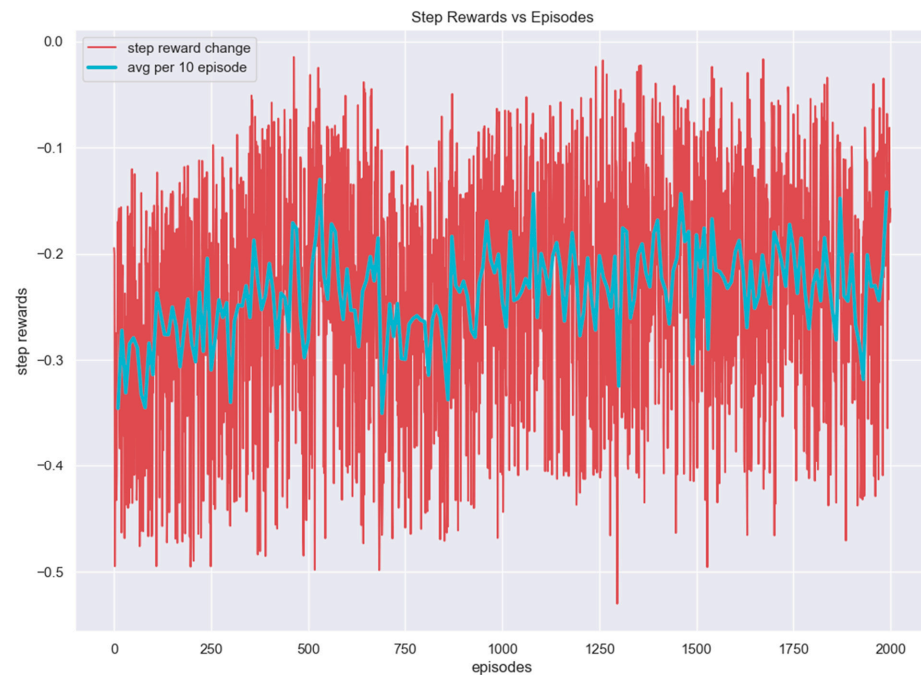


Figure 11. Traffic Scenario 1 setup.

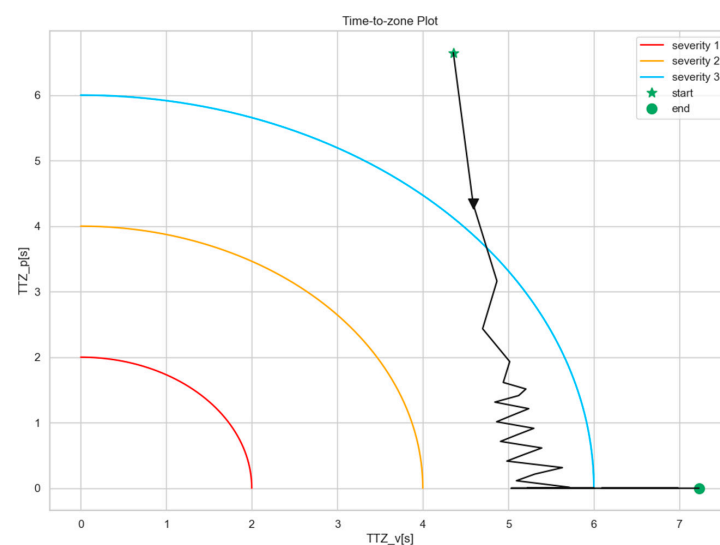
Figure 12 shows the progression of the step rewards as the training episodes increase. It can be observed that the rewards increase with episodes and tend to plateau after 1000 episodes, indicating convergence and hence a successful search for the optimal policy. It is notable that this particular test case is relatively straightforward, allowing for a scenario where even randomly generated actions have a reasonable probability of completing the driving task successfully. Consequently, the reward progression does not show significant improvement across training episodes after the initial period, reflecting the simplicity of the task. However, the proposed deep reinforcement learning (DRL) model demonstrates

its effectiveness by enabling the vehicle to successfully complete test cases all of the time, showcasing its robust capability in typical traffic scenarios.

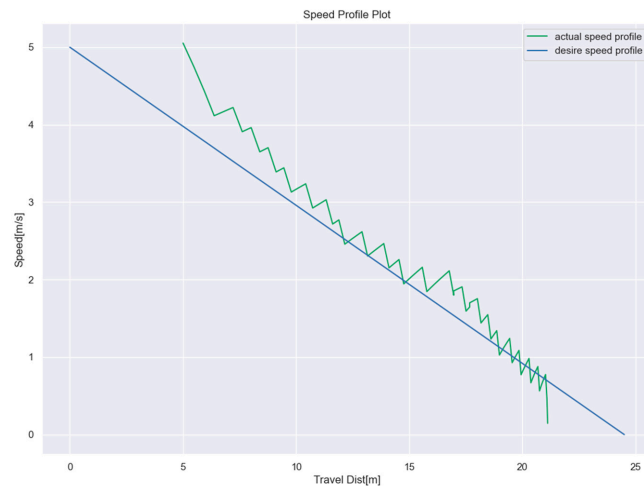


**Figure 12.** Scenario 1 step rewards vs. training episodes.

Figure 13 illustrates the evolution of TTZ for both the pedestrian and the vehicle. The three colored arcs denote the boundaries of the severity levels, with level one being the most critical, requiring emergent actions to avoid likely collisions, and level three being the least urgent, where the vehicle has plenty of time and space to react to possible collision risks. It can be observed that the TTZ for both the pedestrian and the vehicle is consistently larger than four seconds, indicating the low likelihood of collision, proving the efficacy of the optimal policy. Figure 14 displays the progression of vehicle speed versus time, where it can be observed that the DRL agent is capable of reasonable speed following performance. It is worth noting that the vehicle's speed does not track the desired slow-down profile initially, and this is most likely due to actuator saturation. The link to the Scenario 1 demo video is attached here: <https://youtu.be/fgXN1hhA6qk> (accessed on 14 May 2024).



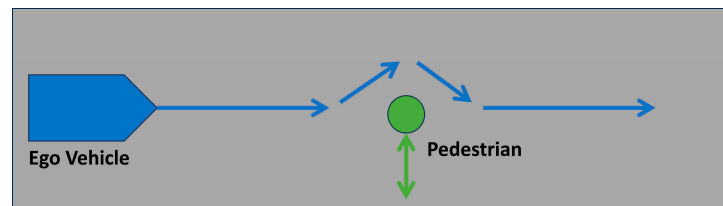
**Figure 13.** Scenario 1 TTZ progression.



**Figure 14.** Scenario 1 speed following performance.

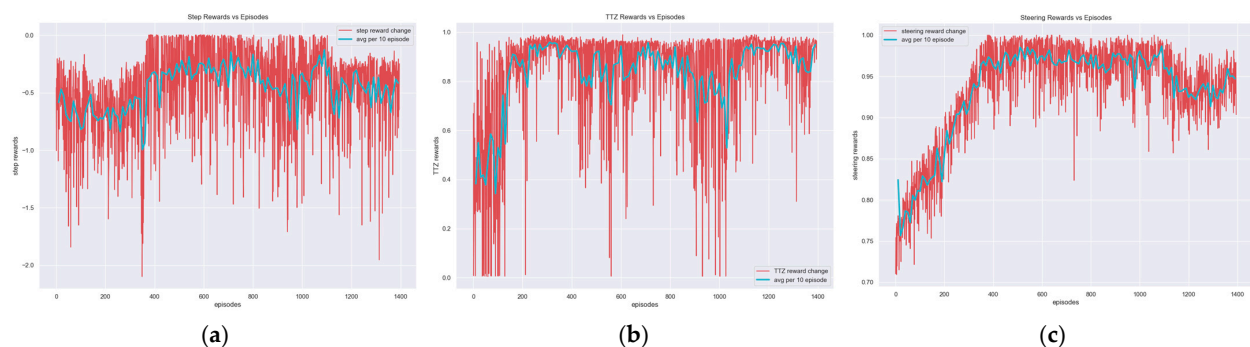
### 3.2. Scenario 2

The second traffic scenario is displayed in Figure 15. In this scenario, the pedestrian enters the road at a random point as the vehicle approaches, establishing a potential collision risk. In this case, the pedestrian is treated as a moving obstacle for the vehicle, and the vehicle must go around the pedestrian by steering clear to avoid possible collisions. With this setup, the action space of the DRL module will include not only longitudinal controls of throttle and brake inputs but also steering actions.

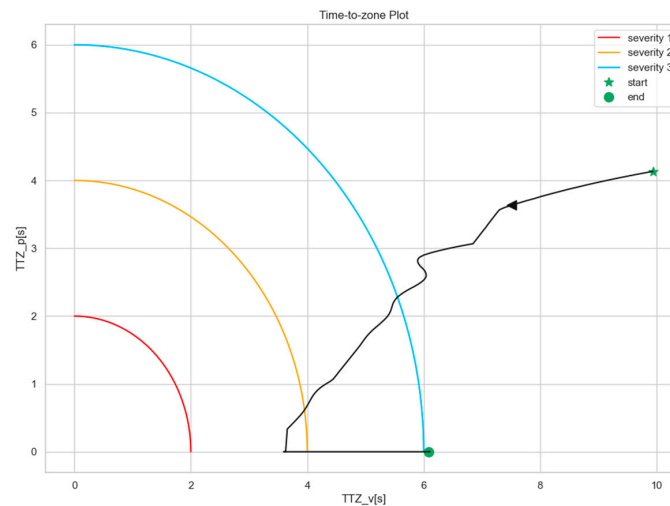


**Figure 15.** Traffic Scenario 2 setup.

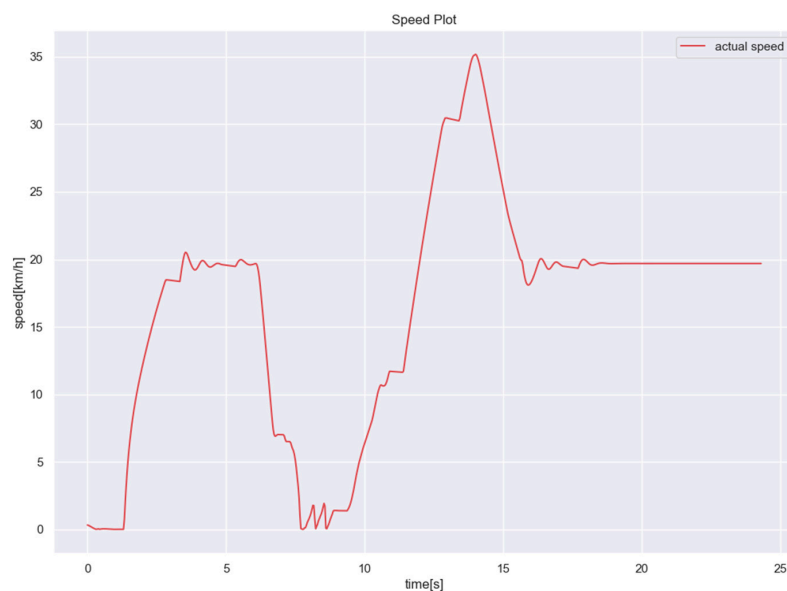
Figure 16 illustrates the reward progression of Scenario 2 training. It can again be observed that the rewards increase as more episodes are completed, and they eventually level off at a high level, indicating a successful optimal policy search. In particular, the convergence of the TTZ reward suggests that the agent learns to avoid collision with the pedestrian, a point that is proven by Figure 17, where collision risk is low. The convergence of the steering reward indicates that the agent also tends to select a reasonable amount of steering during the maneuver. It is also worth noting that the agent tends to slow down as it approaches the pedestrian and then speeds up when the collision risk is clear, as shown in Figure 18, and this is the desirable reaction during this kind of maneuver. The link to the Scenario 2 demo video is attached here: [https://youtu.be/CmGtaAjZ\\_x4](https://youtu.be/CmGtaAjZ_x4) (accessed on 14 May 2024).



**Figure 16.** Training rewards vs. episodes. (a) Step reward. (b) TTZ reward. (c) Steering reward.



**Figure 17.** Scenario 2 TTZ progression.



**Figure 18.** Scenario 2 vehicle speed progression.

#### 4. Conclusions and Future Work

This paper addressed vulnerable road user (VRU) safety by integrating a novel hierarchical deep reinforcement learning (DRL) framework into the autonomous driving system (ADS) to enable automated collision avoidance. A traditional PID controller was used for path-following and speed-following when no VRU was in the vicinity of the vehicle, and the DRL module was activated to start collision avoidance when VRU(s) (pedestrians) enter(s) the vehicle detection zone. A model-in-the-loop (MIL) case study was conducted, and the proposed approach demonstrated satisfactory performance, which demonstrates the capability of the proposed DRL framework to handle moving obstacle problems, instead of the traditional static obstacle navigation problem. For future work, this approach can be further implemented using the vehicle-in-virtual-environment (VVE) method. Other DRL frameworks and neural network structures can also be applied. CARLA is also not mandatory for testing. Instead, other virtual environments, possibly combined with more powerful and realistic vehicle models, such as high-fidelity CarSim models, can be utilized to show its multi-actor capability. In addition, mixed-reality (XR) goggles can be integrated into the test, which, together with the VVE method, can facilitate both safe testing and enhanced realism for the participants.

**Author Contributions:** Conceptualization, all authors; methodology, H.C. and X.C.; validation, H.C. and X.C.; formal analysis, all authors; investigation, all authors; resources, B.A.-G. and L.G.; data curation, H.C. and X.C.; writing—original draft preparation, H.C. and X.C.; writing—review and editing, B.A.-G. and L.G.; visualization, all authors; supervision, B.A.-G. and L.G.; project administration, B.A.-G. and L.G.; funding acquisition, B.A.-G. and L.G. All authors have read and agreed to the published version of the manuscript.

**Funding:** This project is funded in part by Carnegie Mellon University’s Safety21 National University Transportation Center, which is sponsored by the US Department of Transportation.

**Data Availability Statement:** Data are contained within the article.

**Acknowledgments:** The authors thank NVIDIA for its GPU donations. The authors thank the Automated Driving Lab at Ohio State University for its support.

**Conflicts of Interest:** The authors declare no conflicts of interest.

## References

1. World Health Organization. *Global Status Report on Road Safety 2015*; World Health Organization: Geneva, Switzerland, 2015; Available online: <https://iris.who.int/handle/10665/189242> (accessed on 24 October 2023).
2. Medina, A.; Lee, S.; Wierwille, W.; Hanowski, R. Relationship between Infrastructure, Driver Error, and Critical Incidents. *Proc. Hum. Factors Ergon. Soc. Annu. Meet.* **2004**, *48*, 2075–2079. [\[CrossRef\]](#)
3. J3016\_202104: Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles—SAE International. Available online: [https://www.sae.org/standards/content/j3016\\_202104/](https://www.sae.org/standards/content/j3016_202104/) (accessed on 24 October 2023).
4. Ye, F.; Zhang, S.; Wang, P.; Chan, C.-Y. A Survey of Deep Reinforcement Learning Algorithms for Motion Planning and Control of Autonomous Vehicles. In Proceedings of the 2021 IEEE Intelligent Vehicles Symposium (IV), Nagoya, Japan, 11–17 July 2021; pp. 1073–1080. [\[CrossRef\]](#)
5. Kiran, B.R.; Sobh, I.; Talpaert, V.; Mannion, P.; Al Sallab, A.A.; Yogamani, S.; Pérez, P. Deep Reinforcement Learning for Autonomous Driving: A Survey. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 4909–4926. [\[CrossRef\]](#)
6. Zhu, Z.; Zhao, H. A Survey of Deep RL and IL for Autonomous Driving Policy Learning. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 14043–14065. [\[CrossRef\]](#)
7. Zha, Y.; Deng, J.; Qiu, Y.; Zhang, K.; Wang, Y. A Survey of Intelligent Driving Vehicle Trajectory Tracking Based on Vehicle Dynamics. *SAE Int. J. Veh. Dyn. Stab. NVH* **2023**, *7*, 221–248. [\[CrossRef\]](#)
8. Autonomous Road Vehicle Path Planning and Tracking Control | IEEE eBooks | IEEE Xplore. Available online: <https://ieeexplore.ieee.org/book/9645932> (accessed on 24 October 2023).
9. Wang, H.; Tota, A.; Aksun-Guvenc, B.; Guvenc, L. Real time implementation of socially acceptable collision avoidance of a low speed autonomous shuttle using the elastic band method. *Mechatronics* **2018**, *50*, 341–355. [\[CrossRef\]](#)
10. Morsali, M.; Frisk, E.; Åslund, J. Spatio-Temporal Planning in Multi-Vehicle Scenarios for Autonomous Vehicle Using Support Vector Machines. *IEEE Trans. Intell. Veh.* **2021**, *6*, 611–621. [\[CrossRef\]](#)
11. Zhu, S. Path Planning and Robust Control of Autonomous Vehicles. Ph.D. Thesis, The Ohio State University, Columbus, OH, USA, 2020. Available online: <https://www.proquest.com/docview/2612075055/abstract/73982D6BAE3D419APQ/1> (accessed on 24 October 2023).
12. Chen, G.; Yao, J.; Gao, Z.; Gao, Z.; Zhao, X.; Xu, N.; Hua, M. Emergency Obstacle Avoidance Trajectory Planning Method of Intelligent Vehicles Based on Improved Hybrid A\*. *SAE Int. J. Veh. Dyn. Stab. NVH* **2023**, *8*, 3–19. [\[CrossRef\]](#)
13. Ararat, Ö.; Güvenç, B.A. Development of a Collision Avoidance Algorithm Using Elastic Band Theory. *IFAC Proc. Vol.* **2008**, *41*, 8520–8525. [\[CrossRef\]](#)
14. Emirler, M.T.; Wang, H.; Güvenç, B. Socially Acceptable Collision Avoidance System for Vulnerable Road Users. *IFAC Pap.* **2016**, *49*, 436–441. [\[CrossRef\]](#)
15. Gelbal, S.Y.; Guvenc, B.A.; Guvenc, L. SmartShuttle: A Unified, Scalable and Replicable Approach to Connected and Automated Driving in A Smart City. In *Proceedings of the 2nd International Workshop on Science of Smart City Operations and Platforms Engineering*; In SCOPE '17; Association for Computing Machinery: New York, NY, USA, 2017; pp. 57–62. [\[CrossRef\]](#)
16. Guvenc, L.; Guvenc, B.A.; Emirler, M.T. Connected and Autonomous Vehicles. In *Internet of Things and Data Analytics Handbook*; John Wiley & Sons, Ltd.: Hoboken, NJ, USA, 2017; pp. 581–595. [\[CrossRef\]](#)
17. Wang, Z.; Delahaye, D.; Farges, J.L.; Alam, S.S. A quasi-dynamic air traffic assignment model for mitigating air traffic complexity and congestion for high-density UAM operations. *Transp. Res. Part C Emerg. Technol.* **2023**, *154*, 104279. [\[CrossRef\]](#)
18. Maruyama, R.; Seo, T.T. Integrated public transportation system with shared autonomous vehicles and fixed-route transits: Dynamic traffic assignment-based model with multi-objective optimization. *Int. J. Intell. Transp. Syst. Res.* **2023**, *21*, 99–114. [\[CrossRef\]](#)
19. Kendall, A.; Kendall, A.; Hawke, J.; Janz, D.; Mazur, P.; Reda, D.; Allen, J.M.; Lam, V.-D.; Bewley, A.; Shah, A. Learning to Drive in a Day. In Proceedings of the 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; pp. 8248–8254. [\[CrossRef\]](#)

20. Yurtsever, E.; Capito, L.; Redmill, K.; Ozgune, U. Integrating Deep Reinforcement Learning with Model-based Path Planners for Automated Driving. In Proceedings of the 2020 IEEE Intelligent Vehicles Symposium (IV), Las Vegas, NV, USA, 19 October–13 November 2020; pp. 1311–1316. [\[CrossRef\]](#)
21. Peng, B.; Sun, Q.; Li, S.E.; Kum, D.; Yin, Y.; Wei, J.; Gu, T. End-to-End Autonomous Driving Through Dueling Double Deep Q-Network. *Automot. Innov.* **2021**, *4*, 328–337. [\[CrossRef\]](#)
22. Jaritz, M.; de Charette, R.; Toromanoff, M.; Perot, E.; Nashashibi, F. End-to-End Race Driving with Deep Reinforcement Learning. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, QLD, Australia, 21–25 May 2018. [\[CrossRef\]](#)
23. Merola, F.; Falchi, F.; Gennaro, C.; Di Benedetto, M. Reinforced Damage Minimization in Critical Events for Self-driving Vehicles. In *Proceedings of the 17th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications; Online Streaming, —Select a Country—: SCITEPRESS—Science and Technology Publications; SciTePress: Setúbal, Portugal, 2022; pp. 258–266. [CrossRef]*
24. Cao, Z.; Biyik, E.; Wang, W.Z.; Raventos, A.; Gaidon, A.; Rosman, G.; Sadigh, D. Reinforcement Learning based Control of Imitative Policies for Near-Accident Driving. *arXiv* **2020**, arXiv:2007.00178. [\[CrossRef\]](#)
25. Nagesh Rao, S.; Tseng, H.E.; Filev, D. Autonomous Highway Driving using Deep Reinforcement Learning. In Proceedings of the 2019 IEEE International Conference on Systems, Man and Cybernetics (SMC), Bari, Italy, 6–9 October 2019; pp. 2326–2331. [\[CrossRef\]](#)
26. Deep Reinforcement-Learning-Based Driving Policy for Autonomous Road Vehicles—Makantasis—2020—IET Intelligent Transport Systems—Wiley Online Library. Available online: <https://ietresearch.onlinelibrary.wiley.com/doi/full/10.1049/iet-its.2019.0249> (accessed on 24 October 2023).
27. Aksjonov, A.; Kyrki, V. A Safety-Critical Decision-Making and Control Framework Combining Machine-Learning-Based and Rule-Based Algorithms. *SAE Int. J. Veh. Dyn. Stab. NVH* **2023**, *7*, 287–299. [\[CrossRef\]](#)
28. Knox, W.B.; Allievi, A.; Banzhaf, H.; Schmitt, F.; Stone, P. Reward (Mis)design for autonomous driving. *Artif. Intell.* **2023**, *316*, 103829. [\[CrossRef\]](#)
29. Wang, Y.; Wei, H.; Yang, L.; Hu, B.; Lv, C. A Review of Dynamic State Estimation for the Neighborhood System of Connected Vehicles. *SAE Int. J. Veh. Dyn. Stab. NVH* **2023**, *7*, 367–385. [\[CrossRef\]](#)
30. Lu, S.; Xu, R.; Li, Z.; Wang, B.; Zhao, Z. Lunar Rover Collaborated Path Planning with Artificial Potential Field-Based Heuristic on Deep Reinforcement Learning. *Aerospace* **2024**, *11*, 253. [\[CrossRef\]](#)
31. Xi, Z.; Han, H.; Cheng, J.; Lv, M. Reducing Oscillations for Obstacle Avoidance in a Dense Environment Using Deep Reinforcement Learning and Time-Derivative of an Artificial Potential Field. *Drones* **2024**, *8*, 85. [\[CrossRef\]](#)
32. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Graves, A.; Antonoglou, I.; Wierstra, D.; Riedmiller, M. Playing Atari with Deep Reinforcement Learning. *arXiv* **2013**, arXiv:1312.5602. [\[CrossRef\]](#)
33. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A.A.; Veness, J.; Bellemare, M.G.; Graves, A.; Riedmiller, M.; Fidjeland, A.K.; Ostrovski, G.; et al. Human-level control through deep reinforcement learning. *Nature* **2015**, *518*, 7540. [\[CrossRef\]](#)
34. van Hasselt, H.; Guez, A.; Silver, D. Deep Reinforcement Learning with Double Q-Learning. *Proc. AAAI Conf. Artif. Intell.* **2016**, *30*, 1. [\[CrossRef\]](#)
35. Zhang, X.; Liniger, A.; Borrelli, F. Optimization-Based Collision Avoidance. *IEEE Trans. Control Syst. Technol.* **2021**, *29*, 972–983. [\[CrossRef\]](#)
36. Mu, C.; Liu, S.; Lu, M.; Liu, Z.; Cui, L.; Wang, K. Autonomous spacecraft collision avoidance with a variable number of space debris based on safe reinforcement learning. *Aerosp. Sci. Technol.* **2024**, *149*, 109131. [\[CrossRef\]](#)
37. Feng, S.; Sebastian, B.; Ben-Tzvi, P. A Collision Avoidance Method Based on Deep Reinforcement Learning. *Robotics* **2021**, *10*, 73. [\[CrossRef\]](#)
38. Wang, C.; Zhang, X.; Yang, Z.; Bashir, M.; Lee, K. Collision avoidance for autonomous ship using deep reinforcement learning and prior-knowledge-based approximate representation. *Front. Mar. Sci.* **2023**, *9*, 1084763. [\[CrossRef\]](#)
39. Sun, Z.; Fan, Y.; Wang, G. An Intelligent Algorithm for USVs Collision Avoidance Based on Deep Reinforcement Learning Approach with Navigation Characteristics. *J. Mar. Sci. Eng.* **2023**, *11*, 812. [\[CrossRef\]](#)
40. de Curtò, J.; de Zarzà, I. Analysis of Transportation Systems for Colonies on Mars. *Sustainability* **2024**, *16*, 3041. [\[CrossRef\]](#)
41. Cao, X.; Chen, H.; Gelbal, S.Y.; Guvenc, B.A.; Guvenc, L. *Vehicle-in-Virtual-Environment Method for ADAS and Connected and Automated Driving Function Development, Demonstration and Evaluation*; SAE Technical Paper 2024-01-1967; SAE International: Warrendale, PA, USA, 2024; Available online: <https://www.sae.org/publications/technical-papers/content/2024-01-1967/> (accessed on 7 March 2024).

**Disclaimer/Publisher’s Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.