Article

Game-Theoretic Adversarial Interaction-Based Critical Scenario Generation for Autonomous Vehicles

Xiaokun Zheng [1,2,*], Huawei Liang [1,3,4], Jian Wang [1,2] and Hanqi Wang [1,2]

1   Hefei Institutes of Physical Science, Chinese Academy of Sciences, Hefei 230031, China;
     hwliang@iim.ac.cn (H.L.); sa20wj@mail.ustc.edu.cn (J.W.); whq97@mail.ustc.edu.cn (H.W.)
2   University of Science and Technology of China, Hefei 230026, China
3   Anhui Engineering Laboratory for Intelligent Driving Technology and Application, Hefei 230031, China
4   Innovation Research Institute of Robotics and Intelligent Manufacturing, Chinese Academy of Sciences,
     Hefei 230031, China
*   Correspondence: zhengkun@mail.ustc.edu.cn

Abstract: Ensuring safety and efficiency in the rapidly advancing autonomous vehicle (AV) industry presents a significant engineering challenge. Comprehensive performance evaluations and critical scenario testing are essential for identifying situations that AVs cannot handle. Thus, generating critical scenarios is a key problem in AV testing system design. This paper proposes a game-theoretic adversarial interaction method to efficiently generate critical scenarios that challenge AV systems. Initial motion prediction for adversarial and surrounding vehicles is based on kinematic models and road constraints, establishing interaction action spaces to determine possible driving domains. A novel evaluation approach combines reachability sets with adversarial intensity to assess collision risks and adversarial strength for any state, used to solve behavior values for each interaction action state. Further, equilibrium action strategies for the vehicles are derived using Stackelberg game theory, yielding optimal actions considering adversarial interactions in the current traffic environment. Simulation results show that the adversarial scenarios generated by this method significantly increase incident rates by 158% to 1313% compared to natural driving scenarios, while ride comfort and driving efficiency decrease, and risk significantly increases. These findings provide critical insights for model improvement and demonstrate the proposed method's suitability for assessing AV performance in dynamic traffic environments.

Keywords: autonomous vehicles; critical scenario generation; game theory; adversarial interaction; testing efficiency

1. Introduction

1.1. Motivation

Autonomous vehicles (AVs) present a promising solution for enhancing traffic efficiency and safety [1,2]. However, recent incidents during both testing and actual deployment of AVs [3] have raised significant safety concerns. Therefore, ensuring the safe operation of AV systems in various complex and dynamic traffic scenarios is a critical prerequisite for advancing autonomous driving technology.

Comprehensive and rigorous testing is essential to guarantee the safety of AVs. Traditional testing methods typically depend on predefined scenarios and the expertise of human specialists. These methods are not only time-consuming and labor-intensive but also inadequate in covering all possible situations, especially extreme or rare scenarios. Hence, the efficient generation of critical scenarios that can robustly challenge AV systems has become a focal point of current research.

1.2. Related Work

In recent years, with the proliferation of deep generative models, evaluating autonomous vehicles (AVs) in artificially generated scenarios has become a mainstream approach [4]. Significant breakthroughs have been made in testing AVs in natural driving environments [5–7]; however, critical safety scenarios rarely occur in the real world. This makes it difficult to identify failure modes of AVs and improve models accordingly. Therefore, there is an urgent need for research on the generation of safety-critical scenarios.

According to the literature [8], methods for generating safety-critical scenarios can be classified into three categories: data-driven generation, knowledge-based generation, and adversarial generation.

Data-driven generation methods use information from collected datasets to generate scenarios. Since driving scenarios usually follow a distribution, collected data can be used to train density models to fit this distribution [9]. For instance, importance sampling theory is used in [10] to evaluate critical driving events, and the cross-entropy method is employed to find importance sampling functions within pre-designed parameter combinations. Huang et al. [11] also use importance sampling theory to generate critical interactions between different vehicles, accelerating intelligent driving tests, and establishing a piecewise mixed distribution model to simulate the maneuvers of leading vehicles.

Knowledge-based generation methods use predefined rules or explicit knowledge to guide the generation of critical scenarios. For example, in [12], prior knowledge is used to design random risk scenarios, and reinforcement learning agents are trained in various scenarios to achieve comparable results. In [13], explicit knowledge is encoded as first-order logic [14] and embedded into a tree structure to enable controlled generation. Reference [15] extracts various global road data from OpenStreetMap [16] and employs the SUMO [17] traffic simulator to generate safety-critical scenarios for these road networks.

Adversarial generation is another popular research topic, aimed at obtaining specific scenarios to evaluate the robustness of AVs [4,18]. The main purpose of adversarial generation is to degrade the performance of the target model or create safety risks. For example, ref. [19] proposes fusion and parallel attack models to attack AV perception algorithms, making intelligent driving systems dangerous. Xu et al. [20] use iterative projection gradient algorithms to attack semantic segmentation models. Kong et al. [21] propose using PhysGAN to generate physically robust adversarial examples that continuously mislead AVs. Additionally, ref. [22] adopts generative adversarial imitation learning models to generate adversarial driving behaviors. Reference [23] uses deep reinforcement learning to generate adversarial environments. Reference [24] proposes a scenario generation algorithm that frames the adversarial objective as a black-box attack, testing lidar-based autonomous systems by disrupting the trajectories of actors.

These methods can generate adversarial interaction scenarios to interfere with the tested AV systems and evaluate their performance. However, data-driven generation methods require a large number of prior accident scenarios, which are often lacking. Knowledge-based methods heavily depend on knowledge and experience, making it challenging to handle diverse and complex scenarios. Although adversarial methods can quickly generate numerous safety-critical scenarios, they primarily use machine learning and deep learning theories, requiring extensive data training. This process is inherently a black-box model, leading to many unreasonable adversarial behaviors.

1.3. Contributions

To address the dependency on original data and the issue of unreasonable adversarial behaviors in current adversarial scenario generation methods, this paper focuses on the behavior of adversarial vehicles (Advs) and proposes a game-theoretic method for generating critical scenarios for autonomous vehicles. The main contributions of this paper are as follows:

(1) **Construction of Game-Theoretic Adversarial Interaction Behaviors:** By introducing game theory, a decision-making algorithm specifically targeting Advs is designed,

make adversarial behaviors more reasonable and human-like, thereby avoiding unreasonable adversarial actions.

(2) **Quantitative Grading of Adversarial Intensity:** Drawing on the concept of reachability sets, the adversarial intensity of scenarios is quantitatively graded to test AVs with different capabilities. By quantifying adversarial intensity, the performance of AV systems under different levels of adversarial challenges can be more accurately assessed.

(3) **Comprehensive Evaluation and Testing Efficiency Improvement:** In addition to safety, the evaluation metrics include the comfort and driving performance of autonomous vehicles, providing a more comprehensive assessment system. The proposed method enables the intuitive generation of a large number of critical scenarios, significantly enhancing the testing efficiency of autonomous driving systems.

1.4. Structure of the Paper

The overall structure of this paper is as follows:

Section 2: **Model Establishment and Preparation:** This section provides a detailed introduction to the modeling methods for vehicle state sequences and trajectory predictions during multi-agent game interactions. It includes the construction of kinematic models for Adv motion trend predictions and the establishment of interaction action spaces.

Section 3: **Establishment of Evaluation Functions:** This section describes the design of various action payoff functions, including the evaluation of vehicle driving performance, the establishment of risk evaluation functions, and the design of a quantitative model for adversarial intensity.

Section 4: **Solution of the Adversarial Interaction Decision-Making Model:** Based on the evaluation functions established in Section 3, this section further considers the process of interaction games and derives the equilibrium actions for two vehicles using Stackelberg game theory.

Section 5: **Experiments:** This section verifies the rationality and effectiveness of the proposed game-theoretic adversarial interaction-based critical scenario generation method through the design of adversarial interaction scenarios. It includes experiments on adversarial interaction capabilities, validation of graded adversarial capabilities, and the generation of critical test scenarios.

Section 6: **Conclusions:** This section summarizes the main research work and findings of this paper and proposes directions for future research.

2. Model Establishment and Preparation

In the process of multi-agent interactions among AVs, vehicle state sequences and trajectory predictions are essential. To extract these profiles from the vehicle–road system, the preparation work mainly includes: **Kinematic Modeling:** This involves obtaining the transformation from action states (throttle, braking, and steering control) to vehicle trajectory states. **Physics-Based Surrounding Vehicle Trajectory Prediction:** This method reflects the future trajectory trends of various vehicles in the traffic scene. **Construction of Interaction Action Spaces:** This represents the possible driving domains of AVs within the decision-making time horizon, providing intent choices for the interactive game decision-making process among multiple AVs.

2.1. Construction of the Kinematic Model for Adversarial Vehicles

To derive the equations governing the changes in vehicle state based on control inputs, dynamic models and bicycle models have been widely used, as seen in [25,26]. For motion planning, a particle motion model is sufficient to generate smooth and continuous trajectories. The particle motion model is established in the Frenet coordinate system, as shown in Figure 1.
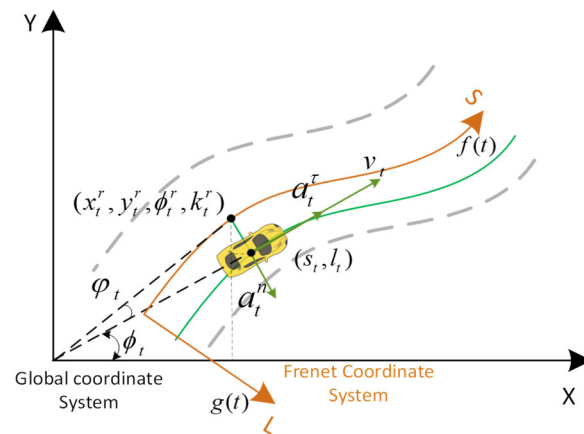
Figure 1. Conversion Relationship between Frenet Coordinate System and Global Coordinate System.

Typically, the trajectory of a moving vehicle can be expressed as a function of longitudinal and lateral positions (s_t, l_t) over time, from which states such as speed, acceleration, and heading angle can be derived. These are represented by the equations:

$$\begin{cases} s_t & = f(t) \\ l_t & = g(t) \end{cases} \tag{1}$$

where $f(t)$ and $g(t)$ are functions of the longitudinal and lateral positions in the Frenet coordinate system, respectively. Due to the smoothness of the trajectory, the position functions can be expanded using the Taylor series. Additionally, as the errors are sufficiently small and continuously re-planned, higher-order terms are maintained as second-order, as shown in Equation (2).

$$\begin{bmatrix} s_{t+1} \\ l_{t+1} \end{bmatrix} = \begin{bmatrix} f(t) \\ g(t) \end{bmatrix} + \begin{bmatrix} \dot{f}(t) \\ \dot{g}(t) \end{bmatrix} T + \begin{bmatrix} \ddot{f}(t) \\ \ddot{g}(t) \end{bmatrix} \frac{T^2}{2!} \tag{2}$$

where T is the planning period. Each of these can be expressed in terms of actual states as follows.

$$\begin{cases} f(t) = s_t; g(t) = l_t; \dot{f}(t) = v_t\cos\varphi_t; \dot{g}(t) = v_t\sin\varphi_t \\ \ddot{f}(t) = a_t^\tau\cos\varphi_t - a_t^n\sin\varphi_t; \ddot{g}(t) = a_t^\tau\sin\varphi_t + a_t^n\cos\varphi_t \end{cases} \tag{3}$$

where v_t is speed, φ_t is heading angle, a_t^τ is longitudinal acceleration determined by the throttle and brake, and a_t^n is lateral acceleration determined by the steering, which defines the curvature of the path.

Thus, from the inputs of longitudinal and lateral accelerations (a_t^τ, a_t^n), the vehicle state $(s_{t+1}, l_{t+1}, v_{t+1}, \varphi_{t+1})$ can be updated as follows:

$$\begin{cases} \begin{bmatrix} s_{t+1} \\ l_{t+1} \end{bmatrix} = \begin{bmatrix} s_t \\ l_t \end{bmatrix} + T \begin{bmatrix} v_t \cdot \cos\varphi_t \\ v_t \cdot \sin\varphi_t \end{bmatrix} + \frac{T^2}{2!} \begin{bmatrix} \cos\varphi_t & -\sin\varphi_t \\ \sin\varphi_t & \cos\varphi_t \end{bmatrix} \cdot \begin{bmatrix} a_t^\tau \\ a_t^n \end{bmatrix} \\ \begin{bmatrix} v_{t+1} \\ \varphi_{t+1} \end{bmatrix} = \begin{bmatrix} v_t \\ \varphi_t \end{bmatrix} + T \cdot \begin{bmatrix} 1 & 0 \\ 0 & 1/v_t \end{bmatrix} \begin{bmatrix} a_t^\tau \\ a_t^n \end{bmatrix} \end{cases} \tag{4}$$

Additionally, the positions, speed, and heading angle $(s_t, l_t, v_t, \varphi_t)$ in the Frenet coordinate system can be converted to the global coordinate system $(x_t, y_t, v_t^o, \phi_t)$ based on the road reference information $(x_t^r, y_t^r, \phi_t^r, k_t^r)$.

$$\begin{cases} x_t = x_t^r - l_t \sin\phi_t^r \\ y_t = y_t^r + l_t \cos\phi_t^r \\ v_t^o = \sqrt{[v_t \cos\varphi_t(1 - k_t^r l_t)]^2 + [v_t \sin\varphi_t]^2} \\ \phi_t = \phi_t^r + \varphi_t \end{cases} \tag{5}$$

where $(x_t^r, y_t^r, \phi_t^r, k_t^r)$ represent the longitudinal and lateral positions, heading angle, and curvature in the global coordinate system, respectively, as shown in Figure 1. This abstraction process enables the motion planning decisions to be transformed into specific sequences of longitudinal and lateral accelerations.

2.2. Motion Trend Prediction

In complex traffic scenarios, predicting a vehicle's future trajectory is influenced not only by its own motion constraints but also by the driver's intent, making accurate trajectory prediction challenging. This section first uses a motion prediction model to anticipate the motion trends of surrounding vehicles, then combines these with game theory to infer specific driving intentions.

When predicting the future trajectories of surrounding vehicles, the main information from sensors such as radar, lidar, and cameras includes:

$$(\Delta s_t, \Delta l_t, v_t^s, a_t^s, \Delta\varphi_t, w_t^s, \alpha_t^s) \tag{6}$$

where Δs_t and Δl_t are the relative positions, v_t^s and a_t^s are the speed and acceleration of the surrounding vehicle, respectively, $\Delta\varphi_t$ is the relative heading angle, w_t^s is the yaw rate, and α_t^s is the lateral acceleration of the surrounding vehicle.

Therefore, based on the sensor data, the future state of the surrounding vehicle in the Frenet coordinate system can be expressed as:

$$\begin{cases} s_t^s = s_t + \Delta s_t; l_t^s = l_t + \Delta l_t; \varphi_t^s = \varphi_t + \Delta\varphi_t \\ a_t^{s-\tau} = a_t^s; a_t^{s-n} = w_t^s \cdot v_t^s \end{cases} \tag{7}$$

where (s_t^s, l_t^s) are the future positions of the surrounding vehicle (SV). φ_t^s is the future heading angle, and $(a_t^{s-\tau}, a_t^{s-n})$ are the longitudinal and lateral accelerations of the SV, respectively.

Due to the inherent uncertainties in these conditions, directly predicting the trajectory based on sensor information is challenging. Therefore, this paper separately predicts the longitudinal and lateral trajectories and then combines them to form the overall trajectory.

Typically, the motion of a vehicle stabilizes after a single driving maneuver. At the endpoint $t + N_p$, the lateral velocity and acceleration tend to zero. The initial position, lateral position, speed, and acceleration are subject to constraints. The five boundary conditions at the initial and endpoint can be used to fit a fourth-degree polynomial for the lateral trajectory as follows:

$$\begin{cases} l_t = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 \\ \text{s.t.} \\ l|_t = l_t^s, \dot{l}\big|_t = v_t^s \sin\varphi_t^s, \ddot{l}\big|_t = a_t^{s-\tau}\sin\varphi_t^s + a_t^{s-n}\cos\varphi_t^s \\ \dot{l}\big|_{t+N_p} = 0, \ddot{l}\big|_{t+N_p} = 0 \end{cases} \tag{8}$$

where $a_i(i = 0 \sim 4)$ are the fitting coefficients of the fourth-degree polynomial used to fit the lateral motion trajectory.

Similarly, the longitudinal trajectory can also be represented by a polynomial. The initial position constraints are the same. At $t + N_p$, only the acceleration is constrained:

$$
\begin{cases}
s_t = b_0 + b_1 t + b_2 t^2 + b_3 t^3 \\
\text{s.t.} \\
s\big|_t = s_t^s, \; \dot{s}\big|_t = v_t^s \cos\varphi_t^s, \; \ddot{s}\big|_t = a_t^{s-\tau}\cos\varphi_t^s - a_t^{s-n}\sin\varphi_t^s \\
\ddot{s}\big|_{t+Np} = 0
\end{cases}
\tag{9}
$$

where $b_i (i = 0 \sim 3)$ are the polynomial coefficients representing the longitudinal trajectory.

The longitudinal and lateral trajectories predicted based on the kinematic model are obtained separately. Since the path and speed are planned in a coupled manner, the longitudinal and lateral motion trajectories need to be coupled.

For the interval $[t, t + N_p]$, the states can be coupled as follows using functions s and l.

$$
\begin{cases}
v_{t+i}^s = \sqrt{\left(v_{t+i}^{s-s}\right)^2 + \left(v_{t+i}^{s-l}\right)^2} \\
\varphi_{t+i}^s = \arctan\left(v_{t+i}^{s-l}/v_{t+i}^{s-s}\right) \\
a_{t+i}^{s-\tau} = a_{t+i}^{s-l}\sin\varphi_{t+i}^s + a_{t+i}^{s-s}\cos\varphi_{t+i}^s \\
a_{t+i}^{s-n} = a_{t+i}^{s-l}\cos\varphi_{t+i}^s - a_{t+i}^{s-s}\sin\varphi_{t+i}^s
\end{cases}
\tag{10}
$$

where $v_{t+i}^{s-s} = \dot{s}\big|_{t+i}; v_{t+i}^{s-l} = \dot{l}\big|_{t+i}; a_{t+i}^{s-s} = \ddot{s}\big|_t; a_{t+i}^{s-l} = \ddot{l}\big|_t$.

Thus, the perceived state can derive the local trajectories of surrounding vehicles and the ego vehicle. The prediction results for three positions are shown in Figure 2. The predicted trajectories reflect the current motion trends.
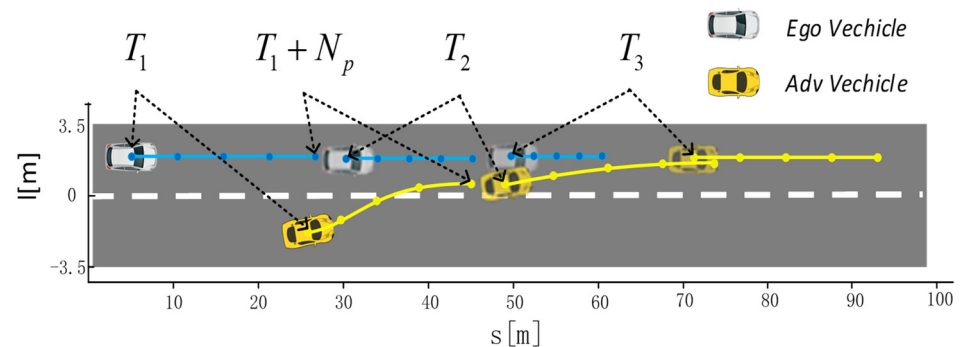

Figure 2. Kinematic model-based trajectory prediction at the moments T_1, T_2, and T_3 with a prediction duration of N_p.

2.3. Construction of Interaction Action Spaces

After roughly predicting the trajectories of the surrounding vehicles using the motion trend prediction model, it is necessary to represent the potential driving regions of both the vehicle and surrounding vehicles within the decision-making horizon to facilitate further interaction.

Typically, human drivers' behaviors include acceleration, deceleration, lane change (left or right), and maintaining current behavior. As shown in Figure 3, the motion space starts with five basic action sequences. Here, B^1 is the acceleration sequence, B^2 and B^4 are the lane change sequences (left and right, respectively), B^3 is the deceleration sequence, and B^5 is the sequence maintaining the current behavior. B^5 is based on the polynomial model described in Equations (8)–(10), representing the corresponding endpoints for maintaining the current driving intention.
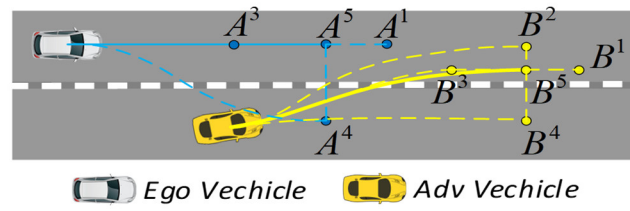
Figure 3. Action Spaces for the Vehicle and Adversarial Vehicle under Different Driving States.

Taking $B^5 \sim \left(S_{t+N_p}, L_{t+N_p} \right)$ as the baseline, the endpoints for the acceleration, deceleration, and lane change actions can be derived for B^1, B^3, and B^2, B^4 as follows:

$$
\begin{cases}
\| B^1 - B^5 \| = s_t + v_t N_p + a_{\max}(N_p)^2/2 - S_{t+N_p} \\
\| B^3 - B^5 \| = S_{t+N_p} - \left(s_t + v_t N_p - a_{\min}(N_p)^2/2 \right) \\
\| B^2 - B^5 \| = l_{up} - l_{t+N_p} \\
\| B^4 - B^5 \| = l_{t+N_p} - l_{down}
\end{cases}
\tag{11}
$$

where N_p is the planning time horizon, $a_{\max}$ is the maximum acceleration (approximately $2 \, \mathrm{m/s^2}$), and $a_{\min}$ is the deceleration (approximately $-3 \, \mathrm{m/s^2}$). For lane-changing actions, the target points will always be on the center of the lane. l_{up} and l_{down} are the positions of the upper and lower lane boundaries, respectively.

For AVs, a macro path is usually predetermined. AVs follow this macro path for local trajectory planning, using Frenet coordinates and polynomial motion predictions similar to Equation (11) to generate five target points. Based on these predicted trajectories, actions leading to road departure or collisions are filtered out, resulting in the feasible action space $\mathbb{A} = \left\{ A^1, A^3, A^4, A^5 \right\}$. These feasible actions are then transformed into the global coordinate system for evaluation to determine the optimal trajectory.

3. Establishment of Evaluation Functions

The previous section predicted the future trajectories of vehicles and constructed the interaction action spaces between the AV and surrounding vehicles. This section will evaluate the benefits of the corresponding actions in the interaction action spaces.

3.1. Design of Action Payoff Functions

When AVs take actions in the interaction space, an immediate payoff R_t is generated. The payoff function is designed to relate to the driving performance DP_t and the adversarial penalty AP_t between the AV and surrounding vehicles:

$$
\begin{cases}
R_t = DP_t + AP_t \\
DP_t = F_t + E_t - C_t
\end{cases}
\tag{12}
$$

Common driving performance metrics DP_t include safety, efficiency, and comfort. Here, F_t represents the safety measure against surrounding vehicles; E_t denotes the efficiency, represented by the speed deviation $|v_t - v_p|$ between the AV's speed v_t and the desired speed v_p; C_t is the cost associated with taking the action.

For the comfort cost C_t, lane-changing actions generally have higher costs, while acceleration and deceleration actions have relatively lower costs. The specific cost can be expressed as the deviation of the action's endpoint from the reference action's endpoint:

$$
C_t = \lambda_1 \cdot \left| x_{\mathrm{d}} - x_{t+N_p} \right| + \lambda_2 \cdot \left| y_{\mathrm{d}} - y_{t+N_p} \right|
\tag{13}
$$

where $(x_{\mathrm{d}}, y_{\mathrm{d}})$ are the trajectory endpoints corresponding to the decision action; $\left(x_{t+N_p}, y_{t+N_p} \right)$ are the predicted trajectory endpoints; λ_1 and λ_2 are the cost coefficients for acceleration/deceleration and lane-changing actions, respectively.

3.2. Risk Assessment Functions

Evaluating risk in traffic scenarios involves multiple vehicle behaviors and varying conditions. To account for position, posture, and speed, this paper introduces the road risk assessment function F^m to evaluate risks between vehicles. Combined with the static road risk function F^s, a comprehensive risk assessment of the AV is performed.

Firstly, risks based on relative positions and speeds can be calculated using the safety distance model. As shown in Figure 4, r_e is the distance from the AV's front to the edge of the vehicle ahead, and D_s is the safe distance between the two vehicles, expressed as:

$$D_{\mathrm{s}} = \frac{v_{\mathrm{e}}^2 - v_{\mathrm{a}}^2}{2|a_{\min}|} + v_{\mathrm{e}}\varsigma + G \tag{14}$$

where v_e is the current speed of the AV, v_a is the speed of the Adv, a_{min} is the deceleration, G is the minimum distance between the two vehicles when stopped, and ς is the response time of the AV. $v_e > v_a$ indicates that the AV requires a longer braking distance to avoid colliding with the vehicle ahead, thus increasing risk. This is because, given the higher speed of the AV, braking must commence sooner. When $v_e \leq v_a$, the AV's speed is lower, and the safe distance between vehicles decreases, hence the risk is lower. The safe distance D_s relative to the separation r_e expresses the risk level between the vehicles.
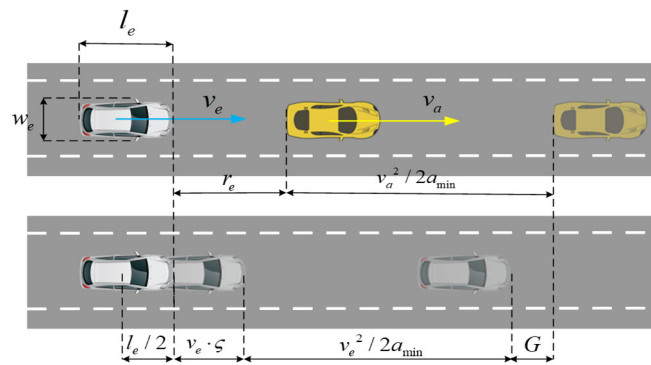

Figure 4. Safety Distance Model for Vehicles in the Same Direction.

When vehicles are in the same direction but not in the same lane, the primary risk comes from lateral displacement. Thus, the risk from surrounding vehicles to the AV is mainly due to the AV's speed v_e and its lateral deviation relative to the adversarial vehicle's speed v_a, expressed as:

$$
\begin{aligned}
D_s &= \frac{v_a^2 - (v_e\cos\Delta\varphi)^2\mathrm{sgn}(\cos\Delta\varphi)}{2|a_{min}|} + v_a\varsigma + G \\
&= k_1\left[v_a^2 - (v_e\cos\Delta\varphi)^2\mathrm{sgn}(\cos\Delta\varphi)\right] + b_1
\end{aligned}
\tag{15}
$$

where $\Delta\varphi$ s the lateral deviation between the AV and the adversarial vehicle, $\Delta\varphi = \varphi_e - \varphi_a$; $\mathrm{sgn}(\cdot)$ is the sign function, and $\mathrm{sgn}(\cos \Delta\varphi)$ determines whether the speeds are aligned. Constants k_1 and b_1 represent the safety factor and distance constant, respectively. Assuming the vehicle speed in urban areas is 16 m/s, with a deceleration $a_{min} = -5$ m/s^2 and a reaction time ς of 0.8 s, and the stopping distance $G = 4$ m, the safety factor can be derived as $k_1 = 0.1$, and the distance constant as $b_1 = 17$.

The risk assessment function F^m for the AV and adversarial vehicle in the same direction is based on their safety distance. When the relative distance r_e exceeds the safe distance D_s, the vehicles are considered safe with zero risk. As the distance decreases, the risk increases, influenced by road conditions, and is expressed as:

$$F^m = \mathrm{poslin}\left(R_a \cdot \frac{D_s}{\|\,\boldsymbol{r}_e\,\|} - 1\right) = \begin{cases} 0 & \|\,\boldsymbol{r}_e\,\| \geqslant R_a \cdot D_s \\ R_a \cdot \frac{D_s}{\|\boldsymbol{r}_e\|} - 1 & \text{other} \end{cases} \tag{16}$$

The function poslin() is a positive linear function, retaining only positive values. R_a is the relative risk factor, which increases with worsening road conditions. The specific values are referenced from [27], as shown in Table 1.

Table 1. Relative Risk Factors R_a under Different Road Conditions.

Road Condition	Probability	Number of Accidents	Relative Risk R_a
Good	0.845	7992	0.9
Rain	0.088	978	1.06
Rain and Snow	0.005	77	1.46
Snow	0.062	1417	2.18

Additionally, when the vehicles are not in the same direction, the distance r_e is extended to include the elliptical safety field, as shown in Equation (17):

$$F^m = \text{poslin}\left(R_a \cdot \frac{k_1\left[v_a^2 - (v_e\cos\Delta\varphi)^2\text{sgn}(\cos\Delta\varphi)\right]\text{sgn}(\cos\theta_a) + b_1}{\| r_e \|} - 1 \right) \tag{17}$$

where θ_a is the angle between the distance r_e and the surrounding vehicle's speed direction; $\text{sgn}(\cos\theta_a)$ determines if the surrounding vehicle is in front of or behind the AV.

For the distance vector r_e, collisions typically occur laterally, making it reasonable to add a relaxation factor ξ to form an elliptical safety field. When the sensor is at the AV's center, the distance vector is expressed as:

$$r_e = [x_a - x_e - l_e/2, \xi \cdot (y_a - y_e - w_e/2)] \tag{18}$$

where ξ is the relaxation factor determining the horizontal spread of the elliptical field. typically set to 2. l_e and w_e are the length and width of the AV, respectively, with the sensor at the vehicle's center.

Additionally, when multiple vehicles interact with the AV, the risk posed by the AV is defined as the maximum of the individual risks:

$$F^m = \max(F_1^m, F_2^m, \cdots, F_k^m) \tag{19}$$

where $F_i^m (i = 1 \sim k)$ represents the risk posed by each surrounding vehicle. The highest risk indicates the most dangerous interacting vehicle.

The risk corresponding to this function can be visualized, as shown in Figure 5, illustrating the collision risk posed by surrounding vehicles to the AV. Similarly, the risk posed by the AV to surrounding vehicles can be calculated using this method.

Risk assessment must consider both the interactions with surrounding vehicles and the static road regions. This includes assessing the road centerline, lane positions, and road boundaries. Safe positions, where the AV can drive without violating road regulations, are denoted by set $\mathcal{B}$. The risk at any road position $g \in \mathcal{B}$ is represented by U_r, given by:

$$U_r(g) = a_r - b_r \sum_{i=1}^{M_{lane}} \exp\left(-(g - x_{center,i})^T \Sigma_r^{-1}(g - x_{center,i})\right) \tag{20}$$

where a_r and b_r are coefficients, $x_{center,i}$ is the lane center position, and Σ_r is the covariance matrix.

When a vehicle deviates from the road $g \notin \mathcal{B}$, collision risk increases significantly. For off-road areas, a high-risk factor is applied. The comprehensive road risk function, integrating on-road and off-road risks, is defined as:

$$F^s = \begin{cases} U_r(g) & g \in \mathcal{B} \\ \dfrac{c_r}{\left[\eta_r(g - x_{edge})\right]^2 + 1} + \tau_r & g \notin \mathcal{B} \end{cases} \tag{21}$$

where c_r is a scaling factor used to adjust the maximum risk cost when the vehicle deviates from the road, η_r adjusts boundary risk, τ_r is the boundary adjustment parameter, and x_{edge} is the road edge coordinate.
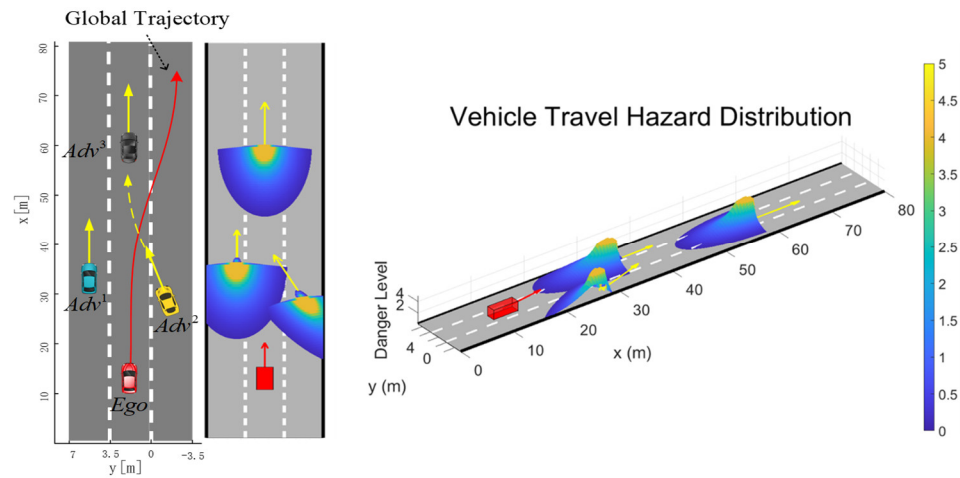
Figure 5. Schematic of the hazard level of the surrounding vehicles, where the red vehicle is the ego vehicle and the others are the adversarial vehicles.

To better visualize road risks, Figure 6 shows the results of the road risk construction. This figure illustrates the risk distribution on a three-lane road. Risk levels are indicated by colors: yellow for high risk, green for medium risk, and blue for low risk.
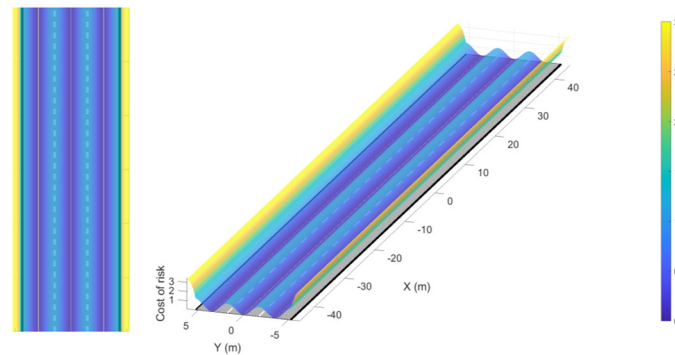
Figure 6. Road risk mapping.

In summary, the overall risk F is derived from the road risk assessment function F^m and the static road risk function F^s:

$$F = \alpha_1 * F^m + \alpha_2 * F^s \tag{22}$$

3.3. Quantitative Model of Adversarial Intensity

In the development of autonomous driving systems, calculating the reachable set is crucial for ensuring safety and efficiency. The reachable set encompasses all possible states a vehicle can achieve within a certain time frame from a given initial state and control inputs. This section employs a fast marching method [28] to calculate the reachable set under various conditions: without obstacles, and considering obstacles and future trajectories of surrounding vehicles. The ratio of the online reachable set to the offline reachable set represents the adversarial intensity faced by the AV in different traffic scenarios.

The mathematical definition of the reachable set is as follows:

$$\operatorname*{reach}_{t}(\mathcal{X}_0, \mathcal{U}, \mathcal{F}(t)) := \{x(t; u(\cdot), x_0) \mid x_0 \in \mathcal{X}_0,$$
$$\forall \tau \in [t_0, t_h] : u(\tau) \in \mathcal{U}, \operatorname{proj}(x(\tau; u(\cdot), x_0)) \notin \mathcal{F}(\tau)\} \tag{23}$$

$$x(t; u(\cdot), x_0) = x_0 + \int_{t_0}^{t} f(x(\tau), u(\tau))d\tau \tag{24}$$

where $x(t; u(\cdot), x_0) \in \mathbb{R}^n$ represents the vehicle state at time t, and $u(\cdot)$ represents the control inputs of the vehicle. According to the kinematic model of adversarial vehicle motion from Section 2.1, $u(t) = (a_t^\tau, a_t^n) \in \mathcal{U}$, where $\mathcal{U}$ is the set of all possible control inputs. $f(x, u)$ is the dynamic model from Equation (4). The state $x(t)$ consists of multiple variables $(s_t, l_t, v_t, \varphi_t)$, $x \in \mathbb{R}^n$, which need to be projected into a two-dimensional space, so we define the projection operator $\text{proj}(x) : \mathbb{R}^n \to \mathbb{R}^2$. From the perspective of the vehicle's safety, $\mathcal{F}(\tau)$ represents road obstacles and forbidden areas. Therefore, Equation (23) defines the reachable set starting from the initial state x_0 over the time interval $t \in [t_0, t_h]$, avoiding a set of forbidden states $\mathcal{F}(t)$.

From Equation (23), we know the reachable set evolves over time and is calculated incrementally. Let $t_k = k \cdot \Delta t$, $k \in \mathbb{N}$ is an integer. Thus, the evolution of the reachable set over time can be expressed as:

$$\mathcal{R}_{k+1} := \underset{t_{k+1}}{\text{reach}}(\mathcal{R}_k, \mathcal{U}, \mathcal{F}(t)) \tag{25}$$

To discretize the reachable set, we partition the position space into a uniform grid of non-overlapping cells aligned with the coordinate axes:

$$\mathcal{C}_k^{(i)} = \left[\underline{c}^{(i)}, \bar{c}^{(i)}\right] \subset \mathbb{R}^2, \bigcup_{i=0}^{n_c} \mathcal{C}_k^{(i)} \supseteq \text{proj}(\mathcal{R}_k) \tag{26}$$

where *i* refers to the *i-th* cell. We construct a directed graph $\mathcal{G} = (V, E)$, where nodes $v_{k,i} \in V$ correspond to cells in the reachable region $\mathcal{C}_k^{(i)}$ at time step k, An edge $\left(v_{k,i}, v_{k+1,j}\right) \in E$ represents the trajectory from one cell to another in the next time step.

The calculation diagram of the offline reachable set and the online reachable set is shown in Figure 7.

(a) Offline Reachable Set

(b) Online Reachable Set
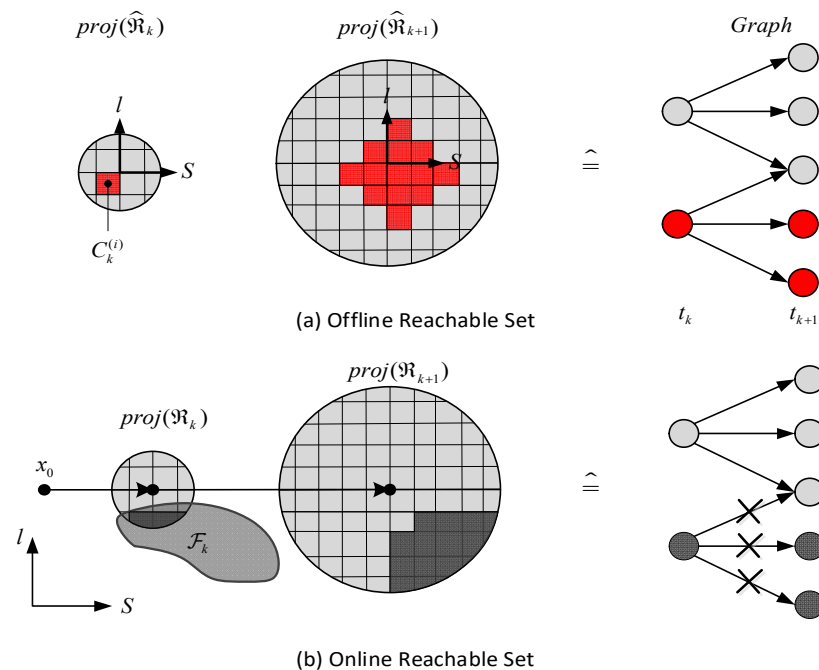
Figure 7. (**a**) Offline Reachable Set: Calculate the reachable cells $\mathcal{C}_{k+1}$ at time t_{k+1} and encode them into the graph. (**b**) Online Reachable Set: Remove nodes that intersect with forbidden sets $\mathcal{F}(t_k)$ or have no predecessor cells (indicated in dark gray).

Calculation of Offline Reachable Sets

Offline reachable sets are calculated without considering obstacles or other vehicles. This represents all possible states a vehicle can theoretically reach without any interference.

Let $\hat{\mathcal{R}}_k$ denote the offline reachable set at time k.

The calculation of offline reachable sets includes propagation, discretization, and graph construction.

Propagation: Based on vehicle dynamics, the longitudinal and lateral accelerations $a_t^\tau \in [\underline{a_t^\tau}, \overline{a_t^\tau}]$ and $a_t^n \in [\underline{a_t^n}, \overline{a_t^n}]$ form the control input set $\mathcal{U}$. $\underline{a}$ and $\overline{a}$ represent the lower and upper bounds of the input control quantity. Thus, we can compute the reachable set $\text{reach}_{t_k}(\mathbf{0}, \mathcal{U}, \varnothing)$ for the initial state at the origin. For further propagation, the reachable set $\hat{\mathcal{R}}_k$ for any initial state x_0 is:

$$\hat{\mathcal{R}}_k = \underset{t_k}{\text{reach}}(x_0, \mathcal{U}, \varnothing) = \underset{t_k}{\text{reach}}(x_0, \mathbf{0}, \varnothing) \oplus \underset{t_k}{\text{reach}}(\mathbf{0}, \mathcal{U}, \varnothing)(\mathcal{R}_k) \tag{27}$$

where the symbol $\oplus$ represents the Minkowski sum. Since the offline reachable set does not exclude any forbidden states, the propagation step is given by: $\hat{\mathcal{R}}_{k+1} = \underset{t_k}{\text{reach}}\left(\hat{\mathcal{R}}_k, \mathcal{U}, \varnothing\right)$, $\mathcal{F}(t) = \varnothing$. Standard reachability analysis tools such as CORA [29], Flow* [30], ARCH-COMP23 [31], and SpaceEx [32] can be used for this computation.

Discretization and Graph Construction: For ease of computation and storage, the continuous reachable set is discretized into a grid or cells, each representing a possible state region. The reachable set $\hat{\mathcal{R}}_k$ is divided into non-overlapping subsets $\hat{\mathcal{R}}_k = \left\{ \mathcal{R}_k^{(1)}, \ldots, \mathcal{R}_k^{(i)} \right\}$. Using Equation (26), we compute the propagation of each cell to obtain:

$$\forall i \in \mathcal{I}: \hat{\mathcal{R}}_{k+1}^{(i)} = \mathcal{C}^{(i)} \times \left[\underline{\dot{s}_k^{(i)}}, \overline{\dot{s}_k^{(i)}} \right] \times \left[\underline{\dot{l}_k^{(i)}}, \overline{\dot{l}_k^{(i)}} \right], \mathcal{I} = \left\{ i \mid \hat{\mathcal{R}}_{k+1} \cap \mathcal{C}^{(i)} \neq \varnothing \right\} \tag{28}$$

where $\left[\underline{\dot{s}_k^{(i)}}, \overline{\dot{s}_k^{(i)}} \right]$ and $\left[\underline{\dot{l}_k^{(i)}}, \overline{\dot{l}_k^{(i)}} \right]$ represent the longitudinal and lateral speed ranges in the Frenet coordinate system, respectively, limiting the reachable set $\hat{\mathcal{R}}_k$ to these boundaries. $\mathcal{R}_k^{(i)}$ is the prerequisite for obtaining a node in graph $\mathcal{G}$.

After obtaining $\mathcal{R}_k^{(i)}$, the reachable set can be represented in a discretized form. We introduce the discretization operator:

$$\hat{\mathfrak{D}}_k = \text{discr}\left(\hat{\mathcal{R}}_k\right) = \bigcup_{i=0}^{n_b} \hat{\mathcal{R}}_k^{(i)} \supseteq \hat{\mathcal{R}}_k \tag{29}$$

Similarly, $\mathfrak{D}_k = \text{discr}(\mathcal{R}_k)$, and the condition for adding edges $\left(v_{k,i}, v_{k+1,j} \right)$ in graph $\mathcal{G}$ is:

$$\underset{t_{k+1}}{\text{reach}}\left(\mathcal{R}_k^{(i)}, \mathcal{U}, \varnothing \right) \cap \mathcal{R}_{k+1}^{(j)} \neq \varnothing. \tag{30}$$

To efficiently implement this process, we use an adjacency matrix $P_k^{k+1} \in \mathbb{R}^{q_k \times q_{k+1}}$ to represent edges, where q_k is the number of grid cells occupied at time k. Each element p_{ji} of P_k^{k+1} is binary, defined as:

$$p_{ji} = \begin{cases} 1 & \text{if } \text{reach}_{t_{k+1}}\left(\mathcal{R}_k^{(i)}, \mathcal{U}, \varnothing \right) \cap \mathcal{R}_{k+1}^{(j)} \neq \varnothing \\ 0 & \text{otherwise} \end{cases} \tag{31}$$

The adjacency matrix P_k^{k+1} is the main result of the reachability calculation and can be efficiently stored as a sparse matrix. It represents the edges in the directed graph $\mathcal{G}$, indicating the reachable path from one cell to the next over time.

The overall algorithm process is shown in Algorithm 1.

Algorithm 1 Offline Reachability Analysis

Require: input set $\mathcal{U}$

1: **for** k = 0 to h **do**

2: $\quad \hat{\mathcal{R}}_{k+1}^0 \leftarrow \underset{t_{k+1}}{\text{reach}} \left(\hat{\mathcal{R}}_k^0, \mathcal{U}, \varnothing \right)$

3: $\quad \hat{\mathfrak{D}}_{k+1} \leftarrow \text{discr} \left(\hat{\mathcal{R}}_{k+1}^0 \right)$ $\qquad\qquad\qquad\qquad \triangleright$ see (29)

4: $\quad \left(v_{k,i}, v_{k+1,j} \right) \leftarrow \text{REACHABLECELLS} \left(\hat{\mathfrak{D}}_k, \hat{\mathfrak{D}}_{k+1} \right)$ $\quad \triangleright$ see (30)

5: $\quad P_k^{k+1} \leftarrow \text{CONSTRUCTGRAPH} \left\{ \left(\mathcal{R}_k^{(i)}, \mathcal{R}_{k+1}^{(i)} \right) \right\}$ $\quad \triangleright$ see (31)

6: **end for**

7: **return** P_k^{k+1}, $\hat{\mathfrak{D}}_k$

Calculation of Online Reachable Sets

The goal of online calculation is to exclude obstacles and other vehicles' predicted trajectories from the reachable region $\mathcal{F}(k)$. Therefore, during each propagation step, cells that intersect with $\mathcal{F}(k)$ are excluded from the reachable set.

We use the adjacency matrix P_k^{k+1} to propagate the drivable region and introduce a binary variable $r_k \in \{0,1\}^q$ to indicate whether each cell belongs to the drivable region:

$$r_k^{(i)} = \begin{cases} 1 & \text{if } \left(x(t_k; 0, x_0) \oplus \mathcal{C}_k^{(i)} \right) \cap \text{proj}(\mathfrak{D}_k) \neq \varnothing \\ 0 & \text{otherwise.} \end{cases} \tag{32}$$

Combining with Equation (31), we can write the propagation as:

$$\widehat{r}_{k+1} = P_k^{k+1} r_k \tag{33}$$

To exclude $\mathcal{F}(k+1)$ from the binary representation $\widehat{r}_{k+1}$, we need to discretize $\mathcal{F}(k+1)$. In motion planning, the vehicle's shape is often approximated by a bounding semi-disk ρ to represent obstacle occupancy, allowing for efficient collision checking. Similar to r_k, we introduce the binary variable o_{k+1} to indicate occupancy:

$$o_{k+1}^{(i)} = \begin{cases} 1 & \text{if } x(t_k; 0, x_0) \oplus \mathcal{C}_{k+1}^{(i)} \subseteq \mathcal{F}_{k+1} \\ 0 & \text{otherwise.} \end{cases} \tag{34}$$

where $\mathcal{F}_{k+1}$ is typically non-contiguous, so multiple obstacles need to be discretized. For efficiency, only obstacles intersecting the drivable region's boundary are discretized. Combining Equations (32) and (34), we exclude the occupancy states:

$$r_{k+1} = \widehat{r}_{k+1} \wedge \neg o_{k+1} \tag{35}$$

Logical operations $\neg$ and $\wedge$ are executed element-wise. Thus, we can express the propagation and exclusion of the reachable set as:

$$\mathfrak{D}_{k+1} = \text{discr} \left(\underset{t_{k+1}}{\text{reach}} (\mathfrak{D}_k, \mathcal{U}, \varnothing) \right) \smallsetminus \text{discr}(\mathcal{F}(t_{k+1})),$$
$$\mathfrak{D}_0 = x_0 \tag{36}$$

Combining Equations (33) and (35), the simplified exclusion occupancy state is:

$$r_{k+1} = P_k^{k+1} r_k \wedge \neg o_{k+1} \qquad (37)$$

The overall algorithm process is shown in Algorithm 2.

Algorithm 2 Online Reachability Analysis

Require: graph represented by propagation matrices P_k^{k+1}, forbidden set $\mathcal{F}(k)$, number of time steps h, Discrete sets of offline reachable sets $\mathfrak{D}_k$

1: **for** k = 0 to h **do**
2:      $x(t_k; 0, x_0) \leftarrow$ HOMOGENEOUSSOLUTION(x_0)
3:      $\hat{r}_{k+1} \leftarrow$ IFINAREA$\left(P_k^{k+1}, \mathfrak{D}_k, \mathcal{C}_k, x(t_k; 0, x_0)\right)$ ▷ see (33)
3:      $o_{k+1} \leftarrow$ OCCUPANCYGRI$(\mathcal{F}_{k+1}, \mathcal{C}_{k+1}, x(t_k; 0, x_0))$ ▷ see (34)
4:      $r_{k+1} \leftarrow$ EXCULDEICS $(\hat{r}_{k+1}, o_{k+1})$ ▷ see (35)
5:      $\mathfrak{D}_{k+1} \leftarrow$ DISCR $\{(\mathfrak{D}_k, r_{k+1}, \mathcal{F}(t_{k+1}))\}$ ▷ see (36)
6: **end for**
7: **return** r_{k+1}, $\mathfrak{D}_{k+1}$

Evaluation of Adversarial Intensity

By comparing the online reachable set $\hat{\mathfrak{D}}_k$ and the offline reachable set $\mathfrak{D}_k$, we can calculate the adversarial intensity index AP_k. The ratio of the online to offline reachable sets indicates the adversarial pressure on the AV, as shown in Figure 8. The expected adversarial intensity λ_{AP} is set to 0.2, 0.4, and 0.6 for low, medium, and high intensities. AP_k quantifies the deviation from the expected intensity:

$$AP_k = \left| \lambda_{AP} - \omega(k) * \frac{|\mathfrak{D}_k|}{\left|\hat{\mathfrak{D}}_k\right|} \right| \qquad (38)$$

where $|\mathfrak{D}_k|$ and $\left|\hat{\mathfrak{D}}_k\right|$ are the sizes of the online and offline reachable sets, respectively, λ_{AP} is the expected intensity, and $\omega(k)$ is a dynamic weighting factor.
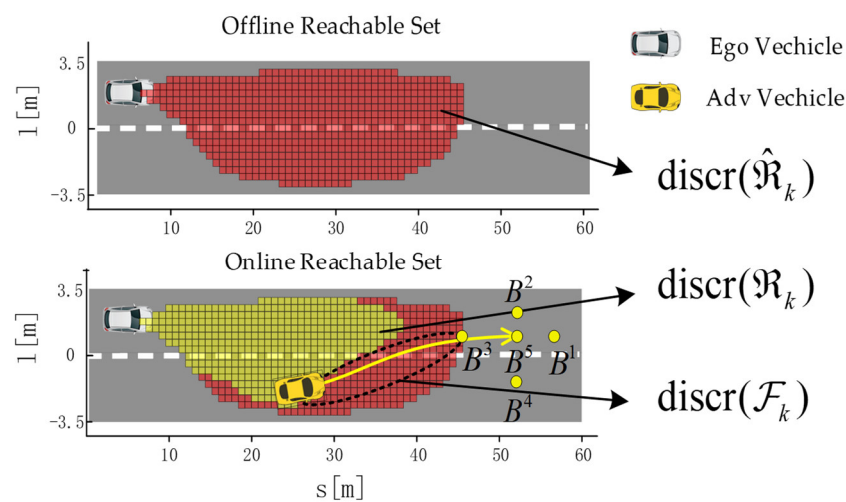
Figure 8. Visualization of online reachable set and offline reachable set calculation.

In summary, the algorithm flow of action payoff function is solved as follows Algorithm 3:

Algorithm 3 Calculate Immediate payoffs

Input: Current state x(t) includes ego- and adv-vehicles, control input u($\cdot$), expected speed v_p, parameters $\alpha_1, \alpha_2, \lambda_{AP}, \lambda_1, \lambda_2$

Output: Immediate payoff R_t

  1: Initialize payoff $R_t = 0$
  2: Calculates afety performance F_t
     2.1 Calculate dynamic danger F^m based on relative distance and speed
        $\triangleright$ see (17)
     2.2 Calculates taticr oadr iskF^s
        $\triangleright$ see (21)
     2.3 Combine to get overall danger $F = \alpha_1 * F^m + \alpha_2 * F^s$
  3: Calculate efficiency performance E_t
$$E_t = \left| v_t - v_p \right|$$
  4: Calculate comfort cost C_t
$$C_t = \lambda_1 \cdot \left| x_d - x_{t+Np} \right| + \lambda_2 \cdot \left| y_d - y_{t+Np} \right|$$
  5: Calculate vehicle performance DP_t
$$DP_t = F_t + E_t - C_t$$
  6: Calculate adversarial intensity AP_t
$$AP_k = \left| \lambda_{AP} - \omega(k) * \frac{\left| \mathfrak{D}_k \right|}{\left| \dot{\mathfrak{D}}_k \right|} \right|$$
    $\triangleright$ see (Algorithm 1 and Algorithm 2)
  7: Calculate immediate payoff R_t
$$R_t = DP_t + AP_t$$
  8: **return** R_t

---

4. Solution of the Adversarial Interaction Decision-Making Model

Based on Sections 2 and 3, we can determine the interaction space $(\mathbb{A}, \mathbb{B})$ and the corresponding action payoff R_t. This section further discusses the game-theoretic process and derives the equilibrium actions using the Stackelberg game model.

4.1. Determining Leader and Follower Vehicles

In normal traffic scenarios, to ensure safety and efficiency, vehicles usually yield to the vehicle ahead. Here, the leading vehicle and the following vehicle establish a leader–follower relationship. The leading vehicle sets the base for interaction, maximizing its driving advantages. The process can be described using the Stackelberg game model.

Specifically, we need to identify the leading and following vehicles. Typically, we judge this based on their positions in the lane. For instance, in a scenario where both vehicles maintain their lane, the one ahead is the leader, and the one behind is the follower. This can be determined by the relative distance between the AV and surrounding vehicles, as shown in Figure 9.
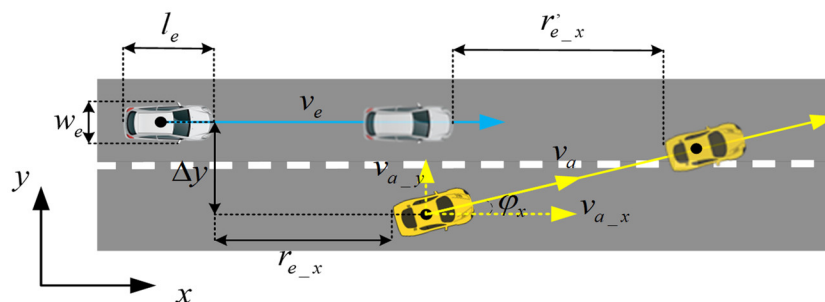
Figure 9. Process for Determining Leader and Follower Vehicles.

First, the time T_{TF} for the surrounding vehicle to reach the front of the AV is given by:

$$T_{TF} = \frac{\Delta y - w_e/2}{v_a sin\varphi_x} \tag{39}$$

where w_e is the width of the AV, accounting for the relative distance between the AV and the surrounding vehicle on both sides. Based on this time, the relative distance in the x-direction is given by:

$$r'_{e\_x} = \boldsymbol{r}_{e\_x} + (v_a cos\varphi_x - v_e) \cdot T_{TF} \tag{40}$$

where $\boldsymbol{r}_{e\_x}$ is the relative distance component in the x-direction.

Thus, the leader–follower status of the AV can be determined based on the relative distance in the x-direction:

$$\text{EgoVehicle} = \begin{cases} \text{master} & r'_{e\_x} \geq 0 \\ \text{slave} & r'_{e\_x} < 0 \end{cases} \tag{41}$$

4.2. Solving for Nash Equilibrium

The evaluation function established in Section 3 can assess the driving ability of the AV and its adversarial intensity with surrounding vehicles at a specific time. However, the specific solution requires a sequence of actions (A^1, B^1) to be evaluated comprehensively over time, rather than at a single moment. According to the Markov decision process, the state–action value function $Q(\boldsymbol{s}_t, A^1, B^1)$ can be derived by the cumulative discounted payoff, defined as:

$$Q\left(\boldsymbol{s}_t, A^1, B^1\right) = \sum_{i=0}^{n} \gamma^i R_{t+i+1} \tag{42}$$

where γ is the discount factor, and R_{t+i+1} is the immediate payoff at time $t + i + 1$. In this paper, $n = 5$, meaning the value function is computed over 5 discrete states for the action sequence.

The AV and Adv adopt actions A^5 and B^2, respectively, indicating the state–action sequences with a medium expected adversarial intensity. Using Equation (12) to solve the immediate payoff function at each time step, we derive the value function map, showing collision risk and adversarial intensity.

From Figure 10, it is clear that the Adv's lane-changing behavior increases the risk for the AV. Thus, the AV must take proactive actions (e.g., deceleration) to avoid risks. Similarly, the Adv uses its state–action value function to decide actions like deceleration or lane change based on the payoff function.
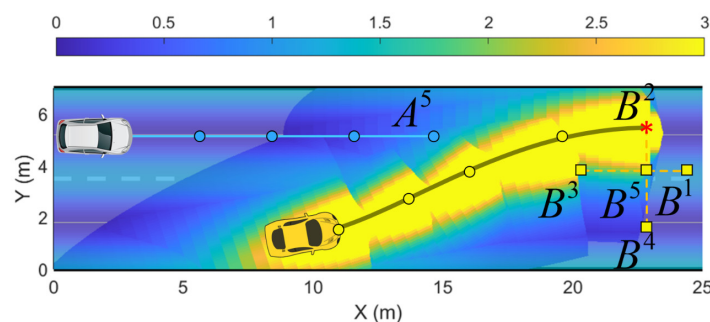
Figure 10. Illustration of Solving the State–Action Value Function.

First, we need to solve the response function of the follower vehicle to the leader vehicle. In the scenario shown in Figure 10, the AV is the follower, and Adv is the leader. For the leader vehicle, a set of acceleration and turning actions B^i is given, and the follower

vehicle will derive the optimal response action $\widehat{A}$ based on this set. The specific action is defined as:

$$\widehat{A} = r(B^i)$$
$$r(B^i) \triangleq \left\{ \zeta \in \mathbb{A} : Q_t^1(s_t, \zeta, B^i) \leqslant Q_t^1(s_t, A^j, B^i), \forall A_j \in \mathbb{A} \right\} \tag{43}$$

where r is the optimal response function of the follower vehicle, and Q_t^1 is the state–action value function of the follower. The smaller Q_t^1 is, the better the action A.

To determine the leader's optimal action, it selects the action that minimizes its own state–action value function Q_t^2:

$$B^* = \operatorname{argmin}\left[\max\left(Q_t^2\left(s_t, A^j, \zeta\right)\right), A^j \in r\left(B^i\right), \zeta \in \mathbb{B}\right] \tag{44}$$

where B^* is the leader's optimal action from its action space. Using Equation (44), the follower's optimal response to B^* can be found. This solution $\left(A^*, B^*\right)$ represents the Stackelberg equilibrium.

For the scenario shown in Figure 10, the state–action value function matrix for both vehicles can be derived, as illustrated in Table 2. The optimal actions for both the AV and the Adv can be identified from this matrix, representing the equilibrium. The same process can be applied to more vehicles by considering the most challenging and dangerous scenarios for evaluation.

Table 2. State–Action Values for Vehicle Interactions.

| **State–Behavior Values**		Movement of the Vehicle in Front (Leader)				
		B^1	B^2	B^3	B^4	B^5
Movement of the vehicle behind (follower)	A^1	8.36, 10.06	12.46, 17.36	12.31, 18.50	10.57, 19.17	11.50, 16.92
	A^3	5.03, 5.07	6.12, 5.26	4.58, 4.66	8.03, 8.17	6.05, 6.17
	A^4	6.59, 5.68	9.45, 9.46	12.00, 6.61	13.19, 8.35	9.42, 7.67
	A^5	5.07, 7.31	3.43, 5.46	6.15, 15.11	6.16, 11.73	4.22, 10.15

5. Experiments

To verify the proposed game-theoretic adversarial interaction-based critical scenario generation method for AVs, this section includes three main experiments.

First, we designed a single adversarial vehicle lane-change scenario to demonstrate the rationality, human-like decision-making, and interpretability of our algorithm. Second, we designed two scenarios to show that our algorithm enables adversarial vehicles to exhibit different levels of adversarial behavior. Third, in the Carla simulator, we tested two autonomous driving systems: an RRT (Rapidly exploring Random Tree)-based path-planning system [33] and a DDPG (Deep Deterministic Policy Gradient)-based system [34], using three levels of adversarial strategies. This demonstrated our method's efficiency in building critical scenarios and improving testing efficiency.

5.1. Experiment to Demonstrate Adversarial Interaction Capability

To test the adversarial interaction capability of the proposed algorithm in multi-vehicle traffic scenarios, we constructed a test scenario as shown in Figure 11. The yellow vehicle is the Adv, while the other vehicles follow human driving sequences extracted from the NGSIM [35] database. The initial speeds' and positions' parameters are shown in Table 3. The decision parameters for the Adv are listed in Table 4. To verify the Adv's adaptability to different driving conditions, we designed two specific driving styles for the black vehicle.
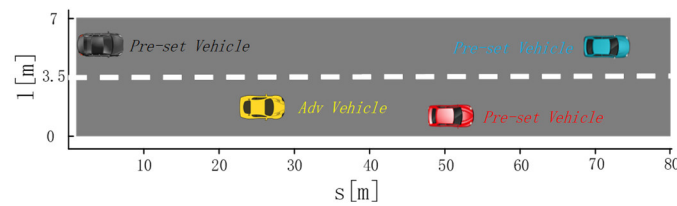
Figure 11. Initial experimental scenarios for adversarial interactivity validation.

Table 3. Initial setting of parameters.

Vehicle ID	Initial Position (m)	Initial Speed
Adv vehicle (yellow)	(27, −1.75)	13 m/s
Ego vehicle (white)	(2, 1.75)	10 m/s
Background vehicle (red)	(51, −2)	9 m/s
Background vehicle (blue)	(74, 1.75)	13 m/s

Table 4. Decision-making parameterization.

Parameters	Value
Prediction Period T/s	0.1
Prediction Time Horizon N_p/s	2
Acceleration for normal driving $a_{max}/(\text{m/s}^2)$	2
Deceleration for normal driving $a_{min}/(\text{m/s}^2)$	−3
Extra transverse angular velocity $\Delta w/(\text{rad/s})$	0.15
Value Function Discount Factor γ	0.98
Acceleration/deceleration/lane change corresponding cost factor λ_1/λ_2	0.02/0.04
Dynamic/Road risk assessment factor α_1/α_2	0.8/0.2
Expected speed $v_p/(\text{m/s})$	13
Road width W/m	3.5
Safety constant k_1	0.17
Spacing constant b_1	10
Cell size $dx_s/dx_l(m)$	0.5
Radius ρ	1.25
Level of adversarial desired λ_{AP}	0.6

Interaction in a cautious driving scenario: In this scenario, the black vehicle drives cautiously and remains in its current lane. Due to the lower speed of the red vehicle and the need to occupy the reachable set of the black vehicle, the Adv seeks opportunities to change lanes. While performing adversarial behaviors, the Adv also interacts with surrounding vehicles to ensure safety and efficiency. The results are shown in Figure 12, including the driving paths and speed profiles of the vehicles. The trends of risk and corresponding game objects are depicted in Figure 12c. The game process at three specific moments is illustrated in Figure 12d, which describes the predicted trajectories of the game vehicles, the action space of the Adv, and the selected optimal actions (indicated by "*").

The adversarial decision-making process consists of three stages: Deceleration and following stage; Lane-changing and acceleration stage; Completing lane change and maintaining lane stage. In the first stage (t = 0~1.8 s), the adversarial vehicle (yellow) travels at high speed. Due to the slower speed of the vehicle ahead and the high risk of direct lane change, it chooses to decelerate and follow. In the second stage (t = 1.8~4.5 s), the yellow vehicle starts to change lanes, and the game vehicle switches to the black vehicle. As shown in the speed profile in Figure 12b, the black vehicle's speed increases. At this point, decelerating the yellow vehicle would increase the risk of collision. Therefore, the decision outcome for the yellow vehicle is as shown in Figure 12d. In the third stage (t > 4.5 s), after the yellow vehicle completes the lane change, it stabilizes and maintains its current lane.
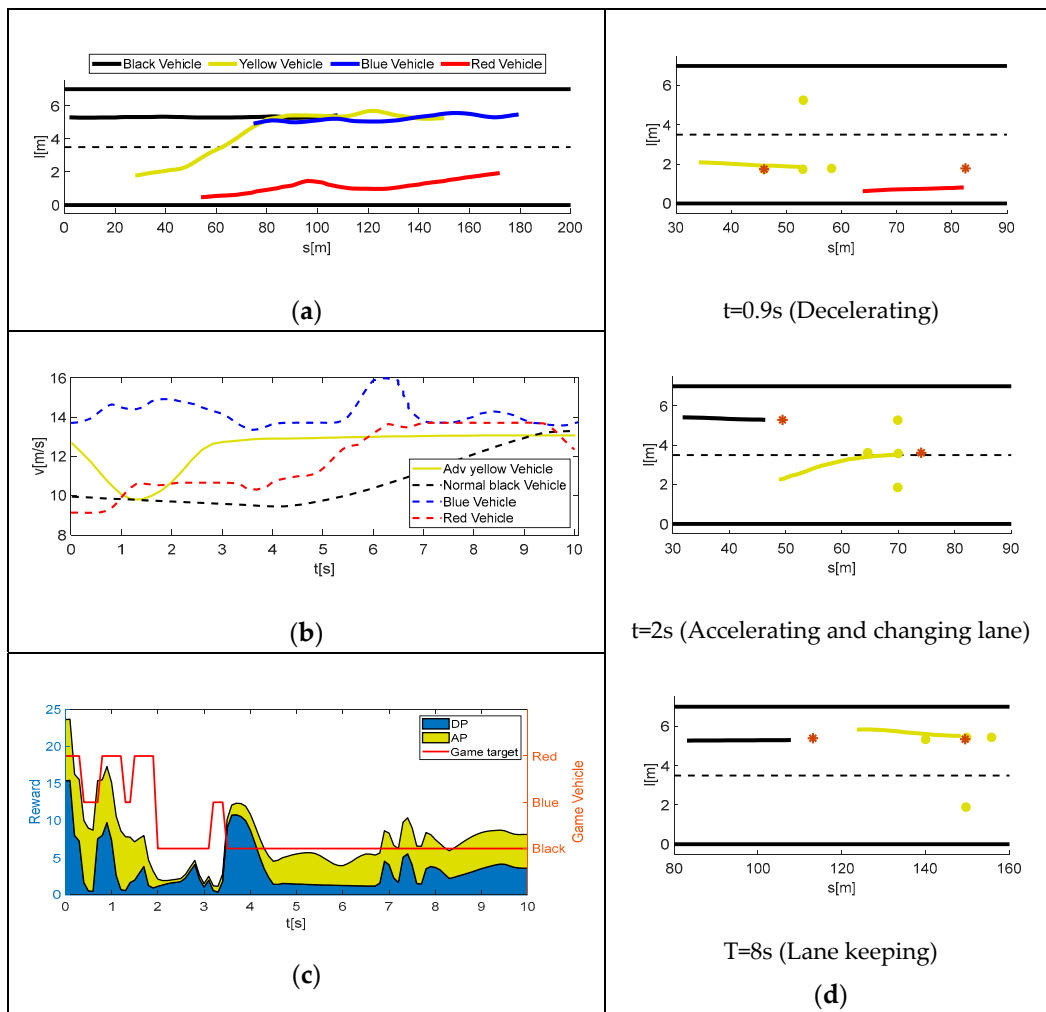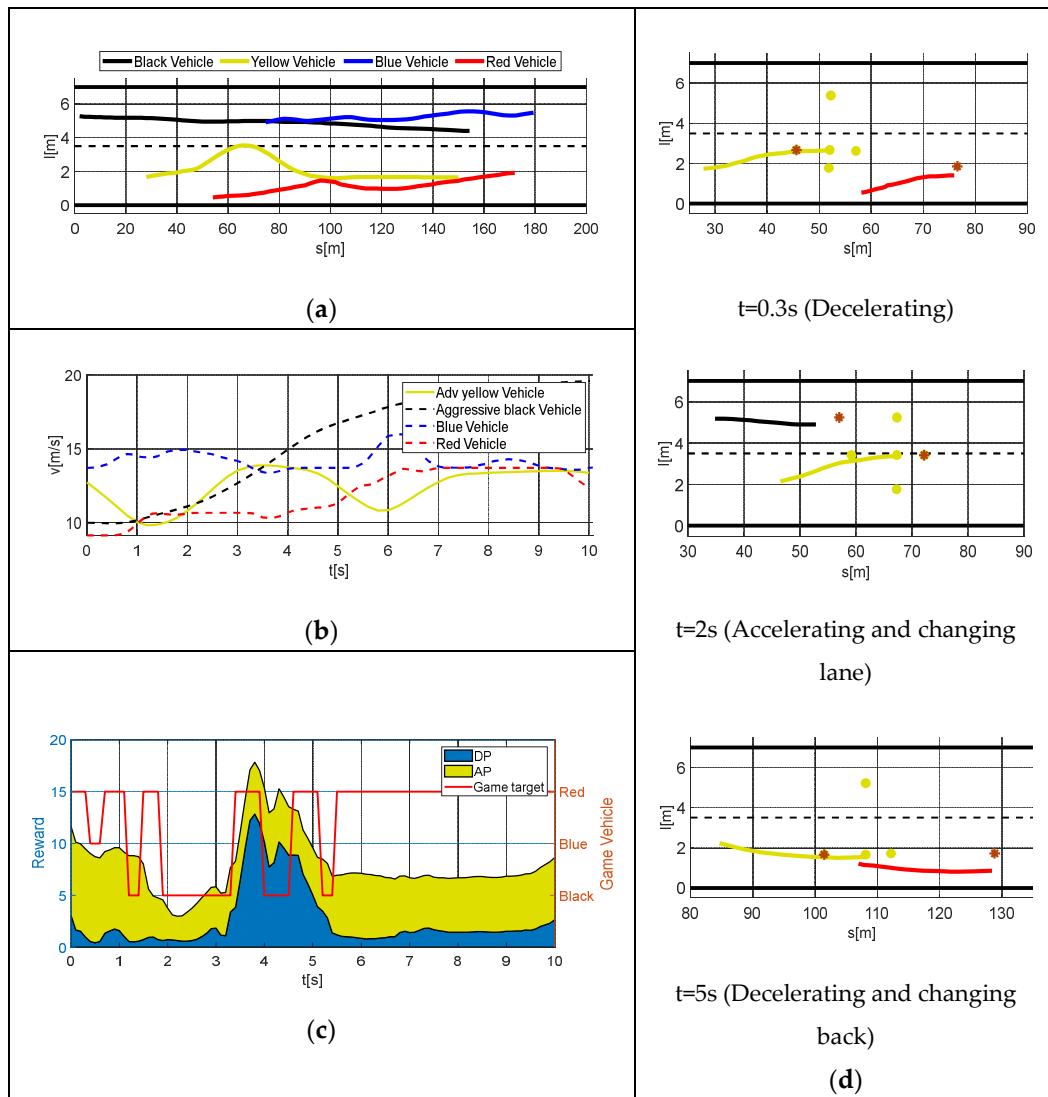
Figure 12. Interaction in a cautious driving scenario. (**a**) Path of this process. (**b**) Speed profile. (**c**) Performance of vehicles and adversarial intensity between vehicles at different times, including game vehicles (black, blue, red). (**d**) Basic actions of the ego vehicle and the interactive vehicle.

Interaction in an Aggressive Driving Scenario: In this scenario, the black vehicle drives aggressively, making it difficult for the yellow vehicle to change lanes. The black vehicle's continuous acceleration complicates adversarial interactions, highlighting the importance of game theory. The results are shown in Figure 13.

The decision-making process includes three stages: Deceleration and following (0~1.8 s): Similar to the previous scenario; Attempting and abandoning lane change (1.8~4.2 s): The black vehicle's continuous acceleration increases the yellow vehicle's risk, leading it to revert to its original lane; Stable following (after 4.5 s): Once the black vehicle surpasses the yellow vehicle, the yellow vehicle loses the ability to invade the black vehicle's reachable set and switches to a stable following strategy.

In Scenario 1, the Adv changes lanes at an appropriate time and speed, ensuring safety and reasonable interaction while occupying the black vehicle's reachable set. In Scenario 2, the Adv compromises, abandoning the adversarial action. In summary, the proposed algorithm for the adversarial vehicle demonstrates rationality, interpretability, and similarity to human driving behavior in adversarial interactions.

Figure 13. Interaction in an Aggressive Driving Scenario. (**a**) Path of the process. (**b**) Speed profile. (**c**) Driving performance and adversarial intensity between vehicles (black, blue, and red) at different times. (**d**) Basic functions of the ego vehicle and interacting vehicles.

5.2. Multi-Level Adversarial Capability Verification

This section verifies the algorithm's ability to handle different levels of adversarial behavior by designing two test scenarios. As shown in Figure 14, Scenario 1 is a high-speed dual-lane scenario, while Scenario 2 is a more complex high-speed three-lane scenario. Increasing complexity tests the algorithm's adaptability.

In Scenario 1, the black vehicle uses an RRT-based path-planning system for decision-making, while the yellow vehicle uses our proposed adversarial planning strategy. The red vehicle follows a pre-set trajectory. We set three adversarial levels for the yellow vehicle: low, medium, and high. We then analyzed the interaction between these vehicles to test the adaptability of the driving behavior under different adversarial conditions. Figure 15 illustrates the interactions for these levels.
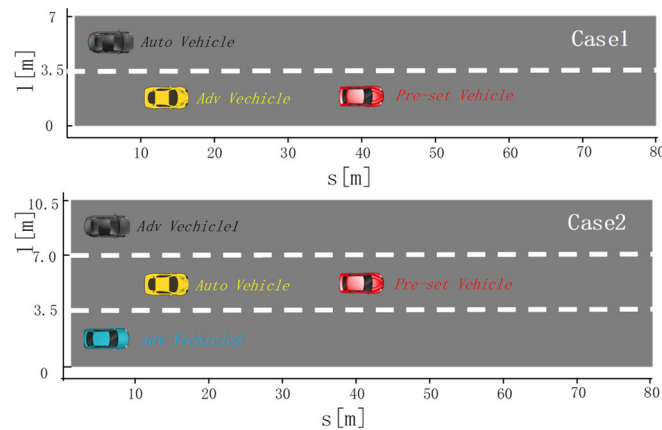
Figure 14. Test Scenarios for Evaluating Adversarial Capability Levels.
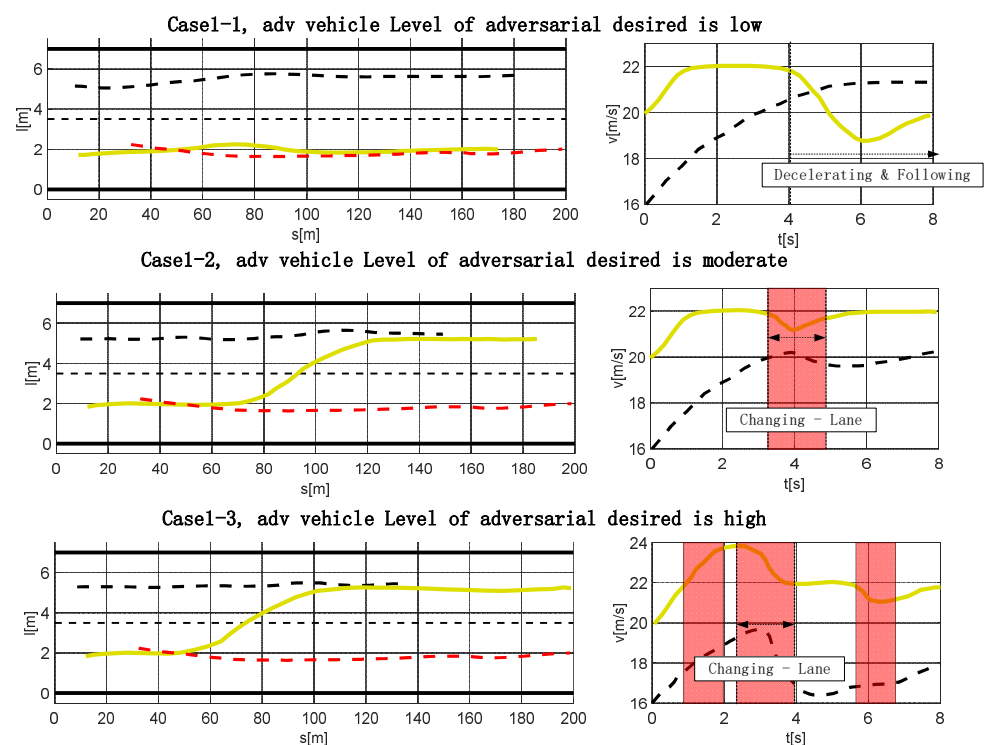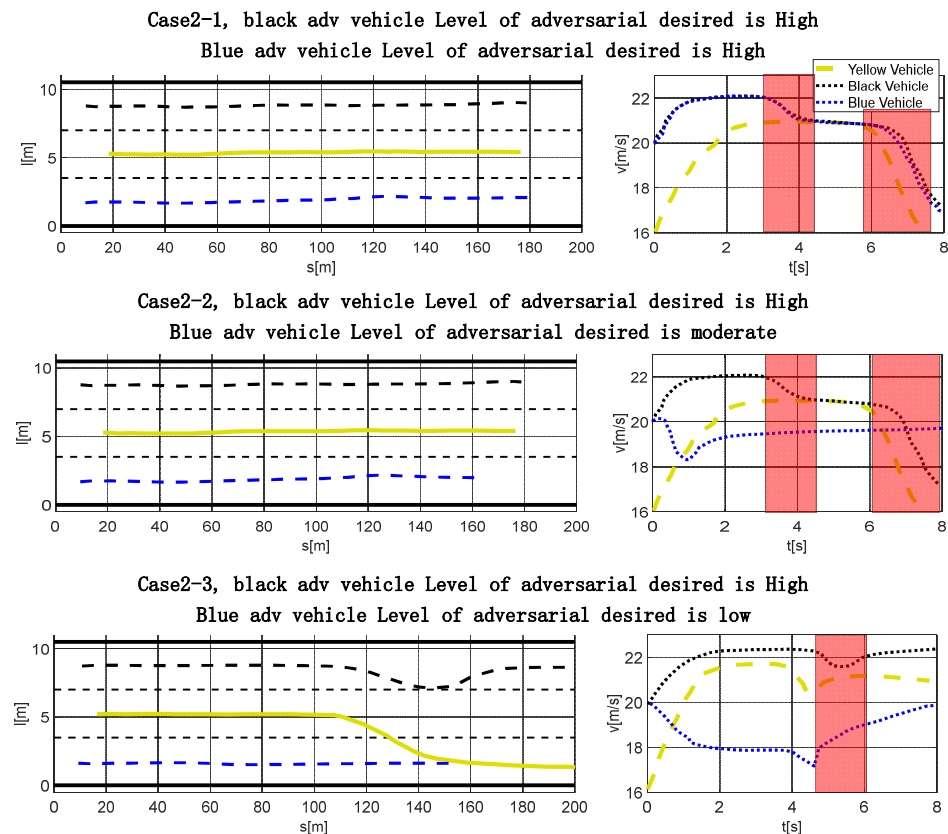

Figure 15. The left side shows the planning results for different adversarial levels, and the right side shows the speed profiles of the black vehicle (autonomous) and the yellow vehicle (adversarial). Red areas in the speed profiles indicate adversarial behavior.

Low Adversarial Level ($\lambda_{AP} = 0.6$): In a dual-lane scenario, the yellow and red vehicles already occupy part of the black vehicle's reachable set, reducing the proportion of the black vehicle's online reachable set to about 0.6. Consequently, the black vehicle does not perform a lane change, instead maintaining its desired speed and then decelerating to follow the red vehicle once it matches its speed. **Medium Adversarial Level** ($\lambda_{AP} = 0.4$): The yellow vehicle begins to ensure its safety and efficiency by attempting to occupy the black vehicle's reachable area. At 3.2 s, it decides to change lanes, prompting the black vehicle to decelerate in response to the yellow vehicle's lane change. **High Adversarial Level** ($\lambda_{AP} = 0.2$): The yellow vehicle adopts an aggressive driving style, performing three adversarial maneuvers as shown in the red areas. These occur at 1 s (accelerating to prevent the black vehicle from occupying its lane), 2.4 s (changing lanes earlier and faster than at the medium level), and 5.8 s (decelerating, forcing the black vehicle to decelerate as well).

In Scenario 2, the yellow vehicle uses an RRT-based path-planning system for decision-making, while the black and blue vehicles use our proposed adversarial planning strategy. The red vehicle follows a pre-set fixed trajectory. We designed three sets of experiments with varying adversarial levels for the black and blue vehicles: high–high, high–medium, and high–low. Figure 16 illustrates the interactions for these levels.


Figure 16. The left side shows the planning results for different adversarial levels, and the right side shows the speed profiles of the yellow vehicle (autonomous), the black vehicle (adversarial), and the blue vehicle (adversarial). Red areas in the speed profiles indicate adversarial behavior.

In Scenario 2-1, both the black and blue vehicles maintain a high adversarial level. The yellow vehicle stays in the middle lane and decelerates to follow the red vehicle when it gets too close due to the red vehicle's low speed. The black and blue vehicles first accelerate to their desired speeds and then execute two deceleration maneuvers to maintain a small longitudinal distance from the yellow vehicle, occupying the space needed for the yellow vehicle to change lanes. In Scenario 2-2, the yellow vehicle remains in the middle lane throughout. The black vehicle behaves similarly to Scenario 2-1, while the blue vehicle adopts a more moderate driving style with reduced speed. The longitudinal distance between the blue and yellow vehicles remains moderate, with the blue vehicle exerting less pressure on the yellow vehicle compared to the black vehicle. In Scenario 2-3, the blue vehicle's adversarial level is lower, leading to more noticeable deceleration and providing enough longitudinal space for the yellow vehicle to execute a right lane change. At 4.8 s, the black vehicle attempts a right lane change to encroach on the yellow vehicle's space but returns to its original lane due to the high risk of a forced lane change near the red vehicle.

5.3. Validation of Key Scenario Construction

To verify the effectiveness of the game-theoretic adversarial interaction-based planning control algorithm in improving testing efficiency and constructing critical scenarios, we utilized initial states and trajectories from the NGSIM natural driving dataset. We tested two

intelligent driving systems: an RRT-based path-planning autonomous driving system [33] and a DDPG-trained autonomous driving system [34]. The tests were conducted on the open-source Carla simulator platform with ROS for communication. The overall testing framework is shown in Figure 17.
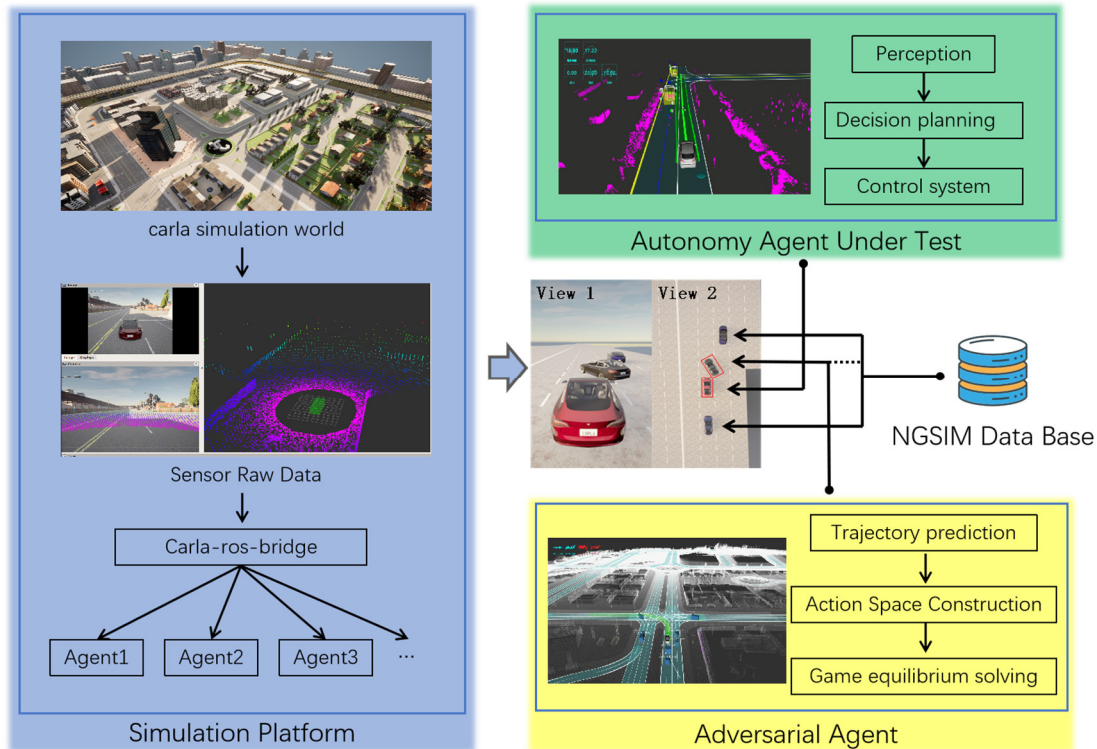
Figure 17. Test Framework Based on Carla (0.9.13) Software-in-the-Loop.

We designed multiple driving states for the adversarial vehicle, including fixed natural driving behavior from the NGSIM database and three adversarial levels (low, medium, high) using the game-theoretic planning control algorithm. Considering the complexity of black-box autonomous systems, we selected 20 initial states that can induce lane changes and tested each scenario 200 times. Metrics recorded include collision rate, changes in lateral and longitudinal acceleration, average risk factor, and time to reach the endpoint.

Table 5 shows that the collision rate for the proposed game-theoretic adversarial interaction algorithm increased significantly compared to natural driving scenarios. As the adversarial level increased, the frequency of collisions rose. At $\lambda_{AP} = 0.6$, the collision rate for the RRT-based system increased by 158%, while the DDPG-based system's rate increased by 385%. At $\lambda_{AP} = 0.4$, the collision rate for the RRT-based system increased by 347%, while the DDPG-based system's rate increased by 794%. At $\lambda_{AP} = 0.2$, the collision rate for the RRT-based system increased by 594%, while the DDPG-based system's rate increased by 1313%. Testing efficiency significantly improved.

Table 5. Collision Rates of Different AVs Under Various Adversarial Levels.

	Adversarial Vehicle Level of Adversarial Desired			
	None $AP_t = 0$	**Low** $\lambda_{AP} = 0.6$	**Moderate** $\lambda_{AP} = 0.4$	**High** $\lambda_{AP} = 0.2$
NGSIM [35]	0.81%	0.72%	1.09%	1.94%
RRT [33]	1.68%	2.67 (0.99)%↑	5.83 (4.15)%↑	9.98 (8.30)%↑
DDPG [34]	1.28%	4.94 (3.66)%↑	10.17 (8.89)%↑	16.81 (15.53)%↑

The arrows ↑ indicate that the data is incremental.

When the test indicators of the AV under test (comfort, risk factor, traffic efficiency, collision rate) remain at a relatively high level, we consider it to be in a critical test scenario. Table 6 provides further details on the impact on the tested objects. With higher adversarial levels, the comfort, risk, and average time to reach the destination for the tested intelligent driving vehicles all showed an increase. Overall, the RRT-based system performed better than the DDPG system, highlighting the proposed game-theoretic adversarial interaction algorithm's ability to produce more adversarial scenarios and generate a large number of critical scenarios. These critical scenarios are especially valuable due to the lack of scenarios encountered by the DDPG system during training.

Table 6. Impact of Different Adversarial Levels on AV States.

| | AV under Test | | | | | |
| | DDPG | | | RRT | | |
	a^+(m/s^3)	$agv(DP_t)$	$agv(T_{finish})$	a^+(m/s^3)	$agv(DP_t)$	$agv(T_{finish})$
$AP_t = 0$	1.22	4.48	10.54	0.94	4.44	10.08
$\lambda_{AP} = 0.6$	2.17 (0.95) ↑	7.29 (2.81) ↑	10.76 (0.22) ↑	0.89 (−0.05) ↓	6.82 (2.38) ↑	10.49 (0.41) ↑
$\lambda_{AP} = 0.4$	3.05 (1.83) ↑	13.81 (9.33) ↑	11.37 (0.83) ↑	1.27 (0.33) ↑	9.31 (4.87) ↑	10.88 (0.8) ↑
$\lambda_{AP} = 0.2$	3.65 (2.43) ↑	15.86 (11.38) ↑	11.93 (1.39) ↑	2.34 (1.4) ↑	14.57 (10.13) ↑	11.34 (1.26) ↑

The arrows ↑ indicate that the data is incremental, ↓ indicate that the data is decreased.

6. Conclusions

This study explores the decision-making and planning models for autonomous vehicles based on game-theoretic adversarial interactions through a series of experiments and validations. The behavior characteristics of vehicles during interactive games were examined, and the effects of different driving strategies on adversarial interactions were discussed. A method to generate adversarial interaction behaviors using the Stackelberg game model was proposed, and its application in various traffic scenarios was investigated through numerical experiments. The main conclusions are as follows:

Effectiveness of the Model: Our game-theoretic decision algorithm effectively simulates human driving behavior. It shows robust adversarial interaction capabilities across different driving strategies, providing rational and interpretable results.

Generation of Graded Adversarial Behaviors: The algorithm can generate adversarial behaviors of different levels for testing autonomous driving systems. Experimental results show that the algorithm can produce corresponding adversarial behaviors based on the set adversarial levels, thereby validating the response capabilities of autonomous driving systems.

Improved Testing Efficiency and Critical Scenario Generation: The proposed algorithm enhances testing efficiency significantly. High adversarial levels increase collision rates, offering crucial data for system optimization. It generates critical interaction scenarios, crucial for training deep reinforcement learning-based systems.

Future research will focus on enhancing the algorithm's adaptability to more complex traffic scenarios. Integrating real-world data and refining the adversarial behavior models will further improve the system's robustness. Additionally, expanding the scope to include interactions with non-vehicular road users, like pedestrians and cyclists, will be crucial. These steps aim to create safer and more reliable autonomous driving systems.

Author Contributions: Conceptualization, X.Z. and H.L.; methodology, X.Z.; formal analysis, J.W.; investigation, J.W. and H.W.; resources, X.Z.; data curation, J.W.; writing—original draft preparation, X.Z.; writing—review and editing, X.Z., H.L. and J.W.; visualization, X.Z., J.W. and H.W.; supervision, X.Z.; project administration, H.L.; funding acquisition, H.L. All authors have read and agreed to the published version of the manuscript.

Data Availability Statement: The scene data in the article comes from NGSIM, and the relevant data link is https://data.transportation.gov/Automobiles/Next-Generation-Simulation-NGSIM-Vehicle-Trajector/8ect-6jqj/about_data (accessed on 12 June 2024), At the same time, the author of this article will open source a software-in-the-loop test system based on CARLA that integrates the full-stack algorithm of autonomous driving, please contact the author for specific data details.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Pei, H.; Zhang, J.; Zhang, Y.; Pei, X.; Feng, S.; Li, L. Fault-tolerant cooperative driving at signal-free intersections. *IEEE Trans. Intell. Veh.* **2023**, *8*, 121–134. [CrossRef]
2. He, Z.; Pei, H.; Guo, Y.; Yao, D.; Zhang, Y.; Li, L. Comparative analysis of cooperative driving strategies for CAVs at on-ramps. *IEEE Trans. Veh. Technol.* **2023**, *72*, 8099–8104. [CrossRef]
3. Lee, S.; Arvin, R.; Khattak, A.J. Advancing investigation of automated vehicle crashes using text analysis of crash narratives and Bayesian analysis. *Accid. Anal. Prev.* **2023**, *181*, 106932. [CrossRef] [PubMed]
4. Sun, J.; Zhang, H.; Zhou, H.; Yu, R.; Tian, Y. Scenario-based test automation for highly automated vehicles: A review and paving the way for systematic safety assurance. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 14088–14103. [CrossRef]
5. Feng, S.; Yan, X.; Sun, H.; Feng, Y.; Liu, H.X. Intelligent driving intelligence test for autonomous vehicles with naturalistic and adversarial environment. *Nat. Commun.* **2021**, *12*, 748. [CrossRef] [PubMed]
6. Yan, X.; Zou, Z.; Feng, S.; Zhu, H.; Sun, H.; Liu, H.X. Learning naturalistic driving environment with statistical realism. *Nat. Commun.* **2023**, *14*, 2037. [CrossRef] [PubMed]
7. Feng, S.; Sun, H.; Yan, X.; Zhu, H.; Zou, Z.; Shen, S.; Liu, H.X. Dense reinforcement learning for safety validation of autonomous vehicles. *Nature* **2023**, *615*, 620–627. [CrossRef] [PubMed]
8. Zhou, J.; Wang, L.; Wang, X. Online Adaptive Generation of Critical Boundary Scenarios for Evaluation of Autonomous Vehicles. *IEEE Trans. Intell. Transp. Syst.* **2023**, *24*, 6372–6388. [CrossRef]
9. Ding, W.; Xu, C.; Arief, M.; Lin, H.; Li, B.; Zhao, D. A survey on safety-critical driving scenario generation a methodological perspective. *IEEE Trans. Intell. Transp. Syst.* **2023**, *24*, 6971–6988. [CrossRef]
10. Zhao, D.; Lam, H.; Peng, H.; Bao, S.; LeBlanc, D.J.; Nobukawa, K.; Pan, C.S. Accelerated evaluation of automated vehicles safety in lane-change scenarios based on importance sampling techniques. *IEEE Trans. Intell. Transp. Syst.* **2017**, *18*, 595–607. [CrossRef] [PubMed]
11. Huang, Z.; Lam, H.; LeBlanc, D.J.; Zhao, D. Accelerated evaluation of automated vehicles using piecewise mixture models. *IEEE Trans. Intell. Transp. Syst.* **2018**, *19*, 2845–2855. [CrossRef]
12. Rana, A.; Malhi, A. Building safer autonomous agents by leveraging risky driving behavior knowledge. In Proceedings of the 2021 International Conference on Communications, Computing, Cybersecurity, and Informatics (CCCI), Beijing, China, 15–17 October 2021; pp. 1–6.
13. Ding, W.; Lin, H.; Li, B.; Eun, K.J.; Zhao, D. Semantically adversarial driving scenario generation with explicit knowledge integration. *arXiv* **2021**, arXiv:2106.04066.
14. Smullyan, R.M. *First-Order Logic*; Courier Corporation: Chelmsford, MS, USA, 1995.
15. Klischat, M.; Liu, E.I.; Holtke, F.; Althoff, M. Scenario factory: Creating safety-critical traffic scenarios for automated vehicles. In Proceedings of the 2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC), Rhodes, Greece, 20–23 September 2020; pp. 1–7.
16. Bennett, J. *OpenStreetMap*; Packt Publishing Ltd.: Birmingham, UK, 2010.
17. Lopez, P.A.; Behrisch, M.; Bieker-Walz, L.; Erdmann, J.; Flötteröd, Y.P.; Hilbrich, R.; Lücken, L.; Rummel, J.; Wagner, P.; Wiessner, E. Microscopic traffic simulation using SUMO. In Proceedings of the 2018 21st International Conference on Intelligent Transportation Systems (ITSC), Maui, HI, USA, 4–7 November 2018; pp. 2575–2582.
18. Wachi, A. Failure-scenario maker for rule-based agent using multi-agent adversarial reinforcement learning and its application to autonomous driving. In Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI-19), Macau, China, 10–16 August 2019; pp. 6006–6012.
19. Xiong, Z.; Xu, H.; Li, W.; Cai, Z. Multi-source adversarial sample attack on autonomous vehicles. *IEEE Trans. Veh. Technol.* **2021**, *70*, 2822–2835. [CrossRef]
20. Xu, X.; Zhang, J.; Li, Y.; Wang, Y.; Yang, Y.; Shen, H.T. Adversarial attack against urban scene segmentation for autonomous vehicles. *IEEE Trans. Ind. Inform.* **2021**, *17*, 4117–4126. [CrossRef]
21. Kong, Z.; Guo, J.; Li, A.; Liu, C. PhysGAN: Generating physical-world-resilient adversarial examples for autonomous driving. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 14242–14251.

22. O'Kelly, M.; Sinha, A.; Namkoong, H.; Duchi, J.; Tedrake, R. Scalable end-to-end autonomous vehicle testing via rare-event simulation. In Proceedings of the 32nd Conference on Neural Information Processing Systems (NeurIPS 2018), Montréal, QC, Canada, 3–8 December 2018; pp. 9827–9838.

23. Chen, B.; Chen, X.; Wu, Q.; Li, L. Adversarial evaluation of autonomous vehicles in lane-change scenarios. *IEEE Trans. Intell.Transp. Syst.* **2022**, *23*, 10333–10342. [CrossRef]

24. Wang, J.; Pun, A.; Tu, J.; Manivasagam, S.; Sadat, A.; Casas, S.; Ren, M.; Urtasun, R. AdvSim: Generating safety-critical scenarios for self-driving vehicles. In Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 20–25 June 2021; pp. 9904–9913.

25. Szczepanski, R.; Tarczewski, T.; Erwinski, K. Energy efficient local path planning algorithm based on predictive artificial potential field. *IEEE Access* **2022**, *10*, 39729–39742. [CrossRef]

26. Wei, C.; Romano, R.; Merat, N.; Wang, Y.; Hu, C.; Taghavifar, H.; Hajiseyedjavadi, F.; Boer, E.R. Risk-based autonomous vehicle motion control with considering human driver's behavior. *Transp. Res. Part C Emerg. Technol.* **2019**, *107*, 1–14. [CrossRef]

27. Malin, F.; Norros, I.; Innamaa, S. Accident risk of road and weather conditions on different road types. *Accid. Anal. Prev.* **2019**, *122*, 181–188. [CrossRef] [PubMed]

28. Liu, E.I.; Würsching, G.; Klischat, M.; Althoff, M. CommonRoad-Reach: A toolbox for reachability analysis of automated vehicles. In Proceedings of the 2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC), Macau, China, 8–12 October 2022; pp. 2313–2320.

29. Althoff, M.; Kochdumper, N. *CORA 2016 Manual*; TU Munich: Garching, Germany, 2016; p. 85748.

30. Chen, X.; Sankaranarayanan, S.; Abraham, E. Flow* 1.2: More effective to play with hybrid systems. In *ARCH14-15. 1st and 2nd International Workshop on Applied veRification for Continuous and Hybrid Systems*; EasyChair: Stockport, UK, 2015; Volume 34, pp. 152–159.

31. Geretti, L.; Sandretto, J.A.D.; Althoff, M.; Benet, L.; Collins, P.; Forets, M.; Ivanova, E.; Li, Y.; Mitra, S.; Mitsch, S.; et al. ARCH-COMP23 category report: Continuous and hybrid systems with nonlinear dynamics. In *10th International Workshop on Applied Verification of Continuous and Hybrid Systems*; EasyChair: Stockport, UK, 2023; pp. 61–88.

32. Li, R.; Zhu, H.; Banach, R. Translating and verifying Cyber–Physical systems with shared-variable concurrency in SpaceEx. *Internet Things* **2023**, *23*, 100864. [CrossRef]

33. Eshtehardian, S.; Khodaygan, S. A continuous rrt*-based path planning method for non-holonomic mobile robots using b-spline curves. *J. Ambient. Intell. Humaniz. Comput.* **2023**, *14*, 8693–8702. [CrossRef]

34. Radwan, M.O.; Sedky, A.A.H.; Mahar, K.M. Obstacles avoidance of self-driving vehicle using deep reinforcement learning. In Proceedings of the 2021 31st International Computer Theory and Applications (ICCTA), Alexandria, Egypt, 1–13 December 2021; pp. 215–222.

35. Coifman, B.; Li, L. A critical evaluation of the Next Generation Simulation (NGSIM) vehicle trajectory dataset. *Transp. Res. Part B Methodol.* **2017**, *105*, 362–377. [CrossRef]