future internet

MDPI

Article

Safety Verification of Driving Resource Occupancy Rules Based on Functional Language

Zhixi Hu [1], Yi Zhu [2,3,*], Xiaoying Chen [2] and Yu Zhao [2]

[1] Academic Affairs Office, Changzhou Institute of Technology, Changzhou 213032, China; huzx@czu.edu.cn
[2] School of Computer Science and Technology, Jiangsu Normal University, Xuzhou 221116, China; cxy@jsnu.edu.cn (X.C.); zhaoyu@jsnu.edu.cn (Y.Z.)
[3] Key Laboratory of Safety-Critical Software, Nanjing University of Aeronautics and Astronautics, Ministry of Industry and Information Technology, Nanjing 211106, China
* Correspondence: zhuy@jsnu.edu.cn

Abstract: Autonomous driving is a safety-critical system, and the occupancy of its environmental resources affects the safety of autonomous driving. In view of the lack of safety verification of environmental resource occupation rules in autonomous driving, this paper proposes a verification method of automatic driving model based on functional language through CSP_M. Firstly, the modeling and verification framework of an autopilot model based on CSP_M is given. Secondly, the process algebra definition of CSP_M is given. Thirdly, the typical single loop environment model in automatic driving is abstracted, and the mapping method from automatic driving model to CSP is described in detail for the automatic driving environment and the typical collision, overtaking, lane change and other scenes involved. Finally, the autopilot model of the single loop is mapped to CSP_M, and the application effect of this method is discussed by using FDR tool. Experiments show that this method can verify the safety of autonomous driving resources, thereby improving the reliability of the autonomous driving model.

Keywords: autonomous driving model; environmental resources; safety verification; process algebra

check for updates

Citation: Hu, Z.; Zhu, Y.; Chen, X.; Zhao, Y. Safety Verification of Driving Resource Occupancy Rules Based on Functional Language. *Future Internet* **2022**, *14*, 60. https://doi.org/10.3390/fi14020060

Academic Editor: Vijayakumar Varadarajan

Received: 16 January 2022
Accepted: 15 February 2022
Published: 17 February 2022

1. Introduction

The development of the Automated Driving System (ADS) has been around for a long time, and the degree of intelligence has continued to increase after the 21st century [1–6]. Autonomous driving technology is an important research topic at present. Autonomous driving technology has many advantages. First of all, the automatic driving system is very intelligent and has high-efficiency computing power. It can accurately identify various parameters of the car when it is running, and the activity status of the surrounding cars, which can assist humans in grasping the driving environment. This is the first benefit of autonomous driving. The second benefit is that autonomous driving can enhance road safety. Now that the traffic is developed, the car ownership rate has been rising, and traffic accidents have been happening all the time, but 80% of the traffic accidents are caused by human factors. Further, autonomous driving technology excludes human factors and can effectively reduce road accidents. The third benefit is that autonomous driving technology can take advantage of the ecological environment. One of the best aspects of a self-driving car is that it uses electricity as its primary energy source. As a result, these cars are more environmentally friendly and cost less. In addition, self-driving cars do not emit any tailpipe, thus helping to reduce air pollution [7–9]. However, with the continuous development of autonomous driving, the safety issues of autonomous driving have gradually emerged [10,11]. In 2016, a Model S car collided due to the fact that it did not recognize the white tractor ahead when the autopilot was turned on [12]. In 2018, an Uber car hit a pedestrian to death because it did not recognize the pedestrian in the shadow [13]. At present, the main causes of autonomous driving accidents are: violation of traffic rules, passive collisions and insufficient environmental perception information [14]. Among them, there are many accident problems caused by insufficient environmental information perception, and the autonomous driving environment has uncertain factors, for example, it is affected by current roads or other vehicles. The current movement of vehicles has a certain degree of uncertainty. The problem of resource occupancy in autonomous driving scenarios also affects the safety of autonomous driving. Therefore, the safety verification of the resource occupation rules of automatic driving has also become a key issue to ensure the safety of automatic driving.

The traffic situation of autonomous driving is complex, and the road is also very complex, so it is very necessary to extract scene information. The roundabout has a central island in the center of the intersection, and all vehicles entering the roundabout travel in a counterclockwise direction. The roundabout is an accident-prone area, and we chose the roundabout to study its safety. On the one hand, the roundabout scene is a complex form of straight road sections, and the research on the roundabout scene is representative to a certain extent, and the safety verification of the roundabout scene can also guide the safety research of other scenarios. On the other hand, due to the frequent occurrence of accidents in the roundabout, the safety is difficult to guarantee, and it has great research significance for the abstraction of the roundabout scene and the safety verification of the vehicle form. In a single-loop scenario, vehicles from multiple directions are often involved. How to coordinate the vehicles coming from multiple parties and ensure the safe driving of vehicles is also particularly important. Therefore, this paper uses the roundabout scenario to study the safety of autonomous driving.

Formal methods are accurate and verifiable, have strict mathematical language and strict mathematical semantics, and can model and verify the system. It originated from the program verification of Dijkstra and Hoare, which is convenient for machine support and automatic processing [15]. Due to its accuracy and verifiability, formal methods are widely used in safety-critical scenarios such as autonomous driving. Process algebra is a formal method used to solve the communication problems of concurrent systems. It can describe the concurrent, asynchronous and non-deterministic behaviors of the system. It is very suitable for the modeling of automatic driving environment model and the analysis of resource security occupation. Communication sequential process (CSP) was proposed by Hoare [16] in 1978. CSP can perform model checking on concurrent systems. CSP_M is an inert functional programming language based on CSP, which can be executed by machines. Failure Divergence Refinement (FDR) is an analysis program tool based on CSP_M description [17].

This paper combines the autopilot model with the functional language CSP_M to verify the resource occupation rules in the autopilot scene. We abstract the scene into multiple segments, and abstract the typical collision, overtaking, lane change and other scenarios of automatic driving into the occupation and operation of the resources of the segment. Firstly, the environment of autonomous driving is modeled, and the environment model is abstracted (Abstract Environment, AE). Secondly, the typical single-loop environment model in autonomous driving is abstracted, and the mapping method from automatic driving model to CSP is described in detail for the automatic driving environment and the typical collision, overtaking, lane change and other scenes involved. Then, the model checking tool FDR is used to test the CSP_M model and analyze its state space diagram to determine whether it will produce a collision unsafe situation. Finally, the results of CSP_M model checking are used to modify the autopilot model, which can improve the reliability of the autopilot model to a great extent.

We abstract the scene into simpler segments, simplify the driving of the vehicle into segment operations, verify the whole process, and analyze the safety of the vehicle in a typical roundabout scenario. Therefore, it has certain guiding significance for the research on the scene safety of autonomous driving.

Section 2 of this paper compares related work; Section 3 proposes a CSP_M-based modelling and verification framework for automated driving models; Section 4 gives the process algebraic definition of $\mathrm{CSP_M}$; Section 5 abstracts the environment model; Section 6 describes the mapping method from automatic driving environment to $\mathrm{CSP_M}$ in detail; Section 7 summarizes the full text and prospects the future work.

2. Related Work

The current research on the environment of automatic driving mainly focuses on the decision-making of safe driving and environmental perception.

Aiming at the decision-making of safe driving, Xin [18] et al. studied the influence of the driver's state on driving, thereby modeling and verifying the driver's state. Doan [19] et al. proposed multi strategy decision making, by assuming that the controlled vehicles and other traffic participants execute a strategy from a set of plausible closed-loop strategies at each time step, so as to find the best strategy that can be executed by controlling vehicles.

For environment perception, Galceran [20] et al. proposed a method to enhance the standard dynamic target tracker. On the one hand, it estimates the occlusion state of other traffic agents, on the other hand, it closely links the occlusion estimation with the new observation data after the tracked target reenters. Xu [14] et al. abstracted the environment and mapped it to Stochastic Hybrid Automata to study the probability of occurrence of events to quantitatively describe uncertain events. Tscharn [21] et al. also used the environment as a reference, combining language and indicating gestures to guide the car. Luo [22] et al. simulated the variable phase traffic light control system and identified its accident scenarios to avoid accidents completely. Zeng [23] et al. summarized the taxi scheduling problem in autonomous driving. Tang [24] et al. shared their research on building a production distributed autonomous driving simulation platform. Zhang [25] et al. proposed a safety risk intelligent prediction model for the train control system to predict which risk state will occur under specific operating conditions. Kinoshita [26] et al. consider the delay of driver behavior to verify the automatic driving system in the model test.

Some researchers focus on motion planning problems for autonomous driving [27–29]. Claussmann [30] et al. proposed an advanced prediction scheme for motion planning from three aspects: risk evaluation, criteria minimization, and constraint submission. The detailed conditions of motion planning are described under the framework of highway and restricted environment with small curvature and specific driver rules, and the corresponding motion planning algorithm is proposed. Li [31] et al. proposed a novel Integrated Local Trajectory Planning and Tracking Control (ILTPTC) framework for autonomous vehicles traveling along a reference path and avoiding obstacles. The framework adopts an efficient state-space sampling-based trajectory planning scheme to smoothly follow the reference path. A model-based predictive path generation algorithm is applied to generate a set of feasible motion paths that take into account both motion speed and driver safety and comfort.

In addition, there are some studies to improve the safety of automatic driving from the technical aspect. Considering the environmental changes caused by the appearance changes caused by short-term (such as weather, light) and long-term (such as season, vegetation growth, etc.), literature [19] proposed a new location recognition technology, which can effectively retrain and compress. Fujiyoshi [32] et al. described the method in the field of deep learning image recognition.

Compared with the above research work, this paper extracts the automatic driving environment model, maps the running cars into processes, maps the roads into corresponding resources, maps the driving of cars on the roads into the occupation of resources and specifies a series of transformation rules to convert the environment model into a process algebra model for verification. It provides a new idea for the research of the automatic driving model. In addition, using algebra to abstract the environment model can effectively improve the formulation of environment model strategy and ensure the safety of automatic driving.

3. Autonomous Driving Model Modeling and Verification Framework Based on CSP$_\mathrm{M}$

In order to improve the reliability of autopilot, we verify the security of autopilot resource occupation rules, and propose the current modeling and verification framework of autopilot resource occupation rules based on CSP$_\mathrm{M}$ in this section. As shown in Figure 1.

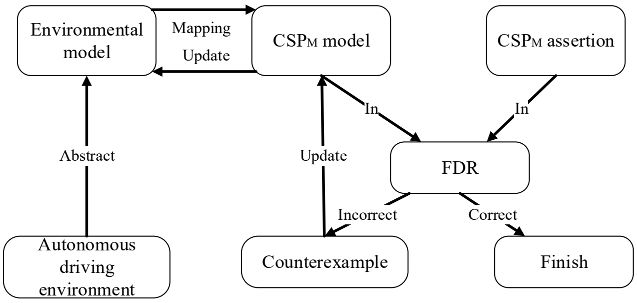


Figure 1. CSP$_\mathrm{M}$-based modeling and verification framework for autonomous driving resource occupancy rules.

The framework first abstracts the autonomous driving environment and abstracts the environment model. Secondly, formulate a series of conversion rules to map the autonomous driving model to the CSP$_\mathrm{M}$ model. Then, input the converted CSP$_\mathrm{M}$ model into the FDR tool to verify it, and modify the CSP$_\mathrm{M}$ model according to the counterexample, thereby modifying the corresponding autopilot model. Finally, an autopilot model that meets the safety rules can be obtained. Among them, the environment model is a simplified abstract form of the model of the current scene, and we use the formal model CSP$_\mathrm{M}$ to describe the abstract environment model. CSP$_\mathrm{M}$ asserts that in order to satisfy the attribute specification of the current environmental security, we input the model and the specification of the model together into the CSP's model checking tool FDR to verify the security attribute, and modify it according to the result.

4. CSP$_\mathrm{M}$ Definition

4.1. The Syntax of CCSP

For a system S, the process variable set $P{:=}\{X,Y,P,Q \dots \}$ represents the process in the CSP, and the discrete variable set $DV{:=}\{d,t,m,n,x,y,z \dots \}$ represents the system variable. The conditional communication sequence process CCSP can be defined as: $STOP|SKIP|WAIT\ t|a \to Q|P;Q|P\Box Q|P \sqcap Q|P \overset{d}{\triangleright} Q|PSQ|P\Delta Q|P\backslash A|f(\mathrm{X})|\mu X \cdot f(X)|P/(K)Q$.

STOP: which means the interruption of a process. The process does not communicate with the outside, which can indicate deadlock or process non convergence;

SKIP: which indicates that a process does nothing except terminate;

WAIT t: which means that the process terminates after t time without doing anything;

$a{\to}Q$: prefix operation, which means that process Q is executed after event a is executed;

P;Q: sequential composition, which means that the process Q is executed after the process P is executed;

$P\Box Q$: external selection, which means that the execution process P or Q depends on the first event of the process execution;

$P \sqcap Q$: internal selection, which means that the execution process P or Q is determined by the process internally;

$P \overset{\mathrm{d}}{\triangleright} Q$: timeout, if the two processes do not communicate within d time, it is considered timeout, and the control right is handed over from P to Q;

PSQ: time interruption, regardless of whether the P process is completed after t time, interrupt process P and execute process Q;

$P\triangle Q$: interruption, the execution of any event of Q can cause the interruption of P;

P[R]: change the mark, *P[R]* and process P have the same structure, but the event in P is mapped to another name through the relationship R. For example, the following process

P continuously executes event a. Process Q is equal to all occurrences of a in process P and replaced with b: *P = a→P, Q = P[a→b];*

P/A: set hiding means that any event belonging to A in process P is not displayed;

$P||Q$: synchronous concurrency, P and Q are concurrent in the events of the set C, and the events of other sets are interleaved;

$P|||Q$: asynchronous concurrency, each event executed by the process is an event in the process P or Q;

$\mu X \cdot f(X)$: *f(X)* is a prefix expression containing the process variable X;

P/(K)Q: which is our extended condition operator, indicates that process P and process Q are not allowed to be executed under K at the same time. K is a set form, $K := \{(t_0,k_0),(t_1,k_1) \ldots (t_n,k_n)\}$.

4.2. Conditional Time Migration System of CCSP

The following discusses the semantics of the conditional time migration system of CCSP. Literature [32] defines the semantics of CSP as a time migration system. The execution of an event in the process can be regarded as a migration in the time migration system. We introduce the concept of conditional time migration system when studying CCSP.

Definition 1. *The semantics of a communication sequential process CSP is described as a migration system* $TS_{CSP} = <NODES, \sum, \rightarrow>$, *NODES is a collection of nodes, representing each process.* $\sum$ *is the event set, i.e.,* $\{a_0, a_1 \ldots a_n\}$, $\rightarrow$ *is a migration relationship,* $\rightarrow$ *is a ternary relationship.* $\rightarrow \subseteq NODES \times \sum \times NODES$, $N_1 \overset{a}{\rightarrow} N_2$, *indicates that* N_1 *executes event a and becomes the process represented by* N_2.

Definition 2. *A conditional time migration system CCSP is* $CTS_{CCSP} = <NODES, \sum_{(C,P)}, \rightarrow>$, *NODES is a collection of nodes, representing each process.* $\sum_{(C,P)}$ *is the event set with conditional execution, i.e.,* $\{(c_0, a_0), (c_1, a_1) \ldots (c_n, a_n)\}$, *and* $\sum_T \subseteq \sum_{(C,P)}$, *when* $c_n = true$, $\sum_T = \sum_{(C,P)}$. $\rightarrow$ *is a migration relationship,* $\rightarrow$ *is a ternary relationship.* $\rightarrow \subseteq NODES \times \sum_{(C,P)} \times NODES$, $N_1 \overset{(c,a)}{\rightarrow} N_2$ *means that the process executed by* N_1 *executes event a when the condition c is met, and becomes the process represented by* N_2.

Definition 3. *A conditional time communication sequence process can be described as a conditional time migration system* $CTS_{CCSP} = <NODES, \sum_{(C,P)}, \rightarrow>$.

4.3. CCSP Operational Semantics

Definition 4. *The operational semantics of CCSP. The operation semantics of the above conditional operator P/(K)Q are given below:*

$$\frac{t = t', a \notin K}{s \overset{(c,a)}{\rightarrow} s'} \tag{1}$$

$$\frac{t = t', a \in K}{s \overset{(c,a)}{\rightarrow} s} \tag{2}$$

At t = t′, if the current event a is not in the K set, the state s will execute the event a and become the s′ state, i.e., Equation (1), when the condition c is met, otherwise no state transition, i.e., Equation (2).

4.4. Refinement Relationship

It is necessary to prove that the semantic model of CSP is a sub-semantic model of CCSP. We first define its sub-semantic model, and then give the theorem proof that CCSP contains CSP semantics, proving that CCSP is extended on the basis of CSP.

Definition 5. *Suppose a communication sequence process* P_A *with semantic* M_A *of type A can obtain another communication sequence process* P_B *with semantic* M_B *of type B through a refined* *relationship, then semantic* M_B *is the sub-semantic model of semantic* M_A [33]. *The refined models of* P_A *and* P_B *are as follows:*

$$\frac{P_B\ sat\ S_B\ in\ M_B}{P_A\ sat\ S_A\ in\ M_A}(P_B \sqsubseteq P_A) \tag{3}$$

S_A *is the semantics of* P_A *, and* S_B *is the semantics of* P_B.

Definition 6. *All acceptable languages of a conditional communication sequence process CCSP are a collection of conditional migration sequences.*

Theorem 1. The semantic model of the communication sequence process CSP is a sub-semantic model of the conditional communication sequence process CCSP.

Proof of Theorem 1. Let $\mathrm{P}_{\mathrm{CSP}}$ be a communication sequence process. From **Definition 1** we know that CSP is a migration system $\mathrm{TS}_{\mathrm{CSP}}$ = <*NODES*,Σ,→>, and its accepted language is *L*. Construct a conditional communication sequence process according to the communication sequence process. From **Definition 3** we know that P_{CCSP} is TS_{CCSP} = <*NODES*,$\Sigma_{(C,P)}$,→>. Suppose the position time language that P_{CCSP} can accept is L'. At this time, take any conditional transition sequence *R* = <(a_0),(a_1) … ($a_{n\text{-}1}$)> in *L*, and there is only one R' = <(c_0,a_0),(c_1,a_1) … ($c_{n\text{-}1}$,$a_{n\text{-}1}$)> corresponding to it in L'. Therefore, $\sum\subseteq \sum_{(\mathrm{C,P})}$ and when $c_n = true$, $\sum_T = \sum_{(C,P)}$ is the refined relationship from P_{CCSP} to P_{CSP}. □

Then, according to **Theorem 1**, all model checking related to functional properties on CCSP can be completed by CSP's model checking tool. Conditional analysis is required for the extended conditions to determine whether the current resource occupancy rules meet the safety. When it is not, perform certain fault-tolerant measures [34].

4.5. Analysis of Resource Occupation Rules

The position time migration system of CCSP is obtained by expanding the position factor on the time migration system of CSP, and its state space can construct a reachable graph *G*. *G*:=(*NODES*,*EDGES*), *NODES* = {N_i | N_i *is the node where the process executes*, $0 \leq I \leq n$}, *EDGES* = {*e*(*i*,*j*) | *e*(*i*,*j*) *represents a directed edge from* N_i *to* N_j,$0 \leq i \leq n$, $0 \leq j \leq n$, *and* $I \neq j$}.

The inspection of resource occupation rules needs to check whether multiple entities at a certain moment will be in the same segment.

In this paper, to solve this problem, on the basis of the state space graph *G*, we judge whether *G* is a canonical reachable graph, that is, whether all the paths of the state space can satisfy the different segment spaces occupied at the same time. A resource occupancy rule security satisfiability check algorithm is used to determine whether *G* is a standardized reachability graph. In this algorithm, we set the current time to *t*, use *set_path* to store the visited path, and *abnormal* to store the abnormal node. The Algorithm 1 details are as follows:

Algorithm 1

```
Initialization: time:= t; abnormal:= ∅; set_path:= {$N_0$}; total:= 0;
Repeat
    ln:= last node in set_path; //take the last node of the current path
    If successor nodes of last node have been visited; //delete visited child nodes
        Then delete last node of set_path;
        total:= total − curr_t; //when deleting the last node, the total time must be subtracted from the relevant edge
    Else
        Begin
            bn:= take a unvisited successor node of ln; //take a child node bn of ln that has not been visited
            total:= total + curr_t;
            result:= true;
            If $k_{ln} = k_{bn}$ then
            result:= false; //if the time is the same in the same segment k, there is a danger
            abnormal = abnormal ∪ {en};
            Else
            set_path:= set_path ∪ {bn};
        End
Until set_path = ∅;
If abnormal = ∅ then
Return true;
Else return false;
```

The input of this algorithm is graph G, and the output is whether graph G satisfies the security occupancy rules of the segment. If the algorithm is satisfied, the return value is true, otherwise the return value is false.

5. Abstraction of the Environmental Model

The single-loop model is a typical autonomous driving scenario. We take the single-loop as an example to realize the abstraction of the environment model. As shown in Figure 2, we abstract the single-loop model.

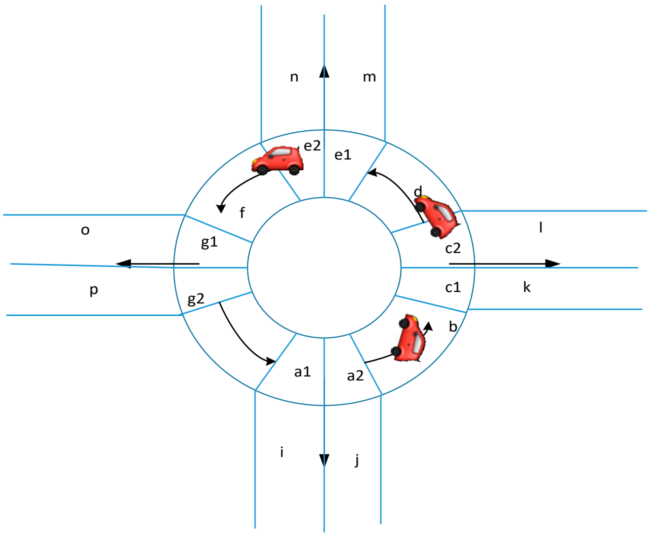


Figure 2. Single-loop diagram.

Environmental Model

Definition 7. *Environment. The environment is a section of autonomous driving environment, including the static environment and dynamic environment of autonomous driving. For example, the big tree on the roadside is the static environment, and the moving car belongs to the dynamic environment. In this paper, the single ring road and the car running in it are an environment.*

Definition 8. *Segment. The roads included in the environment can be divided into several segments, as shown in Figure 2. The segments of single ring roads are {a1,a2,b,c1,c2,d,e1,e2,f,g1,g2,h,i,j,k,l,m,n,o,p}.*

Definition 9. *Arc. The line segment connecting two adjacent segments is an arc, and some arcs are drawn in the Figure 2. For example, if a1 is connected with a2,* $a\overset{\frown}{1}a2$ represents the arc of segment a1 and a2, connecting segments a1 and a2.

Definition 10. *Route. The route is a segment that does not violate traffic rules and is composed of several adjacent arcs. For example, in the Figure 2, j->a2->b->c1->k is a route.*

Definition 11. *Car. The moving objects that appear in the scene are marked as car and numbered in the corresponding order. In the Figure 2, the three moving objects are car1, car2 and car3.*

Definition 12. *Scene.*

(1) Collision The cars are in the same segment at the same time. For example, at the same time, car1 and car2 are both in the f segment, which is a collision.
(2) Overtake For multiple roads, the car behind overtakes the car in front. Overtaking is not allowed in the single ring road in this paper.
(3) Lane change Drive to other lanes while driving. Lane changes are not allowed in the single ring road in this paper.
(4) Message transmission Message transmission refers to the signal transmitted by the signal lights of the preceding vehicle during driving. For example, if the right turn signal of the vehicle in front is turned on, the signal is transmitted and the vehicle in front will turn right.

6. Mapping Method from Autonomous Driving Decision Language to CSP_M Language

Next, we map the autonomous driving environment to the CSP, and we divide the process into environment mapping and autonomous vehicle mapping. To simplify the model, this paper only considers several typical driving behaviors in autonomous driving: collision, overtaking and lane change.

6.1. Environment Mapping

(1) Segment. In the autonomous driving environment, the segment variable of the highway is mapped to the event variable in CSP.
(2) Arc. Arcs are mapped to the order of execution of events in the process. For example, $a\overset{\frown}{1}a2$ is mapped to execute *a1* first before executing *a2*.
(3) Car. The driving path of the car is mapped to a process and executed synchronously in the driving environment according to certain rules in the environment. For example, the driving path *j->a2->b->c1->k* of car2 can be mapped to process *CAR2 = j->a2->b->c1->k->STOP*.
(4) Other environment. Any road segment that is not in the environment is not considered, we only consider the road segment partially, and the road segment that is not in the environment is mapped as *STOP* in the process.

6.2. Autonomous Car Scene Mapping

(1) Information transfer between vehicles, such as the right turn signal of the front vehicle, is mapped into communication between processes.

(2) Vehicle collision, that is, occupying the same process resources at the same time or within a time period.
(3) Vehicle lane change, that is, the change of driving path, is mapped to the change of execution sequence of events in the process.
(4) Vehicles overtaking, that is, passing through other lanes.
(5) Driving back to the original lane after changing lanes is also mapped to a change in the execution sequence of events in the process.

6.3. Safety Verification of Resource Occupation in Driving Scenes

According to the above single-loop road map, the car can enter the single-loop road from four segments *{j,l,n,o}*. We take into account cars in four directions.

channel *a1,a2,b,c1,c2,d,e1,e2,f,g1,g2,h,i,j,k,l,m,n,o,p*

```
CAR1 = j− > ((a2− > b− > c1− > k− > CAR1)
[] (a2− > b− > c1− > c2− > d− > e1− > m− > CAR1)
[](a2− > b− > c1− > c2− > d− > e1− > e2− > f− > g1− > 0− > CAR1)
[](a2− > b− > c1− > c2− > d− > e1− > e2− > f− > g1− > g2− > h− > a1− > CAR1)
)
CAR2 = n− > ((e2− > f− > g1− > 0− > CAR2)
[] (e2− > f− > g1− > g2− > h− > a1− > i− > CAR2)
[](e2− > f− > g1− > g2− > h− > a1− > a2− > b− > c1− > k− > CAR2)
[](e2− > f− > g1− > g2− > h− > a1− > a2− > b− > c1− > c2− > d− > e1− > m− > CAR2)
)
CAR3  = l− > ((c2− > d− > e1− > m− > CAR3)
[](c2− > d− > e1− > e2− > f− > g1− > 0− > CAR3)
[](c2− > d− > e1− > e2− > f− > g1− > g2− > h− > a1− > i− > CAR3)
[](c2− > d− > e1− > e2− > f− > g1− > g2− > h− > a1− > a2− > b− > c1− > k− > CAR3)
)
CAR4 = p− > ((g2− > h− > a1− > i− > CAR4)
[](g2− > h− > a1− > a2− > b− > c1− > k− > CAR4)
[](g2− > h− > a1− > a2− > b− > c1− > c2− > d− > e1− > m− > CAR4)
[](g2− > h− > a1− > a2− > b− > c1− > c2− > d− > e1− > e2− > f− > g1− > 0− >  CAR4)
)
```

We use Figure 2 as the scene to test the scenes of car1, car2, and car3. Driving car1 enters the single ring road from segment *j*, car2 enters the single ring road from segment *n*, and car3 enters the single ring road from segment *l*:

$$CARS1 = CAR1 \; [| \; \{||\} \; |] CAR2$$
$$CARS2 = CARS1 \; [| \; \{||\} \; |] CAR3$$

when there are vehicles in other directions, continue to add a process similar to that shown below:

CARS = CARS2 [| { | | } |] CAR4

We save the above CSP$_M$ code as a CAR.csp file and open it through FDR3, and then we use FDR3 to perform model checking on its security and deadlock-free respectively. The CSP$_M$ assertion and model checking security results established in FDR3 are as follows:

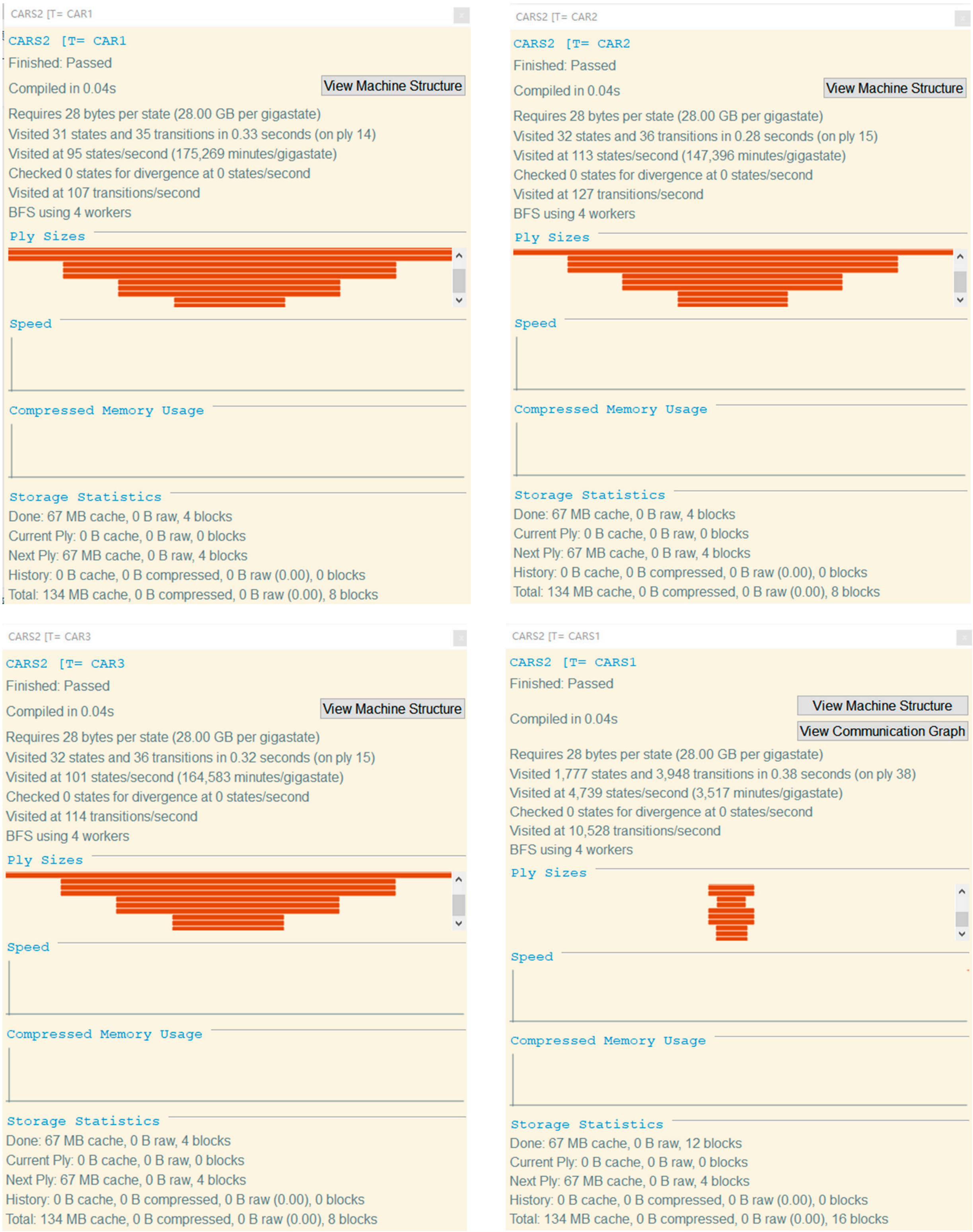
assert CARS2[T = CAR1
assert CARS2[T = CAR2
assert CARS2[T = CAR3
assert CARS2[T = CARS1

The above assertion indicates that the process *CAR1, CAR2, CAR3, CARS1* trace is refined in the process *CARS*, that is:

trace(CAR1)$\subseteq$*trace(CARS2)*
trace(CAR2)$\subseteq$*trace(CARS2)*

trace(CAR3)⊆trace(CARS2)
trace(CARS1)⊆trace(CARS2)

The result of model checking is shown in Figure 3. The results obtained by inputting the above four security statements into the FDR tool are displayed as Passed, which conforms to the security attribute verification of the formal model.



```
CARS2 [T= CAR1
CARS2 [T= CAR1
Finished: Passed
Compiled in 0.04s
Requires 28 bytes per state (28.00 GB per gigastate)
Visited 31 states and 35 transitions in 0.33 seconds (on ply 14)
Visited at 95 states/second (175,269 minutes/gigastate)
Checked 0 states for divergence at 0 states/second
Visited at 107 transitions/second
BFS using 4 workers
Ply Sizes
Speed
Compressed Memory Usage
Storage Statistics
Done: 67 MB cache, 0 B raw, 4 blocks
Current Ply: 0 B cache, 0 B raw, 0 blocks
Next Ply: 67 MB cache, 0 B raw, 4 blocks
History: 0 B cache, 0 B compressed, 0 B raw (0.00), 0 blocks
Total: 134 MB cache, 0 B compressed, 0 B raw (0.00), 8 blocks
```

```
CARS2 [T= CAR2
CARS2 [T= CAR2
Finished: Passed
Compiled in 0.04s
Requires 28 bytes per state (28.00 GB per gigastate)
Visited 32 states and 36 transitions in 0.28 seconds (on ply 15)
Visited at 113 states/second (147,396 minutes/gigastate)
Checked 0 states for divergence at 0 states/second
Visited at 127 transitions/second
BFS using 4 workers
Ply Sizes
Speed
Compressed Memory Usage
Storage Statistics
Done: 67 MB cache, 0 B raw, 4 blocks
Current Ply: 0 B cache, 0 B raw, 0 blocks
Next Ply: 67 MB cache, 0 B raw, 4 blocks
History: 0 B cache, 0 B compressed, 0 B raw (0.00), 0 blocks
Total: 134 MB cache, 0 B compressed, 0 B raw (0.00), 8 blocks
```

```
CARS2 [T= CAR3
CARS2 [T= CAR3
Finished: Passed
Compiled in 0.04s
Requires 28 bytes per state (28.00 GB per gigastate)
Visited 32 states and 36 transitions in 0.32 seconds (on ply 15)
Visited at 101 states/second (164,583 minutes/gigastate)
Checked 0 states for divergence at 0 states/second
Visited at 114 transitions/second
BFS using 4 workers
Ply Sizes
Speed
Compressed Memory Usage
Storage Statistics
Done: 67 MB cache, 0 B raw, 4 blocks
Current Ply: 0 B cache, 0 B raw, 0 blocks
Next Ply: 67 MB cache, 0 B raw, 4 blocks
History: 0 B cache, 0 B compressed, 0 B raw (0.00), 0 blocks
Total: 134 MB cache, 0 B compressed, 0 B raw (0.00), 8 blocks
```

```
CARS2 [T= CARS1
CARS2 [T= CARS1
Finished: Passed
Compiled in 0.04s
Requires 28 bytes per state (28.00 GB per gigastate)
Visited 1,777 states and 3,948 transitions in 0.38 seconds (on ply 38)
Visited at 4,739 states/second (3,517 minutes/gigastate)
Checked 0 states for divergence at 0 states/second
Visited at 10,528 transitions/second
BFS using 4 workers
Ply Sizes
Speed
Compressed Memory Usage
Storage Statistics
Done: 67 MB cache, 0 B raw, 12 blocks
Current Ply: 0 B cache, 0 B raw, 0 blocks
Next Ply: 67 MB cache, 0 B raw, 4 blocks
History: 0 B cache, 0 B compressed, 0 B raw (0.00), 0 blocks
Total: 134 MB cache, 0 B compressed, 0 B raw (0.00), 16 blocks
```

Figure 3. Safety inspection diagram.

The deadlock-free results are as follows:

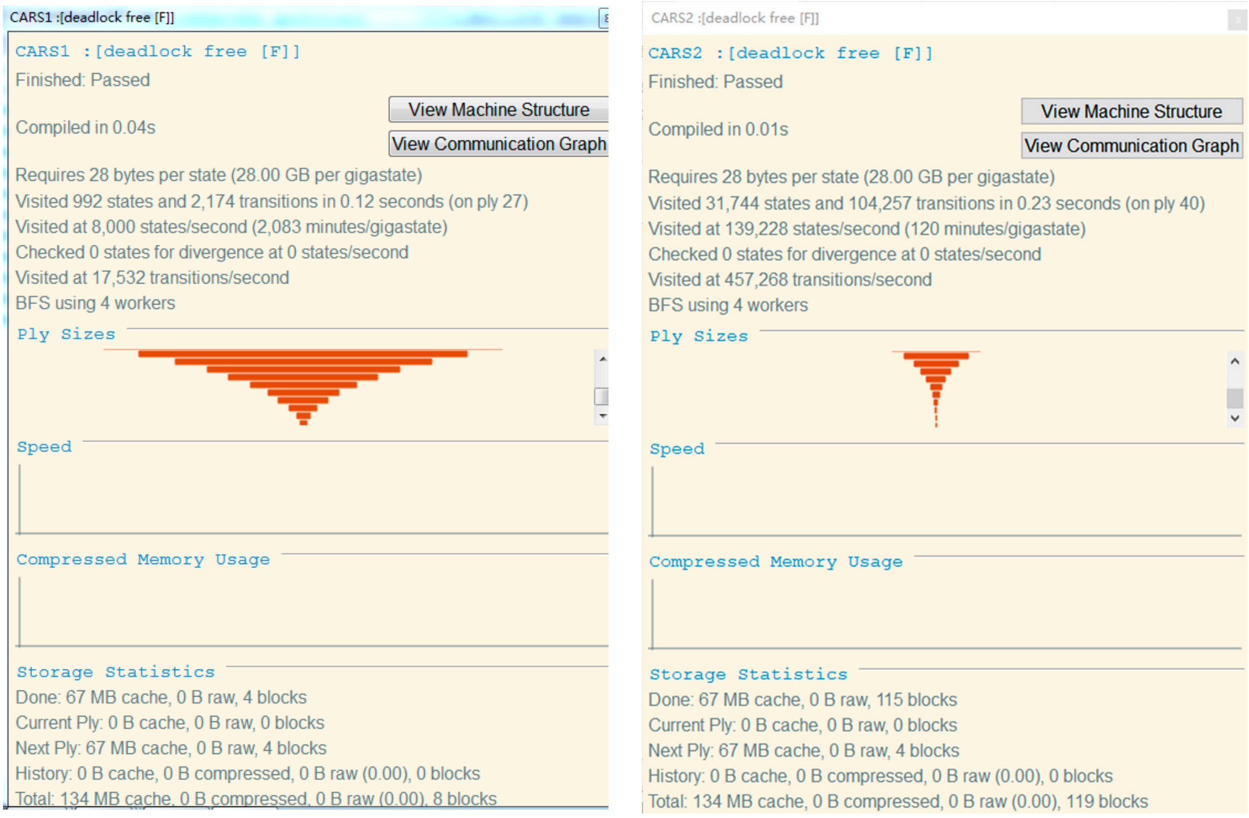
assert CARS1:[deadlock free [F]]

assert CARS2:[deadlock free [F]]

This assertion indicates that the processes *CARS1* and *CARS2* are deadlock-free. For example, for the process *CARS2*, *CARS2* no deadlock indicates that for any trace *s*, $(s, \sum) \notin$ *failures(CARS2)*.

We input the processes CARS1 and CARS2 into FDR, and the result shows Passed, indicating that the two processes have no deadlock situation. That is, in this single-loop scenario, car1, car2 and car3 occupy segment resources to execute corresponding processes without causing deadlock.

The above experimental results in Figure 4 show that all processes have passed the FDR model checking, indicating that the CSP$_M$ model of the autonomous driving environment has extremely high reliability, and meets the characteristics of safety and deadlock-free. Next, we need to check whether the car will be in the same segment at the same time, that is, whether there will be a collision. Since the segment is defined as a unit segment, that is, only one car is allowed to enter in a segment at the same time. Then the definition of collision is that, in the same time period, the segment occupied by the vehicle is the same segment. We represent the state space diagrams of Car1 and Car2, as shown in Figures 5 and 6.



```
CARS1 :[deadlock free [F]]
CARS1 :[deadlock free [F]]
Finished: Passed

Compiled in 0.04s

Requires 28 bytes per state (28.00 GB per gigastate)
Visited 992 states and 2,174 transitions in 0.12 seconds (on ply 27)
Visited at 8,000 states/second (2,083 minutes/gigastate)
Checked 0 states for divergence at 0 states/second
Visited at 17,532 transitions/second
BFS using 4 workers
Ply Sizes
Speed
Compressed Memory Usage
Storage Statistics
Done: 67 MB cache, 0 B raw, 4 blocks
Current Ply: 0 B cache, 0 B raw, 0 blocks
Next Ply: 67 MB cache, 0 B raw, 4 blocks
History: 0 B cache, 0 B compressed, 0 B raw (0.00), 0 blocks
Total: 134 MB cache, 0 B compressed, 0 B raw (0.00), 8 blocks
```

```
CARS2 :[deadlock free [F]]
CARS2 :[deadlock free [F]]
Finished: Passed

Compiled in 0.01s

Requires 28 bytes per state (28.00 GB per gigastate)
Visited 31,744 states and 104,257 transitions in 0.23 seconds (on ply 40)
Visited at 139,228 states/second (120 minutes/gigastate)
Checked 0 states for divergence at 0 states/second
Visited at 457,268 transitions/second
BFS using 4 workers
Ply Sizes
Speed
Compressed Memory Usage
Storage Statistics
Done: 67 MB cache, 0 B raw, 115 blocks
Current Ply: 0 B cache, 0 B raw, 0 blocks
Next Ply: 67 MB cache, 0 B raw, 4 blocks
History: 0 B cache, 0 B compressed, 0 B raw (0.00), 0 blocks
Total: 134 MB cache, 0 B compressed, 0 B raw (0.00), 119 blocks
```

Figure 4. Deadlock-free diagram.

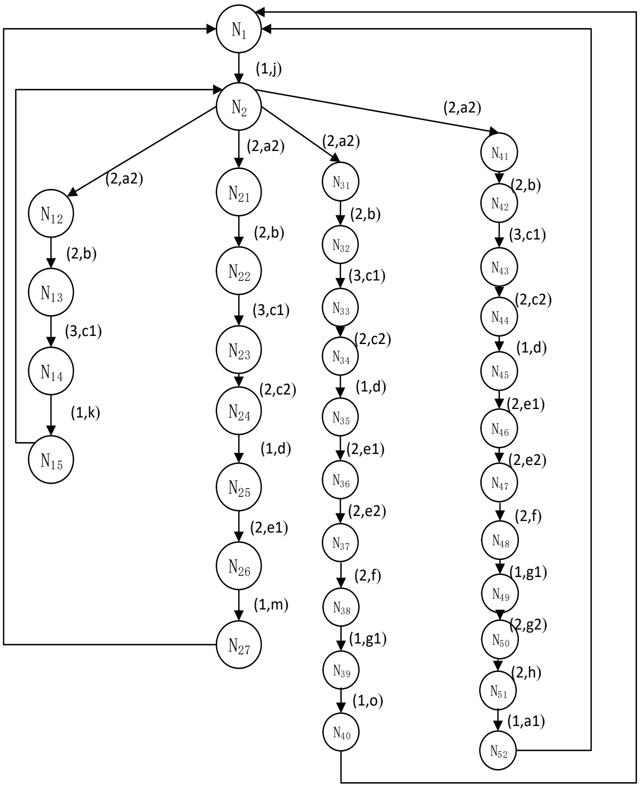


Figure 5. State space of *Car1*.

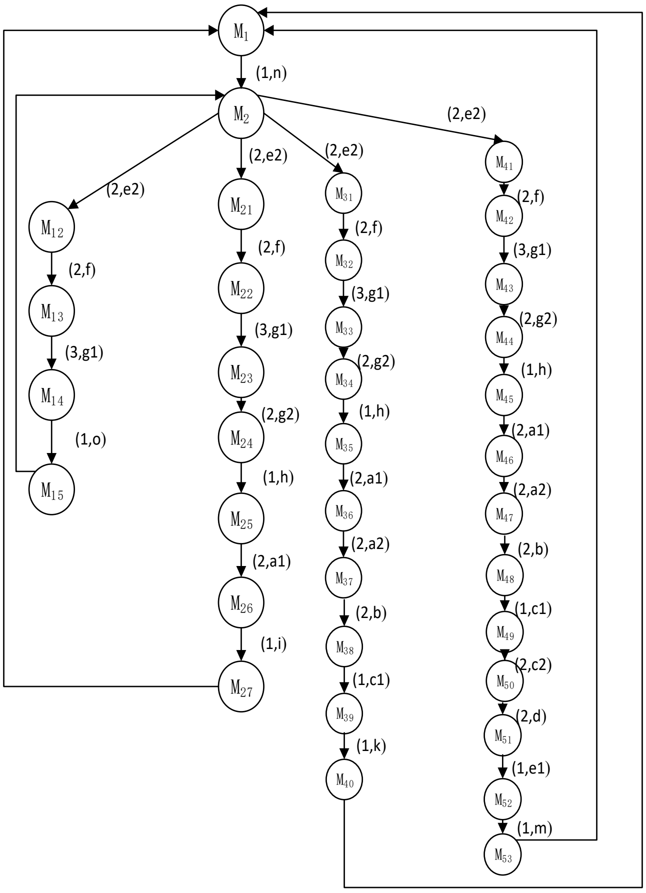


Figure 6. State space of *Car2*.

Figure 5 shows the CAR1 process, that is, the state diagram of Car1. The nodes represent the current state, and there are corresponding edges between the nodes. The edges are marked with (t, frag), indicating the duration t and the segment name frag to enter the next segment. The four branches in the state diagram are the four paths for Car1 to enter the scene.

Figure 6 is the CAR2 process, that is, the Car2 state diagram. The representation and meaning of its edges are the same as those in Figure 5, that is, the duration t and the segment name frag to enter the next segment are marked. The four branches in the state diagram are the four paths for Car2 to enter the scene.

Analyze the segments that may be occupied by *Car1* and *Car2* through the analysis of resource occupancy in the state space. For example, segment *d* will be occupied from N_{24} to N_{25}, and segment *d* will also be occupied from M_{50} to M_{51}. The current time is 1. We use the resource security occupation checking algorithm to calculate that the occupation time of node N_{25} in *Car1* for segment *d* is 11 and that of node N_{25} in *Car2* for segment *d* is 22. Therefore, *Car1* and *Car2* will not occupy segment *d* at the same time. Therefore, the security occupation rules of resources are met. Check the occupancy of other segments in turn to detect the safe occupancy of resources.

7. Summary and Outlook

This paper combines the autonomous driving model with the functional language CSP_M to verify the resource occupancy rules in autonomous driving scenarios. The scene is abstracted into multiple fragments, and the typical collision, overtaking, lane change and other scenes of automatic driving are abstracted into the occupation and operation of fragment resources. The scene is actually modeled as simpler segments, and the driving behavior of the vehicle is modeled as the operation of the segment, on this basis, the whole process of automatic driving is verified. In this paper, the safety of vehicle driving is analyzed in the roundabout scenario, and the results of model checking are used to modify the automatic driving model, which greatly improves the reliability of the automatic driving model. At the same time, the research on the safety of automatic driving scenarios has certain guiding significance.

However, this paper has made certain attempts in terms of environmental resources through a simple single-loop autonomous driving environment abstraction. This model can be used for the operation and verification of the formal rules of this simple scene. However, real autonomous driving scenarios are more complicated. This paper only focuses on simple single loop roads, which is relatively simple and does not consider issues such as time and vehicle energy consumption. Therefore, the next step will be to expand multiple roads and transform the model on the basis of this model, in addition to taking into account issues such as car travel time and energy consumption. In addition, the scenario in this paper is relatively simple. How to complete the abstraction and combination of multiple scenarios, and verify the safety of autonomous driving in more complex scenarios is also the focus of the next research.

Author Contributions: Conceptualization, Z.H. and Y.Z. (Yi Zhu); methodology, Z.H.; software, Z.H. and X.C.; validation, Y.Z. (Yi Zhu) and Y.Z. (Yu Zhao); formal analysis, Z.H.; investigation, Z.H.; resources, Z.H.; data curation, Z.H.; writing—original draft preparation, Z.H.; writing—review and editing, Y.Z. (Yi Zhu) and X.C.; visualization, X.C.; supervision, Y.Z. (Yu Zhao); project administration, X.C.; funding acquisition, Y.Z. (Yi Zhu). All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the National Natural Science Foundation of China (62077029); the Open Project Fund of Key Laboratory of Safety-Critical Software Ministry of Industry and Information Technology (NJ2020022); the Future Network Scientific Research Fund Project (FNSRFP-2021-YB-32); the Applied Basic Research Program of Xuzhou (KC19004).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: There is no problem with the dataset in the paper. For any other questions, please contact the corresponding author or first author of this paper.

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

References

1. Sung, K.; Min, K.-W.; Choi, J.; Kim, B.-C. A formal and quantifiable log analysis framework for test driving of autonomous vehicles. *Sensors* **2020**, *20*, 1356. [CrossRef] [PubMed]
2. Yu, L.; Kong, D.; Yan, X. A driving behavior planning and trajectory generation method for autonomous electric bus. *Future Internet* **2018**, *10*, 51. [CrossRef]
3. Han, J.; Shi, H.; Chen, L.; Li, H.; Wang, X. The car-following model and its applications in the V2X environment: A historical review. *Future Internet* **2022**, *14*, 14. [CrossRef]
4. Zaidi, K.; Rajarajan, M. Vehicular internet: Security & privacy challenges and opportunities. *Future Internet* **2015**, *7*, 257–275.
5. Yoon, Y.; Kim, H. Resolving persistent packet collisions through broadcast feedback in cellular V2X communication. *Future Internet* **2021**, *13*, 211. [CrossRef]
6. Hao, J.; Han, G. On the modeling of automotive security: A survey of methods and perspectives. *Future Internet* **2020**, *12*, 198. [CrossRef]
7. Lijarcio Cárcel, I.; Useche, S.A.; Llamazares, J.; Montoro, L. Perceived benefits and constraints in vehicle automation: Data to assess the relationship between driver's features and their attitudes towards autonomous vehicles. *Data Brief* **2019**, *27*, 104662. [CrossRef]
8. Panagiotopoulos, I.; Dimitrakopoulos, G. An empirical investigation on consumers' intentions towards autonomous driving. *Transp. Res. Part C Emerg. Technol.* **2018**, *95*, 773–784. [CrossRef]
9. Montoro, L.; Useche, S.A.; Alonso, F.; Lijarcio, I.; Bosó-Seguí, P.; Martí-Belda, A. Perceived safety and attributed value as predictors of the intention to use autonomous vehicles: A national study with Spanish drivers. *Saf. Sci.* **2019**, *120*, 865–876. [CrossRef]
10. Alonso, F.; Faus, M.; Esteban, C.; Useche, S.A. Is there a predisposition towards the use of new technologies within the traffic field of emerging countries? The case of the Dominican Republic. *Electronics* **2021**, *10*, 1208. [CrossRef]
11. Useche, S.A.; Peñaranda-Ortega, M.; Gonzalez-Marin, A.; Llamazares, F.J. Assessing the effect of drivers' gender on their intention to use fully automated vehicles. *Appl. Sci.* **2022**, *12*, 103. [CrossRef]
12. Banks, V.A.; Plant, K.L.; Stanton, N.A. Driver error or designer error: Using the perceptual cycle model to explore the circumstances surrounding the fatal Tesla crash on 7th May 2016. *Saf. Sci.* **2018**, *108*, 278–285. [CrossRef]
13. Kohli, P.; Chadha, A. Enabling pedestrian safety using computer vision techniques: A case study of the 2018 uber inc. In Proceedings of the self-driving car crash//Future of Information and Communication Conference, San Francisco, CA, USA, 14–15 March 2019; Springer: Cham, Switzerland, 2019; pp. 261–279.
14. Xu, B.Q. *Research on Formal Modeling and Verification for Safety Analysis of Autonomous Driving*; East China Normal University: Shanghai, China, 2019.
15. Zhu, Y.; Huang, Z.Q.; Zhou, H. Formal method for verifying BPEL model used by functional programming language. *J. Front. Comput. Sci. Technol.* **2018**, *12*, 185–196.
16. Hoare, C.A.R. Communicating sequential processes. *Commun. ACM* **1978**, *21*, 666–677. [CrossRef]
17. Available online: https://cocotec.io/fdr/ (accessed on 15 January 2022).
18. Bai, X.; Xu, C.; Ao, Y.; Chen, B.; Du, D. Learning-based probabilistic modeling and verifying driver behavior using MDP. In Proceedings of the 2019 International Symposium on Theoretical Aspects of Software Engineering (TASE), Guilin, China, 29–31 July 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 152–159.
19. Doan, A.D.; Latif, Y.; Chin, T.J.; Liu, Y. Scalable place recognition under appearance change for autonomous driving. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Korea, 27–28 October 2019; pp. 9319–9328.
20. Galceran, E.; Olson, E.; Eustice, R.M. Augmented vehicle tracking under occlusions for decision-making in autonomous driving. In Proceedings of the 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Hamburg, Germany, 28 September–3 October 2015; IEEE: Piscataway, NJ, USA, 2015; pp. 3559–3565.
21. Tscharn, R.; Latoschik, M.E.; Löffler, D.; Hurtienne, J. "Stop over there": Natural gesture and speech interaction for non-critical spontaneous intervention in autonomous driving. In Proceedings of the 19th ACM International Conference on Multimodal Interaction, Glasgow, UK, 13–17 November 2017; pp. 91–100.
22. Luo, J.; Huang, Y.S.; Weng, Y.S. Design of variable traffic light control systems for preventing two-way grid network traffic jams using timed Petri nets. *IEEE Trans. Intell. Transp. Syst.* **2019**, *21*, 3117–3127. [CrossRef]
23. Zeng, W.L.; Wu, M.M.; Sun, W.J. Comprehensive review of autonomous taxi dispatching systems. *Comput. Sci.* **2020**, *47*, 181–189.
24. Tang, J.; Liu, S.; Wang, C.; Wang, Q. Distributed simulation platform for autonomous driving. In Proceedings of the International Conference on Internet of Vehicles, Kanazawa, Japan, 22–25 November 2017; Springer: Cham, Switzerland, 2017; pp. 190–200.
25. Zhang, Y.; Liu, J.; Sun, J.; Chen, X. Intelligent-prediction model of safety-risk for CBTC system by deep neural network. In Proceedings of the International Conference on Collaborative Computing: Networking, Applications and Work sharing, London, UK, 19–22 August 2019; Springer: Cham, Switzerland, 2019; pp. 669–680.

26. Kinoshita, S.; Nishimura, H.; Yun, S.; Kitamura, N. Introduction of driver's delay into "model checking" for verification of safe interactions between a driver and an automated driving system. In Proceedings of the 2016 IEEE International Symposium on Systems Engineering (ISSE), Edinburgh, UK, 3–5 October 2016; IEEE: Piscataway, NJ, USA, 2016; pp. 1–7.
27. Claussmann, L.; Revilloud, M.; Gruyer, D.; Glaser, S. A review of motion planning for highway autonomous driving. *IEEE Trans. Intell. Transp. Syst.* **2019**, *21*, 1826–1848. [CrossRef]
28. Delling, D.; Sanders, P.; Schultes, D.; Wagner, D. Engineering route planning algorithms. In *Algorithmics of Large and Complex Networks*; Springer: Berlin, Germany, 2009; pp. 117–139.
29. Lefèvre, S.; Vasquez, D.; Laugier, C. A survey on motion prediction and risk assessment for intelligent vehicles. *ROBOMECH J.* **2014**, *1*, 1–14. [CrossRef]
30. Claussmann, L.; Carvalho, A.; Schildbach, G. A path planner for autonomous driving on highways using a human mimicry approach with binary decision diagrams. In Proceedings of the 2015 European Control Conference (ECC), Linz, Austria, 15–17 July 2015; IEEE: Piscataway, NJ, USA, 2015; pp. 2976–2982.
31. Li, X.; Sun, Z.; Cao, D.; Liu, D.; He, H. Development of a new integrated local trajectory planning and tracking control framework for autonomous ground vehicles. *Mech. Syst. Signal Processing* **2017**, *87*, 118–137. [CrossRef]
32. Fujiyoshi, H.; Hirakawa, T.; Yamashita, T. Deep learning-based image recognition for autonomous driving. *IATSS Res.* **2019**, *43*, 244–252. [CrossRef]
33. Lugaresi, G.; Alba, V.V.; Matta, A. Lab-scale models of manufacturing systems for testing real-time simulation and production control technologies. *J. Manuf. Syst.* **2021**, *58*, 93–108. [CrossRef]
34. Stewart, D.; Liu, J.J.; Cofer, D.; Heimdahl, M.; Whalen, M.W.; Peterson, M. AADL-Based safety analysis using formal methods applied to aircraft digital systems. *Reliab. Eng. Syst. Saf.* **2021**, *213*, 107649. [CrossRef]