Master’s Thesis

Systematic Fault Injection Scenario Generation for the Safety Monitoring of the Autonomous Vehicle

MIN U SHIN

Department of Mechanical Engineering

Ulsan National Institute of Science and Technology

2023

Systematic Fault Injection Scenario Generation for the Safety Monitoring of the Autonomous Vehicle

MIN U SHIN

Department of Mechanical Engineering

Ulsan National Institute of Science and Technology

ULSAN NATIONAL INSTITUTE OF
SCIENCE AND TECHNOLOGY

Systematic Fault Injection Scenario Generation for the Safety Monitoring of the Autonomous Vehicle

A thesis submitted to
Ulsan National Institute of Science and Technology
in partial fulfillment of the
requirements for the degree of
Master of Science

MIN U SHIN

12.16.2022 of submission
Approved by

Kwon

Advisor
Cheolhyeon Kwon

ULSAN NATIONAL INSTITUTE OF SCIENCE AND TECHNOLOGY

Systematic Fault Injection Scenario Generation for the Safety Monitoring of the Autonomous Vehicle

MIN U SHIN

This certifies that the thesis of Min U Shin is approved.

12.16.2022 of submission

Signature

Kwon

Advisor: Cheolhyeon Kwon

Signature

Hungsun Son

Signature

Hyondong Oh

Signature

Abstract

The Object and Event Detection and Response (OEDR) assessment of Automated Vehicles(AVs) must be thoroughly conducted on the entire Operational Design Domain(ODD) to prevent any potential safety risk caused by corner cases. In response to these challenges, AVs must be tested over hundreds of millions of kilometers before deployment to convince its OEDR capabilities. However, claiming safety through years of testing on the entire ODD is not practically sound. Therefore, many studies have addressed this problem, focusing on efficiently and effectively finding corner cases within high-fidelity simulation environment. In particular, one of key OEDR functionalities is a collision risk assessment system alarming the driver about an impending collision in advance. In AV ODD context, the collision risk assessment is confronting challenging situations such as incorrect sensor information and unexpected algorithmic errors derived from uncertain environments (weather, traffic flow, road conditions, obstacles). Whereas the widely employed collision risk assessment methods relies on the first principle, e.g., Time-To-Collision (TTC), the aforementioned situations cannot be properly assessed without appropriate scene understanding toward the each situation. To this end, AI-based research that leverages previous experience and sensor information (especially camera image) to assess collision risk through visual cues has been developed in recent years. Inspired by the above research trends, this paper aims to develop: 1) systematic corner case generation using a scenario-based falsification simulation; and 2) an AI-based safety monitoring system applicable in complex driving scenarios. The implemented simulation is shown to competently find the corner case scenarios, through which the developed system is validated that it can be used as an alternative to an existing collision risk indicator in complex AV driving scenarios.

Contents

List of Figures

I Introduction

In recent years, with the evolution of autonomous driving technology, cars are getting closer to fully autonomous driving. The demand for autonomous driving technology is increasing worldwide, and as a result, securing the reliability of various autonomous driving core technologies is essential for the commercialization of safe AVs [1]. To this end, a worldwide consortium (e.g., ISO, UNECE [2]) is developing policies for dependable AVs.

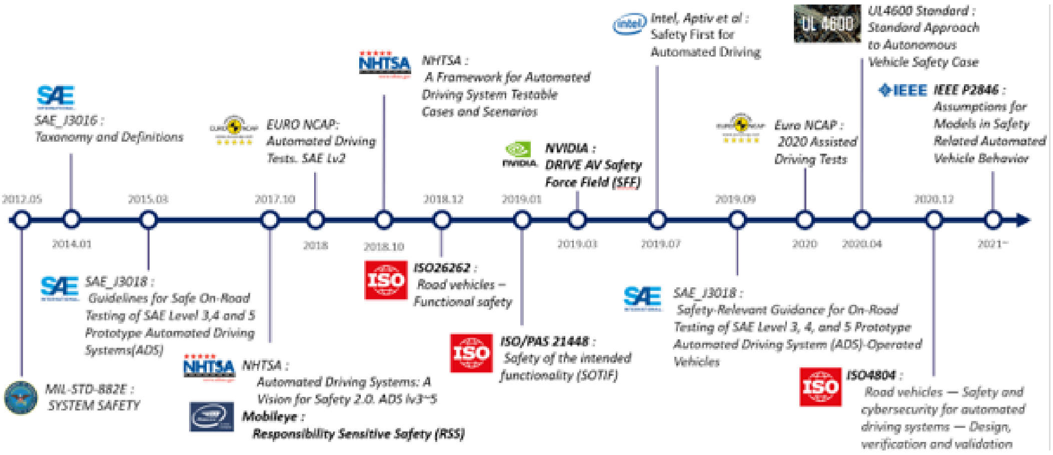


Figure 1: A national consortium is being formed for safe autonomous driving

In practice, however, because the development of the truly perfect technology is infeasible [3], critical-scenario can occur during testing or operation. The critical-scenario, so-called corner cases, represents the safety violation due to the unforeseeable situation or malfunctions in the AVs [4]. It is challenging to ensure that safe behavior can be guaranteed when AVs are deployed in real life due to the widespread application of AI and ML technologies [5], [6], [7], [8], [9]. In addition, environmental uncertainties such as weather and road conditions and hardware faults can have a significant impact on the safety of autonomous systems [10], [11], [12]. Reducing these unforeseen and unsafe areas is the challenge confronting autonomous driving technology today (see Figure. 2). In response to these challenges, AVs must drive hundreds of millions of kilometers without failure to convince the reliability of the AV's OEDR capabilities in terms of fatality and injury [3]. It is apparent that claiming safety through years of testing on the entire ODD is clearly inefficient. Also, even if the corner case occurs during on-road testing, the testing results are challenging to reproduce(e.g., sensor noise) resulting from the previous problem. Therefore, many studies have addressed this problem, focusing on efficiently and effectively finding corner cases within the simulation environment [13], [14], [15], [16], [17], [18]. To find corner cases in simulations, it is important to create scenarios that can happen in real life but are difficult to find based on a systematic scenario generation methodology [19], [20], [21], [22]. In particular, research is actively being conducted on how to automatically find corner case scenarios in infinite scenario generation parameter spaces [23], [24]. In addition, the fault injection technique recommended in ISO26262 [25] V&V process can be applied to finding extremely rare corner cases from the propagation of the internal fault of the AVs components [26], [27], [28], [29]. Especially, the safety claims can be forged by injecting system-level fault injection in AVs according to the safety analysis technique such as System-Theoretic Process Analysis(STPA) [30], [31]. Machine learning-based fault injection studies were also conducted to automate fault injection and efficient corner case exploration [32], [33], [34], [35], [36]. In this way, corner case testing to probe for weak spots that might be activated via unforeseen situations can be implemented more efficiently.

In accordance with the importance of finding the corner cases, one of the key OEDR functionalities of AVs is a safety monitoring system alarming the driver about an impending collision in advance. In AV ODD context, safety monitoring is confronting challenging situations such as incorrect sensor information and unexpected algorithmic errors derived from uncertain environments (weather, traffic flow, road conditions, obstacles) [37]. In order for AVs to drive reliably in all situations, it is essential to detect and decision-make these complex situations properly. Whereas the widely employed safety monitoring methods rely on the first principle, e.g., Time-To-Collision (TTC), the aforementioned situations cannot be properly assessed without appropriate scene understanding toward each situation. To this end, AI-based research that leverages previous experience and sensor information (especially camera image) to assess collision risk through visual cues has been developed in recent years. Therefore, in this paper, our interest is to use AI to monitor the safety of AVs even in complex situations.

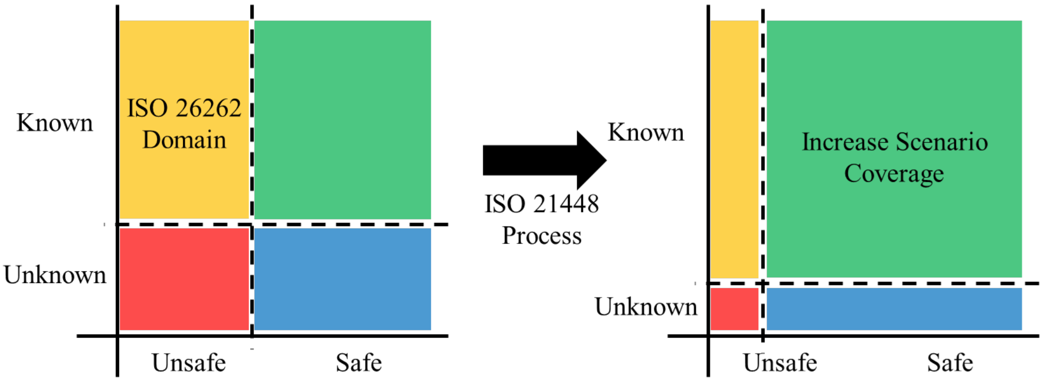


Figure 2: The purpose of scenario-based testing in simulation: minimize unsafe unknown areas by generating test cases.

1.1 Related Work

When driving in complex or uncertain situations, a person deliberately takes conservative action to ensure safety by judging the risk. In AI-based safety monitoring systems, several types of research have been conducted to predict impending collision or driveability in complex situations like humans.

In [38], the author presented an AI model to predict situations in which too challenging situations can lead to the failure of the OEDR function of AVs. To this end, they first learn a convolutional Long Short Term Memory(convLSTM) network that uses the real-world driving historical image sequence of the front camera and the steering and speed sequences to predict the current steering and speed values. Next, to predict the failure of the OEDR function of AVs, failure scores were defined based on the discrepancy between predicted maneuvers (steering and speed) and human driver maneuvers. According to this definition, the safe or hazardous situations were distinguished by threshold. Through this, they have developed a system that allows us to monitor the safety of AVs in real situations by learning models that predict failures from the present to specific sequences in the future.

The authors of [39] suggest an AI model that allows us to predict the degree of risk of the current scene through a threatening situation element to AVs. First, the pre-trained Mask R-CNN model creates a semantic segmentation mask in each video frame. The masked images are then fed into the convLSTM network model to learn to predict the risky action in the actual lane change situation. Ground truth for determining the risky action was subjectively assessed in each case by ten commentators watching the video clip.

Both of the above models can predict the risks posed by challenging situations that AVs may face. However, the problem with these models involves an individual's subjectivity to the ground truth because they use real-world driving data to predict risks. In addition, the real-world driving data is 1) difficult to handle all driving scenarios, 2) difficult to obtain collision data, and 3) cannot address algorithmic errors from components faults in the AVs.

On the other hand, in [40], the author obtained accident data from simulations to predict impending collisions. The data includes the front camera image, camera position, vehicle position, speed, acceleration, and command values. This data was used to learn the Bayesian convLSTM network. The advantage of this model is that simulations can be used to leverage real accident data as well as hard-to-face scenarios in the real-world. Nevertheless, they still do not address 1) systematic scenario generation to secure realistic scenarios and 2) algorithmic errors due to component failures that may pose a fatal risk in autonomous vehicles.

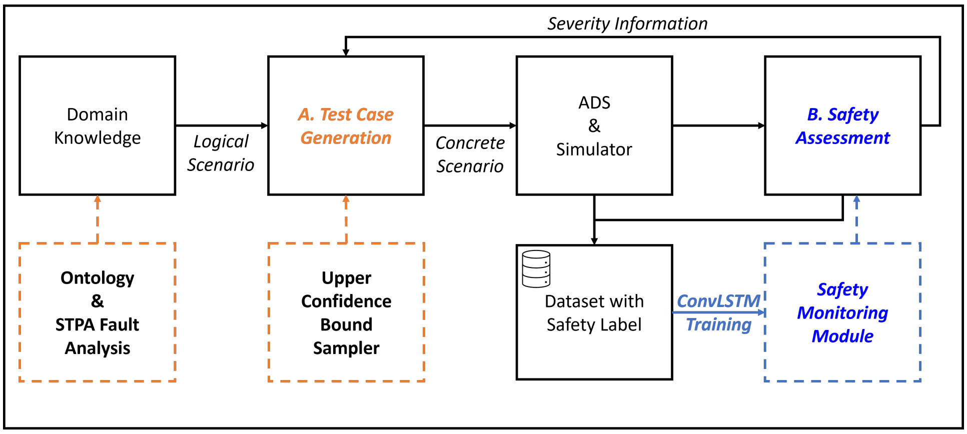


Figure 3: Overview of proposed main idea

1.2 Contribution

Inspired by the above research trends, this paper aims to develop:

1) systematic corner case generation using a scenario-based falsification simulation; and

2) an AI-based safety monitoring system applicable in complex driving scenarios.

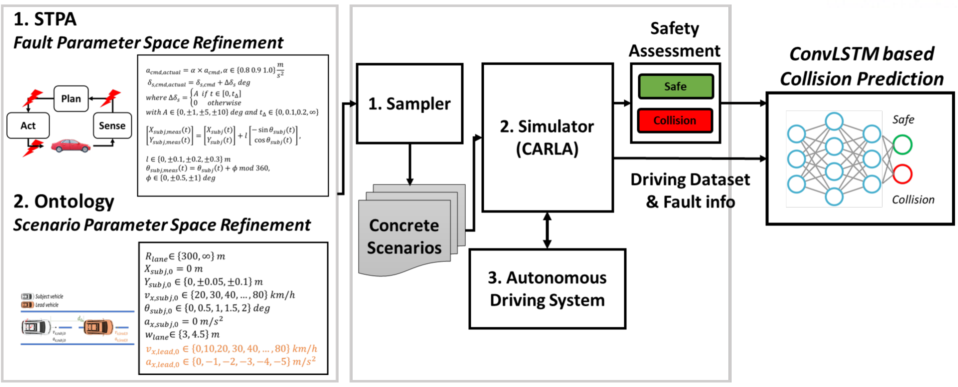

1. STPA
Fault Parameter Space Refinement
Plan
Act
Sense
$a_{cmd,actual} = \alpha \times a_{cmd}, \alpha \in \{0.8\ 0.9\ 1.0\} \frac{m}{s^2}$
$\delta_{s,cmd,actual} = \delta_{s,cmd} + \Delta\delta_s\ deg$
$where\ \Delta\delta_s = \begin{cases} A & if\ t \in [0, t_\Delta] \\ 0 & otherwise \end{cases}$
$with\ A \in \{0, \pm1, \pm5, \pm10\}\ deg\ and\ t_\Delta \in \{0, 0.1, 0.2, \infty\}$
$\begin{bmatrix} X_{subj,meas}(t) \\ Y_{subj,meas}(t) \end{bmatrix} = \begin{bmatrix} X_{subj}(t) \\ Y_{subj}(t) \end{bmatrix} + l \begin{bmatrix} -\sin\theta_{subj}(t) \\ \cos\theta_{subj}(t) \end{bmatrix},$
$l \in \{0, \pm0.1, \pm0.2, \pm0.3\}\ m$
$\theta_{subj,meas}(t) = \theta_{subj}(t) + \phi\ mod\ 360,$
$\phi \in \{0, \pm0.5, \pm1\}\ deg$
2. Ontology
Scenario Parameter Space Refinement
Subject vehicle
Lead vehicle
d_0
$v_{x,subj,0}$
$a_{x,subj,0}$
$v_{x,lead,0}$
$a_{x,lead,0}$
$R_{lane} \in \{300, \infty\}\ m$
$X_{subj,0} = 0\ m$
$Y_{subj,0} \in \{0, \pm0.05, \pm0.1\}\ m$
$v_{x,subj,0} \in \{20, 30, 40, \ldots, 80\}\ km/h$
$\theta_{subj,0} \in \{0, 0.5, 1, 1.5, 2\}\ deg$
$a_{x,subj,0} = 0\ m/s^2$
$w_{lane} \in \{3, 4.5\}\ m$
$v_{x,lead,0} \in \{0, 10, 20, 30, 40, \ldots, 80\}\ km/h$
$a_{x,lead,0} \in \{0, -1, -2, -3, -4, -5\}\ m/s^2$
1. Sampler
Concrete Scenarios
2. Simulator (CARLA)
3. Autonomous Driving System
Safety Assessment
Safe
Collision
Driving Dataset & Fault info
ConvLSTM based Collision Prediction
Safe
Collision

Figure 4: Specific structure of each module

To overcome the aforementioned problems, we define logical scenarios through the ontology method for systematic scenario generation and STPA-based signal-level falsification approaches to respond to various component faults. A Multi-Armed Bandit-based sampling technique is applied to efficiently explore corner cases in an infinite combination of logical scenario parameters. Concrete scenarios were generated through sampling, and front camera images, fault information, vehicle information (speed, yaw rate), and collision information were collected through open source-based autonomous driving system simulations. The collected data was used to train the convLSTM network to monitor future collisions after a specific time step. Simulations for various situations have been carried out to verify the effectiveness of the trained model against image-based models.

The remainder of this paper consists of the following sections. In Section 2, scenario-based test methodologies are described into two categories. Next, the methodology for generating concrete fault injection scenarios and the specific process for obtaining training datasets are described in Section 3. Next, the implementation of the online prediction system is described in section 4. Finally, the experimental results and conclusions are described in sections 5 and 6.

II Simulation-Assisted Training Dataset Generation

In this section, the process of dataset acquisition that is necessary for training AI algorithms is demonstrated. The following section gives a consecutive process of dataset generation.

2.1 STPA and Ontology-based Scenario Generation

System-Theoretic Process Analysis(STPA)

Systems-Theoretic Process Analysis(STPA) [31] is a safety analysis approach to identify the potential unsafe control actions that can cause a failure of AV by modeling the hierarchical control structure of the AVs. This approach can be applied to the fault injection test to improve the probability of generation of safety-critical scenarios and hazard coverage [41].

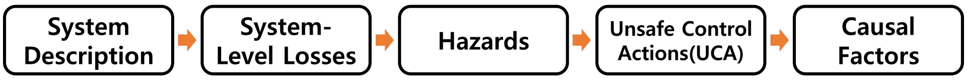


Figure 5: System-Theoretic Process Analysis(STPA) process

STPA fault analysis is defined in the order shown in Figure.5. First, complete the control structure (see Figure.6) by representing all system elements within the analysis scope and defining the interfaces between each element. Next, we define situations that are unintentional or undesired to occur that may lead to accidents, such as injury or property damage. In this paper, collisions between AVs and all other environments are considered accidents. The unsafe control actions(UCAs, see Figure.7) that can cause such collisions are then applied to the interfaces between all system elements according to the guide words to derive parameters for the fault injection test. Finally, the STPA analysis is completed by analyzing and identifying the causal factors(CFs) that trigger these UCAs.

Ontology

Ontology is known as a formal and explicit conceptualization of entities, interfaces, behaviors, and relationships. It has been applied to various applications such as decision-making, traffic description, autonomous driving, etc [42], [43]. Scenario-based AV testing can be performed systematically through domain ontology construction, test case creation, and test execution. Figure.8 shows the relationship between the elements that make up the ontology, and the following is about the concept of each element.

Scenario is a sequence of *Scenes* and usually contains static, dynamic obstacles around and its/their interactions between AVs(see Figure.9. *Ego vehicle* consists of specifically stated autonomous functions, sensors, and vehicles. Depending on the purpose of the test, the scope of the self-driving car can be limited. In this study, an autonomous driving system, including an open source based localization-perception-planning-control system was used for the universality of the study.

The scenario formalized according to the ontology methodology is represented as *Abstract scenario* according to ISO 34501, and an example is shown in the following figure. Based on the *Abstract sce-*

nario, all the elements that set up the scenario are parameterized to define a logical scenario represented by means, range, and distribution. Finally, a concrete scenario is created to represent exactly one specific scene to execute the simulation.

Sampling Method for Concrete Scenario Generation

Various sampling techniques can be used for efficient corner case exploration while automatically generating concrete scenarios. In this paper, concrete scenarios are automatically generated using the Upper Confidence Bound(UCB) algorithm, one of the multi-armed bandit problem strategies. The problem with multi-armed bandits is finding the most rewarding slot machines through several attempts on N slot machines with different rewards. The core of the multi-armed bandit problem is exploration and exploitation. For this reason, the UCB algorithm was used to select slot machines that showed good rewards through reasonable exploration rather than random exploration, and that could be the optimal

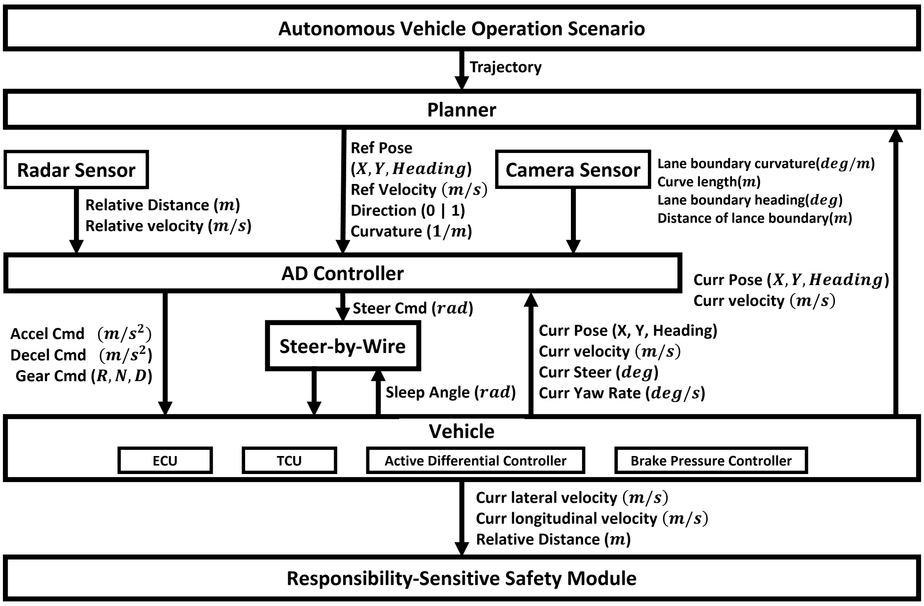


Figure 6: Example control structure

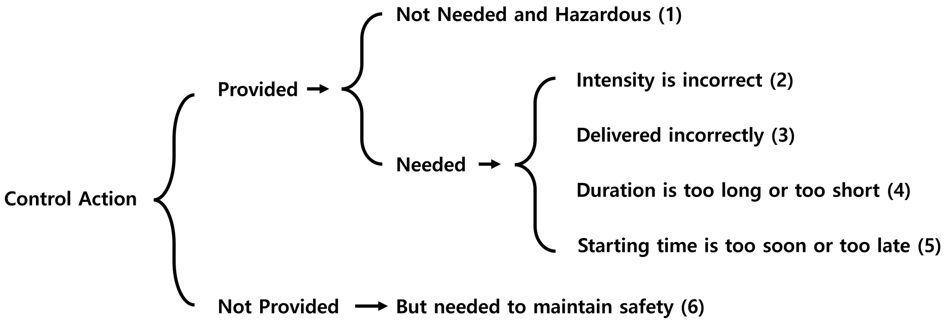

Control Action
Provided
Not Needed and Hazardous (1)
Needed
Intensity is incorrect (2)
Delivered incorrectly (3)
Duration is too long or too short (4)
Starting time is too soon or too late (5)
Not Provided
But needed to maintain safety (6)

Figure 7: Guide words for Unsafe Control Actions(UCAs)

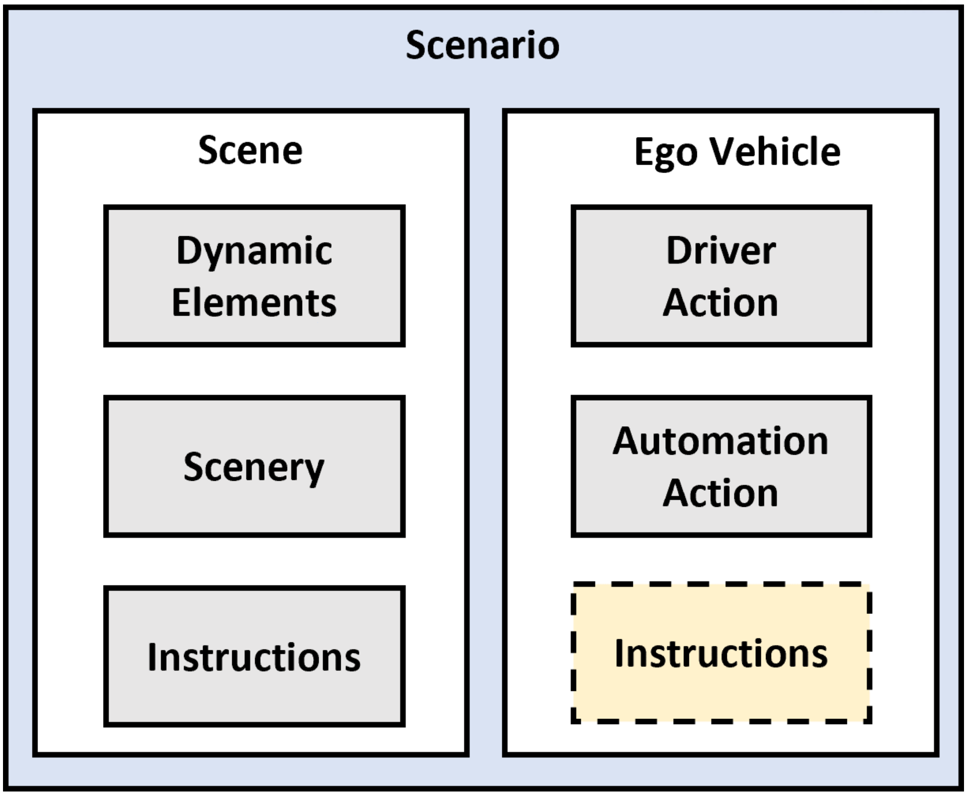


Figure 8: Fundamental ontology for AV guidance

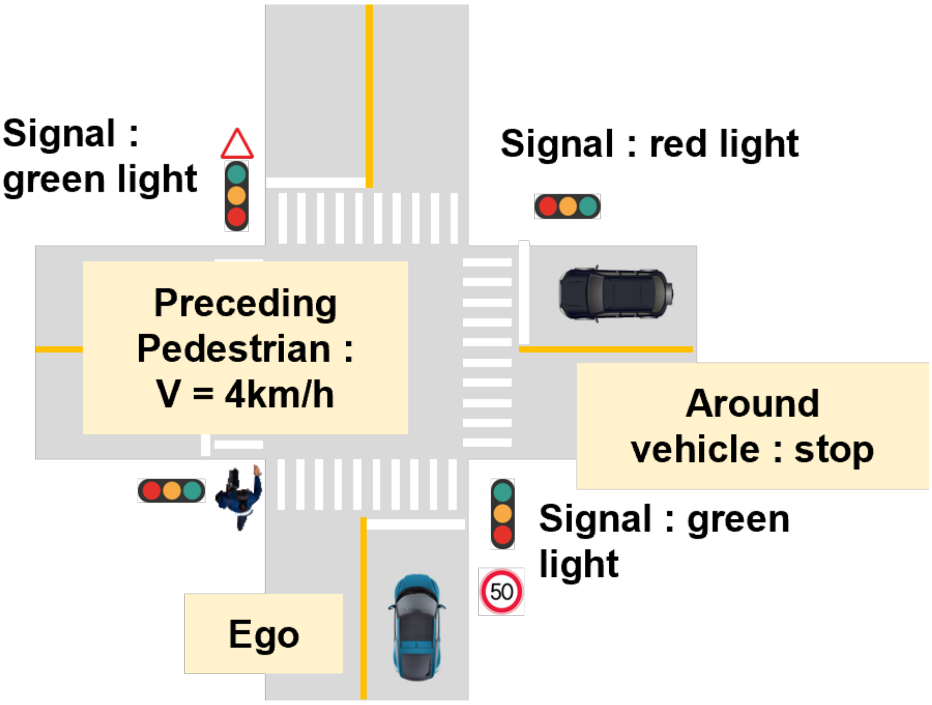


Figure 9: Example scene

choice. The UCB algorithm for selecting an action is represented by the following formula:

$$A_t \doteq argmax_a[Q_t(a) + c\sqrt{\frac{\log(t)}{N_t(a)}}] \tag{1}$$

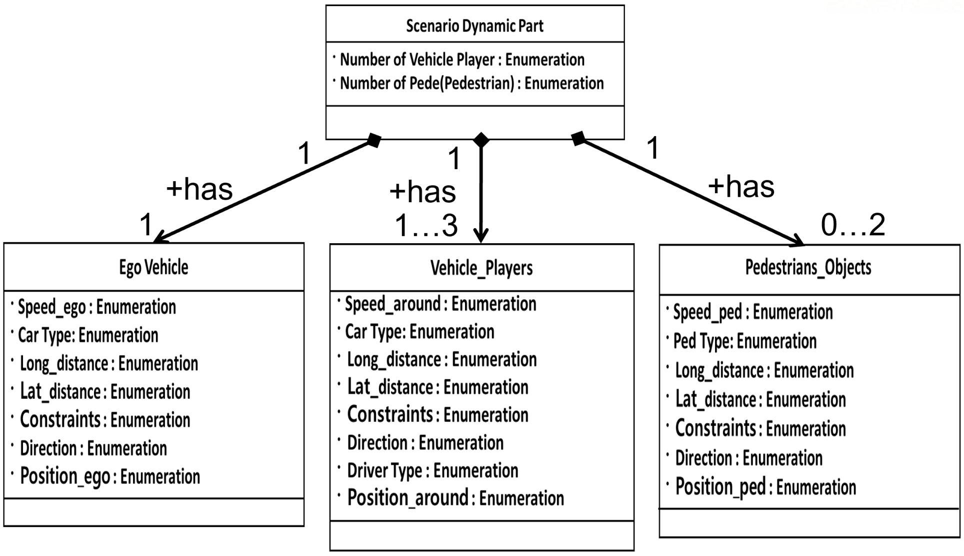

Scenario Dynamic Part
· Number of Vehicle Player : Enumeration
· Number of Pede(Pedestrian) : Enumeration
1
+has
1
1
+has
1…3
1
+has
0…2
Ego Vehicle
· Speed_ego : Enumeration
· Car Type: Enumeration
· Long_distance : Enumeration
· Lat_distance : Enumeration
· Constraints : Enumeration
· Direction : Enumeration
· Position_ego : Enumeration
Vehicle_Players
· Speed_around : Enumeration
· Car Type: Enumeration
· Long_distance : Enumeration
· Lat_distance : Enumeration
· Constraints : Enumeration
· Direction : Enumeration
· Driver Type : Enumeration
· Position_around : Enumeration
Pedestrians_Objects
· Speed_ped : Enumeration
· Ped Type: Enumeration
· Long_distance : Enumeration
· Lat_distance : Enumeration
· Constraints : Enumeration
· Direction : Enumeration
· Position_ped : Enumeration

Figure 10: Example of ontology-based scenario parametrization

In Formula 1, c is a hyperparameter that can adjust the degree of exploration. $N_t(a)$ is the number of times the slot machine was selected. where t is the sum of the number of times all slot machines have been selected, and $Q_t(a)$ is the average compensation value for the slot machine selection.

Specific scenario generation pseudo-code with UCB algorithm is as follows:

Algorithm 1 UCB-based concrete scenario generation

```
Initialize each logical scenario parameter with a specific resolution
Bandit Machines = $product$(logical scenario parameters)
for $iteration = 1, 2, \ldots$ do
    Initialize each bandit with 0 reward
end for
while $Simulation$ do
    for $iteration = 1, 2, \ldots$ do
        Estimates = [Equation.1 for each bandit]
    end for
    Concrete scenario = $Bandits[argmax(Estimates)]$
    Reward = [0 for safe, 1 for collision]
    Update reward
end while=0
```

2.2 Automated Driving System and Simulator

Building a scenario-based test process for AVs requires a general-purpose autonomous driving system and a simulator that can realistically mimic the surrounding environment, such as self-driving cars and sensors, and static and dynamic obstacles. Autoware [44] is a widely used open-source autonomous driving system that provides localization, perception, planning, and control systems that are core algorithms for AVs. The CARLA simulator [45] is also the most well-known open source-based AVs simulator that offers a wide range of sensor models, including RGB cameras, LiDAR, IMU, GPS, collisions, and lane invasion. Not only that, but it also offers a wide variety of maps, actors, car models, and traffic simulations, enabling simulations in a wide variety of environments. Because of these advantages, the Autoware system and CARLA simulator were used to perform scenario-based falsification tests.

Autoware

Localization: The localization algorithm implemented in Autoware is an NDT algorithm that leverages scan matching between a 3d map and a LiDAR scan. The NDT algorithm's computing costs are not dominated by map size, making it suitable for high-definition and high-resolution 3D maps.
Detection and Decision: A CNN-based detection algorithm was used to detect the surrounding environments, follow traffic rules, and avoid collisions. It also uses the nearest neighbors algorithm to cluster point clouds and calculate the euclidean distance between AVs and peripheral obstacles. Once the obstacles and traffic signals are detected, the mission planning and decision module use an intelligent state machine to determine the appropriate trajectory for AVs to travel.
Planning and Control: The planning module generated trajectories according to the output of the decision-making module, and in this paper, we used the global and local algorithms. Finally, a pure-pursuit algorithm was used to generate the actuation command according to the local trajectories for AVs.

CARLA simulator

The CARLA simulator provides RGB camera, LiDAR, IMU, radar, and GPS sensor models that can be used for signal-level fault injection simulations of AVs core components. Signal-level faults that can be injected into these sensor models can mimic various situations where noise, bias, lens flare strength, and so on can cause algorithmic errors. It also provides a variety of maps, including downtown roads and highways, making it easy to manage areas suitable for road models of ontology-derived logical scenarios. We implemented a core sensor fault and road model using the CARLA API and verified the results through simulation.

Sensor List and Fault Parameters	
Sensor Name	Fault Parameters
RGB Camera	Bloom intensity, Lens flare intensity, Blur amount
GNSS	Noise lat, lon Bias, Noise lat, lon stddev
IMU	Noise Accel Stddev x,y,z
LiDAR	Atmosphere attenuation rate, Dropoff general rate, Dropoff zero intensity, Noise stddev

Table 1: Various sensor fault example

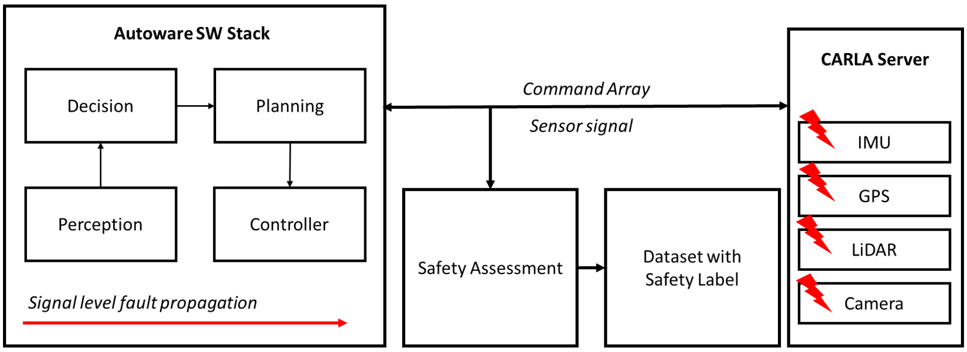


Figure 11: CARLA with Autoware system for simulation and data collecting

2.3 Generation of Training Dataset

Using the methodologies and tools described above, we created concrete scenarios and built a process to classify the simulation results according to the safety metric(e.g., CARLA collision sensor). NHTSA [46], WP29 documents were referenced to identify test scenarios, and parameter types and ranges of logical scenarios were defined based on ontology. The Upper Confidence Bound sampling technique creates concrete scenarios in logical scenarios, and the process of testing scenarios through simulators and classifying simulation results stores the data needed for learning. The generated data includes front camera images $V_{[t-k,t]}$ stored at 5 Hz, state values of AVs $S_{[t-k,t]}$(e.g., fault, velocity, yaw rate), and simulation results based on safety metrics(see Figure.13). The entire dataset was generated by 2000 simulations and identified 1000 safe scenarios and 1000 collision scenarios to balance the dataset. See Figure.12 for the overall process of data generation.

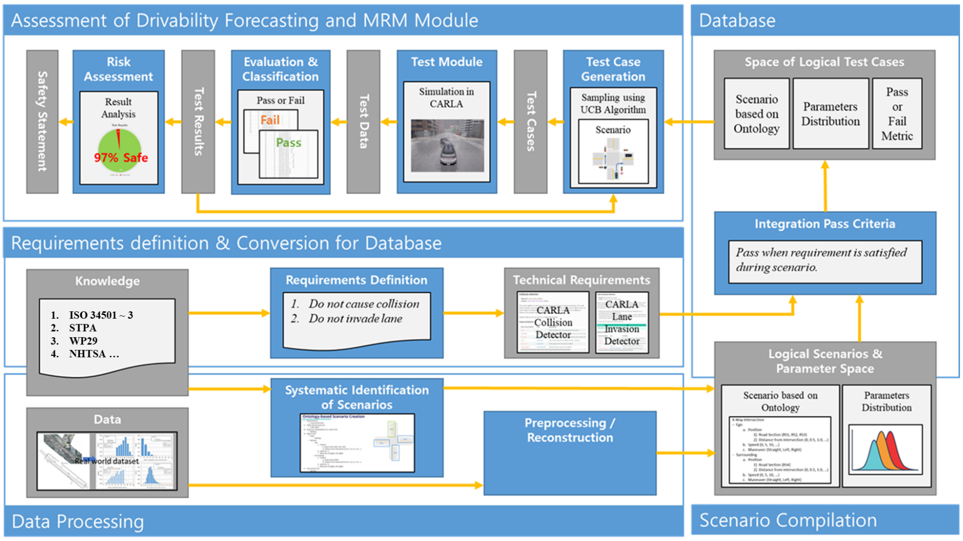

Assessment of Drivability Forecasting and MRM Module
Safety Statement
Risk Assessment
Result Analysis
97% Safe
Test Results
Evaluation & Classification
Pass or Fail
Fail
Pass
Test Data
Test Module
Simulation in CARLA
Test Cases
Test Case Generation
Sampling using UCB Algorithm
Scenario
Database
Space of Logical Test Cases
Scenario based on Ontology
Parameters Distribution
Pass or Fail Metric
Integration Pass Criteria
Pass when requirement is satisfied during scenario.
Requirements definition & Conversion for Database
Knowledge
1. ISO 34501 ~ 3
2. STPA
3. WP29
4. NHTSA ...
Requirements Definition
1. Do not cause collision
2. Do not invade lane
Technical Requirements
CARLA Collision Detector
CARLA Lane Invasion Detector
Data
Real world dataset
Systematic Identification of Scenarios
Preprocessing / Reconstruction
Logical Scenarios & Parameter Space
Scenario based on Ontology
Parameters Distribution
Data Processing
Scenario Compilation

Figure 12: Dataset generation Pipeline

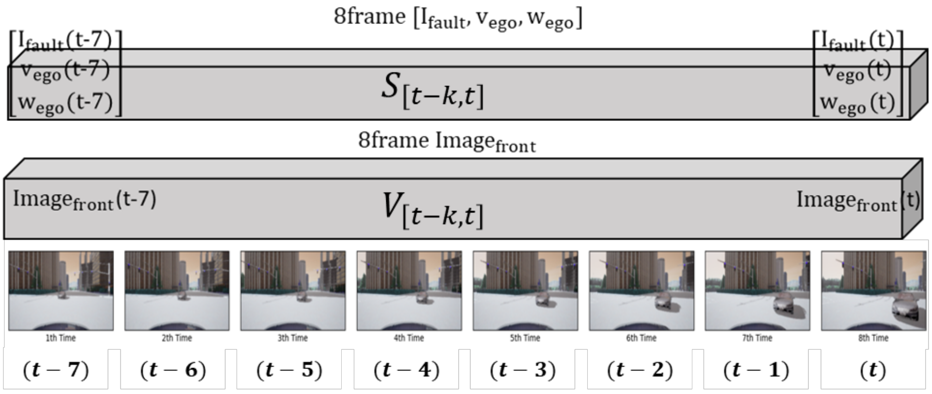


Figure 13: Training Dataset structure

III Formulation of AI-based Safety Monitoring System

Typically, in deep learning models, RNN networks have played an important role in modeling inputs and outputs that are related to time. In particular, the Convolutional Long-Short Term Memory(ConvLSTM) network is a type of RNN network that combines Convolutional Neural Network(CNN) with LSTM networks to provide proper characteristics for spatiotemporal data learning. Therefore, convLSTM was used to learn the safety monitoring system by the front camera image and state information obtained through the simulation. In this section, The process and components of the proposed AIFAF are described in detail. This section details the convLSTM model architecture for safety monitoring.

3.1 AI-based Safety Monitoring: An Overview

The goal of the safety monitoring system is to predict the safety information of AVs using $V_{[t-k,t]}$ and $S_{[t-k,t]}$ obtained through a systematic fault injection simulation. In the deep learning framework, this problem can be modeled by using spatiotemporal data to predict safety information after a specific time step. The convLSTM can store long-term input information in internal memory, which is proper for solving spatiotemporal dependency problems. The convLSTM network applies the same basic task to each input sequence in a phased process and switches the input information sequence to a single output. In conclusion, the problem we want to solve using convLSTM networks can be modeled as follows:

$$\begin{aligned}
&f(V_{[t-k,t]}, S_{[t-k,t]}) \rightarrow G_{[t+m]} \\
&V_{[t-k,t]} : \text{RGB camera image sequence} \\
&S_{[t-k,t]} : \text{Fault information, Current speed, Yaw rate} \\
&G_{[t+m]} : \text{Safety information after m time step}
\end{aligned} \tag{2}$$

3.2 Insight into the Convolution Long-Short Term Memory Network

An LSTM network is an RNN-series network that can store historical information. To address the problems modeled using sequence data, LSTM networks have proven to be ideal choices and stable networks to model long-term dependencies for learning complex dynamics [47]. LSTM's memory cells, which act like status information accumulators, are the core elements of the network. Cells are accessed, written, and cleared by several self-parameterized controlling gates. The key formula is shown in 3 where $'\circ'$ denotes the Hadamard product.

$$\begin{aligned}
&i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci} \circ c_{t-1} + b_i) \\
&f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf} \circ c_{t-1} + b_f) \\
&c_t = f_t \circ c_{t-1} + i_t \circ tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \\
&o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co} \circ c_t + b_o) \\
&h_t = o_t \circ tanh(c_t)
\end{aligned} \tag{3}$$

Modeling time information in this way is an advantage, but encoding space information over an LSTM network has lots of redundancy. The ConvLSTM network is used to overcome this problem and follows the formula described in [48] for the safety monitoring system problems. The spatiotemporal data is given by the network's input tensor $X_1,...,X_t$, the cell output is $C_1,...,C_t$, the hidden state $H_1,...,H_t$, and the gates are input i_t, forgotf_t, and the output o_t gate of the convLSTM cell is shown in Figure.14. The input to convLSTM is a 3D tensor whose last two dimensions are spatial feature dimensions. The input gate determines whether to include information about the memory cell and the forget gate f_t plays a role in removing information in the cell state. The output gate o_t is responsible for transferring information from C_t to the following hidden states H_t. ConvLSTM determines the extraction of features from current and past states through convolution operations. The key formula [48] for convLSTM is shown in (eq.5) below, where $'*'$ denotes the convolution operator and $'\circ'$, as before, denotes the Hadamard product:

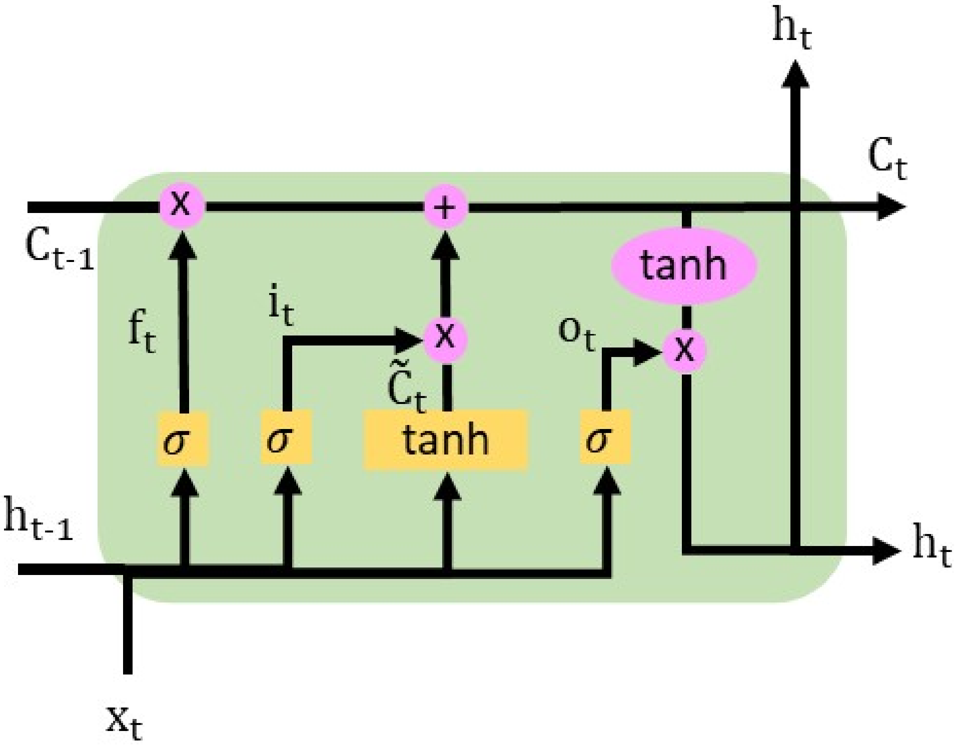


Figure 14: A general form of a single LSTM cell

$$
\begin{aligned}
& i_t = \sigma(W_{xi} * X_t + W_{hi} * H_{t-1} + W_{ci} \circ C_{t-1} + b_i) \\
& f_t = \sigma(W_{xf} * X_t + W_{hf} * H_{t-1} + W_{cf} \circ C_{t-1} + b_f) \\
& C_t = f_t \circ C_{t-1} + i_t \circ tanh(W_{xc} * X_t + W_{hc} * H_{t-1} + b_c) \\
& o_t = \sigma(W_{xo} * X_t + W_{ho} * H_{t-1} + W_{co} \circ C_t + b_o) \\
& H_t = o_t \circ tanh(C_t)
\end{aligned}
\tag{4}
$$

3.3 The Proposed System Architecture

A key characteristic of a safety monitoring system is to learn the relevance of space-time data through continuous camera scene information and state information and monitor safety online with high accuracy and appropriate sampling rates. The architecture of the proposed model to monitor safety through risk prediction is shown in Figure.15. The model has a structure in which convolution cells for learning images and LSTM cells are stacked, and a structure in which Dense cells and LSTM cells are stacked and these two structures are separated. The features embedded by the architecture of the separated structure were fed into a fully connected structure network and modeled to predict safe or collision. Among the input data sets, image sequences are supplied through the convolution layer of each convLSTM cell to extract spatial features for classifying the safety state of AVs. A layer of encoder structure was used to compress image information to generate meaningful feature vectors. The encoder processes repeated inputs through the laminated convLSTM layer and output an embedded tensor, the full sequence of hidden states, in all encoder convLSTM cells that indicate scene propagation. The input data set fault, and state information sequence is fed to a fully connected LSTM cell. The last fully connected network has a tensor built in through each encoder as an input sequence and outputs the predicted safety through the classification layer. The label used to learn the network is the safety information five steps from the present for AVs, which can be adjusted appropriately for the use case.

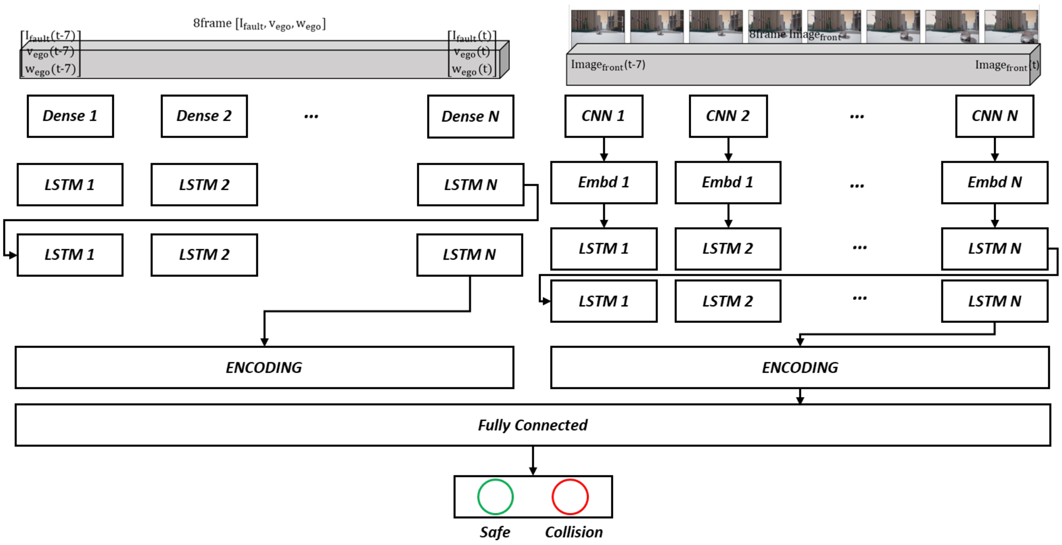


Figure 15: A schematic of a data-driven simulation-assisted-severity learned AI model for modeling course severity prediction

IV Implementation of the Proposed System for Safety Monitoring

In this section, previous experience and sensor information (especially camera image) is taught to the safety monitoring system while training the network to predict the collision risk as a binary classification. The test dataset and simulation environment quantitatively analyze the performance of the safety monitoring system. We also present the difference in predictive performance in the inclusion of fault information by comparison with baseline models.

4.1 Training the Proposed System Using Simulation-assisted Dataset

The proposed safety monitoring system is trained using simulation-aided training sequences generated by CARLA-Autoware and domain knowledge-based sampling in Section 2. These training datasets are a set of images and state values obtained from the simulation with a sampling period of 0.2 seconds per frame during the total simulation time. Each simulation set consists of data of different frame lengths according to different simulation times, of which the remaining data is removed from the events of interest except for 17 past frames. This ensures that all simulation data sequences have 17 constant sizes and that eight frames of sequential time size are selected in chronological order. Of a total of 2000 simulated data sets, 70% were used as training data, 20% as validation data, and 10% as test data. A 140x210 size RGB input image was used to train the network. Image pixel values [0, 255] are normalized to [0, 1] for fast network convergence. The training dataset also provides scalar value vehicle state and fault information to the network along with an input image sequence. The system (see Figure.15) is implemented in the TensorFlow of the Python platform, an open-source framework, to train the network through multiple GPUs. The model architecture consists of a convLSTM layer for image learning, an LSTM layer for scalar value learning, and a fully connected neural network layer for learning embedded tensors. Each layer consists of a different kernel size and hidden units. The safety monitoring network is trained to minimize binary cross-entropy loss functions using the backpropagation through time(BPTT) algorithm. The binary cross-entropy loss function is as follows:

$$L(\hat{y},y) = -[ylog(\hat{y}+(1-y)log(1-\hat{y}] \tag{5}$$

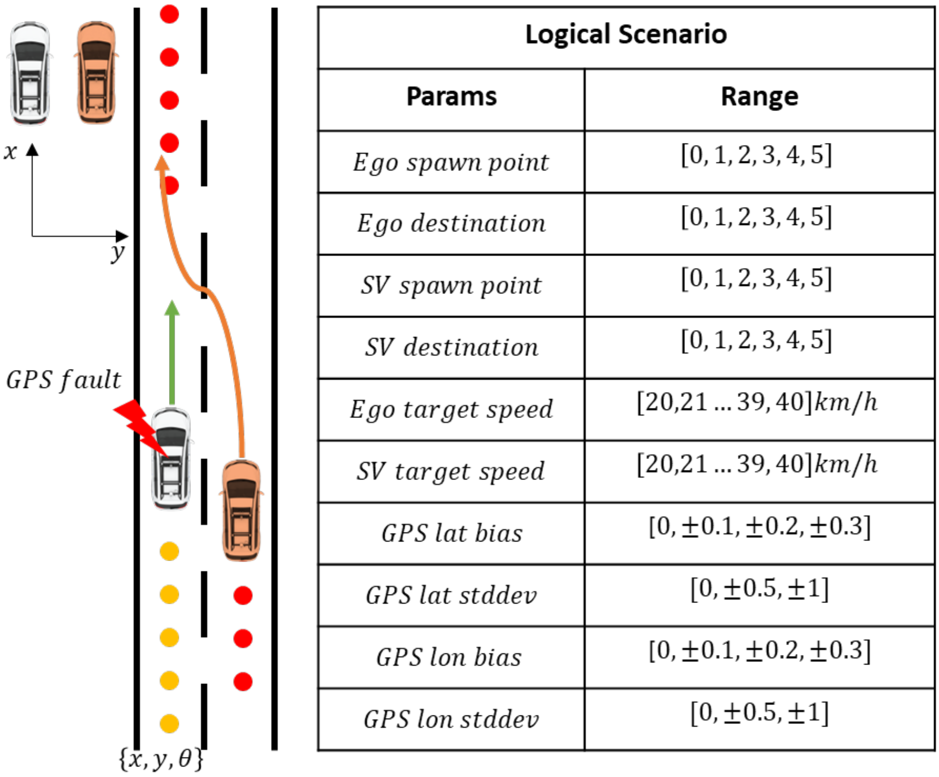
where is the ground truth safety value and y is the predicted safety value. The Adam optimizer is used to optimize weights and bias at different layers, and the hyperparameters required for learning are the same as the Table.2. Figure.17-19 shows the accuracy, loss, and area under the curve(AUC) values as a function of the number of iterations. Because loss and accuracy are stabilized over the number of iterations, the predicted results are determined to be similar to ground truth based on the specified learning rate. Training loss for 500 epoch iterations is 0.144. The NVIDIA GeForce RTX 1080Ti* 4 processor takes approximately 3 hours and 30 minutes to train the model.

4.2 Testing the Proposed System on Simulation-assisted Dataset

The trained risk prediction model is evaluated using a 10% test dataset to quantify performance. The model uses eight previous image frames and state information as input to generate one future risk pre-

Params	Value
CPU	Intel® Core™ i9-10900X CPU@ 3.70GHz
RAM	126 GB
GPU	GeForce RTX 2080 Ti x4
Total layer num	39
Input size	8x140x210x3
Output size	1x2
Total parameters	14,814,338
Total trainable parameters	14,811,202
Non-trainable parameters	3,136
Optimizer	Adam
Learning rate	1e-4
Decay rate	1e-5
Epochs	500
Mini-batch size	8
Training time	3h 29m 44s

Table 2: Parameters related to network training



Logical Scenario

Params	**Range**
$Ego\ spawn\ point$	$[0, 1, 2, 3, 4, 5]$
$Ego\ destination$	$[0, 1, 2, 3, 4, 5]$
$SV\ spawn\ point$	$[0, 1, 2, 3, 4, 5]$
$SV\ destination$	$[0, 1, 2, 3, 4, 5]$
$Ego\ target\ speed$	$[20,21 \ldots 39, 40] km/h$
$SV\ target\ speed$	$[20,21 \ldots 39, 40] km/h$
$GPS\ lat\ bias$	$[0, \pm 0.1, \pm 0.2, \pm 0.3]$
$GPS\ lat\ stddev$	$[0, \pm 0.5, \pm 1]$
$GPS\ lon\ bias$	$[0, \pm 0.1, \pm 0.2, \pm 0.3]$
$GPS\ lon\ stddev$	$[0, \pm 0.5, \pm 1]$

Figure 16: Training scenario example : Cut-in scenario with GPS faults

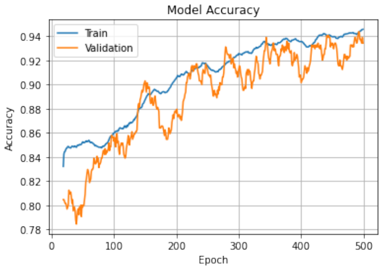


Figure 17: Training history of training accuracy

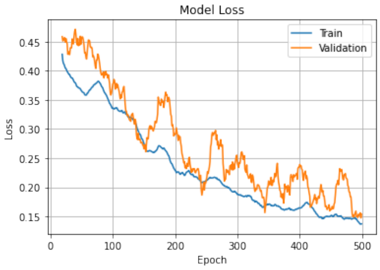


Figure 18: Training history of training loss

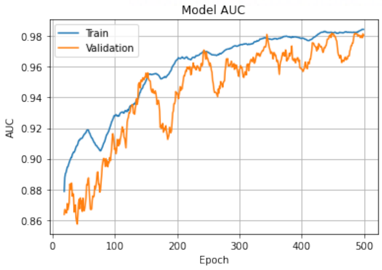


Figure 19: Training history of training AUC

diction value as output. For quantitative evaluation of the trained risk prediction model, scenarios such as GPS fault, detection failure due to camera faults, and localization failure due to LiDAR fault were used in various situations. The risk predicted by the safety monitoring system was compared to the actual simulation result value (taken as ground truth) stored for quantitative evaluation. Figure.16 shows one of the scenarios used in the test. Figure.20 shows the confusion matrix derived from the actual and the result values predicted using these sequences. Performance evaluated through a total of 4728 sequences showed 94% accuracy and 88% F1score. See Table.3 for more information.

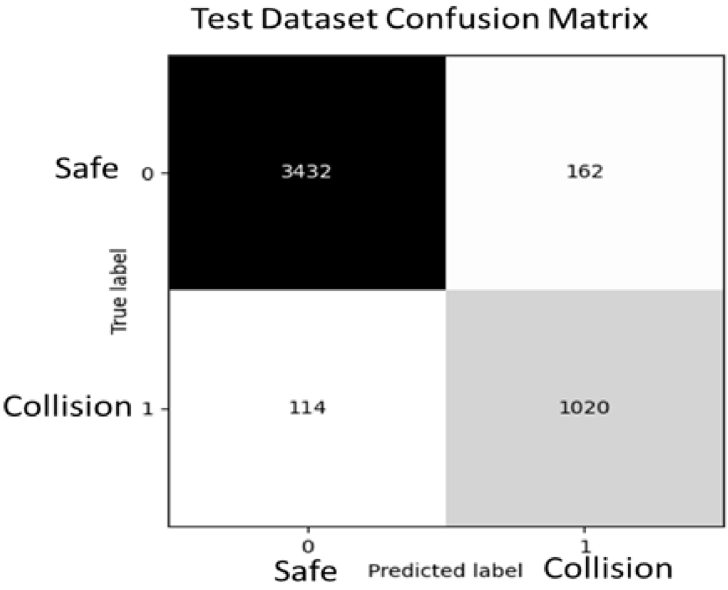


Figure 20: Analytical results for label and predicted values (confusion matrix)

4.3 Case Study the Proposed System on Simulation Environment

Scenarios were configured in the CARLA simulator to test learned networks online. The scenario set for this is a situation in which a surrounding vehicle around the right side attempts to change lanes to the lane of an AV in a two-lane situation. In this situation, quantitative tests were conducted to verify that the safety monitoring system works properly in a scenario where the forward vehicle detection fails due

Metric	Value
Sequence Num	4728
Accuracy	94.0%
Precision	86.0%
Recall	90.0%
F1score	88.0%

Table 3: Test dataset prediction result

to a sensor fault necessary for detection and so an accident occurs. The simulation confirmed that the learned network successfully predicts future accidents in online video streaming situations.

4.4 Comparative Analysis

Finally, we compared the networks we learned using images and state information to those using images only. Compared to the baseline model, the network using the status information of AVs showed higher performance during the training process. The training process is shown in Figure.21-23. As a result of measuring performance over the same test set for two trained networks, the network containing state information was 1.9 percentage points more accurate than the existing network. The architecture of the two networks used for the performance comparison is shown in Figure.24, and information about the performance comparison is shown in Table.4.

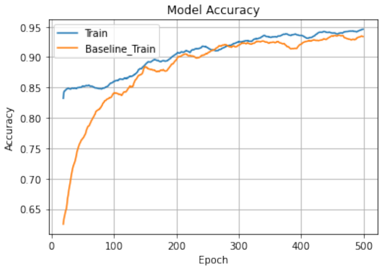


Figure 21: Comparison of training accuracy between baseline and our model

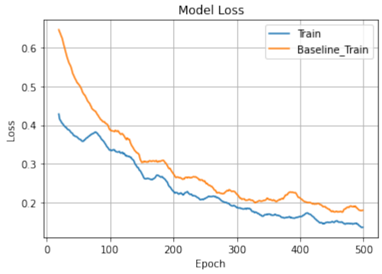


Figure 22: Comparison of training loss between baseline and our model

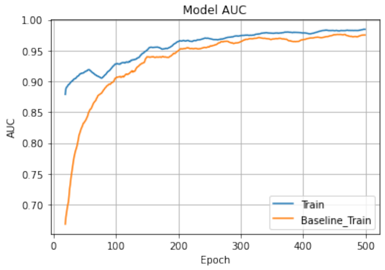


Figure 23: Comparison of training AUC between baseline and our model

	Baseline (only image)	Our model
Accuracy	92.7%	94.6%
Loss	0.209	0.144
AUC	97.0%	98.4%

Table 4: Comparative analysis of safety monitoring system performance

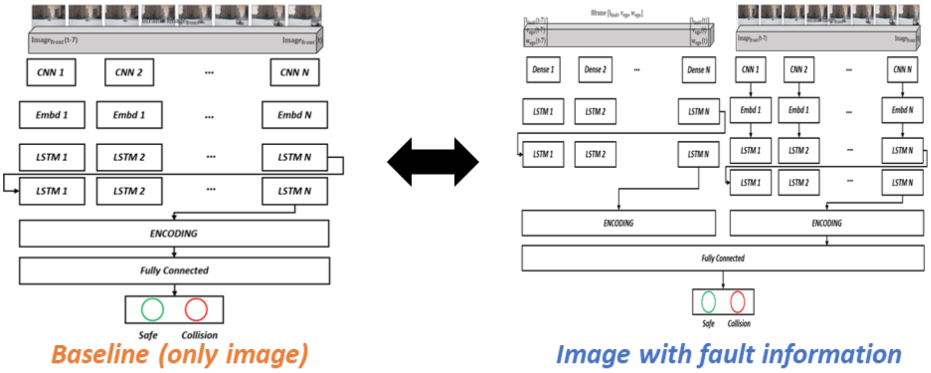


Figure 24: Baseline model architecture vs Our model architecture: The baseline model only fed into the images, but our model uses images and state information.

V Conclusion

In this study, simulation-aided datasets through systematic scenario generation were acquired, and AI was learned to monitor the safety of AVs in various sensor fault and algorithmic failure situations. The safety monitoring system was trained using datasets obtained through fault injection tests in a CARLA-Autoware environment. The trained safety monitoring system can predict future collisions with data up to the current point in time when an algorithmic error is caused by a sensor fault in an AV. A trained safety monitoring system can be deployed online to predict future collisions in complex situations, including failures that traditionally could not be resolved. The results obtained from the proposed safety monitoring system show higher performance than traditional image-based models and show that the risk of failure conditions can be effectively predicted.

References

[1] P. Koopman and M. Wagner, “Autonomous vehicle safety: An interdisciplinary challenge,” *IEEE Intelligent Transportation Systems Magazine*, vol. 9, no. 1, pp. 90–96, 2017.

[2] W. UNECE, “29, new assessment/test method for automated driving (natm) master document, united nations, 2021b.”

[3] N. Kalra and S. M. Paddock, “Driving to safety: How many miles of driving would it take to demonstrate autonomous vehicle reliability?” *Transportation Research Part A: Policy and Practice*, vol. 94, pp. 182–193, 2016.

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

[5] N. E. Boudette, “Tesla’s autopilot technology faces fresh scrutiny,” *The New York Times*, vol. 23, 2021.

[6] J. Plungis, “"what uber’s fatal self-driving crash can teach industry and regulators",” Nov. 2019.

[7] G. Motors, “Self-driving safety report,” *General Motors, Detroit*, 2018.

[8] D. J. Fremont, A. L. Sangiovanni-Vincentelli, and S. A. Seshia, “Safety in autonomous driving: Can tools offer guarantees?” in *2021 58th ACM/IEEE Design Automation Conference (DAC)*. IEEE, 2021, pp. 1311–1314.

[9] M. Wagner and P. Koopman, “A philosophy for developing trust in self-driving cars,” in *Road Vehicle Automation 2*. Springer, 2015, pp. 163–171.

[10] F. Heidecker, J. Breitenstein, K. Rösch, J. Löhdefink, M. Bieshaar, C. Stiller, T. Fingscheidt, and B. Sick, “An application-driven conceptualization of corner cases for perception in highly automated driving,” in *2021 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2021, pp. 644–651.

[11] J. Breitenstein, J.-A. Termöhlen, D. Lipinski, and T. Fingscheidt, “Systematization of corner cases for visual perception in automated driving,” in *2020 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2020, pp. 1257–1264.

[12] J.-A. Bolte, A. Bar, D. Lipinski, and T. Fingscheidt, “Towards corner case detection for autonomous driving,” in *2019 IEEE Intelligent vehicles symposium (IV)*. IEEE, 2019, pp. 438–445.

[13] S. Riedmaier, T. Ponn, D. Ludwig, B. Schick, and F. Diermeyer, "Survey on scenario-based safety assessment of automated vehicles," *IEEE access*, vol. 8, pp. 87 456–87 477, 2020.

[14] D. J. Fremont, T. Dreossi, S. Ghosh, X. Yue, A. L. Sangiovanni-Vincentelli, and S. A. Seshia, "Scenic: a language for scenario specification and scene generation," in *Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation*, 2019, pp. 63–78.

[15] K. Viswanadha, E. Kim, F. Indaheng, D. J. Fremont, and S. A. Seshia, "Parallel and multi-objective falsification with scenic and verifai," in *International Conference on Runtime Verification*. Springer, 2021, pp. 265–276.

[16] D. J. Fremont, E. Kim, Y. V. Pant, S. A. Seshia, A. Acharya, X. Bruso, P. Wells, S. Lemke, Q. Lu, and S. Mehta, "Formal scenario-based testing of autonomous vehicles: From simulation to the real world," in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2020, pp. 1–8.

[17] M. Saraoglu, A. Morozov, and K. Janschek, "Mobatsim: Model-based autonomous traffic simulation framework for fault-error-failure chain analysis," *IFAC-PapersOnLine*, vol. 52, no. 8, pp. 239–244, 2019.

[18] S. Ghosh, Y. V. Pant, H. Ravanbakhsh, and S. A. Seshia, "Counterexample-guided synthesis of perception models and control," in *2021 American Control Conference (ACC)*. IEEE, 2021, pp. 3447–3454.

[19] H. Nakamura, H. Muslim, R. Kato, S. Préfontaine-Watanabe, H. Nakamura, H. Kaneko, H. Imanaga, J. Antona-Makoshi, S. Kitajima, N. Uchida *et al.*, "Defining reasonably foreseeable parameter ranges using real-world traffic data for scenario-based safety assessment of automated vehicles," *IEEE Access*, vol. 10, pp. 37 743–37 760, 2022.

[20] S. Ulbrich, T. Menzel, A. Reschka, F. Schuldt, and M. Maurer, "Defining and substantiating the terms scene, situation, and scenario for automated driving," in *2015 IEEE 18th international conference on intelligent transportation systems*. IEEE, 2015, pp. 982–988.

[21] T. Menzel, G. Bagschik, and M. Maurer, "Scenarios for development, test and validation of automated vehicles," in *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2018, pp. 1821–1827.

[22] W. Ding, C. Xu, H. Lin, B. Li, and D. Zhao, "A survey on safety-critical scenario generation from methodological perspective," *arXiv preprint arXiv:2202.02215*, 2022.

[23] J. Duan, F. Gao, and Y. He, "Test scenario generation and optimization technology for intelligent driving systems," *IEEE Intelligent Transportation Systems Magazine*, vol. 14, no. 1, 2022.

[24] M. Althoff and S. Lutz, “Automatic generation of safety-critical test scenarios for collision avoidance of road vehicles,” in *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2018, pp. 1326–1333.

[25] ISO, “Road vehicles – Functional safety,” 2011.

[26] P. Koopman and M. Wagner, “Challenges in autonomous vehicle testing and validation,” *SAE International Journal of Transportation Safety*, vol. 4, no. 1, pp. 15–24, 2016.

[27] S. Jha, T. Tsai, S. Hari, M. Sullivan, Z. Kalbarczyk, S. W. Keckler, and R. K. Iyer, “Kayotee: A fault injection-based system to assess the safety and reliability of autonomous vehicles to faults and errors,” *arXiv preprint arXiv:1907.01024*, 2019.

[28] T. Goelles, B. Schlager, and S. Muckenhuber, “Fault detection, isolation, identification and recovery (fdiir) methods for automotive perception sensors including a detailed literature survey for lidar,” *Sensors*, vol. 20, no. 13, p. 3662, 2020.

[29] Y. Fu, A. Terechko, T. Bijlsma, P. J. Cuijpers, J. Redegeld, and A. O. Örs, “A retargetable fault injection framework for safety validation of autonomous vehicles,” in *2019 IEEE International Conference on Software Architecture Companion (ICSA-C)*. IEEE, 2019, pp. 69–76.

[30] A. Abdulkhaleq, D. Lammering, S. Wagner, J. Röder, N. Balbierer, L. Ramsauer, T. Raste, and H. Boehmert, “A systematic approach based on stpa for developing a dependable architecture for fully automated driving vehicles,” *Procedia Engineering*, vol. 179, pp. 41–51, 2017.

[31] N. G. Leveson, *Engineering a safer world: Systems thinking applied to safety*. The MIT Press, 2016.

[32] S. Jha, S. Banerjee, T. Tsai, S. K. Hari, M. B. Sullivan, Z. T. Kalbarczyk, S. W. Keckler, and R. K. Iyer, “Ml-based fault injection for autonomous vehicles: A case for bayesian fault injection,” in *2019 49th annual IEEE/IFIP international conference on dependable systems and networks (DSN)*. IEEE, 2019, pp. 112–124.

[33] S. Jha, S. S. Banerjee, J. Cyriac, Z. T. Kalbarczyk, and R. K. Iyer, “Avfi: Fault injection for autonomous vehicles,” in *2018 48th annual ieee/ifip international conference on dependable systems and networks workshops (dsn-w)*. IEEE, 2018, pp. 55–56.

[34] M. Moradi, B. J. Oakes, M. Saraoglu, A. Morozov, K. Janschek, and J. Denil, “Exploring fault parameter space using reinforcement learning-based fault injection,” in *2020 50th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN-W)*. IEEE, 2020, pp. 102–109.

[35] K. Viswanadha, F. Indaheng, J. Wong, E. Kim, E. Kalvan, Y. Pant, D. J. Fremont, and S. A. Seshia, “Addressing the ieee av test challenge with scenic and verifai,” in *2021 IEEE International Conference on Artificial Intelligence Testing (AITest)*. IEEE, 2021, pp. 136–142.

[36] M. O'Kelly, A. Sinha, H. Namkoong, R. Tedrake, and J. C. Duchi, "Scalable end-to-end autonomous vehicle testing via rare-event simulation," *Advances in neural information processing systems*, vol. 31, 2018.

[37] D. Muoio, "6 scenarios self-driving cars still can't handle," *Business Insider*, 6.

[38] S. Hecker, D. Dai, and L. Van Gool, "Failure prediction for autonomous driving," in *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2018, pp. 1792–1799.

[39] E. Yurtsever, Y. Liu, J. Lambert, C. Miyajima, E. Takeuchi, K. Takeda, and J. H. Hansen, "Risky action recognition in lane change video clips using deep spatiotemporal networks with segmentation mask transfer," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2019, pp. 3100–3107.

[40] M. Strickland, G. Fainekos, and H. B. Amor, "Deep predictive models for collision risk assessment in autonomous driving," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 4685–4692.

[41] A. H. M. Rubaiyat, Y. Qin, and H. Alemzadeh, "Experimental resilience assessment of an open-source driving agent," in *2018 IEEE 23rd Pacific rim international symposium on dependable computing (PRDC)*. IEEE, 2018, pp. 54–63.

[42] Y. Li, J. Tao, and F. Wotawa, "Ontology-based test generation for automated and autonomous driving functions," *Information and software technology*, vol. 117, p. 106200, 2020.

[43] G. Bagschik, T. Menzel, and M. Maurer, "Ontology based scene creation for the development of automated vehicles," in *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2018, pp. 1813–1820.

[44] S. Kato, S. Tokunaga, Y. Maruyama, S. Maeda, M. Hirabayashi, Y. Kitsukawa, A. Monrroy, T. Ando, Y. Fujii, and T. Azumi, "Autoware on board: Enabling autonomous vehicles with embedded systems," in *2018 ACM/IEEE 9th International Conference on Cyber-Physical Systems (ICCPS)*. IEEE, 2018, pp. 287–296.

[45] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," in *Conference on robot learning*. PMLR, 2017, pp. 1–16.

[46] E. Thorn, S. C. Kimmel, M. Chaka, B. A. Hamilton *et al.*, "A framework for automated driving system testable cases and scenarios," United States. Department of Transportation. National Highway Traffic Safety . . . , Tech. Rep., 2018.

[47] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural networks*, vol. 61, pp. 85–117, 2015.

[48] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo, “Convolutional lstm network: A machine learning approach for precipitation nowcasting,” *Advances in neural information processing systems*, vol. 28, 2015.

Acknowledgements

Two and a half years have passed since I started my internship in the laboratory, and my master's degree has ended before I know it. I met many people for a long time and shared difficult or fun moments, and I will write this with all my heart and tell them that I was able to finish my master's degree well.

First of all, I would like to thank Professor Cheolhyeon Kwon, who is my adviser. When the number of people in the laboratory was small, I started a relationship, and since I was an intern, they have always thought about the research direction that suits my field of interest and gave me many opportunities to have various experiences. Also, I would like to express my deep appreciation for the fact that I have been able to develop a lot through various aspects of teaching and that it has become a foundation for creating good opportunities in the future.

Also, despite your busy schedule, I would like to thank Professor Hungsun Son and Professor Hyondong Oh for their willingness to take charge of the master's degree defense, and I hope you stay healthy.

Next, thank you to the team members in the lab who have always shared every moment. I will say hello to Eunmin, Hyunchul, Hojung, Hojin, Youngim, and Hyunjun who helped me adapt to the laboratory since I was an intern. It was fun to be with Junseong, Ilseung, and Youngim, who spent every moment together from commuting to the laboratory to exercising and drinking. I want to tell you that Hyun Bin and Sanghyun who worked together on the research and project said that they did a great job. I want to say thank you to Sanghoon, Kwangrok, and Jusan who always give me fun.

And I'd like to tell all the team members and professors that the 2022 self-driving challenge was a really good time to remember, also it was a precious time to share the joys and sorrows with them.

Lastly, I would like to say thank you and love to my father, mother, and sister for supporting and supporting me so that I can focus only on my graduate life.

Acknowledgements

연구실 인턴 생활을 시작하고 벌써 2년 반이라는 시간이 흘러 어느덧 석사과정이 마무리되었습니다. 긴 시간동안 여러 사람을 만나 힘들거나 즐거웠던 순간을 함께했고, 진심을 다해 이 글을 쓰며 그들이 있어 석사 과정을 잘 마무리 할 수 있었다고 전합니다.

먼저, 저의 지도 교수님이신 권철현 교수님께 가장 먼저 감사를 드립니다. 연구실 인원이 몇 없던 때에 인연을 시작해 인턴시절부터 항상 제가 가진 관심분야에 맞는 연구방향을 고민해주셨고, 많은 기회 또한 부여해주셔서 다양한 경험을 할 수 있었습니다. 연구 외적으로도 교수님의 가르침을 통해 많은 발전을 할 수 있었고, 앞으로도 좋은 기회를 만들어 갈 수 있는 밑거름이 되었음에 깊은 감사 말씀을 올립니다.

또한, 바쁘신 일정에도 불구하고 흔쾌히 석사 학위 심사를 맡아주신 손흥선 교수님과 오현동 교수님께도 감사드리며 항상 건강하시길 바라겠습니다.

다음으로 항상 모든 순간을 함께했던 연구실 팀원들에게 감사를 표합니다. 인턴때부터 연구실 적응을 도와줬던 은민이, 형철이형, 호정이형, 호진이형, 영임이, 형준이형에게 가장 먼저 인사를 전합니다. 항상 연구실 출퇴근부터 운동, 술자리까지 모든 순간을 같이 하며 지냈던 준성이형, 일승이, 영임이에게 함께해서 즐거웠다 전하고 싶습니다. 함께 연구와 프로젝트를 수행한 현빈이, 상현이이게도 정말 고생했고 수고했다는 말을 전합니다. 항상 옆에서 즐거움을 주는 상훈이, 광록이, 주상이에게도 챙겨줘서 고맙다는 말을 전하고 싶습니다. 또한, 항상 열심히 연구하는 동화형, 강민이에게도 많은 것을 배울 수 있어 고마웠다는 말을 전하고 싶습니다.

그리고 2022 산업자원통상부 자율주행 대회는 정말 기억에서 잊지 못할 좋은 시간이었다고, 정말 아쉬운 결과였지만 너무나도 고생했고 동고동락 할 수 있어서 값진 시간이었다고 모든 팀원들 그리고 교수님께 전하고 싶습니다.

마지막으로, 자주 연락드리지 못하지만 대학원 생활에만 매진할 수 있도록 응원하고 지지해주신 아버지와 어머니, 그리고 누나에게 감사하고 사랑한다는 말을 전하고 싶습니다.