ARTICLE INFO
Article ID: 12-05-04-0025

doi:10.4271/12-05-04-0025

# Co-Simulation Platform for Modeling and Evaluating Connected and Automated Vehicles and Human Behavior in Mixed Traffic

***Xuanpeng Zhao,[1] Xishun Liao,[1] Ziran Wang,[2] Guoyuan Wu,[1] Matthew Barth,[1] Kyungtae Han,[2] and Prashant Tiwari[2]***

*[1]University of California at Riverside, Center for Environmental Research & Technology, USA*
*[2]Toyota Motor North America, InfoTech Labs, USA*

## Abstract

Modeling, prediction, and evaluation of personalized driving behaviors are crucial to emerging advanced driver-assistance systems (ADAS) that require a large amount of customized driving data. However, collecting such type of data from the real world could be very costly and sometimes unrealistic. To address this need, several high-definition game engine-based simulators have been developed. Furthermore, the computational load for cooperative automated driving systems (CADS) with a decent size may be much beyond the capability of a standalone (edge) computer. To address all these concerns, in this study we develop a co-simulation platform integrating Unity, Simulation of Urban MObility (SUMO), and Amazon Web Services (AWS), where Unity provides realistic driving experience and simulates on-board sensors; SUMO models realistic traffic dynamics; and AWS provides serverless cloud computing power and personalized data storage. To evaluate this platform, we select cooperative on-ramp merging in mixed traffic as a study case, and establish human-in-the-loop (HuiL) simulations. The results show that our proposed platform can facilitate data collection and performance assessment for modeling personalized behaviors and interactions in CADS under various traffic scenarios.

### History

Received: 17 Jun 2021
Revised: 22 Oct 2021
Accepted: 04 Apr 2022
e-Available: 21 Apr 2022

### Keywords

Co-simulation, Vehicle-to-Everything Communication, cooperative ramp merging, Amazon Web Services, Human-in-the-Loop

### Citation

Zhao, X., Liao, X., Wang, Z., Wu, G. et al., "Co-Simulation Platform for Modeling and Evaluating Connected and Automated Vehicles and Human Behavior in Mixed Traffic," *SAE Int. J. of CAV* 5(4):313–326, 2022, doi:10.4271/12-05-04-0025.

ISSN: 2574-0741
e-ISSN: 2574-075X

This article is part of a Special Issue on Emerging Simulation Tools and Technologies for Testing and Evaluating Connected and Automated Vehicles: Part 1.



# Introduction

Nowadays, driving behavior has various impacts on the driving assistant system [1]. Based on the personalized driving behavior model, advanced driver-assistant systems (ADAS) can give a better suggestion to the target human driver. To model the personalized driving behavior, a large amount of driving data is needed. However, collecting such customized and large datasets from the real-world may result in significant time and labor costs, and sometimes may be implausible. Therefore, seeking a cost-effective solution such as a well-designed simulation platform, has been receiving more and more attention.

To address the above needs, several game engine-based simulators, such as CARLA [2] and SVL [3], have been developed for evaluating autonomous driving system (ADS). Specifically, SVL is based on the Unity game engine, which provides sensor information and allows human interaction. Although game engine-based simulators can leverage their high-fidelity visualization and delicate modeling capability for on-board sensing, the dynamic traffic environment may not be realistic [4].

On the other hand, legacy vehicles and connected and automated vehicles (CAVs) are considered having to share the roads during the transition period, which may span several decades. The interactions between them may introduce new challenges to the transportation systems in terms of safety risks, efficiency degradation, and flow instability. To evaluate the performance and effects of ADS in mixed traffic, a realistic traffic environment model is required. Microscopic traffic simulators including VISSIM, Aimsun, and Simulation of Urban MObility (SUMO) can not only generate realistic traffic flows, but also provide well-calibrated car-following and lane-changing behaviors.

Furthermore, computational loads for high-fidelity modeling of CAVs may be overwhelming if only a single personal computer is used to handle the entire simulation scenario. To guarantee the system scalability, it is appropriate to leverage the power of "cloud" by integrating the co-simulation with Amazon Web Services (AWS), which provides a serverless cloud computing resource and enables personalized data storage.

The contribution of this article is listed as follows:

- An integrated co-simulation platform is set up to connect Unity, SUMO, and AWS, where vehicle models, traffic networks, and cloud computing are seamlessly integrated.
- Human-in-the-loop (HuiL) simulation is conducted with an immersive interface to provide high-fidelity interactions between a human-controlled vehicle and other background traffic.
- A scalable cloud-based platform (with AWS in this study) is developed, which is readily extended for synchronous or asynchronous multiplayer games, and personalized data storage and mining to enhance ADAS performance.

The remainder of this article is organized as follows: Section 2 presents background information including state-of-the-art driving simulators, microscopic traffic simulators, and co-simulators, followed by the introduction of system architecture and key components of the proposed integrated platform in Section 3. Section 4 demonstrates the capability of our co-simulation platform via human behavior-related research with a cooperative on-ramp merging ADAS for mixed traffic. Section 5 further discusses major challenges in the development of co-simulation platform, and the last section concludes this study and gives an outlook on the future.

# State of the Art

## Game Engine-Based Simulation

Game engines, such as Unity [5] and Unreal Engine 4 (UE4) [6], are computer software that consist of a graphics rendering engine, a physical engine, and an UI interface for managing elements. Based on these game engines, several open-source simulators are built for advanced autonomous driving implementations, such as SVL based on Unity [3] and CARLA based on UE4 [2]. These simulators normally provide 3D visualizations of vehicles and traffic network with high rendering quality and realistic physics. Moreover, the on-board sensors, such as radar, LiDAR, camera, and GPS, can be also fully customized. With the integration of these sensors, physical models of CAV can be built. Besides, the simulation environment can be well customized, such as roadside infrastructure, pedestrian, buildings, and extreme weather [3]. In the research community, game engine-based simulation environments were widely used to prototype vehicle cooperation [7, 8], model driver behavior [9, 10], and simulate autonomous driving [11].

## Microscopic Traffic Simulation

Currently, there are several microscopic simulators available to support the modeling and evaluation of cooperative automated driving systems (CADSs). They are able to provide realistic traffic environment after calibration with real-world data. Among them, PTV VISSIM [12], and Aimsun [13] are two commercial simulation tools. Specifically, VISSIM is a behavior-based multipurpose microscopic simulation that can be linked with MATLAB through a Component Object Model (COM) interface [12]. Aimsun [13] is a hybrid traffic modeling simulator, which allows simultaneous application of multimodel analysis with large networks. On the other hand, SUMO [14] is an open-source traffic simulator that can be used for a variety of applications, such as dynamic navigation, traffic surveillance systems evaluation, and traffic light algorithm development [15]. Furthermore, it is flexible enough to extend its capability by linking with other add-ons or simulators (e.g., OMNET++ for communication simulation) [15].

In addition, SUMO provides application programming interfaces (APIs), called Traffic Control Interface (TraCI), to facilitate the interaction with external applications through a socket (bidirectional) connection. With TraCI API, SUMO can be easily coupled with a network simulation, e.g., NS2 and NS3. Comparatively, Aimsun has a built-in vehicle communication module V2X SDK to enable network simulation [16].

## Advanced Co-Simulation

Currently, each individual simulator has its own advantages and focus arenas. However, a single simulator is not enough for modeling and evaluating CADS design as well as establishing the realistic testing environment. To leverage the capabilities of multiple simulators, some recent research integrated VISSIM with driving simulators to assess the influence of adverse weather on traffic flow characteristics [17]. In the meantime, SUMO was coupled with an open-source software framework, called CommonRoad [18], which can provide a benchmark for motion planning of automated vehicles. With the integration, motion planners can be evaluated and compared under realistic traffic environments provided by SUMO [19]. In the rest of this section, two types of co-simulation, i.e., HuiL and cloud-in-the-loop (CiL) will be introduced.

HuiL is a prototype platform for quickly exploring novel in-the-loop applications that can enhance the interactions between human beings and the physical world [20]. HuiL is widely used in different research topics highly related to human interaction with control systems. For example, in the work of rollover prevention for sport utility vehicles, the researchers validated the performance of the anti-rollover control via HuiL [21]. They emphasized that the reason for involving HuiL is that the driver's interaction with and perception of the system performance may cause a problem and many safety systems are designed without such consideration. With HuiL, some researchers proposed a synthetic approach to solving safety-critical interaction problems in the SAE Level 3 automated vehicles, which are mostly autonomous and only need limited driver intervention [22]. It is important to develop a safe and comfortable system via HuiL simulation. In previous study [23], researchers developed a microscopic driver-centric simulator by integrating Unity and SUMO. The simulator is based on the connection between two individual simulators through TraCI protocols. However, the information of the traffic flow is transferred only in one-way direction (i.e., from SUMO to Unity), which means that only the position of legacy vehicles can be updated in Unity. As a result, although the simulator allowed the human driver to react to the background traffic, the human driving behavior cannot affect the traffic flow. In the paper [24], researchers created a mixed reality simulation environment allowing real-world vehicles to interact with virtual traffic flow generated by SUMO and to be visualized in Unity. It collected the human driving behavior information with real-world vehicles' sensors to serve as a validation procedure for autonomous vehicle development. Despite this, it did not take full advantage of the game engine, because the mixed reality environment simulation ignored the virtual sensors, such as camera, radar, and LiDAR, which can be created in Unity via advanced sensor models.

CiL simulation leverages the powerful computing ability of a cluster of servers such as AWS for more efficient simulation and modeling purposes [25]. In a recent work [26], researchers developed a CiL simulation testbed using AWS and SUMO. The study took a speed advisory application as a test case and showed that the cloud infrastructure can provide an alternative for addressing the computing needs without sacrificing performance and reliability. However, the testbed neglected human factors, a key component for the entire system.

# Methods

## Co-Simulation Platform Architecture

The purpose of building the platform is to create a comprehensive simulator for modeling and evaluating the performance of human behavior with CADS under a mixed traffic scenario through the HuiL simulation. To achieve this purpose, there are three key components of the integrated platform: vehicle-human interfacing, traffic modeling, and cloud computing. To implement these components, we select a game engine, a microscopic simulator, and AWS as the respective solution, where (1) game engine is suitable for modeling and visualizing the surrounding environment of ego vehicle, and providing high-fidelity (on-board) sensor information; (2) microscopic simulator provides realistic traffic flow under various congestion levels and penetration rates of CAVs; and (3) AWS can store the data from each simulator, provide computation power for time-consuming algorithm, and archive personalized driving data for further investigation.

We select Unity as a solution for game engine due to its following advantages:

1. Visualization: Unity is capable of handling the trade-off between the realism of the vehicular dynamics and the computation load of the physics.
2. Inputs: Unity provides easy access to human interference device (HID), such as inputs from driving simulator hardware for human involving testing.
3. Asset store: Unity provides an Asset Store, which contains a library of assets created by the Unity Technologies and the community members. Wide-variety convenient assets are available to developers, such as vehicle dynamics models and waypoint systems.

As for the microscopic simulator, SUMO become our choice for these reasons:

1. SUMO provides APIs, called Traffic Control Interface (TraCI), to facilitate the interaction with external applications through a socket (bidirectional) connection.
2. SUMO also provides multiple solid microscopic traffic models, which are used and tested in various studies [9].
3. SUMO is an open-source traffic simulator, which provides well-organized and easy-to-read documentation.
4. SUMO provides an active community forum supported by Eclipse, which is good for researchers to exchange ideas and get inspired.

In terms of vehicle modeling, we define two different controller models that are CAVs and legacy vehicles. As illustrated in Figure 1, CAVs with the on-board sensors modeled by Unity are driven by the user-defined control algorithm, and legacy vehicles are controlled by the SUMO default car-following model and lane-changing model. In terms of traffic modeling, traffic demand can be defined in SUMO on an individual trip basis (e.g., specifying trip starting time and route) or in an aggregated manner (such as total traffic flow from origins) [27]. For cloud computing, AWS provides us enough storage space and the high-performance computation ability to support the training or running of advanced modeling algorithms, such as deep learning and reinforcement learning, for modeling personalized human behaviors.

To integrate all these components, we develop a python script called *Edge Gateway* to handle the data exchange and information synchronization. As shown in Figure 1, each component can share locally processed data with others via the *Edge Gateway*. The Edge Gateway uses asynchronous processes to avoid simulator blocking that may introduce frame per second (FPS) drops. There are several User Datagram Protocol (UDP) server and client threads handling the data exchange with queues. With the *Edge Gateway*, FPS performance is improved to provide a better human driver experience in Unity. In addition, the *Edge Gateway* also provides the possibility to extend the entire platform for incorporating other simulators or even real-world vehicles.

Since the connection and information synchronization is critical to the integrated platform, we detail the chronological sequence of each executed component. As shown in Figure 2, SUMO generates vehicles based on a predefined route file. Keeping the route file unchanged can reproduce the same traffic flow. With this feature, we can easily control the environment variables in different tests. For the first time when vehicles' statuses are shared with Unity, predefined 3D prefabs can be created based on vehicle types selected by the *vehicle control-type classifier*. Although vehicles are initially all governed by SUMO, Unity may take over the control once some (or even all) of them are assigned to be CAVs or the human control vehicle (HCV). After creating vehicles in both simulators and determining the type of each vehicle, legacy vehicles will be controlled based on SUMO models and their information will be shared in real-time to Unity and AWS

**FIGURE 1** The general architecture of the proposed Unity-SUMO-AWS co-simulation platform.

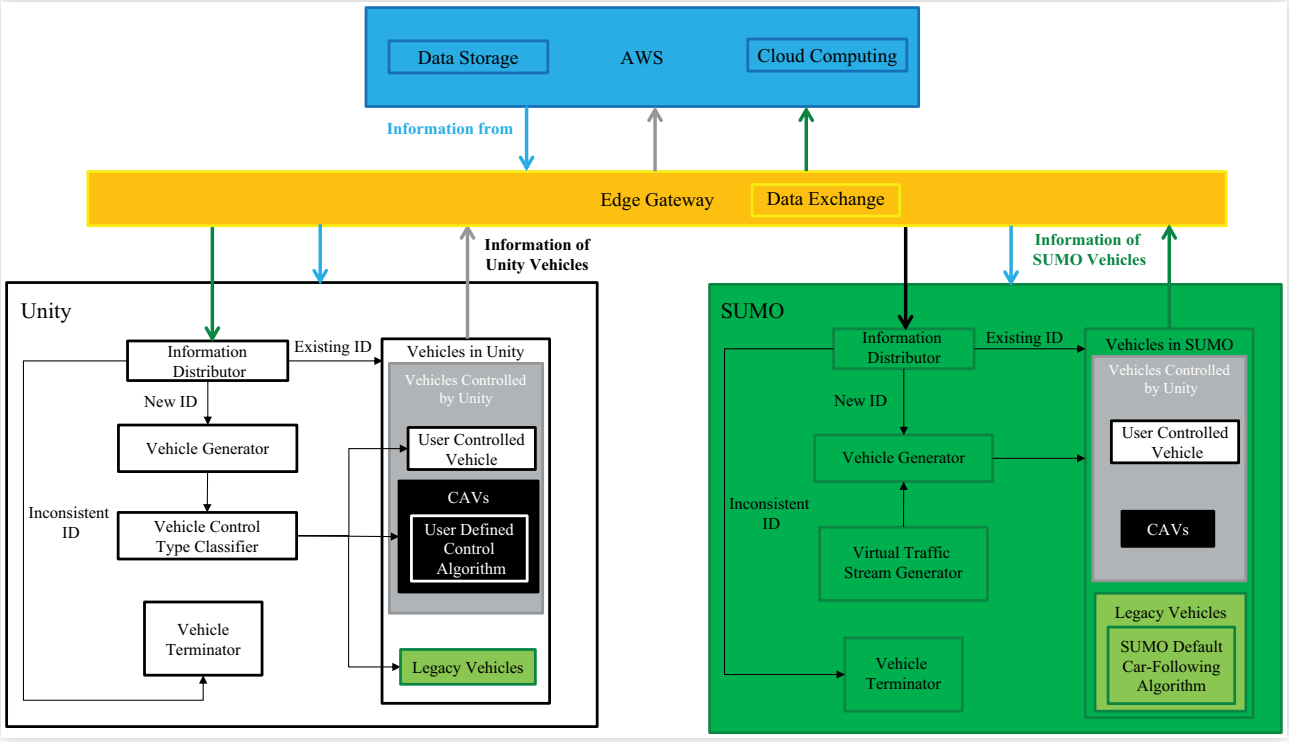
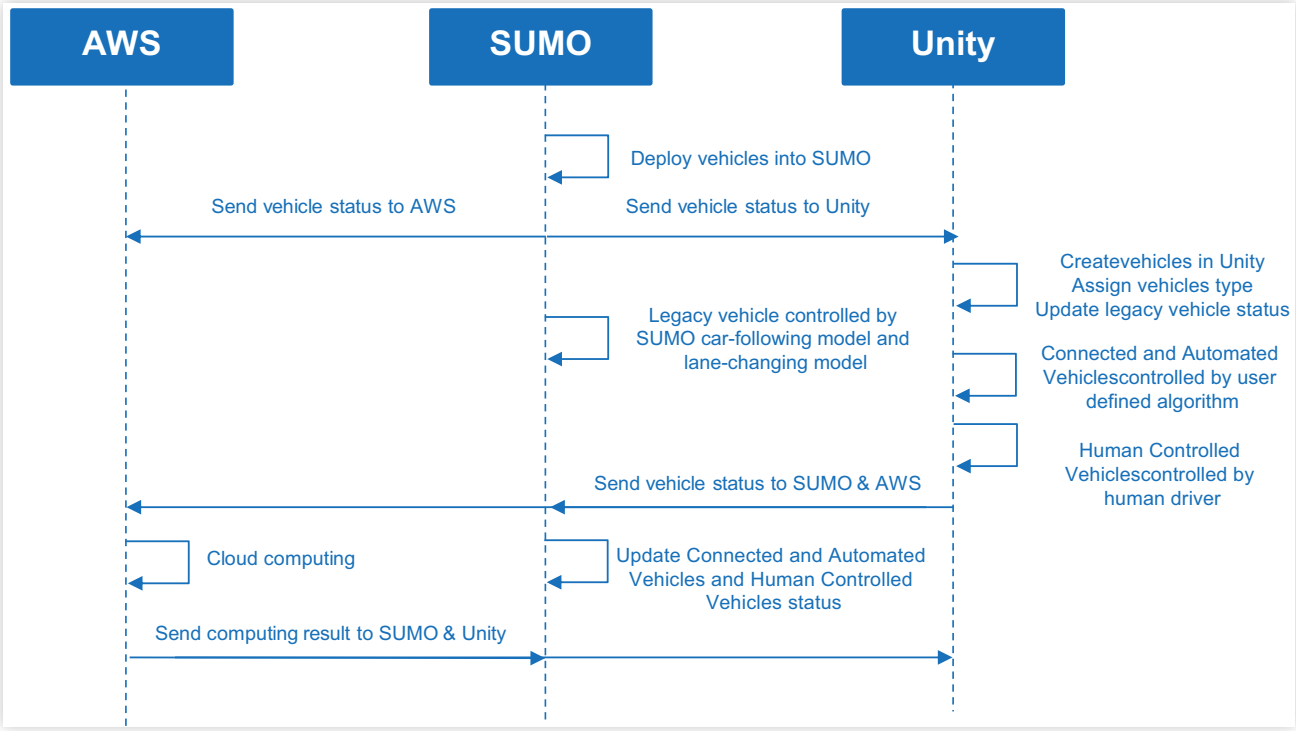


**FIGURE 2** Sequence diagram for information synchronization.






through the *Edge Gateway*. Then, Unity will update the corresponding vehicles' statuses. Likewise, the Unity-controlled vehicles, including HCV and CAVs, will share their states, such as velocities and positions to SUMO, to ensure statuses of the corresponding vehicles on different simulators are synchronized. In the meantime, AWS processes the information collected from both SUMO and Unity to provide the required services.

## Key Components within the System Architecture

**Vehicle Modeling** To create a realistic mixed traffic flow for CADS designing through the HuiL simulation, we define three general vehicle types participating in the simulation, which are legacy vehicle, CAV, and HCV. At the simulator level, they can also be classified into two types: Unity- and SUMO-controlled vehicles. Unity-controlled vehicles include CAVs and HCV.

Specifically, the legacy vehicle is totally controlled by SUMO via car-following and lane-changing model. SUMO can deploy these legacy vehicles based on a predefined route file and remove them when they reach the end of the trip. In terms of CAVs, they are controlled by user-defined algorithm coding in Unity. The control algorithm can be coded with intelligent driver model (IDM) [35] or other advanced (data-driven) algorithms. In addition, the on-board sensors (e.g., camera, radar, LiDAR, GPS) equipped on CAVs can provide realistic data formats such as point clouds and images.

As a key player in the HuiL simulation, we create the HCV model in Unity via Vehicle Control script to simulate high-fidelity vehicular dynamics, enabling the driver-vehicle interface. The script is defined in the *Vehicle Physics Pro* (VPP), which is an advanced vehicle simulation kit for Unity that supports efficient, realistic, and accurate vehicle simulation. Moreover, multiple sensors that can be defined by users are equipped with the model. The HCV model has two side mirrors and one rearview mirror, as shown in Figure 3, providing a realistic driving experience for CADS evaluation. To involve more realistic human maneuvers such as the pedal and steering wheel control, Logitech gaming steering wheel and pedal are set as the input interface of the simulation.

**Simulation Environment Construction** Taking advantage of the game engine, we construct a high-quality simulation environment in Unity including the network, infrastructure, and buildings based on a real-world map. In addition, to facilitate the lateral and longitudinal control of CAVs, we create a set of waypoints for each lane in the network.

In general, there are two ways to construct the simulation environment in both Unity and SUMO:

1. SUMO provides a tool called NETCONVERT, which can convert the OpenStreetMap (OSM) file into a 2D SUMO network file. Once we have the 2D map, we can build the 3D map in Unity accordingly.

**FIGURE 3** Hardware setup for HuiL in a lab environment.



2. According to the 3D network in Unity, we create the same 2D map in SUMO, ensuring two maps to have the same reference point for position synchronization between two simulators.

Typically, the second approach is more laborious than the first one, however OSM does not always provide high-quality map compared to the real world, which may affect the fidelity of simulation. In this article, we also choose the second method as we note that the OSM model of study area could not well reflect the real-world network geometry and situations.

**Traffic Flow Generation** To generate a realistic traffic flow, we apply the Poisson distribution. We assume the departure of vehicle follows a Poisson process, which means that given a random period $T$, the probability of the time interval between two departures is larger than $T$ can be calculated as follows:

$$P(T \le t) = 1 - e^{-\lambda t} \qquad \text{Eq. (1)}$$

where $\lambda$ represents the departure rate within a period time. Based on this model, the time interval between departures is exponentially distributed and can be calculated as follows:

$$t = \frac{\log(1-T)}{-\alpha} \qquad \text{Eq. (2)}$$

where $\alpha$ represents the traffic volume in vehicle per hour.

According to the equation, we can assign departure time for each vehicle and save them as a route file containing each individual vehicle's properties, such as vehicle ID, departure time, and the predefined route. SUMO can spawn vehicles based on this route file and Unity can create vehicles with the same properties after the first time it receives their information from SUMO. Once creating vehicles in Unity, the platform may determine if each individual vehicle is a CAV, legacy vehicle, or the HCV based on the logic described in the following section.

**Vehicle Control-Type Classifier** To assign the role for each vehicle, we create a vehicle control-type classifier (see Figure 1), which can select a vehicle to be a CAV or a legacy vehicle based on a random number generator. It uses the vehicle ID as a seed number to generate a random number between 0 and 1. Compare the random number $p$ and a predefined threshold $P$ that represents the penetration rate:

$$\text{Vehicle type} = \begin{cases} \text{CAV}, & p < P \\ \text{Legacy vehicle}, & p \ge P \end{cases} \qquad \text{Eq. (3)}$$

Besides this, the HCV is selected by a predefined vehicle ID. At the first time Unity receives the HCV ID from SUMO, the HCV prefab is deployed in Unity with the same position and velocity as SUMO.

**Traffic Flow Terminator** The traffic flow terminator mainly fulfills two functions: (a) to remove vehicles that already finished their trips and (b) to remove vehicles that are not synchronized in both simulators.

With the traffic flow generation, we can spawn vehicles and have them driven in both SUMO and Unity. After vehicles finish their trips or reach the end of their destinations, SUMO can automatically remove those SUMO-controlled vehicles. Similarly, Unity is able to delete those vehicles controlled by Unity after they reach their last waypoints.

However, whenever a simulator removes a vehicle in its local side, the other simulator may not remove the same vehicle simultaneously. To resolve the problem, we check the vehicle ID list sent from the other simulator every time step, compare it with the vehicle ID list in the target simulator, and then remove those vehicles that only show up in one of the simulations.

**Cloud Computing and Personalized Profile Storing** To ensure scalability of the integrated platform, we take advantage of the *high-speed data processing* and *secure data storing* features of AWS. One previous study [28] developed a serverless cloud computing architecture with a real-time platoon-based speed advisory algorithm using AWS and proved that cloud computing provided by AWS could be a cost-effective solution for backend data processing and storing.

The cloud architecture consisted of three parts: AWS, Amazon Virtual Private Cloud (VPC) [29], and the external space. VPC is part of the AWS and includes the most important modules: Real-Time Processing, Data Stores, and Analytics Workbench. Specifically, Amazon S3 [30], Amazon DocumentDB [31], and Redis [32] are used for data storing. Inside the AWS, except VPC, there is the AWS IoT Core [33], which enables the connection between AWS to IoT devices (*Edge Gateway* in this study). It supports various devices and messages, which can be reliably and securely transmitted to AWS. Position, speed, and acceleration data acquired from simulation can be uploaded from the *Edge Gateway* to AWS IoT Core via MQTT [34].

Outside AWS, a web portal acts as a frontend for displaying data diagrams and showing personalized driving performance. All data collected by the integrated platform is uploaded to the cloud through the Edge Gateway via MQTT. In the meantime, the cloud will display the driving data, such as trajectory, velocity, acceleration, and gap between vehicles in real time. Besides, after a driving test run is over, the uploaded data will be stored in the cloud for playback and further analysis. In addition, we also create personal profile for each human driver and archive his or her historical trajectory, speed, acceleration, and other critical information from multiple driving test runs. By analyzing individual user's driving records, we can explore and evaluate his or her driving style and driving performance (e.g., risk class), and provide references for other serverless cloud services (e.g., personalized speed advisory, personalized navigation). In addition, we may create personalized driving behavior model for each individual driver and develop the associated "digital twin" agent in the cloud.

# Results

In this section, we implement a game theory-based CADS for on-ramp merging [36] in mixed traffic as a case study to demonstrate the application of the integrated platform, where personalized human behaviors are modeled and evaluated.

## Simulation Environment Setup

The simulation environment consists of two main parts: the software part and the hardware part. In the software part, to evaluate the CADS for on-ramp merging scenario, we first create the virtual environment in Unity, as shown in Figure 4(b), based on a real road network from the intersection of Chicago Avenue to the intersection of Iowa Avenue along Columbia Avenue in Riverside, CA [see Figure 4(a)]. Then according to the 3D map, we create the same 2D map, as shown in Figure 4(c), with NETEDIT provided by SUMO. The network consists of a two-lane mainline and a single-lane on-ramp. The simulation timestep for both simulators are set to be 0.02 s.

In terms of hardware setup, there are three monitors placed around the driver, as shown in Figure 4. The Logitech G27 racing steering wheel and pedal set has been placed in front of the middle monitor. Switching Unity gaming inputs from keyboard and mouse to joystick allows us to collect more realistic human behavior data in the integrated platform.

## Mixed Traffic Modeling

Regarding the mixed traffic flow, three types of vehicles are modeled in the case study. The driving behavior of legacy vehicles is modeled by the default car-following model and lane-changing model in SUMO, and CAVs are fully controlled by the game theory-based on-ramp merging algorithm [36], which governs the decision-making process of both car following and lane changing. Besides, the HCV is controlled by subject driver through the steering wheel and pedal input. Only CAVs and the HCV are equipped with on-board LiDARs as described by a relatively simple model (see Figure 5). All vehicles created in Unity are based on real Toyota and Lexus models, with white, red, and blue colors representing legacy vehicles, CAVs, and the HCV, respectively.

Specifically, the SUMO car-following model is based on the Krauß model [37], and the lane-changing model is based on the LC2013 model [38]. The Krauß model is defined below [37]:

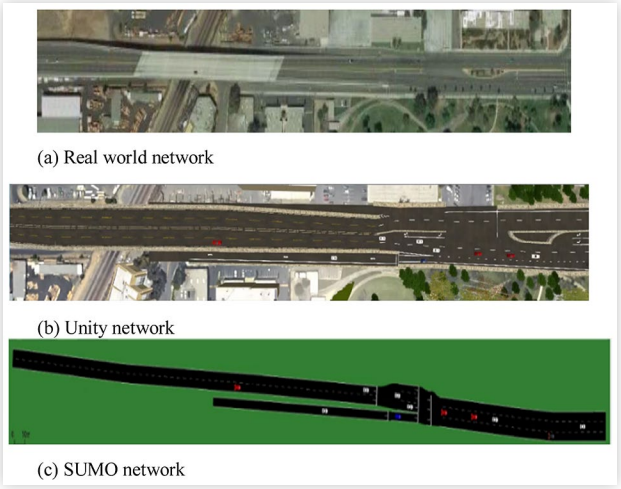
$$v_{safe}(t) = v_l(t) + \frac{g(t) - g_{des}(t)}{\tau + \tau_b}, \quad \text{Eq. (4)}$$

$$v_{des}(t) = \min\left[v\_\max,\ v(t) + a(v)\Delta t,\ v_{safe}(t)\right], \quad \text{Eq. (5)}$$

$$v(t + \Delta t) = \max\left[0,\ v_{des}(t) - \eta\right] \quad \text{Eq. (6)}$$

$$x(t + \Delta t) = x(t) + v\Delta t \quad \text{Eq. (7)}$$

**FIGURE 4** Network of simulation environment.



(a): Map data: Google, © 2021 County of San Bernardino, Maxar Technologies, U.S. Geological Survey, USDA Farm Service Agency

(a) Real world network

(b) Unity network

(c) SUMO network

**FIGURE 5** A simple LiDAR model.

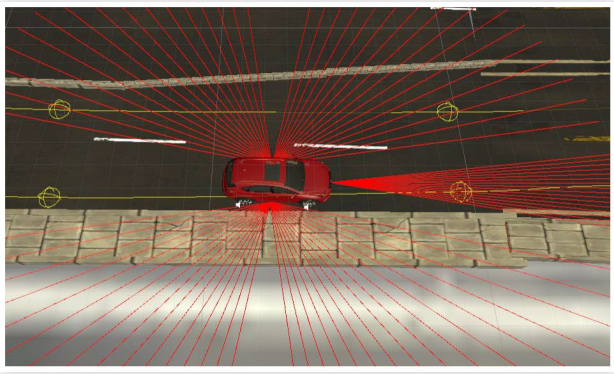




**TABLE 1** Configuration of three vehicle types.

| Vehicle type | CAV | Legacy vehicles | HCV |
|---|---|---|---|
| Car-following model | Game theory-based algorithm [36] | Krauß | Human driver |
| Lane-changing model | | LC2013 | |
| Initial speed | 10 m/s | | |
| Minimum gap | 5 m | | |
| Maximum acceleration | 4 m/s$^2$ | | |
| Minimum deceleration | –5 m/s$^2$ | | |
| Maximum speed | 20 m/s | | |
| Length | 5 m | | |



where $g_{des}$ is the desired gap, $\tau$ is the reaction time of the drivers, and $\tau_b$ is the time for deceleration. The lane-changing model contains four hierarchy layers based on different motivations for lane-changing: strategic change, cooperative change, tactical change, and regulatory change [38].

A game theory-based ramp merging strategy has been applied to CAVs in the co-simulation. Compared with the baseline, the algorithm can compute the target speed and make the decision for lane changing. It significantly improves the system mobility and reduces the fuel consumption [36].

A human driver uses the steering wheel and pedal set to control an HCV in the Unity through the Unity-supported joystick input interface. We have also applied a CADS, which runs the same CAV algorithm that provides speed advisory and lane-changing instruction via the customized graphical user interface (GUI) presented on the windscreen [39].

## Simulation Scenarios and Result

We verify the function and capability of the integrated platform, using 48 different highway on-ramp merging scenarios with realistic traffic flows. For the background traffic, we have three penetration rates of CAVs (i.e., 0%, 50%, and 100%) and two congestion levels (i.e., medium and heavy). In the case of HCV, drivers may or may not be provided with the suggested speed generated by the driving assistance system. In addition, HCV may be spawn into the network from the mainline or on-ramp. In the case study, two drivers are invited to participate in testing and 24 different scenarios for each driver.

As aforementioned, a predefined route file is generated based on the Poisson process. The background traffic is spawned at an initial speed of 10 m/s in the integrated simulator according to the departure time of each vehicle defined in the route file. The human driver needs to wait until the HCV to be spawn. In the scenarios with driving assistance, once the HCV is spawn, the speed advisory and the lane-changing guidance provided by the game theory-based ramp merging strategy will be displayed on the windscreen. The traffic demand ratio of highway to on-ramp is set to be 3:1. Furthermore, the total traffic flow rate for heavy and medium congestion level are 3200 passenger car units per hour (pcu/hr) and 1600 (pcu/hr), respectively. The configuration of all three vehicle types is shown in Table 1.

**Data Collection and Analysis** Among 48 testing scenarios, we select a typical one with 50% penetration rate and medium congestion level to demonstrate the collection and archiving of data and how we can utilize them. In this scenario, driver 1 controls the HCV as a ramp vehicle and follows the instructions from ADAS to try to merge into the mainline.

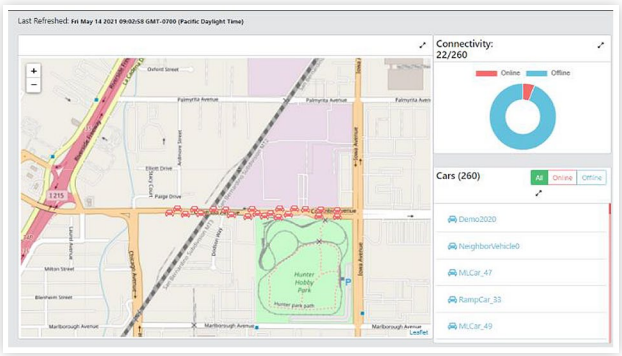
As the simulation is running, an overview map on the web portal is shown in Figure 6, and all simulation vehicles are displayed on the map. Connected vehicles are shown as red vehicles and disconnected vehicles are shown as blue vehicles. A vehicle is set to be disconnected when it stops uploading its information for more than 5 minutes.

During the simulation, the information from every vehicle is uploaded and stored on AWS, but only the data from HCV is plotted on the web portal, as shown in the Figure 7(b) and Figure 8(b). In the following figures, we compare the HCV driving data (velocity and gap) uploaded to AWS and displayed on the web portal (online), with the data stored and processed locally (offline). It is noted that for AWS, the information uploading sample rate is 1 Hz, which may lead to slight difference between the online data and offline data. The drastic drop in Figure 8 is caused by lane changing.

Using the stored data, we can easily recap a trip and analyze driving behavior, which is useful for personalize related research. In Figure 9, we display the trajectory of the HCV and its surrounding vehicles to show the capability of collecting information from all vehicles. Then we use the data to evaluate the human behaviors via two measurements: time to collision (TTC) and velocity volatility. TTC is computed by [41]:

$$TTC_i = \frac{X_i(t) - X_{i-1}(t) - l_i}{\dot{X}_i(t) - \dot{X}_{i-1}(t)} \quad \forall \dot{X}_i(t) > \dot{X}_{i-1}(t) \quad \text{Eq. (8)}$$

**FIGURE 6** Overview map on the web portal during simulation.






where $X_i(t)$ is the position of the HCV; $X_{i-1}(t)$ is the position of its preceding vehicle; and $l_i$ is the length of the HCV. The average TTC is 44.40 s and the lowest TTC is 3.90 s. Previous study [42] suggested that scenarios with TTC value larger than 4 s can be considered as safe driving. Although the lowest TTC is lower than 4 s, there are only two occurrences where TTC is lower than 4 s. In this case, the driver is proved to have a safe driving behavior.

The volatility of velocity represents how fluctuating the velocity signal is. A steady state of the velocity is quantified by low velocity volatility.

$$\text{Speed volatility} = \frac{c > \text{Threshold}}{n} \times 100\% \qquad \text{Eq. (9)}$$

$$\text{Threshold} = \bar{x} \pm 2x \cdot S_{dev} \qquad \text{Eq. (10)}$$

where $c$ is the number of observations over the *threshold* and $n$ is the total number of observations [43]. The volatility of the HCV is 1.36%, showing the velocity is relatively steady. According to the TTC and volatility performance, the driving behavior shows the driver is a safe driver. Through the example case study, the capability of collecting and analyzing data of the integrated platform is verified.

## Environmental Impact on Driving Behavior

In this section, we compare 48 defined trip scenarios in multiple aspects and show that the impacts of different traffic conditions on driving behaviors always match our expectation.

To quantify the driving behavior, we choose a driver rank method proposed in [40], which can determine aggressiveness of a trip with a continuous number ranging from −1 to 1 (representing the mildest behavior and the most aggressive behavior, respectively). In this article, aggressiveness is specific to describe the aggressive driving behavior, which is characterized by higher speeds, shorter headways, more frequent lane changes, and more rapid accelerations and decelerations. Aggressiveness is quantified by the following terms:

$$z_{ij} = 2 \times \frac{x_{ij} - \min x_{ij}}{\max x_{ij} - \min x_{ij}} - 1 \qquad \text{Eq. (11)}$$

$$Q_i = \frac{1}{s}\sum_{j=1}^{s} z_{ij} \quad (i = 1,2\ldots,r) \qquad \text{Eq. (12)}$$

where, $x_{ij}$ is diagnostic variables; $i$ is the number of driving trips; $j$ is the number of synthetic variables. In this article, standard deviation (STD) of the acceleration and STD of the velocity are selected as diagnostic variables. Therefore, the number of synthetic variables ($j$) is two. The rank of driver is determined by the average of the synthetic variables.

To determine the personalized driver's behaviors under different environment settings, we first explore individual driver's rank with respect to congestion level, penetration rate, and trip origin (i.e., mainline vs. on-ramp), and then compare

**FIGURE 7** The time-speed diagram of the HCV.

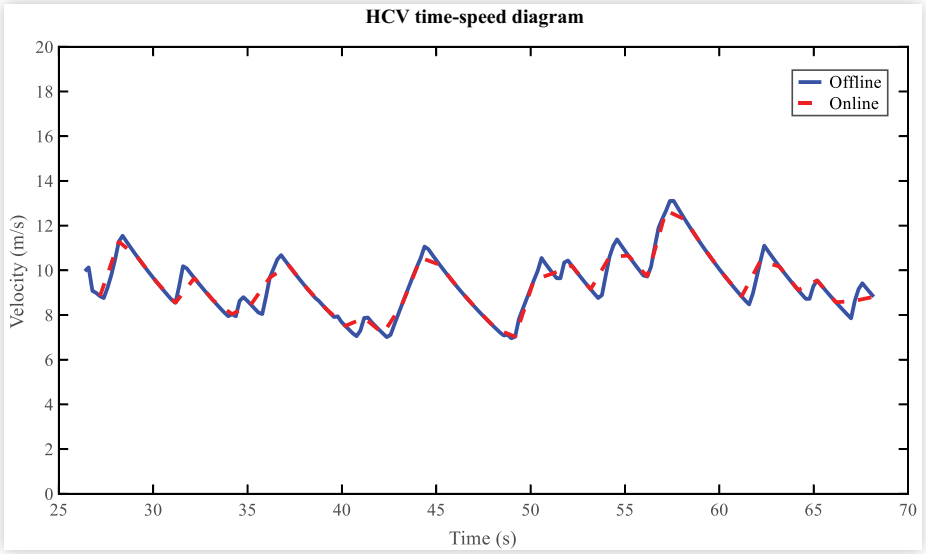
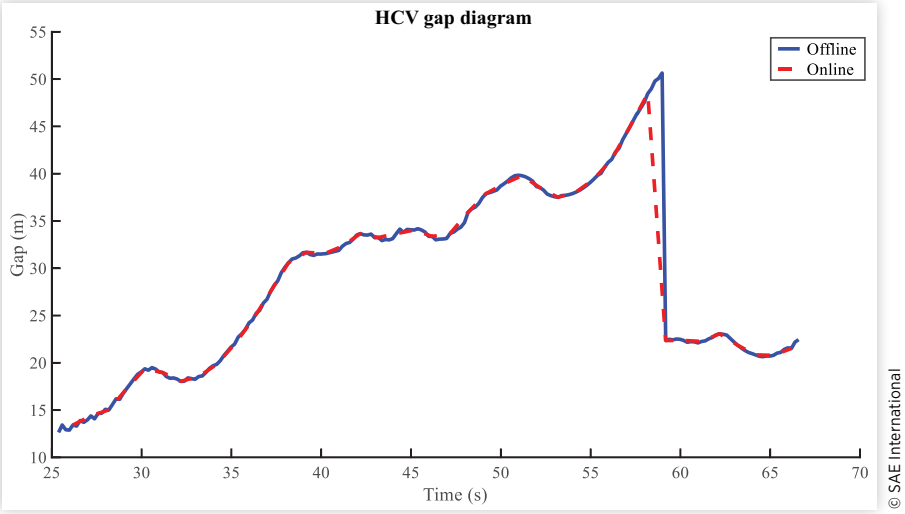


**FIGURE 8** The gap between the HCV and its preceding vehicle.




the difference in behavior changes between two drivers due to the introduction of driving assistance system. In Tables 2-4, we only display partial results, but the observations are consistent across most of the cases.

By observing the driver rank variation due to different penetration rates, which is shown in Table 2, we can find that as the penetration rate increases, almost all the driving ranks drop. According to the driving ranks, lower number represents drivers who have less variance in speed and acceleration. In this case, as the penetration rate rising, the drivers perform less aggressively. It matches the conclusion in previous study [36] that the game theory-based ADAS can help drivers of on-ramp vehicles to have smoother merging trips under higher penetration rates.

Then we evaluate the simulation results regarding the congestion level-impacts on driving behaviors for trips along different lanes (mainline vs. on-ramp). Under the large traffic volume, little space is left for the driver to perform aggressive lane changing or frequent acceleration and deceleration. In line with this expectation, in high-density traffic scenarios, the driver's ranks are lower as shown in Table 3. Since on-ramp vehicles need to find suitable gaps to merge into the highway, operation of the throttle and brake by the driver of the ramp vehicle would be frequent. Compared to the on-ramp

**FIGURE 9** The time-space diagram of HCV and its neighboring vehicles.

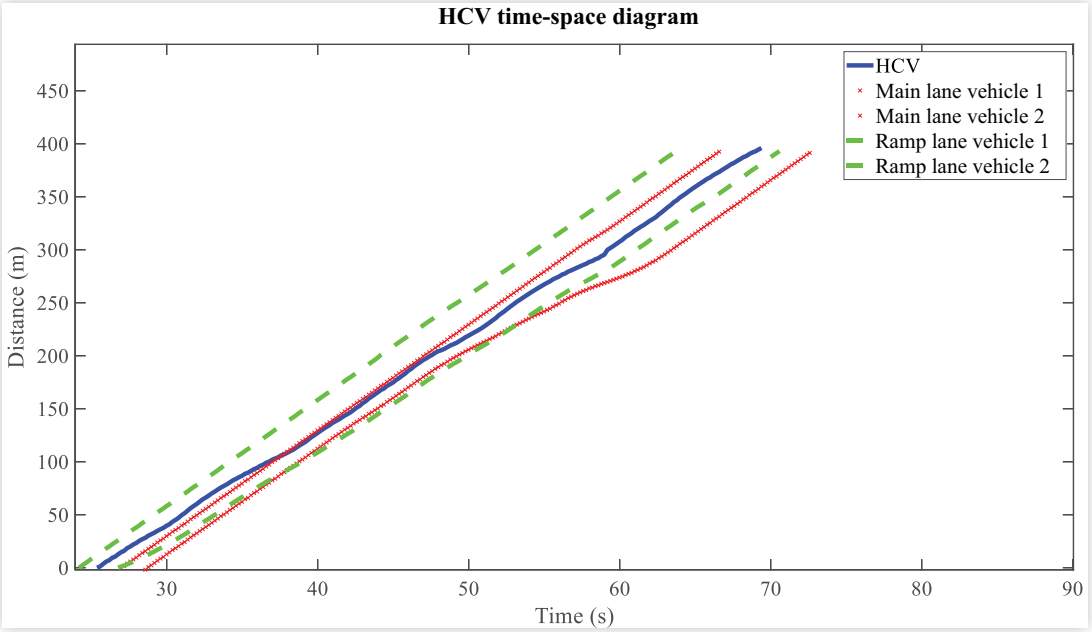


**TABLE 2** Driver ranks affected by penetration rates and ADAS (Driver 1, Medium CL, On-ramp).

| PR (%) | ADAS | Driver rank |
|---|---|---|
| 0 | W/ | −0.51 |
| 50 | W/ | −0.68 |
| 100 | W/ | −0.69 |
| 0 | W/o | −0.34 |
| 50 | W/o | −0.47 |
| 100 | W/o | −0.64 |



**TABLE 3** Driver ranks affected by different lanes and congestion levels (Driver 2, w/o ADAS, 100% PR).

| Lane | Congestion level | Driver rank |
|---|---|---|
| On-ramp | Heavy | −0.34 |
| On-ramp | Medium | −0.22 |
| Mainline | Heavy | −0.53 |
| Mainline | Medium | 0.37 |



scenarios, highway mainline scenarios have milder driving behaviors or lower driver's ranks as shown in Table 3.

Under the same penetration rate (e.g., 100%) and the same congestion level (e.g., medium), driver ranks are lower when ADAS is present, as shown in Table 4. It proves that drivers tend to be milder if they are provided with the speed guidance.

To evaluate the behavior difference between two drivers, we first present results without the ADAS under different conditions as baseline. The average driver ranks of drivers 1 and 2 are −0.53 and −0.40, respectively. This means that driver 2 is more aggressive than driver 1 in general. Then we take the previous cases in Table 4 as a typical scenario to analyze the ADAS impact on different drivers. With the ADAS, the driver ranks drop to around −0.67 for both drivers, but the original driver rank of driver 2 is much higher than driver 1. It shows that the ADAS leads to larger impacts on the more aggressive driver.

With the comparison, we demonstrate the capability of the integrated co-simulation platform to analyze the impact of ADAS on driver behaviors under various traffic scenarios and validate that the platform can reflect the environmental impact of driving behavior.

# Discussion

In the future development and utilization of co-simulation platforms, quite a few challenges need to be addressed from the perspectives of both research and engineering. In this section, we discuss the major issues that stand out among others during our development and implementation of this platform.

**TABLE 4** Driver ranks of different drivers affected by ADAS (On-ramp, Medium CL, 50% PR).

| Driver | ADAS | Driver rank |
|---|---|---|
| Driver 1 | W/ | −0.68 |
| Driver 1 | W/o | −0.48 |
| Driver 2 | W/ | −0.67 |
| Driver 2 | W/o | −0.33 |



## Asynchronous Communications

One of the purposes of using game engine is to create an immersive driving environment to obtain realistic human reactions. Hence, user experience and data fidelity are of high priority. Synchronous communication that requires strict time sequences for information flowing between different components may result in dramatic drops in FPS of the game engine, when the number of road users (e.g., vehicles) to be modeled increases and the computational load gets heavier. This would significantly impact the user experience and real-time performance. To mitigate this issue, we implement asynchronous communication technique, which can keep simulation running smoothly even if the computational time may be longer than the update interval. Nevertheless, if the computational load is too heavy or the information latency from the other component is high, then vehicle "jumping" (from the driving simulation perspective) can still be observed. In that case, we need to leverage the power of cloud computing, which will be explained in the following.

## Integration with the Cloud

In SUMO, TraCI provides a lot of valuable functions for retrieving vehicle states (e.g., position, acceleration) and further calculating useful information, such as *getCO2Emission* and *getFuelConsumption*. However, running all these functions in a single simulation time step would significantly increase the computational load, thus requiring nontrivial computing power from the platform. By integrating AWS into the co-simulation and leveraging its cloud computing resources, we are able to transfer those time-consuming functions to the cloud server and balance the resource utilization among components of the entire platform. Toward this end, we can optimize the overall simulation performance (e.g., modeling fidelity and information timeliness), while satisfying hardware and software constraints of each component.

## Platform Extension

It is noted that we consider the potential extensibility of the platform during its development. For example, as an open-source traffic simulator, SUMO is well compatible with OMNET++ for modeling a realistic connected vehicle environment, and also provides a set of APIs to flexibly connect with other software and tools. On the other hand, the game engine, Unity, provides plug-and-play access to virtual reality (VR) equipment for enabling more immersive environment and testing scenarios with multi-modal (such as pedestrians, bicyclists, e-scooters) interactions. More importantly, the

developed data exchanging center, *Edge Gateway*, facilitates the integration with real-world vehicles and other hardware (e.g., ROS-enabled miniature RC cars) to build up a mixed reality platform. A variety of real-world driving scenarios can be modeled and evaluated using the integrated platform to address the associated research questions (especially related to human behaviors), such as car following on highways, eco-driving along signalized corridors, and lane change prediction.

# Conclusions and Outlook

This study proposed an integrated Unity-SUMO-AWS platform allowing performance evaluation of personalized driving behaviors with ADAS under a variety of mixed traffic scenarios through HuiL simulation. We showed that this platform is able to collect necessary information, such as realistic driving behaviors, detailed dynamics at the individual vehicle level, delicate interactions between neighboring vehicles, and aggregated parameters of traffic flows. The equipped hardware, such as steering wheel and pedals, allowed collecting personalized driving information (e.g., turning wheel angles, brake/accelerator forces). All this information was uploaded to AWS for archiving and behavior modeling. Moreover, we used cooperative ramp merging assistance as an example to showcase the capabilities of this platform and investigate the environmental impacts of the driving behaviors from two subject drivers under different scenarios (with varying penetration rates of CAVs and congestion levels). The results indicated that the proposed platform is well suitable for comprehensive human factor-related research.

The development of such co-simulation platforms would definitely unlock unprecedented opportunities to study high-fidelity driving behaviors and delicate interactions between human-driven vehicles and CAVs under various controlled scenarios, although there remain some challenges to be addressed in the future. For example, stricter calibration procedures of co-simulation process, better time synchronization across different components (of the platform) in a mixed reality environment, and latency reduction of data exchanging with the cloud are open questions deserving further exploration.

# Contact Information

**Guoyuan Wu**
University of California Riverside
gywu@cert.ucr.edu

# Definitions/Abbreviations

**ADAS** - Advanced Driver-Assistance Systems
**API** - Application Programming Interfaces
**AWS** - Amazon Web Services
**CADS** - Cooperative Automated Driving Systems
**CAV** - Connected and Automated Vehicle
**CL** - Congestion Level
**COM** - Component Object Model
**FPS** - Frame per Second
**HCV** - Human-Controlled Vehicle
**HID** - Human Interference Device
**HuiL** - Human-in the-Loop
**IDM** - Intelligent Driver Model
**PR** - Penetration Rate
**RC Cars** - Radio-controlled Cars
**SDK** - Software Development Kit
**STD** - Standard Deviation
**SUMO** - Simulation of Urban MObility
**TTC** - Time to Collision
**UDP** - User Datagram Protocol
**V2X** - Vehicle to Everything
**VPC** - Virtual Private Cloud
**VR** - Virtual Reality

# References

1. Toledo, T., Koutsopoulos, H.N., and Ben-Akiva, M., "Estimation of an Integrated Driving Behavior Model," *Transportation Research Part C: Emerging Technologies* 17 (2009): 365-380.
2. Dosovitskiy, G., Ros, F., Codevilla, A., and Lopez, V.K., "CARLA: An Open Urban Driving Simulator," ArXiv abs/1711.03938 (2017): n. pag, accessed 26 April 2022.
3. Rong, G., Hyun Shin, B., Tabatabaee, H., Lu, Q. et al., "SVL Simulator: A High Fidelity Simulator for Autonomous Driving," in *2020 23th International Conference on Intelligent Transportation Systems*, Rhodes, Greece, September 2020, accessed 26 April 2022.
4. Craighead, J., Murphy, R., Burke, J., and Goldiez, B., "A Survey of Commercial & Open Source Unmanned Vehicle Simulators," in *Proceedings IEEE International Conference on Robotics and Automation*, Roma, Italy, 852-857, 2007, https://doi.org/10.1109/ROBOT.2007.363092.
5. Unity, "Unity for All," February 29, 2020, https://www.unrealengine.com.
6. Epic Games, "Unreal Engine," April 28, 2019, https://www.unity.com.
7. Wang, Z., Wu, G., Boriboonsomsin, K., Barth, M.J. et al., "Cooperative Ramp Merging System: Agent-Based Modeling and Simulation Using Game Engine," *SAE Int. J. CAV* 2, no. 2 (2019): 115-128, https://doi.org/10.4271/12-02-02-0008.
8. Wang, Z., Han, K., and Tiwari, P., "Digital Twin-Assisted Cooperative Driving at Non-Signalized Intersections," *IEEE Transactions on Intelligent Vehicles* Vol. abs/2105.01357, (2021): 1-1, https://doi.org/10.1109/TIV.2021.3100465.
9. Wang, Z., Liao, X., Wang, C., Oswald, D. et al., "Driver Behavior Modeling Using Game Engine and Real Vehicle: A

Learning-Based Approach," *IEEE Transactions on Intelligent Vehicles* 5, no. 4 (2020): 738-749.

10. Liu, Y., Wang, Z., Han, K., Shou, Z. et al., "Vision-Cloud Data Fusion for ADAS: A Lane Change Prediction Case Study," *IEEE Transactions on Intelligent Vehicles* Vol. abs/2112.04042, (2020): 1-1, https://doi.org/10.1109/TIV.2021.3103695.
11. Codevilla, F., Müller, M., López, A., Koltun, V. et al., "End-to-End Driving Via Conditional Imitation Learning," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, Brisbane, Australia, 2018, 4693-4700, https://doi.org/10.1109/ICRA.2018.8460487.
12. Fellendorf, M. and Vortisch, P., "Microscopic Traffic Flow Simulator VISSIM," *International Series in Operations Research & Management Science* 145 (2010): 63-94.
13. Aimsun, "Aimsun Next 20 User's Manual," Aimsun Next Version 20.0.3 [in software], Barcelona, Spain, accessed May 1, 2021, qthelp://aimsun.com.aimsun.20.0/doc/UsersManual/Intro.html.
14. Behrisch, M., Bieker, L., Erdmann, J., and Krajzewicz, D., "SUMO—Simulation of Urban Mobility an Overview," Paper Presented at in *the 3rd International Conference on Advances in system simulation*, Barcelona, Spain, October 23-29, 2011.
15. Krajzewicz, D. et al., "Recent Development and Applications of SUMO—Simulation of Urban MObility," *International Journal on Advances in Systems and Measurements*, vol 5 no 3 & 4: 128-138, 2012.
16. Vrbanić, F., Čakija, D., Kušić, K., and Ivanjko, E., "Traffic Flow Simulators with Connected and Autonomous Vehicles: A Short Review," in Petrović, M., and Novačko, L. (eds), *Transformation of Transportation. EcoProduction (Environmental Issues in Logistics and Manufacturing)* (Cham: Springer, 2021), https://doi.org/10.1007/978-3-030-66464-0_2.
17. Chen, X., Zhao, H., Liu, G., Ren, Y. et al., "Assessing the Influence of Adverse Weather on Traffic Flow Characteristics Using a Driving Simulator and VISSIM," *Sustainability* 11, no. 3 (2019): 830.
18. Althoff, M., Koschi, M., and Manzinger, S., "CommonRoad: Composable Benchmarks for Motion Planning on Roads," in *2017 IEEE Intelligent Vehicles Symposium (IV)*, Los Angeles, CA, 2017, 719-726, https://doi.org/10.1109/IVS.2017.7995802.
19. Klischat, M. et al., "Coupling SUMO with a Motion Planning Framework for Automated Vehicles," in *SUMO*, Berlin, 2019.
20. Schirner, G., Erdogmus, D., Chowdhury, K., and Padir, T., "The Future of Human-in-the-Loop Cyber-Physical Systems," *Computer* 46 (2013): 36-45.
21. Chen, J.B.-C. and Peng, H., "Differential-Braking-Based Rollover Prevention for Sport Utility Vehicles with Human-in-the-Loop Evaluations," *Vehicle System Dynamics* 36 (2001): 359-389, https://doi.org/10.1076/vesd.36.4.359.3546.
22. Li, W., Sadigh, D., Sastry, S.S., and Seshia, S.A., "Synthesis for Human-in-the-Loop Control Systems," in Ábrahám, E. and Havelund, K. (eds), *Tools and Algorithms for the Construction and Analysis of Systems. TACAS 2014*, Lecture Notes in Computer Science, vol 8413 (Berlin, Heidelberg: Springer, 2014), https://doi.org/10.1007/978-3-642-54862-8_40.
23. Biurrun, L.S. and Monreal, C.O., "Microscopic Driver-Centric Simulator: Linking Unity3D and SUMO," in Biurrun-Quel, C., Serrano-Arriezu, L., Olaverri-Monreal, C., Rocha, A. et al. (eds.), *Advances in Intelligent Systems and Computing*, Springer International Publishing, Cham, 2017, https://doi.org/10.1007/978-3-319-56535-4_83.
24. Szalai, M., Varga, B., Tettamanti, T., and Tihanyi, V., "Mixed Reality Test Environment for Autonomous Cars Using Unity 3D and SUMO," in *2020 IEEE 18th World Symposium on Applied Machine Intelligence and Informatics (SAMI)*, Herlany, Slovakia, January 2020, 73-78, https://doi.org/10.1109/SAMI48414.2020.9108745.
25. Lee, K., Murray, D., Hughes, D., and Joosen, W., "Extending Sensor Networks into the Cloud Using Amazon Web Services," in *2010 IEEE International Conference on Networked Embedded Systems for Enterprise Applications*, Suzhou, November 2010, 1-7, https://doi.org/10.1109/NESEA.2010.5678063.
26. Deng, H.W., "COSACC: Cloud-Based Speed Advisory for Connected Vehicles in a Signalized Corridor," All Theses, 2020, 3466, https://tigerprints.clemson.edu/all_theses/3466.
27. Lopez, P.A., Behrisch, M., Walz, L.B., Erdmann, J. et al., "Microscopic Traffic Simulation Using SUMO," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, Maui, HI, 2018, 2575-2582, https://doi.org/10.1109/ITSC.2018.8569938.
28. Deng, H.-W., "COSACC: Cloud-Based Speed Advisory for Connected Vehicles in a Signalized Corridor," All Theses, 2020, 3466.
29. AWS, "Amazon Virtual Private Cloud," June 5, 2021, accessed 26 April 2022, https://aws.amazon.com/vpc/?vpc-blogs.sort-by=item.additionalFields.createdDate&vpc-blogs.sort-order=desc.
30. AWS, "Amazon S3," June 5, 2021, https://aws.amazon.com/s3/, accessed 26 April 2022.
31. AWS, "Amazon DocumentDB (with MongoDB Compatibility)," June 5, 2021, accessed 26 April 2022, https://aws.amazon.com/documentdb/.
32. Redis, "Redis," June 5, 2021, accessed 26 April 2022, https://redis.io/.
33. Amazon, "AWS IoT Core," May 31, 2021, accessed 26 April 2022, https://aws.amazon.com/iot-core/.
34. Hunkeler, U., Truong, H.L., and Stanford-Clark, A., "MQTT-S—A Publish/Subscribe Protocol for Wireless Sensor Networks," in *2008 3rd International Conference on Communication Systems Software and Middleware and Workshops (COMSWARE '08)*, Bangalore, India, 2008, 791-798, https://doi.org/10.1109/COMSWA.2008.4554519.
35. Arne, K., Martin, T., and Dirk, H., "Enhanced Intelligent Driver Model to Access the Impact of Driving Strategies on Traffic Capacity," *Philos. Trans. R. Soc.* A368 (2010): 4585-4605, http://doi.org/10.1098/rsta.2010.0084.

36. Liao, X., Zhao, X., Wu, G., Barth, M.J. et al., "A Game Theory Based Ramp Merging Strategy for Connected and Automated Vehicles in the Mixed Traffic: A Unity-SUMO Integrated Platform," arXiv-CS-Systems and Control, January 2021, https://doi.org/arxiv-2101.11237.

37. Krauß, S., Mobilitat, H., and Koln, S., "Microscopic Modeling of Traffic Flow: Investigation of Collision Free Vehicle Dynamics," *Sustainable and Intelligent Transportation Systems* 11 (1998): 23-24, https://doi.org/10.3390/su11030830.

38. Erdmann, J., "SUMO's Lane-Changing Model," in Behrisch, M. and Weber, M. (eds), *Modeling Mobility with Open Data*, Lecture Notes in Mobility (Cham: Springer, 2015), https://doi.org/10.1007/978-3-319-15024-6_7.

39. Wang, Z., Liao, X., Wang, C., Oswald, D. et al., "Driver Behavior Modeling Using Game Engine and Real Vehicle: A Learning-Based Approach," *IEEE Transactions on Intelligent Vehicles* 5, no. 4 (2020): 738-749.

40. Augustynowicz, A., "Preliminary Classification of Driving Style with Objective Rank Method," *International Journal of Automotive Technology* 10, no. 5 (2009): 607-610, https://doi.org/10.1007/s12239-009-0071-8.

41. Minderhoud, M.M. and Bovy, P.H.L., "Extended Time-to-Collision Measures for Road Traffic Safety Assessment," *Accident Analysis & Prevention* 33 (2001): 89-97, https://doi.org/10.1016/S0001-4575(00)00019-1.

42. Sultan, B. and McDonald, M., "Assessing the Safety Benefit of Automatic Collision Avoidance Systems (during Emergency Braking Situations)," in *18th International Technical Conference on the Enhanced Safety of Vehicles (ESV)*, Nagoya, Japan, 2003.

43. Kamrani, M., Arvin, R., and Khattak, A.J., "Extracting Useful Information from Basic Safety Message Data: An Empirical Study of Driving Volatility Measures and Crash Frequency at Intersections," *Transportation Research Record* 2672, no. 38 (2018): 290-301, https://doi.org/10.1177/0361198118773869.