Integrated Simulation Platform for Internet of Vehicles

Xing Liu*, Wei Yu*, Cheng Qian*, David Griffith†, and Nada Golmie†
*Towson University, USA
Emails: {xliu10, cqian1}@students.towson.edu, wyu@towson.edu
†National Institute of Standards and Technology (NIST), USA
Emails:{david.griffith, nada.golmie}@nist.gov

***Abstract*—The interconnection and digitization of the physical world has increased dramatically with the widespread deployment of network communication and the rapid development of the Internet of Things (IoT). Application scenarios and requirements in IoT are more complex and diverse than ever before. To successfully support the design and development of complex IoT systems, a realistic evaluation platform that can accurately simulate both the physical world and network communications is necessary. Yet, most existing simulation tools are limited, simulating only specific subsets of IoT environments, such as communication network simulation or mobility simulation, rather than complete IoT scenarios. Thus, in this paper, we propose a new framework, in which several modules can work together to achieve more realistic simulation of IoT environments. Specifically, we integrate three-dimensional object motion with the OMNET++ network simulator. In our framework, we can configure and direct object movement in 3D and compute the received power of transmitted signals using ray tracing techniques. Within the framework, OMNET++ simulates the communication process based on the received power and communication protocol. As a demonstration of our framework, we conduct several experiments on two classic Internet of Vehicles (IoV) scenarios. The results indicate that our proposed framework can accurately simulate both the physical and communication aspects of IoT systems.**

***Index Terms*—Internet of Things, Integrated simulation, Internet of Vehicles**

I. Introduction

With the deployment of massive numbers of smart devices, a complete vision of the Internet of Things (IoT) is increasingly being realized, through the development and implementation of numerous smart-world systems in the areas of transportation, manufacturing, energy, and others [1]. These smart devices are connected through IoT networking infrastructure to sense a variety of physical properties and environments and communicate with each other so that they can collect data to support intelligent decisions. IoT applications cover a number of aspects relevant to human livelihood and societal wellbeing, such as smart homes, smart electrical grids, smart transportation systems, smart manufacturing systems, and smart healthcare systems, among others. Through these IoT applications, human life can become more convenient and efficient.

Although the topic of IoT has been extensively explored in a variety of research works, a number of issues remain to be addressed. Specifically, there remains a lack of high-fidelity integrated simulation platforms, which are capable of characterizing interactions between cyber and physical components in complex IoT systems, as well as evaluating designed models, algorithms, and protocols. For instance, in the area of smart transportation systems, to simulate a realistic, urban Internet of Vehicles (IoV) scenario, a simulation should consider the mobility of vehicles, conditions of roads, interference from buildings, and other factors. Nonetheless, traditional traffic [2] and network [3] simulation tools are not suitable for such IoV simulation because they lack the necessary level of detail, and cannot reflect the interactions between the cyber and the physical worlds appropriately.

At the same time, machine learning techniques have been widely adopted to assist in data analysis for IoT applications [4]. When using machine learning algorithms in IoT scenarios, it is necessary to collect significant amounts of data from specific training scenarios to achieve high accuracy. Moreover, these trained machine learning models are generally suitable for the trained scenario only. When we apply model to another scenario, we need to at least retrain the model, which requires collecting a large amount of data in the new scenario. Nonetheless, in many scenarios, there will not be enough data to train machine learning models. Thus, many IoT applications cannot achieve their desired outcomes with machine learning.

Developing a general framework that can be used to evaluate IoT applications and generate realistic IoT data for machine learning is of great importance. A realistic IoT evaluation environment should be able to accurately reflect the interactions between cyber and physical domains. For example, in an IoV system, messages exchanged between vehicles will affect the behaviors of the vehicles. A vehicle should also be able to transmit different messages based on driving status. All events occurring during simulation should be able to be stored and retrieved. In this way, we can develop machine learning algorithms for IoV based on the collected data.

To this end, in this paper, we propose a general framework to carry out realistic IoT simulation of IoV. Our designed framework consists of four modules: *Scenario Generation*, *Mobility Simulation*, *Network Simulation*, and *Visualization*. In each module, we combine several tools to improve the simulation accuracy. In the Scenario Generation module, we convert OpenStreetMap (OSM) [5] data to realistic 3D models for a selected area. We use the Java OSM (JOSM) editor [6] to clear the data of OSM and use the Houdini 3D modeling engine [7] to generate the 3D objects. In the Mobility Simulation module, we leverage the open-source Simulation of Urban MObility (SUMO) to generate moving objects for the simulation [8]. SUMO supports a high degree of customization, making it suitable for simulating IoT environments that are typically heterogeneous. In the Network Simulation module, we use the Objective Modular Network Testbed in C++ (OMNET++) to perform network simulation [9][1]. OMNET++, as a discrete event simulator, can evaluate large scale IoT deployments. We also use Optix to introduce ray tracing technology to further improve the accuracy of simulation results. Finally, in the Visualization module, we use the Unity game engine [10] to graphically display the simulation. During simulation, the Mobility Simulation module, Network Simulation module and Visualization module can work collaboratively via communication over TCP/IP (Transmission Control Protocol/Internet Protocol) ports.

To summarize, we make the following contributions: (i) **Framework:** We propose a general framework to perform realistic IoT simulations. In detail, we use a combination of simulation tools to simulate real-world IoT scenarios to the greatest extent currently possible. Additionally, we visualize the simulation to generate image data for future machine learning research and development purposes. (ii) **Extensive Validation:** We conduct an extensive performance evaluation to show that our proposed framework can accurately reflect the interactions between communication and physical domains. We validate our framework on two classic IoV scenarios: an urban area and a highway area. In each scenario, we design two cases: accident and non-accident. The simulation results are discussed from the perspectives of communication and physical. From the communication perspective, we focus on the received power signals and the packet delivery rate. From the physical perspective, we focus on the vehicle travel time.

The remainder of this paper is organized as follows: In Section II, we conduct a brief review of relevant studies regarding IoT simulation. In Section III, we present our proposed framework in detail. In Section IV, we describe the simulation scenarios that we used to test our framework and discuss the evaluation results. Finally, we summarize the paper in Section V.

II. Related Work

With the development of research work on IoT applications, the demand for IoT simulation tools is growing [11]. The state-of-art simulation tools can be categorized into several major types: full-stack simulators, data processing driven simulators, and network simulators. The full stack simulators provide the end-to-end support of all IoT elements. The data processing driven simulators focus on enabling applying big data analysis techniques to the applications [12]. Finally, network simulators support the evaluation of various network protocols in IoT scenarios. For instance, the bidirectional integration capability of OMNET++ makes it viable as a network simulator for heterogeneous IoT applications.

The role of game engines in IoT applications can be divided into two categories: reproducing 3D scenes and creating user interaction interfaces for IoT applications. For example, Yu *et al.* [13] used the Unity game engine to provide a virtual reality environment for college physical education using the data collected from IoT devices. The game engine can also create user interaction interfaces for IoT applications. For example, Olaverei-Monreal *et al.* [14] installed the traffic light assist (TLA) system into a virtual environment developed by a unity game engine.

Finally, because simple models of fading and shadowing effects can be inadequate, ray tracing techniques can be used to provide a high level of detail in the characterization of multipath channels in wireless communication systems. For instance, He *et al.* [15] designed a high performance ray tracing simulation platform specifically for 5G communications.

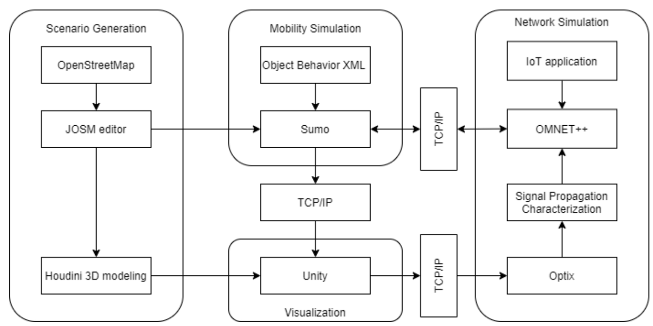

Fig. 1. Simulation Framework

III. Our Approach

In this section, we present our proposed framework for simulating realistic IoT systems.

A. Overview

Our approach consists of four modules: Scenario Generation, Mobility Simulation, Network Simulation, and Visualization. Fig. 1 illustrates steps in the simulation process. In scenario generation module, the map data is converted into a 3D model that can be recognized by the Unity game engine. This allows us to accurately reproduce geographic environments for IoT scenarios in a virtual environment. We use OSM data as the input of this module. The OSM data includes buildings, roads, intersections, and traffic lights. We use the JOSM editor to fix errors and missing information in the OSM data. Finally, we use the processed OSM data to generate the 3D model for the scenarios in the Houdini 3D modeling engine.

In mobility simulation module, the behavior of any moving object in the IoT scenario is defined by an Extensible Markup

[1]Certain commercial equipment, instruments, or materials are identified in this paper in order to specify the experimental procedure adequately. Such identification is not intended to imply recommendation or endorsement by the National Institute of Standards and Technology, nor is it intended to imply that the materials or equipment identified are necessarily the best available for the purpose.

Language (XML) file. This includes the source location, destination location, speed, type, and other properties of mobile objects. Based on the XML file, SUMO will simulate the movement of mobile objects in the IoT scenario that we defined.

In network simulation module, we leverage OMNET++ to provide network simulation. First, we define the IoT applications to be evaluated. Normally, these applications include network configurations (e.g., data rate, routing strategy, encoding method, packet size, and others). Then, OMNET++ will deploy the network configurations on objects in the evaluated IoT scenario. The received power level that OMNET++ uses in the communication simulation is derived from Optix using real-time ray tracing techniques.

In visualization module, the Unity game engine visualizes the buildings, streets and moving objects in the defined IoT scenarios.

As shown in Fig. 1, the overall simulation process is carried out according to the following steps:

- We obtain the geographic data from OpenStreetMap in the form of OSM data.
- In the Scenario Generating Module, the OSM data is sent to the JOSM editor so that unnecessary information can be removed.
- The processed OSM data are sent to SUMO to build the mobility simulation scenario. At the same time, the Houdini 3D modeling engine will take processed OSM data from the JOSM editor and generate the 3D scenario.
- Within the mobility simulation, the behavior data, such as moving objects, are written to XML so that SUMO can directly recognize.
- SUMO uses the behavior XML file and the geographic data to conduct the mobility simulation.
- The position and the velocity of moving objects are sent to the Unity game engine in the Visualization Module and OMNET++ in the Network Simulation module through TCP/IP messages.
- The Unity game engine visualizes the IoT scenario based on the received TCP/IP messages and the 3D model generated by Houdini 3D.
- The Unity engine sends building material and shape information to Optix in the Network Simulation module for ray tracing calculation.
- Optix computes the received signal power and sends it to OMNET++.
- OMNET++ simulates the network communications based on the IoT application, the IoT scenario, and the received signal power.
- OMNET++ sends the network simulation results to SUMO to update the behaviour status of objects.
- The simulation repeats all above steps until the simulation is complete.

B. Scenario Generation

In our approach, we convert real-world scenarios into digital 3D models to build realistic IoT scenarios. The OSM data that we use in this framework contains sufficiently detailed information about IoT scenarios, including buildings, roads, traffic lights, and the speed limit on each road, among others. In addition, the JOSM editor allows us to further streamline the process through the removal of unnecessary information, reducing the complexity of the 3D modeling process. For instance, if pedestrians on the road are not considered in the current IoT scenario, the sidewalk data can be removed. Moreover, we leverage the capabilities of the Houdini 3D modeling engine for generating accurate digital IoT scenarios. The generated digital IoT scenarios can be directly exported to the Unity game engine for visualization.

TABLE I. Simulation Parameters

Parameter	**Value**
MAC type	IEEE 802.11p
Simulation volume	500 m x 500 m x 50 m
Transmission range	500 m
Vehicle max speed	(45,25,15) m/s
Packet size	32 bytes
DCF Inter-Frame Space (DIFS)	34 μs
Short Inter-Frame Space (SIFS)	16 μs
Number of vehicles	200

C. Mobility Simulation

Typical scenarios in IoT consist of a number of moving objects, including vehicles, pedestrians, drones, and others. We leverage the SUMO simulator for generating road networks and traffic in the IoT scenario. SUMO can identify edges, lanes, intersections, and connections in the processed OSM data, to generate a road network. With the generated network, we can then use the trip builder in SUMO to generate the traffic flow of the simulation. We can also customize the mobility of each object manually, enabling a variety of distinct and unique IoT scenarios.

D. Network Simulation

OMNET++ has a well-developed open-source library, which can meet the simulation requirements of heterogeneous network communication for most IoT scenarios. Thus, we leverage OMNET++ as the network simulator to interact with the other components in our framework. We use Optix, developed by NVDIA, to perform the ray tracing analysis on the transmitted signal. Using Optix, we can simulate electromagnetic waves by emitting rays from the transmitter at predetermined angular intervals. These rays will be reflected, diffracted, transmitted, or scattered once they hit an obstacle in the simulation. The received signal power is calculated by adding up the received power of different rays that reach to the receiver. Such intensive calculations require significant computational resources, which can be further leveraged to support the statistical propagation models provided by the OMNET++ modules.

E. Visualization

Visualizing IoT scenarios can increase the diversity of IoT data that we collect. By leveraging the Unity game engine to

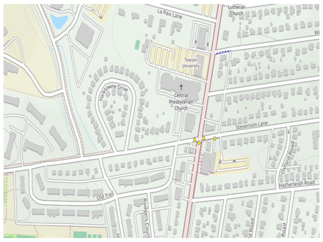

Fig. 2. Urban Area Open Street Map

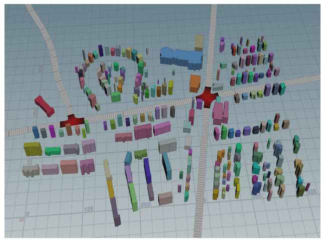

Fig. 3. Urban Area 3D modeling

Fig. 4. Highway Area Open Street Map

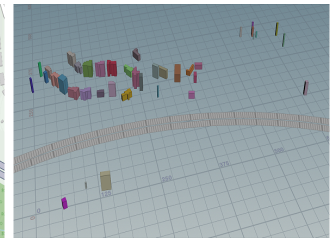

Fig. 5. Highway Area 3D Modeling

visualize the IoT scenarios, we can set up cameras at different locations and angles, and collect and store images of IoT scenes at various points in time. This allows us to simulate real-time images taken by smart cameras deployed in cities. These collected images can then be used as data for developing machine learning applications in IoT. For example, we can use images with 3D models instead of real-life pictures to train the machine learning model for estimating communication channels between moving vehicles without raising personal privacy issues.

IV. Performance Evaluation

We set up two classic IoV scenarios (i.e., urban area and highway area) to demonstrate the efficacy of our proposed framework. For each scenario, we design two cases: accidents and no accidents. We validate the functionality of our proposed framework with respect to received power, packet delivery rate, and travel time of vehicles.

A. Evaluation Methodology

For each scenario, we choose a (500 x 500) m^2 area in Towson, MD, USA. The simulation was run on a workstation with a Windows 10 operating system, 32 GB of memory, and an AMD 2700 CPU. The Optix ray-tracing simulations were run on a standalone NVIDIA 3070 GPU. We use IEEE 802.11p as our communication protocol for all the vehicles in the network. The data rate is set to 6 Mbit/s. If an accident occurs, the accident car will send a warning message to surrounding vehicles every 2 s. Then, vehicles that receive a warning message will send it to other vehicles. According to the specific position of the vehicle on the scene, the vehicle that receives a warning message will adopt one of three actions: slowing down, stopping, or changing lanes. The OMNET++ and SUMO exchange messages every 2 ms through TCP/IP. The Unity game engine and Optix also exchange messages every 2 ms through TCP/IP. The simulation for each scenario runs for 700 s of simulated time. More simulation parameters are shown in Table I.

B. Scenarios

1) Urban Area: In urban area, Dense buildings make the communication environment in the urban area complicated, because the shape of the building has a great impact on communication. On the other hand, the roads in the urban area are also staggered and complicated, with many intersections and traffic lights. This makes the density of vehicles at certain intersections higher. It not only increases the total number of packets transmitted within the communication range, but the vehicle between the data transmitter and the data receiver will also reflect and refract the signal.

We usually use statistical models to simulate signal propagation in IoV applications. Nonetheless, the simulation results obtained by using statistical model cannot accurately reflect some characteristics of a specific urban scenario. We can use ray tracing technology to simulate signal propagation more accurately. The urban scenario that we use in this paper is shown in Fig. 2. It is a small commercial area near the Towson University campus. Depending on the type of road that the vehicle is on, the maximum speed of the vehicle is limited to 15 m/s or 25 m/s. The 3D modeling of this urban area is shown in Fig. 3. As the original map data contains some roads that vehicles cannot drive on (e.g., sidewalks), to efficiently use the computational resources of the simulation, we only keep the arterial road that when generating the 3D model.

2) Highway Area: Most vehicles on highways travel at similar speeds, so that we could consider that two vehicles in the same traffic flow are relatively stationary. Nonetheless, due to the high speed of vehicles on the highway, the relative speed between the vehicle and the road side unit (RSU) is very high, resulting in a heavy Doppler effect on communication signals. The highway scenario that we use in this paper is shown in Fig. 4, which a small section of interstate 695 (I-695) in Towson, MD. The 3D modeling of this highway area is shown in Fig. 5.

C. Evaluation Results

1) Communication Perspective: From the perspective of communication, we focus on two metrics: the received power signal and the packet delivery rate. The received signal power is computed by adding up the received power of different rays that arrive at the receiver. The packet delivery rate is computed by the number of successfully transmitted packets divided by the total number of packets that are transmitted. Note that we run the program 20 times on each scenario and obtain experimental results that display the error bar with 95 % confidence intervals.

Fig. 6 illustrates the trend of the received signal power as the increase of communication distance between the transmitter and the receiver. The ideal situation indicates that there is no obstruction between the transmitter and the receiver. In

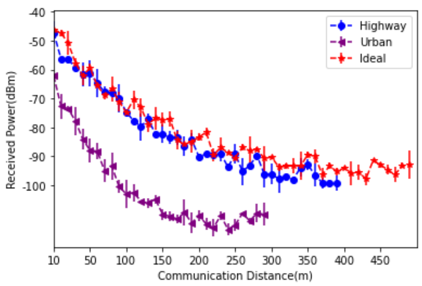

Fig. 6. Received Signal Power

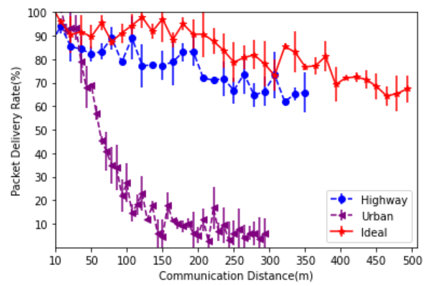

Fig. 7. Packet Delivery Rate without Accident

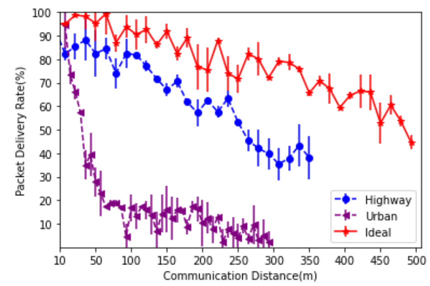

Fig. 8. Packet Delivery Rate with Accident

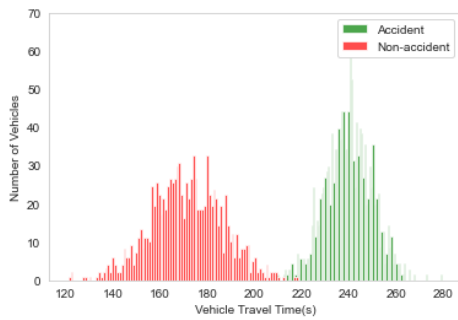

Fig. 9. Vehicle Travel Time in Urban Area

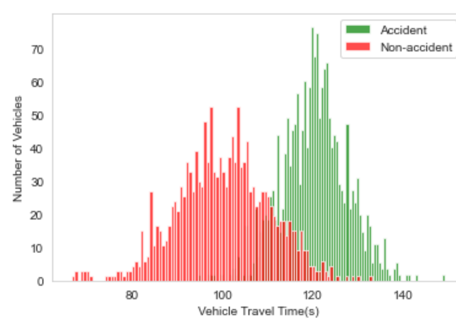

Fig. 10. Vehicle Travel Time in Highway Area

the urban area, the transmitter and the receiver have no line of sight, and all received signals must be reflected by buildings and other vehicles in between. While in the highway area, since there are no buildings between the transmitter and the receiver, only the reflection caused by other vehicles in between will be considered. Obviously, as the distance between the transmitter and the receiver increases, the received signal power in all three scenarios is lower. The received signal power in the ideal case decreases the slowest with the increase of distance. When the communication distance reaches the maximum limit of 500 m, the received signal power strength is still close to -90 dBm (power level expressed in decibels (dB) with reference to one milliwatt (mW)). Considering there is no obstruction between the transmitter and the receiver, the received signal power fits the free space propagation model.

For the highway scenario, the received signal power is slightly lower than the received power in the ideal scenario, but much higher than the received power in the urban scenario. This is because although there are multiple vehicles between the transmitter and the receiver, and there is no building blockage in the middle of the road, the signal can reach to the receiver after one or two reflections from the ground.

In the highway scenario, the length of the highway is around 600 m. Thus, we limit the longest distance between the transmitting vehicle and the receiving vehicle to be within the communication distance of 400 m. The received signal power in the urban area is the lowest. When the communication distance reaches 100 m, the received signal power is only around -100 dBm. When the communication distance reaches 300 m, the received signal power attenuates greatly. Thus, we do not consider the situation where the communication distance in the urban scenario is larger than 300 m. Also, the reason why the received signal power is weak is because the signals reflect multiple times to reach to the receiver, because there are usually buildings between the transmitter and the receiver. Because of the high density of buildings in the city, the longer the communication distance, the more buildings will be between the transmitter and the receiver, which further reduces the power of the received signal.

Fig. 7 illustrates the relationship between packet delivery rate and communication distance when no accident occurs. The packet delivery rate of the ideal scenario is the highest, followed by the highway scenario, and the urban scenario has the lowest packet delivery rate. This result is as expected because the same encoding method and data rate are used in these three scenarios. The received signal power strongly affects on the packet delivery rate and results in a steep decrease in the delivery rate over relatively short distances, with the greatest impact occurring in the urban scenario. Fig. 8 shows the relationship between packet delivery rate and communication distance when an accident occurs. In the event of an accident, the accident vehicle will send a warning signal to other nearby vehicles. Once a vehicle receives the warning signal, it will forward it to other nearby vehicles. Based on their locations, vehicles could slow down, stop, or change the lane. Thus, when an accident occurs, the number of simultaneous broadcasting packets within the communication range increases, and the density of vehicles increases. As a result, these data packets have a higher chance of colliding with other data packets, leading to higher interference. That explains why the packet delivery rate with accident is lower. For the urban scenario, the packet loss rate is almost 70 % at 50 m communication distance, as shown in Fig. 8. For the highway scenario, the packet delivery rate is dropped around 20 % at 400 m communication distance. For an ideal scenario, the packet delivery rate is dropped around 20 % at 500 m communication distance. This is because the urban scenario has the most intersection area, when an accident occurs, the vehicles will easily be stuck on the intersection area. This effect causes an additional increase in the relatively high density of vehicles in the urban scenario so that it is the highest among the three scenarios, and so that the effect on packet delivery rate is the greatest.

2) Physical Perspective: For the physical system perspective, we focus on the trajectories of the vehicles during the simulation. In both urban scenario and highway scenario, we set the total number of vehicles to 200. The generated vehicles have different source and destination locations, usually at the edge of the road. Vehicles also have different speeds according to the road that they travel on. For example, in urban areas, vehicles travel at speeds below 25 m/s, while in highway scenarios, vehicles can travel at 45 m/s. Fig. 9 illustrates the distributions of vehicle travel time in urban scenario with and without accident. In order to reduce the randomness of the results, we ran the same scenario 20 times. For the urban scenario without accident, the travel time of vehicles is mostly between 160 simulated time units (say seconds) to 180 simulation time units. For the urban scenario with accident, the travel time of vehicles is mostly between 220 simulated time units to 260 simulation time units. The travel time increases as the vehicles will slow down or stop once they received accident warning signal. Compared to the situation without accidents, the variance of travel time with accidents is also smaller.

Fig. 10 shows the distributions of vehicle travel time in the highway scenario with and without accident. We also ran the same scenario 20 times. For the highway scenario without accident, the travel time of vehicles is mostly between 90 simulated time units to 110 simulation time units. For the urban scenario with accident, the travel time of vehicles is mostly between 110 simulated time units to 130 simulation time units. By comparing Fig. 9 with Fig. 10, we can observe that, the travel time of vehicles under the highway scenario is less affected by the accident than vehicles under the urban scenario. There are two reasons for this outcome. First, there are intersections and traffic lights in the urban scenario, which will cause more waiting time than the highway scenario when the accident occurs. Second, the packet delivery rate in the highway scenario is higher than that in the urban scenario. As a result, vehicles in the highway scenario are more likely to receive warning signals, so that they can slow down or change lanes in advance. On the other hand, because the vehicle in the urban scenario has not received the warning signal, it may need to stop completely and start again when an accident is discovered ahead.

V. Final Remarks

In this paper, we proposed a general framework for IoT simulation. Our framework consists of Scenario Generation, Mobility Simulation, Network Simulation, and Visualization modules. The framework can be customized to adapt to diverse IoT scenarios, affording individual or collaborative control of the modules, enabling the modification of vehicle mobility, network protocols, communication environments, and others. We validated our proposed framework on two classic IoV scenarios: an urban area and a highway. The simulation results demonstrate that our framework can realistically reflect the interactions between physical objects and communication networks in such scenarios. Our framework also provides various types of simulation results, including images, object trajectories, and communication logs. In our future research, we plan to use this developed framework as a platform to design IoV applications to improve IoV communication performance and traffic patterns. Moreover, we can generate sufficient realistic data from the framework to support machine learning-driven applications in IoT scenarios.

References

[1] H. Xu, W. Yu, D. Griffith, and N. Golmie, "A survey on industrial Internet of Things: A cyber-physical systems perspective," *IEEE Access*, vol. 6, pp. 78 238–78 259, 2018.

[2] Q. Chao, H. Bi, W. Li, T. Mao, Z. Wang, M. C. Lin, and Z. Deng, "A survey on visual traffic simulation: Models, evaluations, and applications in autonomous driving," in *Computer Graphics Forum*, vol. 39, no. 1. Wiley Online Library, 2020, pp. 287–308.

[3] N. Cheng, W. Quan, W. Shi, H. Wu, Q. Ye, H. Zhou, W. Zhuang, X. Shen, and B. Bai, "A comprehensive simulation platform for space-air-ground integrated network," *IEEE Wireless Communications*, vol. 27, no. 1, pp. 178–185, 2020.

[4] W. G. Hatcher and W. Yu, "A survey of deep learning: Platforms, applications and emerging research trends," *IEEE Access*, vol. 6, pp. 24 411–24 432, 2018.

[5] https://www.openstreetmap.org.

[6] https://josm.openstreetmap.de/.

[7] https://www.sidefx.com/.

[8] P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, and E. Wießner, "Microscopic traffic simulation using sumo," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2018, pp. 2575–2582.

[9] https://omnetpp.org/.

[10] https://unity.com/.

[11] M. Chernyshev, Z. Baig, O. Bello, and S. Zeadally, "Internet of things (IoT): Research, simulators, and testbeds," *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 1637–1647, 2017.

[12] A. P. Plageras, K. E. Psannis, C. Stergiou, H. Wang, and B. B. Gupta, "Efficient IoT-based sensor big data collection–processing and analysis in smart buildings," *Future Generation Computer Systems*, vol. 82, pp. 349–357, 2018.

[13] Y. Ding, Y. Li, and L. Cheng, "Application of internet of things and virtual reality technology in college physical education," *IEEE Access*, vol. 8, pp. 96 065–96 074, 2020.

[14] C. Olaverri-Monreal, J. Errea-Moreno, A. Díaz-Álvarez, C. Biurrun-Quel, L. Serrano-Arriezu, and M. Kuba, "Connection of the SUMO microscopic traffic simulator and the unity 3D game engine to evaluate V2X communication-based systems," *Sensors*, vol. 18, no. 12, p. 4399, 2018.

[15] D. He, B. Ai, K. Guan, L. Wang, Z. Zhong, and T. Kürner, "The design and applications of high-performance ray-tracing simulation platform for 5G and beyond wireless communications: A tutorial," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, pp. 10–27, 2018.