
# VEHICULAR 2021

The Tenth International Conference on Advances in Vehicular Systems, Technologies and Applications

ISBN: 978-1-61208-879-2

July 18 – 22, 2021

Nice, France

**VEHICULAR 2021 Editors**

Pascal Lorenz, University of Haute Alsace, France

# VEHICULAR 2021

# Forward

The Tenth International Conference on Advances in Vehicular Systems, Technologies and Applications (VEHICULAR 2021) continued a series of events considering the state-of-the-art technologies for information dissemination in vehicle-to-vehicle and vehicle-to-infrastructure and focusing on advances in vehicular systems, technologies and applications.

Mobility brought new dimensions to communication and networking systems, making possible new applications and services in vehicular systems. Wireless networking and communication between vehicles and with infrastructure have specific characteristics from other conventional wireless networking systems and applications (rapidly-changing topology, specific road direction of vehicle movements, etc.). These led to specific constraints and optimizations techniques; for example, power efficiency is not as important for vehicle communications as it is for traditional ad hoc networking. Additionally, vehicle applications demand strict communications performance requirements that are not present in conventional wireless networks. Services can range from time-critical safety services, traffic management, to infotainment and local advertising services. They are introducing critical and subliminal information. Subliminally delivered information, unobtrusive techniques for driver's state detection, and mitigation or regulation interfaces enlarge the spectrum of challenges in vehicular systems.

We take here the opportunity to warmly thank all the members of the VEHICULAR 2021 technical program committee, as well as all the reviewers. We also kindly thank all the authors who dedicated much of their time and effort to contribute to VEHICULAR 2021. We truly believe that, thanks to all these efforts, the final conference program consisted of top quality contributions. We also thank the members of the VEHICULAR 2021 organizing committee for their help in handling the logistics of this event.

**VEHICULAR 2021 Chairs**

**VEHICULAR 2021 Steering Committee**
Khalil El-Khatib, University of Ontario Institute of Technology – Oshawa, Canada
Éric Renault, ESIEE Paris, France
Xiaohong Peng, Birmingham City University, UK
Yuping He, University of Ontario Institute of Technology, Canada

**VEHICULAR 2021 Industry/Research Advisory Committee**
Clément Zinoune, Renault, France
Yi Ding, US Army RDECOM-TARDEC, USA
William Whyte, Security Innovation, USA
Markus Ullmann, Federal Office for Information Security / University of Applied Sciences Bonn-Rhine-Sieg, Germany
Manabu Tsukada, University of Tokyo, Japan

**VEHICULAR 2021 Publicity Chair**
Mar Parra, Universitat Politecnica de Valencia, Spain
Alvaro Liebana, Universitat Politecnica de Valencia, Spain

# VEHICULAR 2021
# Committee

**VEHICULAR 2021 Steering Committee**
Khalil El-Khatib, University of Ontario Institute of Technology – Oshawa, Canada
Éric Renault, ESIEE Paris, France
Xiaohong Peng, Birmingham City University, UK
Yuping He, University of Ontario Institute of Technology, Canada

**VEHICULAR 2021 Industry/Research Advisory Committee**
Clément Zinoune, Renault, France
Yi Ding, US Army RDECOM-TARDEC, USA
William Whyte, Security Innovation, USA
Markus Ullmann, Federal Office for Information Security / University of Applied Sciences Bonn-Rhine-Sieg, Germany
Manabu Tsukada, University of Tokyo, Japan

**VEHICULAR 2021 Publicity Chair**
Mar Parra, Universitat Politecnica de Valencia, Spain
Alvaro Liebana, Universitat Politecnica de Valencia, Spain

**VEHICULAR 2021 Technical Program Committee**
Nor Fadzilah Abdullah, National University of Malaysia (UKM), Malaysia
Mohammed Al-Ansi, University Malaysia Perlis (UniMAP), Malaysia
Sufyan T. Faraj Al-Janabi, University of Anbar, Ramadi,Iraq
Mustafa S. Al-Jumaily, University of Tennessee, Knoxville, USA
Ali Alfoudi, Al-Qadisiyah University, Irak
Ala'a Al-Momani, Ulm University, Germany
Bhaskar Anand, Indian Institute of Technology Hyderabad, India
Adel Aneiba, Birmingham City University, UK
Andrea Araldo, Telecom SudParis (Institut Polytechnique de Paris), France
Muhammad Asim, Chung-Ang University, Seoul, South Korea
Hakan Aydin, Karadeniz Technical University, Turkey
Eduard Babulak, Fort Hays State University, USA
Manlio Bacco, CNR-ISTI, Italy
Andrea Baiocchi, University of Roma "Sapienza", Italy
Ali Balador, RISE Research Institute of Sweden, Sweden
Paolo Barsocchi, ISTI (Institute of Information Science and Technologies) | Italian National Research Council (C.N.R.), Pisa, Italy
Paulo C. Bartolomeu, University of Aveiro, Portugal
Marcel Baunach, Graz University of Technology, Austria
Rahim (Ray) Benekohal, University of Illinois at Urbana-Champaign, USA
Sylvia Bhattacharya, Kennesaw State University, USA
Christos Bouras, University of Patras, Greece
Marcos F. Caetano, University of Brasilia, Brazil
Rodrigo Capobianco Guido, São Paulo State University (UNESP), Brazil

**Copyright Information**

For your reference, this is the text governing the copyright release for material published by IARIA.

The copyright release is a transfer of publication rights, which allows IARIA and its partners to drive the dissemination of the published material. This allows IARIA to give articles increased visibility via distribution, inclusion in libraries, and arrangements for submission to indexes.

I, the undersigned, declare that the article is original, and that I represent the authors of this article in the copyright release matters. If this work has been done as work-for-hire, I have obtained all necessary clearances to execute a copyright release. I hereby irrevocably transfer exclusive copyright for this material to IARIA. I give IARIA permission or reproduce the work in any media format such as, but not limited to, print, digital, or electronic. I give IARIA permission to distribute the materials without restriction to any institutions or individuals. I give IARIA permission to submit the work for inclusion in article repositories as IARIA sees fit.

I, the undersigned, declare that to the best of my knowledge, the article is does not contain libelous or otherwise unlawful contents or invading the right of privacy or infringing on a proprietary right.

Following the copyright release, any circulated version of the article must bear the copyright notice and any header and footer information that IARIA applies to the published article.

IARIA grants royalty-free permission to the authors to disseminate the work, under the above provisions, for any academic, commercial, or industrial use. IARIA grants royalty-free permission to any individuals or institutions to make the article available electronically, online, or in print.

IARIA acknowledges that rights to any algorithm, process, procedure, apparatus, or articles of manufacture remain with the authors and their employers.

I, the undersigned, understand that IARIA will not be liable, in contract, tort (including, without limitation, negligence), pre-contract or other representations (other than fraudulent misrepresentations) or otherwise in connection with the publication of my work.

Exception to the above is made for work-for-hire performed while employed by the government. In that case, copyright to the material remains with the said government. The rightful owners (authors and government entity) grant unlimited and unrestricted permission to IARIA, IARIA's contractors, and IARIA's partners to further distribute the work.

**Table of Contents**

# Hack The Automotive Simulator

## Setting Up a Simulation Environment for Carrying Out Hacking Attacks on CAN Bus Systems

Dirk Labudde[1], Heiko Polster[2] and Markus Straßburg[3]
Forensic Science Investigation Lab
Hochschule Mittweida – University of Applied Sciences Mittweida, Germany
Email: [1]dirk.labudde@hs-mittweida.de, [2]heiko.polster@hs-mittweida.de, [3]markus.strassburg@hs-mittweida.de

***Abstract*— The paper describes a digital simulation environment to reproduce and test attacks on Controller Area Network (CAN) bus systems. Security researchers repeatedly find vulnerabilities in different software components of networked vehicles. Since investigations on the real system seem too costly, various vehicle functions are to be tested and analysed during attacks with the help of a CAN bus simulator. This simulation environment can also be used to conduct on-site and remote training. For the simulation environment, we use the software Vector CANoe as well as CANUTILS to control and analyse the CAN bus systems. In order to be able to analyse the effects and thus the interrelationships and make them comprehensible, a CAN bus simulator was built in the form of a model car. The connection to the model is realised by means of two ESP32-EVB development boards configured as WLAN CAN gateways and a VN1610 CAN-USB interface. In the model, an AT90CAN128 development board functions as a control unit, which controls the motors and the lighting on the model car. The procedure for setting up a CAN bus simulation environment and using it to analyse and evaluate hacker attacks on automotive bus systems is described. The application possibilities show that the simulation environment can not only be used on-site, but in combination with web conferencing systems for theoretical knowledge transfer with a remote connection for solving practical tasks. It represents the most effective methodology for imparting knowledge in the field of car forensics online. Technological obstacles make it difficult to easily integrate practical tasks on real CAN bus systems into conferencing tools, as this requires a connection to the simulation hardware used. Therefore, this paper also shows how, in addition to the BigBlueButton web conferencing system, the AnyDesk remote maintenance software can be used to establish a remote connection to the control machine. Audio-visual feedback is helpful to clarify the effects of the CAN commands sent. Here, webcams are used to control the model car and a remote connection is used to enter the commands.**

***Keywords-cyberattacks; car forensics; can-bus; demonstrator; remote seminar.***

## I. INTRODUCTION

Today's motor vehicles are no longer controlled by the driver himself using wire ropes, levers or hydraulics, but via digitally networked computer systems. A depressed brake pedal no longer necessarily means that the brakes are actually applied. In modern vehicles, the software systems decide whether this actually happens. Additionally, cars are becoming increasingly networked, both internally and in relation to the outside world. This makes it possible for hackers to locate cars from the network and penetrate their systems. In the worst case, this will make them able to take control of important control systems [21]. Meanwhile, 32% of vehicles in the US are connected to the Internet [1]. In terms of new registrations of the ten largest car brands in the US, 95% are connected cars. By 2020, the three largest manufacturers - General Motors, Toyota and Ford - which together represent almost half of the US car market, had set themselves the goal of installing hardware for connected services in every sold vehicle [2].

In recent years, more and more software vulnerabilities have been found [16]. As software is continously becoming an integral component in modern cars, especially in the area of networked services, it can be assumed that more software vulnerabilities can also be found in vehicles [1][3]. In most cases, attackers penetrate the infotainment centre of vehicles via mobile wireless connections (e.g. for location or emergency call systems), where most security vulnerabilities can be found [2]. From there, it is sometimes possible to penetrate the control electronics. The biggest challenge here is to send vehicle data to the infotainment system without allowing data flow in the other direction [19][20].In this context, the question arises as to what data is actually stored and transmitted in the vehicle? This can be location data, stored routes, telephone data, error messages, time stamps, kilometre statuses or even exact parking locations of a vehicle at a defined point in time [19].

This raises additional questions: which electronic systems are installed in the vehicle under consideration, which interfaces do the various systems have, how can these systems be addressed or evaluated forensically, which hacking and analysis tools enable data evaluation? Furthermore, data is sent to the respective backend with the help of manufacturer-specific online or remote services. This data is not only of great interest to authorities, insurance companies and service providers, but can also help in investigations. The derived information can be assigned to a crime and thus, if necessary, to a person. "Forensic is related to scientific methods of solving crimes, involving examining the objects or substances that are involved in the crime." [22].

 ISBN: 978-1-61208-879-2 

The purpose of car forensics is to assist in solving crimes and to find data in vehicle systems that provide conclusions about the use or manipulation of the vehicles.

The motor vehicle and the automotive industry are undergoing a noticeable transformation: electric drives, autonomous driving and smart mobility require comprehensive networking of the vehicle with its environment. According to estimates, there will be 775 million connected vehicles in 2023, which will be connected by means of in-vehicle telematics or via a smartphone app [3]. The number of cyberattacks on motor vehicles is steadily increasing. For example, 73 attacks on vehicles were recorded in the whole of 2018 and 71 vehicles have been the victims of attacks in the first four months of 2019 [4].

Due to the considerable increase in possible attack vectors resulting from the networking of vehicles and the serious security deficiencies in the implementation of the CAN bus protocol, a subdivision must be made based on the access possibilities [17]. For this purpose, the following classification was established for the differentiation of attacks: direct, short-range and longe-range. It matters how these can be carried out by the attacker. Figure 1 shows a summary of the most frequent attack targets.

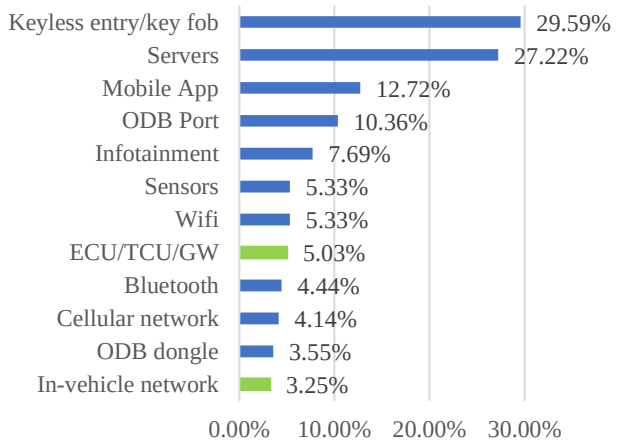



Figure 1. Distribution of attack targets on motor vehicles in descending order of occurrence. Green are the attack targets on which this paper focuses. [5]

In this work, attacks targeting Electronic Control Units (ECU) and in-vehicle network are specifically addressed. However, the represented simulation environment shows the opportunities to address further attack targets. The distance between the attacker and the vehicle plays a decisive role in the attack targets. Therefore, the attacks are classified according to the distance between the attacker and the vehicle. Direct attacks require direct physical access. The attacker must expose himself to the danger of implementing his attack directly, even if sometimes only briefly, on the vehicle itself. In addition, an attacker cannot determine the timing of the attack himself, but is rather dependent on the behaviour of the vehicle owner. This further limits the potential for direct attacks. Short-range attacks, on the other hand, allow the attacker to keep a certain distance from the vehicle by compromising those interfaces of the vehicle that operate wirelessly within a radius (from a few metres for Bluetooth and Tyre Pressure Monitoring System (TPMS) to approx. 100m for WLAN or remote opening systems). Long-range attacks allow the attacker to connect to the vehicle from any point, via the Internet or the mobile network, and carry out an attack. Since the attacker no longer needs to be in the immediate vicinity of a vehicle to be attacked, this greatly increases the chances that a vehicle can be attacked (see Figure 2). Thus, short- and long-range attacks can be carried out remotely. In 2019, 82% of all attacks on motor vehicles were carried out remotely (short-range and long-range) [5]. The number of attacks on motor vehicles increased significantly over the years from 2013 to 2019.

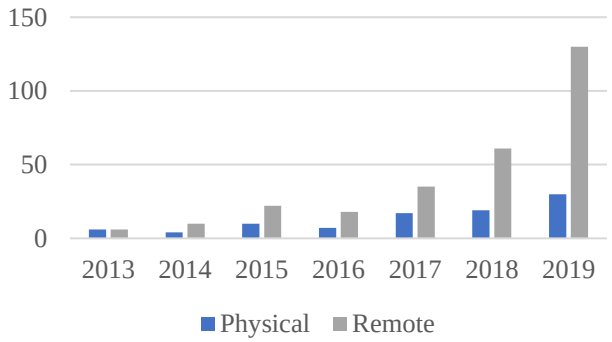



Figure 2. Comparison between physical and remote attacks [4]

Due to cost considerations and the dangers of manipulating vehicle systems during real-world use, a simulator offers a viable alternative to conducting hacking attacks on automotive bus systems. CAN bus simulation provides a safe way to analyse what-if scenarios. For example, the effects of sending CAN bus commands via the On-board Diagnostics (OBD) socket can be tested.

This work focuses on the attack targets of in-vehicle network and ECU. Therefore, direct attacks on automotive systems are primarily carried out and analysed. Dongles on the CAN interface are simulated, which can be used to access CAN bus systems and manipulate their data. In this way, direct attack vectors can be reproduced. In order to be able to use the simulator in the context of remote training, e-learning possibilities are discussed and an environment is described in which participants can access a CAN bus simulator via a remote connection and realise direct connections to this system. A major disadvantage with regard to remote seminars is the lack of feedback for participants. For this reason, this paper additionally describes possibilities to test the effects of CAN commands on a model car using a real CAN interface. The simulation environment was further equipped with cameras that give the participants remote visual feedback on the effects of their CAN commands. To enable several participants to work together on a task, a solution is described that enables remote group work and exchange with experts.

Section II presents the necessary tools that are used to implement the simulation environment. Section III describes

 ISBN: 978-1-61208-879-2 

the use case and the concrete structure of the simulator. A conclusion and the outlook can be found in Section IV.

## II. TOOLS FOR CAN-SIMULATION

### A. Vector CANoe

When developing new control units for motor vehicles, it is possible to simulate CAN bus systems using professional simulation environments. One of the tools used for this is CANoe from Vector [35]. In this tool, virtual ECUs can be tested in real CAN bus environments via a hardware interface, e.g. Vector VN1610. Within the environment, all valid CAN messages for the respective CAN system are stored in a database. These messages can be sent and received in the simulation by means of interactive generators or by virtual ECUs. The virtual ECUs are equipped with their special functionality using Communication Access Programming Language (CAPL). This type of programming is event-oriented. The control of the environment is realised with panels in which corresponding control elements, such as buttons, switches, visual displays, etc., are used. Figure 3 shows a CAN bus simulation of a virtual motor vehicle.

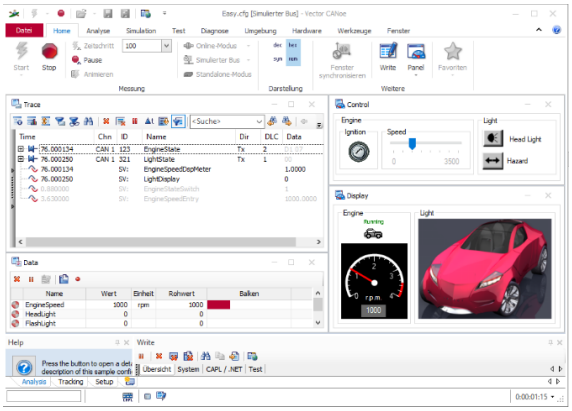


Figure 3. Exemplary representation of a CAN-Simulation with tool CANoe

These controls can be linked to system variables or even message content. Extensive analysis tools are available within the environment. Because of the possibility to freely program the virtual control units and the physical connection to real CAN bus systems, the operation of the CAN bus system can be used for the simulation of an attack.

### B. CANUTILS

CANUTILS are available within Linux systems. With this toolset, it is possible to control and analyse virtual and real CAN bus systems. Figure 4 shows an example of an intersection of CAN traffic with CAN identifiers and corresponding data content.

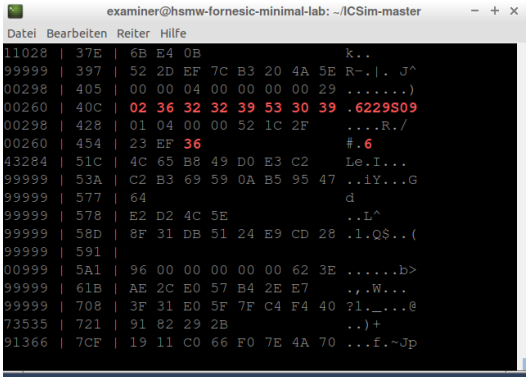



Figure 4. Recoding of CAN traffic using cansniffer from the CANUTILS

The following tools are integrated in the CANUTILS:

- candump: with candump CAN data can be displayed, filtered, logged and saved in log files.
- canplayer: CAN log files can be played back with canplayer.
- cansend: this tool enables the transmission of individual CAN frames.
- cangen: generates random CAN traffic.
- cansniffer: with the cansniffer, the traffic on the bus can be displayed live. In addition, it is possible to filter out individual CAN frames from the data stream

Within the simulation environment, the CANUTILS take care of the analysis of the CAN traffic.

#### 1) Linux with CAN

A prerequisite for working with CAN systems under Linux is the installation of CANUTILS. To connect real CAN systems to the computer, a USB CAN interface is necessary. For example, the PCAN-USB [7] interface from Peak can be used for this. The installation of the corresponding drivers is necessary for use. Further information on this can be found under [25].

#### 2) Red Pitaya

Red Pitaya STEMLAB is a grid-connected Field Programmable Gate Array (FPGA)-based test and measurement board. It can replace many measurement devices in an electronics lab by working as an oscilloscope, spectrum analyser, LCR meter (inductance, capacity, resistance), network analyser or other test and measurement application. Open source software is used and the operating system is based on Linux. Figure 5 shows the Red Pitaya hardware. An FPGA from Xilinx and an ARM Cortex-A processor are implemented on the board. The latter has a CAN interface on the hardware side, which is not activated in the basic configuration. However, this can be realised by means of devmem via the following commands:

- devmem2 0xf8000728 w 0x1221
- devmem2 0xf800072c w 0x1220

 ISBN: 978-1-61208-879-2

Figure 5. Red Pitaya hardware, which provides the backend for generating and recording CAN signals

After that, the pins and CANTX (header E2 pins 3 and 4) of the CAN interface are available on the Red Pitaya board. A CAN transceiver (e.g. Microchip MCP2551) must be connected to these pins to generate the differential CAN signals. To use the CAN interface, the CANUTILS can also be used in this system, since the Red Pitaya works with a Linux system. With a self-implemented application, the CAN interface can be used after installing the CANUTILS. Figure 6 shows an example of the data exchange between the Red Pitaya hardware and the frontend on a personal computer.

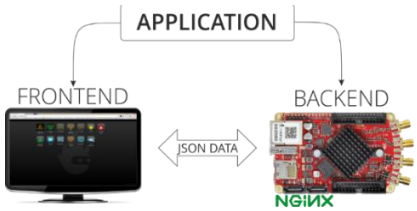



Figure 6. Scheme of data exchange between frontend and backend on the Red Pitaya System [8]

The Red Pitaya Board is connected to a computer via a network interface. The frontend is accessible via a browser with a configured IP address. Figure 7 shows the graphical frontend of the developed Can Sniffer application, which is used to analyse CAN traffic and send individual CAN messages.

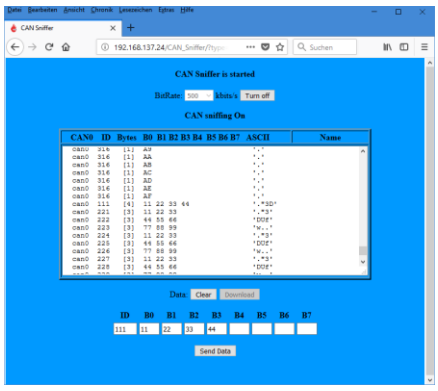



Figure 7. Recording of CAN traffic in the self-created interface for the frontend of the Red Pitaya Systems

The application itself is installed on the RedPitaya system and can be started via the frontend. Figure 7 shows a specially created interface for visualising CAN traffic on the CAN interface of the RedPitaya system. This interface communicates via the network with the RedPitaya application "CAN Sniffer". Via this interface, it is possible to analyse and generate traffic in the CAN simulator. Further information on the RedPitaya system can be found under the following: [13][14].

### C. *Bluetooth OBD-Dongle*

Private users can purchase various OBD Bluetooth adapters for vehicle diagnosis on the Internet. These are largely based on the OBD2 interpreter chip ELM327 [26]. This chip includes, among other things, communication via CAN with 11-bit and 29-bit identifier and supports the ISO 15765-4 protocols with 500 kbps and 250 kbps.


Figure 8. Exemplary representation of a common OBD dongle [9]

Figure 8 shows an OBD interface with Bluetooth interface, which makes vehicle data available with a smartphone application. These adapters can communicate with corresponding Android apps, for example to display current data on speed, acceleration, cooling water temperature, etc. in live operation. The technical description of the ELM327 can be found on the Elmelectronics website [11]. In this data sheet, the configuration variants of the chip are described, whereby one variant makes it possible to configure own baud rates for the interface and to send own CAN identifier with own data contents on a CAN bus. The ELM327 itself enables communication via a virtual RS-232 (Recommended Standard 232), which is available on a computer via Bluetooth. Thus, the chip can be configured and controlled with a console application (Putty) and the AT commands described in the data sheet.

### D. *Microcontroller*

Another variant of CAN communication can be seen in the use of microcontrollers with an integrated CAN interface. Preconfigured hardware can be purchased, for example, from Olimex. The AT90CAN128 development board, shown in Figure 9, is presented here as an example.

Figure 9. AT90CAN128 development board used to generate CAN messages [10]

The central element on the board is the microcontroller from Microchip ATMEL AT90CAN128. In addition to 128 kB of programmable memory and 4 kB of RAM, this controller offers a hardware-integrated CAN interface that is available to the user via a CAN transceiver MCP2551 and a Sub-D 9. Software libraries for controlling the CAN interface are available from a community [27]. By means of these libraries, it is possible to realise flexible configurations with regard to baud rates, CAN identifier and data contents. Firmware for the microcontroller can be created using Microchip Studio [34], which has an integrated C compiler. A JTAG interface is available on the development board as a programming interface, which can be controlled via the ATMEL ICE programmer [12]. The simulator uses this to generate its own messages on the CAN bus.

CANoe, PCAN-USB, CANUTILS and the AT90CAN128 development board with Microchip Studio were selected for practical use in the seminars. The decision to use these presented tools was made in order to achieve the fastest possible learning success for the seminar participants. The participants should be able to take part in the seminar without extensive programming knowledge. Seminar participants can thus quickly find their way around and only need one software for various functionalities. Furthermore, the CANoe tool is a standard tool in the automotive industry; the use of this tool should ensure practice-oriented teaching. This also results from the possibility of using these tools in a goal-oriented manner with regard to the subtasks to be solved in the seminar without major overhead for the participants. The aims of the seminar are:

- Creation of a CAN bus simulation,
- Analysis of CAN messages,
- Control of a CAN demonstrator and
- Compromising a CAN bus system.

CANoe is also a standard tool for the development of automotive bus systems. In addition to the training version used in the seminar, a demo version with a limited range of functions is available. In this version, for example, no real hardware can be controlled. The advantages of the training version are the possibility of connecting real CAN bus systems, the diverse possibilities for visualising control interfaces and the extensive analysis tools in the tool itself. The tool thus combines various functionalities that could otherwise only be used with different tools.

There are alternative adapters for the PCAN-USB adapter, e.g. USBtin [31] from the open source world or the isCAN USB from Thorsis Technologies GmbH [32], which is available for a fee. PCAN-USB was used in other projects and was therefore available. However, it will be examined later to use the USBtin adapter as an alternative solution.

The Microchip Studio is a proprietary Integrated Development Environment (IDE) for Atmel microcontrollers. It includes the AVR Gnu C compiler (AVRGCC) and is provided free of charge by Microchip.

The CANUTILS under Linux are open source and free to use. AVRStudio is an integrated C compiler and a graphical debugger.

An alternative solution for CANoe is the tool PCAN-Developer 4 from Peak [33], which is also available for a fee.

For programming Atmel microcontrollers, a number of commercial IDEs are available for a fee. These are, for example, IAR Embedded Workbench for AVR, JumpStart C for AVR from Image Craft or CodeVisionAVR from HP InfoTech.

The Red Pitaya system and the Bluetooth-OBD adapter are not used in the seminar. Both systems are only intended to show the possibility of compromising CAN bus systems at this point. They are currently to be classified as experimental status with a limited range of functions and thus cannot be used in a seminar.

## III. SIMULATIONS ENVIRONMENT AND USE CASES

Challenges for the simulation environment were the integration of the different systems into the seminar, as these are to be used under the two operating systems Linux and Windows 10. Windows is used to run CANoe, while the ICSIM-VM is based on a Linux system. In addition, there was the challenge of being able to make the seminar, which was actually planned as a face-to-face seminar, available online as well. This section also looks at the effectiveness of delivery formats in e-learning and considers the scenario in which the simulator can be used. Furthermore, a form of live workshop is described, which can use special hardware in remote seminars to solve practical tasks with it. Private and official investigators are particularly conceivable target groups.

First, various mediation formats are considered. The National Training Laboratories Institute for Applied Behavioural Science describes a learning pyramid on this topic, which shows the learning effect depending on the delivery format. The delivery formats range from a lecture to teaching other people. Figure 10 illustrates the learning pyramid [6]. In addition to the learning pyramid, there are a number of approaches to describing the learning process. David Kolb, for example, defines four different learning types and learning phases. In the first phase, practical experience is gained. The second phase involves mental observation and reflection. In the third phase, theory is introduced and problems are defined. Finally, in the fourth phase, the learned knowledge is tested for practical suitability. A solution approach is to be found through active trial and error in order to support the learning effect [28].

Furthermore, the importance of practical application and its feedback can be described with Phil Race's learning model. The five most important factors for successful learning according to Phil Race are as follows:

1. want motivation, interest, enthusiasm
2. need, survival, saving face
3. practice, repetition, experience, trial and error.
4. feedback on other people's reactions to see the results.
5. it makes sense to deal with what you have learned. [29]

 ISBN: 978-1-61208-879-2 

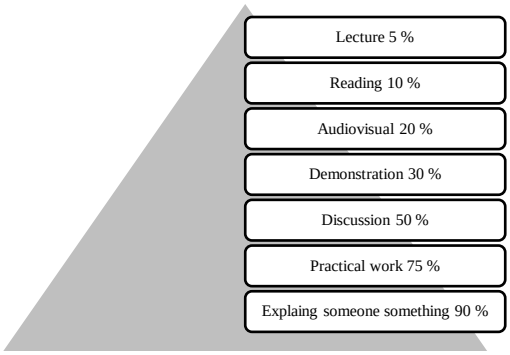



Figure 10. Learning pyramid from National Training Laboratories [6]

With the CAN bus simulator, it is possible to train the application of the learned knowledge on the topic of car forensics and CAN bus systems by means of practical exercises in addition to the classic lecture and discussions among the learners. The simulator can be used in physical presence as well as in online seminars. Remote live workshops offer users the opportunity to work on extensive tasks and solve complex issues. Furthermore, these workshops are characterised by the fact that different teaching formats are used. In addition to the live lecture to convey the theoretical content, the combination of meeting tools and remote connection options such as BigBlueButton and Remote Desktop Connections allow practical exercises to be carried out and the knowledge learned to be applied. In order to be able to expand practical modules in teaching with presence content, the transfer of the practical applications and exercises with an interface into the virtual environment is necessary.

The real learning environment of the module is composed of stationary computer systems with a virtual machine, a development environment for CAN microcontrollers and special simulation software.

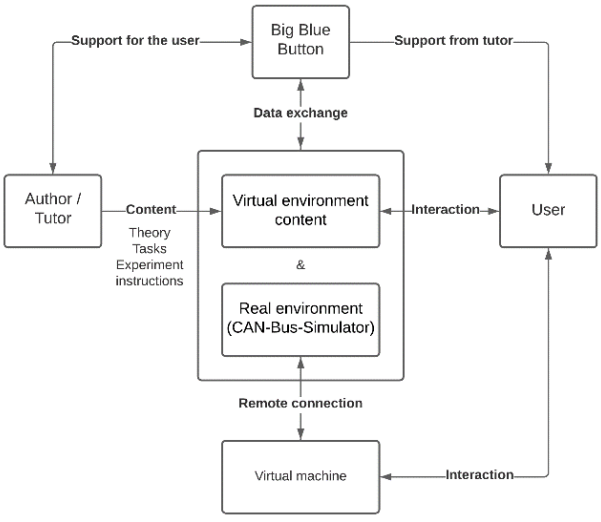



Figure 11. Structure of the virtual and real learning environment, for conducting remote seminars in the field of automotive CAN bus systems.

The target group of the forensics seminars has the possibility to follow implemented tasks on the simulation software via web access. This creates a link between the virtual and the real environment between the learner and the development environment. Support is provided by the tutor, who is on site in the learning lab and can interact with the participants via a conference tool. BigBlueButton was used as the conference tool in this setup. Furthermore, the learners receive the theoretical content via the conference tool. Figure 11 shows an approach in which a virtual machine is accessed via a remote connection. Using this virtual machine, the participants can send CAN commands to the model vehicle. The CAN commands are distributed by the CAN gateway to the individual ECUs and implemented by the hardware. The learner can observe the result of the output via a webcam and thus obtain a direct control of success.

The AnyDesk software is particularly suitable, as different participants can access a desktop at the same time and thus work on a task as a group [36]. Alternatively, the remote software TeamViewer with the same functionality can be used. The remote desktop under Windows is not suitable, as only one user can log in and the on-site tutor has no access to the system. The need for at least two seminar participants and the lecturer to access a desktop is justified by the realisation of group work in the online seminar and the support of the individual groups by the lecturer.

A Linux environment is absolutely necessary for the Instrumentation Cluster Simulator (ICSIM). For this purpose, a Virtual Machine (VM) in Oracle VitualBox was provided for the seminar. In this VM, the drivers for the PCAN-USB adapter, the CANUTILS and the ICSIM are pre-installed to realise a quick use of the tools. This VM can be controlled by the participants via the remote software. The Instrumentation Cluster Simulator (ICSIM) software, shown in Figure 12, is used as a simulation environment for the automotive CAN bus [15]. This simulator is used to display a virtual dashboard with speedometer, indicators and virtual doors, which can be controlled via a control interface.

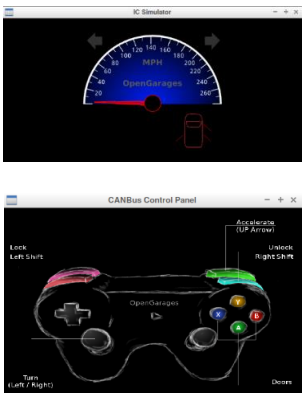



Figure 12. Dashboard simulation for graphical representation of the transmitted CAN commands and control interface for transmitting CAN commands

The simulator can be run under Linux. Within the Linux environment, the CANUTILS are used. The CAN bus is configured in the first part as a virtual CAN bus with the following commands:

- sudo modprobe can
- sudo modprobe vcan
- sudo ip link add dev vcan0 type vcan
- sudo ip link set up vcan0

The respective components of the simulator and the analysis software cansniffer are configured with the commands:

- ./icsim vcan0
- ./controls vcan0
- cansniffer -c vcan0

In the practical exercise, the development board AT90CAN128 from Olimex is connected to this PCAN-USB interface. At the start of the exercises, the firmware of the development board is programmed with a prefabricated firmware, which realises the control of the tachometer in the simulator dashboard. In the further course of the practical exercises, the prefabricated firmware is to be extended and tested with the CAN messages that were determined in the first part. These additional CAN messages can then be used, for example, to control the turn signals and also the doors in the simulation environment via the microcontroller board. Microchip Studio is used as the development environment at this point. This environment is used in the Windows environment of the practical computer, as it is easy and clear to create and debug the microcontroller firmware.

In principle, the Linux environment with integrated CAN simulator can be run independently on a computer as a ready-made virtual machine for VirtualBox or VMware. The computer is made available to the user for an online practical course via remote software. This could be done using Windows Remote Desktop, but this way only one user is able to log in to the computer. However, the aim of the practical session is to enable group work consisting of at least 2 users, whereby the instructor on site should also have access to the desktop in order to be able to monitor the work of the respective groups. For this purpose, Teamviewer and Anydesk were evaluated as remote applications. Both applications enable the realisation of the described online learning scenario.

To make the practical experience of CAN bus programming more tangible, a CAN bus demonstrator in the form of a remote-controlled model vehicle is used. The model car was developed for this simulation environment based on the Robot Car Kit from Joy-It. Figure 13 demonstrates the vehicle.

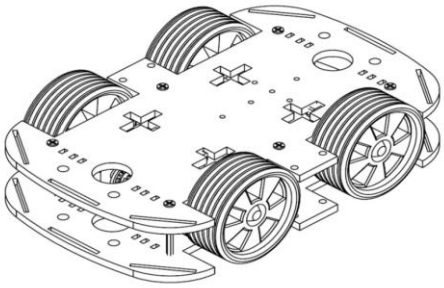


Figure 13. Schematic representation of the Robot Car Kit from Joy-It. [30]

The basic framework for the realisation is the Robot Car Kit from Joy-It including four electric motors. In addition, there is an EVAL6207N evaluation board from STMicroelectronics. Since the demonstrator is to be equipped with a CAN bus, another microcontroller that can handle CAN is needed. With the AVR CAN board from Olimex, the EVAL6207N can receive signals for control. Headlights and turn signals that mimic those of a vehicle are to be added to the setup. The demonstrator is controlled by CANalyzer software, which is used to generate appropriate CAN messages. To transmit messages from a computer with the software to the control unit (AVRCAN board) of the demonstrator, the CAN interface VN1610 from Vector is required. The actual CAN bus exists between the CAN interface and the AVR CAN board. Figure 14 schematically represents the basic structure of the CAN demonstrator.

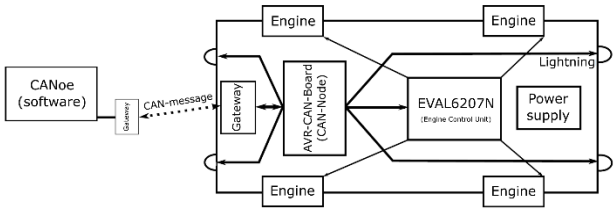



Figure 14. Schematic representation of the complete CAN-Demonstrator

Using the CANoe tool, CAN commands can be sent to the demonstrator via the gateway on a specially created interface. After being received by the gateway, the CAN bus commands are forwarded by the AVR CAN board to the respective control units and implemented there. Figure 15 demonstrates the panel for the experimental setup. Figure 16 shows the complete CAN demonstrator hardware.

Figure 15. Interface to control the CAN-Demonstrators inside the CANoe-Tool

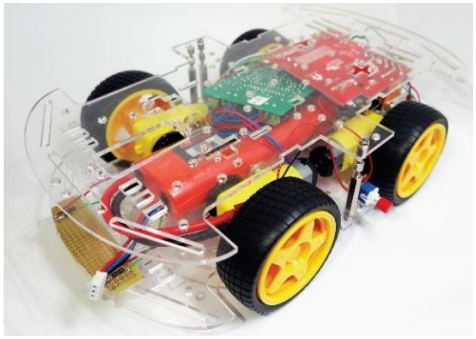


Figure 16. Complete model of the CAN-Demonstrators

## IV. CONCLUSION

In this paper, it was shown that a real CAN system can be simulated with the help of the presented tools. In addition, it is possible to use this simulation environment to analyse and carry out cyberattacks on vehicle CAN bus systems. A methodology and a simulation environment were described which, in addition to the classical lecture and audiovisual media, also integrate the practical part with visual feedback. The simulation environment offers the possibility to manipulate the CAN bus of the model car from the outside. Students of general and digital forensics have already been able to test this setup. The feedback was consistently positive. In particular, solving practical tasks on the real CAN bus demonstrator was mentioned positively and motivated the participants to come up with creative solutions, despite the technological obstacles, when integrating it into an online learning environment. In addition to using this in the field of car forensics, it is conceivable that other subject areas can be applied with special hardware. Use in the area of IoT (Internet of Things) forensics is conceivable here. Furthermore, it is possible to use the setup to realise more attack vectors on automotive systems. For example, the CAN bus demonstrator can be equipped with a keyless go system, which would allow replay station attacks to be realised [18]. Additionally, it would be possible to implement an intrusion detection system to filter out malicious messages in the data stream of the CAN bus. With the solution shown, challenges could be solved to be able to integrate Windows and Linux tools in an online seminar.

## REFERENCES


[1] The 2020 Digital Auto Report, PwC Network, last access date: 21.05.2021, online available from: https://www.strategyand.pwc.com/de/de/studie/2020/digital-auto-report-2020.html

[2] Consumer WatchDog, “Kill Switch – Why connected cars can be killing machines and how to turn them off“, last access date: 21.05.2021, online available from: https://www.consumerwatchdog.org/sites/default/files/2019-07/KILL%20SWITCH%20%207-29-19_0.pdf

[3] B. Sam, “Consumer Connected Cars: Telematics, In-vehicle Apps & Connected Car Commerce 2018-2023.” Hg. v. Juniper Research Ltd., last access date: 21.05.2021, online available from: https://www.juniperresearch.com/press/press-releases/in-vehicle-commerce-opportunities-exceed-775mn, 2018.

[4] J. Bird, “Car hacking threatens vision of connected mobility”. Hg. v. Financial Times LTD (The Future of the Car), last access date: 21.05.2021, online available from: https://www.ft.com/content/163f08c6-6ce3-11e9-9ff9-8c855179f1c4, 2019.

[5] Upstream, “Upstream Security's 2020 Global Automotive Cybersecurity Report.” Hg. v. Upstream Security Ltd. Last access date: 21.05.2021, online available from: https://www.upstream.auto/upstream-security-global-automotive-cybersecurity-report-2020/, 2019.

[6] J. Lalley and R. Miller “The learning pyramid: does itpoint teachers in the right direction.” Education, vol. 128, no.1, pp. 64-79, 2007.

[7] PEAK-Systems PCAN, last access date: 21.05.2021, online available from: https://www.peak-system.com/typo3temp/pics/61ef9f60a0.jpg

[8] Red Pitaya System overview, last access date: 21.05.2021, online available from: https://redpitaya.readthedocs.io/en/latest/developerGuide/software/sysOver.html

[9] OBDII Interface Lescars, last access date: 21.05.2021, online available from: https://images-na.ssl-images-amazon.com/images/I/811dSmBKHfL._AC_SL1300_.jpg

[10] OLIMEX Development Board, last access date: 21.05.2021, online available from: https://www.olimex.com/Products/AVR/Development/AVR-CAN/images/thumbs/310x230/AVR-CAN-01.jpg

[11] Elmelectronis ELM327DS, last access date: 21.05.2021, online available from: https://www.elmelectronics.com/wp-content/uploads/2016/07/ELM327DS.pdf

[12] ATMEL ICE, last access date: 21.05.2021, online available from: https://www.microchip.com/DevelopmentTools/ProductDetails/ATATMEL-ICE

[13] RedPitaya Docs, last access date: 21.05.2021, online available from: https://redpitaya.readthedocs.io/en/latest/

[14] RedPitaya DevGuide, last access date: 21.05.2021, online available from: https://redpitaya.readthedocs.io/en/latest/developerGuide/devGuideTop.html

[15] Instrumentation Cluster Simulator (ICSIM), last access date: 21.05.2021, online available from: https://github.com/zombieCraig/ICSim

[16] Number of common IT security vulnerabilities and exposures (CVEs) worldwide from 2009 to 2019, last access date: 21.05.2021, online available from: https://www.cvedetails.com/browse-by-date.php

[17] R. Buttigieg, M. Farrugia and C. Meli, “Security Issues in Controller Area Networks in Automobiles”, 2017.

[18] A. Francillon, B. Danev and S. Capkun, “Relay Attacks on Passive Keyless Entry and Start Systems in Modern Cars”, IACR Cryptology ePrint Archive. 2010. p. 332., 2010.

[19] K. Koscher et al., “Experimental Security Analysis of a Modern Vehicle.” Hg. v. IEEE. 2010 IEEE Symposium on Security and Privacy. Berkeley/Oakland, 2010

[20] F. Maggi, “A Vulnerability in Modern Automotive Standards and How We Exploited” It. Hg. v. TrendLabs Security Intelligence Blog. TREND MICRO. Bonn, last access date: 21.05.2021, online available from: https://documents.trendmicro.com/assets/A-Vulnerability-in-Modern-Automotive-Standards-and-How-We-Exploited-It.pdf, 2017.

[21] C. Miller and C. Valasek, “Remote Exploitation of an Unaltered Passenger Vehicle.” , last access date: 21.05.2021,

online available from: http://illmatics.com/Remote%20Car%20Hacking.pdf., 2015.

[22] Cambridge Dictionary – forensic , last access date: 21.05.2021, online available from: https://dictionary.cambridge.org/dictionary/english/forensic

[23] S. Checkoway et al., “Comprehensive experimental analyses of automotive attack surfaces”, Proceedings of the 2011 Usenix Security, 2011.

[24] K. Koscher et al., “Experimental security analysis of a modern automobile.”, in IEEE Symposium on Security and Privacy, pp. 447–462, 2010.

[25] Peaks Systems Driver, last access date: 21.05.2021, online available from: https://www.peak-system.com/fileadmin/media/linux/index.htm

[26] ODB ELM327, last access date: 21.05.2021, online available from: https://www.obd-2.de/elm327.html

[27] Mikrocontroller Community, last access date: 21.05.2021, online available from: https://www.mikrocontroller.net/topic/98697

[28] D. Kolb, “Experiential Learning: Experience as the Source of Learning and Development.” New Jersey: Prentice Hall, 1984.

[29] P. Race, “Learning in small groups”, last access date: 21.05.2021, online available from: https://www.heacademy.ac.uk/system/files/resources/id475_learning_in_small_groups_Race.pdf

[30] Joy-It Robot Car Kit 01, last access date: 21.05.2021, online available from: https://cdn-reichelt.de/documents/datenblatt/A300/ANLEITUNGROBOT03.pdf

[31] T. FIschl, “USBtin - USB to CAN interface”, last access date: 21.05.2021, online available from: https://www.fischl.de/usbtin/

[32] CAN USB Adapter – isCAN USB, last access date: 21.05.2021, online available from: https://www.thorsis.com/de/can/can-usb-adapter-iscan-usb/

[33] PCAN-Developer 4, last access date: 21.05.2021, online available from: https://www.peak-system.com/PCAN-Developer-4.461.0.html

[34] Microchip Studio for AVR® and SAM Devices, last access date: 21.05.2021, online available from: https://www.microchip.com/en-us/development-tools-tools-and-software/microchip-studio-for-avr-and-sam-devices

[35] Vector CANoe, last access date: 21.05.2021, online available from: https://www.vector.com/int/en/products/products-a-z/software/canoe/

[36] AnyDesk, last access date: 21.05.2021, online available from: https://anydesk.com/en

 ISBN: 978-1-61208-879-2 

# Hybrid Neural Network Modeling for Multiple Intersections along Signalized Arterials - Current Situation and Some New Results

Wan Li, Chieh Wang, Yunli Shao, Hong Wang
National Transportation Research Center
Oak Ridge National Laboratory,
Oak Ridge: TN37934, USA
e-mail: {w5i; cwang; shaoy; wangh6}@ornl.gov

Guohui Zhang, Tianwei Ma
Department of Civil and Environmental Engineering
University of Hawaii at Manoa
2540 Dole Street, Honolulu, Hawaii 96822, USA
e-mail: {guohui; tianwei}@hawaii.edu

Jon Ringler
Econolite Systems
1250 N. Tustin Ave.
Anaheim, CA 92807, USA
e-mail: JRingler@econolite.com

Danielle Chou
Vehicle Technologies Office
US Department of Energy
1000 Independence Ave SW
Washington, DC 20585, USA
email: danielle.chou@ee.doe.gov

***Abstract*—Traffic flow along signalized arterials is a dynamic, nonlinear, and stochastic system in which the relationship between the signal timing plan and traffic delays is too complicated to be modeled using first principles approaches. With advances in sensing technologies, various data sets are available, allowing effective data-driven modeling to be conducted for further controller design. In this keynote paper, a Hybrid Neural Network (HNN) is proposed to model the multiple intersections along a signalized arterial in Honolulu, in which both modeling structure and the relevant training algorithms have been developed. HNN modeling using real data has shown a set of promising results, with dynamic model performance assessed using model error Probability Density Function (PDF). A simple HNN model can easily be used as a starting point for an artificial intelligence–based closed-loop control design that controls the signal timing to reduce the traffic delay.**

***Keywords - signalized intersections; modeling; neural networks; performance analysis; signalized arterials simulation.***

## I. INTRODUCTION

The nature of the traffic flow system in signalized arterials can be represented as a dynamic and stochastic system [2] – [5] for which the inputs are the traffic demand and signal timing at each intersection, and the outputs are the traffic flow status (e.g., travel delays, queue length, and traffic flow speed) and energy consumed when vehicles pass through the arterial. Since the traffic demand and traffic flows (number of vehicles and their compositions) are random, the system is stochastic in nature. This is a Multi-Input and Multi-Output (MIMO) stochastic dynamic system. If it is in the continuous-time domain, its solution is obtained using partial differential equations induced from the well-known *Ito* stochastic differential equations with random boundary conditions. The solution for such a complicated model is quite difficult to obtain, and it frequently must be solved using high-performance computing, which generally cannot be used for real-time control design and implementation. Therefore, data-driven modeling methods—in particular, those widely used in Artificial Intelligence (AI) technology—are regarded as effective ways to establish simple dynamic models between signal control and traffic flows so that system performance can be controlled and optimized in real time. The advantage of using AI-based models is that these models can be adaptively learned using evolving real-time data. As a result, the use of neural network modeling has been a subject of study for many years.

Indeed, advances in wireless-driven vehicular communications have greatly facilitated modeling exercises, and emerging cooperative intelligent transportation control system operations have enabled many smart traffic control and management applications to improve traffic safety and operational efficiency [1]. Vehicle-to-Everything (V2X) communications allow vehicles to communicate with other vehicles (vehicle-to-vehicle); infrastructure (vehicle-to-infrastructure); pedestrians, bicyclists, and devices (vehicle-to-device); and internet through cellular networks and/or dedicated short-range communication technologies. The information exchanges supported by V2X communication systems can be used to effectively balance traffic demand distribution among traffic networks and facilitate traffic flow progression. With these new data available in a real-time format, it is now possible to further enhance AI-based modeling, and ultimately control, to optimally coordinate signal controls for traffic flow systems along arterials.

In addition, for stochastic modeling of traffic flow systems, one of the important criteria is the reliability of and confidence in the obtained models for control and optimization. Thus, not only do the models need to be built using real-time input and output data, but also there is a need to ensure that the model so obtained is reliable and has a high level of confidence interval for users. In this context, the use of modeling error entropy, or its Probability Density Function (PDF), should be considered as the modeling objective function to be minimized. Ideally, a narrowly distributed modeling error

 ISBN: 978-1-61208-879-2 

PDF centered at zero mean would indicate that the models obtained have high reliability and confidence intervals. This is exactly its novelty compared with existing AI-based models for transportation systems, in which only sum-squares-error has been used to judge whether the obtained model is good or not. The method of using modeling error entropy and PDF to perform online adaptive learning was established several years ago [1], and this approach can be applied in combination with the existing AI modeling tools to establish reliable and robust AI-based models for the traffic flow system.

Based upon the above analysis, it can be seen that the following challenges remain in terms of AI-based modeling and control for signalized intersections along arterials and the urban grid road network:

- Although the theory of AI-based modeling and control for signal control is maturing, the field testing and closed-loop control implementation for a large number of intersections is still limited because of the insufficient real-time data for fast feedback control realization.
- The existing AI-based modeling for transportation systems cannot yet capture the nonlinear and dynamic stochastic nature with high reliability and robustness.
- Guaranteed control performance for energy minimization is still lacking.

In this effort, neural network modeling was studied for signalized intersections along an arterial in Honolulu using the real-time data from the system. A Hybrid Neural Network (HNN) model, which is a subset of neural networks, was constructed, and its learning algorithm was established. A comprehensive assessment of the modeling effort was conducted using least squares and gradient approaches.

The rest of this paper is organized as follows. Section II summarized the literature review on traffic signal control problems with neural network models. Section III describes the system structure and the forms of dynamic models that represent the relationship between the traffic delays and signal timing plans. Section IV presents the linear modeling using recursive least squares to show the nonlinearities of the system. Section V addresses the formulation of HNN and defines its inputs and outputs together with the formulation of training algorithms for both linear and nonlinear parts. The modeling results and modeling performance analysis for an arterial with seven signalized intersections are also discussed in this section. The conclusions and acknowledgement close the article.

## II. RELATED WORK

Traffic system modeling aims to establish linear or nonlinear relationships between traffic states—e.g., traffic volume, travel time (travel delay), and travel speed—given spatiotemporal traffic information. Most studies leverage a single data source. For example, the objective is to predict near-term traffic flow given historical traffic flow data. Other studies using multiple data sources need to capture dominant dependencies between different features. For example, Ke et al. [6] developed a model to predict lane-based traffic speed based on speed and traffic volume data. Transportation system modeling techniques can be divided into two categories: *non-learning* based and *learning* based methods [7]. For example, classical *non-learning-based* methods include autoregressive integrated moving average [8] and K-nearest neighbors [9]. These models are usually more interpretable but cannot capture the spatial correlations of traffic states. Moreover, they are not appropriate for nonstationary data. *Traditional learning-based* methods include regression [10], Kalman filter [11], and support vector machine [12]. These methods are generally more effective than non-learning-based models. However, they usually fail to capture the nonlinear spatiotemporal correlations of traffic data. Nowadays, we have more data sources and increasing computational power, so more *advanced learning*-based methods, e.g., different types of neural networks, have shown promising performance. The most commonly used neural networks for transportation system modeling include Artificial Neural Networks (ANN) [13], Long Short-Term Memory (LSTM) [14], Convolutional Neural Networks (CNNs) [6], and Graph-based Neural Networks (GNN) [15]. Compared with ANNs, CNNs and LSTMs have advantages in capturing nonlinear spatial and temporal dependencies of traffic features. However, their limitations become obvious when the transportation network is very large. GNNs are proved to be powerful tools for large-scale traffic signal control systems. GNNs can extract features from graph-structured data and predict future traffic states in an efficient and effective manner.

With the established dynamic stochastic models for transportation system, the next step is to develop real-time optimal control strategies to reduce travel delay and energy consumptions. Conventional traffic control methods for multiple intersections in a network, such as SCOOT [16], GreenWave [17], SOTL [18], Max-pressure [19], and SCATS [20], usually assumed a simplified traffic conditions with complete traffic information available, e.g., pre-defined traffic flows and driving behaviors. Hence, they are not applicable for real-world traffic control for multiple intersections.

For a large-scale traffic system, it is usually a difficult task to predict the effects of modifying signal timing parameters due to the nonlinear and stochastic nature in a traffic network. Comparing to the conventional signal control methods, Neural Network (NN)-based signal control methods can address the challenges on traffic system modeling and traffic signal optimization. The studies from [21][22] tested a NN-based controller for single intersections. Both studies applied the concept of fuzzy logic and their NNs are five-layer type, e.g., input, fuzzification, inference, consequence, and defuzzification. They used number of vehicles passing the intersection and number of vehicles waiting in the queues as inputs and the outputs are the traffic signal plans. In [22], reinforcement learning and gradient descent method were applied to update the shape of fuzzy membership functions by computing the weights of the NN. The advantages of NN models are more obvious in a larger network. Srinivasan et al. [23] developed a distributed unsupervised traffic responsive

 ISBN: 978-1-61208-879-2 

signal control method for traffic signal control and coordination. Each agent is a local traffic controller for one intersection. They integrated the simultaneous perturbation stochastic approximation theorem in fuzzy NN. Stochastic approximation is a commonly used technique in stochastic optimization for online wight updates in NN. It is usually preferable when the gradient of the loss function is not readily available. The proposed model was tested in a traffic network with 25 intersection in Singapore. The results demonstrated that the model could be used to obtain the controller that reduces significant amount of travel delay. Choy and Srinivasan [24] further improved the study [23] by developing a HNN model with multistage online learning process to solve the distributed traffic signal control problem with an infinite horizon. It is challenging to calculate the analytical optimal solution for the distributed control problems. This study applied an approximation technique, receding-horizon limited-memory, for to approximate optimal solution. Each local signal controller was made up of a five-layered fuzzy NN that aimed at computing the optimal signal plans. Experiment results suggested that HNN model was effective and efficient in solving the large-scale traffic signal control problem in a distributed control manner. There are several limitations in NN-based traffic signal control algorithms. First, as mentioned by [22], NN learning is not efficient under complex continuous system because of lack of stochastic exploration. Second, learning process is usually too long to be implemented in real time in the field.

Recently, Reinforcement Learning (RL) models have been studied extensively and made impressive progress in traffic control domains. RL can learn from observed data and adapt to real-time changes of traffic demands. RL is a trail-and-error learning process without making any unrealistic assumptions on traffic system modeling. There are four key components in Decentralized Reinforcement Learning (DRL): agent, environment, state, and reward. In transportation system, environment is often defined as traffic conditions and state is a feature representation of the environment. DRL will have an agent for each intersection to learn a model and predict whether current signal phase should be changed or not. The decision will be implemented in the environment and the reward (travel delay, vehicle throughput, or energy efficiency) is sent back to the agent to help it improve the decision-making process. The key challenges in RL are (i) how to describe the environment quantitively, (ii) how to model the relationships between decision (signal timings) and reward (traffic states) due to its exponentially expanding complexities; and (iii) how to implement coordination and information sharing between multiple agents/intersections. There are generally two categories of RL: model-free and model-based RL. To successively apply model-based approach, the transition function (predict next state given current action) must be known. However, it is usually difficult to obtain it in real-world. Model-free RL directly estimate the reward given state-action pairs and select the optimal action accordingly [32]. Hence, model-free RL, e.g., Q-learning and SARSA, are commonly used in traffic signal control problems. For model-free RL, exploration is required to gain knowledge by sampling. Model-free RL can be categorized as value-based and policy-based methods [33]. Value-based RL learning the value function (or a generalization called the Q-function) and policy-based methods directly learn the optimal policy or approximate optimal policy. Comparing to the traditional reinforcement learning approach whose states need to be discretized and low-dimensional, DRL can handle high dimensional input data, e.g., image, and learn functions to extract useful information and approximate policy from input states. By combining deep learning with reinforcement learning, it addresses the "curse of dimensionality" issue, helps to improve the model scalability, and reduce learning time. Li et al. [25] set up a Deep Neural Network (DNN) to learn the Q-function of DRL from the sampled traffic states (inputs) and the corresponding traffic conditions (outputs). The objective is formulated as a Q-function which aims to maximize the future rewards given the current state and action. Instead of relying on a conventional Q-table, they used the deep Stacked Autoencoders (SAE) neural network to estimate Q-function. Comparing to the conventional reinforcement learning approaches, their DRL can reduce delay by 14% and largely reduce number of vehicles stops at intersections. Wei et al. [26] developed a DRL model for traffic signal control with real-world large data set. In their method, traffic condition is extracted from an image. The image is directly used as an input for a CNN model to supplement other hand-crafted traffic features (queue length, waiting time, and number of vehicles) of environment. They applied an offline model to test different signal timing plans and collect data samples of signal timings and traffic conditions. After that, an online model will determine the optimal action to take (change signal status or not). Their model was tested on a large-scale real traffic dataset from surveillance cameras. Motivated by Max Pressure (MP) control, Wei et al. [27] developed a reinforcement learning approach for large-scale road network. In RL, the objective is to maximize the long-term rewards by trial-and-error search while MP aims at minimizing pressure by greedy algorithm. This study set the reward function in RL the same as the objective of ML so that they can achieve the same result as MP to maximize vehicle throughput. As claimed by the authors, this is the first study that applies individual RL model and achieves coordination without any prior knowledge. Chen et al. [28] designed a DLR model for a city-level network with more a thousand intersections. DRL for multi-intersection control and coordination is quite a difficult problem due to the scalability and data feasibility. They incorporated DRL agent design with pressure, e.g., different of queue length at downstream and upstream intersections. The DRL agent aims at balancing the distribution of vehicles in the traffic network and maximize the system throughput. They tested their proposed model with Manhattan dataset containing signalized 2510 traffic lights. Comparing to other state-of-the-art signal control methods including fixed time, max pressure, and different variations of reinforcement learning methods, their proposed model was proved to generate shorter travel time and larger vehicle throughput.

 ISBN: 978-1-61208-879-2 

Based on the literature, most studies used average queue length, average waiting time, average speed, and vehicle throughput as reward to evaluate an action in RL. There are various kinds of measures to describe environment states, e.g., queue length, waiting time, speed, and signal phases for each lane or for a road segment. Traditional RL use a tabular or linear model to approximate the state function to improve efficiency [31]. However, the state space in real world is usually very large which limit the capability of traditional RL. With the development of deep learning, DRL models can handle the large state space. For example, recent studies use images as state where vehicle trajectories and queue length can be extracted [26][27] for state representation. The action in RL relates to signal phases changes. It can be the ratio of signal phase duration over the total cycle length [31] or an indicator to decide if an different signal phase should be activated to green [26]. Most of the traffic signal control studies with RL use value-based methods which usually requires discrete actions. The model takes the state presentation as input and parameterized by neural networks.

Although DRL model improves traffic signal control in the complex transportation systems, it treats neighboring intersections as the same and fail to model the spatial dependencies of traffic flows. Different intersections should be modeled carefully in realistic transportation network. To address the issue mentioned above, graph neural networks are proved to be an effective tool to capture the traffic dynamics in large-scale transportation network. The transportation system can be model by a graph consisting of nodes and edges. GNN can handle inputs given on general graphs. Wei et al. [29] proposed a model, *CoLight*, to control traffic signals on a large-scale road network with hundreds of intersections. They applied a graph attentional network to facilitate communication between intersections and consider the temporal and spatial influences of neighboring intersections. The model leverages the attention mechanism to model the influence of upstream and downstream intersections on the target intersection by learning different weights for different intersections. Extensive experiments have been conducted using synthetic and real-world data. Their proposed model outperformed other state-of-the-art methods in terms of reducing average travel time. Zhong et al. [30] developed a probabilistic graph neural network for traffic signal control and cooperation. They used decentralized reinforcement learning to model the transportation system. A graph attention module was then applied to learn dependencies of neighboring intersections. Finally, a graph inference model was proposed to learn the latent representations of adjacent intersections by considering traffic uncertainties. Their model can characterize the posterior with respect to latent variables and allow Bayesian inference. The rationality of model design can be explained by transportation theory.

Coordination is essential for large-scale transportation system with multiple intersections. Wei et al. [34] categorized traffic signal control and coordination problems into three categories: joint action learners, independent (distributed) learners without communication and distributed learner with communication. Joint learners use a single centralized agent to control all intersections [34]. This approach could lead to the curse of dimensionality that the state-action space will grow exponentially as the number of intersections increases. Unlike joint agent, each distributed agent control one intersection. If communication does not exist between distrusted agent, each agent observes its own local environment. This method usually does not perform well when the environment becomes complicated. Distributed learning with communication allows agent to share information on their observations. Graph-based NN model for traffic signal control problems can learn the communication from the message passing on the graph. TABLE I summarizes the representative NN-based traffic signal control studies based on a few characteristics we discussed above.

TABLE I. REPRESENTATIVE NN-BASED TRAFFIC SIGNAL CONTROL STUDIES

| Reference | Method | Traffic features | Coordination | Road Network | # of Intersections |
|---|---|---|---|---|---|
| Wei and Zhang [21] | Fuzzy neural network | # of vehicles; queue length | No communication | Synthetic | 1 |
| Bingham [22] | Neurofuzzy traffic controller | # of vehicles; queue length | No communication | Synthetic | 1 |
| Srinivasan et al. [23] | Fuzzy neural network with stochastic approximation theorem | Traffic flow; occupancy | Distributed control with communication | Real (CBD Singapore) | 25 |
| Choy et al. [24] | HNN with reinforcement learning and evolutionary algorithm | Traffic flow; occupancy | Distributed control with communication | Real (CBD Singapore) | 25 |
| Li et al. [25] | Value-based reinforcement learning | Queue length | No communication | Synthetic | 1 |
| Wei et al. [26] | Value-based reinforcement learning | Queue length; # of vehicles; waiting time; Image | No communication | Synthetic | 1 |
| Wei et al. [27] | Value-based reinforcement learning with max pressure control | # of vehicles | No communication | Real (New York City) | 16 |
| Chen et al. [28] | Value-based reinforcement learning | # of vehicles | No communication | Real (New York City) | 2510 |
| Wei et al. [29] | Graph attention network for cooperation | Queue length | With communication | Real (New York City) | 196 |
| Zhong et al. [30] | Probabilistic graph neural network | Queue length, # of vehicles | With communication | Real (Hangzhou) | 16 |

 ISBN: 978-1-61208-879-2 

In addition, the research team at the University of Hawaii has extensively developed machine learning-based approaches address various traffic data analysis and formulation issues. For example, we to estimate vehicle classification volumes based on single-loop detector outputs [36]. The proposed ANN has three layers with back-propagation architecture. Vehicle classification categories employed by this study were consistent with the four-bin classification system currently used by the Washington State Department of Transportation (WSDOT) dual-loop detection system. To achieve the best bin volume estimates, a specific neural network is designed and configured for each vehicle category. The proposed ANN is trained and tested using data collected from loop detector stations on I-5 in the greater Seattle area. Our test results indicate that the proposed ANN method worked stably and effectively for the studied stations. The estimated bin volumes were reasonably accurate and can be applied to transportation practice. The temporal and spatial transferability tests showed that the proposed ANN is robust and can be applied to estimate bin volumes during different time periods and at different loop stations on I-5 without introducing significant errors.

Work in [37] conducted a study to develop a Deep Learning (DL) framework to predict the taxi-passenger demand while the spatial, the temporal, and external dependencies were considered simultaneously. The proposed DL framework combined a modified density-based spatial clustering algorithm with noise (DBSCAN) and a conditional generative adversarial network (CGAN) model. More specifically, the modified DBSCAN model was applied to produce a number of sub-networks considering the spatial correlation of taxi pick-up events in the road network. And the CGAN model, fed with the historical taxi passenger demand and other conditional information, was capable to predict the taxi-passenger demands. The proposed CGAN model was composed of two LSTM neural networks, which are termed as the generative network G and the discriminative network D, respectively. Adversarial training process was conducted to the two LSTMs. In the numerical experiment, different model layouts were compared. It was found that different network layouts provided reasonable accuracy. With limited training data, more LSTM layers in the generator network resulted in not only higher accuracy, but also more difficulties in training. Comparisons were also conducted between the proposed prediction model and four typical approaches, including the moving average method, the autoregressive integrated moving method, the neural network model, and the LSTM neural network model. The comparison results showed that the proposed model outperformed all the other methods.

Another research effort undertaken in [38] is to investigate how the integration of clustering models and deep learning approaches can learn and extract the network-wide taxi hotspots in both temporal and spatial dimensions. A Density Based Spatiotemporal Clustering Algorithm with Noise (DBSTCAN) was established to extract the historical taxi hotspots, which changed with time. A conditional generative adversarial network with Long Short-term Memory Structure (LSTM-CGAN) model was proposed for taxi hotspot prediction, which is capable of capturing the spatial and temporal variations of taxi hotspots simultaneously. Specifically, the DBSTCAN was applied to process the large-scaled geo-coded taxi pickup data into time-varying historical hotspot information. The proposed LSTM-CGAN model was then trained by the network-wide hotspot data. As illustrated in the numerical tests, it was found that the proposed LSTM-CGAN model provided comparable results with different model layouts and model with 4 LSTM layers in both generator and discriminator performed best. The comparison results indicated that the proposed LSTM-CGAN model outperformed all these benchmark methods and demonstrated great potential to enable many shared mobility applications.

Work in [39] reported a novel multi-agent reinforcement learning method, named as Knowledge Sharing Deep Deterministic Policy Gradient (KS-DDPG) to achieve optimal control by enhancing the cooperation between traffic signals. By introducing the knowledge-sharing enabled communication protocol, each agent can access to the collective representation of the traffic environment collected by all agents. The proposed method is evaluated through two experiments respectively using synthetic and real-world datasets. The comparison with state-of-the-art reinforcement learning-based and conventional transportation methods demonstrates the proposed KS-DDPG has significant efficiency in controlling large-scale transportation networks and coping with fluctuations in traffic flow.

Based upon the above analysis, it can be seen that there are still following challenges on NN based modeling and control strategies for networked signal-timing control:

*1)* The modeling using data driven reqires a good set of data in real-time;
*2)* In therms of control strategies, there is a need to structure the control model so that it can be easily implemented in real-time. A affine type of dynamic model would be a choice. This will be described in the following sections;
*3)* Most studies have been focussed on simulations and real-time 24/7 implementation is lacking.

These challenges constitute research questions to be answered and therefore in the following sections, a novel modeling and control, namely the HNN modeling and control developed by the authors, will be described that summarizes the authors recent work on multiple signalized intersection control.

 ISBN: 978-1-61208-879-2 

## III. TRAFFIC FLOW SYSTEM DESCRIPTION

Figure 1 shows the signalized arterials to be modeled, where the input is the signal timing plan at each intersection and the output is the traffic delays of different phases (left turns, right turns and straight movements).

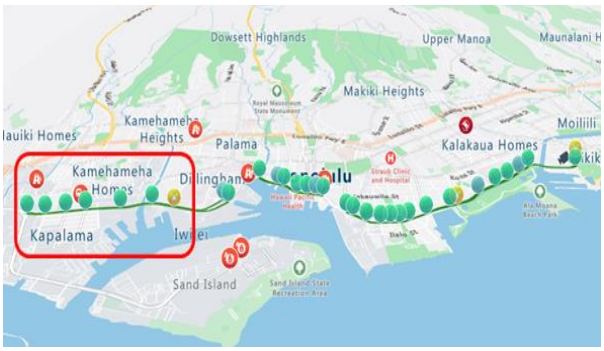



Figure 1. The signalized arterial in Honolulu.

The objective is to build up dynamic models that reflect the dynamics of the system; the data used were collected from Econolite systems.

Taking $u(k)$ as the input and $y(k)$ as the output vector that is composed of the signal timing plan (i.e., green light time duration under fixed cycle length) and the traffic delays for each phase (i.e., through movements, left turns, and right turns) at an intersection respectively, the dynamics of the system can be generally modeled as follows

$$y(k+1) = f\big(y(k), u(k), \omega(k)\big) \tag{1}$$

where $f(\dots)$ is the nonlinear vector function representing the system dynamics, $\omega(k)$ is the random noise term, and $k$ is the sample number, which can be a multiplication of cycle duration in signal timing control.

## IV. LINEAR MODELING

To perform the required data-driven modeling, it was imperative to first check whether the system could be truly represented as a nonlinear system. To answer that question, we performed linear modeling initially. Indeed, if the system was linear, then the modeling error should have a Gaussian-like distribution. Otherwise, the system should be regarded as a nonlinear system in which neural network modeling and other nonlinear system modeling need to be considered to build reliable models for the system.

### A. Modeling structure

When the system is linear, the following simple model can be assumed for each intersection

$$y(k+1) = ay(k) + bu(k) + \omega(k) \tag{2}$$

where $\{a, b\}$ are unknown parameters to be estimated, $\omega(k)$ is noise, and the modeling exercise is to use available data $\{u(k), y(k)\}$ to estimate the parameters $\{a, b\}$. This is a standard application of least squares estimation. For this purpose, denote

$$\theta = \begin{bmatrix} a \\ b \end{bmatrix}, \varphi(k) = \begin{bmatrix} y(k) \\ u(k) \end{bmatrix} \tag{3}$$

Then, the following recursive least squares algorithm is used to estimate $\{a, b\}$ using the data collected from the Econolite/University of Hawaii platform

$$\begin{gathered} \theta(k+1) = \theta(k) + \frac{P(k)\varphi(k)\varepsilon(k)}{1+\varphi^T(k)P(k)\varphi(k)} \\ \varphi^T(k) = [y(k)\ u(k)] \\ \epsilon(k) = y(k+1) - \theta^T(k)\varphi(k) \\ P^{-1}(k+1) = P^{-1}(k) + \varphi(k)\varphi(k)^T \end{gathered} \tag{4}$$

where $\theta(k)$ is the estimate of $\theta$ at sample time $k$ (of every five cycles), $P(k)$ is the variance matrix, with the initial conditions being $\theta(0) = 0$, $P(0) = 100I_{2\times 2}$.

It can be seen that (4) is a typical recursive least squares algorithm with the maximum forgetting factor as the linear modeling here is to just test whether the system in nonlinear or time varying so as to justify the use of nonlinear system model. A "less-than-one" forgetting factor can also be used in order to track time-varying feature of the system. This allows the estimation algorithm to be adaptive and robust with respect to changes of the system such as operational environmental changes or system parameter changes. In this case, standard modification is needed for the above algorithm.

### B. Modeling results showing the nonlinear feature

The modeling results are shown in Figures 2–5. The original data are normalized between zero and one as shown in Figure 2, the estimated parameters are given in Figure 3, the modeling error is displayed in Figure 4, and the corresponding PDF of the modeling error is illustrated in Figure 5. It can be seen that the system is clearly not linear, as the modeling error PDF is not Gaussian.

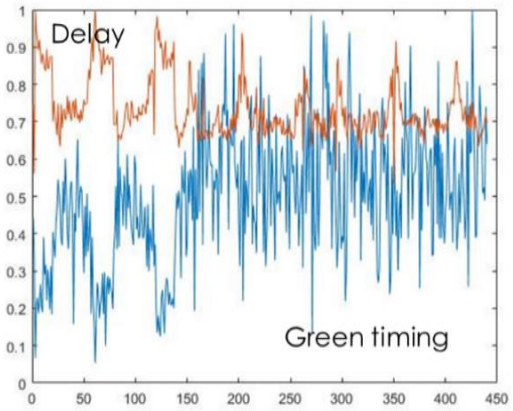



Figure 2. Original data—normalized to [0, 1].

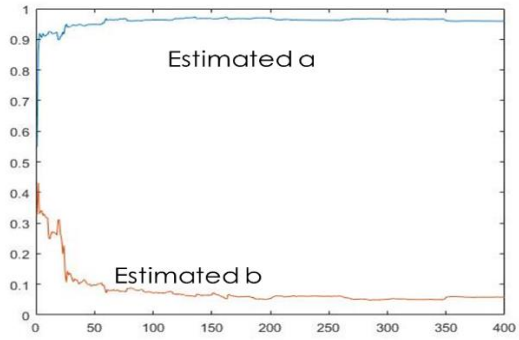



Figure 3. Estimated a and b.

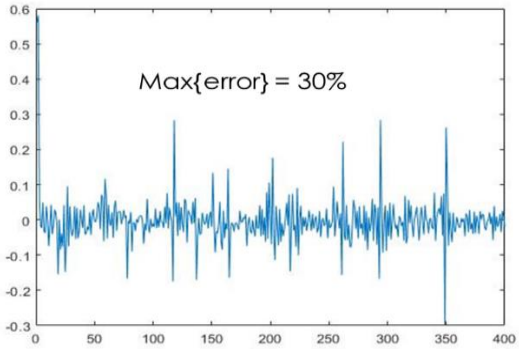



Figure 4. Modeling error—RLS residual signal

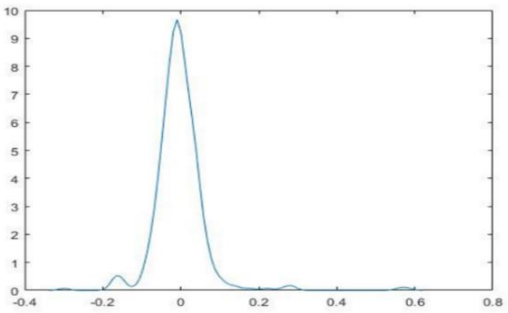


Figure 5. Modeling error PDF.

In addition, figures 6 and 7 show the validation results.

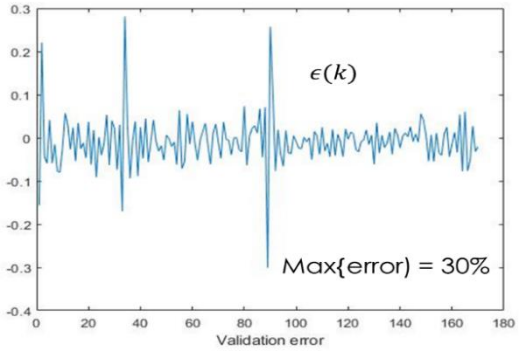



Figure 6. Validation error showing a 30% error for the normalized data.

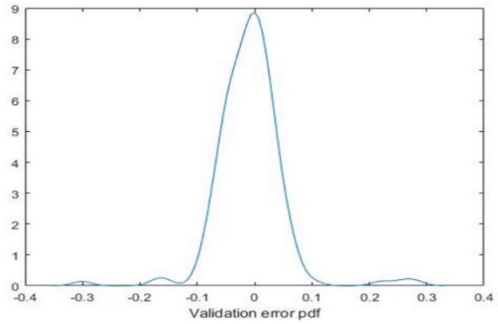



Figure 7. PDF of the validation error showing a non-Gaussian shape.

## V. HYBRID NEURAL NETWORK

As the system is nonlinear and non-Gaussian, HNN modeling is described in this section. In this context, a dynamic model was considered that reflected the relationship between the input and the output. Moreover, to improve the model, traffic volume was also considered as an extra input. Thus, the system had two input vectors (i.e., signal time plan and traffic volume) and one output vector, traffic delays.

The system model was therefore assumed as follows:

$$y(k+1) = Ay(k) + Bu(k) + f(y(k), u(k-1), v(k)) \quad (5)$$

where $y(k)$ and $u(k)$ denote average delay per vehicle and green time for multiple intersections at time index $k$. $f(\dots)$ is an unknown nonlinear vector function to be learnt and $\omega(k)$ is noise. $\{A, B\}\}$ are the weight matrices to be identified simultaneously with the estimate for the unknown nonlinear dynamics.

Let NN be used to approximate $f\big(y(k), u(k-1), v(k)\big)$ by $\hat{f}(y(k), u(k-1), v(k), \pi)$, where $v(k)$ denotes traffic volume; $\pi$ groups all NN weights and biases. Then the training of the NN as well as the two matrices was to obtain accurate and reliable models for the traffic flow system. In this case, we considered seven intersections of an arterial all together, as indicated in the red box in Figure 1.

The objective of training was to minimize the following performance function:

$$\text{Min } J = \frac{1}{2}\big(\hat{y}(k+1) - y(k+1)\big)^2 \quad (6)$$

which is basically a minimum variance error criteria, where it has been defined that

$$\hat{y}(k+1) = Ay(k) + Bu(k) + \hat{f}(y(k), u(k-1), v(k), \pi) \quad (7)$$

and $\{A, B, \pi\}$ are parameters to be trained. In the above equation, vectors $\hat{y}(k)$ and $\hat{f}(\dots)$ are the estimates of $y(k)$

and $\hat{f}(\ldots)$, respectively using the real-time data from Econolite Systems.

### A. Gradient rule for training

Using gradient optimization, the following recursive estimation and training algorithm can be readily obtained to read

$$\hat{A}(k+1) = \hat{A}(k) - \lambda_1 \frac{\partial J}{\partial A}\Big|_{\left(\hat{A}(k),\hat{B}(k),\hat{\pi}(k)\right)} \quad (8)$$

$$\hat{B}(k+1) = \hat{B}(k) - \lambda_2 \frac{\partial J}{\partial B}\Big|_{\left(\hat{A}(k),\hat{B}(k),\hat{\pi}(k)\right)} \quad (9)$$

$$\hat{\pi}(k+1) = \hat{\pi}(k) - \lambda_3 \frac{\partial J}{\partial \pi}\Big|_{\left(\hat{A}(k),\hat{B}(k),\hat{\pi}(k)\right)} \quad (10)$$

where $\lambda_1$, $\lambda_2$ and $\lambda_3$ are pre-specified positive learning rates which are typically selected to be less than 1.0, and the gradients are calculated from

$$\frac{\partial J}{\partial A}\Big|_{\left(\hat{A}(k),\hat{B}(k),\hat{\pi}(k)\right)} =$$

$$\left(\hat{y}(k+1) - y(k+1)\right)\frac{\partial \hat{y}}{\partial A}\Big|_{\left(\hat{A}(k),\hat{B}(k),\hat{\pi}(k)\right)} = \left(\hat{y}(k+1) - y(k+1)\right) y(k) \quad (11)$$

$$\frac{\partial J}{\partial B}\Big|_{\left(\hat{A}(k),\hat{B}(k),\hat{\pi}(k)\right)} =$$

$$\left(\hat{y}(k+1) - y(k+1)\right)\frac{\partial \hat{y}}{\partial B}\Big|_{\left(\hat{A}(k),\hat{B}(k),\hat{\pi}(k)\right)} = \left(\hat{y}(k+1) - y(k+1)\right) u(k) \quad (12)$$

$$\frac{\partial J}{\partial \pi}\Big|_{\left(\hat{A}(k),\hat{B}(k),\hat{\pi}(k)\right)} = \left(\hat{y}(k+1) - y(k+1)\right)\frac{\partial \hat{f}}{\partial \pi}\Big|_{\left(\hat{A}(k),\hat{B}(k),\hat{\pi}(k)\right)} \quad (13)$$

where $y(k+1)$ is the measured real-time data from the Econolite systems.

The selection of the learning rates are also critical here in order to ensure a good balance between the responsiveness of the learning and its stability in providing convergent neural network training. Using the second-order derivative analysis such as Jaccobian Matrices measure one can obtain the ranges for these learning rates.

The training algorithm described in (8) – (13) provides a set of simultaneous estimates for both linear parameters and neural network weights. Also, as the control input $u(k)$ to be designed is linearly involved in the model, the controller design using AI-techniques can be easily implemented as a direct inverse calculation so long as the matrix *B* is of a full column rank. This approach effectively facilitates the real-time implementation for the whole system.

### B. Data and their processing

To model the system in (5), relevant data from the seven intersections were collected along the arterial as shown in Figure 1. In this context, the details of the data collected are as summarized in the Table II.

TABLE II. DATA COLLECTION FOR HNN MODELING

| | |
|---|---|
| Study area | Intersection 1-7 |
| Date collected | March 3–5, 8–12, 15–19, 22–26, 29–31, April 1–2 (23 weekdays) in 2021 |
| Time duration | 4 pm – 7 pm |
| Signal timing | All phases of major and minor streets |
| Traffic volume | All movements |
| Traffic delay | All movements |
| Sampling index | Every five signal cycles (each cycle ≈180 s) |

### C. Modeling results

Before the HNN model was trained, the raw data needed to be preprocessed to remove or reduce volatility, as shown in Figure 9. For traffic signal and traffic volume data, normalization was conducted to scale data between zero and one. For traffic delay data, after normalization, simple exponential smoothing was applied to further filter the data to remove noise, as shown in (14), where $l(k)$ is the filtered delay, $y(k)$ is normalized delay, and $\alpha$ is the smoothing factor between zero and one. As alpha decreases, the observation of delay at *k* has less impact on the output $l(k)$, indicating that the randomness of the delay measurements is reduced. After training of the HNN model, inverse normalization and inverse smoothing were applied to generate actual model output. This process is shown in Figure 8.

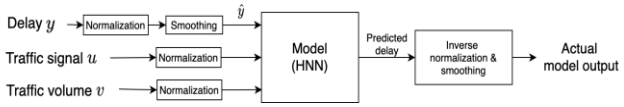



Figure 8. Data preprocessing.

$$l(k) = \alpha y(k) + (1-\alpha) l(k-1) \quad (14)$$

The HNN model was trained by 78% of the total data points and was tested with data from March 22–26 (22% of total data). Figure 9 illustrates the HNN model structure applied in this study.

The modeling results were evaluated by mean absolute percentage error (MAPE), rooted mean square error (RMSE), and mean absolute error (MAE) as in (15)–(17), where $y_n(k)$ is the true delay at time *k* of phase *n* and $\hat{y}_n(k)$ is the predicted delay at time *k* of phase *n*.

$$MAPE = \frac{1}{NK}\sum_{k=1}^{K}\sum_{n=1}^{N}\left|\frac{y_n(k) - \hat{y}_n(k)}{y_n(k)}\right| \quad (15)$$

$$RMSE = \frac{1}{NK}\sum_{k=1}^{K}\sum_{n=1}^{N}\sqrt{(y_n(k) - \hat{y}_n(k))^2} \quad (16)$$

$$MAE = \frac{1}{NK}\sum_{k=1}^{K}\sum_{n=1N}|y_n(k) - \hat{y}_n(k)| \quad (17)$$

Table III and Table IV show the prediction results for all phases of all seven intersections, the phases of main streets and side streets, and the phase of each intersection. Note that delay prediction at main streets is more accurate than at side

streets. The reason is that traffic volumes at side streets are much lower and more stochastic compared with main streets.

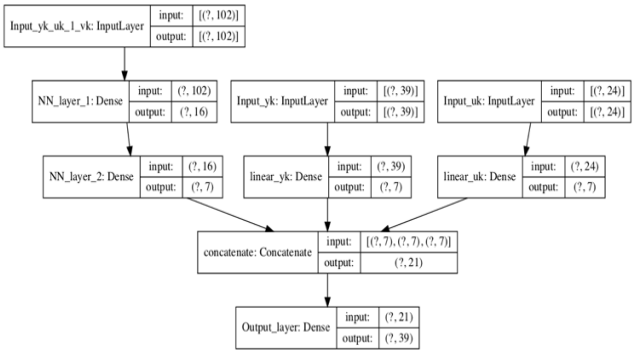



Figure 9. HNN model structure.

TABLE III. TRAINING AND TESTING RESULTS

| | Training (all) | Testing (all) | Testing (main streets) | Testing (side streets) |
|---|---|---|---|---|
| MAPE | 6.3% | 6.5% | 5.6% | 6.9% |
| RMSE | 9.6 s | 10.1 s | 4.1 s | 12.3 s |
| MAE | 6.7 s | 6.9 s | 3.0 s | 9.2 s |

TABLE IV. TESTING RESULTS AT EACH INTERSECTION

| **Intersection** | **1** | **2** | **3** | **4** |
|---|---|---|---|---|
| MAPE | 4.0% | 5.0% | 5.7% | 7.7% |
| RMSE | 3.7 s | 5.7 s | 10.7 s | 11.0 s |
| MAE | 2.2 s | 4.3 s | 6.6 s | 8.7 s |

| **Intersection** | **5** | **6** | **7** |
|---|---|---|---|
| MAPE | 7.7% | 6.7% | 6.1% |
| RMSE | 12.6 s | 8.8 s | 10.3 s |
| MAE | 9.1 s | 6.2 s | 7.6 s |

Figure 10 and Figure 11 show the distribution and PDF of training errors. Training errors are roughly symmetrically distributed along the horizontal axis.


Figure 10. Training error.

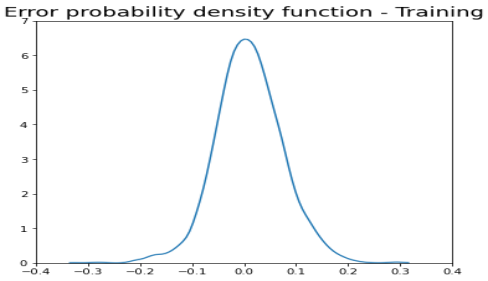



Figure 11. Training error PDF.

Figure 12 and Figure 13 show distribution and PDF of testing errors.


Figure 12. Testing error.

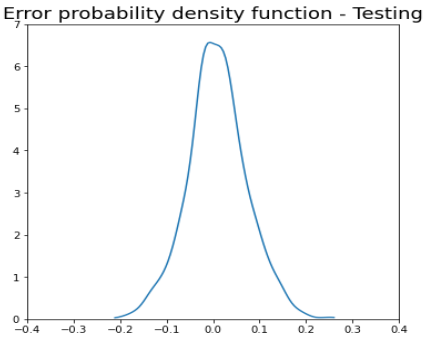
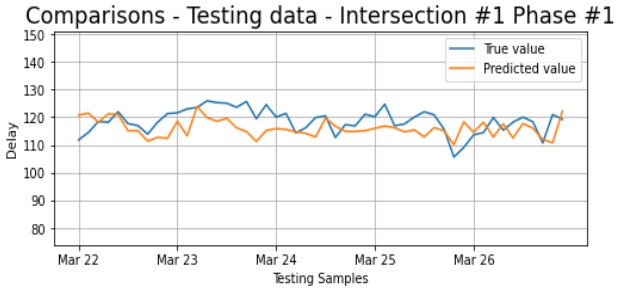
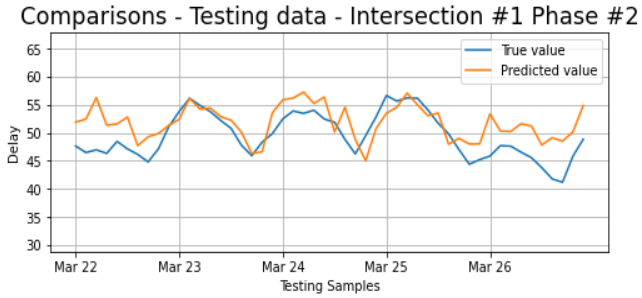



Figure 13. Training error PDF.

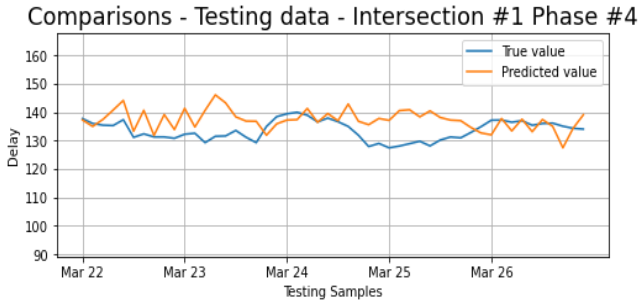
Figure 14 shows comparisons of predicted delay from the HNN model and the true delay of each phase at intersection 1. There are four phases at intersection 1. Figure 14 (a-d) shows the delay comparisons of each phase, respectively.



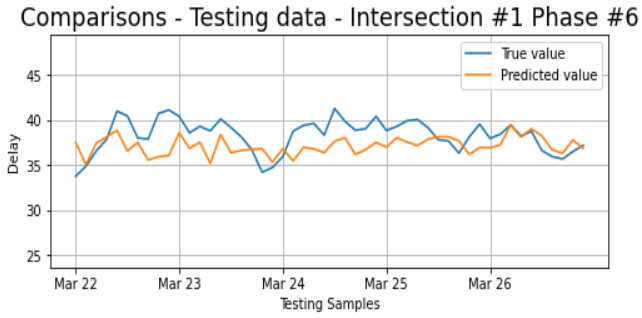

(a) Phase 1: Westbound left turning movement




(b) Phase 2: Eastbound through movement




(c) Phase 4: Northbound through + left turning movements




(d) Phase 6: Westbound through movement
Figure 14. Delay Comparisons at Intersection 1.

## VI. CONCLUSIONS

This keynote paper starts with a survey on the current neural network and Artificial Intelligent based modeling and control for signalized intersections. This is then followed by a study which developed a MIMO HNN model for multiple intersections along a corridor. The model can capture both the linear and nonlinear stochastic natures of multiple traffic features, i.e., traffic signal timings, traffic flows, and travel delays. The proposed model was validated by real-world data extracted from an Econolite system. The MAPE for delay prediction was 6.5% and the MAE was 6.9 s for all movements. The experimental results also suggested that the delay prediction for major streets was more accurate than that for minor streets.

This study demonstrated a first step for the implementation of AI-based transportation system modeling and control. For future work, we will continue to collect data from more intersections and further refine the HNN model. When the model is ready, we will develop an AI-based optimal traffic control system based on the model to minimize entire system costs, including travel delay and energy consumption.

Once a reliable system model is obtained, AI-based control design is required to establish a real-time closed-loop feedback control system that uses the traffic flow state as feedback [40][41]. This approach controls the signal timing intelligently at intersections so that the resulting traffic flow can be made smoother with minimized energy consumption. This control method requires controller design using AI techniques. Because of the random nature of traffic flow systems, stochastic optimal control in a multi-objective Bayesian framework will be investigated in the future.

## ACKNOWLEDGMENT

The work performed is funded by the Vehicle Technologies Office of the US Department of Energy under the AI for Mobility program.

This manuscript has been authored by UT-Battelle, LLC, under contract DE-AC05-00OR22725 with the US Department of Energy (DOE). The US government retains and the publisher, by accepting the article for publication, acknowledges that the US government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this manuscript, or allow others to do so, for US government purposes. DOE will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan (http://energy.gov/downloads/doe-public-access-plan).

## REFERENCES

[1] A. Sumalee and H. Ho, “Smarter and more connected: future intelligent transportation system,” IATSS Res.42, 2018.

[2] D. Srinivasan, M. C. Choy, and R. L. Cheu, “Neural networks for real-time traffic signal control,” IEEE Trans. on Intelligent Transportation Systems. vol. 7, no. 3, pp. 261 – 272, 2006.

[3] G. N. Cadet, “Traffic signal control — a neural network approach,” PhD Thesis - Florida International Univ., 1996.

[4] D. Srinivasan, M. C. Choy, and R. L. Cheu, Neural networks for real-time traffic signal control, IEEE Trans. on Intelligent Transportation Systems, vol. 7, no. 3, pp. 261 – 272, 2006.

[5] K. A. Yau, J. Qadir, H. L. Khoo, M. H. Ling, and P. Komisarczuk, “A survey on reinforcement learning models and algorithms for traffic signal control,” ACM Comput. Surv. Vol. 50, 3, Article 34, 2017.

[6] R. Ke, W. Li, Z. Cui, and Y. Wang, “Two-stream multi-channel convolutional neural network for multi-lane traffic speed prediction considering traffic volume impact,” Transportation Research Record, vol. 2674, no. 4, pp. 459-470, 2020.

[7] H. Yuan and G. Li, “A survey of traffic prediction: from spatio-temporal data to intelligent transportation,” Data Science and Engineering, vol. 6, no. 1, pp. 63-85, 2021.

[8] B. P. Williams, Durvasula, and D. Brown, “Urban freeway traffic flow prediction: application of seasonal autoregressive integrated moving average and exponential smoothing models,” Transp. Res. Rec. vol. 1644, no. 1, pp. 132–141, 1998.

[9] L. Zhang, Q. Liu, W. Yan, N. Wei, and D. Dong, “An improved k-nearest neighbor model for short-term traffic flow

 ISBN: 978-1-61208-879-2 

prediction," Procedia-Social and Behavioral Sciences, vol. *96*, pp. 653-662, 2013.

[10] W. Li, J. Wang, R. Fan, Y. Zhang, Q. Guo, C. Siddique, and X. Ban, "Short-term traffic state prediction from latent structures: accuracy vs. efficiency," Transportation Research Part C: Emerging Technologies, vol. 111, pp. 72-90, 2020.

[11] J. Guo, W. Huang, and B. M. Williams, "Adaptive Kalman filter approach for stochastic short-term traffic flow rate prediction and uncertainty quantification," Transp. Res. Part C: Emerging Technol. vol. 43, pp. 50–64, 2014.

[12] P. V. V. Theja and L. Vanajakshi, "Short term prediction of traffic parameters using support vector machines technique," In 2010 3rd International Conference on Emerging Trends in Engineering and Technology, pp. 70-75, 2010.

[13] L. Vanajakshi and L. R. Rilett, "A comparison of the performance of artificial neural networks and support vector machines for the prediction of traffic speed," In IEEE Intelligent Vehicles Symposium, vol. 2, pp. 194-199, 2004. IEEE.

[14] Z. Cui, R. Ke, Z, Pu, and Y. Wang, "Stacked bidirectional and unidirectional LSTM recurrent neural network for forecasting network-wide traffic state with missing values," Transportation Research Part C: Emerging Technologies, vol. 118, 2020.

[15] Z. Cui, K. Henrickson, R. Ke, and Y. Wang, "Traffic graph convolutional recurrent neural network: a deep learning framework for network-scale traffic learning and forecasting," IEEE Transactions on Intelligent Transportation Systems, vol. 21, no. 11, pp. 4883-4894, 2019.

[16] P. B. Hunt, D. I. Robertson, R. D. Bretherton, and M. C. Royle, "The scoot on-line traffic signal optimisation technique," Traffic Engineering & Control, vol. 23, no. 4, 1982.

[17] W. R. McShane and R. P Roess. Traffic engineering. Prentice Hall, 1990.

[18] S. B. Cools, C. Gershenson, and B. D'Hooghe, "Self-organizing traffic lights: a realistic simulation," In Advances in applied self-organizing systems, pp. 45–55. Springer, 2013.

[19] P. Varaiya, "The max-pressure controller for arbitrary networks of signalized intersections," In Advances in Dynamic Network Modeling in Complex Transportation Systems, pp. 27–66. Springer New York, 2013.

[20] P. R. Lowrie. "SCATS, Sydney cooordinated adaptive traffic system: a traffic responsive method of controlling urban traffic," In Sales information brochure. Roads & Traffic Authority Sydney, Australia, 1990.

[21] W. Wei and Y. Zhang, "FL-FN based traffic signal control," in Proc. 2002 IEEE Int. Conf. Fuzzy Syst., May 2002, vol. 1, no. 12–17, pp. 296–300.

[22] E. Bingham, "Reinforcement learning in neural fuzzy traffic signal control," Euro. J. Operation Res., vol. 131, no. 2, pp. 232–241, 2001.

[23] D. Srinivasan, M. C. Choy, and R. L. Cheu, "Neural networks for real-time traffic signal control," IEEE Transactions on intelligent transportation systems, vol. 7, no. 3, 261-272, 2006.

[24] M. C. Choy, D. Srinivasan, and R. L. Cheu, "Neural networks for continuous online learning and control," IEEE Transactions on Neural Networks, vol. 17, no. 6, 1511-1531, 2006

[25] L. Li, L. Yisheng, and F. Wang, "Traffic signal timing via deep reinforcement learning," IEEE/CAA Journal of Automatica Sinica. vol. 3, no. 3, pp. 247254. 2016.

[26] H. Wei, G. Zheng, H. Yao, and Z. Li, "Intellilight: a reinforcement learning approach for intelligent traffic light control," In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. London, UK. pp. 2496-2505. 2018.

[27] H. Wei, et al., "Presslight: lLearning max pressure control to coordinate traffic signals in arterial network," In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 1290-1298, 2019.

[28] C. Chen, et al., "Toward A thousand lights: Decentralized deep reinforcement learning for large-scale traffic signal control," In Proceedings of the AAAI Conference on Artificial Intelligence. vol. 34, no. 04, pp. 3414-3421, 2020.

[29] H. Wei, et al., "Colight: Learning network-level cooperation for traffic signal control," In Proceedings of the 28th ACM International Conference on Information and Knowledge Management, pp. 1913-1922, 2019.

[30] T. Zhong, Z. Xu, and F. Zhou, "Probabilistic graph neural networks for traffic signal control," In ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 4085-4089, 2021.

[31] M. Abdoos, N. Mozayani, and A. L. Bazzan, "Hierarchical control of traffic signals using Q-learning with tile coding," Applied intelligence, vol. 40, no. 2, 201-213, 2014.

[32] P. Mannion, J. Duggan, and E. Howley, "An experimental review of reinforcement learning algorithms for adaptive traffic signal control," Autonomic road transport support systems, 47-66, 2016.

[33] O. Nachum, M. Norouzi, K. Xu, and D. Schuurmans, "Bridging the gap between value and policy based reinforcement learning," In NeurIPS, 2017.

[34] H. Wei, G. Zheng, G. Gayah, and Z. Li, "Recent advances in reinforcement learning for traffic signal control: a survey of models and evaluation," ACM SIGKDD Explorations Newsletter, vol. 22, no. 2, 12-18 2021.

[35] L. A. Prashanth and S. Bhatnagar, "Reinforcement learning with average cost for adaptive control of traffic lights at intersections," In 2011 14th International IEEE Conference on Intelligent Transportation Systems, pp. 1640-1645, 2011.

[36] G. H. Zhang, Y. H. Wang, and H. Wei, "Artificial neural network method for length-based vehicle classification using single-loop outputs," Traffic Urban Data, Transp. Res. Rec. vol. 1945, pp. 100–108, 2006.

[37] H. Yu, X. F. Chen, Z. N. Li, G. H. Zhang, P. Liu, J. F. Yang, and Y. Yang, "Taxi-based mobility demand formulation and prediction using conditional generative adversarial network-driven learning approaches," IEEE Transactions on Intelligent Transportation Systems, vol. 20, no. 10, pp. 3888-3899, . 2019.

[38] H. Yu, Z. N. Li, G. H. Zhang, P. Liu, and J. Wang, "Extracting and predicting taxi hotspots in spatiotemporal dimensions using conditional generative adversarial neural networks," IEEE Transactions on Vehicular Technology, vol. 69, no. 4, pp. 3680-3692, April 2020.

[39] Z. M. Li, H. Yu, G. H. Zhang, S. J. Dong, C. Z. Xu, "Network-wide traffic signal control optimization using a multi-agent deep reinforcement learning," Transportation Research Part C: Emerging Technologies, vol. 125, 2021.

[40] H. Wang, M. Zhu, W. Hong, C. Wang, G. Tao and Y. Wang, "Optimizing signal timing control for large urban traffic networks using an adaptive linear quadratic regulator control strategy," IEEE Transactions on Intelligent Transportation Systems, doi: 10.1109/TITS.2020.3010725, 2020.

[41] H. Wang, S. Patil, H. Aziz, and S. Young, "Modeling and control using stochastic distribution control theory for intersection traffic flow," IEEE Transactions on Intelligent Transportation Systems, doi: 10.1109/TITS.2020.3028994, 2020.

 ISBN: 978-1-61208-879-2 