Flex-Corridor: Re-configurable Traffic Corridors for Connected and Automated Vehicle Testing

Maddie Juarez[1] and Priyash Misra[2]

[1] *DOE SULI Intern, Loyola University Chicago, Chicago, IL 60660, United States*

[2] *Principal Research Engineer, Sustainable Transportation Education and Partnerships Department, Transportation and Power Systems Division, Argonne National Laboratory, Lemont, Illinois, 60439, United States*

Vehicle-to-Everything (V2X) communication enables the exchange of information between vehicles, traffic infrastructure, pedestrians, and backend networks. Cellular V2X (C-V2X) can be used to develop safety and efficiency features including providing speed advisories, efficient intersection navigation, blind spot alerts, and intersection collision warning. V2X deployment has transitioned from DSRC based radios to C-V2X following directions from the Federal Communication Commission (FCC) in 2020. A common application for V2X is to use traffic signal data on the vehicle to efficiently navigate intersections.To validate such control algorithms, a highly controllable and repeatable track based system is needed. FlexCorridor is such a system that uses C-V2X technology with the intention to simulate controlled and re-configurable environments for testing on-road. Through this project, we hope to discuss methods to develop and deploy custom software and hardware for transmission and reception of V2X information for on-vehicle and roadside unit operation. We also propose methods for simulating upcoming driving scenarios and enabling radios to re-create drive scenarios on the road to enable Hardware-in-the-Loop (HIL) bench testing for validation prior to on-road testing in addition to mechanisms of real-time, on road data collection methods.

I. INTRODUCTION

Cellular Vehicle-to-Everything (C-V2X) is a communication technology that enables the exchange of information between vehicles, traffic infrastructure, pedestrians, and backend networks. C-V2X uses cellular networks, such as 4G LTE and 5G, to enable real-time information exchange that can be used to develop safety and efficiency features including providing speed advisories, efficient intersection navigation, blind spot alerts, and intersection collision warnings. Vehicle-to-Infrastructure (V2I) is a sub-component of the V2X functionality and can be used for communication between vehicles and traffic infrastructure (e.g., traffic signal controllers) as a vehicle approaches a connected intersection.

Previous connected and automated vehicle (CAV) research has been conducted using Dedicated Short-Range Communications (DSRC), which includes vehicle-to-vehicle communication infrastructure. However, in 2020, the Federal Communications Commission mandared a transition from DRSC technology to C-V2X based communication[1]. While DRSC communication is powerful, C-V2X leverages its capabilities by incorporating cellular networks in addition to the 5.9 GHz frequency band used by DSrC to enable network based communicaiton along with direct communicaiton.

C-V2X networks utilize On-Board Units (OBUs) and Roadside Units (RSUs) to transmit and receive information between vehicles and connected intersections. OBUs are radios that are installed inside a vehicle and are part of a CAV network. It can communicate with other vehicles, road infrastructure, and relies on direct C-V2X. RSUs are radios that are installed alongside road infrastructure and communicate wirelessly with vehicles, pedestrians, or other RSUs. It transmits SPaT and MAP messages. Standardized by the Society for Automotive Engineers (SAE) in document J2735[2], Signal Phase and Timing (SPaT) messages contain information about the current traffic signaling phase and timing information at a connected intersection such as the current signal phase, maximum end time, minimum end time, and confidence. MAP messages contain information about the current intersection, and signal group identifiers that connect ingress and egress lanes in a path of travel. OBUs can also transmit Basic Safety Messages (BSMs), which are exchanged between connected vehicles and contain information about the vehicle's current latitude and longitude, speed, acceleration, and other relevant data.

II. FLEXCORRIDOR

A. Motivations

The FlexCorridor project at Argonne National Laboratory's (ANL) Center for Transportation Research under the Sustainable Transportation and Partnerships (STEP) department, focuses on developing and integrating advanced V2X connectivity for CAVs. The project aims to develop standardized on-road test procedures for testing CAV controls for intersection navigation by deploying a set of highly customizeable and controllable traffic intersections. This involves implementation of V2X radios for OBU and RSU operation, transmission and reception of BSM, SPaT, MAP, and visualization of V2X information.

As research makes progress on V2X based vehicle controls development and deployment there is a need to define test methodologies for efficient testing of such vehicle features in a track based environment. Repeatable and reproducible On-road testing involving SPaT data can be difficult owing to the fact the SPaT timings are defined as future times in UTC. Additionally, SPaT timings at real intersections are not constant and can change based on traffic flow and time of the day. FlexCorridor addresses the above concerns by enabling a strictly controlled on-road track based test setup to test I2V based CAV controls.

FlexCorridor, seen in Figure 1, has intersections that will consist of a RSU and a physical traffic light. These intersections will have the following capabilities:

- Track an approaching Vehicle Under Test (VUT) and trigger pre defined scenarios such as SpAT and MAP

- Localize VUT to an approach lane and keep track of its distance from the intersection

- Generate custom SPaT and MAP as defined the test scenario

- Keep track of how many times the vehicle crosses the intersection and generate new scenarios every time

- Log test data in a reliable and reproducible manner

- Ensure flexibility and ease of test setup

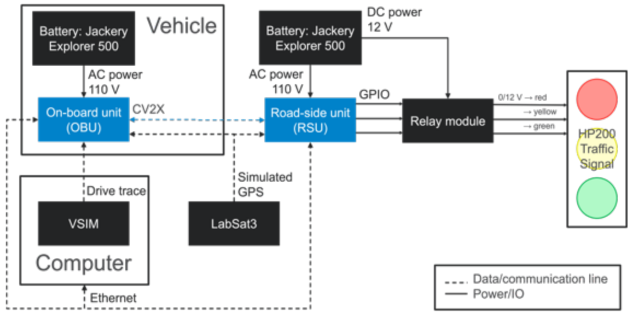

FIG. 1. FlexCorridor Overview

B. Equipment and Methodologies

MK6 OBUs and RSUs manufactured by Cohda Wireless were used for all developmental and experimental test cases. Cohda Wireless' MK6 C-V2X radios were used with a GNSS simulator, and an Ethernet switch as seen in Figure 2. An RSU and two OBUs were used for testing purposes. A shell terminal was used to ping the radios, confirming that they were operational and connected to the network. Using SSH (Secure Shell) to securely access and manage devices over a network, we established a remote connection to each radio. This allowed us to configure and monitor the radio's setting and status from the terminal.

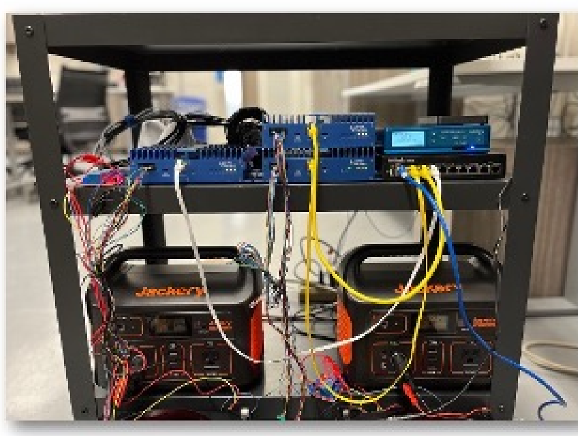

FIG. 2. V2X Radio Setup

Cohda Wireless provided Software Development Kit (SDK) was also included along with a Linux-based Virtual Machine (VM). Wireshark, a network protocol analyzer for network activity, was used to visualize and analyze data logs and User Datagram Protocol (UDP) traffic. Furthermore, we used a virtual simulation tool (VSIM), provided by the SDK, to design, transmit, and visualize simulated drives with playback features. The use of a Global Navigation Satellite System (GNSS) simulator to mimic satellite signals for GPS information was also used in this study to establish a controlled environment for testing. By using the GNSS simulator with a saved GPS log, we can establish a time alignment for the radios to sync to without needing live GPS—a common limitation in a lab environment with limited satellite signal.

C. Flex-Corridor Accomplishments

1. *Successful Implementation of V2X Radio Communication*

We successfully implemented V2X radios on OBUs and RSUs, establishing communication channels between both and ensuring the health of both radios, enabling them to transmit and receive messages. To assess the functionality and health of the radios, we execute the example1609 program, a customizable application source code from Cohda Wireless, and monitor the size of the received data logs. As messages are transmitted and received, the received and transmitted data logs grow in size, indicating a successful connection.

To simulate a vehicle drive trace and transmit it from an OBU to another radio, the OBU's configuration settings were modified to use VSIM information as a data source. For the receiving radio, we disabled the VSIM feature as a data source, configuring it to listen for the transmission from the OBU. This setup allowed us to effectively test the communication and data transfer between the radios in a controlled environment. To confirm the transmission and reception of the drive trace data, we analyzed the BSMs of the Wireshark log from the receiving radio.

2. *Designed Simulated Drive Trace and SPaT/MAP*

We used the U.S. Department of Transportation's ISD Message Creator for Connected Vehicles to create and define information about an intersection and design a SPaT sequence[3]. This tool allows us to define an intersection at a specific latitude and longitude, with a unique intersection identifier. For experimentation purposes, we created our mock intersection at ANL's West side of campus, along Kearney Rd between Watertower Rd and Rock Rd, roughly a one mile stretch of straight road. After initializing these elements, we are able to use the tool to generate the MAP UPER hex encoding, which represents the relevant information about the intersection in hex.

In order to design a SPaT message, we must use the MAP information as a guide to link the relevant traffic signal sequences to the respective unique intersection identifier. From there, we can continue to use the USDOT's Connected Vehicle ISD Creator to define the number and path of lanes at the intersection, the approach box, lane type, and the SPaT sequences. The path of the lanes can be drawn in Figure 3 in the desired travel direction for drivers and can be adjusted based on various factors such as the width of the road, the number of lanes, and the directionality of each lane (e.g., one-way, two-way, right-turn only, etc.).

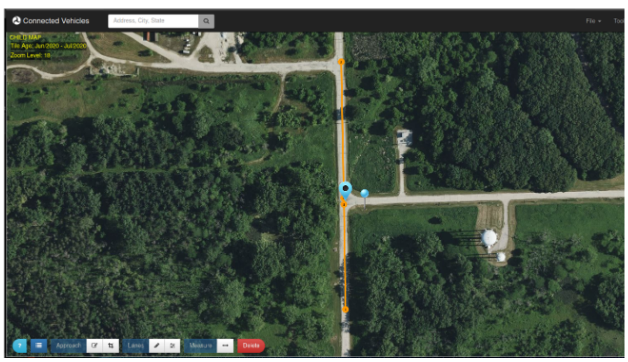

FIG. 3. Connected Vehicles ISD Tool

A user can start designing a SPaT by assigning a lane number and establishing a signal group ID, which is an identifier that links specific signals to corresponding lanes. Using these parameters, the user can then define the traffic signal phases. For example, a phase value of 3 might represent a "Stop-and-Remain" or red light signal. Additionally, the user can specify the start time (in seconds), end time, and the confidence level of the signal's probability. For instance, a confidence value of 15 could indicate a 100% probability that the signal will behave as defined. These configurations allow for precise control over traffic signals to be transmitted by an RSU at a specified intersection at a rate of 1/10 second. For every second of the duration of a traffic signal in real time, such as a one-second yellow light, there will be 10 SPaT messages transmitted for that sequence. Each individual SPaT Sequence will generate a UPER hex encoding.

3. Use of RSU and OBU

After successfully designing the MAP and SPaT sequences and obtaining the unique UPER hex encoding values, we are able to use this data to simulate a connected intersection. We modified the RSU's configuration to include the unique intersection identifier specified in our MAP, in order to configure the connected intersection and send out the SPaT messages from the RSU. Furthermore, we designed a drive trace using VSIM along a one mile stretch of road on ANL's west side to include the intersection included in the MAP and SPaT. We began transmitting BSMs from the OBU using the drive trace data, and visualized the traffic signal sequences in real time as the vehicle approached the designated intersection.

4. Validation of Flex-Corridor Implementation

Building upon previous work on Flex-Corridor, we validated the transmission of BSMs from an OBU to an RSU containing the FlexCorridor start up scripts. These scripts included various python packages included with custom Python code equipped to handle ASN.1 Automotive Messages that are received and forwarded to an RSU. Furthermore, we validated the results of FlexCorridor with a test drive along Kearney Rd, seen in Figure 4, ensured the visualization of custom traffic sequences, and monitored repeated detections along a single connected intersection in a looped test. This validation process ensures the repeatability of the test in a controlled, simulated environment.

5. Python Hex Message Decoder

BSMs, SPaTs, and MAPs are transmitted from an OBU or RSU through UDP in packaged hex streams over a Controller Area Network (CAN) network. Wireshark and other applications use custom imported packages to decode hex streams, convert them to binary, and extract the decimal representations of the data based on positional encodings. However, to reduce reliance on external packages and comply with updated versions, we have opted to develop a custom decoder using standard Python libraries.

The custom decoder takes an input of a hex stream dump sourced from UDP, and first obtains the message type (BSM, SPaT, or MAP) from the pattern first few characters: 0012 for MAP, 0013 for SPaT, and 0014 for BSM. After obtaining the type of message, the

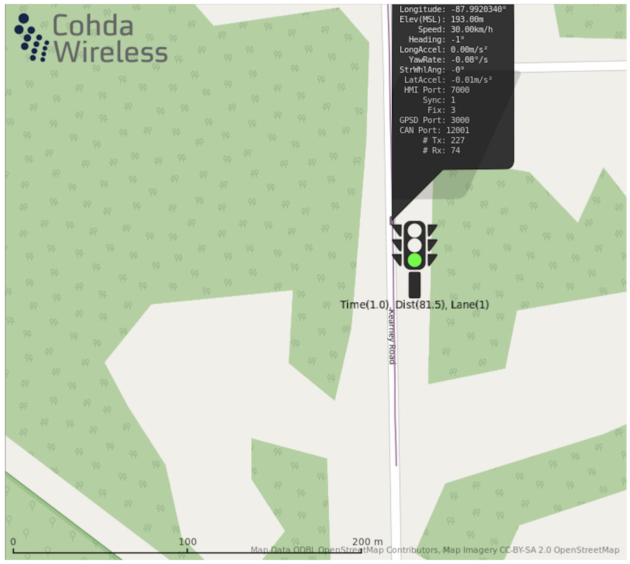

FIG. 4. Kearney Rd Test

hex stream will be decoded by the relative processing function. Each processing function converts the hex stream into binary, and extracts core data using the bit positions of the values according to their data type lengths per SAE J2735. This decoder uses an Object-oriented Programming approach to create custom message objects that returned the decoded data upon method call, allowing for the instantiation of multiple BSM objects, for example, as they are transmitted by an OBU and received by an RSU.

6. Custom Data Logging Mechanism

In drive simulations, various components such as the radio, GNSS simulator, and SDK (Software Development Kit) may operate on different time sources, leading to an inconsistent measure of time. Additionally, relevant data such as a vehicle's BSMs may be captured in a Wireshark log file whereas the number of vehicle detections at an intersection may be captured locally inside a Python script. To ensure accurate validation and data collection, it was essential to establish a standardized time source and localize the collection of data. This was achieved by implementing a standard CSV logging format that includes precise timestamps.

Within the FlexCorridor script, a custom Python function was created to write data to a CSV file. This function utilized the date and time packages from the Python standard library to log timestamps in Coordinated Universal Time (UTC) for the duration of the test. By doing so, we were able to monitor and collect data that is synchronized to a single time source, rounded to the nearest millisecond. Furthermore, this function also captured the Vehicle ID, number of detections through the intersection, the current phase signal, in addition to other BSM information into Python dictionary and appended to a single CSV file for the duration of the test.

D. Other Contributions

1. Live-Testing at Chicago Corridor

We successfully tested Argonne National Laboratory's Hyundai IONIQ 5 in Chicago's test corridor along 7 intersections, and successfully received SPAT and MAP messages in real-time. Figure 5 shows a drive trace that was collected from the radio's logs after our test drive in the Chicago corridor, allowing us to track, log, and monitor the vehicle's state during the test drive. In the test drive, we successfully monitored the transmission of BSM messages to street RSU to obtain advised speed for the moving vehicle based on relative position and signal phase changes using the GLOSA algorithm for advised speed. Green Light Optimized Speed Advisory (GLOSA) is an algorithm that uses the drivers current speed and distance relative to an approaching intersection to use as factors in recommending a speed advisory range for drivers. The GLOSA algorithm used in this test was provided by Cohda Wireless and was required for validation of OBU functionality. Furthermore, the test proved interoperability of V2X devices as the OBU and RSU were from different vendors.

2. Host Vehicle to Remote Vehicle Validation Tests

Tests were also performed to validate BSM transmission and reception with a two vehicle test. The Host Vehicle State data was extracted on internal buffers on the OBU where as the remote vehicle data populated by the Receiving Vehicle from the received BSM. It is important for experimental purposes to validate that two vehicles are transmitting the correct data to each other, and that data can be logged during a test, when vehicles are on-road. Therefore, we added a custom function inside the executable program example1609 provided by Cohda Wireless to create a vehicle state data type that is populated by the

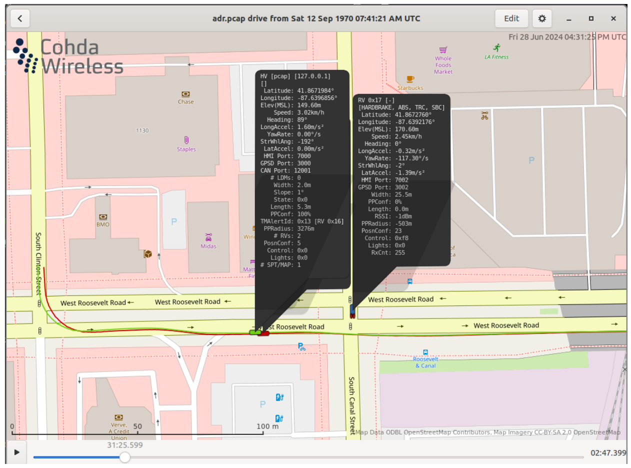

FIG. 5. Chicago Corridor Test

vehicles speed, latitude, longitude, and longitudinal acceleration to be sent over UDP. These values were validated using two OBUs and a VSIM drive trace seen in Figure 6. We analyzed the values using the Wireshark data logs to confirm the reception of the correct information. The validation and confirmation of the simulated tests allowed for on-road testing to confirm these values.

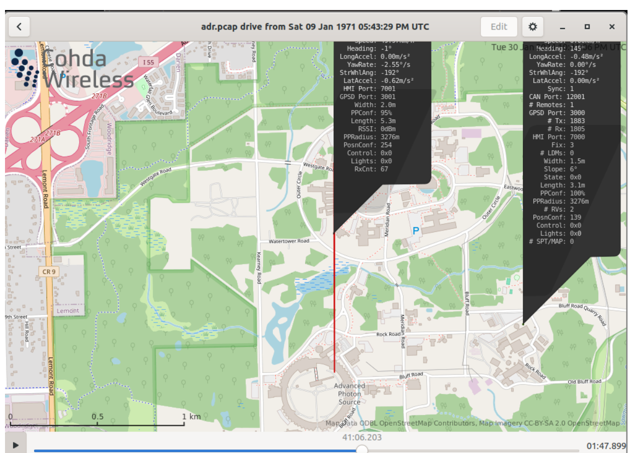

FIG. 6. Host Vehicle to Receiving Vehicle Test

3. Custom Repackaging Pipeline in C

Using Cohda Wireless's SDK and example1609, a custom pipeline was created within the source code using the C programming language to decode BSMs, SPaT, and MAP messages.

After decoding, the pipeline repackages the data and transmits it over the network using UDP. This involved the creation of custom type definitions in the C programming language, and using custom SAE J2735 objects to create a new struct to package data in a formatted method before being forwarded.

III. CONCLUSIONS

Development work from this project will complement activities across multiple projects in the lab including EcoCAR, CDA and Anything-in-the-Loop (XIL). Systems developed as part of this project will play a major role in development and deployment of CAV on-road energy consumption evaluation test procedures. The results of this research will help develop novel ways to estimate and compare efficiency of multiple CAV features and controls in realistic on-road scenarios. Furthermore, this work will expand ANL's vehicle testing capabilities to perform on-road feature and energy evaluation as part of its Lab2Road portfolio. Soon, FlexCorridor will the tested on-site at ANL. On successful testing and validation, it will be used at various test track for evaluating prototype vehicle controls. Support the Department of Transportation's (DOT) vision of large-scale deployment of V2X on-road and Department of Energy's (DOE) vision of developing energy efficient CAVs.

ACKNOWLEDGMENT

This work was supported in part by the U.S. Department of Energy, Office of Science, Office of Workforce Development for Teachers and Scientists (WDTS) under the Science Undergraduate Laboratory Internships (SULI) program. I would also like to recognize the support of my supervisor, Priyash Misra, for the mentorship, guidance, and patience this summer as I learned new skills and grew my confidence in the field as a researcher.

REFERENCES

[1] Federal Communications Commission, "Modernizing the 5.9 ghz band," Tech. Rep. (Federal Communications Commission, 2020) first Report and Order, Further Notice of Proposed Rulemaking, and Order of Proposed Modification – ET Docket No. 19-138.

[2]Society of Automotive Engineers (SAE), "Dedicated short range communications (dsrc) message set dictionary," Tech. Rep. (Society of Automotive Engineers (SAE), 2016) sAE Standard J2735®MAR2016.

[3]USDOT, "Connected vehicle tools," (2024), accessed: 2024-08-01.