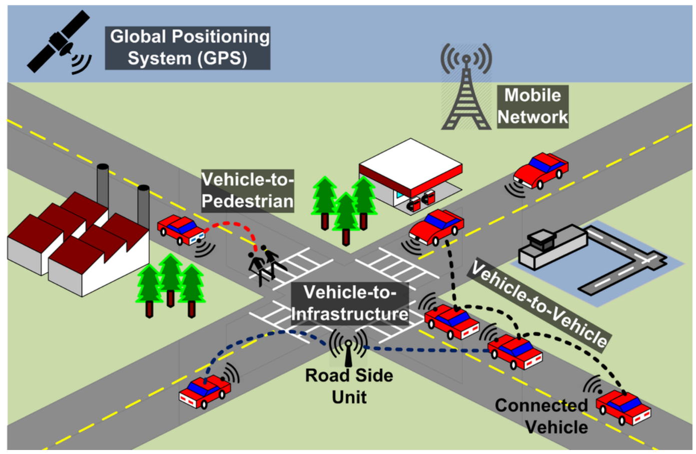
18-847SH Wireless Software Systems Architecture Implementation of IEEE802.11p protocol for V2I communications

Sai Vineeth Kalluru Srinivas, Gauri Shelat, Prathistha Dehadray
Carnegie Mellon University - Silicon Valley
December 17, 2018

Abstract—**Vehicle to Vehicle communications shortly known as V2V, is garnering quite an attention among automobile companies. The standard would allow vehicles to communicate with each other or with any infrastructure(V2I) to share data amongst each other that could be used for analysis. The V2I communication has become the need of the hour as the vehicles on roads have become more diverse. We have autonomous, semi-autonomous and ordinary(non-smart) cars all on the same road. Eye contact and other techniques will not work with an autonomous car. There is a need for a uniform communication system that is fast, effective and universally understood by all vehicles. This paved way for research in V2V and V2I communications. Latency plays a vital role in deciding the effectiveness of any communication protocol. This project is an implementation of an IEEE standard (802.11p protocol) for V2I communications. The project is an effort to understand the latency in a V2I system that implements 802.11p protocol.**

Fig. 1: Pictorial representation of vehicular communications



I. Introduction

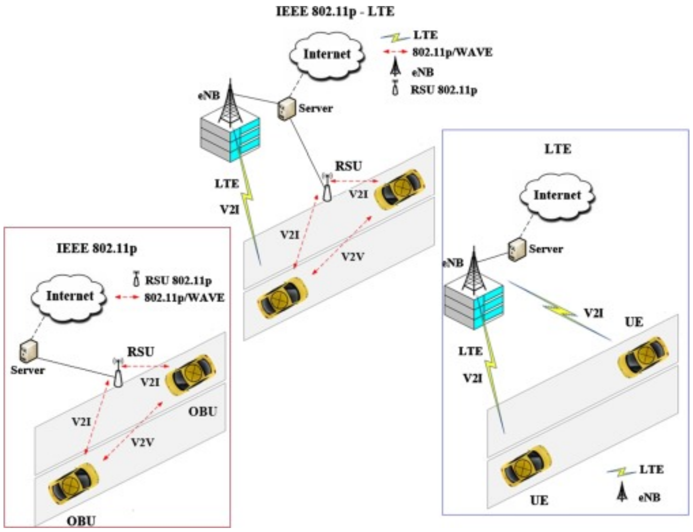
Vehicular communication is an active research in automobile industries. Every company is working hard to find the holy-grail of vehicle communication standard that could be monetized. The benefits of having a robust V2V communication is immense. Generally vehicle data implies vehicle speed, health, battery life, tyre health etc. This data is saved by vehicle manufacturer for improvising the vehicle model. However, vehicle communications involve different set of vehicle data namely, vehicle ID, vehicle speed, acceleration patterns, breaking time and others [1]. This data is shared on the fly to other vehicles nearby. This data is received by two different types of receivers, either it can be another vehicle or it can be an access point. Usually the access points are placed in intersections which process the incoming vehicle data and send the useful insights to the other vehicles nearby. The access point belongs to an *infrastructure* and therefore comes under Vehicle to Infrastructure(V2I) communication. Both V2V and V2I are in generally referred to as V2X protocol, where X can be replaced with V or I based on the application (see Figure 1). With the advent of autonomous vehicles on road, there is a need for an effective communication system to communicate lane changes, prevent accidents, inform braking etc. Daimler, Honda [2], Toyota and multiple other companies have set up research facilities that try different V2X protocols.

The V2X is only a concept and not the name of a protocol. The protocols that implement this concept fall under the category of WAVE(Wireless Access for Vehicular Environment) which is a sub-category of DSRC(Dedicated Short Range Communication. In the WAVE category many standardization organizations have attempted to devise a robust communication protocol with best performance. However, only LTE-V2V and IEEE802.11p protocols are well known in the V2X research community. There is an active discussion about which one of the two is the most effective [3]. We have chosen to work with IEEE802.11p protocol because of the following reasons:

1) IEEE has historically proved its extensiveness in standardization.
2) Follows the 802.11 wireless standard, therefore the physical layer can be easily extended from 802.11b(WiFi) to match 802.11p(V2V) requirements.
3) In 802.11p it is possible to attain the minimum latency possible because the cars can directly communicate with each other. However, in LTE the data has to reach a nearest cell tower which then sends to other vehicles, which adds latency(undesirable property).
4) Current cellular infrastructure can only address basic V2x use-cases, but lacks support for low latency and high mobility use cases. These are the items most closely associated with safety-related use cases.
5) It is difficult to mimic a cellular infrastructure in a closed lab environment.

An interesting use case of V2V communication is *Road collision avoidance system*. This project attempts to devise an algorithm to solve this problem statement with least latency possible. It is well known that no algorithm can cover all corner cases possible. Therefore, we have considered design-

Fig. 2: Network Infrastructure comparison between LTE and 802.11p protocols



ing only for one particular scenario. The scenario involves preventing accidents in road intersections with only two vehicles driving. The vehicles are perpendicular to each other and moving towards the intersection. Usually the field of view between vehicles on perpendicular roads is blocked by buildings around the corner. This might lead to accidents as the vehicles do not know that they are approaching each other. This project aims to prevent this problem by developing an algorithm over a V2I infrastructure implemented using IEEE802.11p protocol.

Contributions: This work is a comprehensive report of the project submitted as a part of Wireless Software Systems Architecture coursework. The team has successfully established V2V communication on IEEE802.11p protocol on USRP(Universal Serial Radio Peripheral) boards that would mimic the aforementioned problem statement. The report discusses about the methods used for implementing the protocol. The report also describes the methods used in determining the latency of the system. The latency of the system was observed to be between 0.03 to 0.1 seconds. This latency was found to be suffice for cars with speeds less than 30mph.

II. Related Work

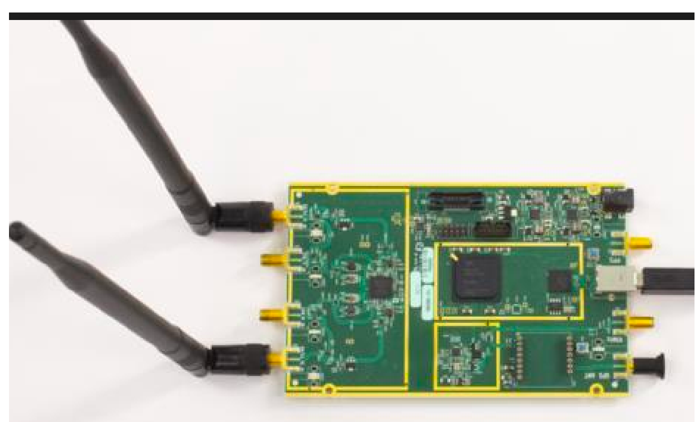
Direct Short-Range Communications (DSRC) communication band is the wireless communication network specifically for vehicular networks. The Federal Communications Commission set aside 75 MHz of spectrum around the 5.9 GHz band (5.850-5.925 GHz) band [4] 1999 to be used for vehicle-related, mobility systems. The DSRC band is a free but licensed spectrum and restricted in terms of the usages and technologies in comparison to other unlicensed bands [5].

In this project, we are developing a V2I network consisting of 2 vehicular nodes and an Infrastructure(Access Point). We worked on developing a collision algorithm for two cars heading towards a common access point and depending on the location and speed of each car we are to detect the possibility of a collision and warn the vehicles accordingly [6]. It is important to note that in this case the 2 vehicles have no direct line-of-sight. The system works by measuring the Received Signal Strength Indicator (RSSI) values for both the vehicular nodes as they transmit to the access point. The access point senses the RSSI values from each of the vehicles and uses it as a measure of proximity of the vehicle to the access point. The access point then uses these RSSI values to implement a collision algorithm which detects and warns the user of a possible collision. We are using the Network Timing Protocol (NTP) [7] to synchronize the 2 vehicular nodes to the NTP server and then use it to calculate the latency of the system.

IEEE802.11p protocol is more of a standard that defines a set of protocols on how vehicle to vehicle communications should happen. In practice there is a need to implement the physical layer on the hardware of network devices. The hardware implementation is quite tedious and requires industry level expertise. However, the team had utilized a the concept of Software Defined Radio(SDR) to implement the protocol on an FPGA based radio peripheral. These radio peripherals are known as USRPs. The most commonly used USRP is B200 (see Figure 3). USRP boards have FPGAs that can be programmed to mimic any physical layer protocol. The programs are usually programmed using GNURadio software. We would like to extend our gratitude to Carnegie Mellon Innovation Lab for providing us with the USRP boards.

The IEEE802.11p protocol was implemented by utilizing a *grc* library developed as a part of Wireless Measurement and Experimentation project by University of Trento and Paderborn University [8]. The library is hosted on github for public access []. It allows the developer to build IEEE802.11p based applications by providing MAC layer blocks that would match the protocol standards. The team could focus directly on implementing the protocol rather than programming the physical layer from scratch.

Fig. 3: USRP B200 Programmable FPGA based SDR



The latency measurement was recorded using the traditional NTP server/client architecture. NTP stands for Network Time Protocol which is a UDP based protocol used for sending timestamps, which can be used for time synchronization. NTP allows various devices in a network to synchronize their clocks within few milliseconds. The closer the server(with respect to router hops) the better the synchronization. The devices send periodic requests to NTP server with their timestamps. The NTP server, responds back with the device's original timestamps and that of a reference clock. The devices can then use this information to adjust their local clocks. Using the four timestamps we can calculate the offset value using the following formula.

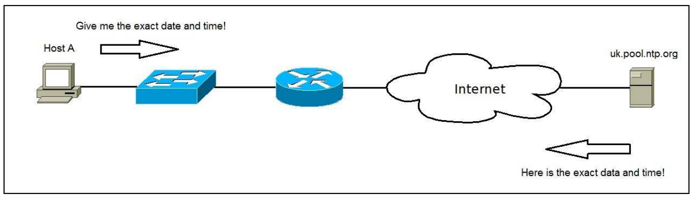
$$offset = [(T2 - T1) + (T3 - T4)]/2 \qquad (1)$$

We also calculate round-trip network delay from the four timestamps. The delay is calculate using the following formula :

$$delay = (T4 - T1) - (T3 - T2). \qquad (2)$$

The offset value obtained from the above equation is used to adjust our local clocks to synchronize with the server clock The NTP server/client architecture is shown in Figure 4

Fig. 4: NTP Server/Client architecture



The IEEE802.11p standard is a part of wireless standards family of IEEE802.11 protocol. The protocol was designed with enhancements required to support Intelligent Transportation Systems (ITS) applications. This includes data exchange between high-speed vehicles and between the vehicles and the roadside infrastructure, so called V2X communication, in the licensed ITS band of 5.9 GHz (5.85-5.925 GHz). The differences between the WiFi protocol and V2V protocol in IEEE is shown in Figure 5

Fig. 5: Comparison between 802.11a and 802.11p in physical layer implementation

Parameters	IEEE 802.11a	IEEE 802.11p	Changes
Bit rate (Mb/s)	6, 9, 12, 18, 24, 36, 48, 54	3, 4.5, 6, 9, 12, 18, 24, 27	Half
Modulation mode	BPSK, QPSK, 16QAM, 64QAM	BPSK, QPSK, 16QAM, 64QAM	No change
Code rate	1/2, 2/3, 3/4	1/2, 2/3, 3/4	No change
Number of subcarriers	52	52	No change
Symbol duration	$4\mu s$	$8\mu s$	Double
Guard time	$0.8\mu s$	$1.6\mu s$	Double
FFT period	$3.2\mu s$	$6.4\mu s$	Double
Preamble duration	$16\mu s$	$32\mu s$	Double
Subcarrier spacing	0.3125 MHz	0.15625 MHz	Half

III. Approach

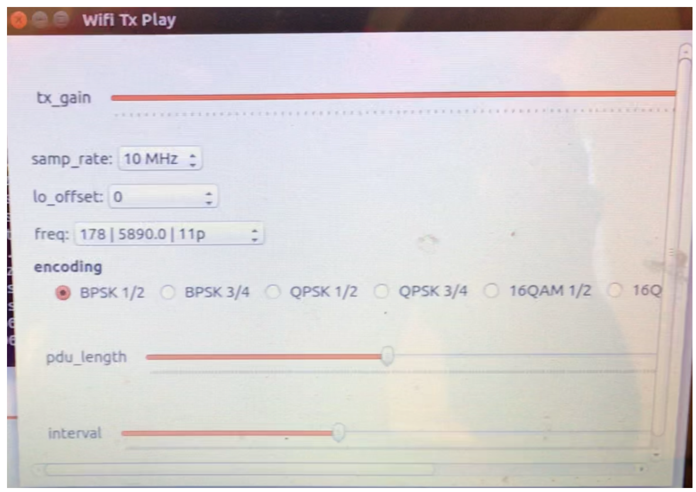
A. Example application

The IEEE802.11p library [9] was installed and implemented in a test program. The test program is a WiFi-loopback program. This particular program is just to check if the library implements physical layer properly. The loopback indicates that the frames are simulated to have been created in the PHY layer and the frames that received over the loopback are displayed on the console. The loopback file works similar to normal loopback application in IP networks. The loopback application test is followed by example wifi_tx and wifi_rx file. At the end of step it is verified that the library is working as intended.

B. Implementation of 802.11p for tx and rx

The 802.11p protocol communication was established using example tx and rx files. The choice of choosing the protocol was available as a drop down in the output GUI (see Figure 6. Once the communication is established the GRC file was modified to fit the requirements of our project. A V2I communication has two main categories of nodes in the network, they are: Vehicle Node and Access Points. The vehicle node is present on the moving vehicles and the access point is present in road intersections.

Fig. 6: The output GUI of transmission file. Notice the *11p* chosen in the drop down.



The vehicle node packs the following data into a IEEE802.11p packet and sends it to the access point [10].

1) Vehicle ID
2) Route Pattern ID
3) Distance and time to intersection
4) Vehicle speed and direction
5) Estimated travel time to the intersection
6) Schedule Adherence
7) Real-time passenger load

The access point processes this data and shares the results to other vehicles nearby. In this project we are worried about sending vehicle ID only. It is trivial that if vehicle id can be sent then other data points can be transmitted as well. The protocol allows for a 150Mbps link between tx and rx. Since we are not working with real cars there is a need for an alternate method to derive *speed of vehicle* and *time to intersection* data points. The alternative method is discussed in the following subsections.

In this subsection, we would discuss about the method used to establish communication between vehicular sensor embedded unit and USRP. The sensor unit in vehicles are usually micro controller units that would sample sensor values over time. This embedded systems is simulated using a PowerDue board(derived from Arduino Due) which sends placeholder sensor data. The data is sent over the serial port at the speed of 9600baud. The GRC file in the USRP board is modified to accept values from serial port. The following steps were followed.

1) Receive input from Serial port by using *File Descriptor source* block. Changed the source file to */dev/ttyACM0*(serial port connected to powerdue) .
2) Convert incoming data from Stream data to Tagged PDUs
3) Convert Tagged PDUs to normal PDUs
4) Parse the PDUs to send data over physical channel.

The above procedure ensured that any data from PowerDue is accepted, converted to packets and transmitted by the USRP board from Vehicle A. These packets would reach the access point which would act as the receiver. Assuming that the packets are processed by access point, the next destination for the packets would be the neighboring vehicle(Vehicle B). Vehicle B receives this packet but needs an MCU unit to process the data. Therefore, PowerDue is used again in the receiver end. However the communication in this case is reversed. We need to establish communication from USRP to PowerDue. To solve this problem the following steps were followed

1) Identified the block that parses the incoming data.
2) Open the corresponding *c file* of the block.
3) Identify the serial port to which receiver's PowerDue is connected
4) Modified the C file to open the powerdue serial port and write data to it, whenever data is parsed in receiver's end.
5) Build the c file again
6) Checked for the proper working of the modified source file.

Therefore, we have established an end to end connection between vehicular nodes(PowerDue boards) via USRP boards. The PowerDue to USRP communication is predominantly serial port communication. The speed of communication can be tuned based on the application.

C. Extracting signal strength information

As discussed above, the speed of the vehicle and time to intersection is not available to us. To tackle this problem we have used a concept that employs signal strength data for determining speed and time calculation. For this project, there is no explicit need to get the exact speed and accuracy. The speeds are required relative to each other. The relative speeds can be calculated by the common entity between both vehicles, i.e. the access point. We know that signal strength increases with decreasing distance between tx and rx. This concept is utilized to determine relative speeds using the procedure below:

1) Find a method to extract signal strength information of each packet at access point.

TABLE I: Distance and signal strength correlation lookup table

Distance from AP(m)	Average Signal Strength(dB)
1.0	22
4.0	20
6.5	16
7.0	13
10.0	9
13.2	7
15	5
19	2
20	1

2) Make a lookup table a priori that would correlate *distance from access point* and *average signal strength* obtained at the given distance.
3) Compare the signal strength of the packet to the lookup table and determine the first distance coordinate(d1) of the vehicle from AP.
4) Wait for fixed time.
5) Receive second packet and correlate its signal strength to determine second distance coordinate (d2).
6) Subtract distance co-ordinates and find out effective distance traveled in the chosen time slot(t).
7) Determine speed of the vehicle given distance traveled and time taken using $Speed_{avg} = (d1 - d2)/t$.

The first step in above procedure is a challenge because the library we use does not explicitly provide blocks to determine signal strength information of the packet. The library only allows us to determine the signal strength by browsing through the corresponding *wireshark capture file*. Reading capture file and then determining signal strength is a slow process which adds latency to the system. Therefore, it was decided to modify the library source code to extract signal strength information directly. The *decode_MAC* block was responsible for decoding the incoming packets which also had first hand information about signal strength. The code of this file was modified for extracting SNR values.

The lookup table mentioned in the second step is shown in Table II. From the table, we can observe that signal strength decreases with increasing distance from access point.

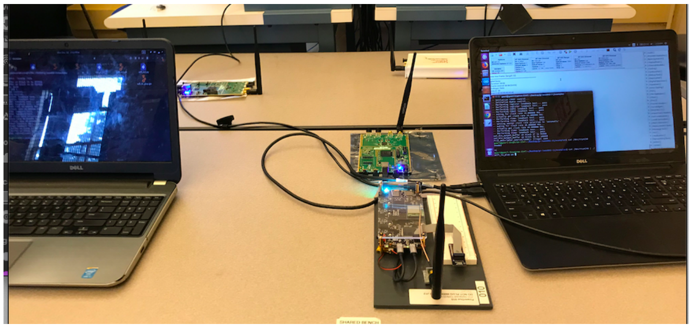
D. Implementing the protocol for two vehicle nodes

The project requires two vehicle nodes for prototype. The same transmitter files are programmed onto to different USRP boards. The third USRP board is programmed with receiver code which would act as access point. Ths system is shown in Figure 7.

The following specifications were implemented in the prototype.

1) 5.89GHZ CENTER FREQUENCY
2) UNLICENSED BAND
3) LOW LATENCY
4) 10 MHZ CHANNEL WITH A CAPACITY OF 27Mbps
5) 3 NODE NETWORK

Fig. 7: The setup involves two USRPs as vehicle nodes and the USRP(white) is used as access point.



6) SYNCHRONISED
7) RSSI AWARE PROTOCOL
8) BPSK ENCODING
9) PACKETS RATE: 10 PACKETS PER SECOND FROM EACH VEHICLE NODE
10) 20 PACKETS PROCESSED PER SECOND AT ACCESS POINT.

E. Signal strength based collision detection algorithm

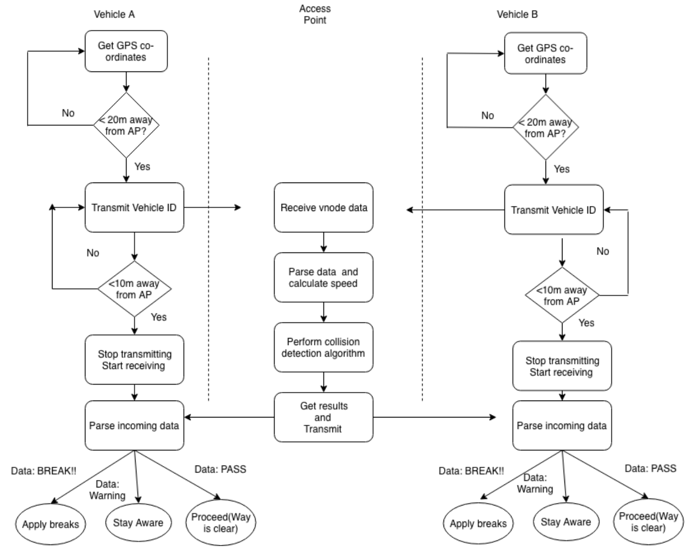
With the system setup and ready for experimentation. We have designed an algorithm that would take 10 packets every second from each vehicle node, analyze the speed and determine if a collision would be possible.

The algorithm is explained in detail in the flowchart shown in Figure 8. The process is enumerated as below:

1) Allow vnode1 and vnode2 to transmit their vehicle IDs within 20 meter distance from Access point.
2) Access points uses the data and correlates with signal strength of packets to the speed of the vehicle.
3) *Redzone* represents the area of 10m radius around the access point.
4) The access point is ready with speed values of both vehicles A and B before the vehicles approach *redzone*.
5) If Vehicle A steps into red zone first, then three scenarios are possible
   a) Scenario 1: If Vehicle B is already inside red zone, then immediately send *BREAK!!* signal.
   b) Scenario 2: If Vehicle B is inside redzone at the exact same time as Vehicle A, then compare speeds and send *WARNING* to the slower of the two vehicles.
   c) Scenario 3: If Vehicle B is far from entering redzone, then send *PASS* signal to Vehicle A.
   d) Allow the vehicles to respond to the signal from access point.
   e) Optionally an acknowledgement signal can be used.

This algorithm is a simple design of collision avoidance system that uses signal strength data to determine relative speeds and take decisions accordingly. Even though signal strength is not the best measure, it was still found effective in implementation for lab environments.

Fig. 8: Figure shows the flowchart depicting a simple collision avoidance algorithm



F. Latency measurement

We calculate latency for the V2I link by using NTP protocol. We installed NTP server on the Access Point Linux computer. This server was switched to the 'PowerDue' network. Next we implemented NTP client for the PowerDue. Keeping both the server and the client on the same network improves the synchronization.

The NTP clients send messages to the server over UDP, hence we created sockets using the lwIP stack for UDP transactions. Next, we created a task that would generate periodic requests to NTP server. This task is created as a FreeRTOS task. The task creates an infinite loop, that sends a packet and then waits for the specified duration. The wait duration is specified according to the required rate at which requests need to be sent to the server. This rate, in turn, is determined by the granularity of synchronization required. The task obtains the current timestamp from the local clock. NTP request packet is then created by embedding this timestamp. Once the packet is created, it is sent to the NTP server and waits for the specified time.

Next, we create a task to receive data from the server. This too is created as a FreeRTOS task, consisting of an infinite loop. This task obtains data from the socket created. The packet received consists of 3 timestamps. The first is the original timestamp obtained from the local clock at the time, the request was generated(T1). The server adds two timestamps from the reference clock. The first server generated timestamp corresponds to the time when the request was received by the server(T2). The second timestamp corresponds to the time when the response was sent(T3). We then obtain a fourth timestamp from our local clocks corresponding to the time when the response from the server is received by our device(T4). Using these timestamps we calculate the offset value. This offset is added to the local clock to synchronize it with the server.

IV. Results

A. Protocol implementation results

The output of the three node V2I system is shown in figures **??** and 10. From the figure we can see that access point is able to extract RSSI value and also the data present in the packet. Here, *fig:vnode1* is the data sent by Vehicle A and *fig:vnode2* is sent by Vehicle B.

Fig. 9: The output at the access point parsing data from Vehicle A

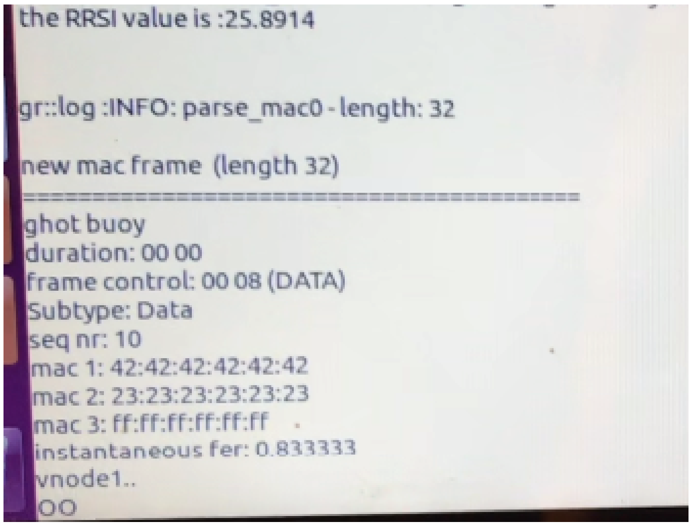
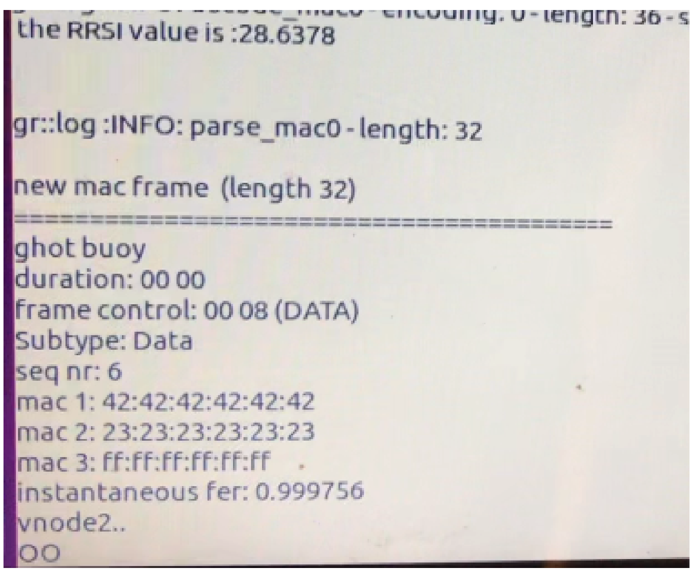


Fig. 10: The output at the access point parsing data from Vehicle B



The look up table discussed in the last section is graphically represented in figure 12. We can observe the signal strength exponentially decreasing with increasing distance from access point.

V. Analysis

A. Analysis of latency data

Table 2 shows the maximum, minimum and average latency values. We can see that the latency varies from 0.1 to 0.3

Fig. 11: Latency value in micro-seconds

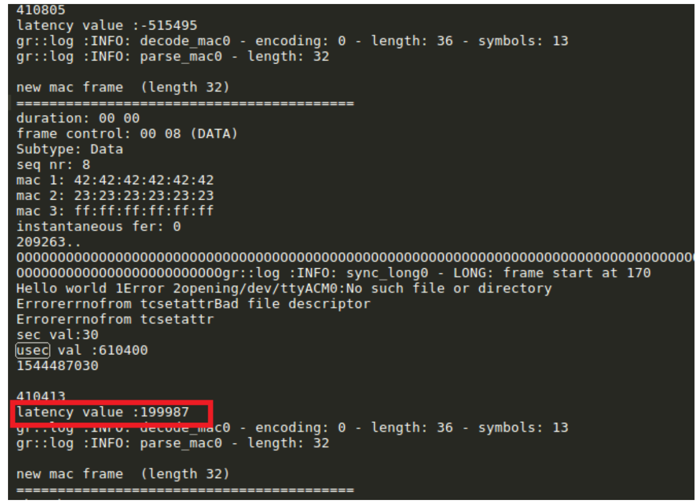
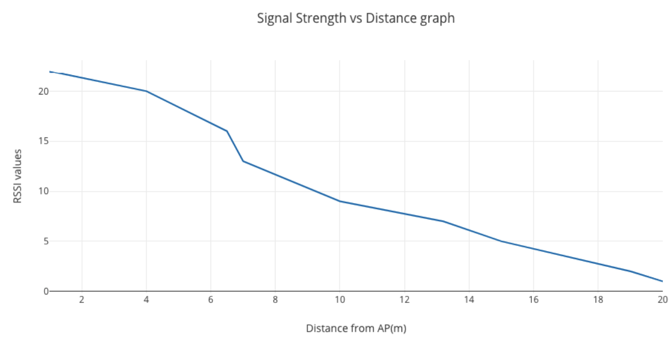


Fig. 12: Signal Strength vs Distance graph



seconds. Most of the current literature discusses the latency values obtained from models in simulation. The value of latency for 27Mbps varies from around 0.2s to 3s depending on the payload size. We have a very small payload size and hence less latency values. Moreover, in our experiment, we do not have the problem of network congestion. 802.11p protocol uses CSMA-CA at MAC level which uses exponential back-off to avoid packet collisions. In real-life scenarios this will increase the latency values. This experiment is also conducted inside a lab, and indoor and outdoor environments significantly change for wireless networks.

The latency values are quite difficult to comprehend as such. The values are analyzed over *car time*. *Car time* indicates the distance traveled by the car within the time considered. The access point will use the time taken for the car to cover from 20m to 10m distance, from access point. Once the speeds are determined, the access point has 10m of car time to react and send information before any possible collision. Given this scenario let us analyze the case for a car moving at a speed of 30mph. 30 miles per hour would be 13.4 meters per second. The latency values range from 0.03 to 0.1 seconds, which translates to a *car time* of 0.402m to 1.34m. If the algorithm takes another 0.3 seconds (worst case) to execute, we would be left with total system latency of 0.33 to 0.4 seconds. This

TABLE II: Latency Values

Latency	Value(s)
Min	0.17
Max	0.58
Average	0.24

would again translate to a *car time* of 4.02m to 5m. This implies that by the time the car covers 5m into the *redzone*, the algorithm would determine collision and transmit information to vehicles accordingly. The usual breaking distance for cars is 4m. So, 5m is within the range of distance that would allow a 30mph vehicle to come to a smooth halt. Therefore, based on the latency values of the system we can define the upper limit of speed of vehicles.

VI. Conclusion

We have successfully implemented 802.11p protocol on USRP boards that would mimic a V2I network infrastructure on road intersections. The latency of the system was chosen as a metric of performance. All measures have been taken to ensure least latency as possible. The latency values ranging between 0.03 to 0.1 seconds is practical only for vehicle speeds less than 30mph. In reality, vehicles are usually faster than 30mph. Therefore, there is a need for an algorithm that could be more faster. The algorithm proposed could not be implemented within the time frame offered for the project. However, the algorithm implementation is added as a part of future work. The current project deals only with two vehicle nodes, the concept can be extended to study multiple vehicle nodes. The project testing was conducted in an indoor environment, whereas wireless communication significantly varies in outdoor environment bringing the need for outdoor testing. Car length should also be considered for collision detection analysis(which was neglected for simplicity). The hardware on the project is bulky, we are planning to use cheaper and smaller hardware that could be more practical for real time usage as a part of future work. We would like to extend our gratitude to Prof. Bob Iannucci for providing us with USRP boards and lab setup for the project.

References

[1] K.-Y. H. et al., “Implementation of WAVE/DSRC Devices for vehicular communications,” 2010 International Symposium on Computer, Communication, Control and Automation (3CA).

[2] “Honda Demonstrates Advanced Vehicle-to-Pedestrian and Vehicle-to-Motorcycle Safety Technologies,” Honda.com, Aug. 28, 2013.

[3] Z. et al., “LTE and IEEE 802.11p for vehicular networking: a performance evaluation,” EURASIP Journal on Wireless Communications and Networking 2014.

[4] J. B. Kenny, “Dedicated Short-Range Communications (DSRC) Standards in the United States,” Proceedings of IEEE Volume: 99, 7 July 2011.

[5] Y. L. M. et al., “Managing DSRC and WAVE standards operations in a V2V scenario,” International Journal of Vehicular Technology, vol. 2010, Article ID 797405, 18 pages, 2010.

[6] “Inter Vehicle Hazard Warning,” https://www.youtube.com/yGm2NVhV9jw.

[7] “Network Time Protocol,” https://en.wikipedia.org/wiki/NetworkTimeProtocol.

[8] “WIME project information page,” http://www.ccs-labs.org/projects/wime/.

[9] “Github link to IEEE802.11p library,” https://github.com/bastibl/gr-ieee802-11.

[10] “DSRC implementation in San Francisco detailed on the type of vehicle data shared,” https://www.apta.com/mc/its/previous/2014/program/Documents/San-Franciscos-Implementation-of-DSRC-5.9-GHz-Daniel-Buckley.pdf.