American Journal of Advanced Research, 2020, 4–2
December. 2020, pages 1-6
doi: 10.5281/zenodo.4027903
http://www.ibii-us.org/Journals/AJAR/
ISBN 2572-8849 (Online), 2572-8830 (Print)

Detection of Wrong Way Driving using V2I Communication

Muhammad Z. Hasan[1,*], Nicholas Trevino, Rene Chavez

School of Engineering, Texas A&M International University, 5201 University Boulevard, Laredo, TX 78041

*Email: muhammad.hasan@tamiu.edu

Received on June 24, 2020; revised on August 30, 2020; published on September 13, 2020

Abstract

Wrong Way Driving (WWD) is an ongoing problem that requires constant attention in order to prevent accidents and to protect the general public. Texas Department of Transportation (TxDOT) in San Antonio has initiated a program to implement solutions to mitigate WWD accidents. As technology continues to advance, it is important to create innovative solutions to counteract the WWD problem. In intelligent vehicles and vehicles with a heads-up display (HUD), we can utilize the on-board computer of Vehicle to Infrastructure (V2I) communication system to detect WWD and to alert the authorities. This project simulates the On-Board Unit (OBU) of the car and the Roadside Equipment (RSE) that will interact with each other to detect the WWD. Proper interaction between OBU and RSE is implemented for data transfer to detect the WWD. Test results show that the RSE takes 3.61 ms to notify the vehicle of a WWD. The worst-case violation can be detected within a distance of 3.81 inches in the ramp.

Keywords: Wrong Way Driving, Vehicle to Infrastructure Communication, Detection Time

1 Introduction

The objective of this project is to design and implement an innovative solution to mitigate wrong-way driving in general and to measure its performance. This was accomplished by working with Texas Department of Transportation (TXDOT) in order to accurately develop a system that successfully contributes to the solution. Several solutions have already been implemented, which can be seen on the road. These solutions may include road bumps, lowered sign heights, and signs that light up as stated in [1][2][3]. However, wrong way driving is a consistent problem, and these solutions have only reduced the number of incidences marginally [4][5][6][7]. With this in mind, the goal of this project is to look at what is already being used and to figure out how to create solution based on future technologies [8]. As such, it is important to continue designing and implementing solutions that have not been completed before or adjusting those that have been done. Some recommendations have been made which includes the use of on-ramp pavement markings because it has been reported that intoxicated drivers have a tendency to look down as mentioned in [2] and in [3]. As such, one of the main focuses is to get the driver's attention. Intelligent Transportation System (ITS) techniques were used and reported by authors in [9] and in [10] stating increased number of detections. However, the performance in terms of detection time is not specified in these studies. A vendor reported performance in terms of detection time in [11]. It lacks the capability of warning the wrong way driver. As such, an innovative system is needed that addresses these constraints. This project uses GPS coordinates to detect the wrong way driver, to warn the driver, to warn potential victims of the risk involved, and to contact the local authorities to handle the situation. This project uses Vehicle to Infrastructure (V2I) communication to interact with the cars and the infrastructure. The project seeks to utilize embedded processor boards to simulate V2I communication that detects a wrong way driver and alert others within a certain vicinity to mitigate possible accidents. The hardware and software of this project was accomplished in the laboratory of the School of Engineering at TAMIU. This paper presents the design, implementation, testing, and performance results of the system.

2 Hardware Development

To utilize future V2I communication technology in order to detect WWD, hardware for both the OBU and RSE need to be developed. As such, for the project to be simulated successfully, RSE and OBU must be implemented and integrated so that they may send and receive required data. In this section we will detail the hardware development involved in the project.

Serial data has proven to be the ideal option for this project, since timing issues, initializations, and software organization issues with parallel data could not be corrected. For serial data to work properly, certain wired connections must be made between OBU and RSE receiving and transmitting ports. RSE and OBU are connected serially, as shown in Figure 1.

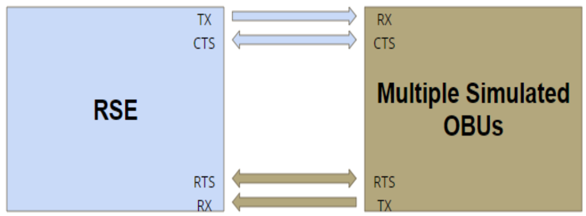

Figure 1. RSE and OBU Ports Connected Serially

Silicon Labs C8051F340-TB board was used which has an 8-bit 8051 microcontroller with 40 data pins, 4Kb ROM storage and 128 Bytes RAM. The built-in serial port of the board was used for data communication. Two of these boards were used, with one board being the RSE and the other the OBU. A Dell Optiplex 7050 with Intel Core i5-7500 was used to run the development environment and debug the programs running on these boards. An oscilloscope was also used to scan data transmission. A breadboard and a red LED light were used to check the communication. With Silicon Labs, a default code is provided to toggle the LED on the board. With the aforementioned equipment, we first tested the default code, then used this as the base for developing and testing the software later on.

3 Software Design

Two different programs need to be created, one for the OBU and the other for the RSE. Each program needs to be developed independently, while maintaining synchronization of sending and receiving data with the other. The flow diagrams for the RSE and OBU are presented and explained below.

In Figure 2, the algorithmic steps that the RSE follows are shown in a flow diagram.

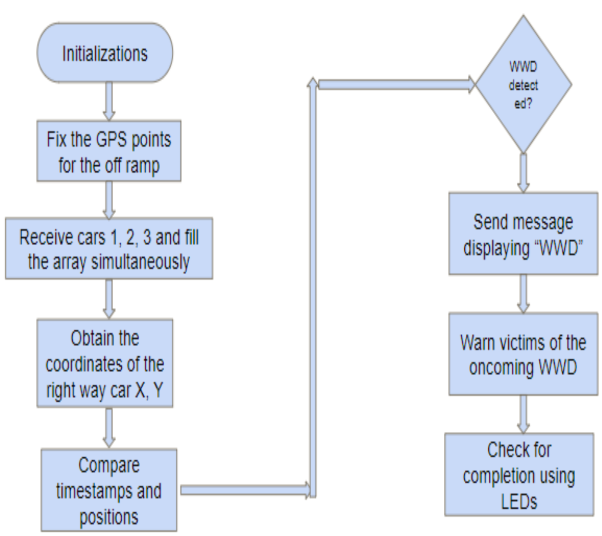

Figure 2. Roadside Equipment Flow Diagram

First, the initialization routine configures the necessary setting for the serial data communication. Next, the GPS reference points (1, 2, 3) are initialized representing right sequence on the ramp as shown in Figure 3.

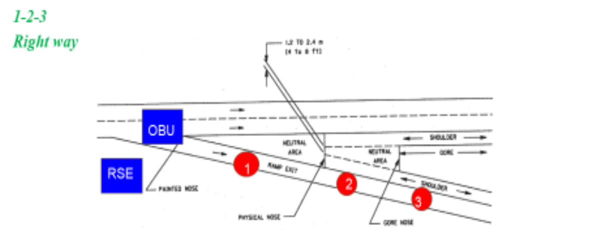

Figure 3: Ramp Showing Right Way Driving

Next, the car information is received through a synchronized two-way serial data communication. Only three car-data are sent at a time due to hardware limitation (that will be further discussed in the conclusion). After data is received and stored, the comparison begins. The RSE checks the GPS points received and the timestamps against the fixed reference points. If there is any violation (car traverses in sequence 3, 2, 1), then the RSE will send a wrong way message to the wrong way driver as well as a warning to the possible victims in the vicinity. However, if no wrong way driver is detected (car traverses in sequence 1, 2, 3), then it will stay in a loop constantly receiving new data from passing vehicles and checking if they are in violation.

In Figure 4, the flow diagram of the OBU is displayed.

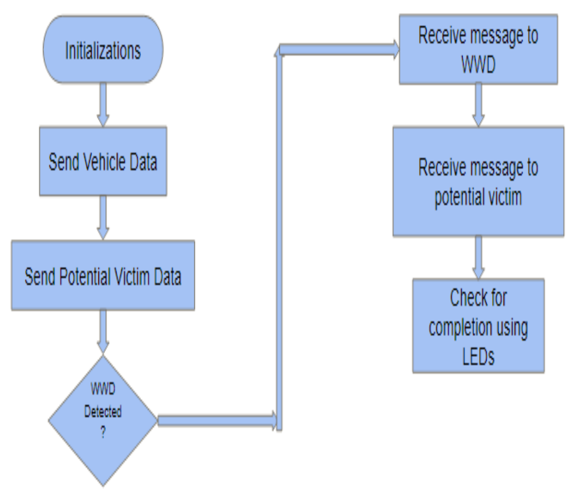

Figure 4. On-Board Unit Flow Diagram

As mentioned in Figure 3, the On-Board Unit will send an initialization byte to the RSE, then it will send the vehicle data which is the first byte. After the vehicle data is sent, then potential victim data is sent and if there is no wrong way driver, the information will continue to flow until there is a wrong way driver detected. Once a wrong way driver is detected, then the OBU will receive the message for the wrong way driver as well as the message for the possible victims in the area.

In Figure 5, the unit communication protocol between the RSE and OBU is presented as layers with sequence.

Figure 5. Unit Communication Protocol

In the figure above, the data on the left pertains to the RSE and the data on the right pertains to the OBU. Initiation starts from the RSE to the OBU, which then waits to send the first byte. Once the OBU sends the first byte then the RSE sends an acknowledgment to the OBU. Once the OBU receives the acknowledgment, then it sends the second byte of information to RSE. This will continue during the whole process of data exchange. The RSE uses this stored data to detect and warn the WWD as well as identify and warn potential victims.

An adapted format of vehicle data is shown in Figure 6. The vehicle ID, longitude and latitude and timestamp are used to perform the above detection. The heading information can be used to determine the direction of the vehicles.

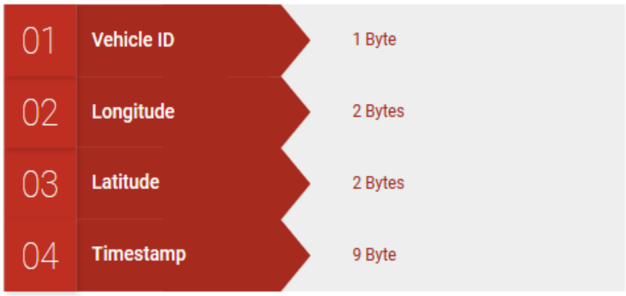

Figure 6. Data Sent by the OBU

Figure 6 also shows the size of data sent by the OBU. As seen, one byte of data is needed for Vehicle ID because it can be any number from 00 to FF in hexadecimal. Longitude and latitude need two bytes of data because of the decimal point. Timestamp requires 9 bytes because it is a 10 digit number.

In Table 1, example real data format received by the RSE is presented. This real data format contains numbers with decimal points. However, in this project simplified integer data formats were considered due to hardware and time constraints.

Table 1. Example Data Received by the RSE

Vehicle ID	Location	Speed	Heading	Timestamp
1225	33.844°, -112.135°	11.81	165.7	14135867
1229	33.875°, -112.124°	11.01	165.2	14135867
1234	33.768°, -112.121°	11.43	166.4	14135867

4 Test bench and Performance Results

The system was formed by integrating the above developed hardware and software. The actual test set up is shown in Figure 7.

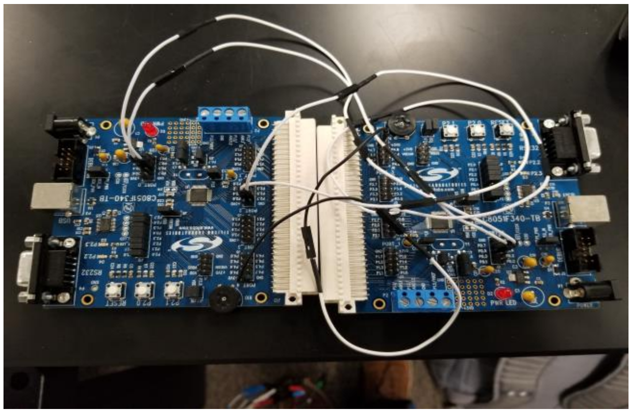

Figure 7. Test set-up of the OBU and RSE

A set of random traffic data was created in Visual Studio for testing the system. The algorithmic steps to generate traffic data are outlined as follows.

1. Randomly generate one byte vehicle ***ID*** (between 1 and 255), convert it to hexadecimal format for OBU.
2. Randomly generate two byte ***Longitude*** (between 1 and 65535), convert it to hexadecimal format for OBU.
3. Randomly generate two byte ***Latitude*** (between 1 and 65535), convert it to hexadecimal format for OBU.
4. Randomly generate one byte ***Timestamp*** (between 1 and 255), convert it to hexadecimal format for OBU.
5. Arrange them as parameters of a function call, write them into the traffic data file.
6. Repeat the steps 1 to 5 for 50 vehicles.

In addition, when a WWD need to be simulated, the following additional steps are taken.

1. For a particular vehicle ID, generate two byte ***Longitude*** (between 1 and 65535), equal to ***Point 3*** of ramp, convert it to hexadecimal format for OBU.
2. For a particular vehicle ID, generate two byte ***Latitude*** (between 1 and 65535), equal to ***Point 3*** of ramp, convert it to hexadecimal format for OBU.

3. For a particular vehicle ID, generate one byte ***Timestamp*** (between 1 and 255), convert it to hexadecimal format for OBU.
4. Repeat the steps 1 to 3 for ***Points 2 and 1*** with increasing ***Timestamps***.

For WWD & potential victims, further steps are added as follows:

1. For a particular vehicle ID, generate two byte ***Longitude*** (between 1 and 65535), in the ***vicinity*** of the ramp, convert it to hexadecimal format for OBU.
2. For a particular vehicle ID, generate two byte ***Latitude*** (between 1 and 65535), in the ***vicinity*** of the ramp, convert it to hexadecimal format for OBU.
3. Randomly generate one byte ***Timestamp*** (between 1 and 255), ***similar*** to that of the WWD, convert it to hexadecimal format for OBU.
4. Repeat the steps 1 to 3 for a preset number of vehicles.

A set of five traffic data generated from this random traffic generator is enlisted as follows.

DataTransfer (0x61, 0xb5, 0xc1, 0x8f, 0x29, 0x50);
DataTransfer (0x2d, 0x4a, 0x9a, 0xa2, 0x53, 0x68);
DataTransfer (0x14, 0x83, 0x67, 0x5c, 0x58, 0x57);
DataTransfer (0x6f, 0x39, 0x2f, 0xba, 0x3c, 0xab);
DataTransfer (0x16, 0x46, 0x99, 0x5d, 0x1e, 0x35);

In the above data set, DataTransfer is the function that invokes data transfer from the OBU to RSE. Its parameters are as follows:

Parameter 1 : Vehicle ID
Parameters 2 and 3 : Vehicle Longitude
Parameters 4 and 5 : Vehicle Latitude
Parameter 6 : Timestamp

This data-set contains vehicle information as shown in Figure 6 for 50 vehicles. Some vehicle-data contained WWD. Three scenarios were considered to cover all realistic possibilities around a highway exit ramp, as shown in Table 2.

Table 2. Characteristics of Overall Data Sets

Set	Characteristics
1	Data without a WWD
2	Data with only a WWD
3	Data with a WWD & Potential victim

These data sets were the base on which other sets were created. In addition, data sets with WWD had following additional properties, shown in Table 3. The philosophy behind the WWD data sets was that the system should be able detect WWD at any point of the group of vehicle-data sent to RSE. Although, more possibilities could be considered, three of them were chosen for simulation.

Table 3. Characteristics of WWD Data Sets

Subset	Characteristics
1	WWD near the beginning of the set
2	WWD near the middle of the set
3	WWD near the end of the set

This data set was fed to the OBU for transmission to RSE. Conceptually the overall test bench can be visualized as in Figure 8.

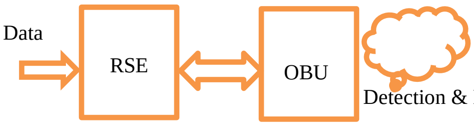

Figure 8. Conceptual Diagram of the Test Bench

By analyzing the underlying code, the time taken to detect the WWD was determined. For the RSE, reception of vehicle data and comparison processing time is about 301 machine cycles, which is about 75.25 microseconds. When the OBU transmits data, the preparation time is about 136 machine cycles, which is about 34 microseconds. OBU requires fewer machine cycles because it is just sending data. In addition, there is transmission time for data. The total time the system takes from the preparation to detection is equal to the sum of the OBU preparation and transmission time as well as the RSE reception and processing time. The following equation shows the total system time from transmission to detection.

Total System Time from Preparation to Detection = $T_{OBU} + T_{RSE}$…………………………………………. ……………………………. (1)

In equation (1),
T_{OBU} = *Preparation time + Transmission Time*
T_{RSE} = *Reception Time of vehicle data + Comparison Processing Time for Detection*
Data is sent for a group of three vehicles. The time to detect WWD with reference to the first, second, and third vehicle are computed as follows:

First Vehicle to Detection = 3.534 ms + 75.25μs = 3.61 ms (***worst case***)
Second Vehicle to Detection = 2.531 ms + 75.25μs = 2.61 ms
Third Vehicle to Detection = 1.268 ms + 75.25μs = 1.34 ms (***best case***)

It takes longer to detect the violation of first vehicle as compared to the violation of third vehicle. It is noteworthy to find the time it takes from detecting WWD to notify nearby vehicles. It is calculated as follows:

Machine cycles from detection to notification = 26
Notification processing time = 26 * 0.25 μs = 6.5 μs

Therefore, the RSE takes about 6.5 microseconds to notify the vehicle of a WWD after detection.

Detection time enables one to calculate the distance traveled by a vehicle before obtaining a warning notification. The calculated distances are presented as follows in Table 4 with the most relevant and realistic speeds that could be observed on a ramp.

Table 4. Distance (in inches) Travelled between Transmission and Detection

Speed	**Best case (3rd in Violation)**	**Worst case (1st in Violation)**
40 mph	0.94	2.54
50 mph	1.18	3.18
60 mph	1.42	3.81
Average	*1.18*	*3.18*

As shown above, the best and worst case scenarios have been calculated to give the best estimates of distance. The average distance was 1.96 inches. In the best case, we can notify the driver within 0.9 inches and in the worst case scenario, the driver will be notified in 3.72 inches.

In addition, the distance traversed by a vehicle in the ramp between its data transmission and notification was also calculated as shown in Table 5.

Table 5. Distance (in inches) Travelled between Transmission and Notification

Speed	**Best case (3rd in Violation)**	**Worst case (1st in Violation)**
40 mph	0.95	2.55
50 mph	1.19	3.18
60 mph	1.42	3.82
Average	*1.19*	*3.18*

It reveals that this distance is slightly more than that in Table 4. Average of all these distances is 2.18 inches. For practical purposes, either of these times (transmission to detection or transmission to notification) could be used with satisfactory outcome.

5 Conclusions and Recommendations

A wrong way driving detection system using V2I communication has been implemented and tested. Test results show that wrong way driving can be detected within 3.61 ms after violation. This means the wrong way driver can be notified within 3.81 inches after crossing the reference detection points. Potential victims can also be notified within additional 6.5 µs after detection.

For this project, challenges and errors were encountered. There was a steep learning curve for the hardware and software design environment. So, appropriate time need to be allocated for learning the design environment. The next significant error encountered was that for data transfer. Sending the data in parallel was causing major synchronization problem. Since this was proving to be ineffective, we decided to switch to serial data which worked effectively. So, use of serial RS 232 communication is recommended. Also, for serial data communication, it was expected that a DB9 cable could connect both units together. However, the data was not sent from the OBU to RSE using this cable. The solution was to connect the receiving and transmitting ports manually with jumper wires.

Beside these errors, there were several limitations of the hardware and software used. For the hardware, there was limited memory available to use. The amount of memory available as stated earlier is 128 bytes of RAM, which meant a maximum of 3 cars data before the program would fail. A larger memory is preferred. The software was limited in terms of libraries available. For instance, there are no random variable functions or time functions available which would have assisted the project. In order to produce random numbers for traffic data, Visual Studio was used to develop the vehicle ID, timestamps, and locations. This random data was then transferred to OBU.

This project works entirely on wired serial communication. Making this project to work wirelessly would be more realistic. Only then could this actually be tested in a real-world application. The OBU could actually be placed inside vehicles to obtain real numbers and actual data. With wireless implementation, new challenges (such as electromagnetic interference and non-exact GPS reference points) need to be addressed. As mentioned before, by implementing these features the project will be a complete proof of concept and become available in real world.

Acknowledgements

The authors would like to appreciate the inputs from Mr. Jorge Ramos of Tx DOT and Mr. Steven Venglar of TTI.

Conflict of Interest: none declared.

References

Finley, M. D., Venglar, S. P., Iragavarapu, V., Miles, J. D., Park, E. S., Cooner, S. A., & Ranft, S. E. "Assessment Of The Effectiveness Of Wrong Way Driving Countermeasures And Mitigation Methods," (Rep. No. FHWA/TX-15/0-6769-1). Austin, TX: Texas Department of Transportation, (2014).

Ramos, Jorge. "Wrong Way Driver Project." Powerpoint Presentation. TxDOT, San Antonio. TX, Nov. 2018.

Jen Fifield, "Technology Is Turning Wrong-Way Drivers Around", December 19, 2017. https://www.pewtrusts.org/en/research-and-analysis/blogs/stateline/2017/12/19/technology-is-turning-wrong-way-drivers-around. Retrieved on June 10, 2020.

Highway Special Investigation Report: Wrong Way Driving. Report NTSB/SIR-12/01. National Transportation Safety Board, Washington, DC, December 11, 2012.

Copelan, J., "Prevention of Wrong Way Accidents on Freeways", Report No. FHWA/CA-TE-89-2. California Department of Transportation, Traffic Operations Division, Sacramento, CA, 1989.

Morena, D, and T. Leix, "Where These Drivers Went Wrong In Public Roads", Volume 75, No. 6, May/June 2012. Available at http://www.fhwa.dot.gov/publications/publicroads/12mayjune/05.cfm. Accessed March 1, 2020.

Zhou, H., J. Zhao, and R. Fries. Investigation of Contributing Factors Regarding Wrong Way Driving on Freeways. Report FHWA-ICT-12-010. Illinois Center for Transportation, Urbana, IL, October 2012.

Cooner, S.A., A.S. Cothron, and S.E. Ranft. Countermeasures for Wrong Way Movement on Freeways: Overview of Project Activities and Findings. Report

No. FHWA/TX-04/0-4128-1, Texas Transportation Institute, College Station, TX, January 2004.

Pour-Rouholamin, Mahdi; Zhou, Huaguo; Shaw, Jeffrey; Tobias, Priscilla (2015). Current Practices of Safety Countermeasures for Wrong-Way Driving Crashes (PDF). Transportation Research Board 94th Annual Meeting. Washington, D.C. Retrieved 28 May 2020.

Zhou, Huaguo; Pour-Rouholamin, Mahdi (May 2014). Guidelines for Reducing Wrong-Way Crashes on Freeways. Illinois Center for Transportation and Illinois Department of Transportation. Retrieved 28 May 2020.

ClearWay wrong way driver detection, https://navtechradar.com/explore/wrong-way-driver-detection/, Retrieved 28 May 2020.