Vehicle Collision Avoidance and Detection System Using Internet Of Things

Muhammad Zafran Muhammad Zaly Shah, Mohd Fo’ad Rohani
School of Computing, Faculty of Engineering
Universiti Teknologi Malaysia
81310 Johor Bahru, Johor, Malaysia
muhammad.zafran@outlook.com, foad@utm.my

Abstract—Vehicle accidents is one of the most common cause of death around the world. Accidents happen due to several factors such as driving distraction, poor driving discipline and not being alert while driving. In this paper, the factor of accidents that this paper is trying to address is the driver’s distraction and not being alert to their surroundings. To solve this problem, a system is developed to improve the driver’s awareness. This is done by collecting data from other vehicles such as location, heading and speed and calculate their likelihood of having an accident with the driver’s vehicle. The advantage of using this method is, relatively low cost compared to other driving assistance available. From testing, the results are very encouraging and shows this system could increase the driver’s awareness on the road. (Abstract)

Keywords—collision avoidance; Collision Detection; Vehicle Networks, Internet of Things;

I. INTRODUCTION

Vehicle accidents is one of the major causes of death in Malaysia [2]. The government tries to combat this by law enforcement and installing Automated Enforcement System (AES) to prevent reckless driving behavior and improving the driver’s discipline on the road. However, 90% of accidents are due to driver’s error in judgement, human error and distractions [1]. Clearly the government is taking the wrong approach to address the problem.

The system that this paper proposes, will try to improve driver’s awareness on the road by monitoring surrounding vehicles and alert the driver when there is an accident predicted in the future. This is achieved by using the Internet to share information such as speed, heading and location between vehicles to be used by an algorithm to predict a future accident.

Besides accident prediction, this system will also warn drivers about an accident that already had occurred down the road. This ensures that the driver is aware of the hazard in front of them. This feature is achieved using accelerometers that is attached to the device to read the G forces (Gravity forces) acting on the vehicle. If the reading reaches a certain threshold, the system will broadcast the alert to nearby vehicles.

Based on testing that had been conducting during the development, the proposed system had achieved encouraging results as will be explained later in this paper.

II. CURRENT SOLUTIONS TO DRIVER AWARENESS

For the vehicles to share their information, a vehicular network needs to be designed. Currently, there are two types of vehicle network that are under active research. These two networks are, Vehicle to Vehicle Networks (V2V) and Infrastructure to Vehicle Networks (I2V). This paper will look into the advantages of these networks. There is also a widely used mobile application called Waze that this paper will investigate.
A. Vehicle to Vehicle Networks (V2V)

Fig. 1 shows an illustration of V2V Networks. In V2V Networks, vehicles communicate directly between each other. They are essentially ad-hoc networks for vehicles. And as any other ad-hoc networks, each vehicle is responsible to perform their own routing to transfer data [3].

An advantage of V2V networks is, they are low cost since no extra infrastructure is needed to implement them. However, since each vehicle is independent, it is hard to conduct monitoring features for law enforcers [5].

B. Infrastructure to Vehicle Networks (I2V)

Infrastructure to Vehicle (I2V) on the other hand uses infrastructures such as servers that relays information to drivers within the control area. This method will incur additional costs for installing the infrastructure. However, since there are servers that transmits and receive vehicle information, there could be a central monitoring function that could be used by law enforcers to monitor the traffic [3].

Moreover, using this method the communication between vehicles could be achieved using cellular networks such as 4G. This will allow low latency communication which are essential for this application [5].

C. Waze

Waze is primarily a navigation application that can be installed in the mobile phone. However, the feature that this paper took interest is the feature where drivers can share traffic information such as accidents and roadblock to alert other drivers in the area.

Waze uses cellular or Wi-Fi networks to relay these information to the user. Fig. 3 shows an example of the user interface. From Fig. 3, the application had relayed some information such as hazards, traffic congestion and also police road blocks.

However, one drawback of Waze is it relies on the community to report the incident so there will be delay from the time of the incident occurs to the time it is reported. So, some drivers may not be aware about the hazard in time. The proposed system will be able to report the incident in real-time so that all driver will be able to be alerted of a hazard immediately.

D. Conclusion

The communication architecture that this system chooses to implement is the I2V communication. However, instead of building the infrastructure, this system will use a cloud server which is the Firebase cloud infrastructure to relay vehicles information. Waze is a mobile application that could improve driver’s safety by relaying hazards that is shared by the community. However, since it relies on the community to report the hazard, the hazard will not be reported in real time.
III. SYSTEM DESIGN

Here, this paper will describe the proposed system design which includes the functional requirements, non-functional requirements and hardware requirements.

A. Functional Requirements

![Use Case Diagram]

Fig. 4 shows the use case diagram for the proposed system. The actor of this system are the driver and law enforcers. The Global Positioning System (GPS) is the external system that supplies the input for this system.

Table 1 describes the summary of the functional requirement based on the use case diagram.

<table>
<thead>
<tr>
<th>Actor</th>
<th>Criteria</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Drivers</td>
<td>1. Vehicle owner. 2. Have driving license. 3. Have this system installed in the vehicle.</td>
<td>1. Verify vehicle. 2. Monitor surrounding vehicle. 3. Receive emergency warning. 4. Transmit emergency warning</td>
</tr>
<tr>
<td>Law Enforcer</td>
<td>1. Employed by the government such as police and highway patrols 2. Use this system to monitor traffic</td>
<td>1. Monitor vehicles</td>
</tr>
<tr>
<td>GPS</td>
<td>1. External system that supplies this system the inputs</td>
<td>1. Monitor surrounding vehicles</td>
</tr>
</tbody>
</table>

B. Non-Functional Requirements

Non-functional requirements are the requirements that needs to be performed by the system. Below is the list of the functional requirements of the proposed system:

1. This system need to be able to detect a collision at least 3 seconds before impact. This time is chosen to allow the driver to have sufficient response time to react to a collision.
2. This system needs to be extremely reliable and could not afford any downtime. To achieve this, we opt to use cloud servers manage by Google which can guarantee 100 percent uptime.
3. This system need to be accurate. All sensors need to be as accurate as possible to ensure an accurate prediction of collision and to avoid false positives and false negatives.
4. This system need to protect the privacy of drivers. Since this system deals with location which might be sensitive for drivers, only authorized personnel or law enforcers can view the location of vehicles.

C. Hardware Requirements

![Hardware Setup]

Hardware requirements are the hardware that is required to develop this system. Here, this paper will describe the hardware that is used to develop this system.

1. Raspberry Pi 3 Model B. This hardware is the central processing unit where all the data are processed to be pushed to the driver’s smartphone. It is also connected to the Internet to retrieve and transmit data to the server.
2. MTK3339 GPS module. This is the hardware that is used to get the information
about the driver’s vehicle such as the location (latitude and longitude), speed and heading.

3. ADXL345 accelerometer. This hardware is to measure the G forces acting on the driver’s vehicle. This is used for the collision detection that will be described later in this paper.

Fig. 5 shows the connection, or the setup of the hardware described previously. The hardware is connected from the Raspberry Pi to the sensors using jumper cables via the breadboard. The smartphone is connected to the setup via local network connection since the Raspberry Pi can act as a server. So no physical connection is required between the smartphone and the Raspberry Pi.

IV. COLLISION PREDICTION ALGORITHM

The collision detection algorithm is the key part in this system. The method that is chosen to implement this algorithm is called the Time to Collision (TTC) algorithm [1]. This algorithm is divided into two parts. The first part is finding the intersection point of both vehicles (point where both vehicles collide), the second part is determining the time for both vehicles to reach the intersection point and finally the determining the likelihood of the accident.

A. Finding the Intersection Point of Both Vehicles.

Fig. 6 shows the illustration of the intersection point of both vehicles. From Fig. 6 the intersection point could be calculated using linear algebra methods.

\[
\begin{bmatrix}
  x_+ \\
  y_+
\end{bmatrix}
= \begin{bmatrix}
  -mx_1 & y_1 \\
  -mx_2 & y_2
\end{bmatrix}^{-1}
\begin{bmatrix}
  c_1 \\
  c_2
\end{bmatrix}
\]  

(1)

Equation (1) shows the linear equation that is used to compute the intersection point. However, there are constants that need to be calculated before (1) can be used. Which are the gradients and the y – intercept. The gradients can be computed directly from the heading of the vehicles while the y – intercept can be computed by substituting the values into a line equation.

\[m = \tan(\theta - 90)\]  

(2)

\[c = mx - y\]  

(3)

Equation (2) is used when the heading of the vehicle is less than 90 degrees while (3) is used when the heading is larger than 90 degrees. After the gradients, to obtain the y – intercept simply use (4).

B. Calculating the Time to Reach Intersection Point

The time to reach the intersection point for each vehicle can be calculated using (5)

\[t = \frac{d}{v}\]  

(5)

From (4) the distance can be obtained by calculating the distance from the vehicle to the intersection point that is obtained from (1). The velocity is obtained from the GPS receiver.

C. Calculating the Likelihood of the Accident

The likelihood of the accident can be calculated by taking the difference in time for each vehicle to reach the intersection point.

\[\Delta = |t_1 - t_2|\]  

(6)

Equation (6) shows the calculation for the difference in time to reach the intersection for both vehicle. From (6) both vehicles are more likely to have an accident when the difference in time for both vehicles to reach the intersection is small. To calculate the likelihood for an accident, a function that outputs a large number when its input is small is chosen.

\[f(x) = \frac{1}{x}\]  

(7)

\[\lim_{x \to 0} f(x) = \infty\]  

(8)

The function that is chosen is shown in (7) while (8) is the property of the function which means that the function approaches infinity when the input is close to zero.
A tolerance value is chosen to trigger the alert for the driver. Based on testing the most suitable tolerance is 0.5. So, if (6) outputs a value larger than 0.5, the alert will be triggered.

Fig. 7 shows the flowchart of the entire collision detection algorithm. As shown in Fig. 4, the algorithm does not require any user input since the input is directly obtained from the GPS receiver.

**TABLE 2: TESTING OF THE ACCIDENT PREDICTION ALGORITHM BASED ON SEVERAL TEST CONDITIONS**

<table>
<thead>
<tr>
<th>Test Case</th>
<th>Vehicle A</th>
<th>Vehicle B</th>
<th>Collision as per predicted likelihood?</th>
</tr>
</thead>
<tbody>
<tr>
<td>TC1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TC2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TC3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TC4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TC5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TC6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TC7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TC8</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2 shows the test for the algorithm. From Table 2 it is found that this algorithm is able to predict a collision.
V. Collision Detection Algorithm

Collision detection is the feature in this system when the system detects that a vehicle had been involved in an accident. Since this system is to increase driver’s alertness, this system need to be able to inform the driver that there is an accident down the road.

This feature also uses the threshold method to trigger the warning to other drivers. Normally, heavy accidents have an impact of about 6G [4]. So, when the system detects that the accelerometer registers a reading larger than 6G, the warning will be sent to the server to be relayed to other drivers. Once the system detects that an accident the system will transmit the alert to the server and relayed to vehicles nearby as will be explained later.

![Flowchart of Collision Detection](image)

Vehicles nearby, assuming they all use this system, will query any new hazards in a 1000m radius. If there are any accident within the area, the server will relay the information to the driver.

VI. System Architecture and Interface Design

A. System Architecture

Fig. 9 shows the system architecture for the proposed system. The vehicles have the Raspberry Pi and the GPS equipped to send the data to the cloud Firebase server. All other vehicles will query the locations of surrounding vehicle and calculate their likelihood of having a collision.

B. Interface Design

The interface design requirement for this system is, the interface needs to be simple in order to avoid any unnecessary distraction for the driver.

C. Interface Design for Collision Prediction

Fig. 10 shows the interface design for the collision detection feature. The arrows around the image of the car indicates the direction of the target vehicle that is predicted to have a collision with the driver’s vehicle. In the event of a predicted collision, the arrows will turn red to grab the driver’s attention.

The minimalistic design of the interface is also important so that the driver can process the information that is displayed quickly and intuitively. Text messages that requires the driver to read is also omitted from the design which is also to avoid unnecessary distraction.
D. Interface for Collision Detection

Fig. 11 shows the interface for the collision detection feature. The interface need to be able to display the information to the driver in a manner that the driver could process the information at a glance while driving.

The interface uses a list style that displays all the hazards vertically. In each list, it shows the distance from the vehicle to the hazard.

VII. TESTING

After the development phase of the system had been carried out, the testing of the system will be conducted to study the usability of the system. In this paper, the testing method that will be used is the black box and white box testing.

A. Black Box Testing

Black box testing is carried out by putting input to the system and recording its output against the expected output while ignoring the structure of the program.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Input</th>
<th>Expected Result</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Log In</td>
<td>Correct email and password</td>
<td>User will be redirected to home page</td>
<td>Success</td>
</tr>
<tr>
<td></td>
<td>Wrong email and password</td>
<td>Log In fail</td>
<td>Success</td>
</tr>
<tr>
<td></td>
<td>Either email or password is wrong</td>
<td>Log In fail</td>
<td>Success</td>
</tr>
<tr>
<td>Add Vehicle</td>
<td>Vehicle license plate, model and key.</td>
<td>Vehicle added to database</td>
<td>Success</td>
</tr>
<tr>
<td></td>
<td>Either vehicle license plate, model and key is not entered</td>
<td>Vehicle not added to database</td>
<td>Success</td>
</tr>
<tr>
<td>Log Out</td>
<td>Log out button</td>
<td>Redirected to log in page</td>
<td>Success</td>
</tr>
</tbody>
</table>

Table 3 shows the results that have been obtained after the black box testing had been carried out. The conclusion that can be made for the black box testing is that the system performed as expected tallying with the expected results.

B. White Box Testing

White box testing is carried out the same way with black box testing but also considering the path of execution of the program. This testing allows to detect bugs that may be hidden or undetected during the development or testing stage.

Table 4 shows the results that is obtained after the white box testing had been carried out. The conclusion that can be made from the results is, the system executed according to plan and so far no bugs that may jeopardize the reliability of this system is detected.

Table 4: White Box Testing Result

<table>
<thead>
<tr>
<th>Code</th>
<th>Expected Result</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>if (task.isSuccessful())</code></td>
<td>True</td>
<td>True</td>
</tr>
<tr>
<td><code>msg = task.getCurrentTask();</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>Intent i = MainActivity.newIntent(getActivity());</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>startActivity(i);</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>getActivity().finish();</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>} else {</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>Toast.makeText(getActivity(), &quot;Authentication Failure&quot;, Toast.LENGTH_LONG).show();</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>vehicle_model = vehicle_model.objects.get(id);</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>user = User.objects.get(id);</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>v = vehicle_model where vehicle_model.user = user;</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>v.license_plate = license_plate,</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>v.key = key;</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>v.save();</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>task.execute();</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>return true;</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Success</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Success</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
I SBN No. : 978-967-2171-30-0

A. Discussion

The problem that this system was trying to solve is the driving safety of the road that is caused by driver distraction and poor judgement.

To address this issue, this paper proposed a system called the Vehicle Collision Detection System using Internet of Things. This system uses the Time to Collision Algorithm (TTC) to predict collision. The TTC algorithm calculates the difference in time to reach the intersection point of both vehicles and triggers an alarm if it is larger than a set tolerance value.

To detect accidents this system uses an accelerometer that triggers an alarm if the value of the accelerometer is larger than 6G.

From testing that had been conducted to test the effectiveness of the algorithm to predict the collision, it had achieved promising results. The testing method is in simulated environment by testing different values and combination of speed and heading of the driver’s vehicle and the approaching vehicle. The algorithm is able to minimize the false positive which raises the alarm when there is no collision in the future by choosing the tolerance value of 0.5.

B. Future Improvement of This System

The suggestion for improvement of this system is to suggest how this system could be improved in order to make this system more accurate and reliable as well as to make this system more user friendly for drivers to use to ensure their safety.

1. Use deep learning method that could learn accident pattern so that this program can adapt to various accident patterns.
2. Use Heads-Up Display (HUD) to prevent driver distraction.

REFERENCES


VIII. CONCLUSION

A. Discussion

The problem that this system was trying to solve is the driving safety of the road that is caused by driver distraction and poor judgement.

To address this issue, this paper proposed a system called the Vehicle Collision Detection System using Internet of Things. This system uses the Time to Collision Algorithm (TTC) to predict collision. The TTC algorithm calculates the difference in time to reach the intersection point of both vehicles and triggers an alarm if it is larger than a set tolerance value. To detect accidents this system uses an accelerometer that triggers an alarm if the value of the accelerometer is larger than 6G.