



UNIVERSITY OF  
CENTRAL FLORIDA

---

## Final Project Report

Name: Zaheen E Muktadi Syed

ID: 5320369

Course: CGN 5340 Internet of Things: Application in Smart Cities

**Project Title: Real time ADAS for Crash and Pedestrian Detection Using Internet of Vehicle.**

## Overview

- Detection and warning system for in vehicle pedestrian and/or crash detection system
- Computer vision algorithm to detect pedestrian and crash
- Offline edge computing using raspberry pi and webcam (Dashcam)
- Wi-Fi based ad hoc network to transmit and receive warning messages

## Table of Contents:

1. Background and Motivation .....	2
2. Literature review.....	3
3. Goals and Objective .....	3
4. Technical Requirements.....	4
5. Experiment Setup.....	4
6. Selected component lists .....	5
7. Final Hardware Setup.....	6
8. Software setup .....	6
9. Results .....	7
10. Challenges and discussions .....	7
11. Future Work .....	7
12. References .....	8

## 1. Background and Motivation

Everyone is a pedestrian at some point of time and specially those who are living in urban setting are supposedly more exposed to traffic than others. Thus, ensuring pedestrian safety and awareness has become one of the most important topics for large traffic organization like National Highway Traffic Safety Administration (NHTSA). In 2020 more than 6000 pedestrians were killed and over 55000 were injured all over USA, thus prompting official to launch pedestrian safety awareness programs [1]. CDC also reported one in six people who died in car crash were pedestrians and over 104000 emergency visits of pedestrian were treated all over USA in the year 2020 [2]–[4]. A recent paper published in Future Generating computing power have discussed various scope of IoT to help tackle the pedestrian crash risk problem.

Automated Driver Assistance System (ADAS) technologies, widely used in modern cars not only helps to keep you and your passengers safe, but also protect surrounding drivers and pedestrians. Some driver assistance technologies are designed as a warning system when at risk of an impending crash, while others are designed to avoid one. [5]However, most of the cars today are not equipped with such sensing

capability. Also added to this problem, sensors have limitation. For example, during a hard brake event it is difficult for following vehicle to anticipate what lies in-front of the leading vehicle. Thus, another popular research area is to share information between vehicles in real time [6]. This not only enhances the sensing capabilities of following or ego vehicles but also can be gateway for older vehicles to enjoy the benefit of ADAS.

In this project we plan to develop near real-time automated pedestrian and crash detection system based on-board video system such as Dashcam and generate a real time warning for surrounding vehicles. The detection system will be stand alone and can be used in any vehicles, and the warning generated from the detection system will be published using two systems: an ad hoc network and Cellular Communication.

## **2. Literature review**

Dash Cam also referred as dashboard cameras, vehicle cameras or on-board cameras in different literatures, forms a monitoring system that can be used for surveillance, insurance claim and various vehicle safety purpose and it is a very rapidly growing industry [7]. The global dashboard camera market size was estimated to be around 3.38 billion in 2021 and it is expected to grow at compound annual growth rate of 9.5% from 2020 to 2030. [8]

As the number of users installing cameras in their vehicles increases, this have led automotive developers to include cameras as one of the fundamental components of a vehicle's equipment. This entails not only capturing the image while driving but also automatically assessing it to reduce risk to the car's occupants. The research by Russo et al. (2015) [9] proposed an algorithm to track and recognize vehicle in real time. Tummala G.K. et al. 2019 [10] proposed an automatic calibration system for dashcam which can be further be used in developing application like lane change detection, safe driving distance estimation etc. Yao et al. 2019 [11] proposed an unsupervised method for first-person (dashboard-mounted camera) video traffic accident detection. The feasibility of developing real time safety application with the help of edge computing and cloud technology is becoming more probable.

A few dash companies' companies have already started integrating AI to generate and record crash event. For example, "Owlcam" utilizes artificial intelligence and high-quality dashCam to generate crash or theft alert. Nexar and Hyperion, two popular Dashcam companies, is selling dashcam with crash detection and gesture technology to record important events. Cobra dashcam have integrated lane changing and forward collision warning system. However, most systems are dependent on internet connection and developed with a cloud-based architecture for computation.

In our project we plan to deploy the system offline with limited hardware and create an ad hoc network such that the warning generated can spread in real time to surrounding vehicles and alerting the respective drivers without the need for a cellular network.

## **3. Goals and Objective**

The Goal of this project is to connect a real-time ADAS application with Internet of Vehicles. For this project to work we must solve to core objectives. The objectives are:

1. Develop a ADAS application for pedestrian and crash detection

2. Develop Ad hoc network for propagation of warning between surrounding vehicles

#### 4. Technical Requirements

To satisfy the objectives of this project we should consider the technical requirements of the project for each objective both in terms of hardware and software. In the first object, we should select a medium size micro processor that is readily available, have enough computation to process image and detect object. Therefore, we need to select a microprocess for this function. In terms of software, we must develop light weight computationally inexpensive machine learning models.

For the second objective to set a broad casting system using an ad-hoc network we have to select the medium of communication, a light wight and readily deployable server and client management system for peripheral devices which may include a nearby vehicle.

#### 5. Experiment Setup

Upon considering the technical requirements and objectives, we have designed the following setup for developing the prototype.

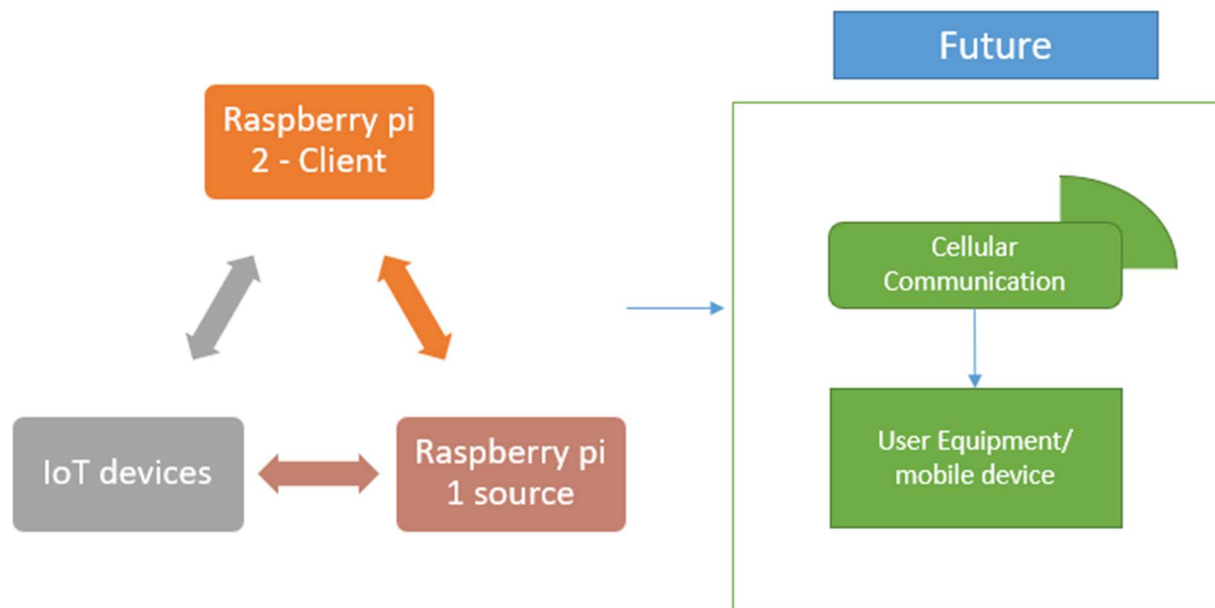





Figure 1: Hardware setup

Figure 1 shows a low-level hardware architecture. The vision of this project is too large for the scope of the project; thus, we will only set the local part as shown in left side of the picture. The right side shows the integration of project with user equipment like mobile phone and cellular communication to setup large central database where we upload a high-level data to the mobile devices onboard and store

important information in a cloud server. For the prototype we plan to use raspberry pi and a webcam to build the detection system[12]. A low-level light weight message will be generated from the detection system and broad casted to all the devices in the network. The message will be propagated via the ad hoc network built using Wi-Fi communication. One of the reasons of choosing Wi-Fi is because of its popularity and cheap hardware availability. To manage the message and communicate peripheral devices a lightweight mosquito server is established using MQTT protocol [13] . The MQTT (IoT) protocol is a common communications protocol for the Internet of Things. It is intended to connect remote devices with a tiny code footprint and low network bandwidth by acting as an incredibly lightweight publish/subscribe messaging transport system. Once a device detects something it can readily generate the message to a particular channel and broadcast to all other listening devices.

## 6. Selected component lists

Item	Component Name	Description	Image
1	ESP32 ESP-32S Development Board	<p>HiLetgo ESP-WROOM-32 ESP32 ESP-32S Development Board 2.4GHz Dual-Mode WiFi + Bluetooth Dual Cores Microcontroller Processor Integrated with Antenna RF AMP Filter AP STA for Arduino ID</p> <ul style="list-style-type: none"> <li>- 2.4GHz Dual Mode WiFi + Bluetooth Development Board</li> <li>- Ultra-Low power consumption, works perfectly with the Arduino IDE</li> </ul> <p>ESP32 is a safe, reliable, and scalable to a variety of applications</p>	
5	Raspberry pi	Main controller board for network adapter .	
6	Webcam	For video recording	

## 7. Final Hardware Setup

All the components were procured and tested for their functional roles. Overcoming a lot of challenges, for our prototype we set two setups. The setups are shown in the figure below.

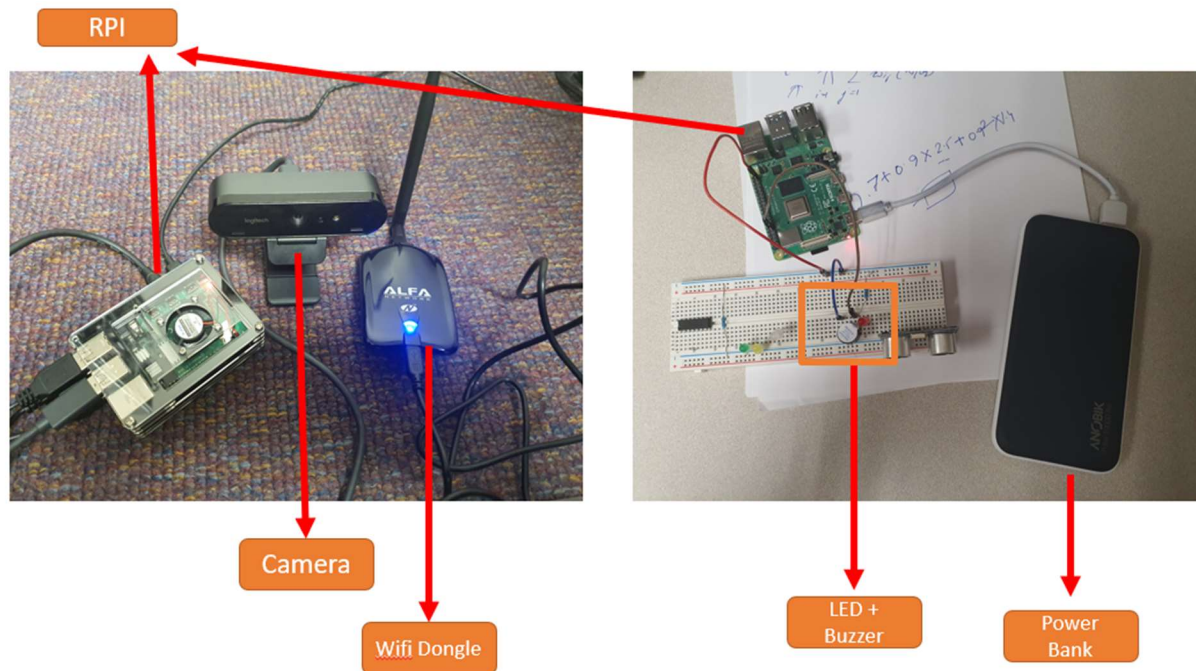


Figure 2: Hardware setup

The left side setup is set to act as object detection device that can be placed in a vehicle or critical locations. The Raspberry pi (RPI) is connected to a Wi-Fi dongle and camera. The Wi-Fi dongle is used for better connectivity and can be set as an access point for surrounding devices to connect. The camera inputs real-time video into the RPI for pedestrian or crash detection. In the left side of the figure, another raspberry pi is set to listen to the broadcasted signal and operate a simple warning system designed using a buzzer and led light connected to the camera.

## 8. Software setup

For the whole system to work, first we set the raspberry pi using a Raspbian operating system and installed micro python packages and other important packages for image processing, server connection and control outer circuit like the buzzer. In this project we mainly used python to write the codes.

For object classification we used tensorflow lite tool. It is the framework for compressing and deploying a trained TensorFlow model to an embedded or mobile application is provided [14]. The application can then use the pre-trained model's ability to draw conclusions for itself by interacting with the TensorFlow Lite Interpreter. We first researched to find some of the best model for object detection that can be

implemented in devices with low computational power. We found that the “efficientdet lite 0” model pretrained using 2017 coco dataset that can detect over 80 object classes works best in terms of frame rate per second and real time detection. We coded a scrip to implement the tflite model to detect person in real time and generate a message.

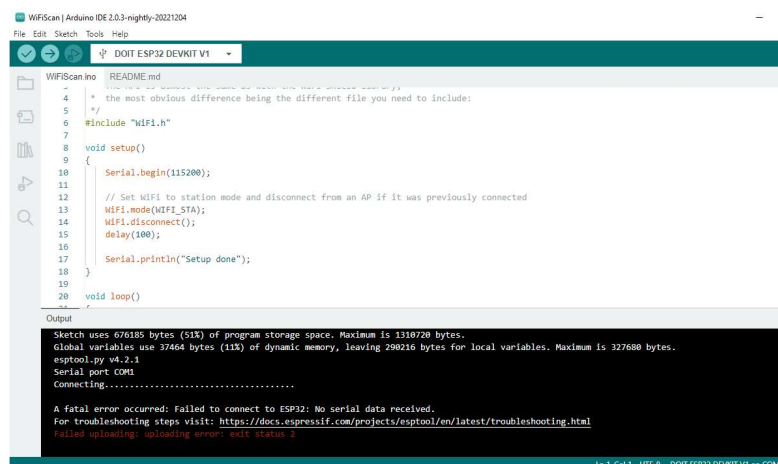
After the detection, we set up the MQTT server and wrote two python script. One script continuously listens to a particular channel for any broad cast message. The Beauty of the ad-hoc network and server is that if it disconnects for any reason, it will automatically reestablish connection upon network availability. Finally, we wrote a small script to control the buzzer and led light whenever a broadcast message is received.

## 9. Results

For the working prototype we successfully deployed a ML model to detect pedestrian with a frame rate of more than 5 per second. Our MQTT server has been actively running without failure for more than 10 hours and three devices continuously broadcasted with each other. The video of the prototype is uploaded in the presentation segment.

## 10. Challenges and discussions

Some challenges of the project is to accommodate sophisticated ML models and achieve low latency. Throughout the project a major issue was to hardware crash and server overload. Often the camera detection system had to be restarted for proper functioning. The biggest challenge was hardware failure. We were unable to deploy the esp32 module for message sensing. The ESP32 crashed in the last minute, and we could not upload any code on the module as shown in figure below. Thus, reliability is major concern for cheap iot sensors.



```
WIFIScan.ino  README.md
4  * the most obvious difference being the different file you need to include:
5  */
6  #include "WiFi.h"
7
8  void setup()
9  {
10     Serial.begin(115200);
11
12     // Set WiFi to station mode and disconnect from an AP if it was previously connected
13     WiFi.mode(WIFI_STA);
14     WiFi.disconnect();
15     delay(100);
16
17     Serial.println("Setup done");
18 }
19
20 void loop()
21 {
22
23 }
```

Sketch uses 676185 bytes (518) of program storage space. Maximum is 1310720 bytes.  
Global variables use 37464 bytes (113) of dynamic memory, leaving 290216 bytes for local variables. Maximum is 327680 bytes.  
esptool.py v4.2.1  
Serial port COM1  
Connecting.....  
A fatal error occurred: Failed to connect to ESP32: No serial data received.  
For troubleshooting steps visit: <https://docs.espressif.com/projects/esptool/en/latest/troubleshooting.html>  
Failed uploading: uploading error: exit status 2

Figure 3: ESP32 failure

## 11. Future Work

In future we will train the module for crash detection and prediction such that we can identify a crash risk in real time. This will be game changing objective for road safety devices. Also there are other minor issues

we need to take care of like stabilizing the MQTT server, creating access point etc. However, due to limitation of time we could not test the system in a real vehicle. In future we need to design experiment to check the reliability and compatibility of the proposed system in real environments.

## 12. References

- [1] "Pedestrian Safety: Prevent Pedestrian Crashes | NHTSA." <https://www.nhtsa.gov/road-safety/pedestrian-safety> (accessed Dec. 05, 2022).
- [2] B. C. Tefft, "Impact speed and a pedestrian's risk of severe injury or death," *Accid Anal Prev*, vol. 50, pp. 871–878, Jan. 2013, doi: 10.1016/j.aap.2012.07.022.
- [3] E. Rosén and U. Sander, "Pedestrian fatality risk as a function of car impact speed," *Accid Anal Prev*, vol. 41, no. 3, pp. 536–542, May 2009, doi: 10.1016/j.aap.2009.02.002.
- [4] "Pedestrian Safety | Transportation Safety | Injury Center | CDC." [https://www.cdc.gov/transportationsafety/pedestrian\\_safety/index.html](https://www.cdc.gov/transportationsafety/pedestrian_safety/index.html) (accessed Dec. 05, 2022).
- [5] "Driver Assistance Technologies | NHTSA." <https://www.nhtsa.gov/equipment/driver-assistance-technologies> (accessed Oct. 15, 2022).
- [6] H. Zhou, W. Xu, J. Chen, and W. Wang, "Evolutionary V2X Technologies Toward the Internet of Vehicles: Challenges and Opportunities," *Proceedings of the IEEE*, vol. 108, no. 2, pp. 308–323, Feb. 2020, doi: 10.1109/JPROC.2019.2961937.
- [7] S. Kadu, N. Cheggoju, and V. R. Satpute, "Noise-resilient compressed domain video watermarking system for in-car camera security," *Multimedia Systems 2017 24:5*, vol. 24, no. 5, pp. 583–595, Dec. 2017, doi: 10.1007/S00530-017-0584-3.
- [8] "Dashboard Camera Market Size & Share Analysis Report, 2030." <https://www.grandviewresearch.com/industry-analysis/dashboard-camera-market> (accessed Oct. 29, 2022).
- [9] G. Russo, E. Baccaglini, L. Boulard, D. Brevi, and R. Scopigno, "Video processing for V2V communications: A case study with traffic lights and plate recognition; Video processing for V2V communications: A case study with traffic lights and plate recognition," *2015 IEEE 1st International Forum on Research and Technologies for Society and Industry Leveraging a better tomorrow (RTSI)*, 2015, doi: 10.1109/RTSI.2015.7325064.
- [10] G. Krishna Tummala, T. Das, P. Sinha, and R. Ramnath, "SmartDashCam: Automatic Live Calibration for DashCams," 2019, doi: 10.1145/3302506.3310397.
- [11] Y. Yao, M. Xu, Y. Wang, D. J. Crandall, and E. M. Atkins, "Unsupervised Traffic Accident Detection in First-Person Videos," *IEEE International Conference on Intelligent Robots and Systems*, pp. 273–280, Nov. 2019, doi: 10.1109/IROS40897.2019.8967556.
- [12] A. P. Kulkarni and V. P. Baligar, "Real Time Vehicle Detection, Tracking and Counting Using Raspberry-Pi," *2nd International Conference on Innovative Mechanisms for Industry Applications*,



*ICIMIA 2020 - Conference Proceedings*, pp. 603–607, Mar. 2020, doi: 10.1109/ICIMIA48430.2020.9074944.

- [13] R. A. Light, “Mosquitto: server and client implementation of the MQTT protocol,” *J Open Source Softw*, vol. 2, no. 13, p. 265, May 2017, doi: 10.21105/JOSS.00265.
- [14] R. David *et al.*, “TensorFlow Lite Micro: Embedded Machine Learning for TinyML Systems,” *Proceedings of Machine Learning and Systems*, vol. 3, pp. 800–811, Mar. 2021.