CS-308-2016 Final Report

SpecPro

Speed and Congestion Detection using Proximity Sensors

Team 6

Prateesh Goyal 120050013

Pratyaksh Sharama 120050019

Ramprakash Krishnan 120050083

Viplov Jain 120050084

Table of Contents

1. Introduction

The number of vehicles in India are ever increasing and so are the accidents associated with it. Over speeding is one of the major reasons for this. In 2014, 1.4 Lakh people died because of over speeding accidents.

A car crash on a highway.

Currently the traffic police deploys speed detection vans often carrying 2 traffic policemen to detect over speeding. This system is very expensive and often can't be deployed on highways because they are far off. Over speeding happens because there is no system to monitor speed of vehicles and do a challan automatically and without any human intervention.

The idea is to build such an autonomous system for detection speed of vehicles and challan in case of over speeding vehicles.

Radar speed gun used by law enforcement.

2. Problem Statement

The aim of the project is to build an automatic system for detection of overspeeding vehicle, their vehicle number and using it for automatic challan.

Use IR proximity sensors embedded in road to detect speed of vehicles and if the vehicle was speeding then take an image of the speeding vehicle and detect the number on the number plate using image processing techniques. Once the number is obtained, the number along with relevant details such as speed, time , location etc is to be emailed to the concerned party. There should be a GUI to display all the challan recorded by the system along with relevant details and picture of speeding vehicle.

For the purpose of our project we will use remote controlled car as vehicles with a number plate attached to it and will showcase the working of the system.

The data collected about traffic can also be sold to third party vendors like google maps.

3. Requirements

3.1 Functional Requirements

1. The system should be able to detect a speed of the vehicle to an acceptable accuracy (5-10% error tolerance).
2. The system should be able to capture an image of the speeding vehicle.
3. The system should be able to read the number plate of the speeding vehicle from the captured image.
4. The system should be able to notify the vehicle owner of the speeding event.
5. The system should update a GUI with the information of the speeding event.

3.2 Non-Functional Requirements

1. The system should be minimal in footprint and should allow ease of installation and use.
2. The system should be responsive and should be able to work at highway throughput.
3. The system should not fail in adverse weather conditions.
4. The system must provide ease of detection of failures if any.

3.3 Harwdare Requirements

1. TIVA Board
2. Connecting jumpers
3. Laptop with a front camera
4. Remote controlled car (demo)
5. IR proximity sensors

3.4 Software Requirements

1. OpenCV - To be used for Image Processing purposes
2. TIVA CCS - To write and burn the TIVA code on the microcontroller
3. MJPG Streamer - To take images from laptop's camera
4. Tesseract - OCR for text recognitions

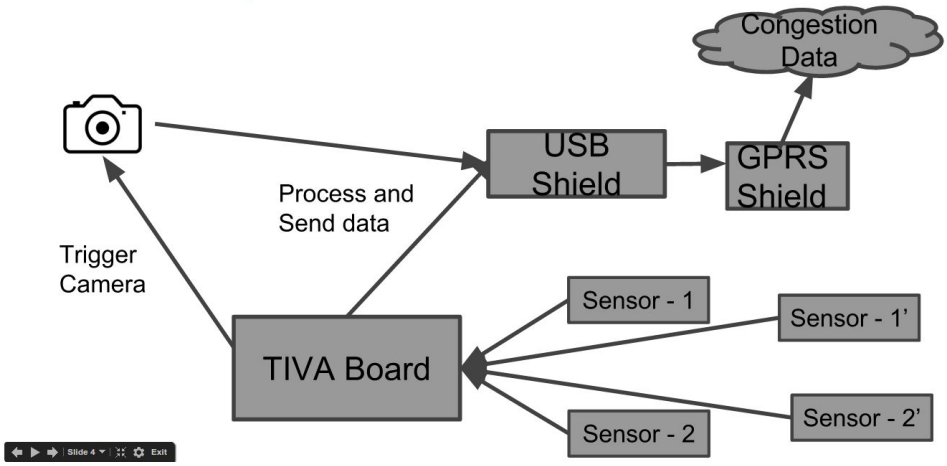
Python modules required:

1. Image
2. pytesseract
3. serial
4. email

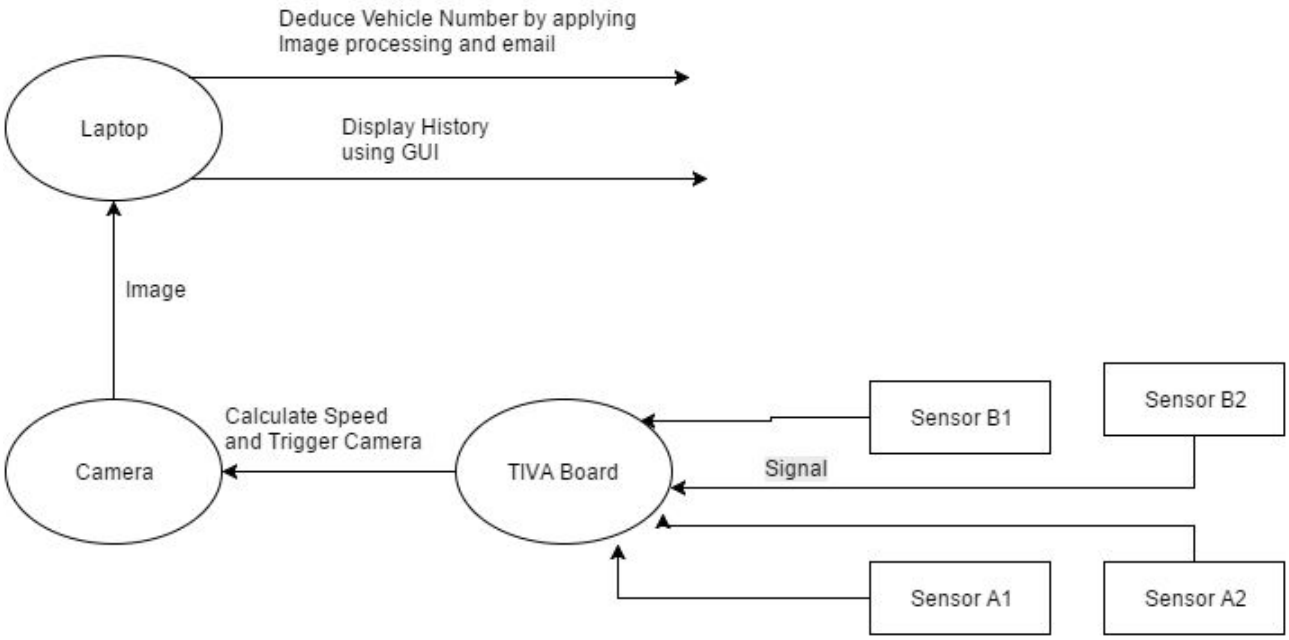
All of the above modules can be installed using `sudo pip2 install --upgrade <module-name>`.

4. System Design

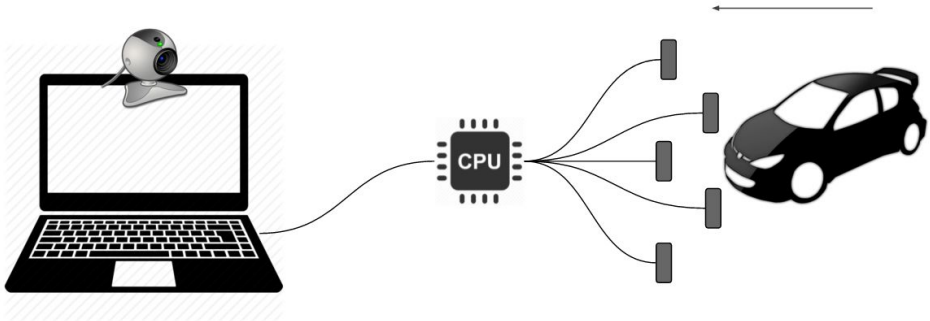
4.1 Block Diagram of SpecPro



4.2 Block diagram for demo

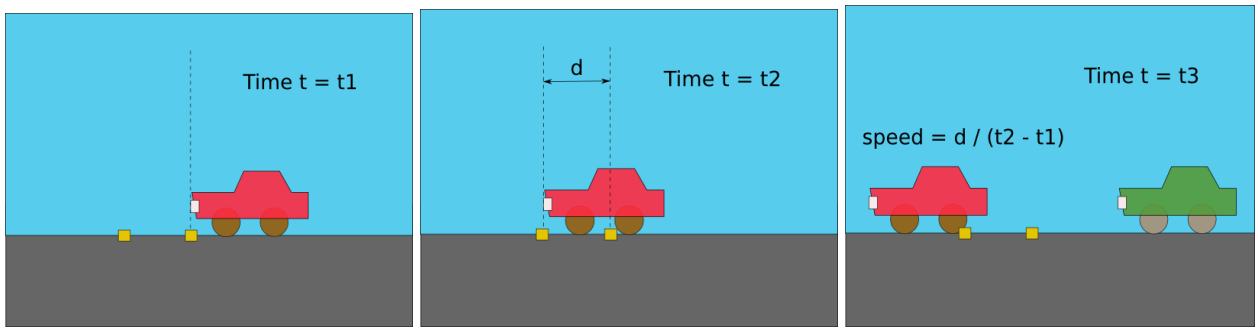


4.3 Demo Schematic



4.4 System Design Description

Proximity Sensors- The proximity sensors are to be embedded in the road and to be placed in pairs of 2, with a fixed distance d between them. Whenever a car passes above the proximity sensor the readings of the sensor change and it can detect whether a car is above it. So a proximity sensor can detect if a car just came above it. Say this happened at time t1. The sensor will convey this information to microcontroller. An since there is one more sensor in front of that car the car will pass that sensor say at time t2. The second sensor also conveys this information to microcontroller.



Proximity Sensor in action

Small Yellow Boxes are Proximity Sensors

Microcontroller/TIVA Board- The microcontroller knows time t1 and time t2 and it also knows the distance between the two proximity sensors d. Hence the speed of the vehicle is calculated as d/(t2-t1). If this is above limit it passes the information to camera/laptop to take an image.

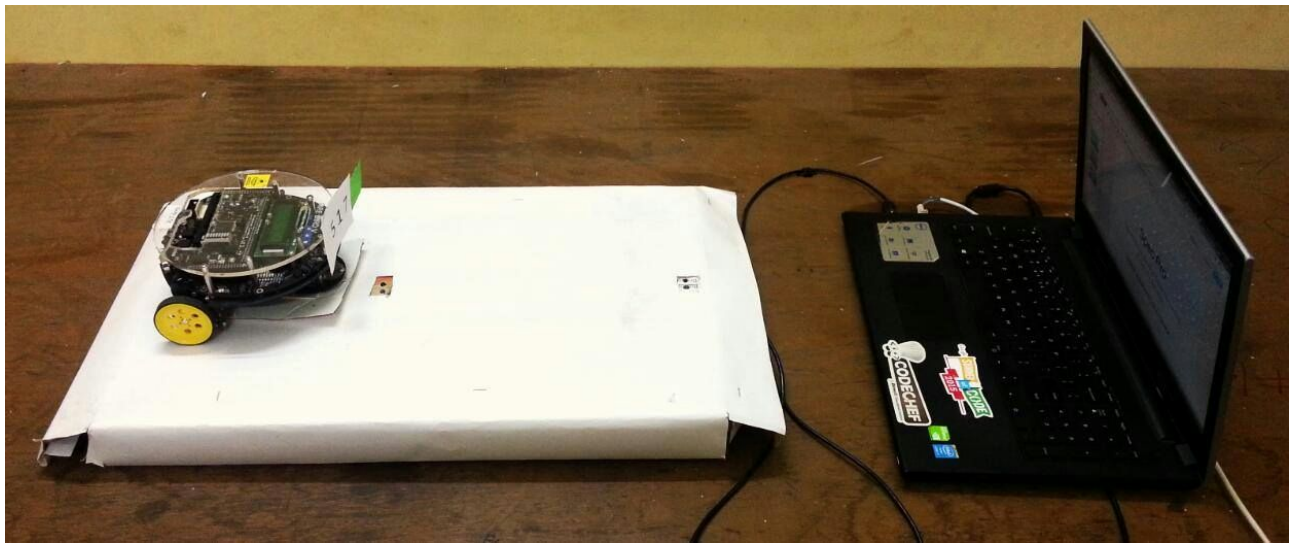
Laptop/Camera- The camera is taking images continuously and storing images corresponding to last 10 seconds. Once it receives the signal the laptop takes the appropriate image from these set of images. Once we have the image, image processing techniques are applied to first extract the number plate and then ocr is used to get the number. The number along with relevant information is emailed to the concerned party. The history can be seen in GUI.

5. Working of the System and Test results

Test car (firebird) with the number plate showing. Note that the number plate has a distinguishing green patch at its right end.

A firebird bot is used as the test car. A number plate is fixed on its front end (camera-facing). The number plate is of fixed size with a distinguishing green square at its right end.

When the car (firebird) crosses the specpro IR sensors installed on the track, the TIVA board computes the time elapsed between event-1: car crosses triggers the first IR sensor, and event-2: car crosses triggers the second IR sensor. This computed time (in units of microprocessor clock cycles) is then sent to a PC which is running the main control logic. The main control logic consists of a while loop which waits for UART signals from the TIVA board, and on receiving a signal (the calculated clock cycles), it captures the image of the moving car.



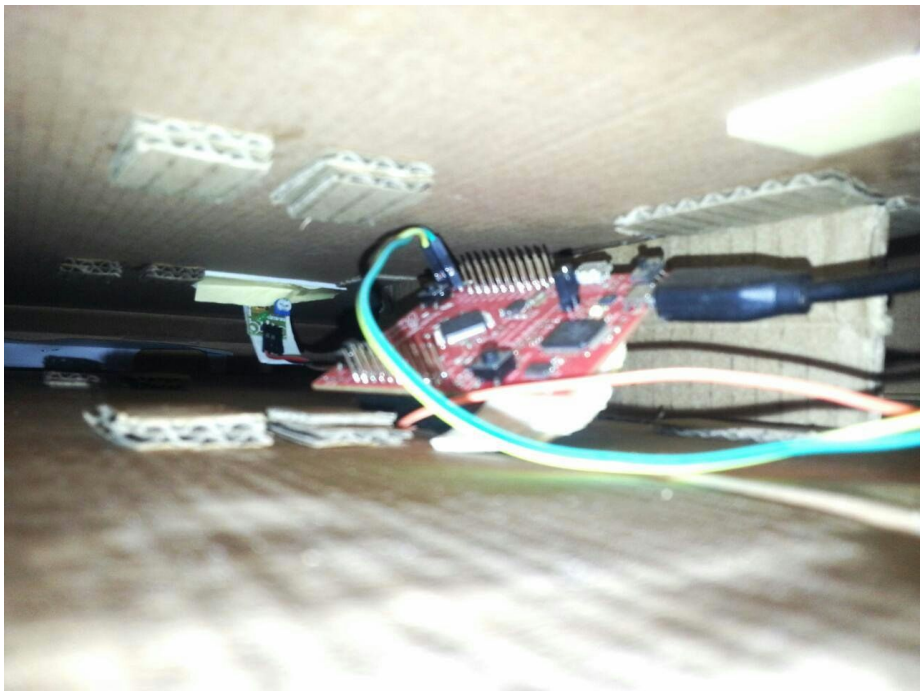
The overall specpro setup. The laptop at the right is equipped with a low resolution webcam that continuously captures images at a rate of 10 image/second. The white plank on the left has two IR proximity sensors embedded which are used to detect the presence of a vehicle.

Multiple image processing techniques are applied on the captured image:

1. The image is cropped to exctract the number plate of the car: the green distinguishing patch is used to identify the location of the number plate in the image. The python program cut.py is reponsible for this process.
2. The cropped image is binarized (converted into black or white pixels), using Otsu's algorithm for thresholding. Again, the same python code cut.py contains the code for this step.
3. The binarized image is sharpened using Gaussian unsharp masking (using ImageMagic tool on linux).
4. The sharpened, binarized, cropped image is processed by the tesseract OCR trained to detect single lines of digits. Finally, the output produced by Tesseract is the number plate of the car.

The multiple images produced in the above steps, and information about the car: number plate, timstamp of speeding, speed detected are saved on disk.

A java GUI displays the saved details of overspeeding cars. Further, an email is sent to the offending vehicle's owner mentioning the details like date/time, number plate, and speed of the event.



Inside specpro

Test Results

Following is an instance of testing the system with a vehicle with a number plate “012”:

Image captured by the webcam

Image after cropping out the number plate

Image after denoising and thresholding

In this case, the number plate was correctly identified as “012”

Test results across different number plates

For each of the configuration below table indicates number of correct number recognition out of 5 tries:

Number plate -->	0 1 2	6 7 7	5 1 7
Low Speed	4	4	5
High Speed	3	4	4

Total accuracy in number detection = (24/30)*100 % = 80%

6. Discussion of System

a) What all components of your project worked as per plan?

1. Proximity Sensors-

The sensors were correctly able to identify a vehicle above them and signal the microcontroller appropriately.

2. Microcontroller-

We were able to identify the speed correctly without much delay.

3. Camera and Processing-

We were able to correctly identify the image to be used, extract the number plate region, binarize it and use Tesseract to obtain correct vehicle number..

b) What we added more than discussed in SRS?

1. GUI

We added a GUI to view the history/records of all the speeding vehicles. This can potentially be used by traffic police personals in future. It makes looking at the history very easy.

2. Email

Every violation is emailed to concerned party with relevant details. This is equivalent to issuing a challan.

3. Car for testing

We used firebird as our car. We ran it at different speeds with different number plates to verify our prototype.

c) Changes made in plan from SRS:

Nothing has been changed from SRS. Only more features have been added to make the prototype more attractive.

7. Future Work

· The image processing part of the project can be improved so that it doesn't require, a green identifier to detect the number plate

· At the moment Tesseract(ocr) works on standard text only, and the tesseract can be specifically trained to recognize number plate

· The accuracy of the number detection algorithm can be further increased by applying advanced image processing techniques.

· Debluring of input image can be further improved

8. Conclusions

In the prototype we build we were correctly able to identify, the speed and number of the speeding vehicle using just the web camera with a very high accuracy (~80%). By using advanced image processing techniques, the accuracy can be further improved. The system can thus be deployed in real world with slight modifications to solve the problem of over speeding, reducing accidents and saving lives.

9. References

1. OpenCV - http://docs.opencv.org/2.4/doc/tutorials/introduction/linux_install/linux_install.html
2. MJPG Streamer - https://sourceforge.net/projects/mjpg-streamer/
3. Tesseract - https://github.com/tesseract-ocr/tesseract/wiki
4. TIVA CCS - http://processors.wiki.ti.com/index.php/Download_CCS