COE 485: Senior Design Project (142)

Design Document

Introduction

Many people exceed the speed limit of the streets while driving unintentionally and drive with unsuitable speed. Each street was given a specific speed limit to assure drivers and pedestrian safety. Exceeding this limit will can cause car accidents and severe injuries.

The United Nations and road safety reported that as a result of car accidents 16 million people get injured and 700 thousand die annually. In Saudi Arabia there were about 485,000 accidents in 1436 AH. 6142 died in these accidents which on average we lose 17 people daily. According to the general Department of traffic "45.88% of the accidents were caused by exceeding the speed limit and it is the first cause of car accidents in Saudi Arabia."(Report 1430 AH, p 31)

This project will contribute to solve this problem by reducing the number of car accidents due to exceeding the speed limits. Many people exceed the speed limit without being aware about the actual speed limit or without realizing that he has already exceeded the limit. So this project will help the driver to know the speed limit of a street once he drive to that street and will be notified if he exceeded the maximum speed allowed.

The project will help to reduce the number of car accidents due to high speeds. it also will saves people money by knowing when exceeding the speed limit to not get a fee by Saher which at least will cost 300 Riyals per fee.

Problem Statement

Many people do not focus on the street while driving; hence they miss the max speed sign. That can lead to violate the speed or even worse get in accidents.

Background

There are some existing solutions for this problem. However, only two of them would be the ones that our solutions needs to compete with. They are as follows:

❖ **<u>Sygic (</u>**<u>mobile app</u>**<u>)</u>**<u>:</u>

This app is similar to google Maps but it has more features and road speed limit is one of them. However although it's not a free app, but it does tells how much the speed limit is in the street you drive in.

❖ **<u>Jaguar (</u>**<u>car</u>**<u>)</u>**<u>:</u>

Jaguar cars (some modules), have a smart system that provides the driver with speed limits in the streets. The system uses GPS and Jaguar libraries to compare the current speed with the speed limit and notify the driver when exceeding the limit. This solution is built in the car and it cannot be implemented in other cars as Jaguar holds patents on that technology.

Requirements and Specifications

4.1 Functional Requirement

- An Interface to show the max speed of the street
- A notification for exceeding the speed limit
- Mechanism to get the street max speed limit

4.2 Non-Functional Requirement

- System Response time should be sufficient for the driver to react
- Should consume small amount of the car battery
- The cost should be reasonable compared to the car cost

4.3 Technical Specification

- The device should be able to get the max speed of the street in less than 10 seconds
- The cost of the device should not exceed 1000 Riyal
- The driver alert must be easily recognized
- The device should consume more than 80 watt of power

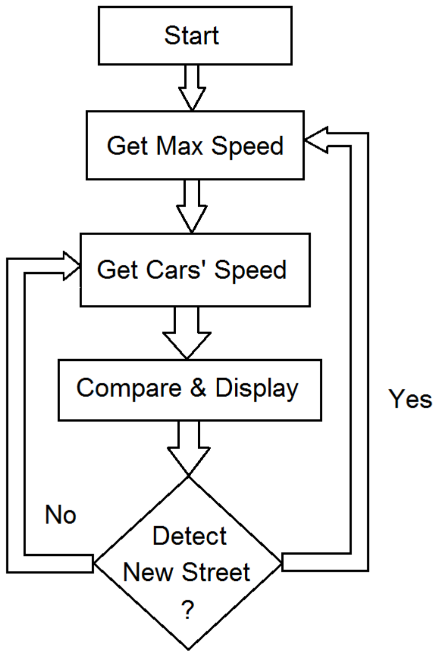
System Design

5.1 Solution Concept

• General approach of solving the stated problem.

The System will notify the driver about the maximum allowed speed once he drive to a specific street.

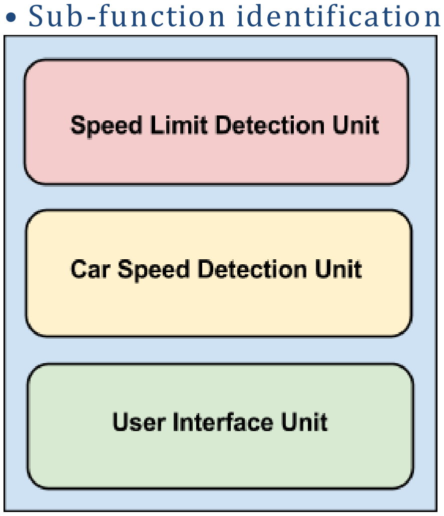
• Description of used/developed algorithms.



• Alternative approaches and algorithms, comparison, and selection criteria.

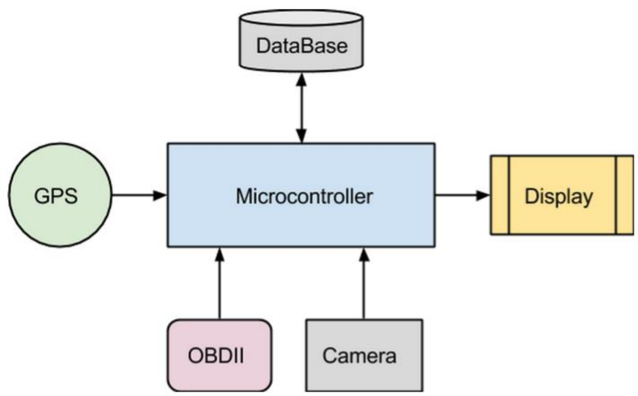
There was another approach to solve this problem by forcing the car not to exceed the speed limit. However that will require a different component design for each car and though its benefit will be limited unlike the selected approach which can be implemented in any car

• Sub-function identification.



5.2 Architecture (abdalaziz)

• System architecture and components.



• Alternative architectures, comparison, and selection criteria.

RFID-Only Architecture:

A possible design for the system was using RFID. That is, the system will have readers in the car and attached tags in the streets (on the signs or roads intersections). This architecture was an option for us, but we eliminated it for the fact that it would involve changes in the current streets infrastructure.

Database-Only Architecture:

We thought on using only a database to provide speed limit of the current street. But this approach will limit the system when it comes to new streets that are not already updated in the database. That will lead to the need of an always manual updating.

Camera-Only Architecture:

Another design would be using the camera only to read the speed limits from the signs on streets. That way, the camera is an essential component and it has to be up all the time taking pictures.

Hardware

GPS

OBDII

Camera

Software

Database

Computer Vision

GPS:

The component responsible for providing the system with the current location.

Database:

Weather it was locally stored or in the cloud, this component will store the information about all the streets. Then our system will request a speed limit for any street.

OBDII:

This component is an interface between our system and the car to get real-time information from the car. The current car speed is what matter for us, and we may get some other information later on.

Camera:

The part of the system that will provide us with live pictures from the street. It is supposed to take as much pictures as the system would need.

Display:

This component is the user interface. The driver will get notified when needed via screen and sounds.

5.3 Component Design

All of our project components are off-the-shelf, since they provide good performance.
The component is as follows:

Microcontroller

	Pi 1 B+	Pi 2 B	BBB	Edison	CI20
CPU	Arm11	Cortex A7	Cortex A8	Atom + Quark	MIPS
Cores	1	4	1	2 + 1	2
Clock	700MHz	900MHz	1000MHz	500MHz	1200MHz
GPU	Videocore IV	Videocore IV	PowerVR SGX530	None	PowerVR SGX540
Memory	512MB	1GB	512MB	1GB	1GB
USB Ports	4	4	2	1*	2
Flash	None	None	2GB	4GB	8GB
Storage	microSD	microSD	microSD	microSD*	SD
Network	10/100	10/100	10/100	None	10/100
GPIO	40-pin	40-pin	2x46-pin	70-pin Hirose	40-pin
Wifi	No	No	No	Yes	Yes
Bluetooth	No	No	No	Yes	Yes
RRP	$35	$35	$49	$85*	$65

FIGURE 1 SHOWS THE DIFFERENCES BETWEEN DIFFERENT PLATFORMS [1]

Raspberry Pi 2 - Model B

GPS: Ultimate GPS Breakout

Reasons to choose: Compatible with Raspberry board, price (40 $), up to 10 location updates a second.

Camera: Raspberry Pi NoIR Camera Board - Infrared-sensitive Camera

Reasons to choose: Compatible with Raspberry board, price (30 $), Infrared-sensitive Camera (for possible night vision feature), capable of shooting 90 frame per second.

Wi-Fi receiver: USB Wi-Fi (802.11b/g/n)

Reasons to choose: Compatible with Raspberry board,

• Custom components:

– Design and implementation, e.g. flow chart, state machine, pseudocode.

– Component design alternatives, comparison, and selection criteria.

Algorithm to check if the street in the data base if not turn on the camera to find the max speed of the street and upload It to the data base as follows:

If (exist in data base)

DB_Get max speed()

Else

Switch_Cam_On();

Cam_Get_Max_Speed();

Upload Max Speed()

Switch_Cam_Off()

5.4 System Integration

• Standard vs. custom interfaces between components, and justification for developing

Custom interfaces.

• Specification of custom interfaces.

• Component interaction, e.g. sequence diagrams.

We are using standard Interfaces for two reasons:

1- Efficient for the project.

2- Easily upgraded in the future

This Activity diagram shows How the system will work

Progress

Week	Task ID	Name	Owner	Start	End	Done
	0	**Preparations**		**4/2/2015**	**9/2/2015**	**Done**
2	0.1	search & Evaluate existing solutions	Jalal	4/2/2015	6/2/2015	Done
3	0.2	map Requirements to Specifications	Hassan	7/2/2015	9/2/2015	Done
	1	**Design**		**10/2/2015**	**20/02/2015**	**UW**
3-4	1.1	Find & Evaluate possible solutions	Hassan	10/2/2015	16/02/2015	**Done**
4	1.2	Evaluate and specify different components	Abdu aziz	17/02/2015	20/02/2015	**Done**
	2	**Working on subsystems**		**21/02/2015**	**20/3/2015**	
5	2.1	Getting familiar with the components	Jalal	21/02/2015	24/02/2015	**On Progress**
5-6	2.2	Define the subsystems and their interface	Abdu aziz	25/02/2015	27/2/2015	
6-8	2.3	Work on each subsystem individually	Hassan	28/2/2015	14/3/2015	
8-9	2.4	Test all subsystems individually	Jalal	15/3/2015	20/3/2015	
9-10	**3**	**Integrate the project**	**Jalal**	**21/3/2051**	**3/4/2015**	
11-12	**4**	**Test and debug**	**Abdu aziz**	**4/4/2015**	**17/4/2015**	

References

[1]Raspberry Pi 2 Benchmarked. (2015, February 4). Retrieved February 28, 2015, from http://www.davidhunt.ie/raspberry-pi-2-benchmarked/