Universidade do Minho
Escola de Engenharia

Mestrado Integrado em Engenharia Eletrónica Industrial e Computadores

---

Embedded Systems

Collision Avoidance and Transit Signals Detection Device

Professor: Adriano Tavares

Authors:

João Faria A85632

Tiago Costa A86122

October 22, 2020

Contents

List of Figures

1 Problem Statement

With the ever-evolving environment in this day and age, some efforts were being made regarding the self-driving capability.

Moreover, with the necessity of mass promoting autonomous driving, in this course, the system developed will facilitate that reality by being able to detect transit signals and by avoiding obstacle collision in real-time. That sensed information is to be sent to the driver by the infotainment system, a display attached to the Raspberry.

The developed product application is twofold. Firstly, in an industry environment, it will be able to detect and recognize QR codes in real-time. Secondly, after proving the initial concept, one will feed a pre-trained neural network with a data-set composed with a few different transit signals, for example, stop, blue crosswalk and prohibition to also detect and recognize them in real-time an alert the user, the driver, with appropriated text and audio alert of the respective signal.

Regarding obstacle avoidance, distance sensors will be used for obstacle detection and send commands to stop the actuating part of the target navigation system (robot vehicle prototype).

2 Problem Statement Analysis

After considering the problem the next step is to identify the features of the problem by detailing what entities are involved and which relations they share. The centre of data processing (in ESRG course) is Raspberry Pi that collects information about the camera, distance sensors and transmits it to speakers, stm, and display. All of that is represented in the diagram below.

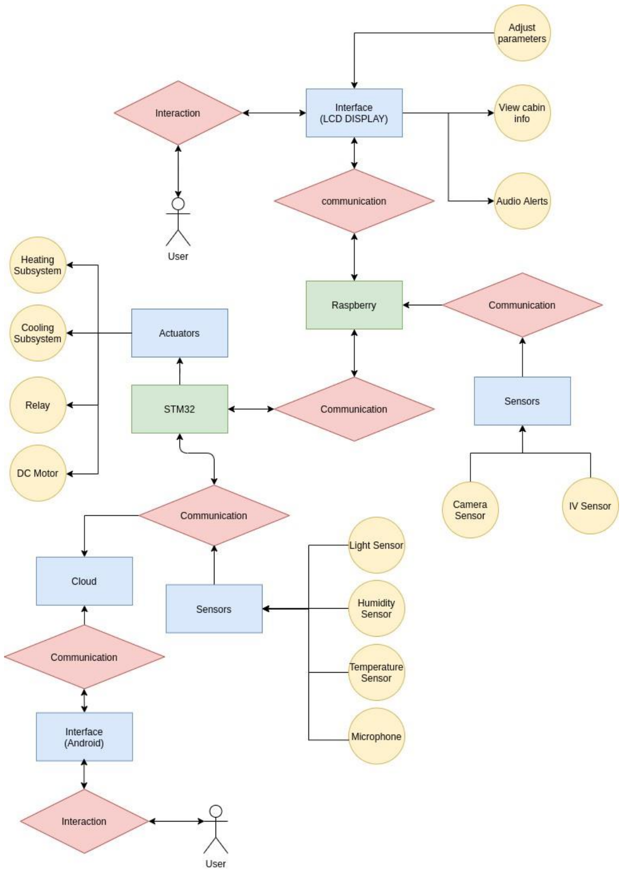

Figure 1: Problem Statement Analysis

3 Market Research

3.1 Target Market

The product developed will aid navigation for all types of drivers.

By combining transit signal detection and recognition with distance sensors the user (driver) can sense near obstacles approaching while it's informed about specific transit signals, increasing safety in the driving experience overall.

3.2 What is in the market?

After some research being made in the transit sign recognition

4 System

4.1 System Overview

To read and control the forementioned features, the input/output list is below described:

Inputs:

- Distance sensor, to sensor obstacules near the vehicle;
- Camera, to detect and recognize the target and transit signals;
- Display, allowing user input interation;

Outputs:

- Speakers, to transmit audio alerts back to the user;
- Display, works as an output too, displaying car signals, both inside and outside;
- STM-32, to send control commands to the respetive actuators;

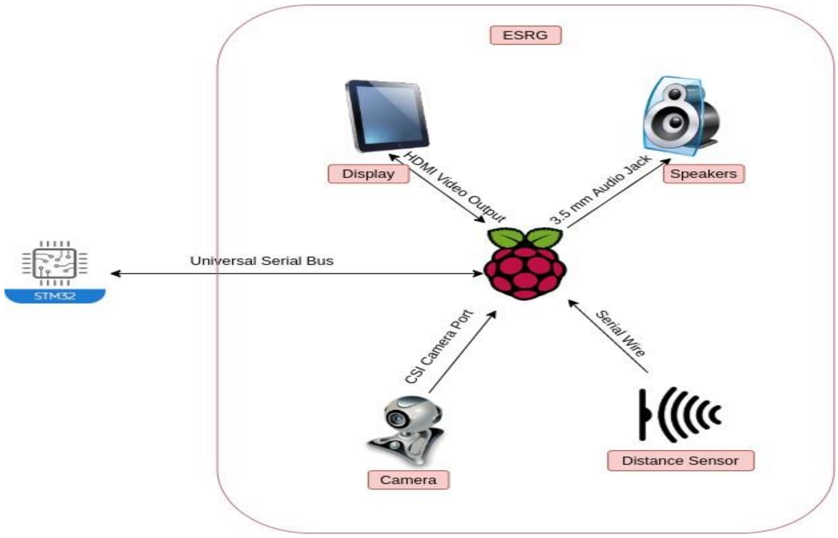

Figure 2: System Overview

4.2 Requirements and Constraints

Here we have 2 different types of requirements, functional requirements and non functional requirements. The functional requirements must specifies what the system is capable of doing and the non functional requirements specifies how the system will perform a certain action.

4.2.1 Project Requirements

Functional Requirements

- Avoid obstacule collision;
- Detect and recognize transit signals via camera vision;
- Displaying back to user driving related information inside and outside car cabin;
- Reproduce audio alerts;

Non Functional Requirements

- Low cost;
- Responsive;
- Reliable;
- Interoperability;
- Power Efficient;
- Adaptability;

4.2.2 Project Constraints

The constraints define the product limitations. They can be technical and non-technical. Technical constraints are associate with the technical part of the project whilst non-technical ones are project management related.

Technical Constraints

- Use RaspberryPi V4 board;
- Use pthreads;
- Use object-oriented programming languages;
- Embedded Linux by buildroot;
- Cross compiling toolchain;
- Device Drivers;

Non-Technical Constraints

- Group composed by 2 members;
- Deadline at end of semester;
- Limited budget;
- Soft Real Time Embedded System;
- Time to prototype;
- Soft Real-Time System;

4.3 System Architecture

The system hardware and the software architecture will be planned in more detail below to accomplish all the product features described in the aforementioned topics. This planning is part of a methodology that recommends breaking the global problem into minor problems to help to understand how things work independently. Finished the preparation, it is easier to build the project components in a structured fashion. So, the next figures present the established solution to fit the referenced architectures.

4.3.1 Hardware Architecture

In this section, we take a view of Hardware Architecture and how the inputs and outputs are grouped and connected to Raspberry V4.

Image data, distance sensor and touchscreen input are multiplexed into the Pi, sending then commands back to STM for motor actuation and for displaying and reproducing driving-related signals. Figure below illustrates that:

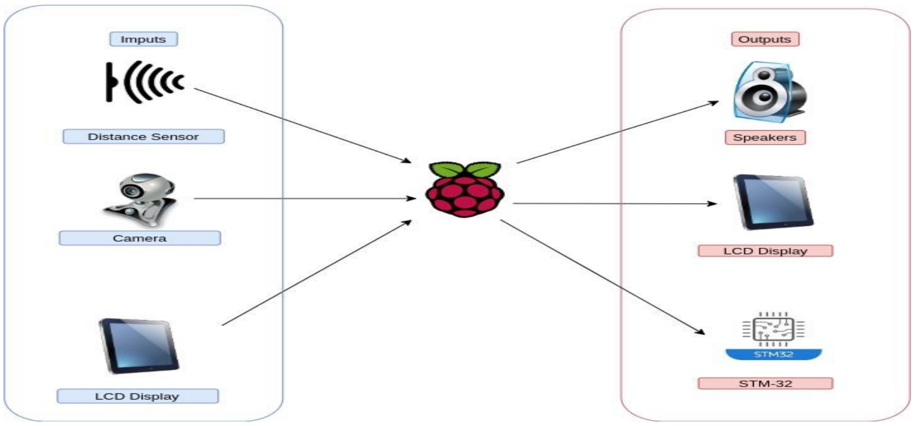

Figure 3: Hardware Architecture

4.3.2 Software Architecture

In the software architecture, specifically on the lower layer are represented the device drivers required for camera, USB, distance sensors as well as for the Touchscreen display and Audio.

The middle layer consists of a data acquisition/processing component, pthreads and openCV libraries for image processing.

Finally, the top application layer is composed of the reinforcement learning, GUI and sound output part.

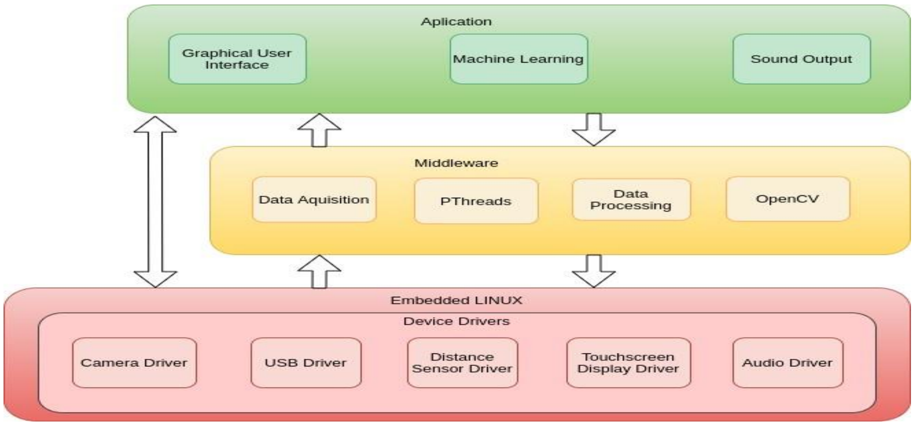

Figure 4: Software Architecture

5 Gantt Diagram

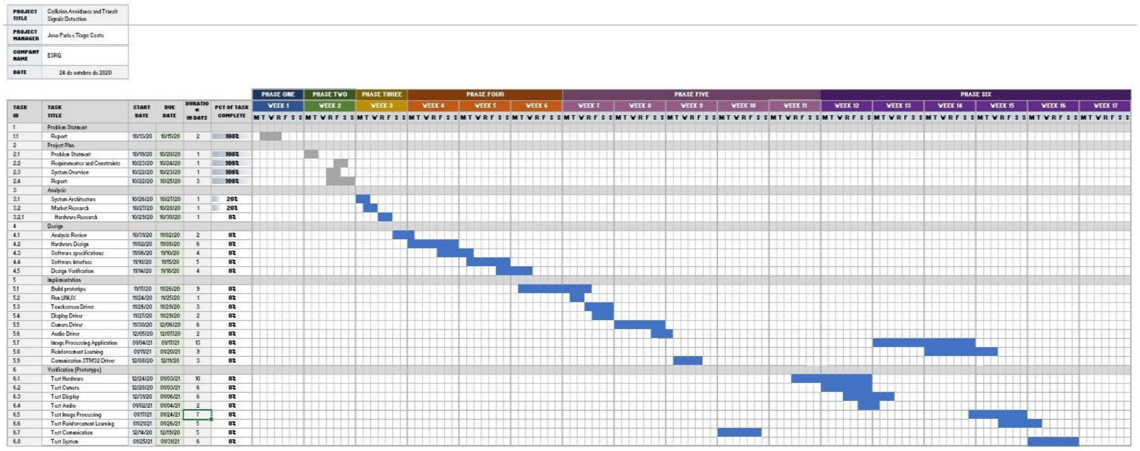

Figure 5: Gantt Diagram