Smart Parking lot

Submitted by: Charlene a. , Yarden m., Oleg s.

Motivation

Seeking a vacant parking space in congested cities during peak hours has always been frustrating and time consuming for many driver, being a serious issue to look after and with the evolution of technology in the age of Internet of Things (IoT), it's easy to see why smart parking solutions are considered innovative the environment of our solution has sensors and devices embedded into parking spaces, transmitting data on the occupancy status, and the vehicle drivers can search for parking availability. Hence using designated website, the driver would know where there is an available spot to park his vehicle in less time, reducing the energy consumption, air pollution and as a bonus enjoy comfortable payment service.

Hardware

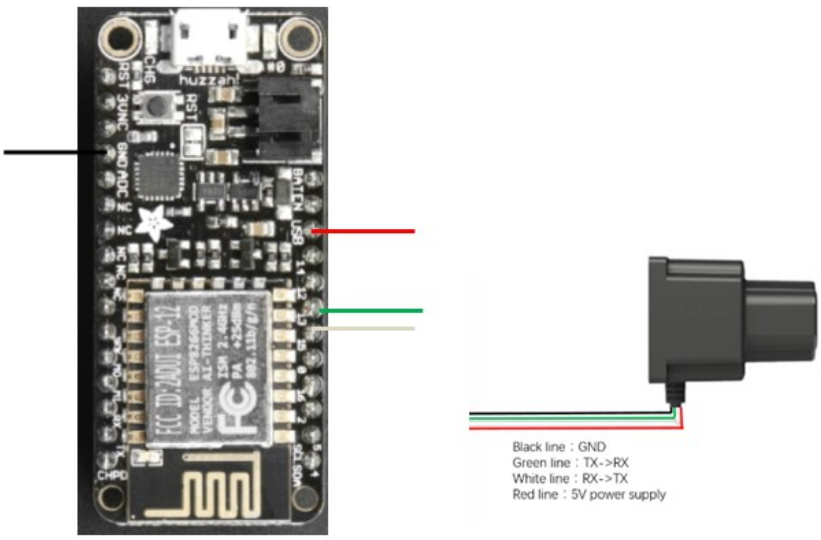
Adafruit HUZZAH Esp8266 and TFMini

The **ESP8266** is a low-cost Wi-Fi microchip with full TCP/IP stack and microcontroller capability. In our project, we connect the Esp8266 to a distance sensor - TFMini.

The TFMini based on ToF (Time of Flight) principle and integrated with unique optical and electrical designs, so as to achieve stable, precise, high sensitivity and high-speed distance detection.

We connect the TFMini to the Esp8266 in order to detect a car entering/exitinga parking place in the smart parking lot.

Connecting the Esp8266 to the TFMini scheme:



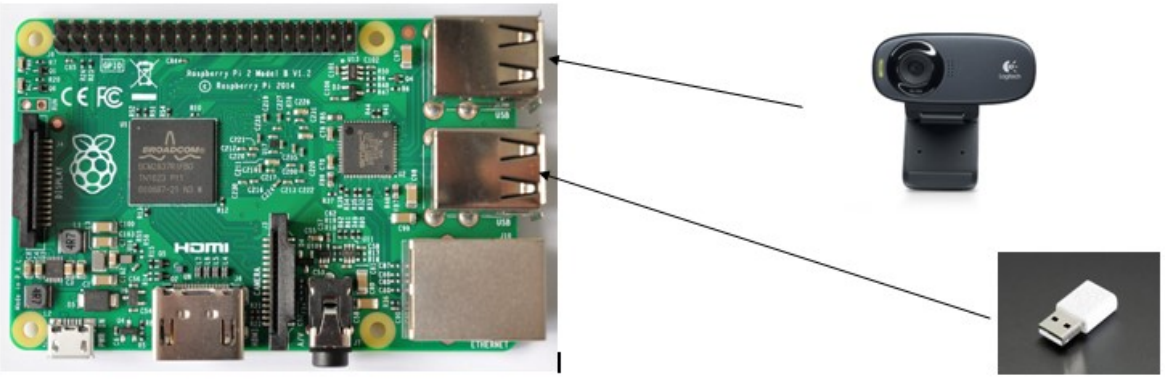
We have connected the gnd pin of the Esp8266 to the gnd of the TFMini (in black).
The TFMini requires 5v to operate, so we cannot connect the red line to the 3v pin of the Esp8266. We have connected the red line to the usb pin (in red) instead, that support 5v.
Then, we have connected the Rx pin of the TFMini to the TX pin of the Esp8266 (in white), and the TX pin of the TFMini to the Rx pin of the Esp8266 (in green).

The sensor detects if the car is entering or exiting a parking space.The Esp8266sends messages to azure iotHub in case there were change of status of the parking space.

(A car is entering? exiting?).Following, we update our website with the new information.

Raspberry Pi 2B, Wifi adapter and Logitec camera c310:

The Raspberry Pi is a series of small single-board computers developed in the United Kingdom by the Raspberry Pi Foundation to promote the teaching of basic computer science in schools and in developing countries. It is affordable and easy to use. The raspberry pi uses an SD card as its main external memory. The Wifi adapter is connected to one of the raspberry's pi ports and grants it internet access. We have installed Raspbian operating system on it. Following, we installed azure python packages, and finally we installed a daemon called motion that monitors the video input through the camera that is connected to one of the raspberry's pi usb ports. When an event occurs, the motion daemon takes a picture and a video. An event is any change of a predefined number of pixels in the picture. At the time of an event, a python script that uploads the image to azure storage blob is triggered.



Server (Azure Cloud)

In our project, we have two main parts. The first is the detection of the license plate of a car via the raspberry pi. The second is the detection of the car entering/exiting a parking space

via the arduino in the parking lot (in theory, we need for each parking space on the parking lot to have anEsp8266 chip, but for this project we only use one, and we monitor only one parking space).

A description of all of the parts of the server (on Azure cloud) we use for this project:

- **SQL Database:**stores parameters - information about car entries and exits. It contains several tables to describe the management of the parking.
    - We store a camera-pictures table that contains only the relevant part of the picture taken by the camera (the license plate number of the car). The relevant part is defined by the OCR API. We will not store the original picture on the database because it is too large, so we save only the relevant part.After using the OCR API, we save the clean number plate of the car in a table as a string(not only the picture). Since we don't have two cameras - one for entrance monitoring and one for exitmonitoring, we check the database if there was a previous entry. If so, we read this information as an exit of the car. Otherwise, we consider it as an entry.
    - A cars table, to store information regarding car movements in the parking lot such as entries, exits, duration of stay, payment, etc...
    - Aparking table, to describe the parking lot including - level, row, number, etc.Using the sensors, we can know if a parking space is busy, and how many available spaces remain.
    - A sensor table to describe the different distances of cars in the parking lot, and where they are located. In our project we use the "sensor2" to be in place "A2" as our TFMini sensor.
    - We also build different tables for the general case where we could use multiplecameras (one for entryand one for exit).We store different prices depending on the time cars stay in the parking lot. For example, acar that stays in the parking lot for 1 hour will be charge differently than a car that stays for 8 hours.
- **IOT HUB:** A component responsible for the communication between the Esp8266 board to the azure cloud. The board recognize an entrance/exit and sends it to the IOT Hub.
- **Storage blob:**We didn't want the server to listen constantly to events that are coming from the Esp8266 and the Raspberry pi so the storage blob for each of them holds the events, and the server polls the storage blobs every 10 seconds and takes all the recent events. The information from the IOT Hub to the storage blobs flows through Azure routing service.

- **Computer Vision – OCR:** The camera at the entrance or exit of the parking takes pictures of the license plates of cars, and sends it to Azure storage blob. Then, computer vision OCR is used to recognize the license plate number from the picture.
- **Web Sites:** we created a web site for the manager of the parking to follow, test the parking lot status (for example to know who paid and who not). In addition, we created a web site for clients of the parking lot to check for free parking spaces, follow charges,parking history, and check the current parking lot rates.
  Manager web site link: https://parkingcharlene.azurewebsites.net/
  User web site link: https://parkinguser.azurewebsites.net/
  The web application reads data from the storages, and updates the SQL database according to the data arriving. The web application updates the two websites, each 10 seconds. If there is a new blob in the camera storage, the web application calls the OCR, and updatesthe SQL database (it updatesthe SQL database also for new data from the Esp8266), and the web site (for example, the number of free places on the parking).

User Interface (Web Site)

We built web sites running on Azure's web service. We chose to build two different interfaces -one for the user of the parking lot, and the other for the admin of the parking lot.

- User: every user can see the number of free spaces in each row of the parking lot, and then he can find a space more easily. The user can also see the rates (for payment purposes), and see the details of his account, where all of his entering and exitingare registered. Each user is registered to the web site with his license plate.

- Admin: the admin has an overview of the parking lot. He can see all the free/occupied spaces in the parking lot, see the list of cars entering and leaving the lot for a selected period of time, see the pictures taken by the camera in the gates (in case of error of the OCR), check which car has entered, which has exited, and who paid.

First video for camera utility:

https://youtu.be/59r05EftpIY

Second video for distance sensor utility:

https://youtu.be/IyF1W3NO-u4

Diagram

The following diagram explains the interactions between all the services.