Guided Parking Structures

Jacob Ditkoff
Department of Computer Science
Binghamton University
Binghamton, NY, US
jditkof1@binghamton.edu

ABSTRACT

In modern times, a major issue affecting many cities is ensuring that there are parking spots available for its population. The stress and anxiety that pairs with slowly traversing a packed parking lot in the hopes that you'll find a spot is a feeling with which many people are accustomed. It is an activity which has the potential to waste a great deal of time while often providing no reward.

The motivation for this project was to provide a proof-of-concept, low-cost design that could be incorporated into both already-existing parking structures and new ones as well. Parking spots can be monitored with either ultrasonic sensors or weight sensors to inform a screen or display at the entrance of a parking structure. If a spot is available, the driver can then be easily guided to the open spot.

KEYWORDS

Arduino Uno, ATMega2560, HC-SR04, Ultrasonic Sensor, C, C++

1 Introduction

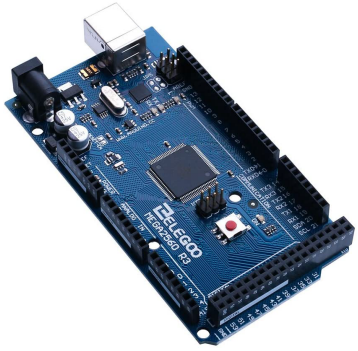
This Guided Parking Structure design will allow for drivers to easily know whether there are available parking spots in a given parking lot. If there are, then they will be guided to an open spot. This design was created to be low-cost and easily integrated into both existing and new parking structures. An Arduino UNO and an ATMega2560 board are used to monitor spots through multiple sensor options and then display that information to a driver at the entrance of the parking lot. A series of LEDs will guide the driver to their designated spot, and the spot's LED will light up as well. If no spots are available, then that information will be displayed at the entrance so that drivers will not need to waste their time searching.

2 Design and Implementation

The design of this parking structure will be explained in the order through which it is intended to be used. When a car pulls up to the entrance, the driver will either press a button, or an ultrasonic sensor will detect that a vehicle has pulled up. A signal will be sent to the ATMega2560 Board stating that a vehicle is ready to receive a spot. The ATMega2560 contains the logic for finding an open spot, displaying the spot number, and then guiding the driver to the designated spot through a series of LEDs. If a spot is available, the spot number will be displayed and a series of LEDs will illuminate to guide the driver to their spot. Once the vehicle is in its intended spot, the Arduino Uno will monitor all the spots and determine when one has become free. It will send a signal back to the ATMega2560 informing it of which spot has opened up.

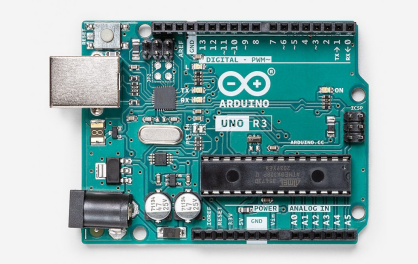
3 Hardware

3.1 ATMega 2560



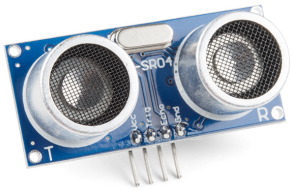
The ATMega 2560 board operates as the primary control unit for this project. It monitors the ultrasonic sensor or button at the entrance of the parking lot, contains the control logic for populating parking spots and guiding drivers to their spot, displays whether spots are available, communicates with the Arduino UNO, and contains the data representation of which spots are available or occupied. This board was chosen for its compatibility with Arduino drivers, Arduino libraries, and its large number of digital pins.

3.2 Arduino UNO



The Arduino UNO board monitors the parking spots within the parking structure by communicating directly with the ultrasonic sensors or weight sensors placed at a given parking spot. The board contains logic which constantly polls the sensors to determine if a vehicle or object is in a given spot. It then sends that data to the ATMega 2560 board.

3.3 HC-SR04 Ultrasonic Sensor



Three HC-SR04 ultrasonic sensors were used in this project. They are placed in front of parking spots and monitor when a vehicle or object gets within close a enough range to be considered a parked vehicle. They communicate this information to the Arduino UNO.

3.4 LEDs

Six LEDs were used in this project, with each spot being designated one LED, and three more LEDs used as a guide to find the spot.

4 Software

4.1.1 Software Implementation

The software was implemented using Arduino's IDE, which allows for direct interfacing, monitoring of, and uploading to Arduino and Arduino-compatible FPGA boards. The programming language used was Arduino's variant of C and C++, which allows for logic to be more easily implemented on Arduino FPGA boards. Aside from Arduino's built-in libraries, "Timer" was also used to allow for extra functionality like timing out when a spot isn't populated within a certain amount of time [1].

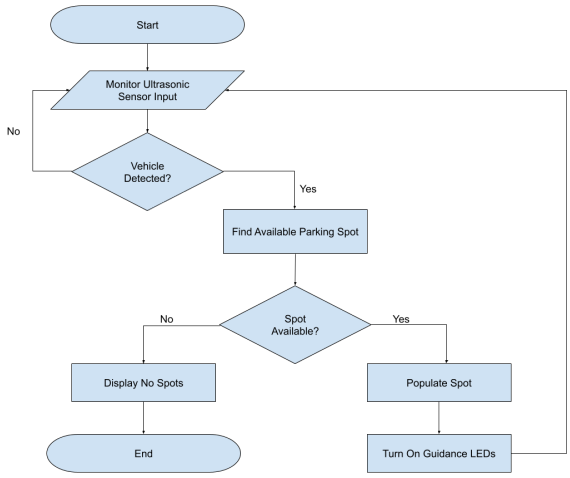


Figure 1: A flowchart depicting the generalized strategy of the software for populating a given parking structure. This was the flow used to design the proof-of-concept parking structure.

5 Results and Future Development

The current version of this proof-of-concept is a solid foundation upon which more complex and individualized variants can be built. I was able to accomplish my task of creating a low-cost project which could be implemented in both already-existing and new parking lots. The logic in the code I've written can be taken and expanded upon to fit larger parking lots of different shapes and sizes.

Due to hardware breaking at an unfortunate time, I had to replace my weight sensors with spare HC-SR04 Ultrasonic Sensors to monitor the parking spots. Because of hardware failures, I also had to replace the ultrasonic sensor I was using to monitor the entrance with a "button". However, the logic in my code allows for the ultrasonic sensors to be easily and readily swapped out with weight sensors with only quick minor adjustments to the code.

With more time and a larger budget, I would have liked to have better quality weight sensors which I could use to monitor the spots rather than ultrasonic sensors. With weight sensors, you could monitor the weight of the parked object to determine if it's a shopping cart that was left in a spot, a bike or a motorcycle, a car, an SUV, or a truck. With this information, you could create zones where different types of vehicles can park to allow for a smoother

flow of traffic. With further time, a web or phone app could be created to monitor the availability of spots in real-time.

REFERENCES

[1] https://github.com/brunocalou/Timer.
[2] https://store.arduino.cc/usa/arduino-uno-rev3
[3] https://www.amazon.com/ELEGOO-ATmega2560-ATMEGA16U2-Compatible-Arduino/dp/B01H4ZLZLQ?th=1
[4] https://www.sparkfun.com/products/15569
[5] https://www.sparkfun.com/products/9590