Quest Journals
Journal of Software Engineering and Simulation
Volume 6 ~ Issue 1 (2020) pp: 01-13
ISSN(Online) :2321-3795 ISSN (Print):2321-3809
www.questjournals.org

Research Paper

Automatic Smart Car Parking System Using Iot and Python

A.Durga Praveen Kumar, Harika Matta, Pravallika Saka, Poojitha Reddy Erusu, Dinesh Chandra Koyyalmudi,

Assistant Professor, Information Technology, Anil Neerukonda Institute of Technology and Sciences, Visakhapatnam, India
Student, Information Technology, Anil Neerukonda Institute of Technology and Sciences, Visakhapatnam, India
Student, Information Technology, Anil Neerukonda Institute of Technology and Sciences, Visakhapatnam, India
Student,Information Technology, Anil Neerukonda Institute of Technology and Sciences, Visakhapatnam, India
Student, Information Technology, Anil Neerukonda Institute of Technology and Sciences, Visakhapatnam, India
Corresponding Author: A.Durga Praveen Kumar

ABSTRACT: *Internet of Things (IOT) plays a vital role in connecting the surrounding environmental things to the network and made easy to access those un-internet things from any remote location. It's inevitable for the people to update with the growing technology. Normally at public places such as multiplex theatres, market areas, hospitals, function-halls, offices and shopping malls, one experiences the discomfort in looking out for a vacant parking slot, though it's a paid facility with an attendant/ security guard. In this study we design an Advanced Smart Parking System (ASPS) which detects the car and displays the vacant slot on the display at the entrance of the parking so that the user gets to know the availability /unavailability of parking space prior to his/her entry into the parking place. This paper presents the intelligent parking system which apply Dijkstra's algorithm in finding the shortest path. The proposed intelligent parking guidance system is a system that assigns the nearest vacant bay to drivers with necessary direction printed on the ticket so that drivers are able to find the 'best' lot with the minimum amount of time. The system will automatically check for the nearest empty lot and reserve the lot for the user so that the next user will not get the same lot again It also provides the best optimal path to reach the parking slot provided with the directions shown using LEDs. Software and hardware implementations have been carried out. Few electronic components such as PICs, IR sensors, push buttons, LEDs, LCDs, counters, comparators, and servo motors have been used to realize the system. Implementation involves minimal human interaction and provides a seamless parking experience thereby reducing a lot of time wasted by the user in parking his/her vehicle.*

Keywords- *IOT,Advanced Smart Parking System,Sensors,Automatic License Plate Recognition System, Optical Character Recognition*

Received 04 Mar., 2020; Accepted 22 Mar., 2020 © The author(s) 2020.
Published with open access at ***www.questjournals.org***

I. INTRODUCTION

The conventional and current parking system in Malaysia requires drivers to receive parking tickets and find the parking lot by themselves. The difficulty in searching the available parking lot leads to time and fuel wasting and causes high frustration and stress level of drivers. carparking area has many lanes/slots for car parking. So to park a car one has to look for all the lanes. A lot of time is wasted in searching vacant slot for parking and many a times it creates jams. , this involves a lot of manual labour and investment. So, there is a need to develop an automated parking system that indicates directly the availability of vacant parking slots in any lane right at the entrance. So the person desirous to park his vehicle is well informed about the status of availability of parking slot. The proposed intelligent parking system is a counter based indoor car parking system which able to count, display, assign the nearest parking slot. Upon arrival at the parking entrance, the system assigns the nearest parking slot and display the way to assigned parking slot. Dijkstra's algorithm is applied to calculate each of the lots distance to the specific mall entrance as the destination. The directions for the assigned slot are given through the LED boards. Once the system passes the node, the LED automatically turnoff. This result in no traffic jams in the path. The system would not only save time but the software and hardware would also manage the Check-in and check-outs of the cars under the control of Automatic License plate Recognition with additional features of Entry exit data logging. In this system, the users are guided to the vacant slot for parking using LED Displays placed along the path of the parking slot, these displays show a visual representation of directions to reach the parking slot. The parking charges are automatically generated based on the time spent inside the parking area

1.1 Problem Statement and Motivation

Nowadays, most of the office buildings and shopping mall had built underground parking and multilevel parking to overcome the number of cars which is increasing rapidly. However, drivers are still difficult to find an available parking slot to park their car. The process of looking for a 2 parking lot is time consuming, confusing and wasting fuel as well. At this point of time, someone may miss or late for their important event. This might cause frustration for the drivers. Eventually the effect of lacking parking slot will causes the officer to have bad mood or consumer to leave the shopping mall without purchase anything. The side effect of this problem is serious and need a better solution to handle it.

1.2 Research Objectives

- Reduces the required manpower to maintain a parking system.
- Senses when a car arrives and scans the license number.
- Checks for available slots and finds the best optimal path the parking slot.
- Shows directions to the available parking slot.
- Updates the database with the car license number, entry time and exit time.

1.3 Motivation for the Work

In present world, there are very large numbers of shopping complexes and cinema theatres are present. Each shopping complex has it's own parking place for their customers vehicles. For that they need to assign some people to show the directions to drivers to park the vehicles. Here is the motivation for this project. To reduce the man power in parking places by making automated system to display the way to the drivers to parking slot. This results in reducing cost in wages to the employees every month.

1.4 Dataset

- First, we will take the input of number of vertices.
- Secondly, we will enter the edges for each vertex until the last vertex.
- Next, we will take the indices of the parking slots.
- Now we will enter the vertex of entry, to find the shortest path.

II. LITERATURE

2.1 Existing Systems for Smart Parking

2.1.1Smart Parking System using RFID

This system uses RFID to match the vehicle's unique RFID tag with the value in the database when it is read by the RFID reader in the parking lot entrance

Advantages:

1.This is a fast method of identification and quite cost efficient.

Disadvantages:

1. If the RFID tags are damaged or more than one tags are read at a time, the system fails to work accurately.

2.1.2 Smart parking reservation system using Bluetooth and Zigbee sensors

This system uses a Bluetooth communication technique which is used for verifying the driver's identity and also to book a slot by identifying the vacant spaces. Zigbee sensors are used to detect the vehicle.

Advantages:

1. Internet usage is not necessary.
2. It is a decentralized system.

Disadvantages:

1.Range of Bluetooth is limited.
2. Installation and maintenance is difficult.
3. Connection gets disconnected if the driver is inactive and again a new slot has to be booked.

2.1.3Smart Parking System using IR sensors

This proposed system uses feedback mechanism to find the availability of parking spaces. Infrared sensors are used to monitor the parking spaces.

Advantages:

1. Proper utilization of slots is managed properly.
2. This could be implemented in a small budget.

Disadvantages:
1.Availability of the space could be found only after the car enters the parking lot, so if parking space is not available it has to avert from there and it might

2.4 Smart Parking System using optic Wireless Sensor Network
This proposed system uses feedback mechanism to find the availability of parking spaces. Infrared sensors are used to monitor the parking spaces.

Advantages:
1. Proper utilization of slots is managed properly.
2. This could be implemented in a small budget.

Disadvantages:
1.Availability of the space could be found only after the car enters the parking lot, so if parking space is not available it has to avert from there and it might

2.2 Optimal Path Finding(Shortest path)

The shortest path problem is about finding a path between 2 vertices in a graph such that the total sum of the edges weights is minimum. The shortest path problem can be solved easily by using many algorithms like BFS, Bellman-Ford algorithm, Dijkstra's algorithm etc. In our project, the optimal path can be find out by using Dijkstra's Algorithm based on certain parameters and requirements.

2.2.1 Breadth-First Search (BFS):

Breadth-First Search algorithm is used to find the shortest path between the two nodes where the distance between them is equal to 1.Here we cannot use the BFS algorithm because the distance between the nodes is variant and it is not equal to 1. Due to variation in distances between the nodes, BFS algorithm can't be implemented.

2.2.2 Bellman Ford Algorithm:

Bellman Ford algorithm is used to find the shortest paths from the source vertex to all other vertices in a weighted graph. It depends on the following concept: Shortest path contains at most N-1 edges, where N is the number of vertices in a graph, because the shortest path couldn't have a cycle. This algorithm depends on the relaxation principle where the shortest distance for all vertices is gradually replaced by more accurate values until eventually reaching the optimum solution. In the beginning all vertices have a distance of "Infinity", but only the distance of the source vertex =0, then update all the connected vertices with the new distances (source vertex distance + edge weights), then apply the same concept for the new vertices with new distances and so on.
The reasons for not using this algorithm in this project are:
1.The time complexity of Bellman-Ford's algorithm is relatively very high i.e., O(V*E). incase E=V^2 then Time Complexity becomes O(V^3).

2.2.3 Floyd's-Warshall's Algorithm:

Floyd–Warshall's Algorithm is used to find the shortest paths between between all pairs of vertices in a graph, where each edge in the graph has a weight which is positive or negative. The 5 advantage of using this algorithm is that all the shortest distances between any 2 vertices could be calculated in O(V^3), where V is the number of vertices in a graph.
The reasons for not using this algorithm in this project are:
1. This project does not contain any negative edges because distance can't be negative.
2. Addition to that the complexity of Floyd's-Warshalls Algorithm is O(V^3), where V is the number of vertices in the graph, which takes long time compared to Dijkstra's algorithm.

2.2.4 Dijkstra's algorithm:

The algorithm exists in many variants; Dijkstra's original variant found the shortest path between two nodes, but a more common variant fixes a single node as the "source"; node and finds shortest paths from the source to all other nodes in the graph, producing a shortest-path tree . For a given source node in the graph, the algorithm finds the shortest path between that node and every other .It can also be used for finding the shortest paths from a single node to a single destination node by stopping the algorithm once the shortest path to the destination node has been determined.

Algorithm:

Let the node at which we are starting be called the initial node. Let the distance of node Y be the distance from the initial node to Y. Dijkstra's algorithm will assign some initial distance values and will try to improve them step by step.

1. Mark all nodes unvisited. Create a set of all the unvisited nodes called the unvisited set.
2. Assign to every node a tentative distance value: set it to zero for our initial node and to infinity for all other nodes. Set the initial node as current.
3. For the current node, consider all of its unvisited neighbours and calculate their tentative distances through the current node. Compare the newly calculated tentative distance to the current assigned value and assign the smaller one. For example, if the current node A is marked with a distance of 6, and the edge connecting it with a neighbour B has length 2, then the distance to B through A will be 6 + 2 = 8. If B was previously marked with a distance greater than 8 then change it to 8. Otherwise, keep the current value.
4. When we are done considering all of the unvisited neighbours of the current node, mark the 6 current node as visited and remove it from the unvisited set. A visited node will never be checked again.
5. If the destination node has been marked visited (when planning a route between two specific nodes) or if the smallest tentative distance among the nodes in the unvisited set is infinity (when planning a complete traversal; occurs when there is no connection between the initial node and remaining unvisited nodes), then stop. The algorithm has finished.
6. Otherwise, select the unvisited node that is marked with the smallest tentative distance, set it as the new "current node", and go back to step 3.

When planning a route, it is actually not necessary to wait until the destination node is "visited" as above: the algorithm can stop once the destination node has the smallest tentative distance among all "unvisited" nodes (and thus could be selected as the next "current").

Why we use this algorithm?
1. It is a single source and multiple destination algorithm.
2. This algorithm can be implemented for variant weights in a graph.
3. No point of negative edges in a graph.
4. Main point is that Time Complexity. Time Complexity of Dijkstra's Algorithm is O (V^2) but with min-priority queue it drops down to O(V+ElogV).

2.3 Automatic License Plate Recognition System(ALPR)

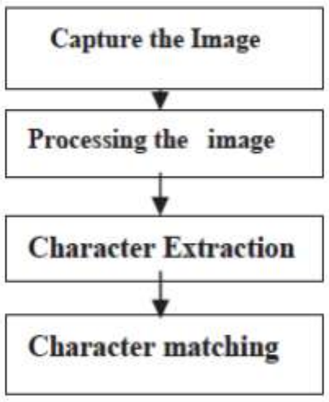
Automatic number-plate recognition (ANPR; see also other names below) is a technology that uses optical character recognition on images to read vehicle registration plates to create vehicle location data. Optical Character Recognition (also optical character reader, OCR) is the mechanical l or electronic conversion of images of typed, handwritten or printed text into machine-encoded text, whether from a scanned document, a photo of a document, a scene-photo (for example the text on signs and billboards in a landscape photo) or from subtitle text superimposed on an image (for example from a television broadcast).

ALPR systems function to automatically capture an image of the vehicle's license plate, transform that image into alphanumeric characters using optical character recognition or similar software, compare the plate number acquired to one or more databases of vehicles. There are two types of ALPR: stationary, which uses infrared (IR) cameras at high fixed points, and mobile, which uses vehicle-mounted IR cameras.

Stationary cameras can be mounted on signs, street lights, highway overpasses or buildings as a cost-effective way to monitor moving and parked vehicles twenty-four hours a day. Camera software is able to identify the pixel patterns that make up a license plate and translate the letters and numbers on the plate to a digital format. The plate data is then sent to a database where it is compared in real-time to a list of plate numbers that belong to "vehicles of interest". If the system detects a match, it sends an alert to the dispatcher or other designated personnel.
Automated License Plate Recognition has many uses including:

- Recovering stolen cars.
- Identifying drivers with an open warrant for arrest.
- Catching speeders by comparing the average time it takes to get from stationary camera A to stationary camera B.
- Determining what cars do and do not belong in a parking garage.
- Expediting parking by eliminating the need for human confirmation of parking passes.

Steps:



A. Capturing of Images:
In this process a high resolution analog/digital camera is used to capture the image

B. Processing the Image:
In this process firstly gray scale conversion is used to convert the image from RGB to grayscale. After that image will be resized by resize function. Then median filter is used to remove the salt and pepper noise.

C. Character Extraction:
After median filtering process we used smearing process to find out the text area from the plate and get the erode image Fig.4 and then we applied morphological process to remove the unwanted edges of the plate as shown in Fig.5,it also include dilution process. Dilution means to fill the gap/to separate the character from the image and after that each character is cut separately from done by finding starting and end points of the characters in horizontal and vertical direction .Characters cut from plate areas are shown in Fig.6

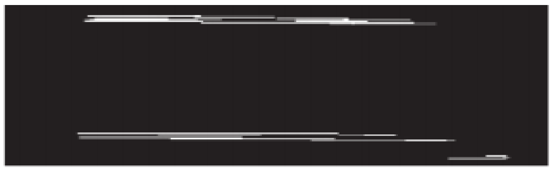


Fig.4

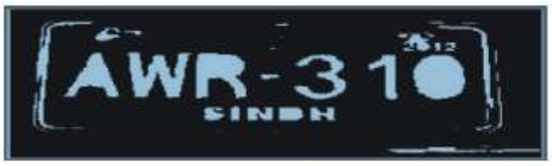


Fig.5

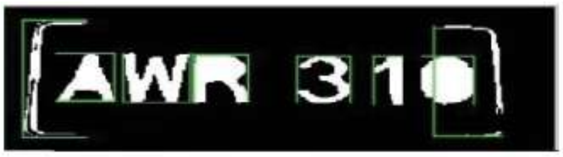


Fig.6

D. Character Matching:

Before the Character Matching Normalization process is occurring in order to normalize the character means not to include any other white or extra spaces in all the four sides of the character. Then each character is to fit to equal size as shown in Fig.10

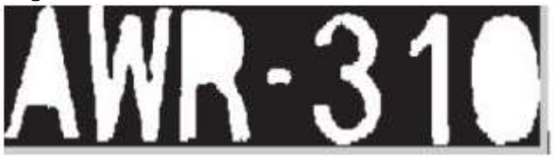


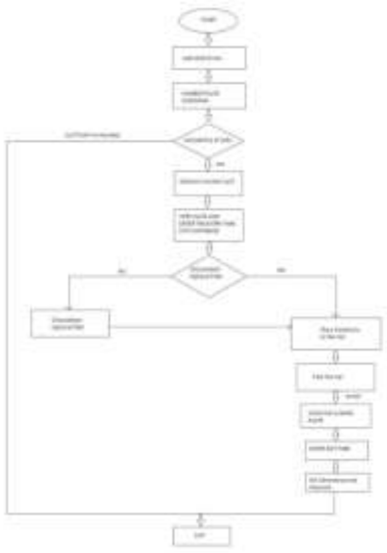
Fig.7

III. METHODOLOGY

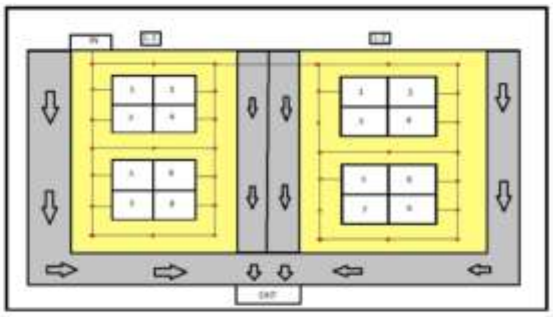
3.1 Proposed System:

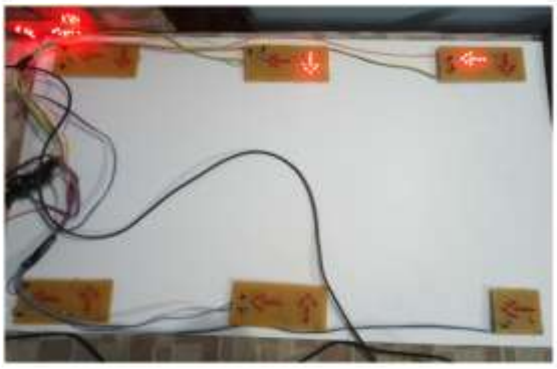
Being in one of the Indian Metro Cities which has a highly active urban life, people face a lot of everyday situations with automobiles. And one of the top few problems is the car-parking, for sure. Car Parking in shopping malls is a task that takes up a lot of human time and effort. And it happens in a very large scale that is difficult for us to comprehend. The proposed automatic smart car parking system will overcome all the challenges and difficulties that are there in conventional car parking system. It saves us time and effort that we put into this task, and also the fuel wastage can be reduced. It could help to bring order out of the chaos that exists at present in the car parking issues and make it really simple and easy.

3.2 System Architecture



3.3 Working Model





IV. EXPERIMENT ANALYSIS AND RESULTS

4.1 System Requirements

4.1.1 Software Requirements

- Python version 2.7.X or 3.6.X or 3.7
- Intel Distribution for Python 2018
- Google Chrome Version 72.0.3626.121 (Official Build) (64-bit)

4.1.2 Hardware Requirements

A. Recommended System Requirements

- Processor: Intel Core i5.2.60 GHz
- RAM: 8 GB DRAM
- Disk Space: 2 to 3 GB
- Operating Systems: Windows 10,mac OS*,and Linux*

B. Minimum System Requirements

- Processor: Intel Core i3
- RAM: 8 GB DRAM
- Disk Space: 2 to 3 GB
- Operating Systems: Windows 7 or Later ,mac OS*,and Linux

4.1.2.1 Sensors

A sensor is a device that detects and responds to some type of input from the physical environment. The specific input could be light, heat, motion, moisture, pressure, or any one of a great number of other environmental phenomena. The output is generally a signal that is converted to human-readable display at the sensor location or transmitted electronically over a network for reading or further processing.

Proximity Sensor

A Proximity Sensor is a non-contact type sensor that detects the presence of an object. Proximity Sensors can be implemented using different techniques like Optical (like Infrared or Laser), Ultrasonic, Hall Effect, Capacitive, etc.

Fig.10

Some of the applications of Proximity Sensors are Mobile Phones, Cars (Parking Sensors), industries (object alignment), Ground Proximity in Aircrafts, etc.

Force Sensing

Resistor The physical object which is able to detect events and changes in various parameters such as environment, temperature, humidity, and so on are termed as sensors. Based on the (events or) changes detected the sensors are capable of generating appropriate output. There are different types of sensors classified based on different criteria such as sound, automotive, electrical, chemical and so on. Most frequently used sensors can be listed as pressure, force, proximity, light, heat, temperature, position, etc.

Fig.11

Force sensing resistor can be defined as a special type of resistor whose resistance can be varied by varying the force or pressure applied to it. The FSR sensor technology was invented & patented by Franklin Eventoff in 1977. The FSR sensors are made of conductive polymer which has a property of changing its resistance based on the force applied to its surface. Hence, these are termed as FSR sensors, force sensing resistor is a combination of resistor and sensor technology.

Even though there various types of force sensors, the force sensing resistors are having several advantages such as thin size (less than 0.5mm), very low cost and also good shock resistance.
The only disadvantage of FSR sensors is low precision, there will be approximately 10% or more difference in measurement results.

4.1.2.2 LED Display Boards

What is LED..?

In the simplest terms, a light-emitting diode (LED) is a semiconductor device that emits light when an electric current is passed through it. Light is produced when the particles that carry the current (known as electrons and holes) combine together within the semiconductor material. Until the mid-90s LEDs had a limited range of colors and in particular commercial blue and white LEDs did not exist. The development of LEDs based on the gallium nitride (GaN) material system had given the availbility of colors and opened up many new colourful LED’s and new applications.

Benefits of LED’s

- Low power requirement
- High efficiency
- Long life

Typical applications include:

- **Indicator lights**: These can be two-state (i.e., on/off), bar-graph, or alphabetic-numeric readouts.
- **LCD panel backlighting**: Specialized white LEDs are used in flat-panel computer displays.

Remote control: Most home-entertainment "remotes" use IREDs to transmit data to the main unit.
LED could be used with a computer

- Keyboard LEDs
- Mouse LED
- Motherboard LED
- Floppy, Hard drive, CD-ROM, and other drives LED
- Printer, speakers, monitor, and other devices.

4.1.2.3 Raspberry pi

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. The original model became far more popular than anticipated, selling outside its target market for uses such as robotics. It does not include peripherals (such as keyboards and mice) and cases.

Various operating systems for the Raspberry Pi can be installed on a Micro SD, Mini SD or SD card, depending on the board and available adapters; seen here is the Micro SD slot located on the bottom of a Raspberry Pi 2 board.

Fig.12

4.1.3 ALPR:(Automatic License Plate Recognition)

Automatic number-plate recognition (ANPR; see also other names below) is a technology that uses optical character recognition on images to read vehicle registration plates to create vehicle location data. Optical Character Recognition (also optical character reader, OCR) is the mechanical l or electronic conversion of images of typed, handwritten or printed text into machine-encoded text, whether from a scanned document, a photo of a document, a scene-photo (for example the text on signs and billboards in a landscape photo) or from subtitle text superimposed on an image (for example from a television broadcast).

ALPR systems function to automatically capture an image of the vehicle’s license plate, transform that image into alphanumeric characters using optical character recognition or similar software, compare the plate number acquired to one or more databases of vehicles. There are two 16 types of ALPR: stationary, which uses infrared (IR) cameras at high fixed points, and mobile, which uses vehicle-mounted IR cameras.

Stationary cameras can be mounted on signs, street lights, highway overpasses or buildings as a cost-effective way to monitor moving and parked vehicles twenty-four hours a day. Camera software is able to identify the pixel patterns that make up a license plate and translate the letters and numbers on the plate to a digital format. The plate data is then sent to a database where it is compared in real-time to a list of plate numbers that belong to "vehicles of interest". If the system detects a match, it sends an alert to the dispatcher or other designated personnel.

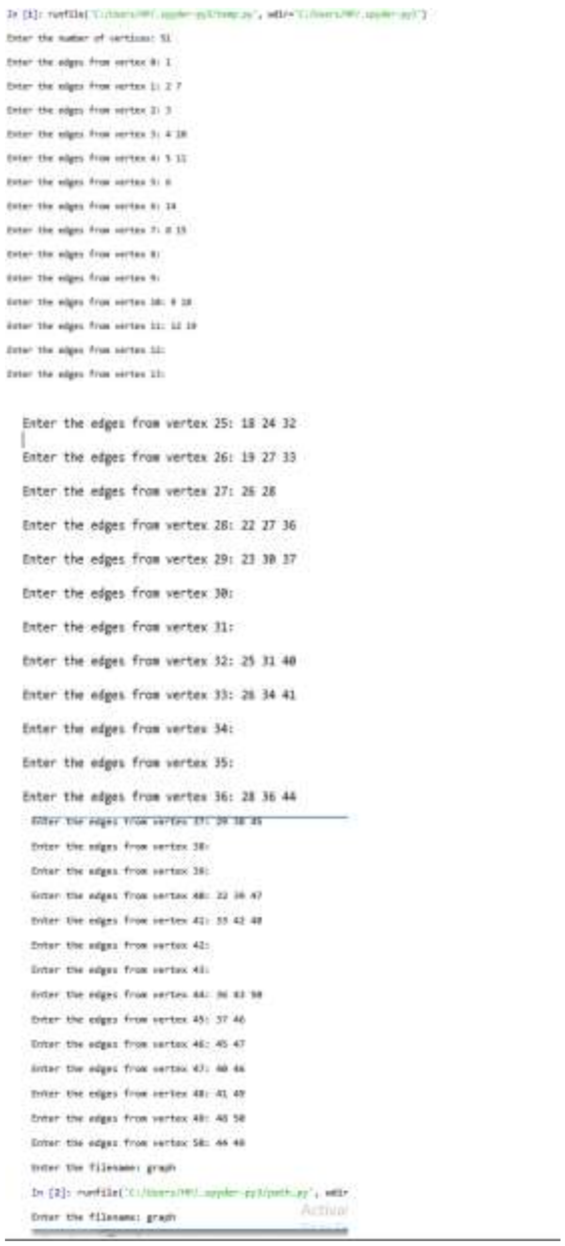
Automated License Plate Recognition has many uses including:

- Recovering stolen cars.
- Identifying drivers with an open warrant for arrest.
- Catching speeders by comparing the average time it takes to get from stationary camera A to stationary camera B.
- Determining what cars do and do not belong in a parking garage.
- Expediting parking by eliminating the need for human confirmation of parking passes

Fig.15

RESULTS

Graph Creation:

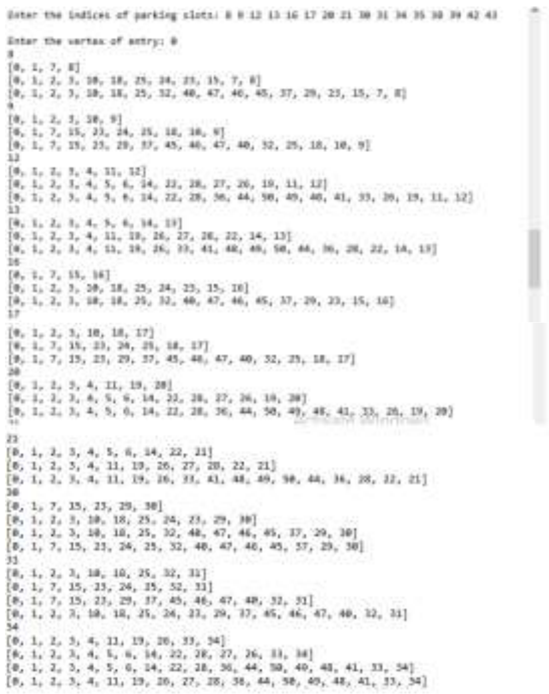


```
In [1]: runfile('C:/Users/HP/.spyder-py3/temp.py', wdir='C:/Users/HP/.spyder-py3')

Enter the number of vertices: 51

Enter the edges from vertex 0: 1

Enter the edges from vertex 1: 2 7

Enter the edges from vertex 2: 3

Enter the edges from vertex 3: 4 10

Enter the edges from vertex 4: 5 11

Enter the edges from vertex 5: 6

Enter the edges from vertex 6: 14

Enter the edges from vertex 7: 8 15

Enter the edges from vertex 8:

Enter the edges from vertex 9:

Enter the edges from vertex 10: 9 18

Enter the edges from vertex 11: 12 19

Enter the edges from vertex 12:

Enter the edges from vertex 13:
```

```
Enter the edges from vertex 25: 18 24 32

Enter the edges from vertex 26: 19 27 33

Enter the edges from vertex 27: 26 28

Enter the edges from vertex 28: 22 27 36

Enter the edges from vertex 29: 23 30 37

Enter the edges from vertex 30:

Enter the edges from vertex 31:

Enter the edges from vertex 32: 25 31 40

Enter the edges from vertex 33: 28 34 41

Enter the edges from vertex 34:

Enter the edges from vertex 35:

Enter the edges from vertex 36: 28 36 44
```

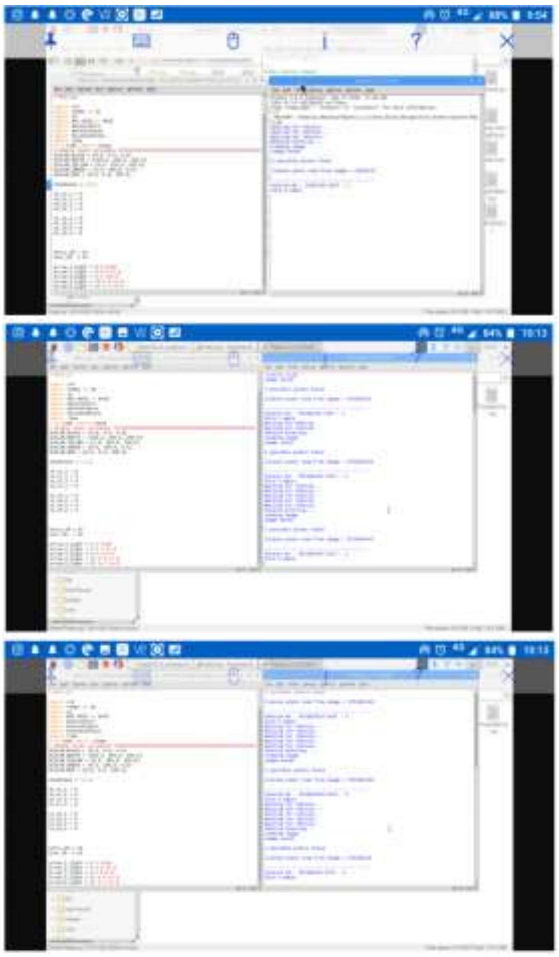
```
Enter the edges from vertex 37: 29 38 45

Enter the edges from vertex 38:

Enter the edges from vertex 39:

Enter the edges from vertex 40: 32 39 47

Enter the edges from vertex 41: 33 42 48

Enter the edges from vertex 42:

Enter the edges from vertex 43:

Enter the edges from vertex 44: 36 43 50

Enter the edges from vertex 45: 37 46

Enter the edges from vertex 46: 45 47

Enter the edges from vertex 47: 40 46

Enter the edges from vertex 48: 41 49

Enter the edges from vertex 49: 48 50

Enter the edges from vertex 50: 44 48

Enter the filename: graph

In [2]: runfile('C:/Users/HP/.spyder-py3/path.py', wdir

Enter the filename: graph
```

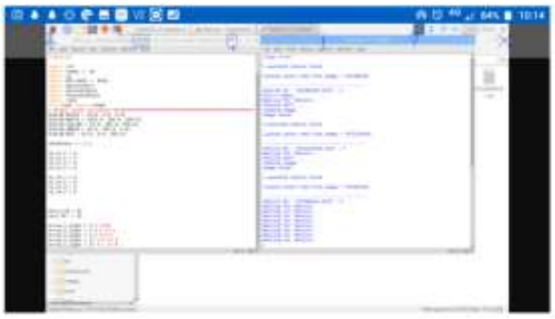
Shortest Path:



```
Enter the indices of parking slots: 8 9 12 13 16 17 20 21 30 31 34 35 38 39 42 43

Enter the vertex of entry: 0
8
[0, 1, 7, 8]
[0, 1, 2, 3, 10, 18, 25, 24, 23, 15, 7, 8]
[0, 1, 2, 3, 10, 18, 25, 32, 40, 47, 46, 45, 37, 29, 23, 15, 7, 8]
9
[0, 1, 2, 3, 10, 9]
[0, 1, 7, 15, 23, 24, 25, 18, 10, 9]
[0, 1, 7, 15, 23, 29, 37, 45, 46, 47, 40, 32, 25, 18, 10, 9]
12
[0, 1, 2, 3, 4, 11, 12]
[0, 1, 2, 3, 4, 5, 6, 14, 22, 28, 27, 26, 19, 11, 12]
[0, 1, 2, 3, 4, 5, 6, 14, 22, 28, 36, 44, 50, 49, 48, 41, 33, 26, 19, 11, 12]
13
[0, 1, 2, 3, 4, 5, 6, 14, 13]
[0, 1, 2, 3, 4, 11, 19, 26, 27, 28, 22, 14, 13]
[0, 1, 2, 3, 4, 11, 19, 26, 33, 41, 48, 49, 50, 44, 36, 28, 22, 14, 13]
16
[0, 1, 7, 15, 16]
[0, 1, 2, 3, 10, 18, 25, 24, 23, 15, 16]
[0, 1, 2, 3, 10, 18, 25, 32, 40, 47, 46, 45, 37, 29, 23, 15, 16]
17
[0, 1, 2, 3, 10, 18, 17]
[0, 1, 7, 15, 23, 24, 25, 18, 17]
[0, 1, 7, 15, 23, 29, 37, 45, 46, 47, 40, 32, 25, 18, 17]
20
[0, 1, 2, 3, 4, 11, 19, 20]
[0, 1, 2, 3, 4, 5, 6, 14, 22, 28, 27, 26, 19, 20]
[0, 1, 2, 3, 4, 5, 6, 14, 22, 28, 36, 44, 50, 49, 48, 41, 33, 26, 19, 20]
21
[0, 1, 2, 3, 4, 5, 6, 14, 22, 21]
[0, 1, 2, 3, 4, 11, 19, 26, 27, 28, 22, 21]
[0, 1, 2, 3, 4, 11, 19, 26, 33, 41, 48, 49, 50, 44, 36, 28, 22, 21]
30
[0, 1, 7, 15, 23, 29, 30]
[0, 1, 2, 3, 10, 18, 25, 24, 23, 29, 30]
[0, 1, 2, 3, 10, 18, 25, 32, 40, 47, 46, 45, 37, 29, 30]
[0, 1, 7, 15, 23, 24, 25, 32, 40, 47, 46, 45, 37, 29, 30]
31
[0, 1, 2, 3, 10, 18, 25, 32, 31]
[0, 1, 7, 15, 23, 24, 25, 32, 31]
[0, 1, 7, 15, 23, 29, 37, 45, 46, 47, 40, 32, 31]
[0, 1, 2, 3, 10, 18, 25, 24, 23, 29, 37, 45, 46, 47, 40, 32, 31]
34
[0, 1, 2, 3, 4, 11, 19, 26, 33, 34]
[0, 1, 2, 3, 4, 5, 6, 14, 22, 28, 27, 26, 33, 34]
[0, 1, 2, 3, 4, 5, 6, 14, 22, 28, 36, 44, 50, 49, 48, 41, 33, 34]
[0, 1, 2, 3, 4, 11, 19, 26, 27, 28, 36, 44, 50, 49, 48, 41, 33, 34]
```

Main Program




[illegible]

Data Storage

[illegible]

CONCLUSION AND FUTURE WORK

6.1 Conclusion

As conclusion, the objectives of this project have been achieved. The hassle in searching for available parking slots has been completely eliminated. The designed system could be applied everywhere due to its ease of usage and effectiveness. It facilitates the problems of urban livability , transportation mobility and environment sustainability. The Internet of Things integrates the hardware, software and network connectivity that enable objects to be sensed and remotely controlled across existing network. Such integration allows users to monitor available and unavailable parking spots that lead to improved efficiency, accuracy and economic benefit.

6.2 Future Work

The smart parking management system can be broadly applied for many future applications. Apart from its basic role of parking management of cars it can also be applied for plane and ship and fleet management. With the ever growing field of Internet of Things many concepts can be interfaced along with our system. For residential and domestic parking system the device can be interfaced with Home Automation

system which can control the various home appliances by sensing whether the user is arriving or departing from the parking space. For instance if the user has arrived then the module will sense the presence and will send information about arrival to the Home automation system which can accordingly switch on the selected appliances like HVAC (Heating Ventilation and Air Conditioning) units, Coffee maker, toaster, Wi-Fi routers etc. For commercial parking system the device can be interfaced with a module which can sense the arrival of employee and can switch on his computer and HVAC systems and accordingly switch off the appliances when the employee departs. The system can also be used to track the reporting and departing time of the employee for all days with precision thus acting as an attendance system.
Thus many such modules can be interfaced with our system to provide better facility, security, and optimization of electricity and resources with the principle idea of flawless fleet management system.

REFERENCES

[1]. Benenson, K. Martens and S. Birr., "Parkagent: An agent-based model of parking in the city", Comput. Environ. Urban Syst. Vol. 32, no. 6, pp.431–439, November 2008.

[2]. Antoine Bagula, "On the design of smart parking networks in the smart cities: An optimal sensor Placement Model",Sensors2015,15,15443-5467;doi:10.3390/s150715443.

[3]. L. Atzori, A. Iera, and G. Morabito, "The Internet of things: a survey," Computer Networks, vol. 54, no. 15, pp. 2787-2805, 2010.

[4]. Kaivan Karimi and Gary Atkinson, ―What the Internet of Things (IoT) Needs to Become a Reality‖, White Paper, FreeScale and ARM, 2013.

[5]. http://www.mdpi.com/journal/sensors Sensors 2014, 14, 22372-22393; doi:10.3390/s141222372

[6]. Choeychuen, K. Automatic parking lot mapping for available parking space detection. In Proceedings of the 5th International Conference on Knowledge and Smart Technology (KST), Chonburi, Thailand, 31 January–1 February 2013; pp. 117–121.

[7]. Keat, C.T.M.; Pradalier, C.; Laugier, C. Vehicle detection and car park mapping using laser scanner. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Edmonton, AB, Canada, 2–6 August 2005; pp. 2054–2060.

[8]. https://www.hackerearth.com/practice/algorithms/graphs/shortest-path-algorithms/tutorial/

[9]. https://algs4.cs.princeton.edu/44sp/

[10]. https://www.tutorialspoint.com/design_and_analysis_of_algorithms/design_and_analysis_of _ algorithms_shortest_paths.htm

[11]. https://en.wikipedia.org/wiki/Dijkstra%27s_algorithm#Algorithm

[12]. Christos-Nikolas E.Anagnostopoulos," License Plate Recognition;a Brief Tutorial",Intelligrnt Transportation Systems Magazine IEEE,Vol 6,Issue 1,pp 59-67,2014.

[13]. Shan Du,MahmoudIbrahim,Mohamoud Shehata and Wael Badawy,"Automatic License Plate Recognition(ALPR) ; A State-of-the-Art Review" IEEE Transactions on Circuits &Systems for Videos Technology,Vol 23,Issue 2,pp 311-325,IEEE2013

A.Durga Praveen Kumar "Automatic Smart Car Parking System Using Iot and Python" *Quest Journals Journal Of Software Engineering And Simulation, Vol. 06, No. 01, 2020, Pp. 01-13.*