



SMART CAR- PARKING SYSTEM USING IoT

Abstract: Nowadays congestion of traffic level increases with the increasing development of population rapidly. With respect to the amount of population, the utilization of personal vehicles also increased. Due to more use of cars the traffic congestion occurred on the road. Most of the people chooses personal vehicles than public transportation. It is very difficult and time consuming to find parking space in most metropolitan areas, commercial areas, especially during the rush hours. It is often costly in almost every big city in all over the world to find proper and secure parking space. The proposed project is a smart parking system that delivers information to people finding a parking space online. It overcomes unnecessary time consuming for finding the problem of parking space in parking areas. Hence, the website is provided by this project based system where users can view various parking areas and choose the space from available slots.

Keyword: Arduino; IoT; Parking Lot; Traffic Congestion; Ultrasonic Sensor.

1. INTRODUCTION

The recent growth in economy and due to the availability of low price cars in the market, an every average middle-class individual can afford a car, which is good thing, however the consequences of heavy traffic jams, pollution, less availability of roads and spot to drive the motor car. One of the important concerns, which is to be taken in accounting, is that problem of parking those vehicles [1]. Though, if there is space for parking the vehicle but so much time is squandered in finding that exact parking slot resulting in more fuel intake and not also environment friendly. It will be great deal if in some way we find out that the parking itself can provide the precise vacant position of parking slot then it'll be helpful not limited to the drivers also for the environment. Therefore, many innovations are created to find solution [2].

2. EXISTING SOLUTIONS

At present some countries have portals which users can gain information about parking areas via the internet. This system can give user the information about parking space, but it won't be able to give which parking slot is vacant and occupied. Hence, such system cannot smartly handle the issue. Car lifts

along with automated robotic system, which automatically takes car to a particular parking spot as soon as the car enters on a platform. This system can not be installed by medium scale shopping malls, movie theatres as it can cost them a huge amount. At many public places, the system only shows the availability but it cannot show the exact slot and path to the slot available. Hence, there is the need to smartly find the path to the vacant spot [2],[3].

3. MOTIVATION OF THIS PROJECT

The main motivation of this project is to reduce the traffic jam that occurs in the urban areas which are caused by vehicles searching for parking. In the newspapers, we saw many articles regarding the parking problem all over Dhaka City.

In Bangladesh we are still using the manual vehicle parking system that why we are facing problems like wastage of time and energy finding free space across the parking surface when we need to park our car which requires a good amount of fuel. We proposed an automated system where the parking ground will only open if it has free slots for parking. The user can also check it before arriving there by a website. It will save the time as well as reduce the gathering in front of parking area.

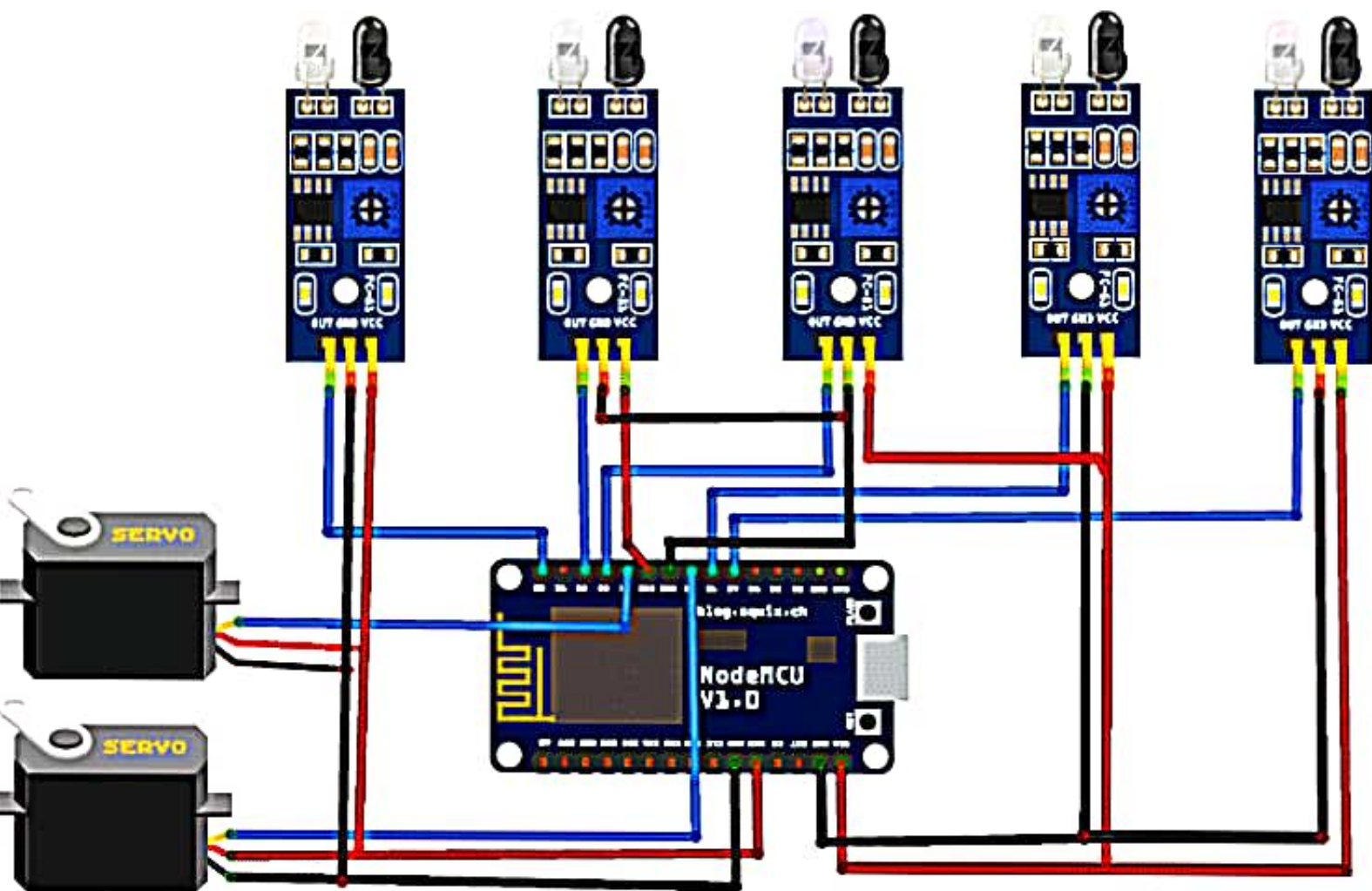
4. DESIGN MODEL

4.1 Block Diagram

The proposed system is the combination of smart

Cite this paper:

Saidur Rahman, Poly Bhounik, "IoT Based Smart Parking System", International Journal of Advances in Computer and Electronics Engineering, Vol. 4, No. 1, pp. 11-16, January 2019.



— □ ×

```
5EO5v`MM□5p5:I55[66] Connecting to DESKTOP
```

```
[1068] Connected to WiFi
```

[1068] IP: 192.168.137.51

[1068]


```

      _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
      / _ ) / / _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
      / _ / / / / / / _ \ / ' _ /
      / _ _ / _ \ _ , / _ / / _ _ \ _ \
      / _ _ / v0.6.7 on NodeMCU

```

```
[1146] Connecting to blynk-cloud.com:80
```

```
[1388] Ready (ping: 95ms).
```

☒ Autoscroll ☐ Show timestamp9600 baud 

9600 baud ▼ Clear output


```

//TECHATRONIC.COM
// BLYNK LIBRARY
// https://github.com/blynkkk/blynk-library
// ESP8266 LIBRARY
// https://github.com/ekstrand/ESP8266wifi
#define TRIGGER D0
#define ECHO D2
// NodeMCU Pin D0 > TRIGGER | Pin D2 > ECHO
#define BLYNK_PRINT Serial // Comment this out to disable
prints and save space
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
// You should get Auth Token in the Blynk App.
// Go to the Project Settings (nut icon).
char auth[] = "Whobi6tSCicbj4W654WdBeo7O4D6Ajw4"; //Auth
code sent via Email
// Your WiFi credentials.
// Set password to "" for open networks.
char ssid[] = "DESKTOP"; //Wifi name
char pass[] = "asdfghjkl"; //Wifi Password
void setup() {
  Serial.begin (9600);
  Blynk.begin(auth, ssid, pass);
  pinMode(TRIGGER, OUTPUT);
  pinMode(ECHO, INPUT);
  pinMode(BUILTIN_LED, OUTPUT);
}
void loop() {
  long duration, distance;
  digitalWrite(TRIGGER, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIGGER, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIGGER, LOW);
  duration = pulseIn(ECHO, HIGH);
  distance = (duration/2) / 29.1;
  if (distance <=200) {
    Blynk.virtualWrite(V0, 255);
  }
}

```

```
else {  
  Blynk.virtualWrite(V0, 0);  
}  
if (distance <= 35) {  
  Blynk.virtualWrite(V1, 255);  
}  
else {  
  Blynk.virtualWrite(V1, 0);  
}  
if (distance <= 30) {  
  Blynk.virtualWrite(V2, 255);  
}  
else {  
  Blynk.virtualWrite(V2, 0);  
}  
if (distance <= 25) {  
  Blynk.virtualWrite(V3, 255);  
}  
else {  
  Blynk.virtualWrite(V3, 0);  
}  
if (distance <= 20) {  
  Blynk.virtualWrite(V4, 255);  
}  
else {  
  Blynk.virtualWrite(V4, 0);  
}  
Serial.print(distance);  
Serial.println("Centimeter:");  
Blynk.virtualWrite(V5, distance);  
delay(200);  
Blynk.run();  
Serial.print(distance);  
Serial.println("Centimeter:");  
Blynk.virtualWrite(V6, distance);  
delay(100);  
Blynk.run();  
}
```

