SMART PARKING

Introduction to Smart Parking:

Smart parking is a modern technological solution aimed at revolutionizing the way we approach parking in urban and suburban environments. As cities around the world face increasing challenges related to traffic congestion, limited parking spaces, and environmentalconcerns, smart parking systems have emerged as a promising answer to these issues.

Smart parking leverages a combination of advanced technologies such as sensors, data analytics, mobile apps, and automation to provide a more efficient and user-friendly parkingexperience. The key idea behind smart parking is to optimize the utilization of parking spaces, reduce the time and fuel wasted in the search for parking, and enhance the overall urban mobility and quality of life.

In a smart parking system, sensors are installed in parking spaces to monitor their availabilityin real-time. This data is then relayed to users through mobile applications or digital displays,allowing them to easily find and reserve parking spots. Additionally, smart parking solutions often include features like automated payment systems and integration with navigation apps to guide drivers to available parking spaces.

The benefits of smart parking extend beyond individual convenience. By reducing traffic congestion and the environmental impact of circling for parking, these systems contribute toimproved air quality and reduced emissions. They also have the potential to boost revenue for municipalities or private operators through dynamic pricing and efficient enforcement.

Furthermore, smart parking aligns with broader urban development goals by enhancing safety, promoting sustainable transportation alternatives, and integrating with other aspects ofurban infrastructure, such as public transportation and sustainable for residents and visitors alike. As technology continues to advance, the future of smart parking promises even more innovativeand intelligent solutions to improve the way we park and move within urban environments.

Project Objectives

Optimize Parking Space Utilization: The primary goal of smart parking is to maximize theuse of available parking spaces. This involves reducing congestion and minimizing the time and fuel wasted by drivers searching for parking spots.

Reduce Traffic Congestion: By guiding drivers to available parking spaces and reducing thetime spent circling for a spot, smart parking systems aim to alleviate traffic congestion in urban areas. This can lead to reduced emissions and improved air quality.

Enhance User Convenience: Smart parking systems should make it easier for drivers to find, reserve, and pay for parking. Mobile apps, online booking, and real-time availabilityupdates contribute to a more convenient and user-friendly experience.

Improve Revenue Generation: Many smart parking initiatives are designed to increaserevenue for municipalities or private operators. This can be achieved through dynamic pricing, efficient enforcement, and improved space turnover.

Enhance Safety and Security: Smart parking solutions can incorporate security features likesurveillance cameras and emergency call buttons to enhance the safety of parking facilities.

Reduce Environmental Impact: Reducing the time vehicles spend idling and circling forparking spots can contribute to lower fuel consumption and emissions, thus helping to combat air pollution and climate change.

Promote Sustainable Transportation: Smart parking projects often aim to encourage the use of public transport, carpooling, and non-motorized modes of transportation by making these options more accessible and convenient.

Data Collection and Analysis: Gathering data on parking space utilization, traffic patterns,and user behavior is a key objective. Analyzing this data can help urban planners make informed decisions and optimize parking policies.

Enhance Accessibility: Smart parking should be designed to cater to the needs of all users,including those with disabilities, by providing accessible parking spaces and user-friendly features.

Economic Development: In certain cases, smart parking projects are expected to stimulate economic growth by making it easier for people to access businesses and attractions in urbanareas.

Enforcement and Compliance: Ensure that parking rules and regulations are effectively enforced through automated methods, such as license plate recognition or ticketing systems

IoT Devices :

1. ESP8266 NodeMCU
2. DC Servo Motor
3. IR Sensors
4. 16x2 LCD Display
5. Jumpers

Devices Setup:

Setting up an IoT project using an ESP8266 NodeMCU board, DC servo motor, IRsensors, a 16x2 I2C LCD display, and jumpers involves several steps. I'll provide an overview of how you can set up this project, but pleasenote that this is a complex project, and you may need to consult specific documentation and libraries for each component. Additionally, coding this project will require programming skills in platforms like Arduino IDE.

Gather the Required Components:

1. ESP8266 NodeMCU board.
2. DC servo motor.
3. IR sensors (for object detection).
4. 16x2 I2C LCD display.
5. Jumper wires and breadboard.

Power supply for the servo motor if needed

1.Connect the DC Servo Motor:

- Connect the positive (red) lead of the servo motor to the 5V output ofNodeMCU.
- Connect the negative (brown) lead of the servo motor to the GND ofNodeMCU.
- Connect the signal (orange/yellow) lead of the servo motor to a GPIO pin(e.g., D4).

2. Connect the IR Sensors:

- IR sensors are usually analog sensors. Connect the VCC and GND pins to 3.3V and GND on the NodeMCU.
- Connect the signal pin of the IR sensors to analog GPIO pins (e.g., A0 and A1).

3. Connect the 16x2 I2C LCD Display:

- Connect the SDA (data) and SCL (clock) pins of the I2C LCD display to the corresponding pins on the NodeMCU (D1 and D2 on the NodeMCU, respectively).
- Connect the VCC of the I2C display to 5V on NodeMCU and GND to GND.

4.**Power Supply:**

Make sure you have a suitable power supply for your servo motor, as the NodeMCU might not be able to provide enough power for it.

5.**Assemble and Test:**

Connect all the components, upload the code to the NodeMCU, and assemble the project Test each component and ensure that the system functions as expected

<u>Platform Development:</u>

1. Define Your Goals and Objectives:

- Clearly define the objectives and goals of your smart parking platform.Understand what problems you want to solve, whether it's reducing congestion, enhancing revenue generation, improving accessibility, or promoting sustainability.

2. Hardware Selection:

- Choose the appropriate hardware components, such as sensors (ultrasonic,infrared, camera-based), microcontrollers (like Raspberry Pi or Arduino), communication modules (Wi-Fi, LoRa, or cellular),

3. Central Server or Cloud Platform:

Develop a central server or cloud-based platform to collect, process, and store data from the sensors. Use a robust database system to manage real-time parking space availability.

4. User-Facing Mobile and Web Applications:

Create user-friendly mobile and web applications that allow users to:

- Find available parking spaces.
- Reserve parking spots in advance.
- Make payments for parking.
- Receive real-time updates on parking availability and guidance.

5. Payment Integration:

Integrate payment gateways into the mobile app for user convenience. This can include options for cashless payments, credit card payments, and mobilewallets.

6. Data Analytics and Prediction:

Implement data analytics to analyze historical and real-time parking data.This can help in predicting parking demand, optimizing pricing, and improving operational efficiency.

7. User Feedback and Support:

Include features for user feedback, customer support, and assistance in caseof issues or emergencies.

8. **Security and Access Control:**

Ensure the security of data and transactions. Implement user authenticationand access control features to prevent unauthorized use or tampering with the system.

9. **Real-time Displays and Signage:**

- Install displays at the parking facility entrances to guide drivers to availablespaces and provide real-time updates on space availability.

10. **Environmental Considerations:**

- For sustainability, consider incorporating features such as electric vehiclecharging stations, solar-powered components, and green infrastructure.

11. **Scalability and Flexibility:**

- Design your platform to be scalable, allowing for easy expansion to moreparking areas as needed.

12. **Regulatory Compliance:**

- Ensure your platform complies with local parking regulations, privacy laws, and other relevant regulations.

13. **Testing and Maintenance:**

- Thoroughly test the system to ensure its reliability and accuracy. Develop amaintenance plan for regular sensor and system checks.

14. **User Education and Onboarding:**

- Provide clear instructions to users on how to use the platform and make thetransition to smart parking smooth.

15. **Marketing and Adoption:**

- Promote your smart parking platform to attract users and encourage adoption. This may involve partnerships with local businesses and marketingcampaigns.

Program:

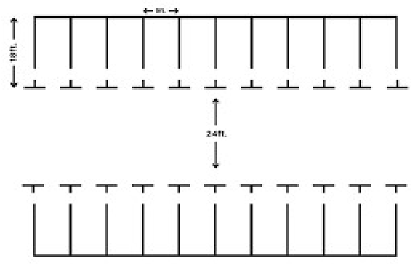
```
#include <ESP8266WiFi.h>

#include <Servo.h>

#include <LiquidCrystal_I2C.h>

#include <Wire.h>

#include <FirebaseArduino.h>

#define FIREBASE_HOST "smart-parking-7f5b6.firebaseio.com"                // the
project name address from firebase id

#define FIREBASE_AUTH
"suAkUQ4wXRPW7nA0zJQVsx3H2LmeBDPGmfTMBHCT"            // the secret
key generated from firebase

#define WIFI_SSID "CircuitDigest"                              // input your home
or public wifi name

#define WIFI_PASSWORD "circuitdigest101"                              //password
for Wifi

String Available = "";                                    //availability string

String fireAvailable = "";

LiquidCrystal_I2C lcd(0x27, 16, 2);        //i2c display address 27 and 16x2 lcd
display

Servo myservo;                          //servo as gate

Servo myservos;                               //servo as gate
```

```
int Empty;                              //available space integer
int allSpace = 90;
int countYes = 0;
int carEnter = D0;                      // entry sensor
int carExited = D4;                    //exi sensor
int TRIG = D7;                   //ultrasonic trig  pin
int ECHO = D8;                 // ultrasonic echo pin
int led = D3;                  // spot occupancy signal
int pos;
int pos1;
long duration, distance;
void setup() {
delay(1000);
  Serial.begin (9600);     // serial debugging
  Wire.begin(D2, D1);       // i2c start
  myservo.attach(D6);      // servo pin to D6
  myservos.attach(D5);       // servo pin to D5
  pinMode(TRIG, OUTPUT);      // trig pin as output
  pinMode(ECHO, INPUT);         // echo pin as input
  pinMode(led, OUTPUT);         // spot indication
  pinMode(carExited, INPUT);    // ir as input
  pinMode(carEnter, INPUT);     // ir as input
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);                              //try to
connect with wifi
  Serial.print("Connecting to ");
  Serial.print(WIFI_SSID);                        // display ssid
  while (WiFi.status() != WL_CONNECTED) {
    Serial.print(".");                        // if not connected print this
    delay(500);
  }
  Serial.println();
  Serial.print("Connected to ");
  Serial.println(WIFI_SSID);
  Serial.print("IP Address is : ");
  Serial.println(WiFi.localIP());                              //print local IP address
  Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH);        // begin firebase
authentication
  lcd.begin();                          //begin lcd
  lcd.home();
  lcd.setCursor(0, 0);                     // 0th row and 0thh column
  lcd.print("Smart Parking");
}
void loop() {
  digitalWrite(TRIG, LOW);        // make trig pin low
  delayMicroseconds(2);
```

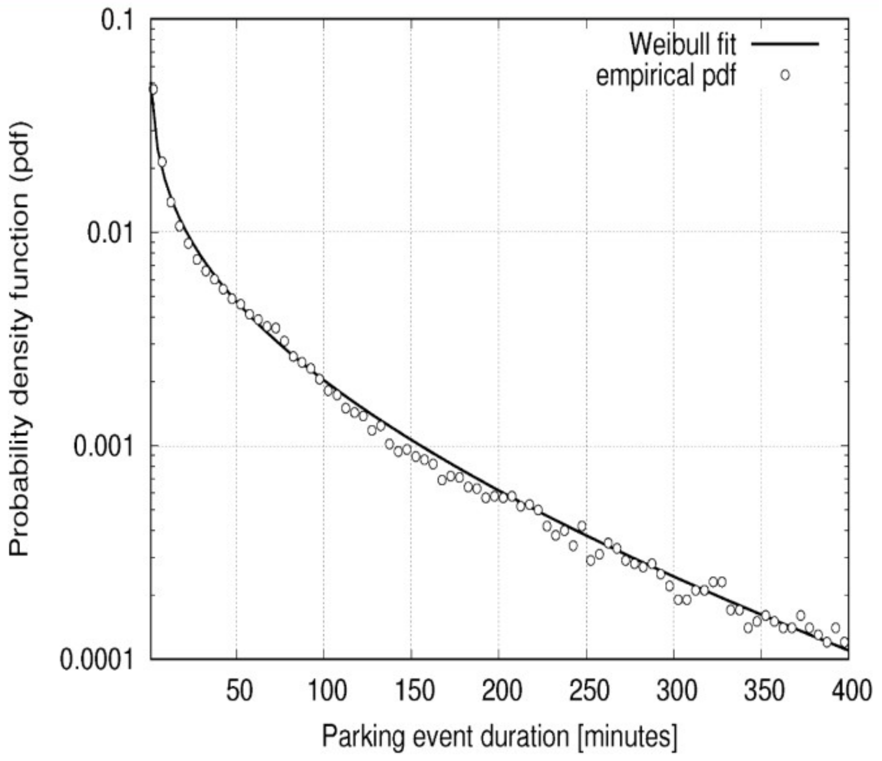
```
digitalWrite(TRIG, HIGH);        // make trig pin high
delayMicroseconds(10);
digitalWrite(TRIG, LOW);
duration = pulseIn(ECHO, HIGH);
distance = (duration / 2) / 29.1;      // take distance in cm
  Serial.print("Centimeter: ");
  Serial.println(distance);
int carEntry = digitalRead(carEnter);      // read ir input
if (carEntry == HIGH) {                     // if high then count and send data
  countYes++;                              //increment count
  Serial.print("Car Entered = " ); Serial.println(countYes );
  lcd.setCursor(0, 1);
  lcd.print("Car Entered");
  for (pos = 140; pos >= 45; pos -= 1) {        // change servo position
    myservos.write(pos);
    delay(5);
  }
  delay(2000);
  for (pos = 45; pos <= 140; pos += 1) {      // change servo position
    // in steps of 1 degree
    myservos.write(pos);
    delay(5);
  }
```

```
    Firebase.pushString("/Parking Status/", fireAvailable );     // send string to
firebase
    lcd.clear();
  }
  int carExit = digitalRead(carExited);               //read exit ir sensor
  if (carExit == HIGH) {                             //if high then count and send
    countYes--;                                    //decrement count
    Serial.print("Car Exited = " ); Serial.println(countYes);
    lcd.setCursor(0, 1);
    lcd.print("Car Exited");
    for (pos1 = 140; pos1 >= 45; pos1 -= 1) {          // change servo position
      myservo.write(pos1);
      delay(5);
    }
    delay(2000);
    for (pos1 = 45; pos1 <= 140; pos1 += 1) {             // change servo position
      // in steps of 1 degree
      myservo.write(pos1);
      delay(5);
    }
    Firebase.pushString("/Parking Status/", fireAvailable );  // send string to firebase
    lcd.clear();
  }
  if (distance < 6) {                    //if distance is less than 6cm then on led
        Serial.println("Occupied ");
    digitalWrite(led, HIGH);
  }
  if (distance > 6) {                      //if distance is greater than 6cm then off led
        Serial.println("Available ");
    digitalWrite(led, LOW);
  }
  Empty = allSpace - countYes;          //calculate available data
  Available = String("Available= ") + String(Empty) + String("/") +
String(allSpace);       // convert the int to string
  fireAvailable = String("Available=") + String(Empty) + String("/") +
String(allSpace);
  lcd.setCursor(0, 0);
  lcd.print(Available);                 //print available data to lcd
}
```

Diagram:

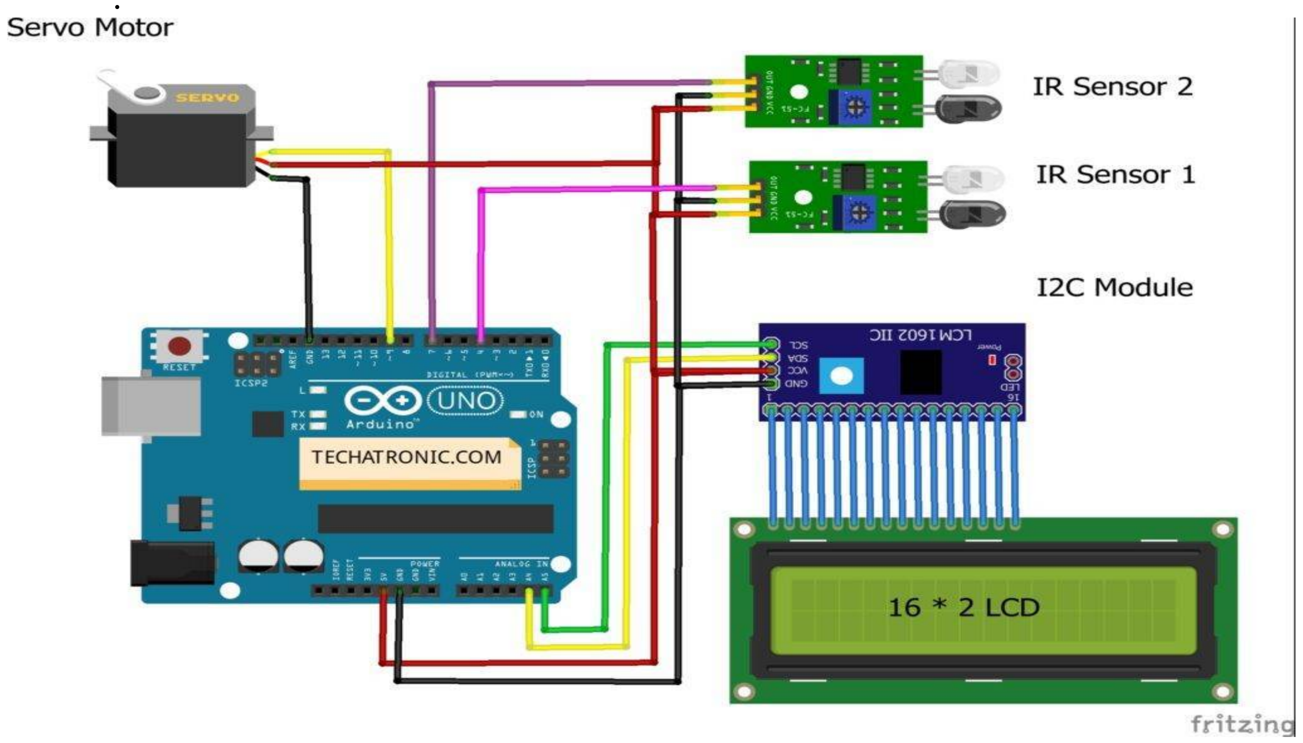


18ft.
24ft.



Weibull fit
empirical pdf
Probability density function (pdf)
0.1
0.01
0.001
0.0001
50
100
150
200
250
300
350
400
Parking event duration [minutes]

.



Project in Detail:

1. Project Overview:

- Start with an executive summary that provides a brief description of the smart parking project. Explain the purpose and expected benefits of the system, such as reduced congestion, enhanced user experience, or increasedrevenue generation.

2. Project Objectives:

- Clearly define the goals and objectives of the project. Be specific about whatyou aim to achieve, such as the number of parking spaces to be monitored, the reduction in search time, or the revenue targets.

3. **Hardware and Sensors:**

- Specify the hardware components and sensors to be used. Detail the types ofsensors (e.g., ultrasonic, infrared, camera-based), microcontrollers or platforms (e.g., ESP8266, Raspberry Pi), communication modules, and any displays or signage.

4. **Network Infrastructure:**

- Describe the network and communication infrastructure. Specify whetheryou'll use Wi-Fi, LoRa, cellular networks, or a combination to connect sensors to the central server or cloud platform.

5. **Central Server or Cloud Platform:**

- Provide details on the central server or cloud platform used to collect, process, and store data. Mention the database system and technologies you'lluse.

6. **User Interfaces:**

- Explain the user interfaces, including mobile apps for users to find and reserve parking spaces, make payments, and receive real-time updates. Also,describe any web-based dashboards for administrators.

7. **Payment Integration:**

- Specify the payment gateways and options for users, such as cashlesspayments, credit card payments, and mobile wallets.

8. **Data Analytics and Prediction:**

- Outline how you plan to use data analytics for optimizing parking, pricing, and improving operational efficiency. Describe any machine learning or predictive analytics techniques.

9. **Environmental Considerations:**

- Detail any sustainable features like electric vehicle charging stations, solarpower, or green infrastructure.

10. Regulatory Compliance:

- Explain how your system will comply with local parking regulations,privacy laws, and other relevant regulations.

11. System Testing and Maintenance:

- Define a testing strategy to ensure system reliability and accuracy. Plan forregular maintenance of sensors and the entire system.

12. User Education and Onboarding:

- Outline how you will educate users on using the platform and ensure asmooth transition to smart parking.

13. Marketing and Adoption:

- Describe marketing strategies to promote the platform and encourage useradoption. Include partnerships and marketing campaigns.

14. Project Timeline:

- Create a detailed project timeline that specifies milestones, deadlines, anddependencies between tasks.

15. Budget and Resources:

Estimate the budget required for the project, including hardware costs,software development, personnel, and ongoing operational expenses.

Conclusion:

IoT based smart parking system significantly enhance parking efficiency by providing real-time information on parking space availability. This minimizes the time and fuel wasted in searching for parking spaces, reducing traffic congestion and carbon emission.