Cyber Physical Systems 2016

Project Report

ANIK BARUA
UTU Student ID: 511580

Automatic Parking Gate

The project is about to make an Automatic Parking Gate for the vehicles. The concept is that, a UltraSonic Ranger will detect the distance of the incoming car and when the car will come closer to the parking gate; the parking gate will be opened for a certain period of time and then will be automatically closed again. The parking gate will be controlled using a Servo. Near the parking gate, there will be Green and Red LED light so that the driver can follow the signal and it will go when the Green light is on and otherwise it will wait when the light is Red. At the same time, a Buzzer will keep beeping in different speed when the gate is opened and closed. Also, there will be wireless connection and status of the parking gate will be calculated and the data will be sent to computer or mobile using this wireless connection and I have used the Bluetooth for this purpose.

Required Instruments:

1. Arduino UNO
2. Ultrasonic Range Finder
3. Grove Base Shield
4. Grove – Servo
5. Grove – Buzzer
6. Grove – LED (Green, Red)
7. Wireless Bluetooth Serial Transceiver
8. 6 Connecting wires
9. Bluetooth

Source Code:

```
#include "Arduino.h"
#include <Servo.h>
#include <SoftwareSerial.h>

#define GREEN_LED_PIN 2
#define RED_LED_PIN 5
#define BUZZER_PIN 4
#define SERVO_PIN 6

Servo myServo;
int servoPos = 0;

SoftwareSerial blueTooth(2, 3);

class Ultrasonic
{
     public:
          Ultrasonic(int pin);
```

```
        void DistanceMeasure(void);
          long microsecondsToCentimeters(void);
          long microsecondsToInches(void);
     private:
          int ULTRASONIC_RANGER_PIN;//pin number of Arduino that
is connected with SIG pin of Ultrasonic Ranger.
               long duration;// the Pulse time received;
};


Ultrasonic::Ultrasonic(int pin)
{
     ULTRASONIC_RANGER_PIN = pin;
}


/*Begin the detection and get the pulse back signal*/
void Ultrasonic::DistanceMeasure(void)
{
    pinMode(ULTRASONIC_RANGER_PIN, OUTPUT);
     digitalWrite(ULTRASONIC_RANGER_PIN, LOW);
     delayMicroseconds(2);
     digitalWrite(ULTRASONIC_RANGER_PIN, HIGH);
     delayMicroseconds(5);
```

```
    digitalWrite(ULTRASONIC_RANGER_PIN,LOW);
    pinMode(ULTRASONIC_RANGER_PIN,INPUT);
    duration = pulseIn(ULTRASONIC_RANGER_PIN,HIGH);
}

/*The measured distance from the range 0 to 400 Centimeters*/
long Ultrasonic::microsecondsToCentimeters(void)
{
    return duration/29/2;
}

/*The measured distance from the range 0 to 157 Inches*/
long Ultrasonic::microsecondsToInches(void)
{
    return duration/74/2;
}

Ultrasonic ultrasonic(7);
```

```
void setup()
{
     Serial.begin(9600);
        blueTooth.begin(115200);



        pinMode(GREEN_LED_PIN, OUTPUT);
        pinMode(RED_LED_PIN, OUTPUT);
        pinMode(BUZZER_PIN, OUTPUT);
        myServo.attach(SERVO_PIN);


 }


void loop()
{
        long RangeInInches;
     long RangeInCentimeters;
     ultrasonic.DistanceMeasure();//  get  the  current  signal
time;
        RangeInInches                                          =
ultrasonic.microsecondsToInches();//convert the time to inches;
```

```
    RangeInCentimeters                                          =
ultrasonic.microsecondsToCentimeters();//convert   the   time   to
centimeters
    Serial.println("The distance of the vehicle is: ");
    // Serial.print(RangeInInches);//0~157 inches
    // Serial.println(" inch");
    Serial.print(RangeInCentimeters);//0~400cm
    Serial.println(" cm");
    delay(1000);



       digitalWrite(GREEN_LED_PIN, LOW);     // turn the LED off
by making the voltage LOW
       digitalWrite(RED_LED_PIN, HIGH);     // turn the LED on
(HIGH is the voltage level)


       if (RangeInCentimeters < 3){


           for(servoPos=0; servoPos <=90; servoPos++){
           myServo.write(servoPos);
           delay(15); // delay of Servo, means how fast the
gate will open
           }
```

```
            digitalWrite(GREEN_LED_PIN, HIGH);   // turn the LED
on (HIGH is the voltage level)
            digitalWrite(RED_LED_PIN, LOW);     // turn the LED
off by making the voltage LOW

            Serial.println("Parking Gate is open");
            blueTooth.println("Parking Gate is open");


            for (int i =0; i<6;i++){
            tone(BUZZER_PIN, 1000); // Send 1KHz sound signal...
            delay(500);        // ...for 1 sec
            noTone(BUZZER_PIN);     // Stop sound...
            delay(500);        // ...for 1sec
            }
           // delay(5000); // the gate will be opened for 5
second

            digitalWrite(GREEN_LED_PIN, LOW);    // turn the LED
off by making the voltage LOW
            digitalWrite(RED_LED_PIN, HIGH);    // turn the LED
on (HIGH is the voltage level)
```

```
            for(servoPos = 90; servoPos >=0; servoPos--){
            myServo.write(servoPos);
            delay(30); // delay of Servo, means how slow the
gate will close
            }
        }

        Serial.println("Parking Gate is closed");
        blueTooth.println("Parking Gate is closed");



}
```

Advantages:

1. Automatic Parking Gate Design and Implementation using simpler and easier way
2. Can correctly measure the distance and control to gate
3. Energy and cost efficient
4. Avoid complex design and devices
5. Signal by Green and Red LED and also Buzzer sound also included
6. The status of Parking gate can be seen by Smart Phone using BlueTooth

Limitation and Future Plan:

1. Using 2 Ultrasonic Ranger will make it more efficient and accurate.