Smart counting system for parking area

~ Lê Hoàng Phúc & Nguyễn Tuấn Minh ~

Describe

We can apply this system in managing and informing the number of slots in the parking area automatically.

By using sensors to detect objects getting in and out of the parking area and increase or decrease the number of available or taken slots respectively. This system can process both in and out gates at the same time. Here, we demo this idea with Arduino kits.

Hardware Required

- Arduino or Genuino Board	(x1)
- Servo SG90	(x2)
- LCD1602	(x2)
- Potentiometer	(x1)
- Ultrasonic distance sensor	(x2)
- 220 ohm resistors	
- Hook-up wires	
- Breadboard	

Circuit

For LCD screens:

2 LCD screens must connect to the same pin in order to display the same information (use breadboard)

- LCD RS pin to digital pin 12

- LCD E pin to digital pin 11
- LCD D4 pin to digital pin 5
- LCD D5 pin to digital pin 4
- LCD D6 pin to digital pin 3
- LCD D7 pin to digital pin 2
- LCD VO pin connect to middle pin of potentiometer
- LCD VSS, RW, K pin to GND
- LCD VDD pin to +5V
- LCD A pin to +5V pin through a 220 ohm resistor

For servo:

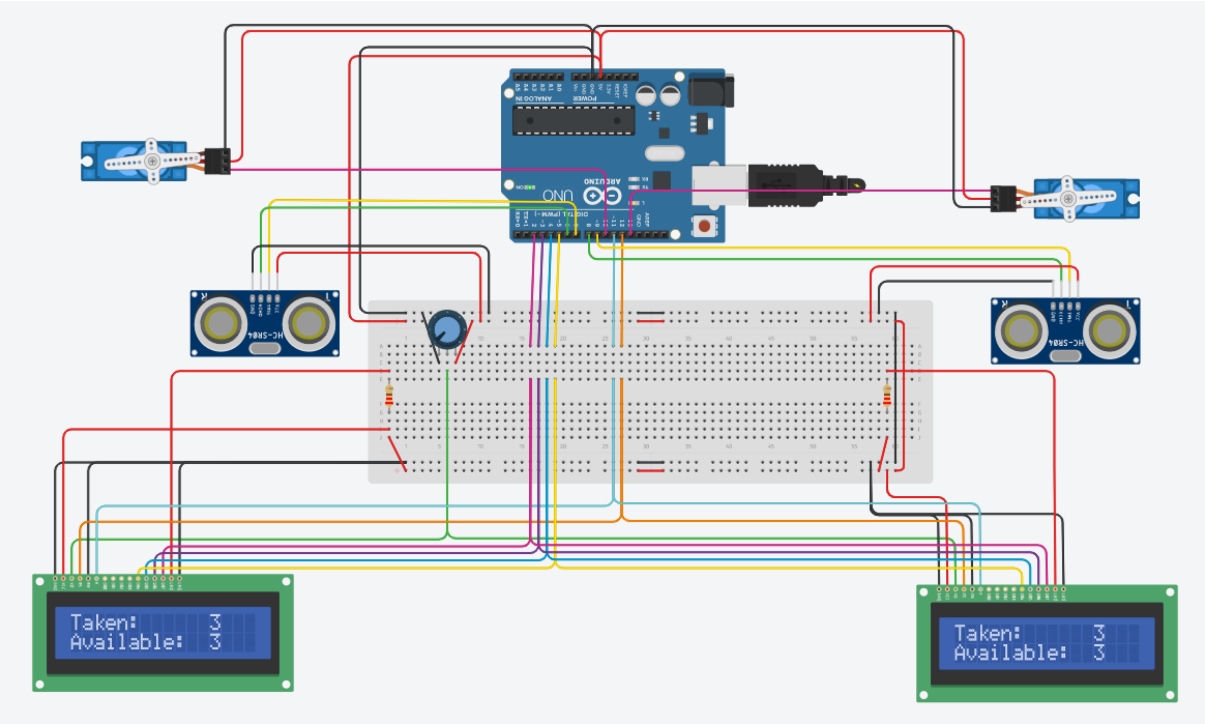
- For enter door:
    - Signal pin to 13
    - Power pin to +5V
    - Ground pin to GND
- For exit door:
    - Signal pin to 10
    - Power pin to +5V
    - Ground pin to GND

For ultrasonic distance sensor:

- For enter door:
    - Trigger pin to 9
    - Echo pin to 8
    - Vcc pin to +5V
    - GND pin to GND
- For exit door:
    - Trigger pin to 7
    - Echo pin to 6
    - Vcc pin to +5V
    - GND pin to GND

For potentiometer:

- Left pin to GND
- Right pin to +5V
- Middle pin to VO pin of LCD screens



Code

```
#include <Servo.h>
#include <LiquidCrystal.h>

Servo myservo_in;
Servo myservo_out;

const unsigned int TRIG_PIN_IN = 9;
const unsigned int ECHO_PIN_IN = 8;
const unsigned int servoPin_in = 13;
const unsigned int TRIG_PIN_OUT = 7;
const unsigned int ECHO_PIN_OUT = 6;
const unsigned int servoPin_out = 10;

int pos_in = 0;
int lastPos_in = 0;
int pos_out = 0;
int lastPos_out = 0;
int slots = 3;  // Available slots
int take = 3;  // Taken slots
long lastTime_in = 0;
long lastTime_out = 0;
float distance_in = 0;
float distance_out = 0;
```

```
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

// This function will return the distance between object and sensor
float getDistance(int TRIG_PIN, int ECHO_PIN) {
  digitalWrite(TRIG_PIN, LOW);
  delayMicroseconds(2);
  digitalWrite(TRIG_PIN, HIGH);
  delayMicroseconds(10);
  digitalWrite(TRIG_PIN, LOW);
  return pulseIn(ECHO_PIN, HIGH) / 29 / 2;
}

void setup() {
  // set up the lcd's number of columns and rows:
  lcd.begin(16, 2);
  pinMode(TRIG_PIN_IN, OUTPUT);
  pinMode(ECHO_PIN_IN, INPUT);
  myservo_in.attach(servoPin_in);
  pinMode(TRIG_PIN_OUT, OUTPUT);
  pinMode(ECHO_PIN_OUT, INPUT);
  myservo_out.attach(servoPin_out);
  Serial.begin(9600);
  myservo_in.write(0);
  myservo_out.write(0);
}

void loop() {
  distance_in = getDistance(TRIG_PIN_IN, ECHO_PIN_IN);
  distance_out = getDistance(TRIG_PIN_OUT, ECHO_PIN_OUT);

  Serial.println(distance_out);
  Serial.println(distance_in);
  Serial.println("");
  lcd.setCursor(0, 0);

  if (distance_in == 0 || distance_out == 0) {
    lcd.clear();
    lcd.print("Sensor error");
    delay(1000);
    lcd.clear();
  } else {
    lcd.print("Taken:");
    lcd.setCursor(12, 0);
    lcd.print(take);
    lcd.setCursor(0, 1);
    lcd.print("Available:");
    lcd.setCursor(12, 1);
    lcd.print(slots);
    // for in
    if (distance_in <= 50) {
      pos_in = 90;
      lastTime_in = millis();
    } else {
      // setting delay for closing the door after object move out of sensor range
      if ( millis() - lastTime_in >= 5000 && pos_in == 90) {
        pos_in = 0;
      }
    }

    // If no available slot left, door will not open
    if (slots > 0) {
      if (pos_in == 0 && lastPos_in == 90) {
        slots -= 1;
        take += 1;
      }
      myservo_in.write(pos_in);
    }

    lastPos_in = pos_in;
    Serial.println(pos_in);

    // for out
    if (distance_out <= 50) {
      pos_out = 90;
      lastTime_out = millis();
    } else {
      // setting delay for closing the door after object move out of sensor range
      if ( millis() - lastTime_out >= 5000 && pos_out == 90) {
        pos_out = 0;
      }
    }

    // If all slots are free, door will not open, for security
    if (take > 0) {
      if (pos_out == 0 && lastPos_out == 90) {
        slots += 1;
        take -= 1;
      }
      myservo_out.write(pos_out);
    }
    lastPos_out = pos_out;
  }
    Serial.println(pos_out);
    delay(1000);    // intentionally limit the scanning cycle to prevent error
}
```