Private Automated Door Access and Monitoring System Using Arduino with Keypad Override

[Submitted in May 3th, 2025]

I. Project Description / Abstract

This project is an Arduino-based automated door access and monitoring system that uses a servo motor, ultrasonic sensor, keypad, LCD, LEDs, and a buzzer. It controls entry based on a 10-person capacity limit. The ultrasonic sensor detects approach, opening the door if space is available. A keypad allows password-based override access. The LCD shows entry count, while LEDs and a buzzer indicate status—green for access, red with sound when full. The system showcases effective use of embedded components for crowd control and secure access.

II. Introduction

Motivation:

In public and private facilities, managing the number of people entering a room is critical for safety, security, and operational efficiency. Manual monitoring can be inefficient, prone to error, or require constant human supervision. This project aims to automate the access control process using low-cost, accessible components.

Problem-Statement:

Conventional door entry systems often lack real-time monitoring and capacity control. In scenarios where space or safety regulations limit the number of occupants, there's a need for an automated system that can prevent overcapacity and provide secure access management.

Proposed-Solution:

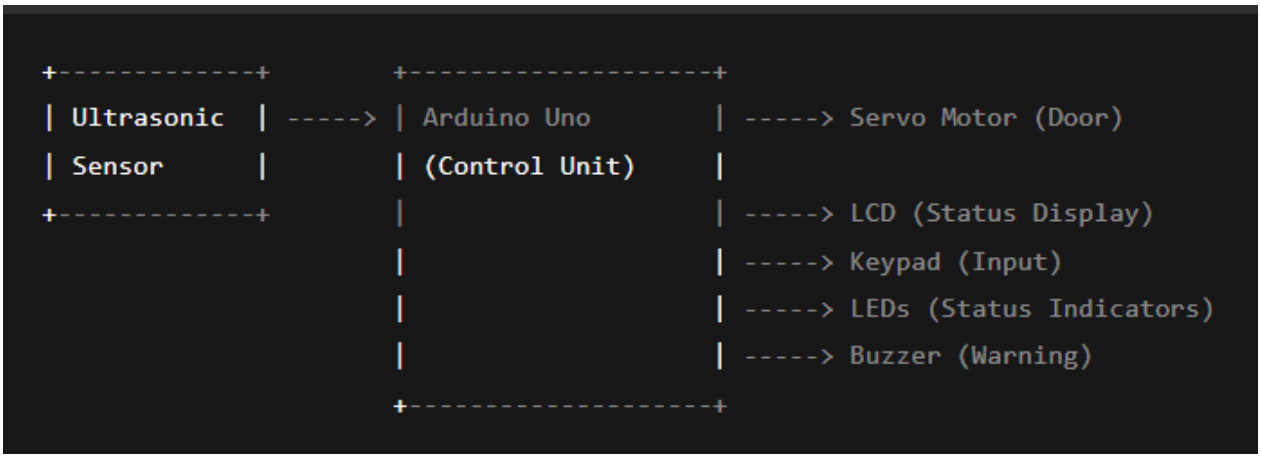
The project introduces an automated door system using an Arduino microcontroller. It monitors entries via an ultrasonic sensor and limits access based on a pre-defined capacity. A keypad allows password-based manual override, while an LCD provides real-time status updates. The use of LEDs and a buzzer enhances user feedback, ensuring both convenience and safety.

III. Objectives

- Develop a smart door system that automatically allows entry based on ultrasonic detection and limits the number of people entering.
- Implement a secure manual override mechanism using a keypad and password.
- Display real-time entry count and system status on an LCD.
- Use visual (LEDs) and auditory (buzzer) signals for feedback.
- Measurable Outcomes:
    - Accurate detection of individuals within 30 cm using the ultrasonic sensor.
    - Reliable capacity limit enforcement (maximum of 10 entries).
    - Password override functionality working with correct input.
    - LCD updates entry count correctly.
    - LED and buzzer respond according to full/available status.

IV. System Architecture

Block Diagram



Component Descriptions

- Arduino Uno: The microcontroller acts as the brain of the system, handling inputs and outputs.
- Ultrasonic Sensor (HC-SR04): Detects presence within a short range (30 cm) to trigger the door.
- Servo Motor: Simulates door opening and closing.
- LCD Display (16x2): Displays system status, entry count, and password prompts.

- Keypad (4x3): Allows manual password input for override access.
- LEDs: Indicate whether entry is allowed (green) or denied (red).
- Buzzer: Emits sound when access is denied due to full capacity or incorrect password.

Interconnections and Control Signals

- Digital pins for servo, LEDs, trigger/echo of ultrasonic.
- Analog pins used for keypad rows/columns and buzzer.
- LCD connected using digital pins in 4-bit mode.

V. Design and Implementation

Hardware Design

- Schematic
    - Vcc and GND lines shared.
    - Trigger/Echo: Pins 9 and 10.
    - Servo: Pin 6.
    - LCD: Pins 12, 11, 5, 4, 3, 2.
    - Keypad: Pins A1–A5, 7.
    - LEDs and buzzer: Pins 8, 13, A0.

Hardware Components

- Arduino Uno
- HC-SR04 Ultrasonic Sensor
- SG90 Servo Motor
- 16x2 LCD Display
- 4x3 Matrix Keypad
- Green and Red LEDs
- Passive Buzzer
- Breadboard and Jumper Wires

Software Design

- Written in Arduino C++
- High-level architecture includes:
    - Input handlers (ultrasonic and keypad)
    - Output modules (servo, LCD, LED, buzzer)
    - Logic module (counting, access control)

- Code:

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

// LCD pins: RS, E, D4, D5, D6, D7
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

// Ultrasonic sensor
const int trigPin = 9;
const int echoPin = 10;

// Outputs
const int servoPin = 6;
const int greenLedPin = 8;
const int redLedPin = 13;
const int buzzerPin = A0;

// State variables
long duration;
float distance;
int openCount = 0;
bool isFull = false;

Servo doorServo;

// Keypad setup (4x3)
const byte ROWS = 4;
const byte COLS = 3;

char keys[ROWS][COLS] = {
```

```
  {'1','2','3'},
  {'4','5','6'},
  {'7','8','9'},
  {'*','0','#'}
};

// Keypad wiring to available pins
byte rowPins[ROWS] = {A1, A2, A3, A4};
byte colPins[COLS] = {A5, 7, A0};

Keypad keypad = Keypad(makeKeymap(keys), rowPins, colPins, ROWS, COLS);
String enteredPassword = "";
const String overridePassword = "1245";

void setup() {
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(greenLedPin, OUTPUT);
  pinMode(redLedPin, OUTPUT);
  pinMode(buzzerPin, OUTPUT);

  digitalWrite(greenLedPin, LOW);
  digitalWrite(redLedPin, LOW);
  digitalWrite(buzzerPin, LOW);

  doorServo.attach(servoPin);
  doorServo.write(0); // Closed

  lcd.begin(16, 2);
  lcd.setCursor(0, 0);
  lcd.print("Door Counter:");
```

```
  lcd.setCursor(0, 1);
  lcd.print("Opened: 0 times");

  Serial.begin(9600);
}

void loop() {
  // Keypad password input
  char key = keypad.getKey();
  if (key) {
    if (key == '*') {
      if (enteredPassword == overridePassword) {
        lcd.setCursor(0, 1);
        lcd.print("Override Access   ");
        openDoor();
        delay(1000);
        closeDoor();
      } else {
        lcd.setCursor(0, 1);
        lcd.print("Wrong Password!   ");
        digitalWrite(buzzerPin, HIGH);
        delay(1000);
        digitalWrite(buzzerPin, LOW);
      }
      enteredPassword = "";
    } else if (key == '#') {
      enteredPassword = "";  // cancel
    } else {
      enteredPassword += key;
      lcd.setCursor(0, 1);
      lcd.print("Pass: ");
```

```
    lcd.print(enteredPassword);
    lcd.print("        ");
  }
}

// Trigger ultrasonic sensor
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);

duration = pulseIn(echoPin, HIGH, 30000);
distance = (duration * 0.034) / 2;

Serial.print("Distance: ");
Serial.print(distance);
Serial.println(" cm");

if (distance > 0 && distance < 30) {
  if (openCount < 10) {
  openCount++;
    lcd.setCursor(0, 1);
    lcd.print("Opened: ");
    lcd.print(openCount);
    lcd.print(" times ");
    openDoor();
    delay(1000); // open for 1s
    closeDoor();
    isFull = (openCount >= 10);
  } else {
```

```
    // Entry denied due to full capacity
    lcd.setCursor(0, 1);
    lcd.print(" FULL - NO ENTRY ");
    digitalWrite(greenLedPin, LOW);
    digitalWrite(redLedPin, HIGH);
    digitalWrite(buzzerPin, HIGH);
    delay(1000);
    digitalWrite(buzzerPin, LOW);
   }
  }

  // LED status
  if (openCount >= 10) {
   isFull = true;
   digitalWrite(greenLedPin, LOW);
   digitalWrite(redLedPin, HIGH);
  } else {
   isFull = false;
   digitalWrite(redLedPin, LOW);
   digitalWrite(greenLedPin, HIGH);
  }

  delay(100);
}

void openDoor() {
  doorServo.write(90);
}

void closeDoor() {
  doorServo.write(0);
```

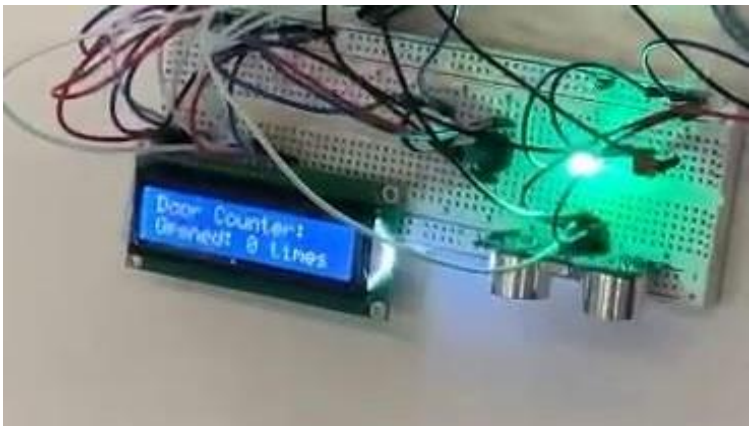
}

Integration

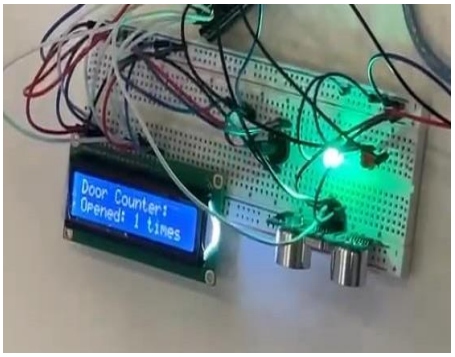
- Sensor input: Ultrasonic uses pulseIn() for distance calculation.
- Servo: Controlled via PWM.
- LCD and Keypad: Use standard Arduino libraries.

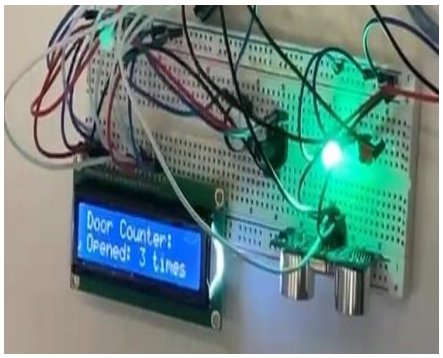
Steps to Use the System (with Screenshots)

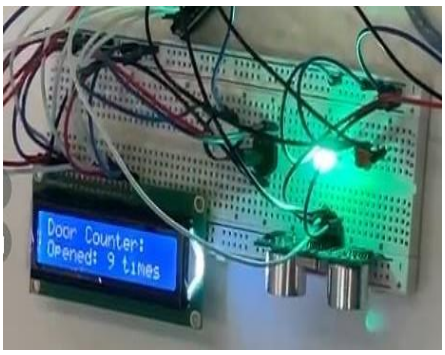
Startup 1: LCD shows "Door Counter" and "Opened: 0 times".



Person Approaches 2: Within 30 cm, door opens, count increases.







When full (count >= 10) 3: LCD shows "FULL - NO ENTRY", buzzer sounds.

Manual Override 4: Press override password then '*'. The door opens regardless of capacity.

- we will enter a password

- after we enter it will appear (Override Access)

VI. Results and Discussion

- The system successfully tracked entries and restricted further access once the count reached 10.
- Password override feature worked reliably for authorized users.
- Real-time feedback using the LCD, LEDs, and buzzer enhanced usability.
- Analysis: All initial objectives were met. Some minor delays in sensor response were observed but did not impact performance significantly. Pin sharing issues were resolved through better wiring and conditional delays.

VII. Conclusion

This project demonstrated a practical and efficient way to automate door access and manage capacity limits using Arduino. The integration of sensors, actuators, and a simple user interface proved effective in delivering a real-world solution for access control. Key lessons learned include the importance of power management, pin assignment, and debouncing user input from the keypad. Future improvements could involve adding data logging, IoT connectivity, or facial recognition for advanced security.