תרגיל בית מעשי C: ארדואינו מפעילים וחיישנים

קבוצה 26

עומר פרידמן, טליה סעדה, יובל לוז, נעם שירי

תיאור ומטרת המערכת

על מנת לעודד אימון גופני עצמאי ולספק חוויית תרגול מדויקת ואינטראקטיבית, החלטנו להקים מערכת חכמה לספירת שכיבות סמיכה מבוססת Arduino. מטרת המערכת היא לאפשר למשתמש להגדיר מטרה של מספר שכיבות סמיכה, לבצע את התרגיל בצורה נכונה ומבוקרת, ולקבל פידבק מיידי לאורך כל התהליך.

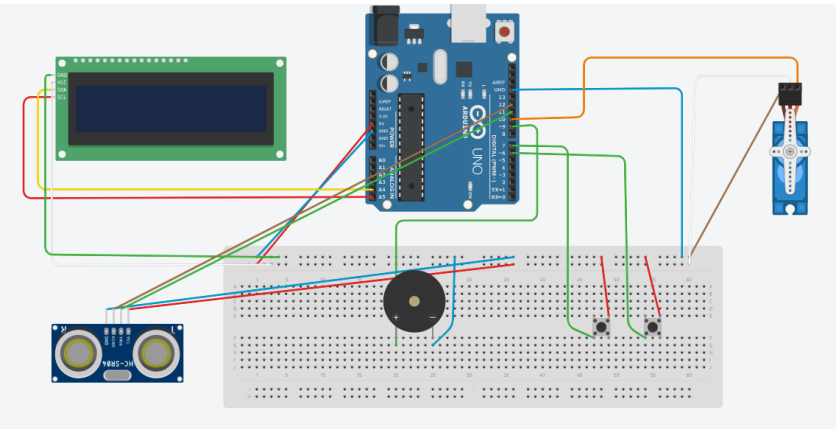
המערכת תפעל בצורה הבאה: המשתמש לוחץ על כפתור ומתחיל בבחירת מטרה (5, 10, 15, 20 שכיבות סמיכה). לאחר אישור, הוא מתרחק מהחיישן כדי להתחיל את הסט. חיישן המרחק המונח על הרצפה מזהה תנועה של ירידה מתחת ל-10 ס"מ ועלייה מעל 30 ס"מ – פעולה זו נחשבת לחזרה מלאה.

בכל חזרה מוצלחת: ישמע צפצוף קצר באמצעות באזר ויוצג מספר החזרות על מסך LCD.

כאשר המשתמש מגיע ליעד שהוגדר: מנוע סרוו יבצע סיבוב קצר כסימן לסיום מוצלח של הסט ותוצג הודעת עידוד במסך.

תיאור התוכן

הפרויקט מומש באמצעות בקר Arduino UNO וכלל את הרכיבים הבאים: חיישן מרחק אולטרסוני HC-SR04 למדידת תנועת המשתמש, מסך LCD בתקשורת I2C להצגת מידע בזמן אמת, מנוע סרוו שביצע תנועה פיזית לסימון סיום סט, באזר (PIEZO בסימולטור) לצפצוף בזמן כל חזרה, ושני כפתורים-לבחירת מטרה ולהתחלת סט. הסימולציה של המערכת בוצעה באמצעות תוכנת Tinkercad מבית AutoDesk, כאשר בסימולטור נעשה שימוש בפיאזו במקום באזר.



שם המשתנה	מספר הפין	תצורה (INPUT/OUTPUT)	הפעלה
trigPin	11	OUTPUT	שולח גל קול מחיישן המרחק
echoPin	12	INPUT	קולט את ההד ומחשב את המרחק מהאובייקט
buzzerPin	9	OUTPUT	משמיע צפצוף על כל חזרה מוצלחת
startButtonPin	6	INPUT_PULLUP	מתחיל סט חדש או מאשר מטרה
targetButtonPin	7	INPUT_PULLUP	מחליף בין מטרות אפשריות(5,10,15,20)
servoPin	10	OUTPUT	מפעיל את מנוע הסרוו בסיום הסט

<u>בדיקת ולידציה לחיישן המרחק</u>

כדי לבדוק את מידת הדיוק של חיישן המרחק מסוג HC-SR04 שבו השתמשנו בפרויקט, בוצעה בדיקת ולידציה על ידי השוואת ערכים שנמדדו בפועל על ידי החיישן לערכים ידועים שנמדדו באמצעות סרגל.
הבדיקה כללה 10 מדידות שונות בטווח של 5–50 ס״מ. בכל מדידה נרשם המרחק האמיתי לצד הערך שהוחזר על ידי החיישן. בנוסף חושב ההפרש בין שני הערכים.
מטרת הבדיקה היא לאמת את התאמת החיישן לשימוש כמד מרחק אמין בזיהוי תנועות גוף, כפי שנדרש במערכת ספירת שכיבות הסמיכה.
לצורך כך, בנוסף לטבלה בוצע גם מבחן t סטטיסטי ברמת מובהקות a=5% כדי לבדוק האם ההפרשים בין הערכים האמיתיים למדודים הם מובהקים סטטיסטית.

השערות המבחן:

$$H_0: \mu_D = 0$$
$$H_1: \mu_D \neq 0$$

הטבלה הבאה מציגה את תוצאות המדידות:

ניסוי	מרחק אמיתי (ס"מ)	מרחק שנמדד (ס"מ)	הפרש
1	5	5.3	0.3
2	10	9.8	-0.2
3	15	14.9	-0.1
4	20	20.2	0.2
5	25	24.6	-0.4
6	30	29.7	-0.3
7	35	35.1	0.1
8	40	39.8	-0.2
9	45	44.9	-0.1
10	50	50.4	0.4

תוצאות המבחן:

	Variable 1	**Variable 2**
Mean	27.47	27.5
Variance	229.7378889	229.1666667
Observations	10	10
Pooled Variance	229.4522778	

Hypothesized Mean Difference	0	
df	9	
t Stat	-0.355478627	
P(T<=t) one-tail	0.365210054	
t Critical one-tail	1.833112933	
P(T<=t) two-tail	0.730420108	
t Critical two-tail	2.262157163	

מכיוון שערך ה- p גדול מ־0.05, לא נמצאה עדות להבדל מובהק סטטיסטית בין המדידות האמיתיות למדידות החיישן, ונקבל את השערת האפס ברמת מובהקות 5%.
לפיכך, ניתן להסיק כי החיישן אמין ומדויק ברמה מספקת עבור מטרות המערכת שתוכננה.

<u>מסקנות</u>

במהלך הפרויקט פיתחנו מערכת אימונים אינטראקטיבית לספירת שכיבות סמיכה, תוך שימוש במסך, חיישן מרחק אולטרסוני, מנוע סרוו, באזר וכפתורי שליטה. המערכת מאפשרת למשתמש לבחור את יעד האימון (מספר שכיבות סמיכה), ומספקת מעקב בזמן אמת אחר ההתקדמות, יחד עם משוב קולי וחזותי בסיום היעד שנבחר.
הצלחנו לתכנן מערכת יציבה שמגיבה בצורה מדויקת לתנועת המשתמש, באמצעות תנאי סף לירידה ועלייה. בנוסף, ביצענו ולידציה לחיישן המרחק באמצעות מבחן t סטטיסטי לזוגות תואמים, שלא הראה הבדל מובהק בין ערכי אמת למדידות החיישן. על סמך זאת הסקנו שהחיישן מדויק ומתאים לדרישות המערכת.
<u>שדרוג אפשרי</u>: שדרוג עתידי אפשרי למערכת הוא הוספת יכולת לשמירה אוטומטית של נתוני האימון (כגון תאריך, יעד ומספר שכיבות סמיכה שבוצעו בפועל) בזיכרון פנימי או על כרטיס SD. שדרוג זה יאפשר למשתמש לעקוב אחר ההתקדמות האישית לאורך זמן ולנתח מגמות שיפור באימונים.

נספחים

קובץ הוראות למפעיל – README:

1. חבר את בקר ה- Arduino UNO למחשב באמצעות כבל USB.
2. ודא שכל הרכיבים מחוברים כהלכה: חיישן מרחק, מסך, LCD, כפתורים, באזר וסרוו.
3. הפעל את המערכת – תופיע הודעת פתיחה על מסך ה־ LCD.
4. לחץ על כפתור 6 כדי להתחיל סט חדש.
5. לחץ על כפתור 7 כדי לבחור יעד (5 / 10 / 15 / 20 שכיבות סמיכה).
6. לחץ שוב על כפתור 6 לאישור והתחלת הסט.
7. בצע שכיבות סמיכה מעל החיישן (המונח על הרצפה, כלפי מעלה).
8. לאחר כל חזרה חוקית – יישמע צפצוף והמסך יתעדכן.
9. בהשלמת היעד – הסרוו יפעל, והמסך יציג הודעת סיום.

קוד לתוכנת Arduino IDE:

```
#include <Wire.h>

#include <LiquidCrystal_I2C.h>

#include <Servo.h>


// יצירת אובייקט סרוו

Servo myServo;


// הגדרת פינים לרכיבים

const int trigPin = 11;

const int echoPin = 12;

const int startButtonPin = 6;

const int targetButtonPin = 7;

const int buzzerPin = 9;
```

```
// הגדרת ספים לגובה ירידה/עלייה בזיהוי שכיבה
const int pushupLowThreshold = 10;
const int pushupHighThreshold = 30;

// משתנים למעקב אחרי מצב המערכת
int pushupCount = 0;
int pushupTarget = 5;
bool reachedBottom = false;
bool inTargetSelection = false;
bool inSet = false;

// אפשרויות ליעדי שכיבות סמיכה
int targetOptions[] = {5, 10, 15, 20};
int targetIndex = 0;

// יצירת אובייקט למסך
LiquidCrystal_I2C lcd(0x27, 16, 2);

void setup() {
  // התחלת תקשורת ל־Serial Monitor
  Serial.begin(9600);
```

```
  // הגדרת תצורת פינים
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(startButtonPin, INPUT_PULLUP);
  pinMode(targetButtonPin, INPUT_PULLUP);
  pinMode(buzzerPin, OUTPUT);
  digitalWrite(buzzerPin, HIGH); // הבאזר כבוי בהתחלה

  // אתחול מסך
  lcd.init();
  lcd.backlight();
  showWelcome();

  Serial.println("System initialized. Waiting for start...");
}

void loop() {
  float distance = measureDistance();  // קריאת מרחק מהחיישן

  // טיפול בלחיצה על כפתור התחלה (6)
  if (digitalRead(startButtonPin) == LOW) {
    delay(250); // מניעת רעידות
```

```
// התחלת סט חדש (כניסה למצב בחירת מטרה)
if (!inTargetSelection && !inSet) {
  Serial.println("Start button pressed → entering target selection");
  resetToTargetSelection();
  return;
}

// אישור מטרה והתחלת סט בפועל
if (inTargetSelection && !inSet) {
  Serial.print("Target selected: ");
  Serial.println(pushupTarget);
  Serial.println("Set started.");

  inTargetSelection = false;
  inSet = true;
  pushupCount = 0;
  reachedBottom = false;

  lcd.clear();
  lcd.print("Set started!");
  delay(1000);
  lcd.clear();
  lcd.print("Pushups: 0");
```

```
    return;
  }
}

// 7 שינוי מטרה עם כפתור
if (digitalRead(targetButtonPin) == LOW && inTargetSelection) {
  delay(250);
  targetIndex = (targetIndex + 1) % 4;
  pushupTarget = targetOptions[targetIndex];

  Serial.print("Target changed to: ");
  Serial.println(pushupTarget);

  lcd.clear();
  lcd.print("Select target:");
  lcd.setCursor(0, 1);
  lcd.print(pushupTarget);
}

// אם בתוך סט – סופרים שכיבות סמיכה
if (inSet) {
  if (distance < pushupLowThreshold) {
    reachedBottom = true;  // המשתמש ירד
```

```
}

if (distance > pushupHighThreshold && reachedBottom) {
  // המשתמש עלה → חזרה שלמה נרשמת
  pushupCount++;
  reachedBottom = false;

  Serial.print("Pushup counted! Total: ");
  Serial.println(pushupCount);

  // צפצוף באזר
  digitalWrite(buzzerPin, LOW);
  delay(100);
  digitalWrite(buzzerPin, HIGH);

  // עדכון מסך
  lcd.clear();
  lcd.print("Pushups: ");
  lcd.print(pushupCount);
  delay(500);

  // אם היעד הושלם
  if (pushupCount >= pushupTarget) {
```

```
        Serial.println("Target reached! Activating servo...");

        // הפעלת מנוע סרוו כסימן לסיום
        myServo.attach(10);
        myServo.write(90);
        delay(500);
        myServo.write(0);
        delay(3000);
        myServo.detach();

        lcd.clear();
        lcd.print("Target reached!");
        lcd.setCursor(0, 1);
        lcd.print("Great job!");
        delay(3000);
        showWelcome();
        inSet = false;

        Serial.println("Set complete. System reset.");
      }
    }
  }
}
```

```
// פונקציה למדידת מרחק (בס"מ)
float measureDistance() {
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  long duration = pulseIn(echoPin, HIGH);
  return duration * 0.034 / 2.0;
}

// הצגת מסך פתיחה
void showWelcome() {
  lcd.clear();
  lcd.print("Pushup Counter");
  lcd.setCursor(0, 1);
  lcd.print("Press Btn 6");
  inSet = false;
  inTargetSelection = false;
}
```

```
// מעבר לבחירת מטרה חדשה
void resetToTargetSelection() {
  inSet = false;
  inTargetSelection = true;
  pushupCount = 0;
  targetIndex = 0;
  pushupTarget = targetOptions[targetIndex];
  reachedBottom = false;

  lcd.clear();
  lcd.print("Select target:");
  lcd.setCursor(0, 1);
  lcd.print(pushupTarget);

  Serial.println("Target selection mode active.");
}
```