**דו"ח מעבדת ארדואינו מפעילים וחיישנים**

**מטרת הפרויקט והנחות יסוד בפיתוח המערכת :**

בשנים האחרונות עולה המודעות לחשיבות השמירה על בריאות הגב והצוואר למרות הישיבה הממושכת מול המסכים. על מנת לאפשר לעובדים לעשות זאת באופן המיטבי החלטנו לפתח עזר לעובדים היושבים מול המחשב לאורך כל היום שיתריע להם כאשר הם יושבים קרוב או רחוק מידי למסך.

כאשר העובד יזוז מהמסך ויצא מהטווח שלו בזמן הישיבה על הכיסא, תתדלק נורה שתתריע לו על כך. בנוסף, יופעל מנוע סרבו שיזיז את המסך כך שהמסך יהיה מול האדם היושב על הכיסא. על המסך תוצג הזווית יוצג השינוי שנעשה.

על מנת שהנורה לא תמשיך לדלוק אחרי שקיבלנו את החיווי על טווח גדול, ניתן לכבותה גם בעזרת לחצן. כמובן כאשר העובד מזיז את הכיסא לתוך טווח ראוי מן המסך , הנורה תיכבה בעצמה.

הנחות יסוד:

1. מנוע הסרבו מחובר ויכול לכוון את מסך המחשב
2. המשתמש זז עם הכיסא ולא מבצע תנועות שעלולות לגרום לסטייה בנתונים ופגיעה באמינות המערכת.

תיאור התכן:

לצורך מימוש המערכת השתמשנו בבקר Arduino Uno, מנוע SERVO, מסך LCD, נורה וכפתור Push. על מנת להשתמש במערכת, נבצע סימולציה באמצעות תכנת הסימולטור [Tinkercad](https://www.tinkercad.com/things/gqoFH9YJgHc-epic-jofo/editel?sharecode=eK8OcohOdO7rCfr30t4CuGtnqTcWr9OABm4AnC1M9P4) מבית חברת Autodesk, Inc, All Rights Reserved.

איור 1: שרטוט חשמלי של הפרוייקט

|  |  |  |  |
| --- | --- | --- | --- |
| הפעלה | תצורה | שם המשתנה | מספר הפין |
| הנורה תדלק כאשר אנחנו לא נמצאים בטווח הרצוי, בהנחה והמערכת דולקת | OUTPUT | Led\_PIN | 4 |
| הוא יעצור את הדלקת הנורה כאשר ילחצו עליו | INPUT | PushButon\_PIN | 7 |
| חיישן למדידת מרחק אשר מוסר שידור קולי באופן קבוע כאשר המערכת מופעלת. | OUTPUT | Trig\_PIN | 3 |
| חיישן למדידת מרחק אשר מקבל שידור קולי באופן קבוע כאשר המערכת מופעלת. | INPUT | Echo\_PIN | 2 |
| מנוע הסרבו - על מנת לבצע תנועה זוויתית. | OUTPUT | RST\_PIN | 6 |

בדיקות ולידציה של המערכת:

1. תיאור הבדיקות שבוצעו, תיאור זה יכלול את מטרת הבדיקות והמבחן סטטיסטי בו נעשה שימוש לבחון את התוצאות (הגדרת המבחן לא תוצאות המבחן). בדיקה אחת תהייה עבור החומרה ובדיקה נוספת עבור הלוגיקה של המערכת (כלומר שהמערכת עובדת בהתאם לתכנון).

בדיקות ולידציה של המערכת-

ביצענו את הבדיקה על מנת לוודא את המערכת בפרט את תאימות המערכת מבחינה פיזית ומבחינה לוגית. תפקידו של חיישן המרחק היה לזהות את המיקום של שמשתמש בטווח בין 0 ל-100 בדיוק קרובה ככל הניתן באמצעות סרגל נגלל. בנוסף מנוע הסרבו אמור להסתובב בזוויות המתאימה למיקום המתקבל. (ולידציות פיזיות)

כמו כן המערכת נדרשת לזהות "טווח התראה ", כלומר היא מרתיעה כאשר המיקום הינו גדול מ100 (ולידציה לוגית)

השערתנו היא שממוצע השגיאות הינו 0 ברמת ביטחון של 90% ((. ההשערה האלטרנטיבית היא כי ממוצע השגיאות שונה מ-0.

סיכום:

פיתחנו מערכת שמטרתה לעזור לשמור על בריאותו של העובד באמצעות בקר ארדואינו. על פי תוצאות המבחן שקיבלנו, אנחנו מסיקים שהמערכת עונה על הדרישות ותקל על המשתמשים בה.

אנחנו ממליצים בהמשך לפתח פיצ'רים נוספים למערכת, כמו יכולת התראה על שימוש במסך קרוב או רחוק מידי ועוד.

למדנו הרבה על ארדואינו אונו ואת האופן לעבוד איתו ועם כלל הרכיבים בהם השתמשנו.

נספחים

תיעוד הקוד:

#include <Servo.h>

#include <Wire.h>

#include <Adafruit\_LiquidCrystal.h>

Adafruit\_LiquidCrystal lcd(0);

#define Echo\_PIN 2

#define Trig\_PIN 3

#define Led\_PIN 4

#define RST\_PIN 6

Servo servo;

int pushButton = 8;

bool ledShouldBeOff = false; // Tracks if LED should be manually kept off

bool distanceWasLessThan100 = true; // Tracks if the last distance was less than 100

void setup() {

servo.attach(RST\_PIN);

pinMode(Led\_PIN, OUTPUT);

pinMode(Trig\_PIN, OUTPUT);

pinMode(Echo\_PIN, INPUT);

pinMode(pushButton, INPUT\_PULLUP); // Use internal pull-up resistor

lcd.begin(16, 2);

lcd.setBacklight(1);

lcd.clear();

lcd.setCursor(0,0);

}

void loop() {

static bool lastButtonState = LOW; // Tracks the last state of the button

bool currentButtonState = digitalRead(pushButton);

int distance = getDistance();

int tiltAngle = map(distance, 0, 100, 0, 180);

// Button press detection (rising edge)

if (currentButtonState == HIGH && lastButtonState == LOW) {

ledShouldBeOff = !ledShouldBeOff; // Toggle LED off state

digitalWrite(Led\_PIN, LOW); // Turn off LED immediately upon pressing

delay(50); // Debounce delay

}

lastButtonState = currentButtonState;

// Control LED based on distance and manual override

if (distance < 100) {

digitalWrite(Led\_PIN, LOW); // Turn off if too close

distanceWasLessThan100 = true;

} else if (distance > 100 && (!ledShouldBeOff || (ledShouldBeOff && distanceWasLessThan100))) {

digitalWrite(Led\_PIN, HIGH); // Turn on if the distance becomes more than 100 and it was previously less or if manual off is not active

distanceWasLessThan100 = false;

}

// Display information

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("Dist: ");

lcd.print(distance);

lcd.print(" cm");

lcd.setCursor(0, 1);

lcd.print("Angle: ");

lcd.print(tiltAngle);

lcd.print(" deg");

tiltScreen(tiltAngle); // Adjust screen tilt based on distance

delay(100);

}

int getDistance() {

digitalWrite(Trig\_PIN, LOW);

delayMicroseconds(2);

digitalWrite(Trig\_PIN, HIGH);

delayMicroseconds(10);

digitalWrite(Trig\_PIN, LOW);

return pulseIn(Echo\_PIN, HIGH) \* 0.034 / 2;

}

void tiltScreen(int angle) {

servo.write(angle);

delay(20); // Ensure servo has time to reach the position

}

1. Download the program to the Arduino controller, and connect the controller to the voltage.
2. The system has now started working.
3. Upon startup, the system initializes and begins measuring distances using the ultrasonic sensor.
4. The LCD screen displays the current distance measured by the sensor and the corresponding tilt angle of the screen.
5. If an object is detected beyond 100 cm, the LED

turns on automatically to give a warning signal.

The led will automatically turn off when the distance is again measured less then 100 cm or when the user forcefully turns the led off by pressing the push button.