# ISTANBUL TICARET UNIVERSITY
# FACULTY OF ENGINEERING

# ENG402 GRADUATION PROJECT

# GRADUATION PROJECT REPORT

**Department : Mechatronics Engineering**

**Year/Term : 2025**

**Title of the Project : Kirpi**

**Student Information:**

**Name-Surname : Abdullah Eren Yalçın**

**ID : 200023619**

**Advisor : Muhammed Ceylan**

**Date :**

**Signature:**

# TABLE OF CONTENTS

## ABSTRACT

This report presents the technical structure, system architecture, and functional components of **"Kirpi,"** a smart robotic vehicle prototype equipped with environmental awareness and controllable through hand gestures. Kirpi is a mobile vehicle system developed through the integrated use of modern sensor technologies, wireless communication infrastructure, and image processing algorithms, enabling direct interaction with the user.

The vehicle is equipped with distance sensors placed on all four sides to detect nearby obstacles, along with temperature and humidity sensors that monitor environmental conditions. The data collected from all sensors is transmitted to a handheld station via the HC-05 Bluetooth module. The handheld station features an LCD screen, which displays real time readings of the temperature and humidity levels, as well as the centimeter-based distance values from each of the directional distance sensors. This configuration ensures that the user has continuous environmental and spatial awareness, making system interaction intuitive and informative.

Kirpi is also equipped with an autonomous obstacle detection and avoidance mechanism. When an object approaches within 20 centimeters of either the front or rear of the vehicle, the system identifies the proximity and automatically moves the vehicle in the opposite direction. This response helps the vehicle prevent potential collisions and ensures safer interaction with its surroundings. Though simple in structure, this behavior represents an effective example of basic autonomous decision-making.

The control system connects to a computer via a second Bluetooth module, the **HC-06**, which enables wireless transmission of movement commands. Hand gestures performed by the user in front of a computer camera are analyzed using image processing algorithms. These gestures are then translated into control commands—forward, backward, left, or right which are sent to the vehicle. This design eliminates the need for traditional physical controllers and offers a touch-free, gesture-based control interface.

The Kirpi project is an interdisciplinary engineering endeavor that combines robotic control, sensor-based automation, wireless communication, and human-machine interaction. The goal of the project is not only to create a technically functional system but also to develop a user-friendly, responsive, and safe mobile platform. In this regard, Kirpi serves as a compelling example of modern innovation in robotics and interactive system design.

# PREFACE

This report covers the technical development and engineering approach of Kirpi, a mobile robotic system, from its initial conceptual phase to its transformation into a functional prototype. The project was designed with the goal of creating a vehicle capable of environmental awareness, remote controllability, and autonomous maneuvering—particularly suited for silent and low-profile operations in confined spaces.

Kirpi is equipped with distance sensors on all four sides, allowing it to detect the proximity of surrounding objects; in the event of a collision risk, the vehicle performs an evasive maneuver if an object is detected in front of or behind it. Additionally, it continuously monitors environmental conditions via temperature and humidity sensors. The vehicle can be directed without physical contact through hand gestures recognized by a computer camera, and it is controlled wirelessly via Bluetooth communication technology.

Kirpi is especially well-suited for operations such as preliminary reconnaissance in abandoned buildings, along minefield perimeters, or within narrow corridors, where its silent operation, lack of visible detection systems like cameras, and gesture-based control make it ideal for radio-silent missions. These characteristics distinguish Kirpi from conventional reconnaissance systems and position it as a passive platform for discreet tasks.

During the development process, several technical challenges were encountered—such as maintaining consistent sensor data flow, ensuring stable Bluetooth communication, and accurately interpreting hand gestures through image processing. These challenges were overcome through iterative development and field testing. In this regard, Kirpi is not just a technical design, but a solution shaped by considerations of safety, interaction, and flexibility.

I would like to express my sincere gratitude to all academic advisors, technical contributors, and those who offered moral support throughout this project. Kirpi represents not only a robotic system but also a vision of engineering shaped by strategic thinking and technological innovation.

## EVOLUTION OF IMAGE PROCESSING TECHNOLOGIES

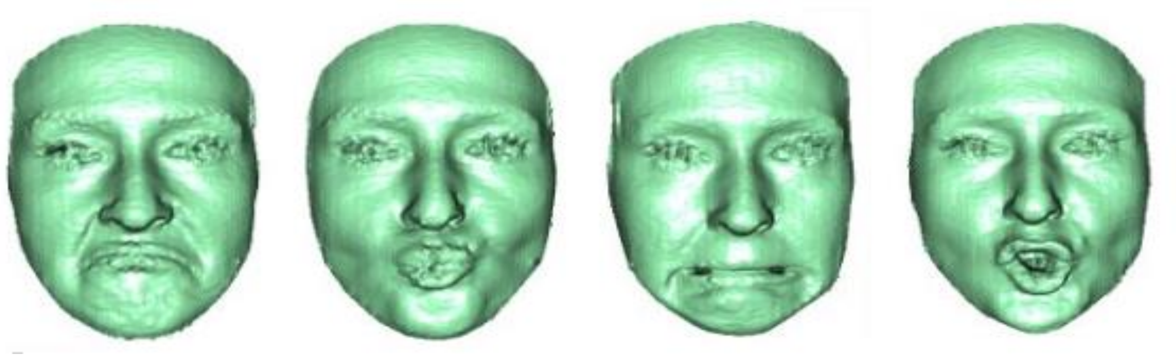
Image processing is a multidisciplinary field of study covering the analysis, transformation and interpretation of digital data through visual content. The foundations of image processing techniques used today in many sectors such as robotics, medicine, automotive, defense and entertainment were laid in the 1960s. In this period, especially the need to analyze satellite images in a digital environment has become the primary application area of image processing. Organizations such as NASA have expanded the use of numerical techniques to analyze images obtained from space.

In the 1970s, academic studies on the analysis of images using mathematical methods gained momentum; basic algorithms such as Fourier transforms, filtering techniques, noise reduction, edge detection and segmentation were developed during this period. Although computer technology is not yet strong enough, the theoretical infrastructure has been largely created during these years.

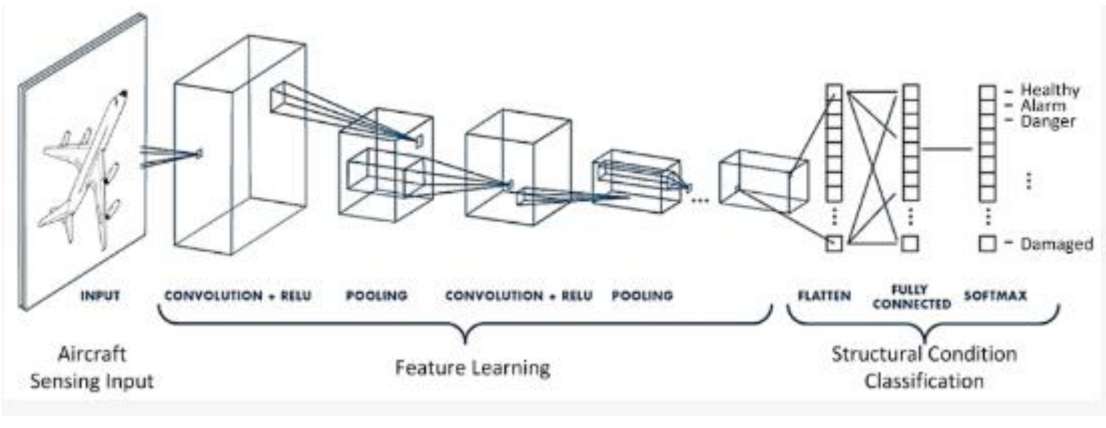
With the spread of microprocessors and the strengthening of personal computers in the 1980s, image processing techniques have become practically applicable. Solutions for digital image processing have started to be developed, especially in applications such as industrial quality control, medical imaging, automatic license plate recognition and face detection. However, at that time, due to the lack of software libraries and limited hardware, applications still required expertise.

The 1990s were a period when image processing was democratized. OpenCV (1999), developed by Intel, has made a breakthrough in this field. Thanks to the open source structure, researchers and developers have found the opportunity to implement their projects faster and more efficiently around the world. With this library, subjects such as motion tracking, object recognition, eye/head/hand detection, camera calibration have become more accessible. In parallel, artificial vision (computer vision) and machine vision (machine vision) applications have also started to be commercialized.

Since the 2000s, image processing has ceased to be only static analysis tools, it has become a basic component of real-time and interactive systems. In particular, subjects such as video-based analysis, moving object tracking, 3D depth analysis have attracted great attention during this period. Image processing has now begun to be used not only to “see” objects, but also to understand and interpret them.



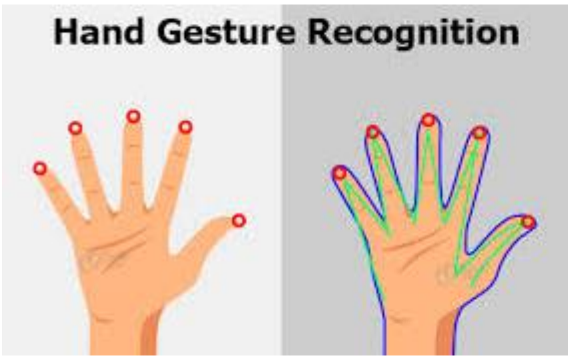
Since the 2010s, deep learning techniques have revolutionized image processing. Traditional algorithms have been replaced by artificial neural networks that can learn patterns on their own. In particular, models such as Convolutional Neural Networks (CNN), YOLO, ResNet have provided high accuracy in feature extraction and classification from images, which has made image processing become the standard in all fields.




hedgehog project is at the last stage of this evolution; it offers a system that detects hand movements using image processing algorithms and applies the commands corresponding to these movements. Thus, not only visual perception, but also interaction and decision-making processes have become a part of automation.

## Hand Gesture Recognition and Non-Contact Interaction

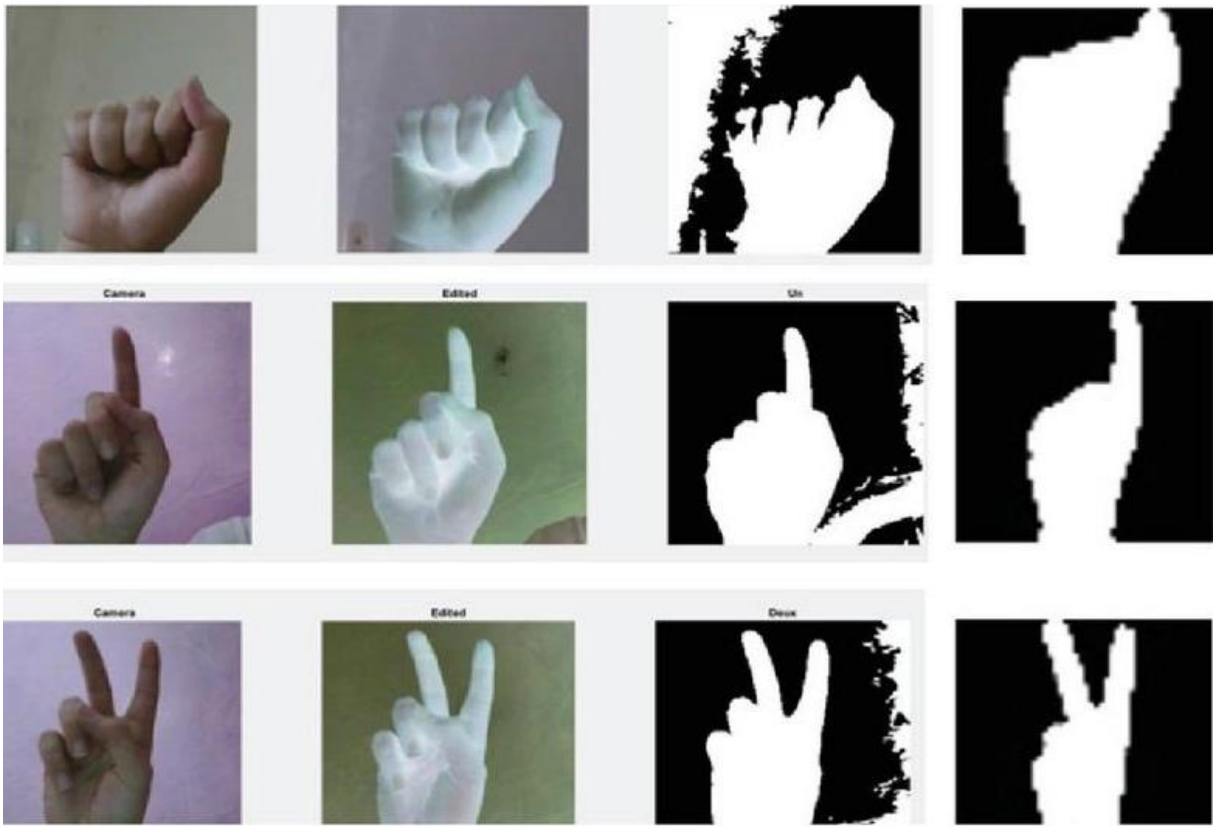
The evolution of image processing in the direction of interacting directly with humans has been possible, especially with the development of hand gesture recognition technology. In traditional human-machine interactions, the user would physically press keys, use a controller, or touch touch screens. However, these methods are not ideal, especially for sensitive, quiet or sterile environments. At this point, contactless control systems have been activated.




Hand gesture recognition technology is based on the principle of detecting the user's hand positions and movements through the camera and translating these movements into predefined commands. This technology was first discovered by the gaming industry in the early 2000s. One of the most well-known examples is Microsoft's Kinect platform. This system could detect not only movements, but also users' body positions in three dimensions. After the success in the entertainment sector, this technology has been used in health, safety, education and robotic systems.

Initially, gesture recognition systems were based on template matching and rule-based approaches. However, these systems were not resistant to speed, personal differences and environmental variables. since the 2010s, machine learning and especially deep learning methods have addressed this deficiency. By training CNN-based models on a large number of images, they have become able to recognize hand movements with high accuracy even from different angles and under changing lighting conditions.

In addition, thanks to developing hardware technologies, such operations have become feasible in real time. This situation has reduced the response time of the systems and improved the user experience. Today, gesture-based interaction systems are actively used in sterile environments such as operating rooms, military applications or robotic control systems.




**PROJECT DESCRIPTION and WORKING PRINCIPLE**

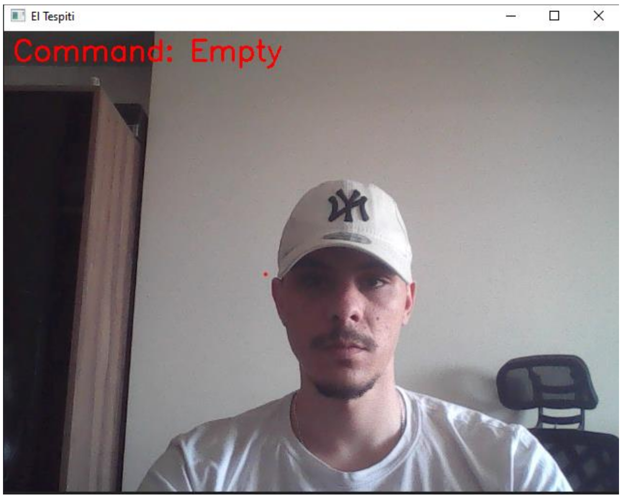
The vehicle developed in this project is equipped with sensors that can detect environmental data and has a structure that can be controlled both autonomously and through hand movements. The vehicle includes four HC-SR04 ultrasonic sensors, one temperature and humidity sensor, one HC-05 Bluetooth module, and an Arduino microcontroller that manages the system. All data obtained from the sensors is wirelessly transmitted to the hand station via the HC-05 Bluetooth module and presented to the user on an LCD screen.

The vehicle possesses autonomous movement features. When the ultrasonic sensors at the front and rear detect an obstacle within 20 cm or less, the vehicle automatically moves in the opposite direction of the detected obstacle to prevent collisions. This automatic avoidance system is only applicable to the front and rear sensors, while the left and right sensors are used solely to provide distance information.

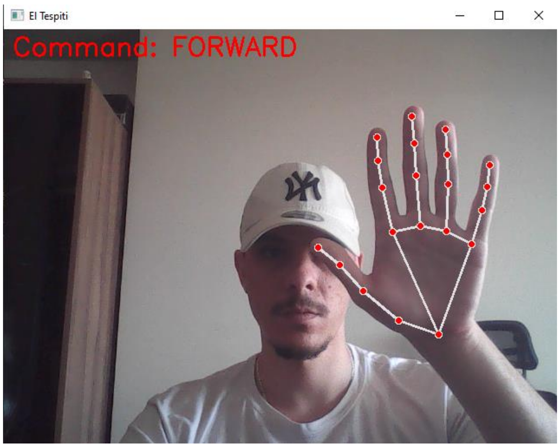
Manual control of the vehicle is achieved through a command system based on hand movements. In this system, the HC-06 Bluetooth module on the vehicle is paired with a computer. A program written in Python using the Mediapipe and OpenCV libraries detects hand movements. There are five defined movements: forward, backward, right, left, and stop. If hand movements are not detected or if the hand is not recognized, the "Empty" command is displayed. This ensures that only valid hand movements are transmitted to the vehicle, preventing the transmission of incorrect movements.

## Defined hand gestures:

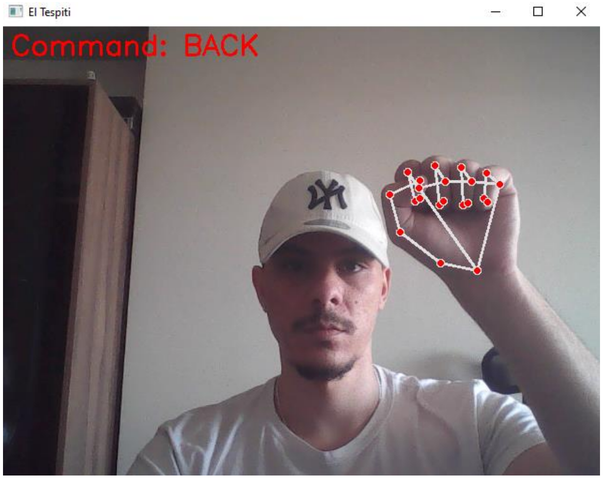
1) **Empty**



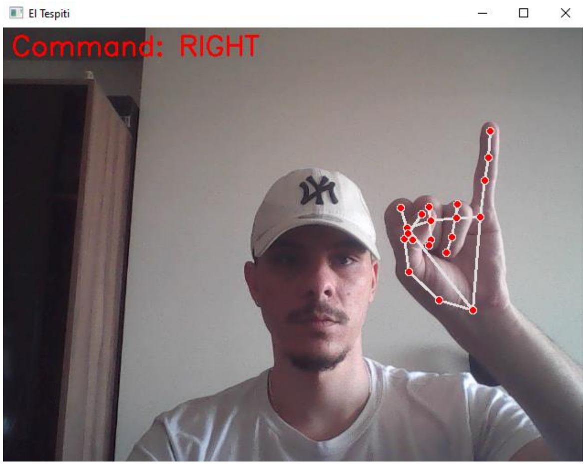

2) **Forward**



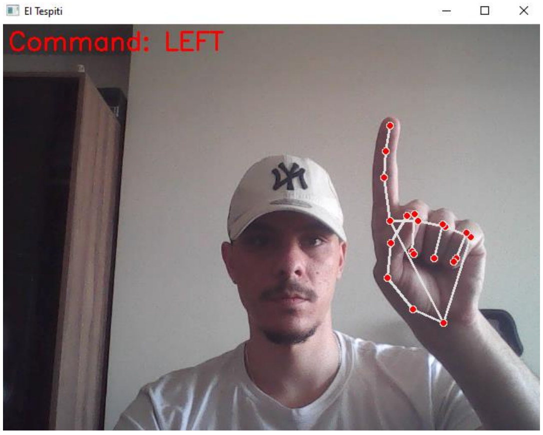

3) **Back**



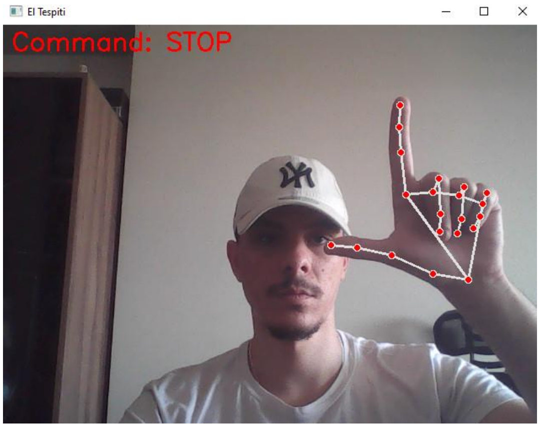

4) **Right**




5) **Left**




6) **Stop**



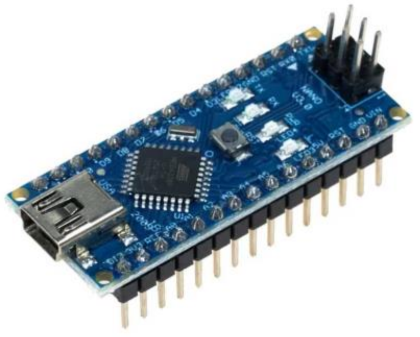

The detected hand movements are converted into defined commands by the system:

- **F** → Forward
- **R** → Right
- **L** → Left
- **B** → Backward
- **S** → Stop

These commands are transmitted to the Arduino as defined. The transmitted commands enter the desired functions on the Arduino, controlling the four motors mounted on the vehicle. Each of these motors is individually connected to the four wheels of the vehicle, and the movement of the wheels allows the vehicle to perform the desired maneuvers.

# Materials used on the project

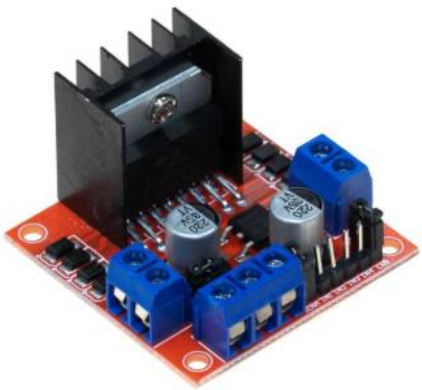
### Arduino Nano



## Arduino Nano  Technical Specifications:

- Microcontroller ATmega328
- Working Voltage 5V
- Input Voltage (recommended) 7-12V
- Input Voltage (limit) 6-20V
- 14 Digital I/O Pins (6 of which are PWM outputs)
- Analog Input Pins 8
- The Current for each I/O is 40 mA
- Current for 3.3V Output is 50 mA
- Flash Memory 32 KB (ATmega328) Up to 2 KB is used by the bootloader
- SRAM 2 KB (ATmega328)
- EEPROM 1 KB (ATmega328)
- Clock Speed 16 MHz
- Length 45 mm
- Width 18 mm
- Weight 5 g

## L298n



### Features of L298N:

- It can control two separate motors independently of each other.
- It can provide 2A current per channel.
- There is a built-in regulator on it.
- It has high temperature and short circuit protection.
- There are LEDs that light up according to the direction of rotation of the engine.
- There is a built-in cooler on the card.
- The current sense pins are exported.

### LCD display with i2c module


### Features

- **Çalışma gerilimi: 5V.**
- **Boyutlar: 54x18x10mm.**
- **Back Lighting özelliğine sahiptir.**
- **LCD arka fon ışığı olmadan 4mA akım çekmektedir.**
- **Çalışma sıcaklığı -20 ile +70 derece arasıdır.**

### TT Motor

## Features of TT Motor:

- Working Voltage: DC3V - 9V (6V Is Recommended)
- Reduction Ratio: 1:48 (3V)
- Speed: 250 Rpm(6V)
- Current: 70mA(3V), 250mA(MAX)
- Torque: 800g.cm (3V)
- Torque: 5500g.cm (6V)
- Shaft: 3.66mm x 5.3mm
- Engine Weight: 29gr.

## HCSR-04


## HC-SR04 Technical Specifications:

- Working Voltage: DC 5V
- Current Drawn: 15 mA
- Operating Frequency: 40 Hz
- Maximum Visual Range: 4m
- Minimum Visual Range: 2cm
- Viewing Angle: 15°
- Trigger Leg Input Signal: 10 us TTL Pulses
- Echo Output Signal: Input TTL signal and Distance Ratio
- Dimensions: 45mm x 20mm x 15mm

## HC-05

## HC05 Bluetooth Modül Özellikleri:

- Seri Haberleşme: UART arayüzü ile kolay entegrasyon
- Uyumluluk: Arduino, Raspberry Pi ve diğer geliştirme kartları ile uyumlu
- Uzun Menzil: Güçlü Bluetooth 2.0 ve 4.0 uyumluluğu
- Düşük Güç Tüketimi: Enerji verimliliği sağlar
- Kompakt ve Dayanıklı Tasarım: Taşınabilir ve uzun ömürlü kullanım için ideal

## HC-06

## HC06 Bluetooth Module Specifications:

- Bluetooth Protocol: Bluetooth 2.0+EDR(Advanced Data Rate)
- 2.4GHz communication frequency
- Sensitivity: ≤-80 dBm
- Output Power: ≤+4 dBm
- Asynchronous Speed: 2.1MBps/160 KBps
- Synchronous Speed: 1 MBps/1 MBps
- Security: Authentication and Encryption
- Working Voltage: 3.6-5V(Recommended 3.6V)
- Current: 50 mA
- Dimensions: 43x16x7mm

## 18650 Battery


## Technical specifications:

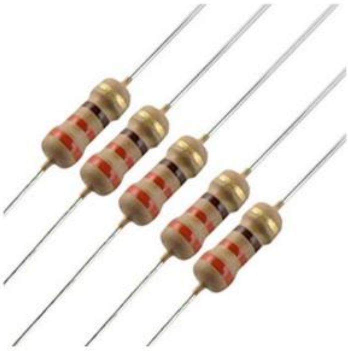
- Size : 18650
- Polar Headless
- Rated Capacity: 2600mAh
- Nominal Voltage. : 3.7V

## 18650 Battery case

**Features:**

- Battery type: 4-series 18650 battery.
- Output voltage: DC 14.8V.
- Cable: 15 cm.
- Material: Abs Plastic and Metal.
- Small size.
- Working temperature: -20C/70C.

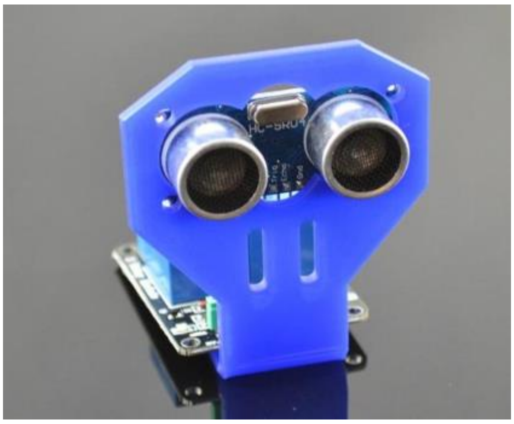
## 330R Resistor



## Klamens

Terminal blocks are elements that provide secure connection of wires in electrical circuits.

## Distance sensor holder



It holds sensors.

## Battery box

It holds batteries.

## LED

## Features

- LEDs are semiconductor materials.
- Their main substance is silicon.
- When a current passes through it, they give light by releasing photons.
- They are produced in such a way that they give light at different angles.

## Wheel

The movement of the vehicle is ensured thanks to the wheels

## Jumper Cable

### Features

- It is usually made of copper wire or copper-coated aluminum.
- Flexible, multi-wire structures are more durable.
- Insulation is usually PVC or silicone sheath.

### 9V Battery


**Technical Specifications:**

- **Battery capacity: 560 mAh**
- **Voltage: 9V**
- **Weight: 43 gr**

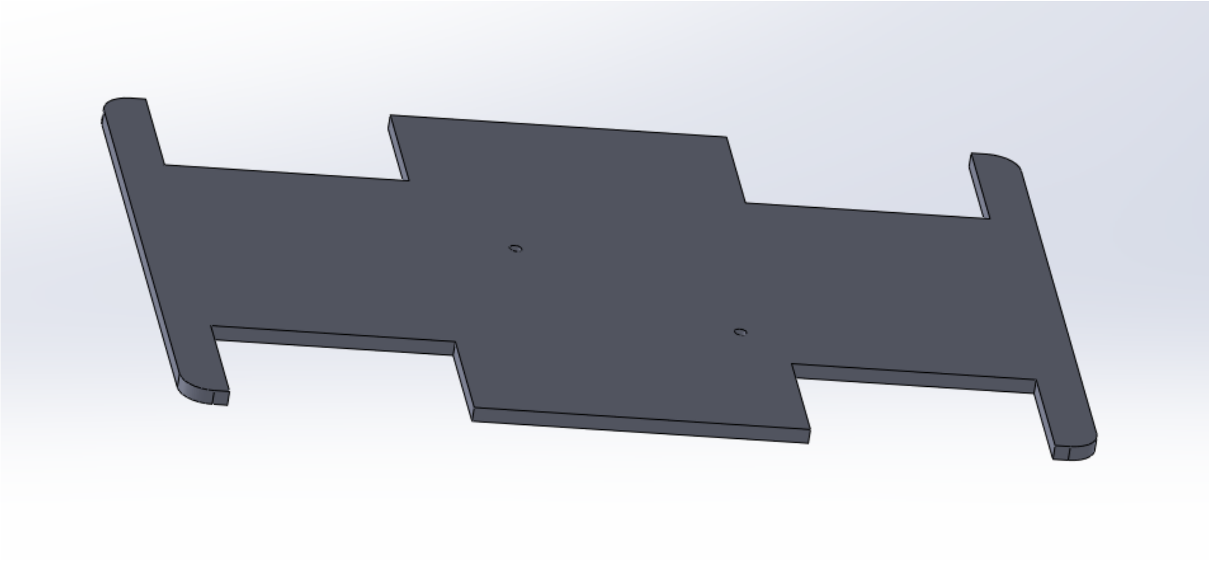
### 9v battery cap

Provides 9V DC voltage output.

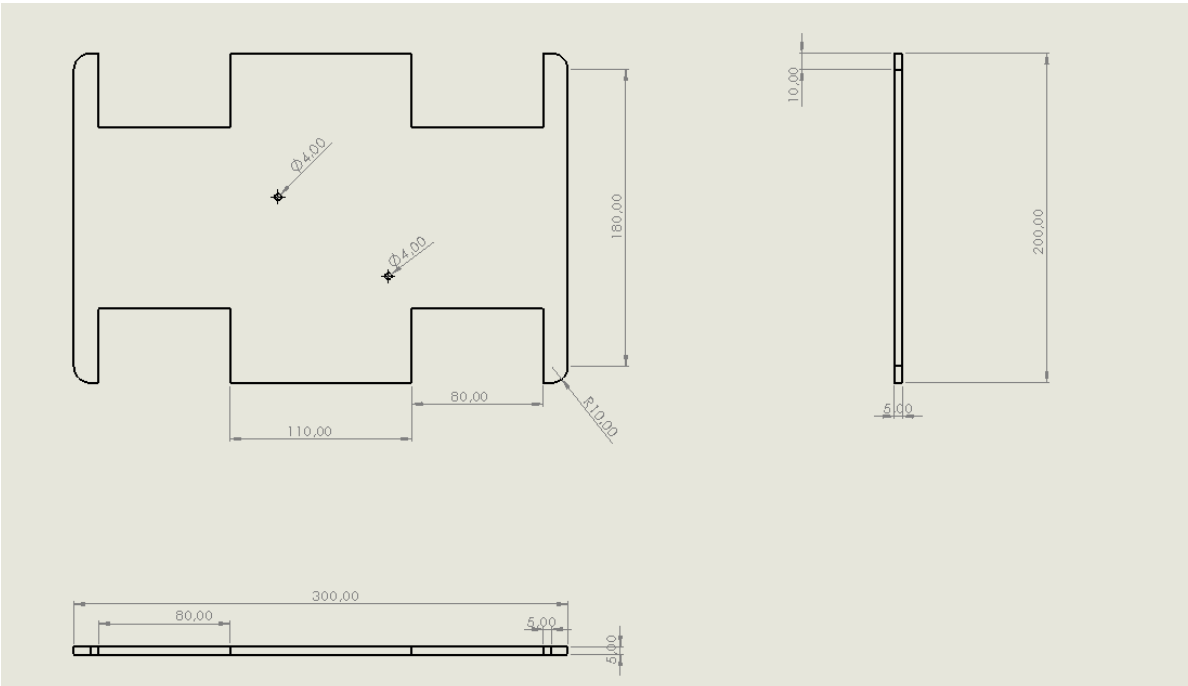
# Mechanical Part

The mechanical structure determines the durability, stability and functionality of the system and ensures the safe and correct operation of electronic and software components.

## SolidWorks Design



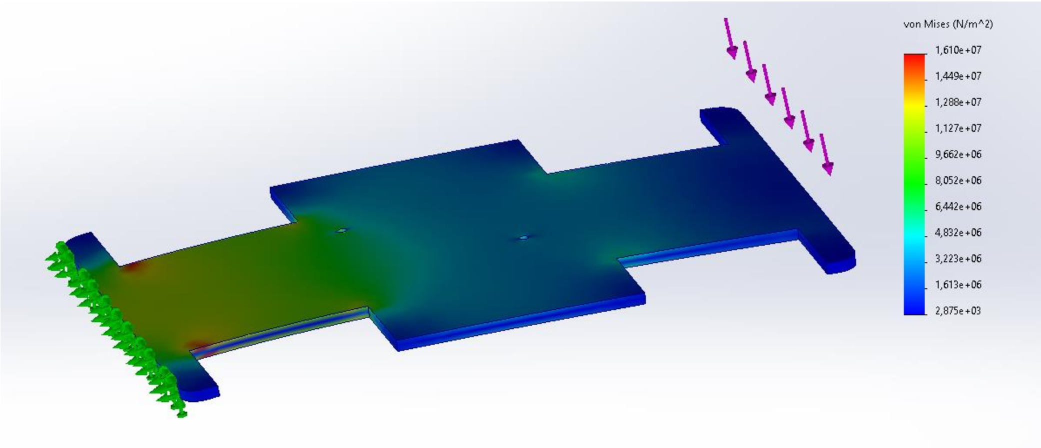
### Technical Drawing



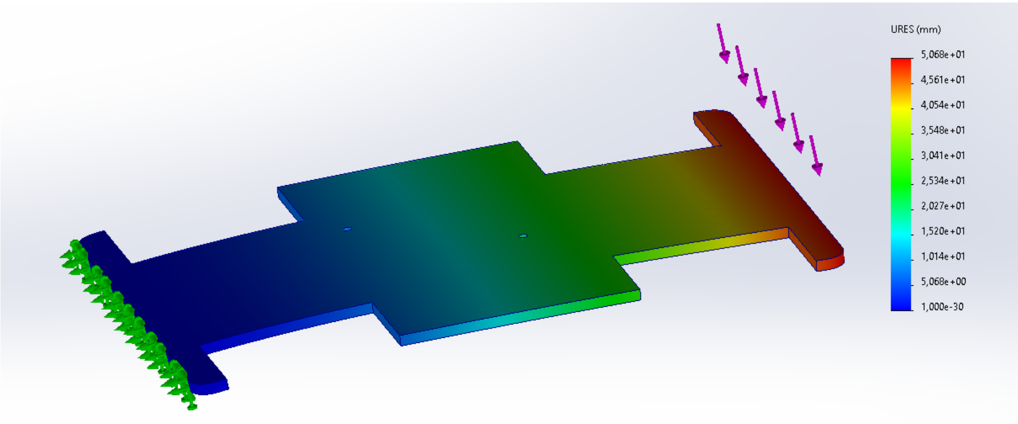

# Simulation Results

## Static analysis results under 20N force

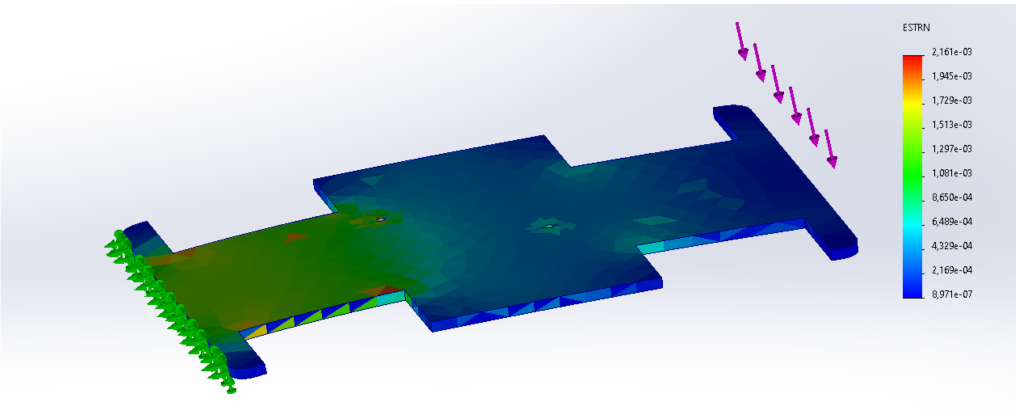
### Stress



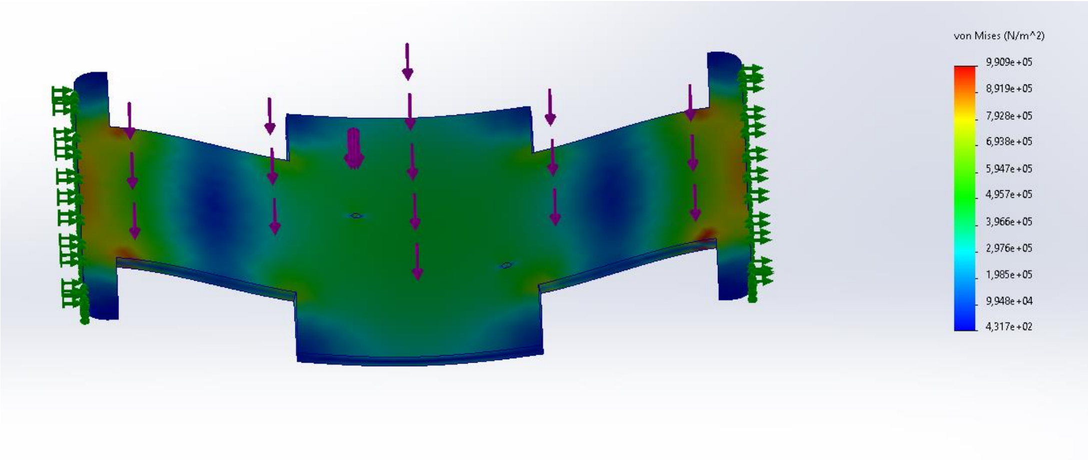

### Displacement



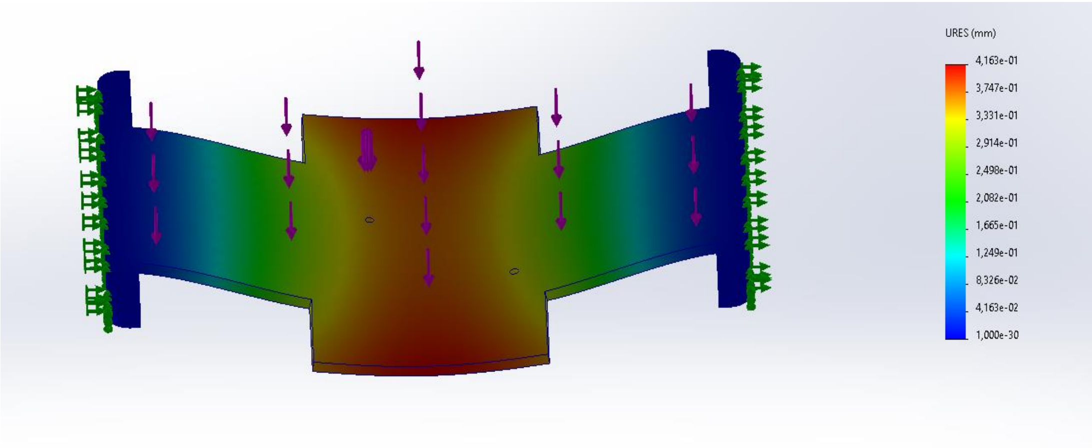

### Strain



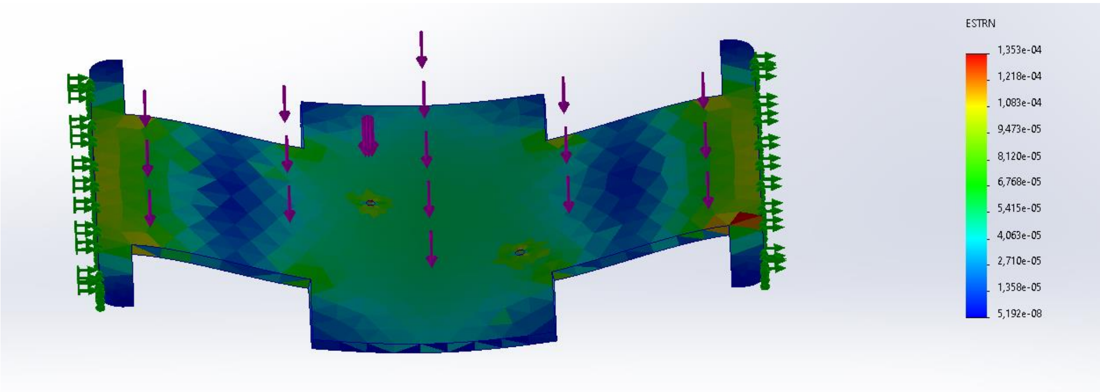

**Stress**




**Displacement**

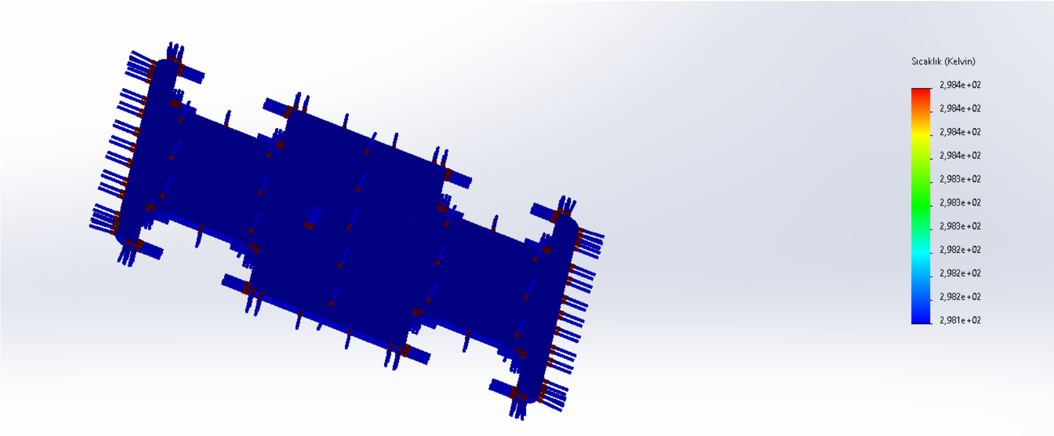



**Strain**

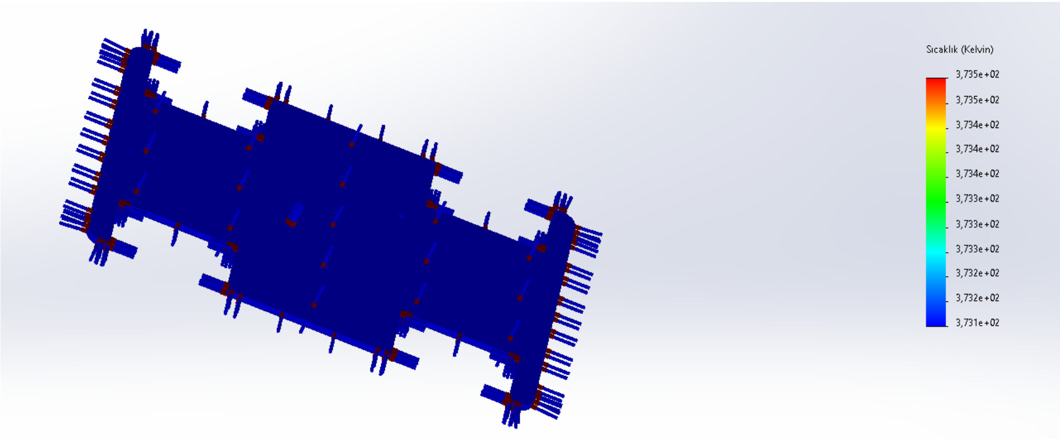



## Thermal simulation results

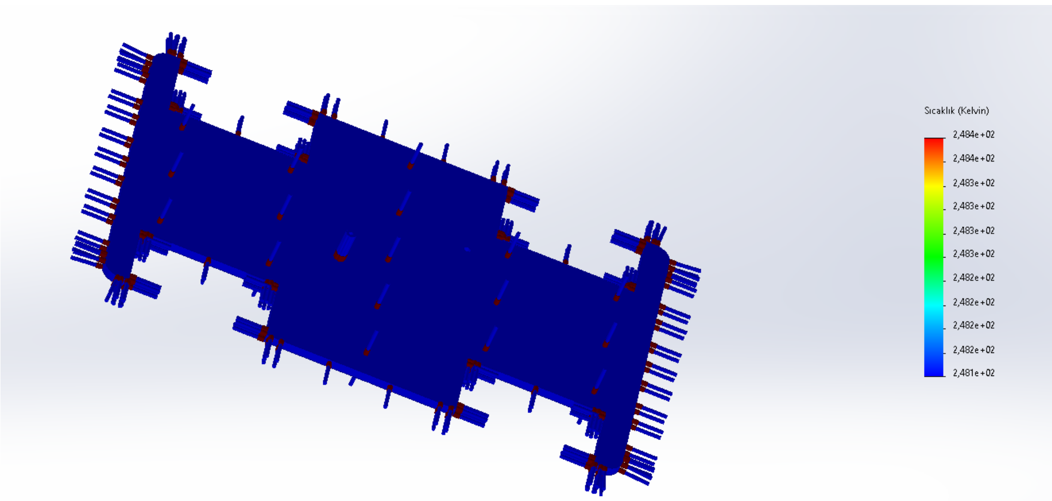
**25 Celcius tempurature**




**100 Celcius tempurature**



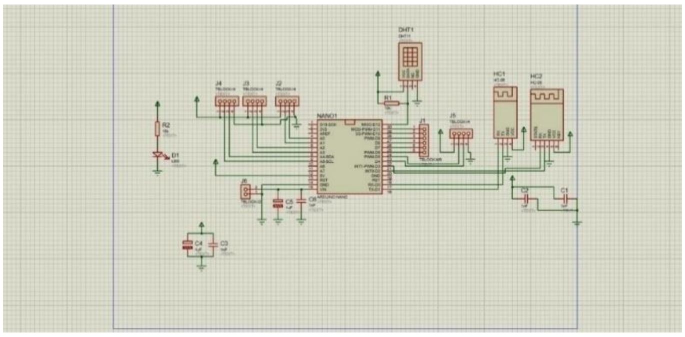

**-25 Celcius Temperature**




# Electrical Part

The electrAcal part of a project encompasses the processes of provAdAng the necessary energy source for the system to operate, regulatAng energy dAstrAbutAon, and ensurAng that cArcuAt components functAon correctly. ThAs sectAon typAcally Ancludes the power supply, wArAng, connectAons, cArcuAt elements (such as resAstors, transAstors, LEDs), and control unAts. The electrAcal part delAvers energy to other sectAons of the project and supports the safe and eAAcAent operatAon of the system. AddAtAonally, At As desAgned to ensure that the components An the cArcuAt are supplAed wAth the correct voltage and current values. In summary, the electrAcal part can be consAdered the "heart" of the project, enablAng all components to work An harmony.

**Circuit diagram:**



1. **Power Supply (DHT1):** The power supply provides the necessary electrical energy to all the components in the system. Obtaining the correct voltage and current values is vital for the stable and reliable operation of the system.
2. **Microcontroller (NANO1):** The microcontroller forms the brain of the system. It processes the data from the sensors and controls the actuators. It provides the overall functionality of the system.
3. **Communication Connections (MIBO-D12, DIB-SCK):** The communication connections between the microcontroller and other components ensure the correct transfer of data. These connections enable the integrated operation of the system.
4. **Input/Output Connections (HC1, HC2):** The input/output connections used to receive data from sensors and control the actuators facilitate the system's interaction with the environment.
5. **Circuit Elements (R1, R2, C4, C5, C3):** The passive circuit elements protect the circuit from overcurrent and voltage fluctuations, ensuring the safe operation of the system.

# Software

## Software used:

### 1) Python

Python is an open-source, high-level, and object-oriented programming language developed in 1991. With its simple and readable syntax, it is preferred by both beginners and professionals. Since it does not require compilation, it speeds up the development and debugging processes.

Python can be used in many areas such as web development, data analysis, artificial intelligence, image processing, and automation. Thanks to its wide library support, complex applications can be developed easily. It runs on all major operating systems, is free to use, and has strong community support. With these features, Python is a versatile programming language widely used in both education and industry.

## Libraries used:

1. **MediaPipe**

   MediaPipe is an open-source library developed by Google for real-time image processing and machine learning tasks. It provides optimized models for hand, face, and body tracking, and is widely used in low-latency computer vision applications**OpenCV**

2. **OpenCV**

   (Open Source Computer Vision Library) is an open-source library developed for image processing and computer vision applications. It can be used with languages like C++ and Python, and is commonly used in tasks such as face recognition, object tracking, and motion detection. With its ability to perform real-time processing, it is widely preferred in robotics, security, and artificial intelligence projects

3. **Pyserial**

   PySerial is a Python library used for communication over serial ports (COM ports). It is commonly used to interact with devices like Arduino, Bluetooth modules (HC-05/06), and sensors.It allows users to open serial connections, send and receive data through Python, making it easy to establish communication between hardware and software.

# Python Code:

```
import cv2
```

```python
import mediapipe as mp
import serial
import time

arduino = serial.Serial('COM4', 9600)
time.sleep(2)

mp_hands = mp.solutions.hands
mp_drawing = mp.solutions.drawing_utils
hands = mp_hands.Hands(min_detection_confidence=0.7,
min_tracking_confidence=0.5)

cap = cv2.VideoCapture(0)

last_command = "empty"
confirmed_command = "empty"
candidate_command = "empty"
candidate_start_time = 0
STABLE_DURATION = 0.5  # Komut sabit kalma süresi

while cap.isOpened():
    success, image = cap.read()
    if not success:
        print("Kamera açılırken hata oluştu.")
        break

    image = cv2.flip(image, 1)
    image_rgb = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
    results = hands.process(image_rgb)
```

```
    current_time = time.time()
    detected_command = "Empty"

    if results.multi_hand_landmarks:
        for hand_landmarks in results.multi_hand_landmarks:
            mp_drawing.draw_landmarks(image, hand_landmarks,
mp_hands.HAND_CONNECTIONS)

            thumb_tip_y =
hand_landmarks.landmark[mp_hands.HandLandmark.THUMB_TIP].y
            index_tip_y =
hand_landmarks.landmark[mp_hands.HandLandmark.INDEX_FINGER_TIP].y
            middle_tip_y =
hand_landmarks.landmark[mp_hands.HandLandmark.MIDDLE_FINGER_TIP].y
            ring_tip_y =
hand_landmarks.landmark[mp_hands.HandLandmark.RING_FINGER_TIP].y
            pinky_tip_y =
hand_landmarks.landmark[mp_hands.HandLandmark.PINKY_TIP].y

            index_mcp_y =
hand_landmarks.landmark[mp_hands.HandLandmark.INDEX_FINGER_MCP].y
            middle_mcp_y =
hand_landmarks.landmark[mp_hands.HandLandmark.MIDDLE_FINGER_MCP].y
            ring_mcp_y =
hand_landmarks.landmark[mp_hands.HandLandmark.RING_FINGER_MCP].y
            pinky_mcp_y =
hand_landmarks.landmark[mp_hands.HandLandmark.PINKY_MCP].y

            if (pinky_tip_y < thumb_tip_y and
                pinky_tip_y < index_tip_y and
                pinky_tip_y < middle_tip_y and
```

```
    pinky_tip_y < ring_tip_y and
    index_tip_y > thumb_tip_y and
    middle_tip_y > thumb_tip_y and
    ring_tip_y > thumb_tip_y):
    detected_command = "RIGHT"
elif (index_tip_y < thumb_tip_y and
      index_tip_y < middle_tip_y and
      index_tip_y < ring_tip_y and
      index_tip_y < pinky_tip_y and
      pinky_tip_y > thumb_tip_y and
      middle_tip_y > thumb_tip_y and
      ring_tip_y > thumb_tip_y):
    detected_command = "LEFT"
elif (index_tip_y < thumb_tip_y and
      middle_tip_y < thumb_tip_y and
      ring_tip_y < thumb_tip_y and
      pinky_tip_y < thumb_tip_y and
      index_tip_y < middle_tip_y):
    detected_command = "STOP"
elif (index_mcp_y < thumb_tip_y and
      middle_mcp_y < thumb_tip_y and
      ring_mcp_y < thumb_tip_y and
      pinky_mcp_y < thumb_tip_y):
    detected_command = "FORWARD"
elif (index_mcp_y > thumb_tip_y and
      middle_mcp_y > thumb_tip_y and
      ring_mcp_y > thumb_tip_y and
      pinky_mcp_y > thumb_tip_y):
    detected_command = "BACK"
```

```
            break

    # Kararlılık kontrolü
    if detected_command != candidate_command:
        candidate_command = detected_command
        candidate_start_time = current_time
    else:
        if (current_time - candidate_start_time) >= STABLE_DURATION and
candidate_command != confirmed_command:
            confirmed_command = candidate_command

            if confirmed_command == "FORWARD":
                arduino.write(b'F')
            elif confirmed_command == "LEFT":
                arduino.write(b'L')
            elif confirmed_command == "RIGHT":
                arduino.write(b'R')
            elif confirmed_command == "BACK":
                arduino.write(b'B')
            elif confirmed_command == "STOP":
                arduino.write(b'S')

            print(f"Gönderilen komut: {confirmed_command}")
            last_command = confirmed_command

    # Görselleştir
    cv2.putText(image, f'Command: {confirmed_command}', (10, 30),
                cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0, 255), 2)
    cv2.imshow('El Tespiti', image)
```

```
    if cv2.waitKey(1) & 0xFF == 27:
        break

cap.release()
cv2.destroyAllWindows()
arduino.close()
```

## 2 ) Arduino

Arduino is an open-source, low-cost, and easily programmable microcontroller platform used for developing electronic systems and physical computing applications. It mainly consists of an Arduino board (hardware) and the Arduino IDE (software) used to program the board.

Arduino boards include digital and analog input/output pins, a microcontroller chip, a USB interface, and sometimes additional components like voltage regulators. This allows easy interaction with sensors, motors, LEDs, buttons, and many other components.

One of Arduino’s key advantages is its ease of programming and modularity. Users can write C/C++-based code using the Arduino IDE, and thanks to built-in libraries and examples, even complex hardware tasks can be achieved with just a few lines of code.

The Arduino platform is widely used in both education and professional prototyping. Students, makers, engineers, and artists use it in various applications such as robotic systems, smart home projects, IoT devices, and automation systems.

Thanks to its open-source nature, Arduino is supported by a large global community that constantly develops new boards, modules, and libraries—making it a powerful and accessible platform for anyone interested in electronics.

## Arduino Codes:

### 1) Hand station

```
#include <Wire.h>

#include <LiquidCrystal_I2C.h>

#include <SoftwareSerial.h>


LiquidCrystal_I2C lcd(0x27, 16, 2);

SoftwareSerial bluetooth(11, 10);   // RX, TX
```

```
String inputData = "";

int d1 = 0, d2 = 0, d3 = 0, d4 = 0;
float temp = 0.0, hum = 0.0;
unsigned long lastSwitch = 0;
bool showDistances = true;

void setup() {
  Serial.begin(9600);
  bluetooth.begin(9600);
  lcd.begin();
  lcd.backlight();
  lcd.setCursor(0, 0);
  lcd.print("Veri bekleniyor");
}

void loop() {
  while (bluetooth.available()) {
    char c = bluetooth.read();
    if (c == '\n') {
      parseData(inputData);
      inputData = "";
    } else {
      inputData += c;
    }
  }

  if (millis() - lastSwitch > 2000) {
```

```
    lcd.clear();
    if (showDistances) {
      lcd.setCursor(0, 0);
      lcd.print("D1:");
      lcd.print(d1);
      lcd.print(" D2:");
      lcd.print(d2);
      lcd.setCursor(0, 1);
      lcd.print("D3:");
      lcd.print(d3);
      lcd.print(" D4:");
      lcd.print(d4);
    } else {
      lcd.setCursor(0, 0);
      lcd.print("Temp: ");
      lcd.print(temp);
      lcd.print("C");
      lcd.setCursor(0, 1);
      lcd.print("Hum : ");
      lcd.print(hum);
      lcd.print("%");
    }
    showDistances = !showDistances;
    lastSwitch = millis();
  }
}

void parseData(String data) {
  d1 = getValue(data, "D1:");
```

```
  d2 = getValue(data, "D2:");
  d3 = getValue(data, "D3:");
  d4 = getValue(data, "D4:");
  temp = getFloatValue(data, "T:");
  hum = getFloatValue(data, "H:");
}

int getValue(String data, String label) {
  int idx = data.indexOf(label);
  if (idx == -1) return 0;
  int endIdx = data.indexOf(',', idx);
  String val = data.substring(idx + label.length(), endIdx == -1 ? data.length() :
endIdx);
  return val.toInt();
}

float getFloatValue(String data, String label) {
  int idx = data.indexOf(label);
  if (idx == -1) return 0;
  int endIdx = data.indexOf(',', idx);
  String val = data.substring(idx + label.length(), endIdx == -1 ? data.length() :
endIdx);
  return val.toFloat();
}
```

## 2) Car

```
#include <SoftwareSerial.h>
#include <DHT.h>

#define DHTPIN 12
#define DHTTYPE DHT11
DHT dht(DHTPIN, DHTTYPE);

int sol_i = 8 ;
int sol_g = 7 ;

int sag_i = 9 ;
int sag_g = 10 ;
```

```
int sol_pwm = 11 ;
int sag_pwm = 6 ;

int hiz  = 220 ;

SoftwareSerial bluetooth(2, 3); // RX, TX

int echoPins[] = {A0, A2, A4, 4};
int trigPins[] = {A1, A3, A5, 5};
long durations[4];
int distances[4];

int data ;

void setup() {
  Serial.begin(9600);
  bluetooth.begin(9600);
  dht.begin();

  pinMode(sol_i , OUTPUT);
  pinMode(sol_g , OUTPUT);
  pinMode(sag_i , OUTPUT);
  pinMode(sag_g , OUTPUT);

  pinMode(sol_pwm , OUTPUT);
  pinMode(sag_pwm , OUTPUT);

  for (int i = 0; i < 4; i++) {
```

```
    pinMode(trigPins[i], OUTPUT);
    pinMode(echoPins[i], INPUT);
  }

  digitalWrite(sol_i, 0);
  digitalWrite(sol_g, 0);
  digitalWrite(sag_i, 0);
  digitalWrite(sag_g, 0);

  analogWrite(sol_pwm, 0);
  analogWrite(sag_pwm, 0);

}

int measureDistance(int trigPin, int echoPin) {
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  long duration = pulseIn(echoPin, HIGH);
  int distance = duration * 0.034 / 2;
  return distance;
}

void loop() {
  for (int i = 0; i < 4; i++) {
    distances[i] = measureDistance(trigPins[i], echoPins[i]);
  }
```

```
float temp = dht.readTemperature();
float hum = dht.readHumidity();

bluetooth.print("D1:");
bluetooth.print(distances[0]);
bluetooth.print(",D2:");
bluetooth.print(distances[1]);
bluetooth.print(",D3:");
bluetooth.print(distances[2]);
bluetooth.print(",D4:");
bluetooth.print(distances[3]);
bluetooth.print(",T:");
bluetooth.print(temp);
bluetooth.print(",H:");
bluetooth.println(hum);

delay(100);


data = Serial.read();

if (distances[1] < 20)
{
  digitalWrite(sol_i, 0);
  digitalWrite(sol_g, 1);
  digitalWrite(sag_i, 0);
  digitalWrite(sag_g, 1);
```

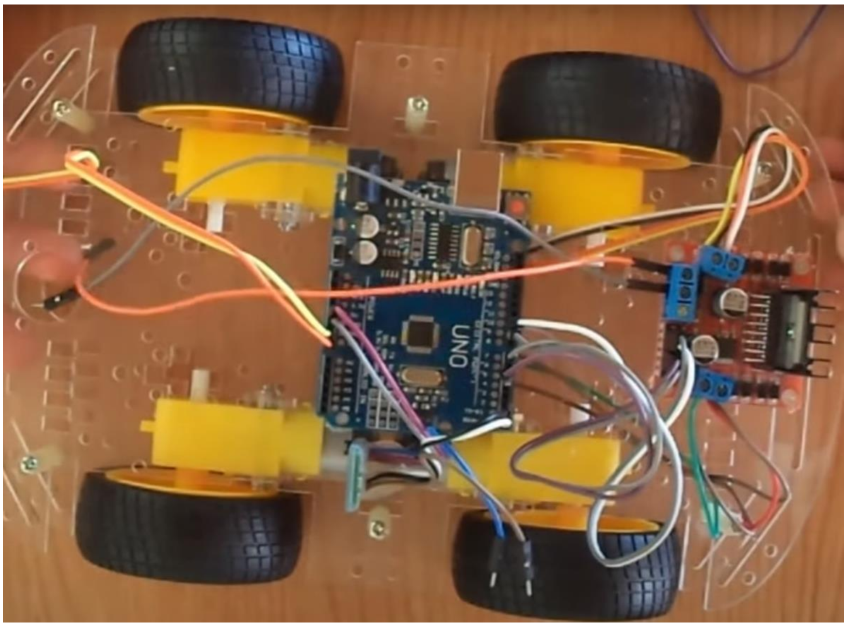
```
  analogWrite(sol_pwm, hiz);
  analogWrite(sag_pwm, hiz);
  delay(1000);
  digitalWrite(sol_i, 0);
  digitalWrite(sol_g, 0);
  digitalWrite(sag_i, 0);
  digitalWrite(sag_g, 0);

  analogWrite(sol_pwm, hiz);
  analogWrite(sag_pwm, hiz);
  delay(100);
}
if (distances[3] < 20)
{
  digitalWrite(sol_i, 1);
  digitalWrite(sol_g, 0);
  digitalWrite(sag_i, 1);
  digitalWrite(sag_g, 0);

  analogWrite(sol_pwm, hiz);
  analogWrite(sag_pwm, hiz);
  delay(1000);
  digitalWrite(sol_i, 0);
  digitalWrite(sol_g, 0);
  digitalWrite(sag_i, 0);
  digitalWrite(sag_g, 0);

  analogWrite(sol_pwm, hiz);
  analogWrite(sag_pwm, hiz);
```

```
}


if (data == 'F')
{

  digitalWrite(sol_i, 1);
  digitalWrite(sol_g, 0);
  digitalWrite(sag_i, 1);
  digitalWrite(sag_g, 0);

  analogWrite(sol_pwm, hiz);
  analogWrite(sag_pwm, hiz);
  delay(1000);
  digitalWrite(sol_i, 0);
  digitalWrite(sol_g, 0);
  digitalWrite(sag_i, 0);
  digitalWrite(sag_g, 0);

  analogWrite(sol_pwm, hiz);
  analogWrite(sag_pwm, hiz);
  delay(100);
}
if (data == 'B')
{

  digitalWrite(sol_i, 0);
  digitalWrite(sol_g, 1);
  digitalWrite(sag_i, 0);
```

```
  digitalWrite(sag_g, 1);

  analogWrite(sol_pwm, hiz);
  analogWrite(sag_pwm, hiz);
  delay(1000);
  digitalWrite(sol_i, 0);
  digitalWrite(sol_g, 0);
  digitalWrite(sag_i, 0);
  digitalWrite(sag_g, 0);

  analogWrite(sol_pwm, hiz);
  analogWrite(sag_pwm, hiz);
  delay(100);
}
if (data == 'L')
{

  digitalWrite(sol_i, 0);
  digitalWrite(sol_g, 1);
  digitalWrite(sag_i, 1);
  digitalWrite(sag_g, 0);

  analogWrite(sol_pwm, hiz);
  analogWrite(sag_pwm, hiz);
  delay(1000);
  digitalWrite(sol_i, 0);
  digitalWrite(sol_g, 0);
  digitalWrite(sag_i, 0);
  digitalWrite(sag_g, 0);
```

```
    analogWrite(sol_pwm, hiz);
    analogWrite(sag_pwm, hiz);
    delay(100);
  }

  if (data == 'R')
  {

    digitalWrite(sol_i, 1);
    digitalWrite(sol_g, 0);
    digitalWrite(sag_i, 0);
    digitalWrite(sag_g, 1);

    analogWrite(sol_pwm, hiz);
    analogWrite(sag_pwm, hiz);
    delay(1000);
    digitalWrite(sol_i, 0);
    digitalWrite(sol_g, 0);
    digitalWrite(sag_i, 0);
    digitalWrite(sag_g, 0);

    analogWrite(sol_pwm, hiz);
    analogWrite(sag_pwm, hiz);
    delay(100);
  }
}
```

## PRELIMINARY DESIGN




## DIFFICULTIES ENCOUNTERED

### 1) Simplicity of the Project

The project was very plain and simple in its initial form

#### Solution:

We decided to add sensors to the vehicle. We got help from our instructor to choose the sensors that are suitable for our project and purpose, and we decided to put temperature, humidity and distance sensors on the vehicle also decided to display the data provided by the sensors to the user via the LCD screen on an external handheld station. I provided communication with the handheld station using the HC-05 Bluetooth device

### 2) Serial communication with cable

Although this idea seemed logical at first, after developing the prototype I realized that it was a serious mistake. After deciding to integrate the sensors into the vehicle, I observed that the mess of cables not only disrupted the overall order of the vehicle but also significantly restricted its mobility. This situation revealed that the design needed to be change.

#### Solution:

I decided to use Bluetooth communication and integrated the HC-06 model into the project.

## 3) Material quality

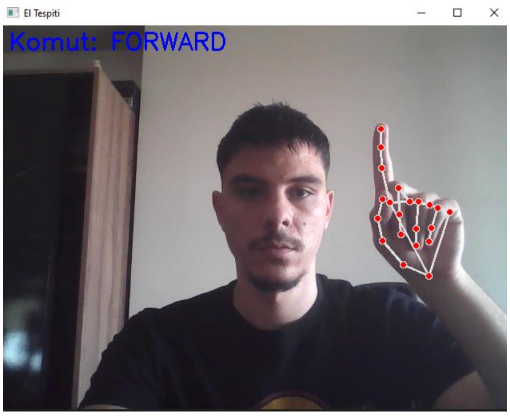
The material of the chassis part of the vehicle was very bad, cracks occurred at the slightest strain, and although the chassis had a simple design, there were many unnecessary cable entry holes.

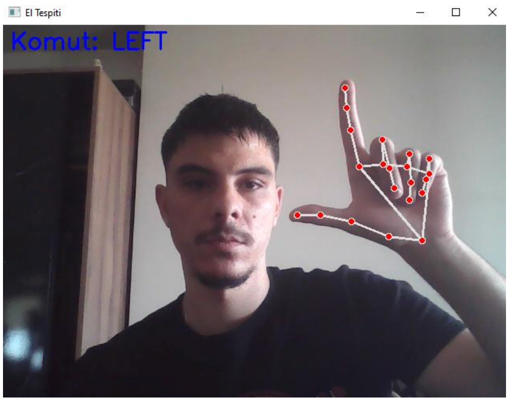
## Solutions:

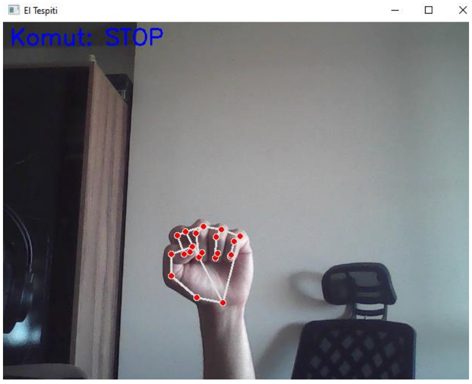
The vehicle chassis was redesigned. Instead of 2 chassis, a single chassis was used. The chassis was made of plexy material and shaped with laser cutting. This made our vehicle's chassis simpler and stronger than before.

## 4) Software problems

I was experiencing problems with defined hand movements, the movements were detected at very precise levels, with a slight hand slip, it either did not detect the command or entered the wrong command. Another problem I encountered was that it detected the commands one after the other and put the vehicle in an infinite loop. The vehicle was performing the commands properly, but the commands sent were that fast.




It suppost to write ‘’

BACK’’

## Solutions:

Various changes have been made to the defined movements. The main purpose of these changes is to enable the system to perceive environmental conditions more sensitively. In this direction, the detection conditions required for triggering movements have been increased; thus, the possibility of the system reacting to false positives has been minimized.

In addition, an important software update has been made to prevent the vehicle from unintentionally entering an infinite loop in certain situations. In this context, a 0.5 second delay mechanism has been added after the system perceives a command so that the action corresponding to that command is performed. This delay prevents the repeated triggering of commands, thus increasing the stability and reliability of the system.

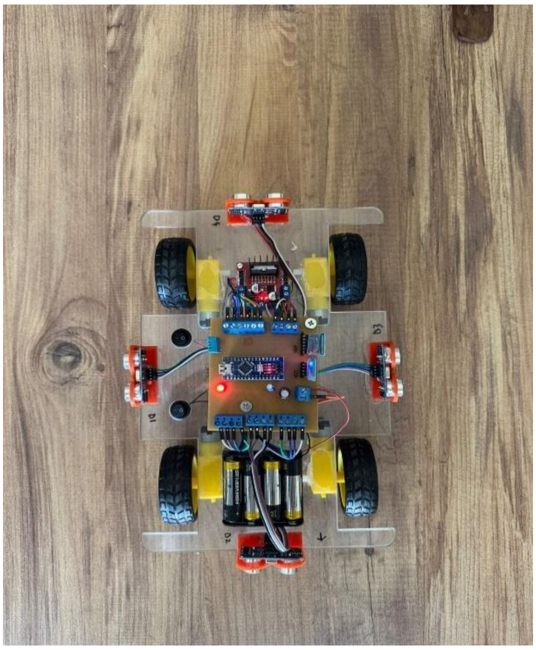
## 5) Design Problem

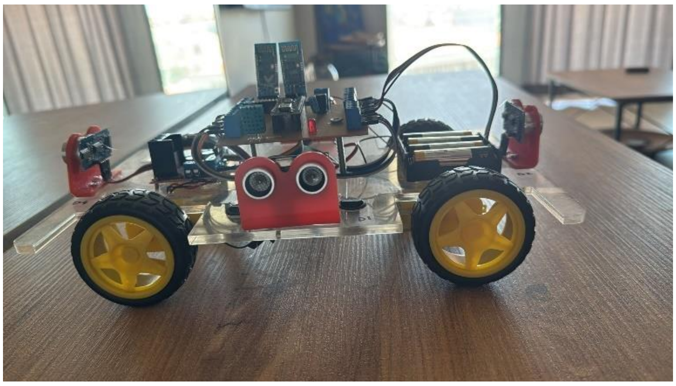
At the beginning, I connected the 2 front and 2 rear engines to each other in reverse so that the car could turn left and right comfortably, I thought the right wheels would go backwards while the left wheels were going forward, but I didn't get the efficiency I wanted

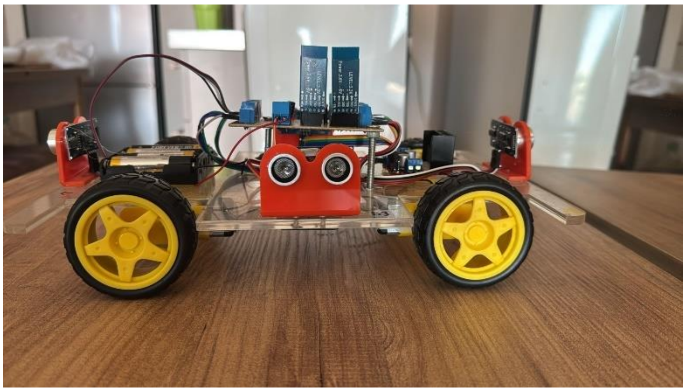
## Solutions:

Disassembled the motors, connected them all the same way and arranged the arduino functions

## FINAL RESULT








D1:66 D2:136
D3:125 D4:791


Temp: 30.50C
Hum : 43.00%


D1:68 D2:87
D3:124 D4:791

## COST

| Item | Quantity | Unit Price | Total Price (TL) |
|---|---|---|---|
| L298N | 1 | 84.00 | 84.00 |
| Arduino Nano | 2 | 176.81 | 353.62 |
| TT Motor & Wheels | 1 | 500.00 | 500.00 |
| HCSR-04 Ultrasonic Sensor | 4 | 62.49 | 249.96 |
| HC-05 Bluetooth Module | 2 | 186.50 | 373.00 |
| HC-06 Bluetooth Module | 1 | 184.07 | 184.07 |
| 100 μF Capacitor | 1 | 75.08 | 75.08 |
| 10 μF Capacitor | 1 | 1.45 | 1.45 |
| 100 nF Capacitor | 1 | 8.72 | 8.72 |
| Terminal Block (Klemens) | 1 | 15.00 | 15.00 |
| 330Ω Resistor | 50 | 7.26 | 7.26 |
| 18650 Battery | 1 | 150.00 | 150.00 |
| 18650 Battery Holder | 1 | 32.00 | 32.00 |
| Plexiglass Chassis | 1 | 500.00 | 500.00 |
| Distance Sensor Holder | 4 | 12.00 | 48.00 |
| 16x2 LCD Display | 1 | 124.92 | 124.92 |
| 9V Battery | 1 | 65.00 | 65.00 |
| 9V Battery Clip | 1 | 4.84 | 4.84 |
| Plastic Enclosure Box | 1 | 250.00 | 250.00 |
| Jumper Cable | 50 | 27.09 | 27.09 |

**Total Price: 3,048.01 TL**

# Background

## SolidWorks

During my academic coursework, I received training in SolidWorks, focusing on both 3D modeling and simulation techniques. This instruction provided me with a strong foundation in computer-aided design (CAD), as well as in performing structural and thermal simulations. In particular, I learned how to design mechanical components, create detailed assemblies, and analyze their physical behavior under various conditions using tools such as static stress analysis and thermal simulations.

These skills were directly applied in the current project, particularly in the design and analysis of the chassis. I utilized SolidWorks to model the chassis structure, ensuring accurate representation of all key dimensions and components. Furthermore, I performed static analysis to evaluate the structural integrity of the chassis under expected loads, and thermal analysis to assess heat distribution and potential temperature-related stress points. The knowledge gained through this training enabled a data-driven, optimized approach to mechanical design, contributing to the overall robustness and reliability of the system.

## Sofware

I took my first step into software development during my university studies by learning C++. This experience sparked a strong interest in programming and motivated me to continue learning. After completing the course, I began studying Python independently. To advance further, I enrolled in a 6-month bootcamp where I strengthened my Python skills and gained practical experience in computer vision using libraries like OpenCV. This journey allowed me to build a solid technical foundation and confirmed my enthusiasm for the software field.

## Electricity

I took my first step into this field during university through an electrical circuits course, where I gained essential theoretical knowledge. This project has given me the opportunity to apply those concepts in practice and develop my skills in a hands-on setting. It allowed me to bridge the gap between theory and real-world application, contributing to both my technical understanding and practical experience.

## KAYNAKÇA

**https://maker.robotistan.com/arduino-dersleri-10-16x2-lcd-ekran/**

https://pypi.org/project/mediapipe/

https://mediapipe.dev

https://projecthub.arduino.cc/

https://www.youtube.com/watch?v=D97WhJh_WqQ&list=PLjusLCxPk1B9vcstOjzLpDqPDj0VLjBMD

https://www.youtube.com/watch?v=-LeuA7jIx8E&list=PLpmnLzMgTTpAucGwBMaLhcUOy60FN5-T8

https://www.youtube.com/watch?v=R7JZqruGmoU

https://github.com/topics/hand-tracking

https://www.theengineeringprojects.com/

https://maker.robotistan.com/hc05-hc06-bluetooth-uart-modulleri-kullanimi/

https://guraysonugur.aku.edu.tr/2023/01/04/goruntu-isleme-python-seri-port-haberlesme-ve-goruntu-isleme-ile-kullanimi/

https://www.ahmetkemalyildiz.com.tr/9-arduino-ile-mesafe-sensoru-kullanimi/