

South Valley University, Faculty of Computers and Information

2023-2024

Smart System for Visually Impaired People (SSVIP)

# Team Members:

**1 -**  Mohamed Nasser Mohamed Khodary

**2 -** Mahmoud Mohamed Nagi

**3 -** Ziad Abdelmohsen Mohamed

**4 -** Yasser Ramdan

**Supervisor: 1 - DR. Nagwa**

**2 - Dr. Esraa**

# Acknowledgement

Team members would like to express our deep appreciation and gratitude to South Valley University and all those who support and provide us to complete this project.

Special gratitude to our supervisors Dr. Nagwa and for their contribution and help to coordinate our project. In addition, the team members would like to extend thanks to South Valley University and the College of Computing and Information for supporting us in this project by providing the components, appropriate labs, and classes.

Finally, no words can describe our huge gratitude and appreciation to our parents and family for their tremendous encouragement and support throughout this challenging and great journey to complete this project.

# Abstract

This project consists of two parts, which are smart wearable systems and smart glasses, under one name which is the Smart System for Visually Impaired People (SSVIP).

project aims to enable the visually impaired to coexist with daily life normally; it provides them - through embedded systems and image processing - the feature of object detection, searching for certain things within their surroundings or reading a book by voice communication with the system, the system also guides the user on the way until he reaches his destination, also system has protection accessories linked to program to alert the user before Hitting obstacles or An accident occurs.

The system also aims to facilitate communication between the user and his companion.

we aim to make the blind person live a normal life, so that he can do what he wants at the time he wants without the help of anyone.

# Table of Contents

[Acknowledgement I](#_TOC_250061)

[Abstract II](#_TOC_250060)

[Table of Contents III](#_TOC_250059)

[List of Figures V](#_TOC_250058)

[List of Tables VI](#_TOC_250057)

[CH1:Introduction 1](#_TOC_250056)

* 1. [Motivations 1](#_TOC_250055)
  2. [Problem Description 1](#_TOC_250054)
  3. [objectives… 2](#_TOC_250053)

[CH2:Background 3](#_TOC_250052)

[2.1 - History of Blind Helpers Technologies… 3](#_TOC_250051)

[CH3: Smart System for Visually Impaired People (SSVIP)… 5](#_TOC_250050)

* 1. [Introduction 5](#_TOC_250049)
  2. [Block Diagram 8](#_TOC_250048)
  3. Advantages 10
  4. Budget… 10

[CH4: SSVIP Software… 11](#_TOC_250047)

4.1.1 Embedded systems 11

* + 1. [Embedded C 12](#_TOC_250046)
    2. [C language… 13](#_TOC_250045)
  1. [Operating System 16](#_TOC_250044)
     1. [Linux operating system 19](#_TOC_250043)

[4.2.1.2. Programming language used to develop systems that use Linux operating systems 20](#_TOC_250042)

* 1. [Object Detection 21](#_TOC_250041)
     1. [Object Detection vs. Image Classification 21](#_TOC_250040)
     2. [Object detection Algorithms 22](#_TOC_250039)
        1. [R-CNN (Region-based Convolutional Neural Network) 23](#_TOC_250038)
        2. [Fast R-CNN… 26](#_TOC_250037)
        3. [SSD (Single Shot MultiBox Detector) 29](#_TOC_250036)
        4. YOLO(You Only Look Once) 32
     3. [the model we choose to our task 34](#_TOC_250035)
        1. [speed 35](#_TOC_250034)
        2. [accuracy 35](#_TOC_250033)

[4.3.4. Open CV… 36](#_TOC_250032)

* 1. [OCR 38](#_TOC_250031)
     1. [Optical Character Recognition algorithm 39](#_TOC_250030)
     2. [Optical Character Recognition mechanism Steps 40](#_TOC_250029)

[4.5 the smart glasses activities 40](#_TOC_250028)

[CH5: SSVIP Hardware… 42](#_TOC_250027)

* 1. [Introduction 42](#_TOC_250026)
  2. Components 42
     1. [Smart Wearable System microcontroller 43](#_TOC_250025)
        1. [AVR 45](#_TOC_250024)
        2. common Avr types 45
        3. ATmega128 Board 46
        4. Atmega128 Programming 48
     2. [Sensors 49](#_TOC_250023)
        1. [Distance sensors 50](#_TOC_250022)
        2. [Ultrasonic 50](#_TOC_250021)
     3. [Buzzer 52](#_TOC_250020)
     4. [Vibration motor 54](#_TOC_250019)
     5. [GPS Chip 55](#_TOC_250018)
        1. [NEO 6M GPS Module… 55](#_TOC_250017)
        2. [Getting Location Data from GPS… 57](#_TOC_250016)
        3. [Interface GPS with AVR Microcontroller 59](#_TOC_250015)
     6. [GSM Chip 59](#_TOC_250014)
        1. [SIM 900A GSM Module 60](#_TOC_250013)
        2. [SIM 900A GSM Module with AVR Microcontroller 62](#_TOC_250012)
     7. [Smart glasses Microprocessor 63](#_TOC_250011)
        1. [Microprocessor 63](#_TOC_250010)
        2. [Raspberry Pi 64](#_TOC_250009)
        3. [Common Raspberry Pi Types… 65](#_TOC_250008)
        4. [Raspberry Pi 3 Model B 65](#_TOC_250007)
        5. [Raspberry Pi 3 Model B Board 67](#_TOC_250006)
     8. [Camera… 71](#_TOC_250005)

[5.2.9 Battery 72](#_TOC_250004)

[CH6: Conclusion and Future Plans 73](#_TOC_250003)

* 1. [Conclusion 73](#_TOC_250002)
  2. [Future Plans 73](#_TOC_250001)

[References 74](#_TOC_250000)

# List of Figures

Figure 2.1 -louis braille method. 3

Figure 2.2-first electronic travel system. 4

Fig 2.3-Smart Stick 4

Figure 3.1-Remote monitoring 5

Figure 3.2-Safety Alarm's Example 6

Figure 3.3-Object Detection Example 6

Figure 3.4-Coin Detection Example 7

Figure 3.5-OCR Scanner Example 7

Figure 3.6 Real-time. 8

Figure 4.1-Classification & Detection. 22

Figure 4.2-Stages of Object Detector. 22

Figure 4.3-One and Two Stages Detector 23

Figure 4.4-R-CNN Working process. 23

Figure 4.5-Faster R-CNN working process. 27

Figure 4.6-Architecture of SSD. 29

Figure 4.7- YOLO Working process. 32

Figure 4.8-Graph of Speed. 35

Figure 4.9-Graph of Accuracy 35

Figure 4.10-General OCR model. 39

Figure 4.11- OCR Mechanism Steps. 40

Figure 4.12- Object Detection camera view 41

Figure 4.13- Face Recognition camera view 41

Figure 4.14- OCR scanning 42

Figure 5.1-Difference between Microcontroller & microprocessor 43

Figure 5.2-Component of Microcontrollers. 44

Figure 5.3-Board ATmega128. 46

Figure 5.4-Atmega128Pins. 48

Fig 5.5. Ultrasonic 50

Figure 5.6-Ultrasonic sensor working principle 51

Fig 5.7-Measuring Distance of Ultrasonic sensorv 51

Fig 5.8-Measuring Distance of Ultrasonic sensor Example 52

Figure 5.9-Buzzer. 52

Figure 5.10-Buzzer working principle 53

Figure 5.11-Circuit Diagram of buzzer. 53

Figure 5.12-Vibration motor. 54

Figure 5.13-GPS module 56

Figure 5.14- ublox NEO-6M GPS pins. 56

Figure 5.15- output GPS data. 57

Figure 5.16- (Global Positioning System Fix Data) 58

Figure 5.17-GPS final output. 59

Figure 5.18-SIM 900A GSM Module 60

Figure 5.19-SIM900A Pins. 61

Figure 5.20-GSM connected with AVR 62

Figure 5.21-Raspberry pi module 64

Figure 5.22-Raspberry Pi 3 Model B 65

Figure 5.23-Raspberry pi 3 model B Details. 67

Figure 5.24-5MP Raspberry Pi Camera Module Rev 1.3 71

Figure5.24-Camera connected to Microprocessor 71

# List of Tables

Table 1-Programming languages comparison 14

Table 2-C vs Embedded C 15

Table 3-Windows vs macOs vs Linux 17

Table 4-Advantages vs Disadvantages of One and Two Stages D Detector 36

Table 5-Microcontroller vs Microprocessor 43

Table 6-Distance Sensors 50

Table 7- ublox NEO-6M GPS pins Description 56

Table 8-ublox NEO-6M GPS Specifications 57

Table 9-Difference between Raspberry Pi Models 65

Table 10-5MP Raspberry Pi Camera Module Rev 1.3 Description 71

# CH1:Introduction

## Motivations

This project consists of two parts, which are smart wearable systems and smart glass, Under one name which is the Smart System for Visually Impaired People (SSVIP).

The development of a Smart System for Visually Impaired People(SSVIP)is driven by the need to improve their quality of life, independence, and access to information.Visual impairment can present significant challenges in daily life, such as navigating unfamiliar surroundings and accessing information that is usually displayed visually.

Wearable smart systems have the potential to address these challenges by providing real- time assistance and feedback through the use of sensors, cameras, microphones, and haptic feedback. By using these devices, visually impaired individuals can navigate their environment more easily, identify objects, and access information more effectively.

The motivations for this project are twofold. Firstly, we aim to improve the lives of visually impaired individuals by designing and developing a wearable smart system that meets their specific needs and requirements. Our goal is to create a device that is user- friendly, portable, and customizable, making it accessible to a broad range of users.

Secondly, we believe that wearable smart systems have the potential to transform the way that visually impaired individuals interact with the world around them. By providing real- time feedback and assistance, these systems can help to break down barriers and enable greater independence and participation in daily life.

Ultimately, our hope is that the development of a wearable smart system for visually impaired people will have a positive impact on the lives of millions of individuals worldwide, improving their ability to navigate their environment, access information, and live independently.

## Problem Description

Visual impairment is a significant challenge faced by millions of people worldwide. Individuals with visual impairments often face difficulties in performing daily activities, such as navigating through unfamiliar environments, identifying objects, and accessing

information that is usually displayed visually. This can lead to a loss of independence and a reduced quality of life.

Traditional solutions to address these challenges, such as the use of canes or guide dogs, have limitations and may not be suitable for all individuals with visual impairments. As a result, there is a growing need for innovative solutions that can provide real-time assistance and feedback to visually impaired individuals.

Wearable smart systems have emerged as a promising solution to address these challenges. These devices use a combination of sensors, cameras, microphones, and haptic feedback to provide real-time assistance and feedback to the user. However, the development of wearable smart systems for visually impaired individuals is still in its early stages, and there is much room for improvement and innovation.

The problem that this project aims to address is the need for a wearable smart system that is specifically designed to meet the needs of visually impaired individuals. This system should be user-friendly, portable, and customizable, making it accessible to a broad range of users. By developing such a system, we hope to improve the quality of life and independence of visually impaired individuals and provide them with a tool that can help them navigate their environment, identify objects, and access information more easily.

## objectives

The primary objective of this project is to design and develop a wearable smart system that can assist visually impaired individuals in their daily lives. The system should be user- friendly, portable, and customizable, making it accessible to a broad range of users.

Specifically, the objectives of this project are:

1. To identify the needs and requirements of visually impaired individuals through user research and feedback.
2. To design and develop a wearable smart system that meets the needs and requirements of visually impaired individuals.
3. To integrate a range of sensors, cameras, microphones, and haptic feedback into the wearable smart system to provide real-time assistance and feedback to the user.
4. To customize the system to the specific needs and preferences of the user.
5. To test and evaluate the effectiveness of the system in real-world scenarios with visually impaired individuals.
6. To refine and improve the system based on user feedback and testing.

By achieving these objectives, we hope to develop a wearable smart system that can improve the quality of life and independence of visually impaired individuals. We believe that this system has the potential to transform the way that visually impaired individuals interact with the world around them and provide them with a tool to navigate their environment, identify objects, and access information more easily.

# CH2:Background

## 2.1 - History of Blind Helpers Technologies

1. louis braille method

The use of technology to assist individuals with visual impairments dates back to the mid- 19th century when Louis Braille invented the Braille system for reading and writing. This system revolutionized the way that individuals with visual impairments could access information and remains in use today.

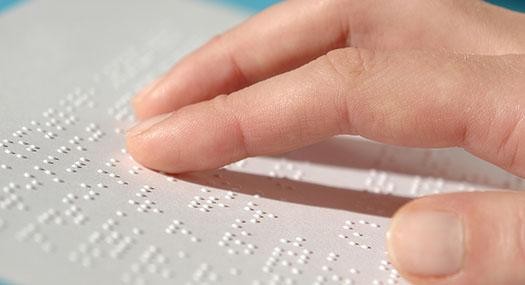


Figure 2.1 -louis braille method

1. first electronic travel aids

In the 1960s, the first electronic travel aids were developed. These devices used ultrasound sensors to detect obstacles and provide feedback to the user through audible tones.

However, these devices were large and bulky and were not widely adopted.

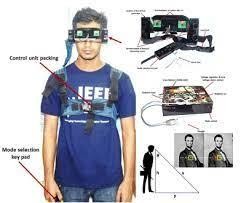


Figure 2.2-first electronic travel system

1. Smart Stick using ARDUINO and electronic sensors

using this smart blind stick. a visually impaired person can walk without anyone help.

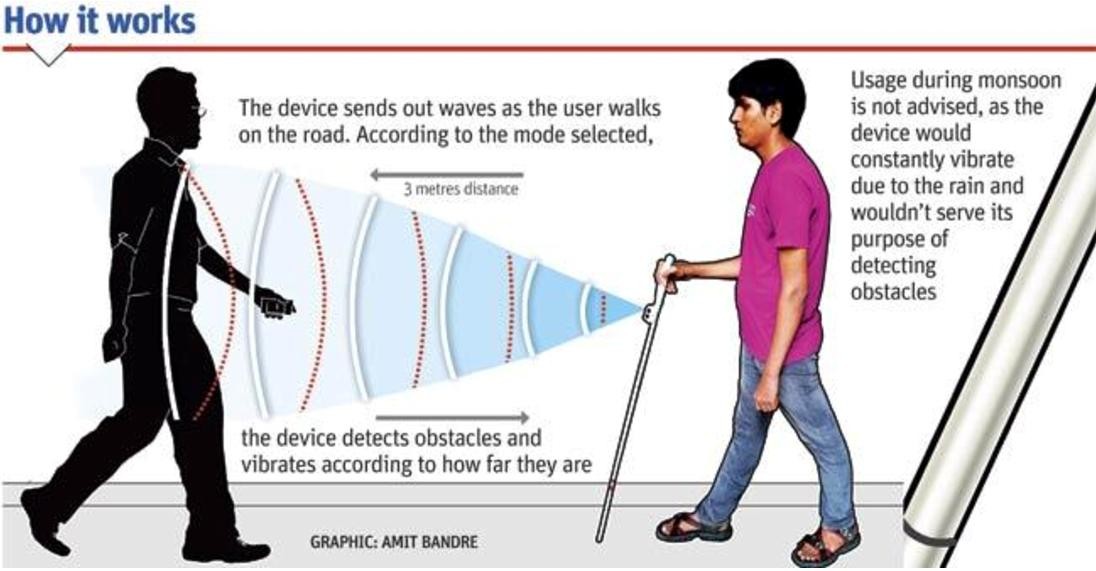


Fig 2.3-Smart Stick

The Smart blind Stick automatically detects the obstacle in front of the person by measuring the distance using Ultrasonic controlled by a microcontroller kit like Arduino,

then it gives him a response by vibrating the stick or a warning sound. this technology has been a great development in this field, but it still lacks a lot of capabilities, as it only tells its user if there is something in front of him, but it does not tell him what that things is in addition , it is a large sized tool that requires effort from the user to hold it and move it everywhere.

**The development of wearable smart systems for visually impaired individuals is still in its early stages, and there is much room for improvement and innovation. This project aims to contribute to this field by designing and developing a wearable smart system that meets the needs and requirements of visually impaired individuals, improving their quality of life and independence.**

# CH3: Smart System for Visually Impaired People (SSVIP)

## Introduction

Smart system to help visually impaired persons (VIPs) walk by themselves through the streets, navigate in public places, seek assistance and Reading and learning about things and people.

Trying to reach our goal we made a system have many features:

### - Remote monitoring of the user

Remote monitoring of the user can be an important feature of a wearable smart system for visually impaired individuals. This feature allows caregivers, family members, or medical professionals to monitor the user's activities and health remotely, providing an added layer of safety and security.



Figure 3.1-Remote monitoring

### - The Safety Alarms

Alarms using sound and vibration are a common feature of wearable smart systems for visually impaired individuals. These alarms are designed to provide feedback to the user when they are approaching an obstacle or danger, helping them to navigate their environment more safely.

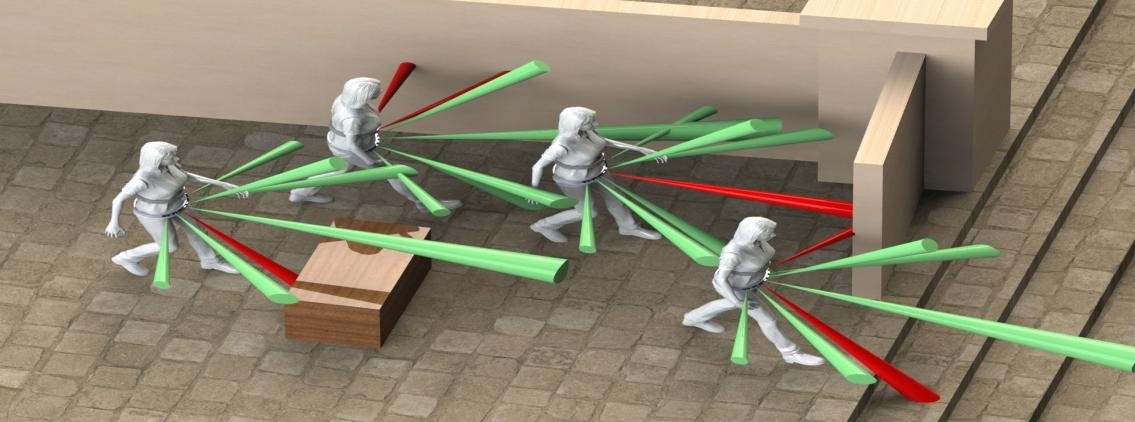


Figure 3.2-Safety Alarm's Example

### - Object Detection

the function of this feature is to know the objects around the user by facing the real-life camera of the phone to the object or the area around.

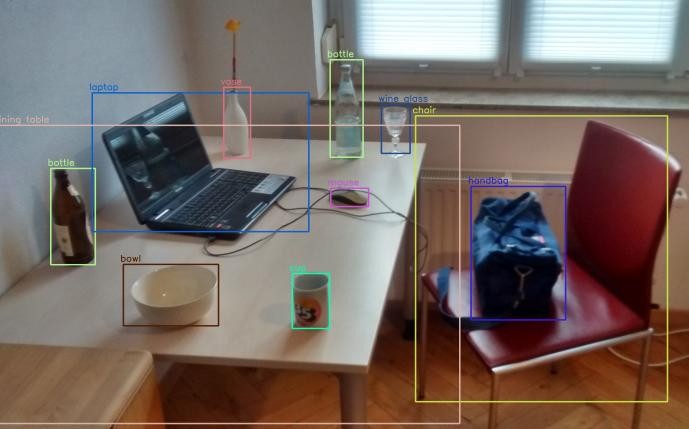


Figure 3.3-Object Detection Example

### Coin Detection

its function is to know the egyptian currency class by facing the real-life camera to the currency and the application will declare in voice output.



### OCR Scanner

Figure 3.4-Coin Detection Example

its function is to turn any text to voice by taking a photo by the mobile camera to the text and it will convert it to speech by the Chosen language.



Figure 3.5-OCR Scanner Example

### - Real-time operation

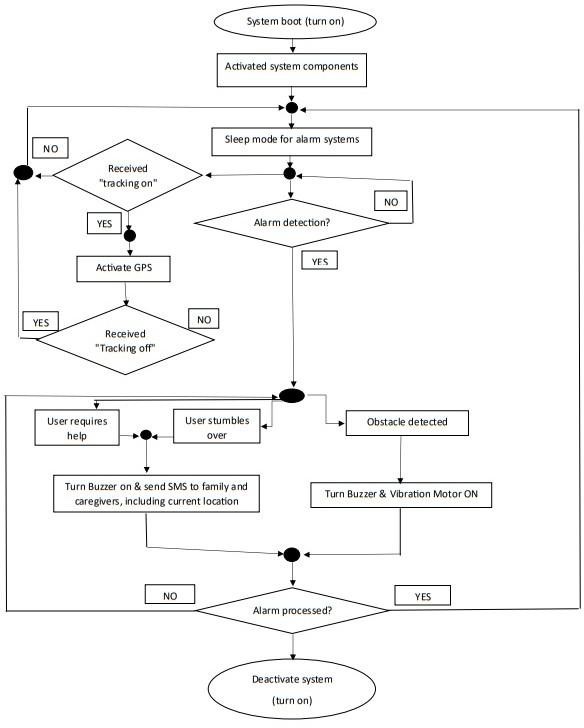
Real-time playback is an important aspect of a smart system for visually impaired people. Real-time operation means that the system can provide feedback and assistance to the user immediately, without any noticeable delay. This is necessary to ensure that the user can effectively navigate their environment and avoid obstacles.



# Block Diagram

Figure 3.6 Real-time

### Wearable Smart System

* + - **Smart glasses**
  1. **Advantages**
     + **Ease of communication between the user and the keeper:** By using GPS and GSM, users can call or send mobile messages to their locations by pressing a button only.
     + **More safety:** By using hardware device which detect anything facing the user while walking.
     + **Not costly:** By using inexpensive components in hardware device and glass.
     + **Make it easy for the user to read anything:** By using scanner and voice output, user can listen any text which he scanned by voice.
     + **Easy reach to anything he wants:** By using object detection.
     + **Ease of identifying people:** By using Face Recognition.
     + **Ease of dealing with cash:** Using coin detection feature.
  2. **Budget**

|  |  |  |
| --- | --- | --- |
| **Item** | **Quantity** | **Total price(LE)** |
| **Raspberry pi 3 model B** | **1** |  |
| **Raspberry camera** | **1** |  |
| **Atmega128** | **1** |  |
| **SIM800L GPRS GSM** | **1** |  |
| **NEO-6M GPS Module** | **1** |  |
| **Sport vest** | **1** |  |
| **ADXL345 Angel acceleration**  **sensor** | **1** |  |
| **Mini vibration motor** | **3** |  |
| **USBASP AVR Programmer** | **1** |  |
| **BUZZER** | **4** |  |
| **Breadboard** | **1** |  |
| **Ultrasonic sensor with holder** | **5** |  |
| **Flexible BreadBoard Jumber**  **Wiers** | **1** |  |
| **Push button** | **3** |  |
| **Glasses** | **1** |  |
| **Power bank** | **2** |  |
| **Headphones** | **1** |  |
| **Velcro Colorful Nylon Hook** | **1** |  |
| **Reinforced wood holder** | **1** |  |
|  |  |  |

# CH4: SSVIP Software

The project is divided into two parts: the smart Wearable system and the smart glasses, and each of them has its own software part.

## Wearable Smart System:

* 1. **Embedded systems**

This wearable smart system is considered a type of Embedded systems.

Embedded systems are computer systems that are designed with specific functions and are integrated into other devices or machines to perform dedicated tasks. They are different from general-purpose computing systems like desktops, laptops, and servers, which are designed to perform a wide variety of tasks. Here are some key differences between embedded systems and other systems:

Purpose: Embedded systems are designed for specific tasks, while general-purpose systems are designed to perform a wide variety of tasks.

Size: Embedded systems are typically smaller in size than general-purpose systems. They are often integrated into other devices, such as cars, appliances, and medical equipment.

Functionality: Embedded systems are designed to perform specific functions, such as controlling the temperature in a refrigerator or monitoring the speed of a car. General- purpose systems, on the other hand, can perform a wide variety of functions, such as word processing, gaming, and web browsing.

Operating system: Embedded systems often use lightweight operating systems that are optimized for the specific task they are performing. General-purpose systems typically use more complex operating systems, such as Windows or Linux.

User interface: Embedded systems often have a simple user interface, such as a display screen with a few buttons or a touchpad. General-purpose systems typically have more complex user interfaces, such as a graphical user interface (GUI).

Power consumption: Embedded systems are often designed to consume minimal power, as they may be battery-powered or have limited power sources. General-purpose systems as they are designed to perform more complex tasks. typically consume more power

Cost: Embedded systems are often designed to be cost-effective, as they are integrated into other devices and manufactured in large quantities. General-purpose systems can be more expensive, as they often require more advanced components and are designed for a broader range of applications.

## Embedded C

Embedded C is a variant of the C programming language that is specifically designed for programming embedded systems. Embedded systems are computer systems that are designed with specific functions and are integrated into other devices or machines to perform dedicated tasks. Here are some key features of Embedded C:

Low-level programming: Embedded C allows low-level programming, which means that programmers can access and manipulate the hardware directly, making it a good language for system programming and embedded systems.

Memory management: Embedded C allows programmers to manage memory

dynamically using functions such as malloc() and free(). This enables programs to allocate and deallocate memory at runtime, making it useful for programs with varying memory requirements.

Reduced size and complexity: Embedded C is designed to be more compact and efficient than standard C, with a reduced set of libraries and functions. This is important for embedded systems, which often have limited processing power and memory.

Interrupt handling: Embedded C provides built-in support for interrupt handling, which is important for real-time systems that need to respond quickly to external events.

Portability: Embedded C is designed to be highly portable, which means that programs written in Embedded C can be easily ported to different hardware platforms and operating systems.

Debugging: Embedded C programs can be debugged using a variety of tools, such as debuggers, profilers, and memory analyzers. These tools help programmers find and fix errors in their code.

Access to hardware: Embedded C provides programmers with direct access to the hardware of the embedded system, which means that they can control and manipulate the system's peripherals, such as timers, sensors, and memory, directly from the code. This is important for developing efficient and optimized code for embedded systems.

Real-time performance: Embedded C is designed to support real-time systems, which require fast response times to external events. Embedded C provides built-in support for interrupt handling, which allows the system to respond quickly to external events and perform tasks in real-time.

## C language

C is a high-level programming language that was first developed in the early 1970s by Dennis Ritchie at Bell Labs. It is a popular language for system programming, embedded systems, and general-purpose programming. Here are some key features of the C language:

Simple syntax: C has a simple and easy-to-learn syntax, which makes it a good language for beginners to learn programming.

Portability: C is a portable language, which means that programs written in C can be compiled and run on different platforms, including Windows, macOS, Linux, and many other operating systems.

Efficiency: C is a compiled language, which means that programs written in C are compiled into machine code, making them very efficient and fast.

Low-level programming: C allows low-level programming, which means that programmers can access and manipulate the hardware directly, making it a good language for system programming and embedded systems.

Standard library: C has a large standard library that provides many functions for performing common tasks, such as input/output, string manipulation, and memory management.

Preprocessor directives: C includes preprocessor directives, which are instructions to

the compiler to perform certain tasks, such as including header files or defining constants.

C is widely used in the development of operating systems, device drivers, embedded systems, compilers, and many other applications. It is also used in the development of other programming languages, such as C++, Java.

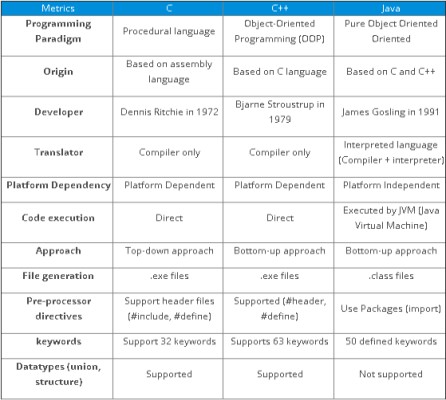


Table 1- Programming languages comparison

Structured programming: C supports structured programming, which means that programs can be organized into functions, loops, and conditional statements. This makes it easier to write and maintain large programs.

Memory management: C allows programmers to manage memory dynamically using functions such as malloc() and free(). This enables programs to allocate and deallocate memory at runtime, making it useful for programs with varying memory requirements.

Interoperability: C can be easily integrated with other programming languages, such as assembly language, C++, and Fortran. This makes it a useful language for writing libraries and other reusable code.

Debugging: C programs can be debugged using a variety of tools, such as debuggers, profilers, and memory analyzers. These tools help programmers find and fix errors in their code.

Difference between C language and Embedded C for C language

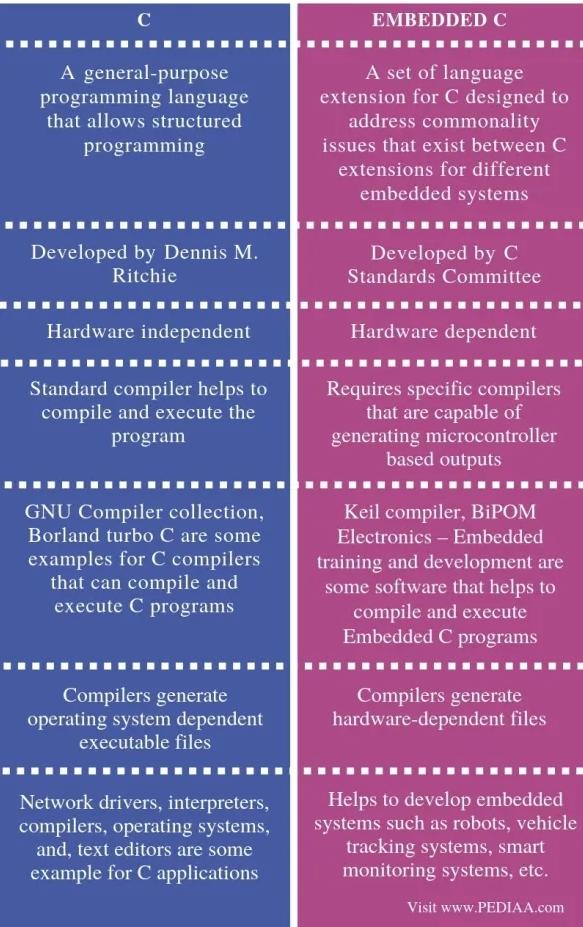


Table 2-C vs Embedded C

In the end, and because we need more features that are found in the Embedded C of C programming language, All parts of our device were coded using the Embedded C of C programming language.

## Smart glasses:

### Operating System

An operating system (OS) is a software that manages and controls the hardware and software resources of a computer or computing device. It acts as an intermediary between the user and the computer, providing an environment for executing applications and enabling efficient utilization of system resources.

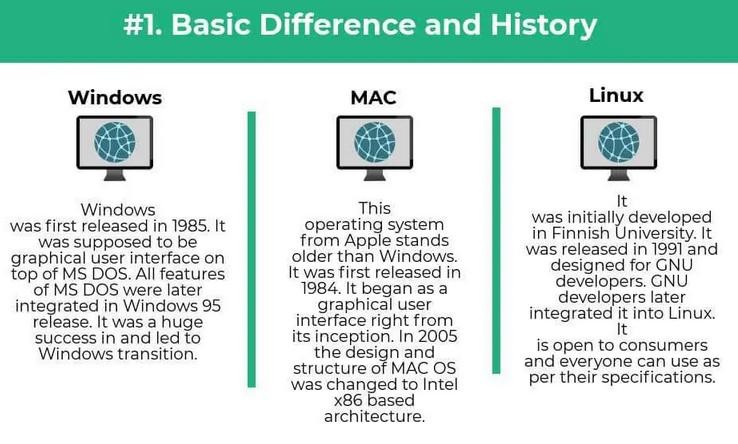
The most common operating systems used today are Windows, Linux and macOS.

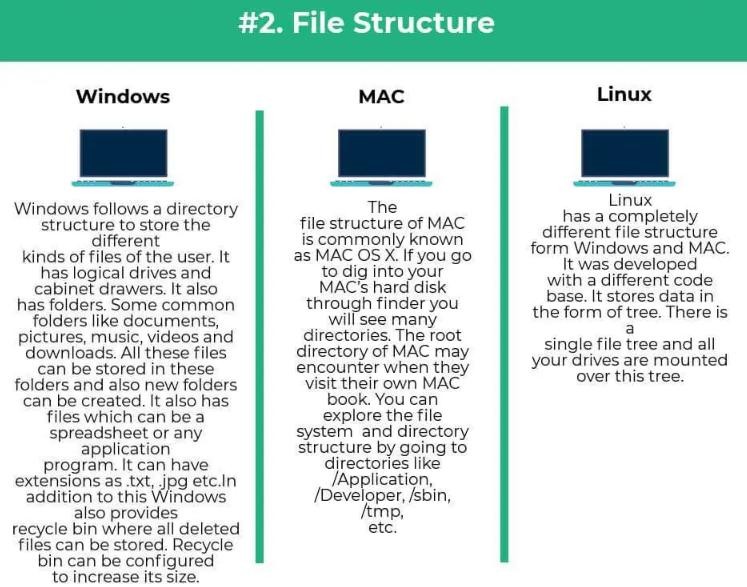
**Windows:** Windows is the dominant operating system for personal computers (desktops and laptops). It is widely used in homes, businesses, educational institutions, and government organizations. Windows offers a range of versions, with Windows 10 being the latest major release at the time of my knowledge cutoff.

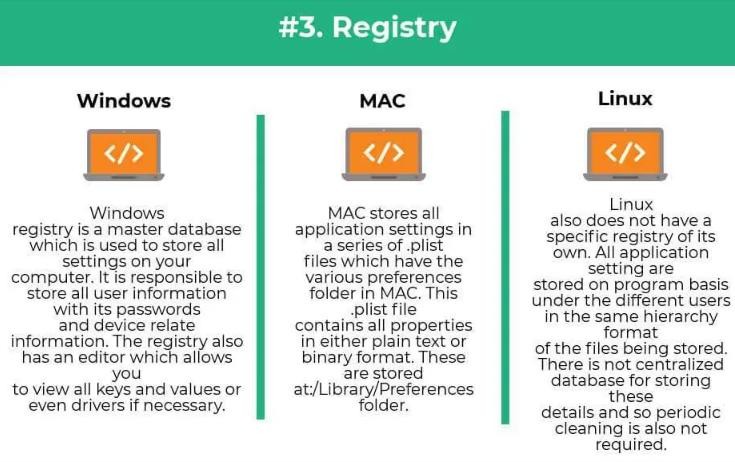
**Linux:** Linux is a popular open-source operating system used in various devices and environments, including desktops, servers, embedded systems, and supercomputers. Linux offers a high level of customization, security, and stability. It is widely used by developers, enthusiasts, and organizations for its flexibility and open-source nature.

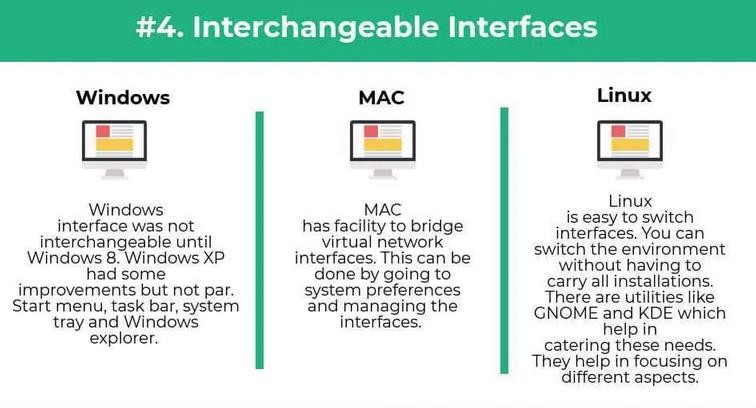
**macOS:** macOS, developed by Apple, is the operating system used on Macintosh computers. It is known for its integration with other Apple devices, intuitive user interface, and popularity among creative professionals, designers, and developers.

Difference between Windows, macOS and Linux









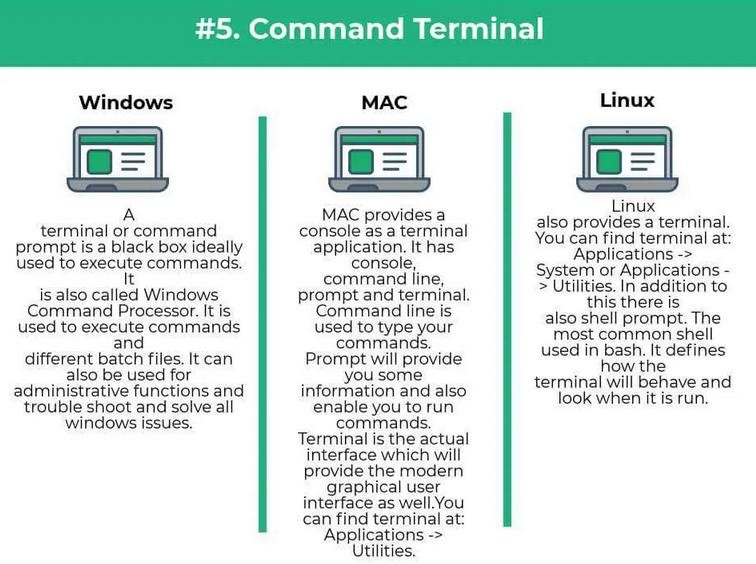


Table 3-Windows vs macOs vs Linux

In terms of these differences and features, we found that the **Linux** operating system is suitable for the system to be made.

## Linux operating system

Linux is a popular open-source operating system kernel that serves as the foundation for numerous Linux distributions. Developed by Linus Torvalds and first released in 1991, Linux has since gained widespread adoption and is used in a variety of computing devices, ranging from servers and desktop computers to embedded systems and mobile devices.

Here are some key features and characteristics of Linux:

* + - 1. Open Source: Linux is distributed under an open-source license, which means its source code is freely available. This allows users to view, modify, and distribute the code, promoting transparency, collaboration, and innovation.
      2. Stability and Reliability: Linux is known for its stability and reliability. It is designed to handle heavy workloads and can run continuously for long periods without issues. Linux-based systems are often preferred for critical applications and servers.
      3. Flexibility and Customization: Linux offers a high degree of flexibility and customization. Users can choose from various distributions, desktop environments, and software packages to create a tailored computing experience. Linux can be adapted to suit different hardware platforms and requirements.
      4. Security: Linux is renowned for its robust security features. Its design principles prioritize security, and its open-source nature enables a large community of developers to identify and fix vulnerabilities promptly. Linux-based systems often benefit from timely security updates and patches.
      5. Wide Hardware Support: Linux supports a wide range of hardware architectures, making it versatile and adaptable. It can run on diverse hardware platforms, from small embedded devices to powerful servers, enabling its use in various applications and industries.
      6. Software Ecosystem: Linux offers a vast ecosystem of open-source software applications and tools. Popular software packages and libraries are readily available, allowing developers to leverage existing solutions and customize them to their needs.

It's worth noting that Linux itself refers specifically to the operating system kernel, while various Linux distributions (such as Ubuntu, Fedora, Debian, and CentOS) package the kernel with additional software, drivers, and utilities to create complete operating systems for specific use cases.

Overall, Linux's open-source nature, stability, flexibility, and robustness make it a popular choice for a wide range of computing devices and applications.

## Programming language used to develop systems that use Linux operating systems

When developing systems that run on Linux operating systems, you have the flexibility to choose from a variety of programming languages based on your requirements and preferences. Here are some commonly used programming languages for developing software on Linux:

* + - * 1. C: C is a popular choice for system-level programming, including the development of operating systems, device drivers, and low-level utilities. It offers direct memory access, efficient code execution, and low-level hardware access.
        2. C++: C++ is an extension of the C programming language and provides additional features such as object-oriented programming. It is commonly used for developing high-performance applications, graphical user interfaces (GUIs), and system-level software.
        3. Python: Python is a versatile and widely adopted language known for its simplicity and readability. It offers extensive libraries and frameworks that make development faster and more efficient. Python is popular for scripting, web development, and automation tasks on Linux systems.
        4. Java: Java is a platform-independent language known for its "write once, run anywhere" approach. It is widely used for developing cross-platform applications, web services, and server-side software. Java applications run on the Java Virtual Machine (JVM) and can be deployed on Linux systems.

Since the workers on this part of the project, for them, the **Python**

programming language is the easiest and most appropriate, so it was chosen.

## Object Detection

Object detection is a computer vision technique that works to identify and locate objects within an image or video. Specifically, object detection draws bounding boxes around these detected objects, which allow us to locate where said objects are in (or how they move through) a given scene.

Object detection is commonly confused with image recognition, so before we proceed, it's important that we clarify the distinctions between them.

## Object Detection vs. Image Classification

Before we move on, let's clarify the distinction between image recognition and object detection.

**Image classification:** sends a whole image through a classifier (such as deep neural network) for it to spit out a tag. Classifiers take into consideration the whole image but don't tell you where the tag appears in the image.

**Object detection:** is slightly more advanced, it takes an image as input and produces one or more bounding boxes with the class label attached to each bounding box. these algorithms are capable enough to deal with multi-class classification and localization as well as to deal with the objects with multiple occurrences.



Figure 4.1-Classification & Detection

## Object detection Algorithms

Object detection generally based on deep learning, and for specific based on convolutional neural networks (CNNs).

Object detection generally is categorized into 2 stages:

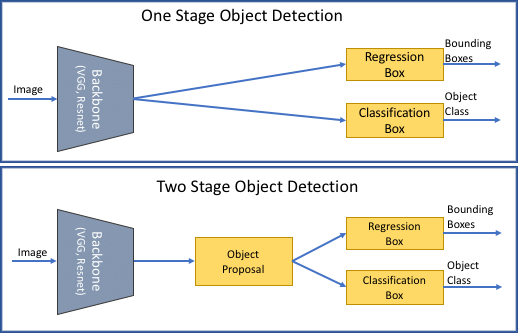
* One-stage object detectors.
* Two-stage object detectors.

Figure 4.2-Stages of Object Detector

Each object detection category has some algorithms to perform it, it classifies as:

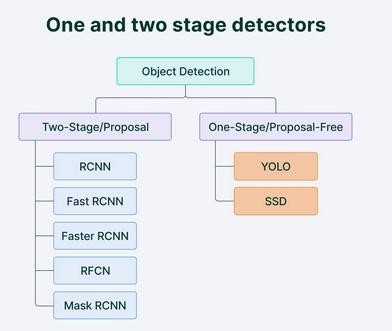


Figure 4.3-One and Two Stages Detector

## R-CNN (Region-based Convolutional Neural Network)

* 1. NN (Region-based Convolutional Neural Network) is a pioneering object detection framework that laid the foundation for modern object detection methods. It was introduced by Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik in 2014. R-CNN is a two-stage object detection approach that effectively combines region proposal generation and object classification.

To bypass the problem of selecting a huge number of regions, [Ross Girshick et al](https://arxiv.org/pdf/1311.2524.pdf). proposed a method where we use selective search to extract just 2000 regions from the image and he called them region proposals. Therefore, now, instead of trying to classify a huge number of regions, you can just work with 2000 regions.

* + - Working process of R-CNN

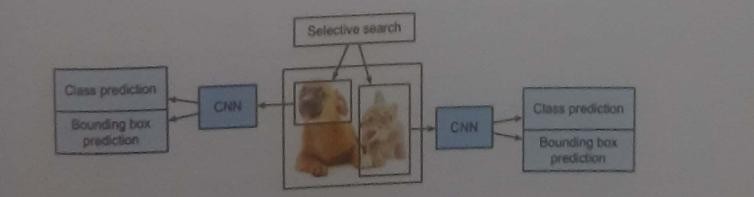


Figure 4.4-R-CNN Working process

1. Region Proposal:
   * R-CNN begins by generating a set of potential object regions in the input image using a region proposal algorithm like selective search.
   * The region proposal algorithm examines the image and identifies regions that are likely to contain objects based on various visual cues such as color, texture, and shape.
   * These proposed regions serve as candidate regions for further analysis.
2. Region of Interest (ROI) Extraction:
   * Each proposed region is cropped or extracted from the input image based on its coordinates.
   * These region proposals are then resized to a fixed size, typically a square shape, to ensure consistency for subsequent processing.
3. Feature Extraction:
   * The extracted region proposals are passed through a pre-trained convolutional neural network (CNN), such as VGG16 or ResNet.
   * The CNN processes each region proposal and extracts high-level features that capture important visual information from the regions.
   * The pre-trained CNN has learned to recognize general visual patterns from large-scale image classification tasks, which helps in capturing meaningful representations of objects.
4. Object Classification:
   * The features extracted from each region proposal are forwarded through a set of fully connected layers.
   * The fully connected layers perform object classification, determining whether the region contains an object or not.
   * Classifiers, such as softmax or sigmoid, produce class probabilities for each object category.
   * This step assigns a class label to each region proposal, indicating the predicted object category.
5. Bounding Box Regression:
   * In addition to object classification, R-CNN performs bounding box regression to refine the localization of objects.
   * The network predicts refined coordinates for the bounding boxes, adjusting their positions and sizes to better align with the actual objects.
   * This allows for more precise localization of the detected objects within the region proposals.
6. Non-Maximum Suppression (NMS):
   * After the object classification and bounding box regression, a post- processing step called non-maximum suppression (NMS) is applied.
   * NMS eliminates redundant or overlapping bounding box predictions by selecting the most confident bounding box for each detected object and suppressing others with significant overlap.
7. Output:
   * The final output of R-CNN includes the bounding box coordinates, class labels, and confidence scores for the detected objects in the input image.
   * The bounding box coordinates represent the precise location and size of the detected objects, while the class labels indicate the predicted object categories.
   * The confidence scores reflect the level of confidence in the detection results.
     + issues with R-CNN

R-CNN (Region-based Convolutional Neural Network) introduced significant advancements in object detection, but it also has some limitations and issues. Here are a few notable challenges associated with R-CNN:

1. Computationally Expensive:
   * R-CNN requires the extraction of features for each region proposal individually, leading to a computationally expensive process.
   * Processing a large number of region proposals in an image can be time- consuming and resource-intensive, making it less suitable for real-time applications.
2. Slow Inference Speed:
   * Due to the individual processing of region proposals, R-CNN has slower inference speed compared to other object detection methods.
   * The need to run the network multiple times for each region proposal limits its real-time applicability.
3. Training Pipeline Complexity:
   * R-CNN involves a multi-stage training pipeline that includes pre-training a CNN on a large-scale image classification dataset and fine-tuning it for object detection using region proposals.
   * This training pipeline is complex and time-consuming, requiring significant computational resources and extensive labeled data.
4. Lack of Spatial Consistency:
   * R-CNN treats each region proposal independently during training and inference, neglecting the spatial consistency between neighboring regions.
   * This may result in redundant or overlapping detections and lack of precise object localization.

Overall, while R-CNN was a significant milestone in object detection research, its limitations led to the development of more efficient and accurate approaches that build upon the R-CNN framework.

# Fast R-CNN

Fast R-CNN improves upon the original R-CNN by sharing the computation of convolutional features across multiple region proposals, making it computationally efficient. It eliminates the need for separate CNN evaluations for each region proposal, resulting in faster inference speed compared to R-CNN. Additionally, the ROI pooling layer enables spatial alignment and improves the accuracy of object localization.

Fast R-CNN served as a significant advancement in object detection and laid the groundwork for subsequent developments like Faster R-CNN and Mask R-CNN, which further improved speed and accuracy by integrating region proposal generation within the network architecture.

* + - * + Working process of Faster R-CNN

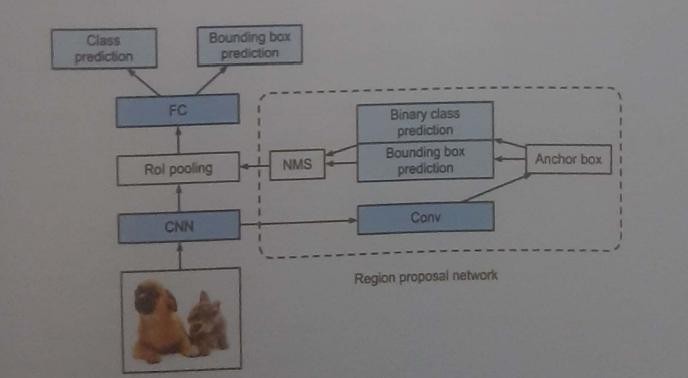


Figure 4.5-Faster R-CNN working process

Faster R-CNN (Region-based Convolutional Neural Network) is an extension of the R- CNN and Fast R-CNN frameworks. It introduces a region proposal network (RPN) that enables end-to-end training and significantly improves the speed and accuracy of object detection. Here's an overview of the working process of Faster R-CNN:

1. Input Image:
   * Faster R-CNN takes an input image as its initial input.
2. Convolutional Backbone Network:
   * The input image is passed through a convolutional neural network (CNN), such as VGG16 or ResNet, to extract high-level features.
   * The CNN processes the image and produces a convolutional feature map.
3. Region Proposal Network (RPN):
   * Faster R-CNN introduces an RPN, which is a small network that operates on the convolutional feature map.
   * The RPN generates a set of potential object proposals, known as region of interest (ROI) candidates, by sliding a small window (called an anchor) over the feature map at multiple scales and aspect ratios.
   * The RPN scores each anchor based on the likelihood of containing an object and refines their coordinates to align with the actual objects.
4. ROI Pooling and Feature Extraction:
   * The ROI candidates generated by the RPN are pooled from the convolutional feature map using the ROI pooling technique.
   * ROI pooling ensures that the proposed regions have a fixed size, regardless of their initial size or aspect ratio.
   * The pooled regions are then passed through additional layers of the CNN to extract features specific to each ROI.
5. Object Classification and Bounding Box Regression:
   * The features extracted from each ROI are fed into separate branches of the network for object classification and bounding box regression.
   * The object classification branch predicts the class probabilities for each ROI, determining the object category it belongs to.
   * The bounding box regression branch refines the coordinates of the bounding boxes to better localize the objects within the ROIs.
6. Non-Maximum Suppression (NMS):
   * Post-processing steps, such as non-maximum suppression (NMS), are applied to eliminate redundant or overlapping bounding box predictions.
   * NMS selects the most confident bounding box predictions while suppressing others that have significant overlap, ensuring that only the most accurate detections are retained.
7. Output:
   * The final output of Faster R-CNN includes the bounding box coordinates, class labels, and confidence scores for the detected objects in the input image.

Faster R-CNN integrates the region proposal network (RPN) into the object detection pipeline, enabling end-to-end training and eliminating the need for external region proposal methods used in R-CNN and Fast R-CNN. This integration improves both the speed and accuracy of object detection by sharing convolutional features and reducing the computation required for region proposal generation. Faster R-CNN has become one of the most widely adopted frameworks for object detection tasks.

## SSD (Single Shot MultiBox Detector)

SSD offers several advantages over previous object detection methods. By using multi- scale feature maps and anchor boxes, SSD can efficiently detect objects at different scales without the need for region proposal networks (RPNs) or multiple stages of processing.

This makes SSD faster and more suitable for real-time applications. Additionally, SSD achieves good accuracy by leveraging the information from different layers of the CNN backbone.

* + - * + Overview of architecture

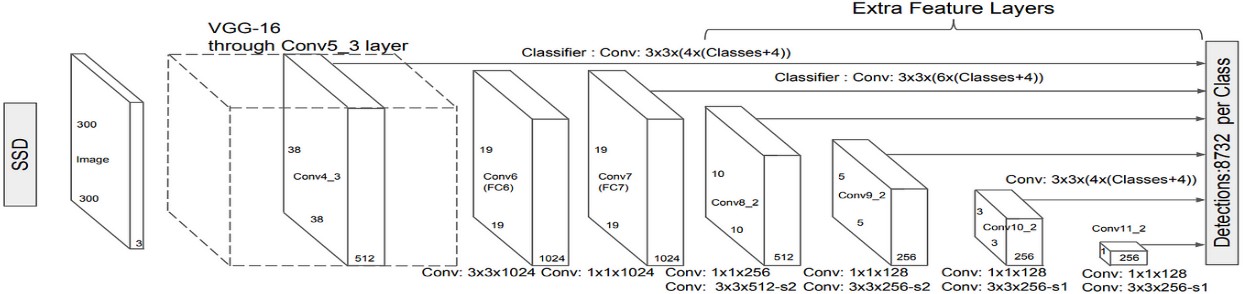


Figure 4.6-Architecture of SSD

1. Input:
   * The SSD architecture takes an input image as its initial input.
2. Convolutional Backbone Network:
   * The input image is passed through a convolutional neural network (CNN) backbone, such as VGG16 or ResNet.
   * The CNN backbone extracts a series of feature maps at different spatial resolutions, capturing information at different levels of abstraction.
3. Multi-scale Feature Maps:
   * SSD utilizes feature maps from different layers of the CNN backbone with varying spatial resolutions.
   * These feature maps capture information at different scales, allowing the detection of objects at various sizes.
4. Anchor Boxes:
   * For each spatial location in the feature maps, SSD associates a set of anchor boxes of different sizes and aspect ratios.
   * Anchor boxes act as reference bounding boxes that are centered at each spatial location and serve as potential object candidates.
5. Prediction Layers:
   * SSD applies a series of convolutional layers, called prediction layers, on top of each feature map to predict the class probabilities and offsets (regression values) for each anchor box.
   * The class probabilities indicate the likelihood of each anchor box containing a particular object class.
   * The regression values adjust the coordinates and dimensions of the anchor boxes to match the ground truth bounding boxes of the objects.
6. Multi-scale Predictions:
   * Since SSD uses feature maps from multiple resolutions, it generates predictions at different scales.
   * The predictions from each scale are combined to produce a final set of object detections.
7. Non-Maximum Suppression (NMS):
   * Post-processing steps, such as non-maximum suppression (NMS), are applied to filter and refine the predicted bounding boxes.
   * NMS selects the most confident bounding box predictions while suppressing others with significant overlap, ensuring accurate and non- redundant detections.
8. Output:
   * The final output of SSD includes the bounding box coordinates, class labels, and confidence scores for the detected objects in the input image.

The architecture of SSD is designed to achieve real-time object detection with high accuracy. By using multi-scale feature maps and anchor boxes, SSD can efficiently detect objects of different sizes and aspect ratios without the need for region proposal networks (RPNs) or multiple stages of processing. This makes SSD faster and more suitable for real-time applications, while maintaining good accuracy.

* Limitations of SSD

While SSD (Single Shot MultiBox Detector) is a powerful and widely used object detection algorithm, it does have some limitations. Here are a few of them:

1. Difficulty in detecting small objects: SSD may struggle to detect small objects in an image. Since it uses anchor boxes of fixed sizes, it can miss small objects that are significantly smaller than the anchor box sizes. The feature maps at higher resolutions tend to capture larger objects better, while smaller objects may not have enough spatial information to be accurately detected.
2. Limited aspect ratio coverage: SSD relies on predefined anchor boxes with fixed aspect ratios. This design choice limits its ability to accurately detect objects with extreme aspect ratios or irregular shapes. It may not capture the full extent of objects that deviate significantly from the predefined anchor box shapes.
3. Localization accuracy: While SSD achieves good overall object detection accuracy, the localization accuracy of detected objects may not be as precise. The bounding box regression process may introduce some errors in accurately localizing the objects, leading to slight misalignments between the predicted bounding boxes and the ground truth boxes.

* Points to consider

When considering the use of object detection algorithms like SSD in a project, here are some important points to consider:

1. Accuracy: Evaluate the accuracy of the object detection algorithm on your specific task and dataset. Assess its ability to detect objects of interest with high precision and recall. Consider the trade-off between accuracy and computational resources required.
2. Speed and Efficiency: Assess the speed and efficiency of the algorithm, especially if real-time performance is a requirement. Look for algorithms that can achieve fast inference times while maintaining acceptable accuracy levels.
3. Model Size and Resource Requirements: Consider the size of the trained model and the computational resources needed for inference. Ensure that the model can be deployed and run efficiently on the target hardware or platform.

## 4.1.2.4 YOLO(You Only Look Once)

YOLO (You Only Look Once) is a popular real-time object detection algorithm that has gained significant attention in the computer vision community. Unlike traditional object detection algorithms that rely on region proposal methods, YOLO takes a different approach by formulating object detection as a single regression problem.

* Working process of YOLO

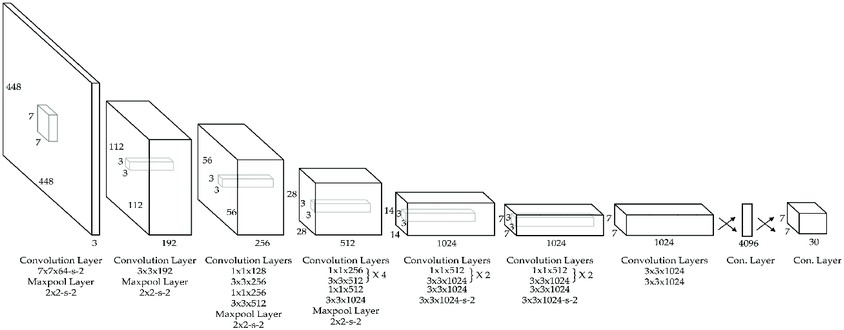


Figure 4.7- YOLO Working process

1. Input Image:
   * YOLO takes an input image of any size as its initial input.
2. Preprocessing:
   * The input image is preprocessed by resizing it to a fixed size that the YOLO model expects. It is then normalized to have pixel values between 0 and 1.
3. CNN Backbone:
   * The preprocessed image is passed through a convolutional neural network (CNN) backbone, such as Darknet, to extract features.
   * The CNN backbone applies a series of convolutional and pooling layers to capture hierarchical features from the input image.
4. Feature Extraction:
   * The CNN backbone generates a feature map that represents the input image at a lower spatial resolution.
   * This feature map retains high-level semantic information while reducing the spatial dimensions.
5. Grid Creation:
   * The feature map is divided into a grid of cells, typically with a size of, for example, 13x13 or 19x19.
   * Each cell is responsible for detecting objects within its spatial region.
6. Anchor Boxes:
   * YOLO uses anchor boxes, which are pre-defined bounding boxes of different sizes and aspect ratios.
   * Each grid cell predicts a fixed number of anchor boxes that are centered on the cell.
7. Predictions:
   * For each grid cell, YOLO predicts multiple bounding boxes and their corresponding class probabilities.
   * Each bounding box prediction consists of its coordinates (x, y, width, height) relative to the cell's spatial location.
   * The class probabilities indicate the likelihood of each bounding box containing a particular object class.
8. Object Confidence:
   * YOLO computes an object confidence score for each predicted bounding box.
   * The object confidence score represents the confidence level that a bounding box contains an object.
   * It is a combination of the class probability and the localization accuracy of the bounding box.
9. Non-Maximum Suppression (NMS):
   * Post-processing step called non-maximum suppression (NMS) is applied to filter and refine the predicted bounding boxes.
   * NMS removes duplicate or highly overlapping bounding boxes, keeping only the most confident ones.
10. Output:

o The final output of YOLO includes the bounding box coordinates, associated class labels, and confidence scores for the detected objects in the input image.

The main idea behind YOLO is to perform object detection in a single pass by directly predicting bounding boxes and class probabilities from the grid cells. This approach enables YOLO to achieve real-time object detection performance while maintaining a reasonable level of accuracy. By leveraging a CNN backbone and anchor boxes, YOLO can handle objects of various sizes and aspect ratios efficiently.

* Points to consider

1. Accuracy: Evaluate the accuracy of YOLO on your specific object detection task and dataset. Consider its performance in terms of both object localization and classification accuracy. Assess whether the level of accuracy achieved by YOLO is sufficient for your application.
2. Speed and Efficiency: YOLO is known for its real-time performance and efficiency. However, the speed and efficiency can vary depending on the version of YOLO and the hardware platform used. Evaluate the inference speed of YOLO on your target hardware and assess if it meets your real-time processing requirements.
3. Model Size and Resource Requirements: Take into account the size of the YOLO model and the computational resources required for training and inference. Consider the memory and storage constraints of your deployment environment and ensure that the YOLO model can be efficiently deployed on your target hardware or platform.

## the model we choose to our task:

It is very hard to have a fair comparison among different object detectors, There is no straight answer on which model is the best.

For real-life applications, we make choices to balance accuracy and speed.

Besides, we need to aware of other choices that impact the performance since mobile applications has limited computation resources and strict power consumption constraints.

## speed

Input image resolutions and feature extractors impact speed. Below is the highest and lowest FPS.

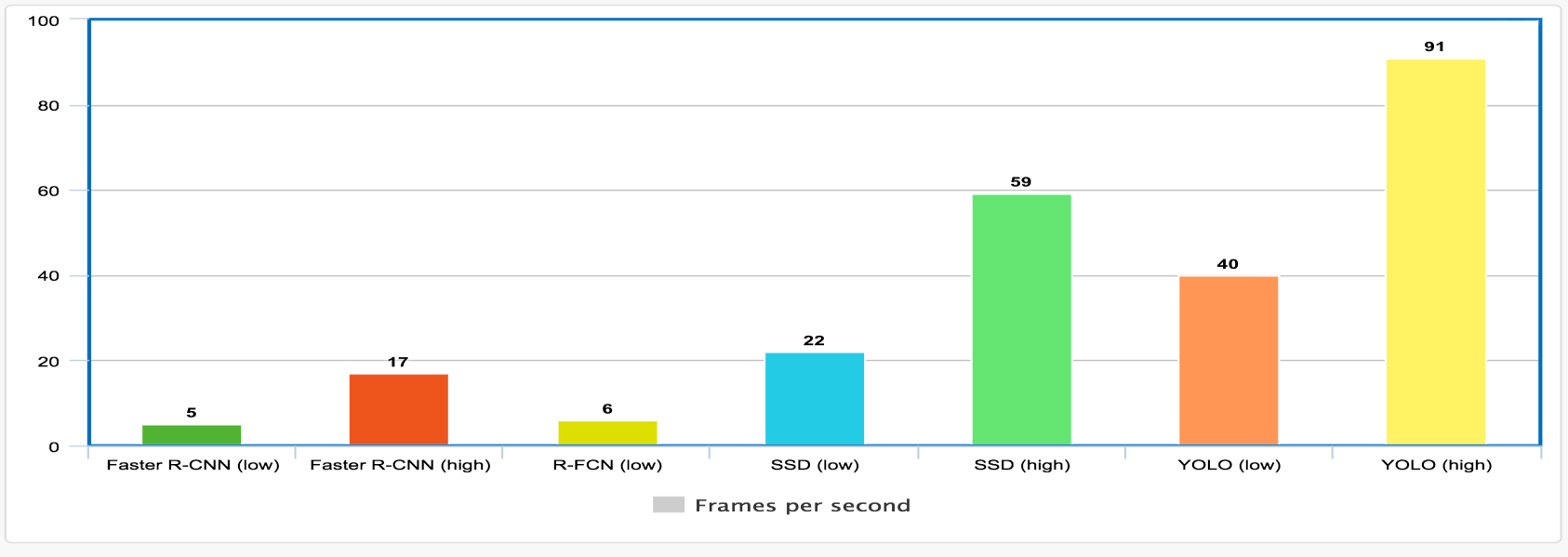


Figure 4.8-Graph of Speed

## accuracy

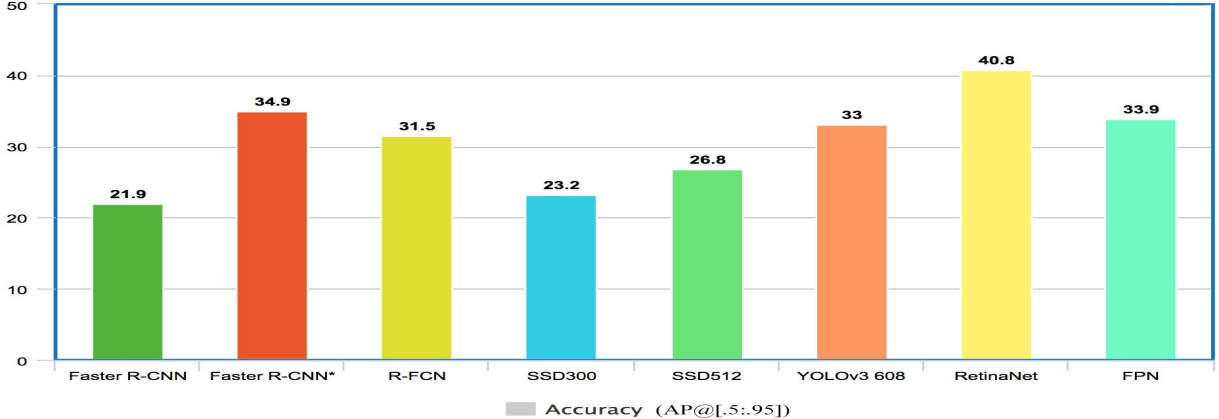


Figure 4.9-Graph of Accuracy

The above items are studying how the resolution of the input images and feature extractors affects accuracy and speed detector.

Overall. Faster R-R-FCN CNN and compared with SSD and YOLO models are slightly slower but more accurate.

YOLO SSDs and methods have difficulty detecting small objects but are faster.

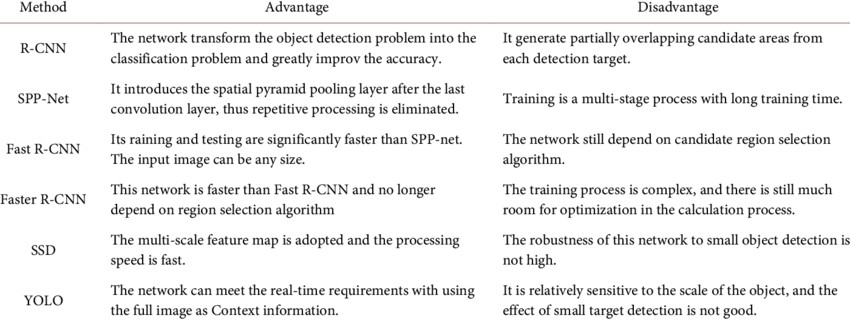


Table 4-Advantages vs Disadvantages of One and Two Stages Detector

**So based of these results we decided to use the 2-stage algorithms for our task.**

## 4.3.4. Open CV

OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being an Apache 2 licensed product, OpenCV makes it easy for businesses to utilize and modify the code.

The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms.

These algorithms can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution image of an entire scene, find similar images from an image database, remove red eyes from images taken using flash, follow eye movements, recognize scenery and establish markers to overlay it with augmented reality, etc. OpenCV has more than 47 thousand people of user community and estimated number of downloads exceeding 18 million. The library is used extensively in companies, research groups and by governmental bodies.

Along with well-established companies like Google, Yahoo, Microsoft, Intel, IBM, Sony, Honda, Toyota that employ the library, there are many startups such as Applied Minds, VideoSurf, and Zeit era, that make extensive use of OpenCV. OpenCV’s deployed uses span the range from stitching street view images together, detecting intrusions in surveillance video in Israel, monitoring mine equipment in China, helping robots navigate and pick up objects at Willow Garage, detection of swimming pool drowning accidents in Europe, running interactive art in Spain and New York, checking runways for debris in Turkey, inspecting labels on products in factories around the world on to rapid face detection in Japan.

It has C++, Python, Java and MATLAB interfaces and supports Windows, Linux, Android and Mac OS. OpenCV leans mostly towards real-time vision applications and takes advantage of MMX and SSE instructions when available. Full-featured CUDA and OpenCL interfaces are being actively developed right now. There are over 500 algorithms and about 10 times as many functions that compose or support those algorithms. OpenCV

is written natively in C++ and has a templated interface that works seamlessly with STL containers.

## Applications of Open cv:

It has a wide range of applications in areas such as image and video processing, object detection and recognition, face detection and recognition, robotic vision, and augmented reality, among others. Here are some examples of how OpenCV can be applied:

-Object detection / face recognition / facial recognition system:

A facial recognition system is a technology that can match a human face form a digital image or a video frame against a database of faces.

-object detection:

Object detection is a computer technology related to computer vision and image processing that focuses on identifying instances of semantic objects of a certain class (such as human , building ,cars) in digital images and videos .

-image segmentation and recognition:

In digital images processing and computer vision, image segmentation is the process of partitioning a digital image into multiple segments (sets of pixel ,also known as image objects). The goal of segments is to simplify and/or change the representation of an image into something that is more meaningful and easier to analyze.

## Advantage of Open cv:

* Ease of use : OpenCV is easy and simple to learn.
* Availability of many tutorials: The fact that there are lots of tutorials available is a big plus as one can access many learning resources.
* Compatibility with leading coding languages: OpenCV works with almost all the leading programming languages today, including python , C++ and java.
* Free to use : Undoubtedly , a big plus is the fact that it is open source and hence free to use.

# OCR

OCR (Optical Character Recognition) is a process of converting printed or handwritten text into digital form that can be processed by a computer. OCR technology allows computers to read and understand text in images, such as scanned documents or photographs of text.

It works by analyzing the shapes and patterns of characters in an image and then converting them into a machine-readable format.

OCR is used in a wide range of applications, including digitizing printed documents, extracting text from images and videos, and creating searchable archives of text documents. OCR technology has significantly improved in recent years, thanks to advances in computer vision and machine learning. OCR algorithms can now recognize a wide range of fonts, text sizes, and styles, and can even recognize handwriting with high accuracy.

OCR technology has many benefits, including reducing the time and cost of data entry and making it easier to search and analyze text data. OCR is widely used in industries such as finance, healthcare, and government, where there is a large volume of paper-based documents that need to be digitized and processed efficiently.

## Optical Character Recognition algorithm

Optical character recognition algorithms can be based on traditional image processing and machine learning-based approaches or deep learning–based methods.

## Traditional OCR

While traditional machine learning-based approaches are fast to develop, they take significantly more time to run and are easily outstripped by deep learning algorithms both in accuracy and inference speed.

Traditional approaches to OCR go through a series of pre-processing steps where the inspected document is cleaned and made noise-free. Following this , the document is binarized for subsequent contour detection to aid in the detection of lines and columns.

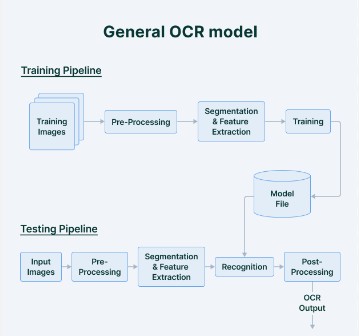


Figure 4.10-General OCR model

Finally, Characters building the lines are extracted, segmented, and identified via various machine learning algorithms like K-nearest neighbors and support vector machines.

While this work great on simple OCR datasets like easily distinguishable printed data and handwritten MNIST data, they miss out on many features, marking them fail when working on complex datasets.

## Optical Character Recognition mechanism Steps

OCR (Optical Character Recognition) works by analyzing the shapes and patterns of characters in an image and then converting them into a machine-readable format.

### OCR Mechanism Steps

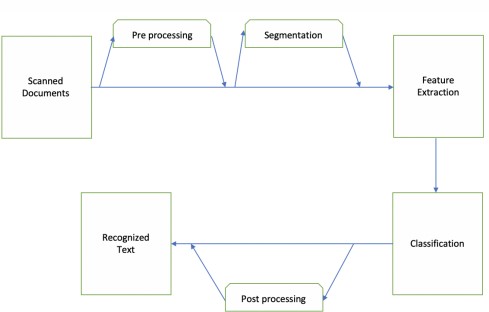


Figure 4.11- OCR Mechanism Steps

## the Smart glasses Activities

* + 1. Main Activity: The "main" activity is the activity that loads first and the rest of the application.

Every system can have multiple activities; therefore, you can list other activities to load and use later on but you can only have one "main" activity in Main activity we used more than one library to make the required tasks

* + - * The RPi.GPIO library: is a Python library that provides an interface for controlling the GPIO pins on a Raspberry Pi. and it refers to the pins on the Raspberry Pi that can be used for various purposes, such as reading inputs from sensors or controlling outputs to devices.
      * gtts: is a Python library that stands for "Google Text-to-Speech." It provides a simple interface to convert text into speech using Google's Text-to-Speech API. With gtts, you can generate audio files from text strings and save them as MP3 files for playback.
      * Pytesseract: is a Python library that provides an interface to the Tesseract OCR engine. which is a technology used to recognize and extract text from images.
    1. Help activity: The function of this activity is to guide the user for how to use the system and navigate through it, by voice. and user can use it from anywhere by tapping the button's.
    2. Object Detection Activity: the function of object detection activity is to automate the process of detecting, localizing, and classifying objects within visual data. It has various applications ranging from surveillance and security to autonomous systems, augmented reality, and content analysis.

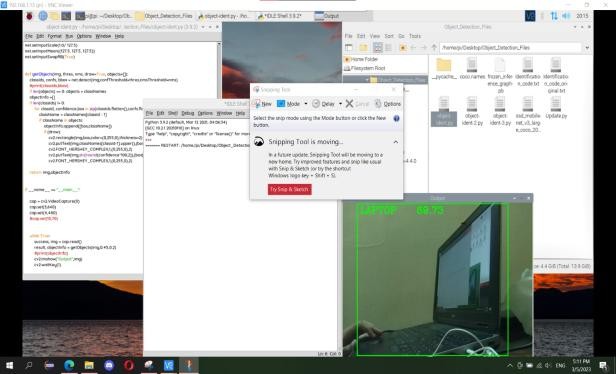


Figure 4.12- Object Detection camera view

* + 1. Face Recognition Activity: the function of face recognition activity is to automatically detect, recognize, and analyze faces within visual data. It has a wide range of applications, including security systems, surveillance, biometric authentication, personalization, and social analysis.

Features and Applications

Face Detection:

Automatic Identification: Utilizes image processing techniques to identify human faces within the camera's field of view.

Real-time Feedback: Provides immediate auditory feedback to the user about the presence and location of detected faces.

Face Recognition:

Biometric Authentication: Verifies the identity of individuals based on stored biometric data.

Personalization: Recognizes familiar faces (family, friends, colleagues) and informs the user through voice communication.

Access Control: Enhances security by allowing access to certain areas or information only to recognized individuals.

Face Analysis:

Emotion Detection: Analyzes facial expressions to determine the emotional state of individuals, providing the user with context about the social interactions.

Social Analysis: Gathers data on the number and identity of people in a certain area, useful for social and security applications.

Implementation

Hardware Components:

High-Resolution Camera: Captures detailed images and videos for accurate face detection and recognition.

Microcontroller/Processor: Executes face recognition algorithms and processes visual data in real-time.

Speakers and Microphone: Facilitates voice communication to provide feedback and receive commands from the user.

Software Components:

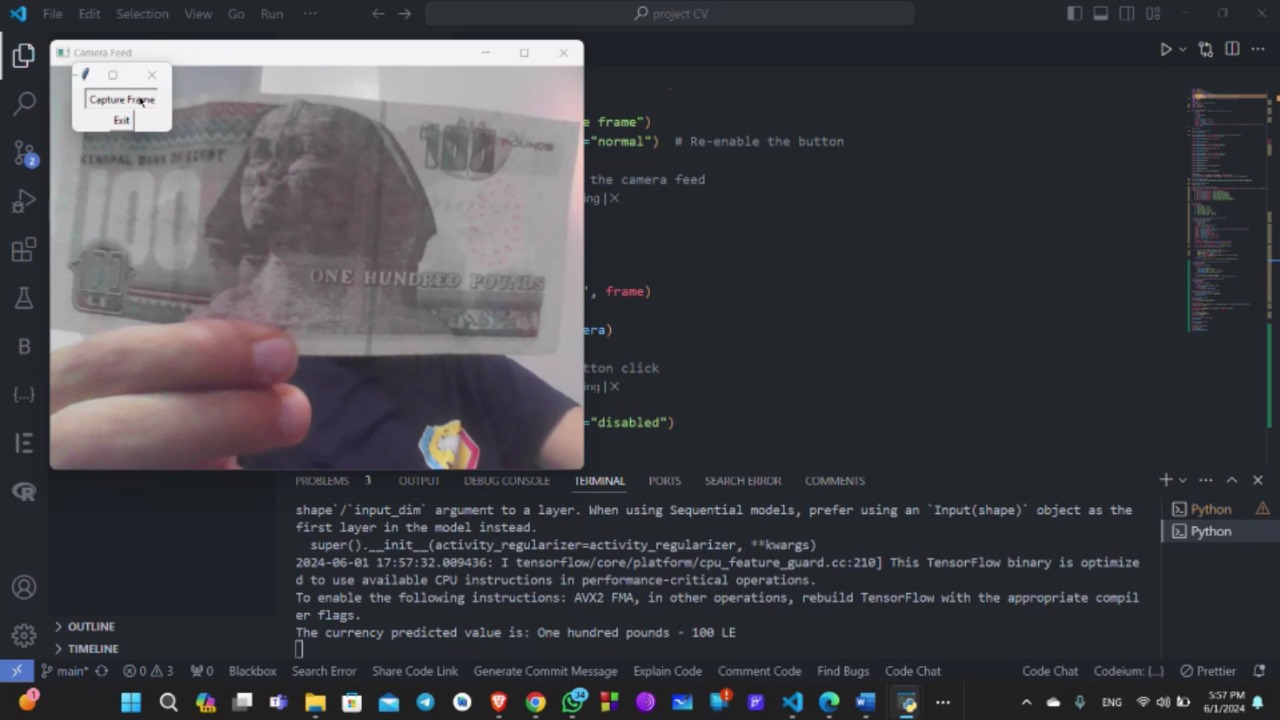
Face Detection Algorithm: Utilizes machine learning techniques (e.g., Haar cascades, HOG + SVM) to detect faces in visual data.

Face Recognition Algorithm: Employs advanced neural networks (e.g., CNNs) and frameworks (e.g., OpenCV, Dlib, FaceNet) to recognize and match faces.

Emotion Detection Algorithm: Analyzes facial landmarks and expressions using pre-trained models to identify emotions.



Figure 4.13- Face Recognition camera view

* + 1. Coin Detection Activity: Our project focuses on developing an advanced Coin Detection System using transfer learning and Convolutional Neural Networks (CNNs). The primary goal is to automate the detection, classification, and analysis of Egyptian coins from visual data. This system has wide-ranging applications in various domains such as vending machines, coin-operated systems, banking, retail, numismatics, and coin collection management.

1. Pre-trained Model Utilization

Instead of collecting a new dataset, we leveraged a pre-trained model to jumpstart our project. Utilizing a pre-trained model allows us to benefit from the extensive training it has already undergone on a large, diverse dataset. This significantly improves performance and reduces the time and resources needed for training.

2. Preprocessing

Preprocessing is a critical step to prepare the collected images for model training. This includes:

- Image Resizing : Standardizing the size of all images to match the input size required by the model.

- Normalization : Adjusting the pixel values to a standard scale.

- Augmentation : Applying transformations such as rotation, scaling, and flipping to increase the diversity of the training data and prevent overfitting.

3. Model Architecture

For the core of our coin detection system, we utilized transfer learning with Convolutional Neural Networks (CNNs). Transfer learning allows us to leverage a pre-trained model on a large dataset and fine-tune it for our specific task, significantly improving performance and reducing training time.

- Base Model : We used a pre-trained CNN model (e.g., VGG16, ResNet50) as the base for our system. These models have already learned rich feature representations from millions of images.

- Custom Layers : On top of the base model, we added custom layers specific to our coin classification task. These included additional convolutional layers, pooling layers, and dense layers to refine the features and perform classification.

4. Training

The training process involved fine-tuning the pre-trained model with our dataset. This process included:

- Feature Extraction : Using the convolutional base of the pre-trained model to extract features from the coin images.

- Fine-Tuning : Training additional layers on top of the pre-trained model while keeping the earlier layers frozen, then gradually unfreezing and training more layers to improve performance.

- Optimization : Using an optimizer such as SGD with momentum to minimize the loss function and improve accuracy.

5. Deployment

The trained model was integrated into a user-friendly application that can be used in various practical scenarios. The deployment phase includes:

- API Development : Creating endpoints to interact with the model.

- User Interface : Designing a simple and intuitive interface for users to upload images and receive results.

- Real-time Processing : Ensuring the system can process images and deliver results quickly.

Tools and Technologies

- Python : The primary programming language used for developing the model and application.

- TensorFlow/Keras : Deep learning frameworks used to build and train the CNN model.

- OpenCV : For image processing tasks such as preprocessing and augmentation.

- Flask : For creating the web application and API.

- NumPy and Pandas : For data manipulation and analysis.

Our Coin Detection System leverages the power of transfer learning and CNN algorithms to provide an automated solution for detecting and classifying Egyptian coins. By using a pre-trained model and fine-tuning it for our specific task, we achieved high accuracy and efficiency. This project demonstrates the effectiveness of modern machine learning techniques in solving practical problems, opening up new possibilities in various fields.

* + 1. OCR Activity: the function of this activity is scanning an image containing text and extracting the text from it then converting textصورة تحتوي على نص, لقطة شاشة, رقم, الخط

       تم إنشاء الوصف تلقائياً to voice.

Figure 4.14- OCR scanning

# CH5: SSVIP Hardware

## Introduction

In our project the hardware part is important like software because all components will be assembled in the form of a device and will be worn on the body like a T-shirt so that it is light and easy to handle and orders will be received via software.

* 1. **Components** 1- Microcontroller's 2- Sensor's

1. Alarm's
2. GPS & GSM Modules 5- Camera

6- Battery

## Smart Wearable System microcontroller

A microcontroller is a small and low-cost microcomputer. which is designed to perform the specific tasks of embedded systems like displaying microwave's information, receiving remote signals,etc.

the general microcontroller consists of the processor, the memory (RAM, ROM,EPROM) Serial ports , peripherals (timers, counters),etc.

Difference between Microprocessor and Microcontroller

The following image highlights the difference between microprocessor and microcontroller

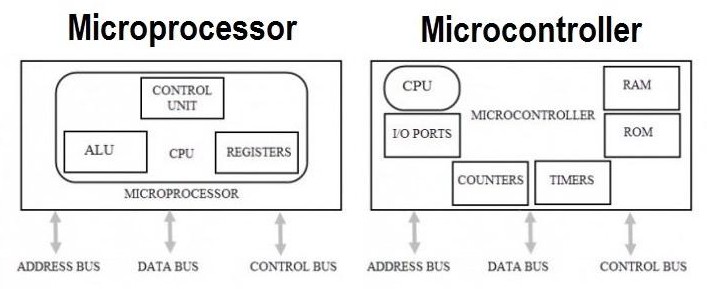
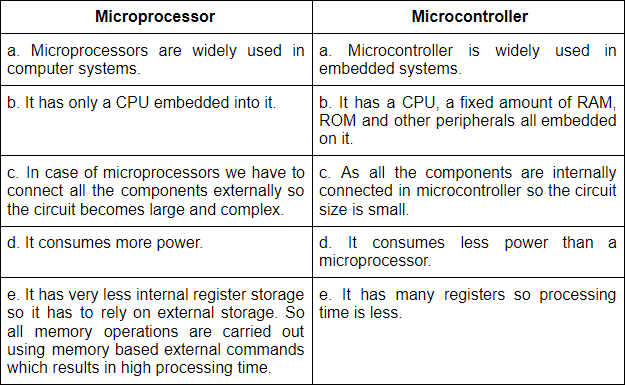


Figure 5.1-Difference between Microcontroller & microprocessor

The following table highlights the difference between microprocessor and microcontroller



**Table 5-Microcontroller vs Microprocessor**

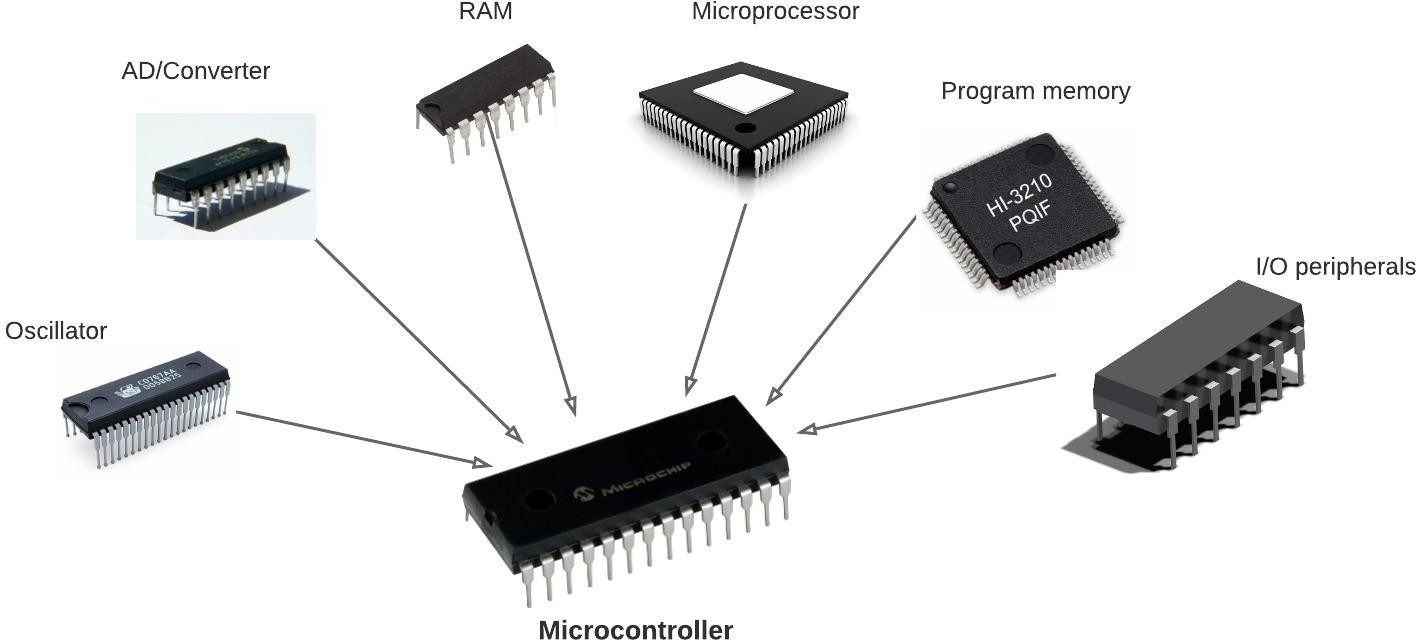
**In the part of the Smart Wearable System, we will use the microcontroller, but in the part of the Smart glasses, we will need the microprocessor, Because this will be more suitable for each part.**

## -Smart Wearable System

there are many microcontrollers, the most common are:

PIC, AVR, and ARM are different families of microcontrollers that are commonly used in embedded systems. Arduino, on the other hand, is a brand of microcontroller boards that use either AVR or ARM microcontrollers.PIC microcontrollers are manufactured by

Microchip Technology and are known for their low power consumption and ease of use. AVR microcontrollers are made by Atmel and are known for their high performance and low cost. ARM microcontrollers are made by a variety of manufacturers and are known for their high performance and scalability.Arduino boards are designed to simplify the process of programming microcontrollers, and they come with a software development environment that is easy to use. Arduino boards are available in a variety of form factors and use either AVR or ARM microcontrollers.In summary, PIC, AVR, and ARM are different families of microcontrollers that are commonly used in embedded systems, while Arduino is a brand of microcontroller boards that use either AVR or ARM microcontrollers and are designed to simplify the process of programming microcontrollers.



## AVR

Figure 5.2-Component of Microcontrollers

The AVR microcontroller is a type of microcontroller developed by Atmel Corporation. It is a family of 8-bit RISC (Reduced Instruction Set Computing) based microcontrollers, which are widely used in various embedded systems applications.

AVR microcontrollers are known for their low power consumption, high processing power, and ease of use. They come with a range of features such as flash memory for program storage, EEPROM for data storage, timers, interrupts, analog-to-digital converters, and more.

AVR microcontrollers are programmed using a variety of programming languages, including C, C++, and Assembly language. They are also supported by several Integrated Development Environments (IDEs) such as Atmel Studio, CodeVisionAVR, and Arduino IDE.

AVR microcontrollers are widely used in various applications such as robotics, home automation, industrial automation, medical equipment, and automotive control systems. They are also used in hobbyist projects, thanks to their ease of use and low cost.

## common Avr types

There are several types of AVR microcontrollers available, each with different features and capabilities. Here are some of the most common AVR types:

ATmega: This is a popular family of AVR microcontrollers and is widely used in various applications. It comes with a range of features such as a high number of input/output pins, timers, and serial communication interfaces.

ATtiny: This is a low-power and low-cost AVR microcontroller, and it is ideal for small applications. It has a limited number of input/output pins and lower memory compared to the ATmega.

ATxmega: This is a high-performance AVR microcontroller, and it comes with

advanced features such as DMA (Direct Memory Access) and a high-speed ADC (Analog to Digital Converter).

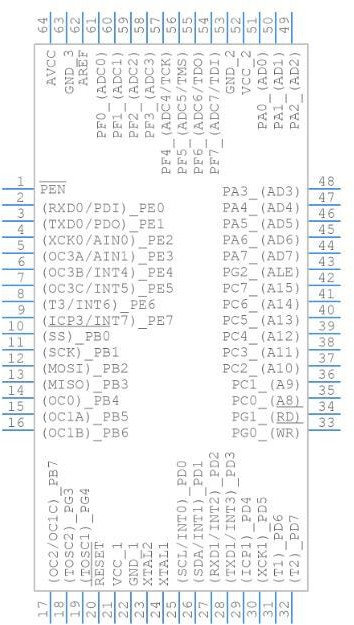
AT90: This is an older family of AVR microcontrollers, but it is still widely used in some applications. It has a limited amount of memory and fewer features than the newer AVR microcontrollers.

ATtiny10: This is the smallest AVR microcontroller, and it comes with very limited features. It has only 1KB of flash memory and 32 bytes of RAM, but it is ideal for small and simple applications.

These are just a few of the most common AVR microcontrollers. There are many other types available, each with different features and capabilities, and choosing the right one depends on the specific requirements of the application.

**ATmega128 more suitable according to (cost, size, and pins we need) for the part of 'Smart Wearable System'.**

## ATmega128 Board



**- ATmega128 pins**

Figure 5.3-Board ATmega128

The ATmega128 microcontroller is a member of the AVR microcontroller family from Atmel. It has a total of 64 pins, including 32 I/O pins. The pins are divided into several groups based on their functionality. Here is a brief overview of the different groups of pins on the ATmega128 microcontroller:

Power supply pins: These pins are used to provide power to the microcontroller. They include Vcc (pin 10) and GND (pins 11 and 12).

Crystal oscillator pins: These pins are used to connect an external crystal oscillator to the microcontroller. They include XTAL1 (pin 13) and XTAL2 (pin 14).

Reset pins: The RESET pin (pin 9) is used to reset the microcontroller.

JTAG pins: The JTAG pins (pins 32-35) are used for in-circuit debugging and programming.

Analog input pins: There are eight analog input pins (ADC0-ADC7) on the ATmega128 microcontroller. These pins can be used to measure analog signals using the built-in ADC.

Digital input/output pins: There are 32 digital input/output pins (PD0-PD7, PE0-PE7, PF0-PF7, and PG0-PG3) on the ATmega128 microcontroller. These pins can be configured as either input or output pins and can be used for a wide range of applications.

USART pins: The ATmega128 microcontroller has two USARTs (Universal Synchronous/Asynchronous Receiver/Transmitter) that can be used for serial communication. Each USART has its own set of pins that include RXD (receive data), TXD (transmit data), and XCK (clock).

SPI pins: The ATmega128 microcontroller has one SPI (Serial Peripheral Interface) that can be used for synchronous serial communication. The SPI has its own set of pins that include MOSI (Master Out Slave In), MISO (Master In Slave Out), SCK (Serial Clock), and SS (Slave Select).

I2C pins: The ATmega128 microcontroller has one I2C (Inter-Integrated Circuit) that can be used for serial communication with other devices. The I2C has its own set of pins that include SDA (Serial Data) and SCL (Serial Clock).

## Atmega128 Programming:

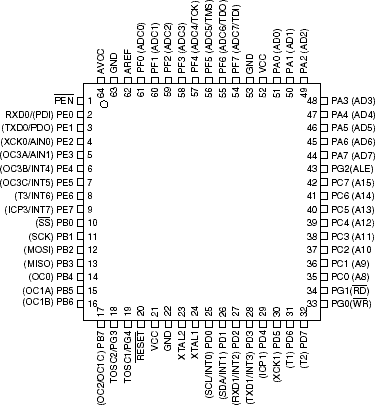


Figure 5.4-Atmega128Pins

To program the Atmega128 microcontroller, we need an integrated development environment (IDE) AVR-GCC, along with a compatible programmer USBasp.

Here are the general steps we do to get started with Atmega128 programming:

**Set up development environment:** Download and install the AVR-GCC. Connect USBasp to the computer.

**Create a new project:** Launch the IDE and create a new project for the Atmega128 microcontroller.

Specify the project settings such as target microcontroller, clock frequency, and programming interface.

**Write code:** In the IDE, create a new source file . Write code using C programming language. Implement the desired functionality, including initialization, I/O operations, and other necessary operations.

**Build the project:** Compile code within the IDE to generate the object files and check for any compilation errors.

Resolve any errors or warnings that may arise during the compilation process.

**Program the Atmega128 microcontroller:** Connect programmer to the appropriate pins of the Atmega128 microcontroller.

Configure the programming settings in the IDE, specifying the programmer and target microcontroller.

Initiate the programming process, which will flash the compiled code onto the microcontroller's memory.

**Test and debug:** Verify that the program is functioning as expected by running it on the Atmega128 microcontroller.

Use debugging tools provided by the IDE to identify and fix any issues or unexpected behavior.

## Sensors

A sensor is a device that detects or measures a physical quantity or environmental condition and converts it into an electrical or digital signal. Sensors are used in a wide range of applications, from simple everyday devices like thermostats, to complex industrial and scientific systems.

Sensors can detect various types of physical quantities, such as temperature, pressure, humidity, light, sound, motion, position, and many others. Some sensors can also detect chemical or biological substances**.**

Overall, sensors play a crucial role in many aspects of our daily lives, and their importance is only increasing as technology continues to advance.

## Distance sensors

|  |  |  |  |
| --- | --- | --- | --- |
|  | Ultrasonic  sensor | IR sensors | LIDAR |
| **Measuring**  **distance** | <16m | **<** 6m | <2200m |
| **Resolution** | High | Variable according  to distance | Very high |
| **Detect angel range** | Narrow or wide  according to its model | Small | Point |
| **Output lincarity** | Linear | Non linear | Linear |
| **Size** | Small | Small | Moderate |
| **Weight** | Light | Light | Heavy |
| **Cost** | Cheap | Cheap | Expensive |

Table 6-Distance Sensors

Ultrasonic is more suitable for project because of cost, size ,and weight.

## Ultrasonic :



**Fig 5.5. Ultrasonic**

The Ultrasonic Sensor is arguably the most common distance measuring sensor,also known as the Sonar sensor .It detects the distance to objects by emitting high-frequency sound waves.

## -Ultrasonic Sensor : Work Principle

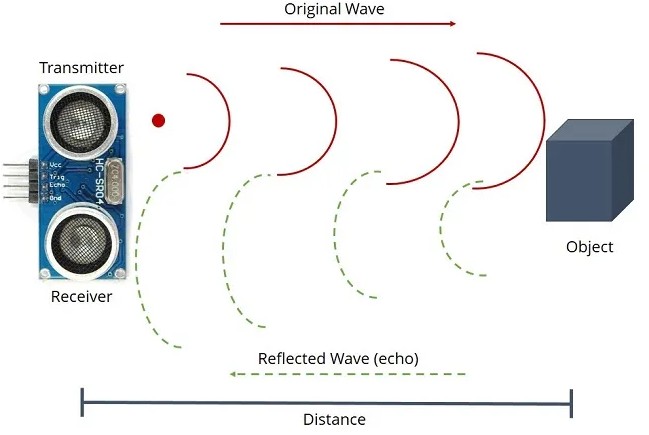


Figure 5.6-Ultrasonic sensor working principle

The Ultrasonic Sensor emit high-frequency sound towards the target object,and a timer is started.

Target object reflects the sound waves back towards the sensor. The receiver picks up the reflected wave and stop the timer.

The time taken for the wave's return is calculated against the speed of sound to determine the distance travelled.

## - Calculating Measurement

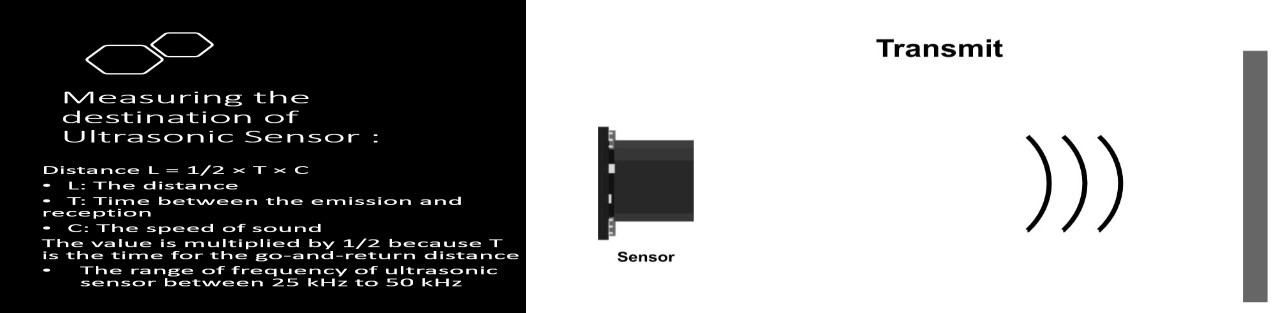
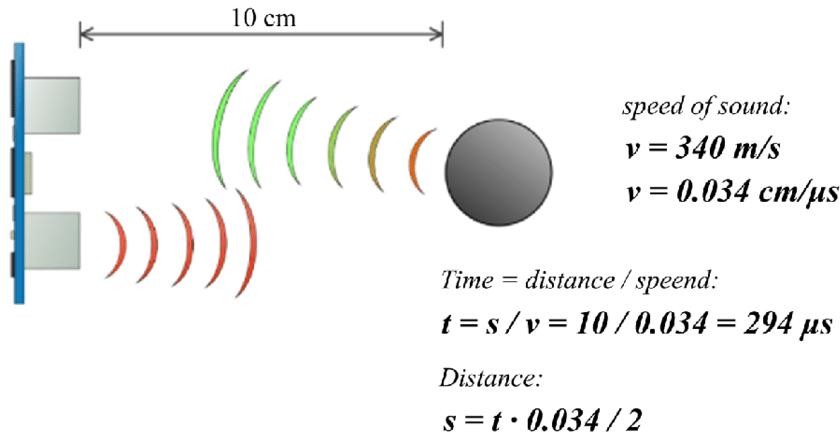


Fig 5.7-Measuring Distance of Ultrasonic sensor

Example :



## Buzzer

Fig 5.8-Measuring Distance of Ultrasonic sensor Example

There are many ways to communicate between the user and a product. One of the best ways is audio communication using a buzzer IC. So during the design process, understanding some technologies with configurations is very helpful.

Buzzer is an audio signaling device like a beeper or buzzer may be electromechanical or piezoelectric or mechanical type. The main function of this is to convert the signal from audio to sound. Generally, it is powered through DC voltage and used in timers, alarm devices, printers, alarms, computers, etc. Based on the various designs, it can generate different sounds like alarm, music, bell & siren.

The pin configuration of the buzzer is shown below. It includes two pins namely positive and negative. The positive terminal of this is represented with the ‘+’ symbol or a longer terminal. This terminal is powered through 6Volts whereas the negative terminal is represented with the‘-‘symbol or short terminal and it is connected to the GND terminal.



Figure 5.9-Buzzer

## - Buzzer : Working Principle

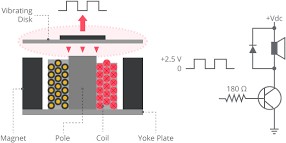


Figure 5.10-Buzzer working principle

The working principle of a buzzer depends on the theory that, once the voltage is given across a piezoelectric material, then a pressure difference is produced. A piezo type includes piezo crystals among two conductors.

Once a potential disparity is given across these crystals, then they thrust one conductor & drag the additional conductor through their internal property. So this continuous action will produce a sharp sound signal.

## - Circuit Diagram :

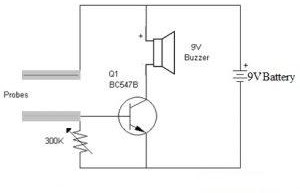


Figure 5.11-Circuit Diagram of buzzer

## Vibration motor

A vibration motor is an electric motor that is equipped with an eccentric weight attached to its output shaft. When the motor rotates, the weight creates a vibration effect, which can be felt as a tactile feedback. The vibration can be adjusted by changing the speed and direction of the motor rotation, as well as the position of the weight.

Vibration motors are commonly used in handheld devices, such as cell phones and gaming controllers, to provide haptic feedback to the user. This feedback can be used to alert the user to incoming calls or messages, provide confirmation of button presses, or simulate the sensation of different actions in a game.

Vibration motors can also be used in industrial and automotive applications to provide feedback to operators or to alert them to potential issues. For example, they can be used to signal when a machine is operating at an optimal level, or to alert an operator to a malfunction or safety hazard.

In our project, the vibration motor was used as an alarm device and A vibration motor called the coin motor was used.

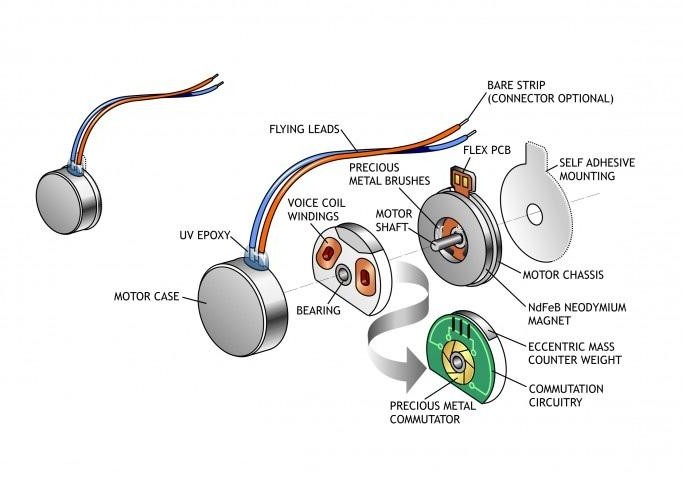


Figure 5.12-Vibration motor

1. Small size makes it easy to mount in or on your project
2. Low noise level enables feedback without unwanted distractions 3- Rotates both CW and CCW for ease of use and installation

## GPS Chip

A GPS (Global Positioning System) chip is a semiconductor device that is designed to receive signals from GPS satellites and calculate the device's location on Earth. GPS chips are used in a wide range of devices, including smartphones, tablets, laptops, and dedicated GPS receivers.

The GPS chip receives signals from multiple GPS satellites orbiting the Earth, and uses the information from these signals to determine the device's location, as well as its speed and direction of movement. The GPS chip typically contains a GPS receiver, which is responsible for receiving and processing the signals from the satellites, and a GPS antenna, which is used to capture the signals.

GPS chips are commonly used for navigation, location-based services, and tracking applications. They are also used in various industries, including aviation, transportation, and logistics, to track the movement of vehicles and assets. GPS chips have become increasingly popular in recent years, as the technology has become more affordable and accessible, and they are now found in many consumer electronics devices.

## NEO 6M GPS Module

The NEO-6M GPS module is a compact and low-power GPS receiver module that is designed to provide accurate positioning and timing information to a wide range of applications. The module is based on the u-blox 6 GPS chipset and is capable of receiving signals from GPS, GLONASS, and QZSS satellites.

The NEO-6M GPS module provides accurate positioning information with a horizontal accuracy of up to 2.5 meters and a vertical accuracy of up to 5 meters. The module also provides velocity and time information, and supports various navigation modes, including autonomous, differential, and assisted GPS.

The module can operate on a wide range of supply voltages, from 3.3V to 5V, and consumes very low power, making it suitable for battery-powered applications. The module also comes with a built-in antenna, which simplifies the integration process and reduces the cost of the overall system.

## - Pin description

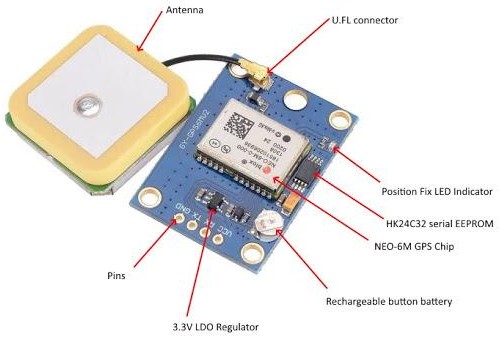


Figure 5.13-GPS module

The pinout of the ublox NEO-6M GPS module is very simple. There is 4 pin available. The module communicates with the microcontroller using the UART communication system. It also supports USB and Serial Peripheral Interface (SPI).

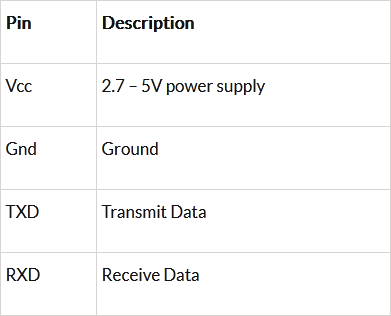


Table 7- ublox NEO-6M GPS pins Description



Figure 5.14- ublox NEO-6M GPS pins

Ublox neo 6M GPS module is TTL compatible and its specifications is given below.

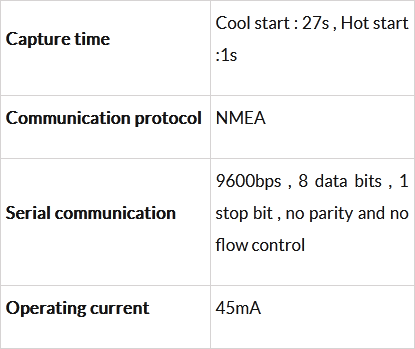


Table 8-ublox NEO-6M GPS Specifications

## Getting Location Data from GPS

The GPS Module will transmit data in multiple strings at 9600 Baud Rate. If we use an UART terminal with 9600 Baud rate, we can see the data received by GPS.

GPS module sends the Realtime tracking position data in NMEA format (see the screenshot above). NMEA format consist several sentences, in which four important sentences are given below. More detail about the NMEA sentence and its data format can be found here.

GPGGA: Global Positioning System Fix Data$

GPGSV: GPS satellites in view$

GPGSA: GPS DOP and active satellites$

GPRMC: Recommended minimum specific GPS/Transit data$

This is the data received by GPS when connected on 9600 baud rate.

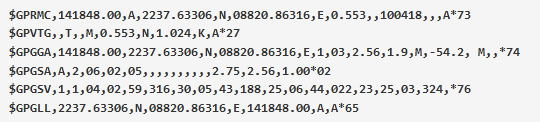


Figure 5.15- output GPS data

When we use GPS module for tracking any location, we only need coordinates and we can find this in $GPGGA string. Only $GPGGA (Global Positioning System Fix Data) String is mostly used in programs and other strings are ignored.



Figure 5.16- (Global Positioning System Fix Data)

What is the meaning of that line?

## Meaning of that line is:-

1. String always starts with a “$” sign
2. GPGGA stands for Global Positioning System Fix Data
3. Comma indicates the separation between two values ”,“

4. 141848.00: GMT time as 14(hr):18(min):48(sec):00(ms)

1. 2237.63306,N: Latitude 22(degree) 37(minutes) 63306(sec) North
2. 08820.86316,E: Longitude 088(degree) 20(minutes) 86316(sec) East
3. 1 : Fix Quantity 0= invalid data, 1= valid data, 2=DGPS fix
4. 03 : Number of satellites currently viewed.
5. 1.0: HDOP
6. 2.56,M : Altitude (Height above sea level in meter)
7. 1.9,M : Geoids height
8. \*74 : checksum

So we need No. 5 and No.6 to gather information about the module location or, where it is located. In this project we have used a GPS Library that provides some functions to extract the latitude and longitude so we don’t have to worry about that.

## Interface GPS with AVR Microcontroller

* + - * 1. Set the configurations of the microcontroller which include Oscillator configuration.
        2. Set the Desired port for LCD including DDR register.
        3. Connect the GPS module to the microcontroller using USART.
        4. Initialize the system UART in ISR mode, with 9600 baud rate and LCD in 4bit mode.
        5. Take two character arrays depending on the Length of Latitude and Longitude.
        6. Receive one character bit at a time and check whether it is started from $ or not.
        7. If $ is received then it is a string, we need to check $GPGGA, this 6 letters including $.
        8. If it is GPGGA, then receive the complete string and set flags.
        9. Then extract the latitude and longitude with directions in two arrays.
        10. Finally print the latitude and longitude arrays in LCD.

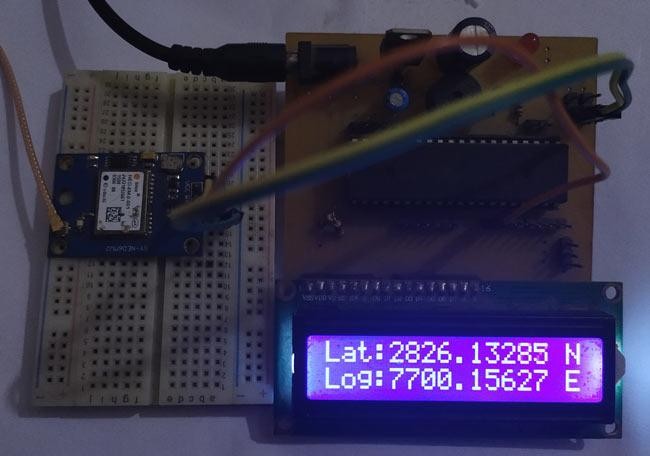


Figure 5.17-GPS final output

import geocoder

from geopy.geocoders import Nominatim

import smtplib

from email.mime.multipart import MIMEMultipart

from email.mime.text import MIMEText

def get\_location\_by\_gps():

try:

# الحصول على الموقع الحالي باستخدام geocoder

g = geocoder.ip('me')

latlng = g.latlng

if latlng:

latitude, longitude = latlng

print(f"Latitude: {latitude}, Longitude: {longitude}")

# استخدام Nominatim لتحويل الإحداثيات إلى عنوان مقروء

geolocator = Nominatim(user\_agent="geoapiExercises")

location = geolocator.reverse((latitude, longitude), exactly\_one=True)

address = location.address if location else "Address not found"

return {

"latitude": latitude,

"longitude": longitude,

"address": address

}

else:

return {

"error": "Could not determine location"

}

except Exception as e:

return {

"error": f"Exception occurred: {str(e)}"

}

def send\_email(receiver\_email, subject, body):

# بيانات حساب البريد الإلكتروني المرسل

sender\_email = "your\_email@gmail.com"

sender\_password = "your\_password"

# تهيئة الرسالة

message = MIMEMultipart()

message["From"] = sender\_email

message["To"] = receiver\_email

message["Subject"] = subject

# إضافة نص الرسالة

message.attach(MIMEText(body, "plain"))

# إرسال الرسالة باستخدام SMTP

try:

with smtplib.SMTP\_SSL("smtp.gmail.com", 465) as server:

server.login(sender\_email, sender\_password)

server.sendmail(sender\_email, receiver\_email, message.as\_string())

print("Email sent successfully")

except Exception as e:

print(f"Failed to send email. Error: {str(e)}")

# استدعاء الدالة للحصول على معلومات الموقع

location\_info = get\_location\_by\_gps()

print(location\_info)

# تجهيز الرسالة للإرسال

receiver\_email = "recipient@example.com" # تعديله إلى البريد الإلكتروني الذي ترغب في إرسال النتائج إليه

subject = "Current Location Information"

body = f"Latitude: {location\_info.get('latitude', 'N/A')}\nLongitude: {location\_info.get('longitude', 'N/A')}\nAddress: {location\_info.get('address', 'N/A')}\nError: {location\_info.get('error', 'None')}"

# إرسال البريد الإلكتروني

send\_email(receiver\_email, subject, body)

## GSM Chip

A GSM (Global System for Mobile Communications) chip is a semiconductor device that provides the functionality required for a mobile phone to connect to a GSM network. The GSM chip is responsible for communication between the mobile phone and the network, including voice and data transmission.

The GSM chip supports various frequency bands, depending on the region and the mobile network, and can operate on different transmission technologies, including 2G, 3G, and 4G/LTE. The chip also supports various communication protocols, including GSM, GPRS, EDGE, and UMTS, which are used for data transmission and internet connectivity.

GSM chips are widely used in mobile phones, smartphones, and other wireless devices, such as modems and routers, that require mobile network connectivity. The chips are also used in various applications, such as vehicle tracking, remote monitoring, and industrial automation, where wireless communication is required.

## SIM 900A GSM Module

The SIM900A GSM module is a compact and low-power GSM/GPRS (General Packet Radio Service) module that is designed to provide mobile network connectivity to a wide range of applications. The module is based on the SIMCom SIM900A chipset and supports the 900/1800 MHz frequency bands.

The SIM900A GSM module is widely used in various applications, including remote monitoring, vehicle tracking, and industrial automation, where wireless communication is required. The module can be connected to a microcontroller or single-board computer using a UART interface and supports various communication protocols, including TCP/IP, HTTP, and FTP.

The module is compact in size and operates on a wide range of supply voltages, from 3.4V to 4.5V, making it suitable for battery-powered applications. The module also supports sleep mode and power-down mode, which helps to conserve power and extend battery life.

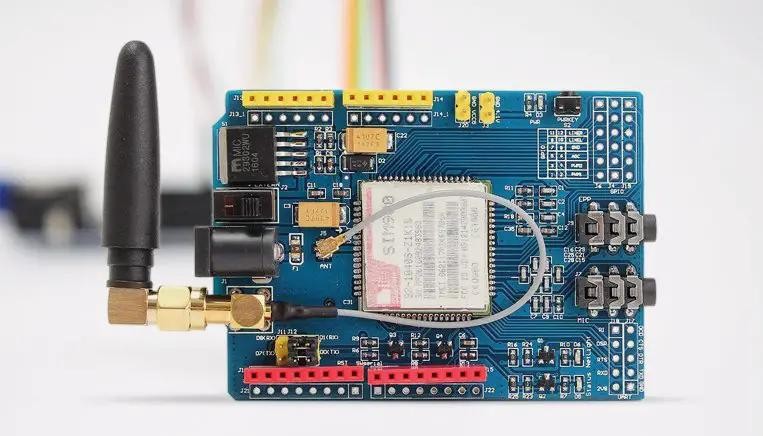


Figure 5.18-SIM 900A GSM Module

## - Pin description

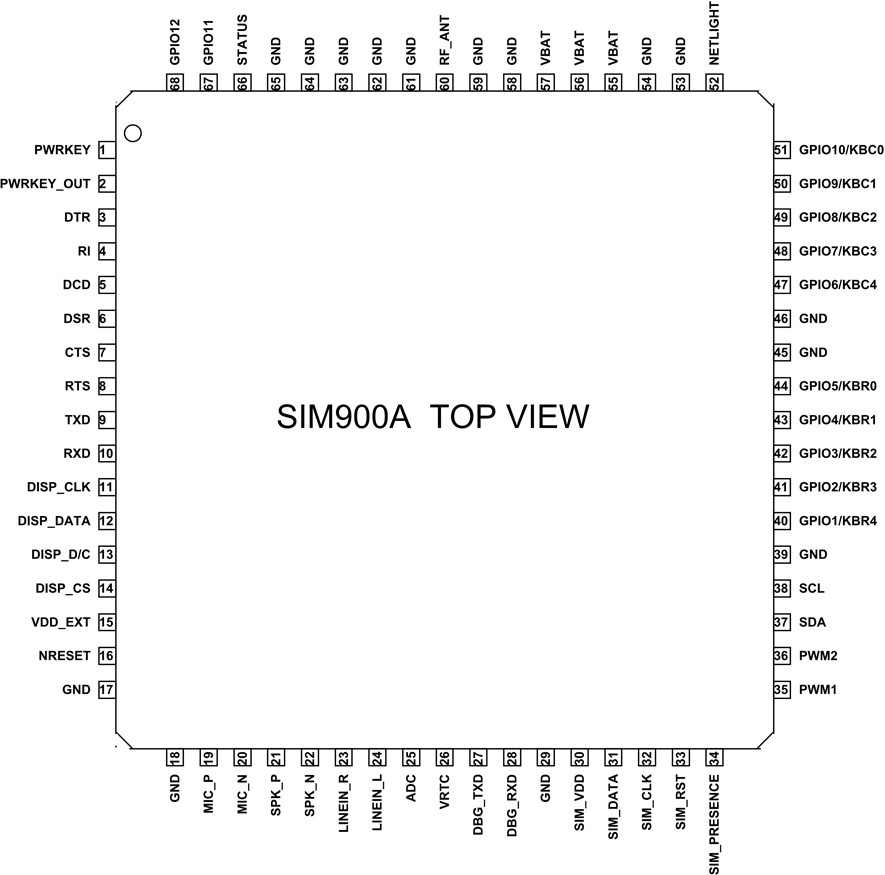


Figure 5.19-SIM900A Pins

The **SIM900A** is a readily available **GSM/GPRS module**,used in many mobile phones and PDA. The module can also be used for developing IOT (Internet of Things) and Embedded Applications. SIM900A is a dual-band GSM/GPRS engine that works on frequencies EGSM 900MHz and DCS 1800MHz. SIM900A features GPRS multi-slot class 10/ class 8 (optional) and supports the GPRS coding schemes CS-1, CS-2, CS-3 and CS-4.

## SIM 900A GSM Module with AVR Microcontroller

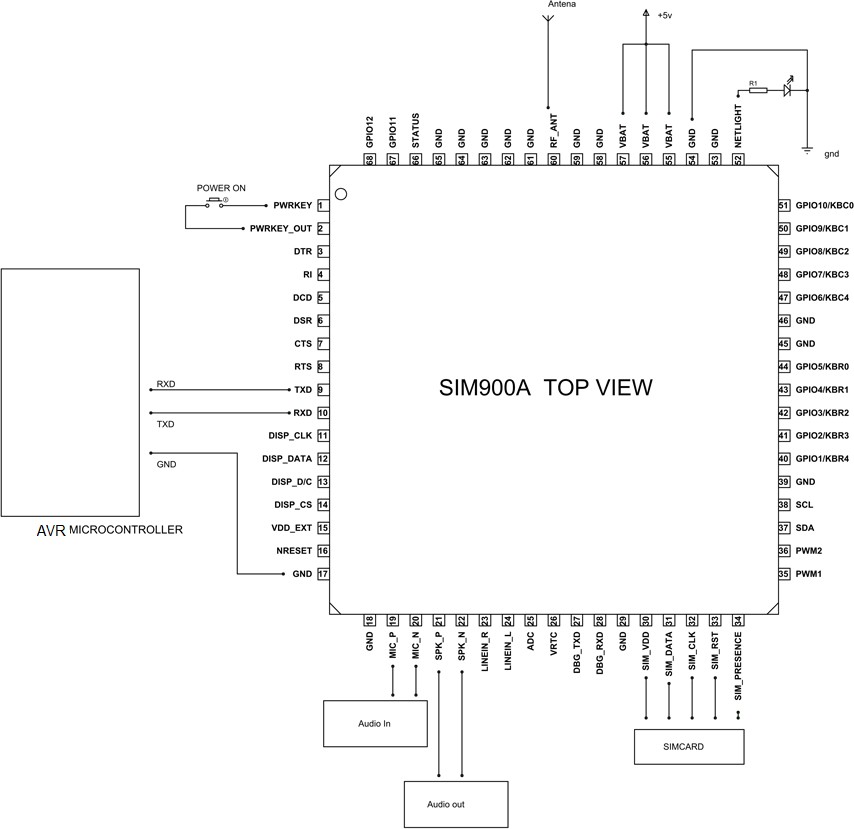


Figure 5.20-GSM connected with AVR

the communication with this module is done through UART or RS232 Interface. The data is sent to the module or received from the module through UART interface.

The module is typically connected to 4.0V standard power supply. It can work on 4.5V regulated power and any higher voltage may damage the module. And the power source should be able to deliver a peak current of 2A. The UART interface is established as shown in figure. All you need to do is connect RXD of module to TXD of Arduino and TXD is connected to RXD of AVR . The ground of controller and module must be connected for voltage reference. Here AUDIO IN is connected to MIC and AUDIO OUT is connected to a speaker or headset. And at last we need to connect

a working GSM SIM card to the module. On powering the module, the NETLIGHT LED will blink periodically to state successful connection.

After all connections are done,we need to write a program for the microcontroller to exchange data with module. Since data exchange sequence between controller and module is really complex we will use libraries prewritten for the module. You can download libraries for controller or module through their websites. Using these libraries makes the communication easy. All you need to do is download these libraries and call them in programs. Once the header file is included, you can use simple commands in the program to tell the controller to send or receive data. The controller sends the data to the module through UART Interface based on protocol setup in libraries. The module sends this data to another GSM user using cellular network. If the module receives any data from the cellular network (or another GSM user) it will transmit it to controller through UART serial communication.

This way we can use GSM900A module to establish cellular connection.

## Smart glasses Microprocessor

In this part We will use microprocessor not microcontroller Because of what this part of orders requires.

## Microprocessor

Microprocessors are central processing units (CPUs) integrated on a single chip. They are designed to execute instructions and perform general-purpose computing tasks.

Microprocessors are widely used in various computing devices, including personal computers, servers, smartphones, and embedded systems.

- In our part of this project we use **Raspberry Pi microprocessor**

## Raspberry Pi



Figure 5.21-Raspberry pi module

Raspberry Pi is a series of single-board computers developed by the Raspberry Pi Foundation in the United Kingdom. The first Raspberry Pi model was released in 2012, and since then, several models have been released with varying specifications.

Raspberry Pi is designed to be a low-cost, credit-card sized computer that can run a variety of operating systems, including Linux, Windows 10 IoT Core, and others. It features a Broadcom system-on-chip (SoC) with an ARM-based CPU, along with integrated graphics, memory, and I/O peripherals such as USB, HDMI, and Ethernet.

Raspberry Pi is widely used in a variety of applications, including as a media center, gaming console, home automation controller, and in education for teaching programming and electronics. Its small size, low power consumption, and versatility make it a popular platform for DIY projects and experimentation.

## Common Raspberry Pi Types

there are several other models of Raspberry Pi with varying specifications, as well as variations of each model with different amounts of RAM and storage options. But Most Commonly used Raspberry Pi Models are:

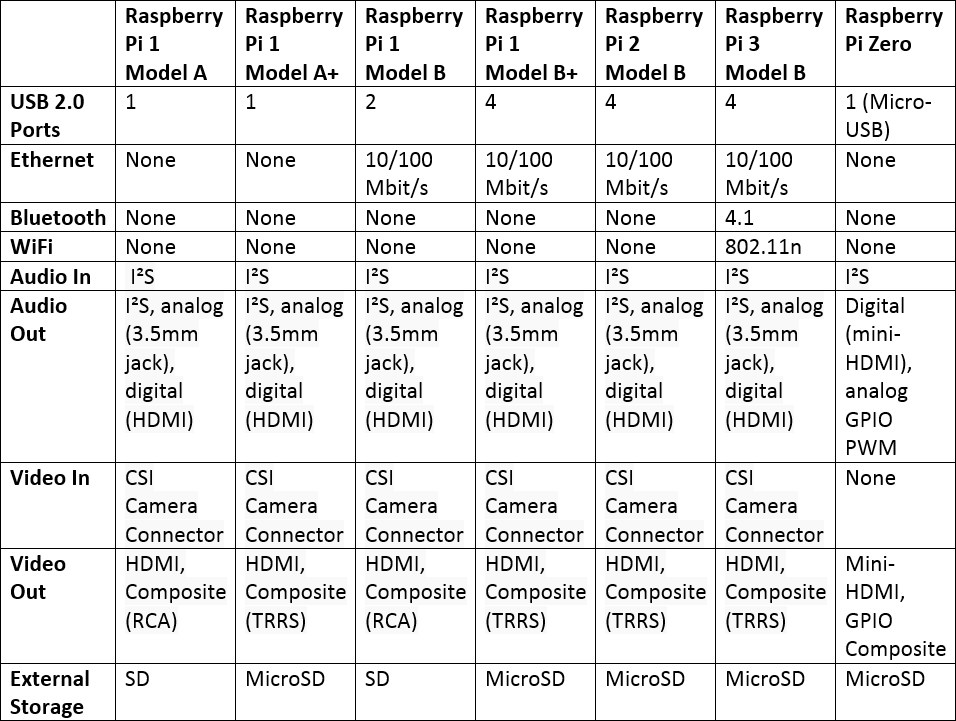


Table 9-Difference between Raspberry Pi Models

**So Raspberry Pi 3 Model B is suitable for (Availability, Cost, Size, USB) we need.**

## Raspberry Pi 3 Model B

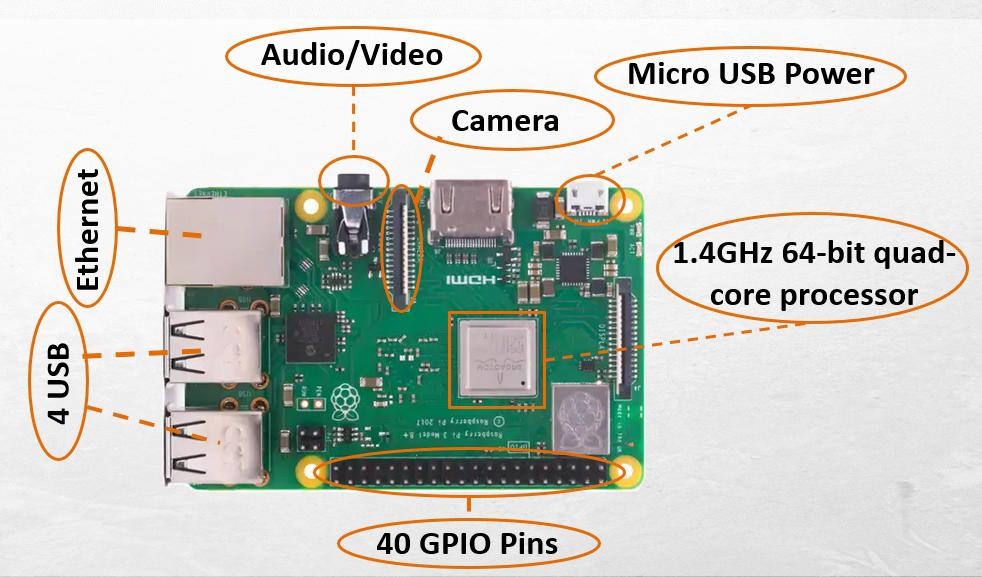


Figure 5.22-Raspberry Pi 3 Model B

**Specifications of Raspberry Pi 3 Model B:**

CPU:

* + - * + 1.2GHz 64-bit quad-core ARM Cortex-A53 CPU GPU:
        + Broadcom Video Core IV @ 400 MHz RAM:
        + 1GB LPDDR2 SDRAM Storage:
        + MicroSD card slot for loading operating system and data storage Ports:
        + 4 USB 2.0 ports
        + HDMI port
        + 3.5mm audio jack
        + CSI camera port
        + DSI display port
        + 10/100 Ethernet port
        + GPIO header (40 pins) Wireless:
        + 802.11n Wireless LAN
        + Bluetooth 4.1 Power:
        + 5V/2.5A DC via micro-USB connector Dimensions:
        + 85 x 56 x 17 mm Weight:
        + 42 g Operating System:
        + Raspbian (a Debian-based Linux distribution) and other operating systems are available for download.

## Raspberry Pi 3 Model B Board

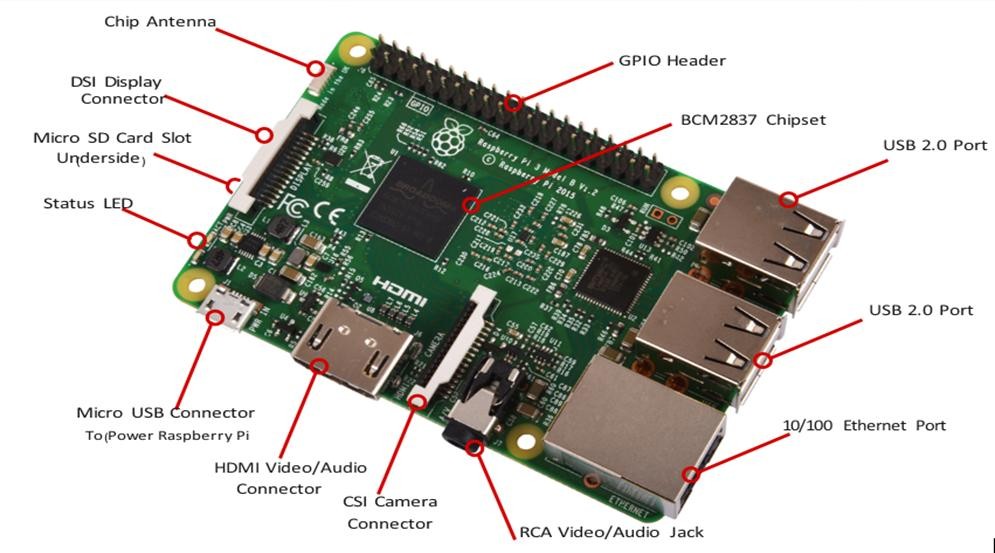


Figure 5.23-Raspberry pi 3 model B Details

The Raspberry Pi 3 Model B has a 40-pin GPIO header that provides a range of input/output (I/O) options and can be used to connect the board to a wide range of external devices and sensors. Here's a brief overview of the different types of pins available on the GPIO header of the Raspberry Pi 3 Model B:

* + - * + Power pins: These include +5V and +3.3V pins, as well as ground (GND) pins, which are used to provide power to the board and external components.
        + GPIO pins: There are a total of 28 General Purpose Input/Output (GPIO) pins that can be used for a variety of tasks, such as controlling LEDs, reading sensors, or communicating with other devices.
        + UART pins: These pins can be used to communicate with other devices using the Universal Asynchronous Receiver/Transmitter (UART) protocol. There are two UARTs available on the Raspberry Pi 3 Model B, with one of them (UART0) also used for the Bluetooth module.
        + I2C pins: These pins can be used to communicate with other devices using the Inter-Integrated Circuit (I2C) protocol, which is commonly used for sensors, displays, and other external components.
        + SPI pins: These pins can be used to communicate with other devices using the Serial Peripheral Interface (SPI) protocol, which is commonly used for displays and other peripherals.
        + PCM pins: These pins can be used to output audio data to external devices using the Pulse Code Modulation (PCM) protocol.
        + PWM pins: These pins can be used to generate Pulse Width Modulation (PWM) signals, which are commonly used for controlling the brightness of LEDs or the speed of motors.
        + ID\_SD and ID\_SC: These two pins are used by the board to communicate with the HAT (Hardware Attached on Top) EEPROM, which is used to identify and configure add-on boards.
        + Camera and Display pins: These pins are used to connect the Raspberry Pi Camera Module and official Raspberry Pi Touchscreen Display, respectively.
        + Power-related pins: In addition to the standard +5V and +3.3V pins, there are also pins for controlling power to the Raspberry Pi, such as the RUN pin (used to reset the board) and the PWR\_LED pin (used to control an external power indicator LED).
        + Ground pins: There are a total of 9 ground (GND) pins on the GPIO header, which are used to provide a common ground reference for the board and external components.
        + Pin numbering: The pins on the GPIO header are numbered from 1 to 40, starting from the top left corner of the header.
        + Pin modes: The Raspberry Pi 3 Model B supports three different pin modes: input, output, and alternate function. Alternate function mode allows pins to be used for specific hardware interfaces, such as SPI or I2C.
        + Pin voltage: The GPIO pins on the Raspberry Pi 3 Model B operate at 3.3V, which means that external components and sensors should be chosen accordingly.
        + Pin protection: The Raspberry Pi 3 Model B has built-in protection circuitry to prevent damage to the board in the event of overvoltage or short circuits on the GPIO pins. However, it's still important to exercise caution when working with external components and to avoid connecting pins to voltages higher than 3.3V.
        + Pin usage examples: Some examples of how the GPIO pins on the Raspberry Pi 3 Model B can be used include controlling LEDs, reading sensors and switches, controlling motors, and interfacing with displays and other external components.
        + Pin voltage levels: The GPIO pins on the Raspberry Pi 3 Model B operate at 3.3V, which means that external components and sensors should be chosen accordingly. Some external components may be designed to operate at 5V, in which case level shifters or voltage dividers may be needed to interface them with the Raspberry Pi.
        + Pin current limitations: Each GPIO pin on the Raspberry Pi 3 Model B is limited to a maximum current of 16mA. This means that external components and sensors that require more current should be connected through a buffer or driver circuit.
        + Pin pull-up and pull-down resistors: The Raspberry Pi 3 Model B has built-in pull- up and pull-down resistors that can be enabled or disabled for each GPIO pin. These resistors can be used to ensure that the GPIO pin has a known state when no external device is connected to it.
        + Pinout diagrams: The Raspberry Pi Foundation provides detailed pinout diagrams for the Raspberry Pi 3 Model B, showing the function of each pin on the GPIO header. These diagrams can be useful for understanding how to connect external components and sensors to the board.
        + Pin usage examples: Some examples of how the GPIO pins on the Raspberry Pi 3 Model B can be used include reading temperature and humidity sensors, detecting motion with a PIR sensor, controlling a servo motor, and interfacing with an OLED display.
        + Pin modes and alternate functions: The Raspberry Pi 3 Model B supports several different pin modes, including input, output, and alternate function. In alternate function mode, pins can be used to interface with specific hardware protocols such as I2C, SPI, and UART.
        + Pin numbering conventions: Different conventions are used for numbering the GPIO pins on the Raspberry Pi 3 Model B, including BCM (Broadcom SOC channel) numbering and physical pin numbering. BCM numbering is commonly used in software and refers to the function of the pin, whereas physical pin numbering refers to the actual physical placement of the pin on the board.
        + Pin usage examples: Some examples of how the GPIO pins on the Raspberry Pi 3 Model B can be used include controlling a robot, creating a weather station, building an alarm system, and interfacing with a touch screen display.
        + Pin usage tips: When working with the GPIO pins on the Raspberry Pi 3 Model B, it's important to avoid short circuits, use appropriate voltage levels for external components, and protect the board from overvoltage or ESD (electrostatic discharge). It's also a good idea to use a breadboard or prototyping board to prototype circuits before connecting them directly to the GPIO header.

It's worth noting that not all of the pins on the GPIO header are available for general- purpose use. Some pins are reserved for specific functions, such as those used by the board's Ethernet and USB ports. Additionally, some pins may have different functions depending on the mode of the board, such as the UART pins, which can be used for Bluetooth communication when the board is in Bluetooth mode. It's important to note that the GPIO pins on the Raspberry Pi 3 Model B are a powerful feature, but working with them requires a basic understanding of electronics and programming. The Raspberry Pi Foundation provides a wealth of resources and documentation to help users get started with the GPIO pins, including tutorials, code examples, and a supportive community of users and developers. So, the GPIO pins on the Raspberry Pi 3 Model B provide a versatile and powerful way to interface with external components and build a wide range of projects.

However, working with them requires some knowledge of electronics and programming, as well as careful attention to safety and best practices.

## Camera

In our project we use 5MP Raspberry Pi Camera Module Rev 1.3 to use in Microprocessor.

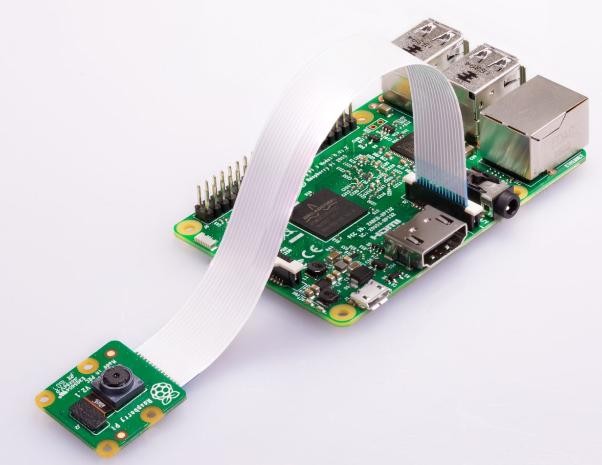


Figure 5.24-5MP Raspberry Pi Camera Module Rev 1.3 Figure 5.24-Camera connected to Microprocessor.

The 5MP Raspberry Pi Camera Module Rev 1.3 is a small, high-quality camera designed specifically for use with the Raspberry Pi single-board computer. It is capable of capturing still images with a resolution of up to 5 megapixels and recording high-definition video at resolutions up to 1080p. The camera module connects to the Raspberry Pi through a ribbon cable and is controlled using software libraries available for the Raspberry Pi's operating system. The camera module also features an onboard processor for image processing and encoding, which offloads some of the computational burden from the Raspberry Pi's CPU. The camera module is equipped with a fixed-focus lens and supports a variety of image- capture modes, including still images, time-lapse photography, and video recording. It is also capable of capturing images in low-light conditions using its infrared filter. The 5MP Raspberry Pi Camera Module Rev 1.3 is a versatile and affordable camera solution for hobbyists, educators, and engineers looking to integrate imaging capabilities into their Raspberry Pi projects.

|  |  |
| --- | --- |
| **Specification** | **Description** |
| Resolution | 5 megapixels |
| Still Image Resolution | Up to 2592 x 1944 pixels |

|  |  |
| --- | --- |
| Video Resolution | Up to 1080p at 30 frames per second |
| Lens | Fixed-focus lens |
| Connectivity | Ribbon cable to Raspberry Pi |
| Image Processing | Onboard processor for image processing and  encoding |
| Additional Features | Time-lapse photography, infrared filter for low- light conditions |
| Applications | Security systems, remote monitoring, robotics projects |
| Cost | Affordable |

## 5.2.9 Battery

Table 10-5MP Raspberry Pi Camera Module Rev 1.3 Description

In the beginning, “Smart Glasses” used Normal battery that provided with the raspberry pi 3 which is 5V and 2. A. Then the team thought that the blind students will use “Smart Glasses” all the time during university time and the way for the battery provided with raspberry not helpful because the student needs to move from one class to another. With the updated “Smart Glasses”, it uses Power Bank 5V and 2.5 A, and the student is able to use the power everywhere in the university or school or outdoor.

# CH6: Conclusion and Future Plans

# Conclusion

The technology still under development, and although this system tried to reduce the burden by collecting some separate features or solving some problems, we hope increase the accuracy of the project soon; Where the artificial intelligence of image analysis is more efficient at the lowest cost. then users will trust the camera instead Of their eyes.

We also hope to establish in our country production lines for safety device for this system , to be accessible to everyone.

* 1. **Future Plans**

One of the most difficult problems facing our system is inability of the system warn the user about potholes or downstairs So, we had an idea that we didn't have time to implement Which is to make glove with very small distance sensors that are fixed at certain angles where they can calculate slopes by the Pythagorean theorem that considers the height of the user then this glove can be linked to the system , just like a device to safer.

# References

1. Wang, T., Wu, D. J., Coates, A., & Ng, A. Y. (2012, November). End-to-end text recognition with convolutional neural networks. In Pattern Recognition (ICPR), 2012 21st International Conference on (pp. 3304-3308). IEEE.
2. Koo, H. I., & Kim, D. H. (2013). Scene text detection via connected component clustering and nontext filtering. IEEE transactions on image processing, 22(6), 2296-2305.
3. Bissacco, A., Cummins, M., Netzer, Y., & Neven, H. (2013). Photoocr: Reading text in uncontrolled conditions. In Proceedings of the IEEE International Conference on Computer Vision (pp. 785-792).
4. Yin, X. C., Yin, X., Huang, K., & Hao, H. W. (2014). Robust text detection in natural scene images. IEEE transactions on pattern analysis and machine intelligence, 36(5), 970- 983.
5. Neumann, L., & Matas, J. (2012, June). Real-time scene text localization and recognition. In Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on (pp. 3538- 3545). IEEE.
6. Neumann, L., & Matas, J. (2013). Scene text localization and recognition with oriented stroke detection. In Proceedings of the IEEE International Conference on Computer Vision (pp. 97-104).
7. Zhou, X., Ylao, C., Wen, H., Wang, Y., Zhou, S., He, W., & Liang, J. (2017). EAST: an efficient and accurate scene text detector. In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (pp. 5551-5560).
8. Jothi, R.; Kayalvizhi, M.; Sagadevan, K. Smart walking stick for visually challenged people. Asian J. Appl. Sci. Technol. 2017, 1, 274–276.
9. World Health Organization. Available online: <http://www.who.int/en/> (accessed on 9 March 2018).
10. Kirchner, C.; Stephen, G.; Chandu, F. Estimated 1987 Prevalence of Non- Institutionalized ‘Severe Visual Impairment’ by Age Based on 1977 Estimated Rates: U.S. AER Yearbook; American Printing House for the Blind: Louisville, KY, USA, 1987.
11. [https://chat.openai.com](https://chat.openai.com/)
12. Foundation, R. P. (n.d.). Teach, Learn, and Make with Raspberry Pi. Retrieved from <https://www.raspberrypi.org/>
13. Mitchell, B. (2019, March 15). 802.11 WiFi Standards Explained. Retrieved from

<https://www.lifewire.com/wireless-standards-802-11a-802-11b-g-n-and-802-11ac->

816553

1. Gudino, M. (2018, March 6). Raspberry Pi 3 vs. Raspberry Pi Zero W. Retrieved from

[https://www.arrow.com/en/research-and-events/articles/raspberry-pi-3-vs-raspberry-pi-](https://www.arrow.com/en/research-and-events/articles/raspberry-pi-3-vs-raspberry-pi-zero) [zero](https://www.arrow.com/en/research-and-events/articles/raspberry-pi-3-vs-raspberry-pi-zero) Savvides, L. (2017, February 15). This 'Star Trek'-like headset helps the legally blind see again.

Retrieved from [https://www.cnet.com/news/esight-video-glasses-restores-sightlegally-](https://www.cnet.com/news/esight-video-glasses-restores-sightlegally-blind-star-trek-visor-headset/)

[blind-star-trek-visor-headset/](https://www.cnet.com/news/esight-video-glasses-restores-sightlegally-blind-star-trek-visor-headset/)

1. News and blogs by author. (2019, April 30). Retrieved from

[https://www.abilitynet.org.uk/news-blogs/three-cool-smart-glasses-help-people-who-](https://www.abilitynet.org.uk/news-blogs/three-cool-smart-glasses-help-people-who-areblind-or-have-sight)

[areblind-or-have-sight](https://www.abilitynet.org.uk/news-blogs/three-cool-smart-glasses-help-people-who-areblind-or-have-sight)

1. Detection based on "ultrasonic"What is an Ultrasonic Sensor? (n.d.). Retrieved from <https://www.keyence.com/ss/products/sensor/sensorbasics/ultrasonic/info/>
2. [https://www.semanticscholar.org/paper/The-Performance-Comparison-of-Object-](https://www.semanticscholar.org/paper/The-Performance-Comparison-of-Object-Detection-in-Ehkan/6032d1c79c3e468bf750b67817b14dde24275d3a)

[Detection-in-Ehkan/6032d1c79c3e468bf750b67817b14dde24275d3a](https://www.semanticscholar.org/paper/The-Performance-Comparison-of-Object-Detection-in-Ehkan/6032d1c79c3e468bf750b67817b14dde24275d3a)

1. [https://www.edureka.co/blog/what-are-the-advantages-and-disadvantages-of-artificial-](https://www.edureka.co/blog/what-are-the-advantages-and-disadvantages-of-artificial-intelligence%20%20%20/)

[/intelligence](https://www.edureka.co/blog/what-are-the-advantages-and-disadvantages-of-artificial-intelligence%20%20%20/)

1. <https://www.opencv-srf.com/p/introduction.html>