# Car Automation


**Project is elaborated by**

1. **Ahmed Hany El-Ghonamy Abd El-Hamid - 21663**
2. **Mahmoud Gamal Ramadan Kotp - 20269**

**Under Supervision of**

1. **Dr. Eman Salah**
2. **Dr. Abd El-Rahman Tamem**

Electrical And Mechatronics Engineering Department

MTI University


**Cairo – 2021**

# CAR AUTOMATION

*12th of April, 2021*
*Graduation Project*

*Mahmoud Gamal*
*Ahmed Hany*
*Supervisor: Dr. Eman Salah Abd El-Rahman Tamem*

# ABSTRACT

This Project mainly aims to automate our regular mechanical car by using two boards Raspberry Pi 3 & ATmega32.

Let's start from deepest part of the project – ATmega32 – which is part of field called Embedded Systems in this part the previously mentioned board will be responsible to make car take basic actions like moving, rotating, and controlling speed plus it will be ready for any urgent interrupt like the car close to crush a wall so immediately it will take a signal from the main board -Raspberry Pi 3 – to take an action at a time which we can call RTOS.

As we are talking about the main board which is Raspberry Pi 3 this board will be responsible for transmitting signal to ATmega32 to take the proper action but how?

By using a sensing device which can be Lidar in modern cars but as known it expensive, so we decided to use camera instead, we will use it to inform us about the surrounded view like human bodies, faces, etc.…

According to the situation of the surrounded view Raspberry Pi 3 will transmit signal to ATmega32 to take the action and every signal will treated as an interrupt to achieve RTOS purpose.

# ACKNOWLEDGEMENT

All praise, glory, and thanks go to Almighty Allah who gave us the strength and patience to carry out this work.

First and foremost, gratitude is to the Modern University for Technology and Information - MTI and its faculty members for the high-quality education they supplied us with through my undergraduate studies.

We're deeply grateful to our supervisors, Dr. Eman Salah, Dr. Abd El-Rahman Tamem, and Prof. Dr. El-Sayed Selit Head of Electrical Engineering Department, for their guidance, inspiration, and support. We consider ourselves very fortunate for being able to work under the guidance of very considerate and encouraging professors.

Our deep appreciation and gratitude go to our thesis advisor Dr. Eman Salah and Dr. Abd El-Rahman Tamem for his constant guidance, scientific knowledge, support, and valuable time. His support was the main key for getting through all challenges and obstacles whether technical or personal and made this work done.

The best and worst moments of our undergraduate journey have been shared with many people, many thanks and appreciation go to our dear parents, brother, friends, and colleagues in Modern University for Technology and Information - MTI for supporting and encouraging us throughout our work in this thesis

# TABLE OF CONTENTS

# LIST OF FIGURES

# CHAPTER 1

## 1. Introduction

At 1886 Carl Benz invented a vehicle powered by gas engine it was a revolutionary invention that changed our world after, but no one was thinking at their days that we will invent a vehicle move without a driver and that what we are going to talk about now using two major fields in our world, Embedded Systems and Computer Vision.

Let's start our talk with Computer Vision and how it works, Computer vision is a branch of artificial intelligence (AI) that allows computers and systems to extract meaningful information from digital images, videos, and other visual inputs — and then act or make recommendations based on that information. If artificial intelligence allows computers to think, computer vision allows them to see, observe, and comprehend.

Computer vision functions similarly to human vision, with the exception that humans have a head start. Human vision has the benefit of lifetimes of context to train how to tell objects apart, how far away they are if they are moving, and if there is something wrong with an image.

Computer vision trains machines to perform these functions, but it must do so in a much shorter amount of time, using cameras, data, and algorithms rather than retinas, optic nerves, and the visual cortex. Because a system trained to inspect products or monitor a production asset can analyze thousands of products or processes per minute, detecting minute flaws or issues, it has the potential to quickly outperform human capabilities.

Computer vision is used in a variety of industries, from energy and utilities to manufacturing and automotive – and the market is expanding. By 2022, it is expected to reach USD 48.6 billion.

How computer learn and train to conclude the information the answer is the Machine Learning, so we can't talk about Computer Vision without mentioning it so what's the Machine Learning?
Machine learning is a subfield of artificial intelligence (AI) and computer science that focuses on using data and algorithms to mimic how humans learn, gradually improving its accuracy.

IBM has a long history of working with machine learning. Arthur Samuel, one of its own, is credited with coining the term "machine learning" with his research on the game of checkers. In 1962, the self-proclaimed checkers master, Robert Nealey, played the game on an IBM 7094 computer and lost to the computer. This feat appears trivial in comparison to what can be done today, but it is regarded as a major milestone in the field of artificial intelligence. [1] Over the next few decades, technological advances in storage and processing power will enable some of the innovative products we know and love today, such as Netflix's recommendation engine or self-driving cars.

Machine learning is a critical component of the rapidly expanding field of data science. Algorithms are trained to make classifications or predictions using statistical methods, revealing key insights in data mining projects. These insights are then used to drive decision-making within applications and businesses, ideally influencing key growth metrics. As big data expands and grows, so will the market demand for data scientists, who will be required to assist in the identification of the most relevant business questions and, ultimately, the data to answer them.

let's back to how Computer Vision works A large amount of data is required for computer vision. It repeatedly runs data analyses until it detects distinctions and, eventually, recognizes images. To train a computer to recognize automobile tires, for example, massive amounts of tire images and tire-related items must be fed into it in order for it to learn the differences and recognize a tire, especially one with no defects.

To accomplish this, two critical technologies are used: deep learning, a type of machine learning, and a convolutional neural network.

Machine learning employs algorithmic models to teach a computer about the context of visual data. If enough data is fed into the model, the computer will "look" at the data and learn to distinguish between images. Algorithms allow the machine to learn on its own rather than having to be programmed to recognize an image.

A CNN aids a machine learning or deep learning model's "look" by breaking down images into pixels that are tagged or labelled. It employs labels to perform convolutions. (a mathematical operation on two functions to produce a third function) and predicts what it is "seeing." In a series of iterations, the neural network runs convolutions and checks the accuracy of its predictions until the predictions begin to come true. It then recognizes or sees images in a manner similar to humans. [2]

A CNN, like a human seeing an image from a distance, first discerns hard edges and simple shapes, then fills in information as it runs prediction iterations. A CNN is used to comprehend individual images. A recurrent neural network (RNN) is used in video applications in a similar way to help computers understand how images in a series of frames are related to one another.

The history of computer vision starts from 60 years, scientists and engineers have been attempting to develop methods for machines to see and understand visual data Experiments began in 1959 when neurophysiologists showed a cat a series of images in an attempt to correlate a response in its brain. They discovered that it responded first to hard edges or lines, which meant that image processing begins with simple shapes such as straight edges. [3]

Around the same time, the first computer image scanning technology was created, allowing computers to digitize and acquire images. Another watershed moment occurred in 1963 when computers were able to convert

two-dimensional images into three-dimensional forms. AI emerged as an academic field of study in the 1960s, which also marked the beginning of the AI quest to solve the human vision problem.

In 1974, optical character recognition (OCR) technology was introduced, which could recognize text printed in any font or typeface. Similarly, using neural networks, intelligent character recognition (ICR) could decipher the handwritten text. OCR and ICR have since made their way into document and invoice processing, vehicle plate recognition, mobile payments, and machine translation.

David Marr, a neuroscientist, established that vision works hierarchically in 1982 and developed algorithms for machines to detect edges, corners, curves, and other basic shapes. Concurrently, computer scientist Kuniko Fukushima created a network of cells capable of pattern recognition. The Noncognition network used convolutional layers in a neural network.

By 2000, the study's focus had shifted to object recognition, and by 2001, the first real-time face recognition applications had emerged. Through the 2000s, standardization of how visual data sets are tagged and annotated emerged. The ImageNet data set was made available in 2010. It contains millions of tagged images from a thousand different object classes and serves as the foundation for today's CNNs and deep learning models. A team from the University of Toronto entered a CNN into an image recognition competition in 2012. Alex Net, the model, significantly reduced the error rate for image recognition. Following this breakthrough, error rates have dropped to a few percent.

We talked about CV meaning, definition, how it works and history of it  now we should talk about Embedded systems, it designed to do some specific task, rather than be a general-purpose computer for multiple tasks. Some also have real-time performance constraints that must be met, for reasons such as safety and usability; others may have low or no performance requirements, allowing the system hardware to be simplified to reduce costs.

Embedded systems are not always standalone devices. Many embedded systems consist of small parts within a larger device that serves a more general purpose. For example, the Gibson Robot Guitar features an embedded system for tuning the strings, but the overall purpose of the Robot Guitar is, of course, to play music. Similarly, an embedded system in an automobile provides a specific function as a subsystem of the car itself.
The program instructions written for embedded systems are referred to as firmware and are stored in read-only memory or flash memory chips. They run with limited computer hardware resources: little memory, small or non-existent keyboard, or screen.

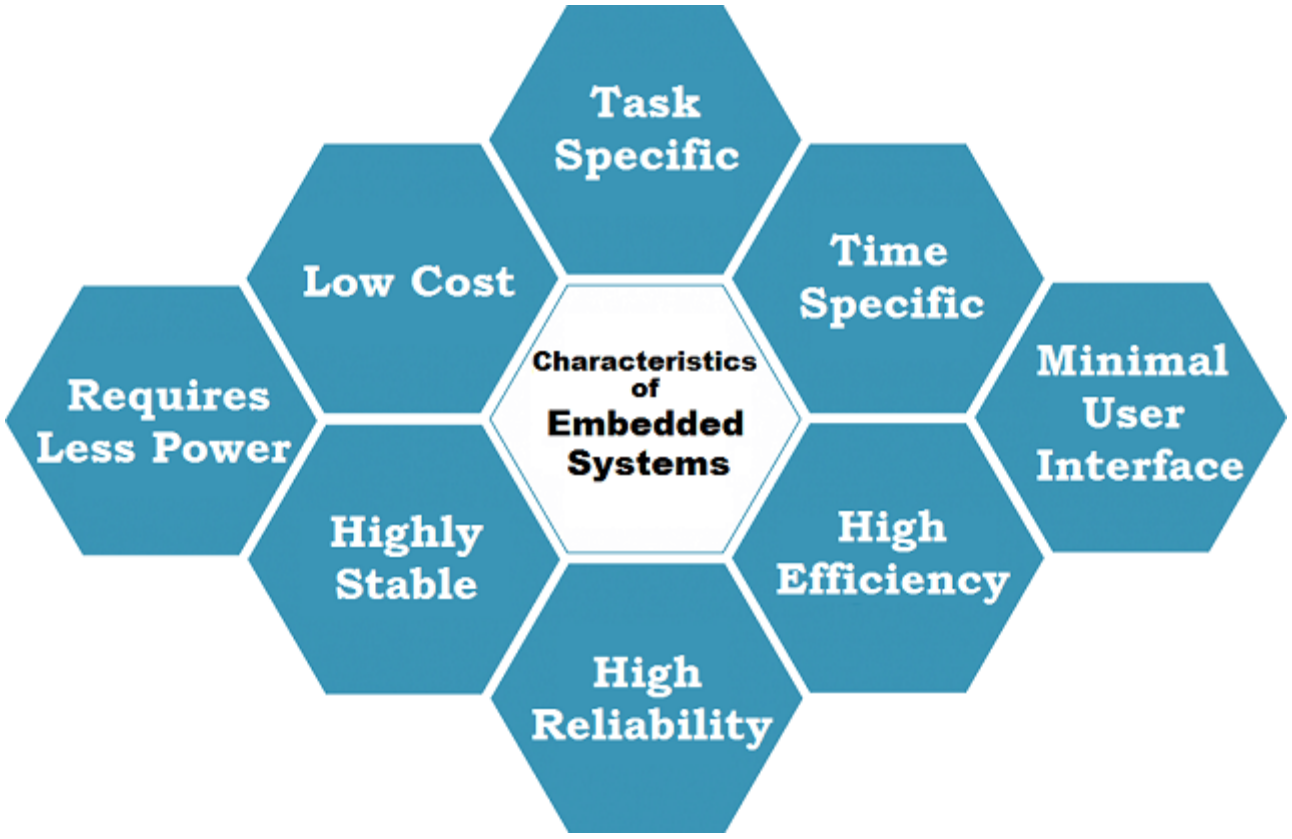



*Figure 1.1*

The origins of the microprocessor and microcontroller can be traced back to the integrated MOS circuit, an integrated circuit chip made of MOSFETs

(metal-oxide-semiconductor field-effect transistors) and developed in the early 1960s. By 1964, MOS chips had achieved higher transistor density and lower manufacturing costs than bipolar chips.
MOS chips further increased complexity at the rate predicted by Moore's law, leading to large-scale integration (LSI) with hundreds of transistors on a single MOS chip by the late 1960s. The application of MOS LSI chips to computing was the basis for the first microprocessors, as engineers began to recognize that a complete computer processor system could be installed on a number of MOS LSI chips. [4]

The first multi-chip microprocessors, the Four-Phase Systems AL1 in 1969 and the Garrett AiResearch MP944 in 1970, were developed using multiple MOS LSI chips. The first single-chip microprocessor was the Intel 4004, which was released in 1971. It was developed by Federico Faggin, using his MOS silicon gate technology, together with Intel engineers Marcian Hoff and Stan Mazor, and Busicom engineer Masatoshi Shima.

## 2. Car Automation

A self-driving car is a vehicle that does not require human intervention to operate. Instead, it employs advanced sensory technology such as lidar, sonar, GPS, radar, or odometry, as well as inertial measurements, to detect environmental changes and adapt to re-establish a safe speed or distance.

Self-driving cars are here, but they are not yet; both statements are correct. It's because we haven't yet reached the pinnacle of self-driving car technology.

Self-driving cars are becoming more popular by the day. The global market for self-driving cars is expected to grow at a 36.2 percent CAGR, reaching USD 173.15 billion by 2023. As a result, businesses that invest in the autonomous car segment stand to benefit greatly. Tesla is currently the industry leader, but the door is wide open for new entrants to capitalize on the growing market.

Computer vision, artificial intelligence, IoT sensors, and radars are all used in self-driving cars. These technologies are implemented to varying degrees, allowing for varying degrees of automation. The Society of Automobile Engineers (SAE) has established global automation standards and divided them into six levels. The following are the six levels of self-driving cars.

Cars are classified into six levels based on the driver's involvement in controlling and driving the vehicle. Levels range from 0 (no automation) to 5, which is a fully autonomous vehicle. Here are the definitions of each level of automation:


*Figure 2.2*

Level 0: There is no automation.
Level 0 automation is the most basic. A level 0 self-driving car, in essence, has no self-driving capabilities at all. The driver must maintain complete control and driving of the vehicle at all times. We can consider cars in their early iterations, as well as cars from the early 2000s, to be level 0 vehicles. These

were the cars that didn't have a computer or even power steering, which is now a standard feature. They only had emergency braking and lane departure warning systems as driver assistance features. Level 0 self-driving cars include vehicles from the early 1900s to the early 2000s, such as the Ford Model T, 2007 Ford Focus, and 2010 Toyota Prius.

Level 1: Driver assistance
The majority of modern automobiles can be classified as autonomous at the level 1 level, as defined by the SAE. The automobiles in this category have built-in capabilities for vehicle operation. The driver is in charge of the vehicle's movement. The vehicle does, however, include one advanced driver-assistance system, such as lane-keeping technology or adaptive cruise control. During the commute, the driver has complete control of the vehicle, and the integrated systems can only assist the driver with tasks like steering or accelerating and de-accelerating the vehicle. Level 1 autonomous vehicles include vehicles you may be familiar with, such as the 2011 Jeep Cherokee and Ford F150.

Level 2: Partial automation
A vehicle with a level 2 rating has two or more advanced driver assistance systems (ADAS). These automated systems can work together to partially relieve the driver of control. The sophistication of the technology used in these assistance systems varies, but they are becoming more common. The driver must remain focused on the driving task, but he can delegate vehicle control to the machine. The driver, on the other hand, must be ready to intervene at any time and take control of the vehicle. Level 2 vehicles include those with popular autonomous features like Tesla Autopilot, General Motors' Super Cruise, and Nissan ProPilot Assist.

Level 3: Conditional automation
The driver, on the other hand, must be ready to intervene at any time and take control of the vehicle. Level 2 vehicles include those with popular autonomous features like Tesla Autopilot, General Motors' Super Cruise, and Nissan ProPilot Assist. The driver is still required at level three, but the

vehicle can take over all aspects of driving tasks under certain conditions. A vehicle that can control itself on a highway and drive at the speed limit, for example, is considered to have level 3 automation. The driver can take his eyes off the road with level 3 autonomy, but his presence is still required in emergencies. The Audi A8 with AI Traffic Jam Pilot system is classified as level 3 automated.

Level 4: High automation
Level 4 automation is where AI systems begin to take over. This is where true autonomous driving comes into play. A vehicle with level 4 autonomy is capable of completing an entire journey without the need for human intervention. It can even operate without a driver, but there are some limitations. If it is raining or snowing, the driver may need to take control because driving conditions can become hazardous. There are currently no level 4 self-driving cars available to consumers. However, there are test vehicles that fall into the level 4 category. Waymo LLC, for example, is testing its level 4 vehicles. A Waymo vehicle does not have a safety driver and can complete the journey on its own. These vehicles, however, are currently being tested under controlled conditions. The vehicles are being tested in Arizona, specifically in dry conditions.

Level 5: Full automation
Most industry experts believe that level 5 cars will take more than a decade to become a reality. A level 5 car will be completely self-driving. The vehicle will be able to operate autonomously in any given situation. At this level, there is no need for human controls, a gear-shifting system, pedals, or even a steering wheel. In a level 5 self-driving car, humans will be merely passengers. Level 5 automated cars are a long way off, but the automobile industry's leaders are very vocal about them. Their ultimate goal is to make level 5 cars commercially available. General Motors has been experimenting with autonomous test vehicles called Cruise in order to make these vehicles a reality as soon as possible. Nuro, too, is testing out level 5 autonomous vehicles. The cars are currently being tested to deliver groceries over short

distances. The vehicles carry groceries from the store to the customer's house, and once the groceries are taken out, they drive back to the store. [5]

## 3. Main Objectives

We aim to drive our car to the owner by detecting face or body of him by the camera, Raspberry Pi will recognize the captured frame from the camera then it will detect the position of the owner then it will send signal to ATmega32 to drive the car to the owner, this the detailed points of our project:

1. Raspberry Pi 3
    1. Computer Vision
        - Face Detection
        - Full Body Detection
        - Owner Body Detection

            It aims to detect the car owner's body to make car follow him.
        - Owner Face Detection
        - Owner Position Determination
    2. Car Owner Info
        - Face Samples
        - Owner Name
        - Owner T-Shirt Color

2. ATmega32
    1. Car Control Functions
        - Speed
        - Move Forward
        - Move Forward Right
        - Move Forward Left

- Move Backward
- Move Backward Right
- Move Backward Left
- Rotate Right
- Rotate Left
- Stop

2. RTOS
    - External Interrupts
        When raspberry pi transmit signal to atmega32 it will cause interrupt.
    - Schedule Tasks
    - Timer Functions

3. Communication Protocol
    1. Implementing UART Driver in Both Boards
    2. Raspberry Pi Should Transmit Data (Function Signal) to ATmega32 Every Specific Time

# CHAPTER 2

## 1. Computer Vision Components Selection

### 1. 1 Raspberry Pi VS Alternatives

The Raspberry Pi is a small, low-cost development board or computer that serves the purpose of computing and learning to program for people of all ages. It is used for small-scale projects by beginners, hackers, and artists, and it can also function as a desktop computer. It can be used to browse the internet, watch high-definition videos, play games, create artificial intelligence (AI) assistants, and record videos. [6]

It is the most well-known single-board computer, but there are numerous raspberry pi alternatives that can do the same job or create even larger projects. Some boards have machine learning capabilities, while others are designed for beginners and do-it-yourself projects. The options are listed below.

Top Raspberry Pi Alternatives

1. Onion Omega2+
2. Banana Pi M3
3. NanoPi NEO4
4. Odroid XU4
5. RockPi 4 Model C
6. NVIDIA Jetson Nano Developer Kit
7. ASUS Tinker Board S

## 8. Udoo Bolt V8

### 1. Onion Omega2+

The Onion Omega2+ is one of the most affordable IoT (Internet of Things) single board computers, featuring a Linux operating system and built-in Wi-Fi. It is designed in such a way that users of all skill levels can create hardware applications with it. When Omega is plugged in, the operating system immediately improves, and the user can begin developing in their programming language or creating web applications.

It uses an MT7688 SoC with a 580MHz MIPS CPU and supports 2.4GHz IEEE 802.11 b/g/n Wi-Fi as well as 10M/100M Wired Ethernet Network connectivity. It has 128 MB DDR2 RAM, 32 MB onboard flash storage, a MicroSD slot, USB 2.0, and 12 GPIO pins and runs at 3.3V. [7]

### 2. Banana Pi M3

The Banana Pi M3 is powered by an Octa-core processor, a PowerVR GPU, and 2GB of RAM. An open platform device is ideal for people who want to learn about developer technology and create projects with it. It has an in-built microphone and IR receiver, as well as power and reset buttons.

It has Gigabit Ethernet, SATA, two USB ports, Wi-Fi, Bluetooth, and an HDMI port. It supports a wide range of operating systems, including Ubuntu, Android, Lubuntu, Debian, and Raspbian. It has a CSI camera interface as well as a DSI display interface and operates at 5V, 2A.

### 3. NanoPi NEO4’

The NanoPi NEO4 is a low-cost six-core single-board computer with a 60x45mm footprint and 1GB DDR3-1866 RAM. It combines dual and quad-core Cortex processors and is powered by the Rockchip RK3399 chipset. Its

small size makes it easy to integrate into hardware projects, and it supports a variety of operating systems including Android, Lubuntu, FriendlyCore, FriendlyDesktop, and Armbian.

It has an eMMC socket and can support up to 128 GB microSD cards. It has four USB ports, one HDMI port, and one Gigabit Ethernet port. It requires a 5V/3A power supply and is ideal for machine learning, artificial intelligence, robots, industrial cameras, games, blockchain, and other applications.

4. Odroid XU4

With an octa-core Heterogeneous Multi-Processing ARM CPU and 2GB of high-speed RAM, the Odroid XU4 has powerful and energy-efficient hardware. It is based on a Samsung Exynos5422 SoC and includes an HDMI 1.4 port with 1080p output, 3 USB ports, Gigabit Ethernet, and a 30-pin GPIO header. It can run a full desktop version of Ubuntu, supports Android, and includes a power button.

It supports eMMC 5.0 storage and comes with a cooling system and a power adapter. It has a fast ethernet connection and excellent performance, as well as faster booting, allowing users to browse websites, network, and play 3D games.

5. RockPi 4 Model C

The Rock 4 Model C features a six-core Rockchip RK3399 processor and up to 4GB LPDDR4 RAM, as well as support for a 4K display with a refresh rate of 60 Hz. It has two external displays, as well as a micro-HDMI and a mini-DisplayPort port. Storage options include a microSD card, an external NVMe

SSD, and an optional eMMC module. It also has a 3.5 mm audio port, Gigabit Ethernet, a 40 pin GPIO interface, and a MIPI-CSI2 camera connector.

It has a Wi-Fi connection, as well as Bluetooth and a USB port. It can be used to build retro gaming arcades, robotics, and AI frameworks for object recognition and machine learning.

6. NVIDIA Jetson Nano Developer Kit

The NVIDIA Jetson Nano Developer Kit is ideal for modern AI, with a quad-core ARM A57 CPU running at 1.43 GHz and a 128 CUDA core Maxwell GPU with 4GB LPDDR4 memory. It has a 16 GB eMMC storage capacity and a 5-watt power supply. It also includes four USB ports and an HDMI 2.0 display.

*Figure 2.1*

This is a small, powerful computer that aids in the parallel operation of multiple neural networks for speech processing, image recognition, object detection, and segmentation. Many popular AI frameworks are supported, including PyTorch, TensorFlow, Caffe, and MXNet.

7. ASUS Tinker Board S

The Tinker Board S is a single-board computer that provides excellent durability, stability, and user experience for DIY projects and other applications. It has a powerful quad-core ARM-based processor, the Rockchip RK3288, which outperforms other similar boards. The board looks similar to the Raspberry Pi and has 2GB RAM, 16GB eMMC storage, and a 1 Gigabit Ethernet port. [8]

It has a 40-pin GPIO color-coded header block that is Raspberry Pi compatible. It has 4 USB ports, an HDMI port, Bluetooth, and a CSI and DSI interface, in addition to 4 USB ports, an HDMI port, and a CSI and DSI interface. The powerful and energy-efficient design is compatible with next-generation graphics and can be used for gaming, computer vision, high-quality media playback, gesture recognition, image stabilization, and processing, among other things.

8. Udoo Bolt V8

The Udoo Bolt V8 is a portable supercomputer with an AMD Ryzen Embedded V1000 SoC that can reach 3.6 GHz. For AR, VR, and AI projects, it is nearly twice as fast as the MacBook Pro 13. It has a 32GB eMMC 5.0 High-Speed Drive and a 2x DDR4 Dual-channel 64-bit SO-DIMM socket with ECC support up to 32GB 2400 Mt/s.

It can provide an excellent gaming experience and has applications such as AAA gaming, cryptocurrency mining, high-end VR, AI, IoT, real-time Big Data analysis, and so on.

9. Micro Controllers

Despite it the cheapest one of the pervious mentioned boards but it's not comparable with these boards because it can be considered unlimited resources in this comparison according to this Micro Controllers can't do Image Processes that Raspberry pi or these boards do.

Conclusion

Despite its popularity, the original Raspberry Pi faces stiff competition from the numerous raspberry pi alternatives listed. Every year, a number of companies develop and release single-board computers with significant advantages over the Raspberry Pi. However, the user should examine all of the technical specifications of each single-board computer and decision making based on their intended use.

## 1. 2 LiDAR VS Cameras

Experts in the self-driving car industry are divided on whether LiDAR (Light Detection and Ranging) or cameras are better suited for SAE Level 4 and Level 5 driving. The debate is whether to use LiDAR in conjunction with camera systems or just camera systems without the LiDAR. Waymo, Cruise, Uber, and Velodyne are among the companies that support LiDAR. Tesla has been the least enthusiastic about LiDAR, preferring camera systems. Which is the most effective solution? The answer is which is better at identifying and recognizing objects. Self-driving cars must be able to recognize what they see on the road. Is one superior to the other?

LiDAR

The application of LiDAR is not limited to self-driving cars. It is used in a variety of fields, including meteorology, seismology, geology, and atmospheric physics. LiDAR detects objects using light pulses, similar to how radar detects objects using radio waves. These pulses can determine an object's distance and range, providing much-needed data to self-driving cars. To avoid a collision, for example, LiDAR can detect the distance to an object and apply the brakes to slow the vehicle down. Engineers have used LiDAR for a variety of applications, including self-driving cars because it is a proven technology for measuring distance. Based on the readings from the light pulses, LiDAR can assist self-driving cars in creating a visual map. The LiDAR system sends thousands of pulses per second to create a 3D map, which is

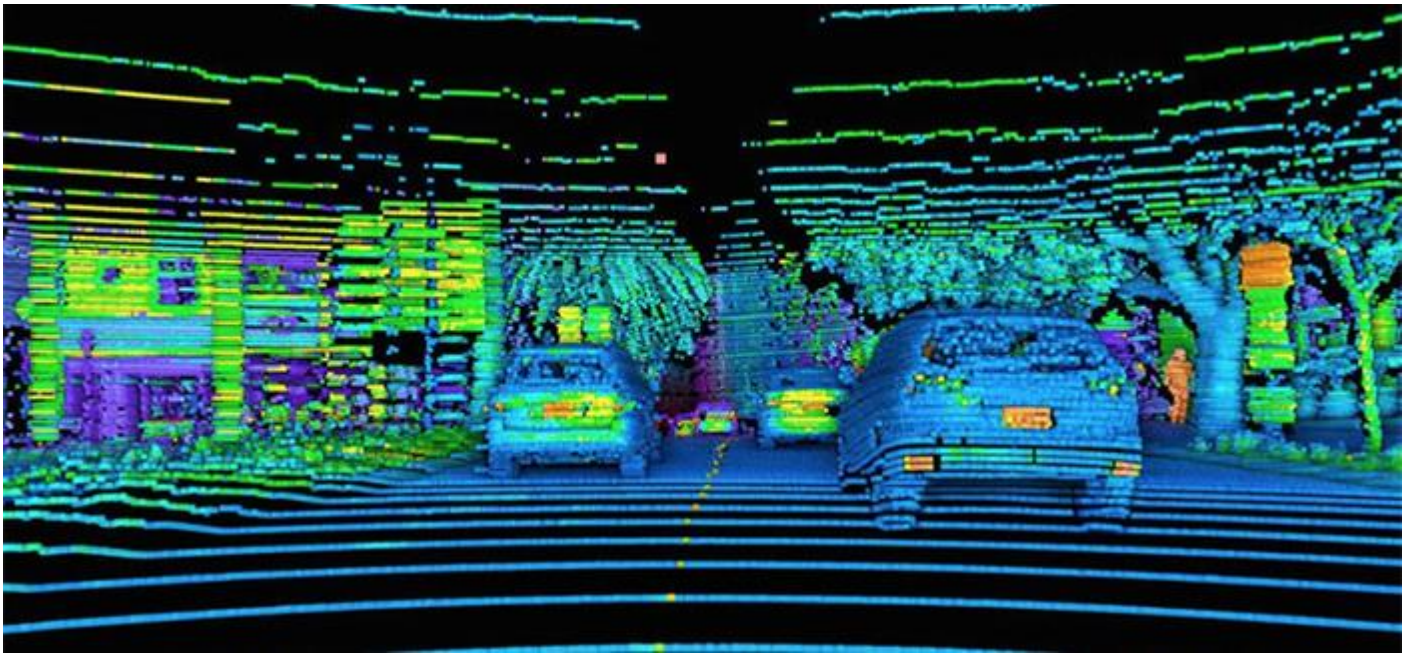


*Figure 2.2*

then used by the car's onboard software to provide information about its surroundings. This provides a 360-degree view, allowing the car to drive in any condition. LiDAR is used in conjunction with cameras in self-driving cars, so it is not a stand-alone solution. [9]

## Cameras

If you want to drive like a human, visual object recognition is the way to go. That is the primary justification for using camera systems. Cameras produce images that AI-powered software can analyze with high precision. Tesla models' cameras are used by the Autopilot self-driving feature to provide a 360-degree view of their surroundings. It is entirely visual and does not rely on ranging and detection as LiDAR does. Instead of light pulses, the cameras use visual data returned from the optics in the lens to onboard software for analysis. Objects can be identified to provide card information while driving thanks to advancements in neural networks and computer vision algorithms. This assists the car in avoiding collisions, slowing down in traffic, safely changing lanes, and even reading text from road or highway signs using OCR (Optical Character Recognition). So far, Tesla has demonstrated that self-driving cars can function without LIDAR by employing cameras.

## Conclusion

If safety is our top priority, sensor fusion combining the best elements of LiDAR and camera systems will be required. In terms of public safety, a combination of LiDAR and other sensors, including cameras, can provide a lot. If the visioning system (e.g., software, sensors) becomes more accurate and reliable for public safety, LiDAR may become obsolete. After all, reducing accidents caused by human error is one of the primary reasons for self-driving cars. The software is what connects LiDAR and cameras in self-driving cars. To analyze data, both systems employ AI techniques such as machine learning and neural networks. As the algorithms improve, the results should improve object recognition accuracy and allow self-driving cars to make better decisions. This could mean the difference between a collision and safe driving. This is not a straightforward problem with a straightforward solution. When it comes to making critical decisions, machines are not wired the same way as humans. More data and training are required for software developers to

improve. To accommodate self-driving cars, the current infrastructure will almost certainly need to be modified as well (e.g., V2X). The debate will continue until self-driving cars demonstrate consistent data pointing to the use of one technology over the other.

## 2. Raspberry Pi

The Raspberry Pi is a credit card-sized computer that runs Linux on an ARM processor. The Raspberry Pi Model B, revision 2.0, has 512 MB of RAM, an Ethernet port, HDMI output, RCA composite video output, audio output, two USB ports, and 0.1″-spaced pins for general-purpose inputs and outputs (GPIO). The Raspberry Pi requires an SD card that contains an operating system (not included). The Raspberry Pi is very popular, and there are numerous example projects and resources available online.

The Raspberry Pi is a credit card-sized computer that is powered by the BCM2835 system-on-chip (SoC), which includes an ARM11 processor and a powerful GPU. The Raspberry Pi supports a number of Linux distributions, including Debian, Fedora, and Arch Linux. The Raspberry Pi Model B, revision 2.0, is the item in question.

The Raspberry Pi Foundation created the Raspberry Pi to provide a low-cost platform for computer programming experimentation and education. The Raspberry Pi can perform many of the same tasks as a standard desktop PC, such as word processing, spreadsheets, high-definition video, games, and programming. USB peripherals like keyboards and mice can be connected to the board's two USB ports. The Raspberry Pi, with its 0.1″-spaced GPIO header and small size, can also be used as a programmable controller in a wide range of robotics and electronics applications. Over two million

Raspberry Pis have been sold, and a wealth of Raspberry Pi resources are available online.

Features

- 512 MB of RAM
- Ethernet port
- Two USB ports
- Two video output options: HDMI or composite
- 3.5 mm audio output jack
- 26-pin GPIO header with 0.1″-spaced male pins that are compatible with our 2×13 stackable female headers and the female ends of our premium jumper wires.

What you will need to use the Raspberry Pi, you will need a few additional things that are not included:

- A 5V power source with a micro-USB connector. We recommend this 5 VDC 1 A wall power adapter and a USB A-to-Micro-B cable.
- An SD card with an operating system on it, which also serves as the main storage for the device.
- Input and output devices, such as a keyboard and monitor.

## 2.1 NOOBS OS Installation

The Raspberry Pi is a fantastic device, but it is useless without an operating system. Fortunately, selecting and installing a suitable operating system for your Raspberry Pi has never been simpler. NOOBS, or "New Out of Box Software," is a simple method. NOOBS, as the name implies, is ideal for Pi newcomers. It allows you to select your preferred operating system and install

it right away. But how do you install NOOBS? Here's a step-by-step guide to installing NOOBS on the Raspberry Pi.

Fortunately for us, the procedure is extremely simple. A Raspberry Pi, a computer, and an SD or microSD card are all you'll need. See the complete instructions below. [10]

How to Setup NOOBS on a Raspberry Pi

Our article is titled "How to Install NOOBS on the Raspberry Pi," but what we're actually doing is installing it on a flash drive, booting to the drive on the Raspberry Pi, and then using NOOBS to choose and install an operating system.

When we get to that point, NOOBS has a plethora of operating systems to choose from, the most notable of which is Raspbian. But for now, let's focus on how to install NOOBS on the Raspberry Pi. In our final step, we will go over the operating system installations briefly.

The easy way out is to purchase a NOOBS SD card.

Installing NOOBS on an SD card is not difficult, but it is also not required. If you prefer, you can purchase an SD card pre-loaded with NOOBS. If you take that path, you can skip all the way to the end!

But if you want to do things yourself, keep reading.

What you'll need to get NOOBS installed on the Raspberry Pi? This is a very simple project. Aside from your Raspberry Pi and the necessary peripherals, you will require the following:

- A computer that includes an SD card slot
- An SD or microSD card with at least 8 GB of storage space

Step 1: Download NOOBS and extract it

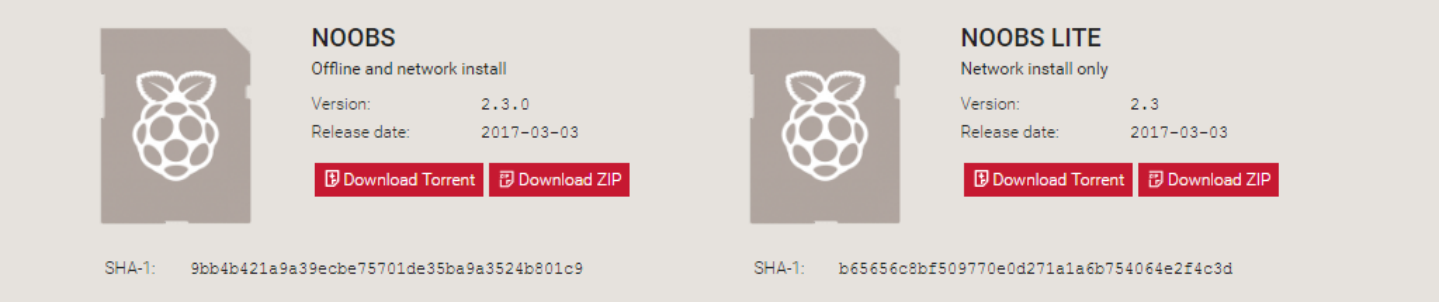



*Figure 2.3*

You'll be using your computer to put NOOBS on an SD card, so the first step is to get NOOBS onto your computer!

**Click here to head to the NOOBS download page.**

You can choose between NOOBS and "NOOBS Lite" on the NOOBS download page. NOOBS includes a full version of Raspbian, allowing you to install that operating system without ever connecting to the internet. NOOBS Lite, on the other hand, requires a network connection to install any of the operating systems offered by NOOBS, including Raspbian. [11]

Go ahead and select whichever version you prefer. NOOBS will download as a.zip file, so extract it before you do anything else.

Step 2: Format an SD card

Now you should insert your SD card into the corresponding slot on your computer. You'll want to format it as FAT32. There are several options for accomplishing this:

Use the SD Association's Formatting Tool on Mac or Windows (Mac users can also just use the disc utility). Check that the “Format size adjustment” option is turned on. Then format it in FAT (or MS-DOS).

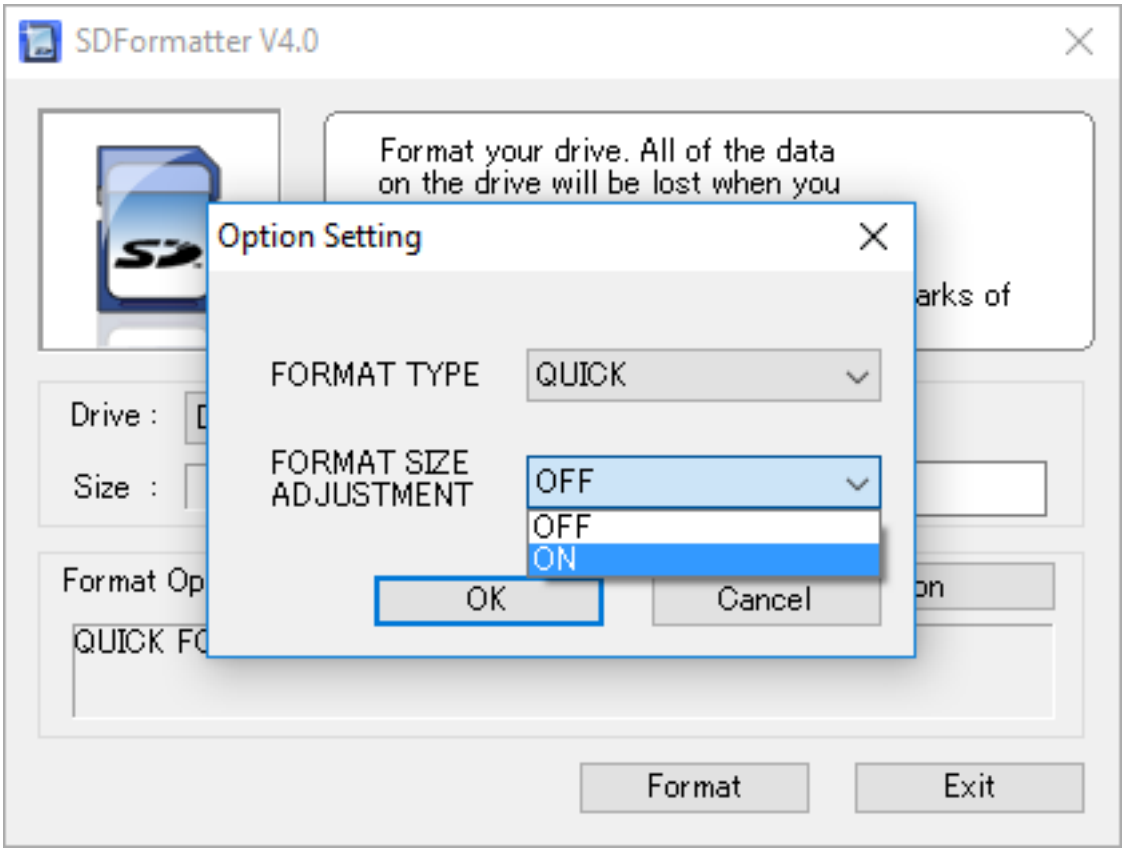



*Figure 2.4*

Step 3: Put the NOOBS files on the SD card

Simply drag and drop the NOOBS files onto your freshly formatted SD card. You only want the files, so if your.zip was extracted to a folder, open that folder and select only the contents.

Step 4: Put your SD card into your Raspberry Pi and boot it up

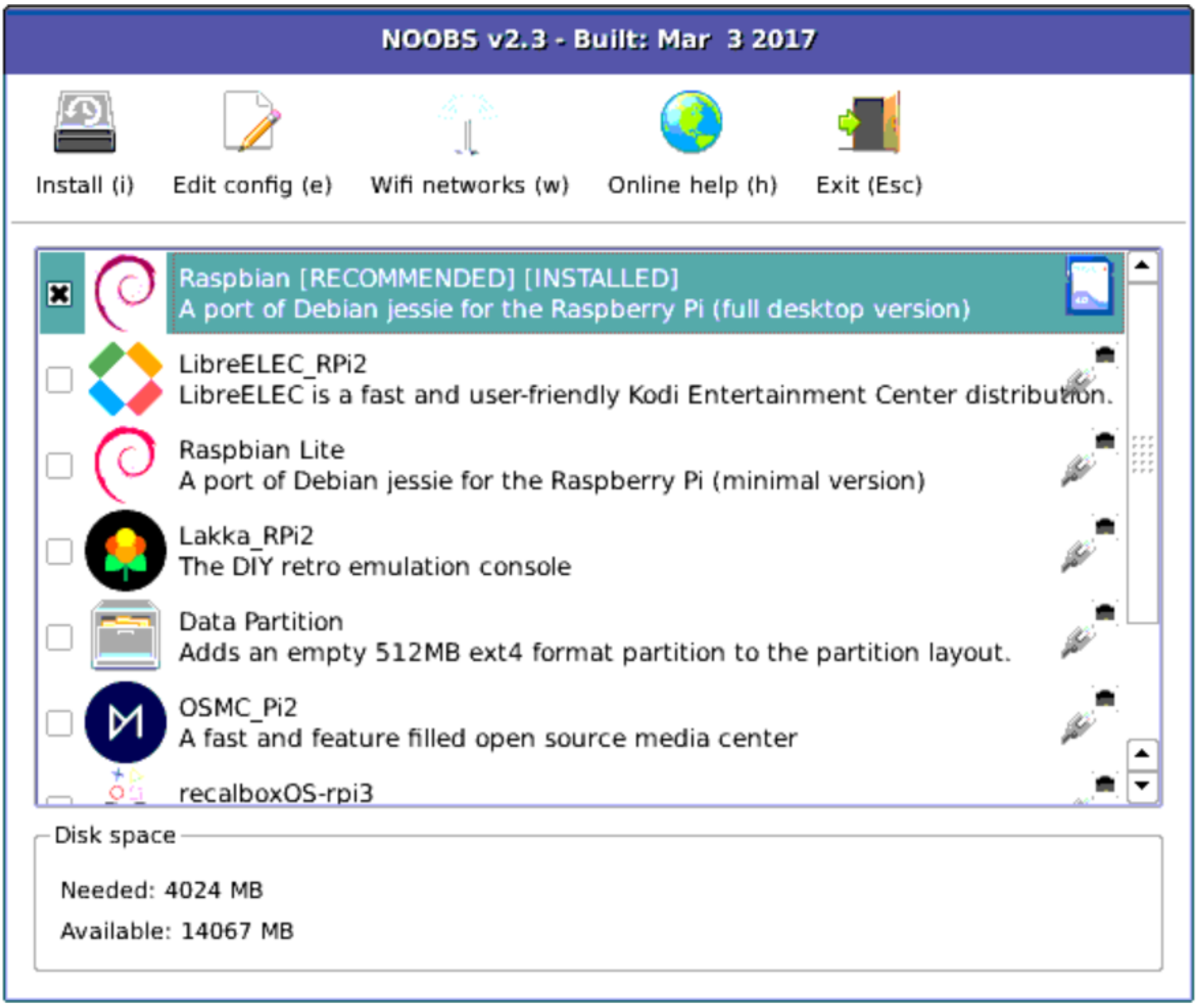



*Figure 2.5*

Once you have NOOBS on your SD card, it is incredibly simple to use. Simply insert the SD card into your Raspberry Pi and boot it up. As previously stated, while this guide is titled "How to Install NOOBS on the Raspberry Pi," the end goal is to install an operating system such as Raspbian, LibreELEC, OSMC, or any of the others that NOOBS provides access to.

This is the stage at which that occurs. After booting into NOOBS, you'll be presented with a menu from which you can select which operating system to install on your Pi. Because NOOBS cleverly adapts to your generation and model of Raspberry Pi, your menu may look slightly different than the one in the screenshot above. [12]

Which operating system should you use? That is entirely up to you. Raspbian is probably the most popular, and you'll find plenty of projects that use it on our site. OSMC serves as a media center, and LibreELEC launches the popular media center app Kodi. In the end, it all comes down to personal preference!

Once you've made your decision, simply click "Install" and relax. Your Pi will now boot directly into that operating system. Isn't it simple?

And you're not locked in if you don't like the operating system you choose. Simply hold down the SHIFT key while booting up to return to the NOOBS menu and try out a different option.

## 2.2 Recommended Configuration

Before getting into to the development we need to configure raspberry pi to improve ability of writing code so do as the following:

Step 1: Enable SSH

This will help us to remote session on raspberry pi and controls terminal, view, and files transfer.

There two ways to enable SSH first one is through the system as follows:

- Click the raspberry logo at the top-left corner.
- Select Preferences > Raspberry Pi Configuration.

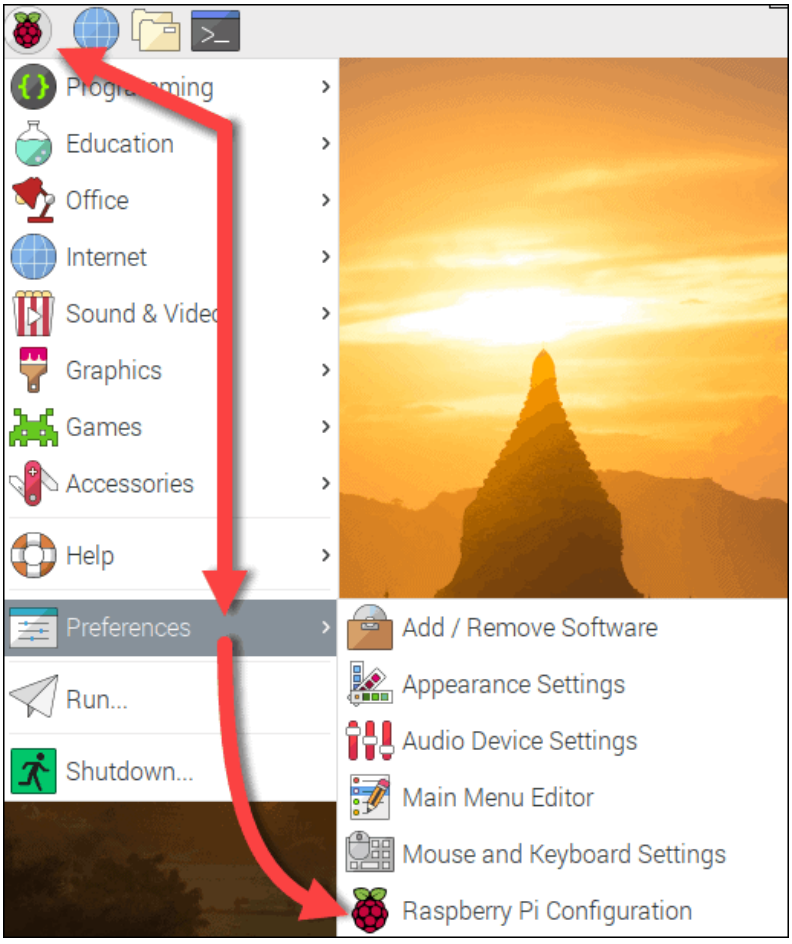



*Figure 2.6*

- Navigate to the Interfaces tab in the configuration window.
- Enable SSH in the second line.

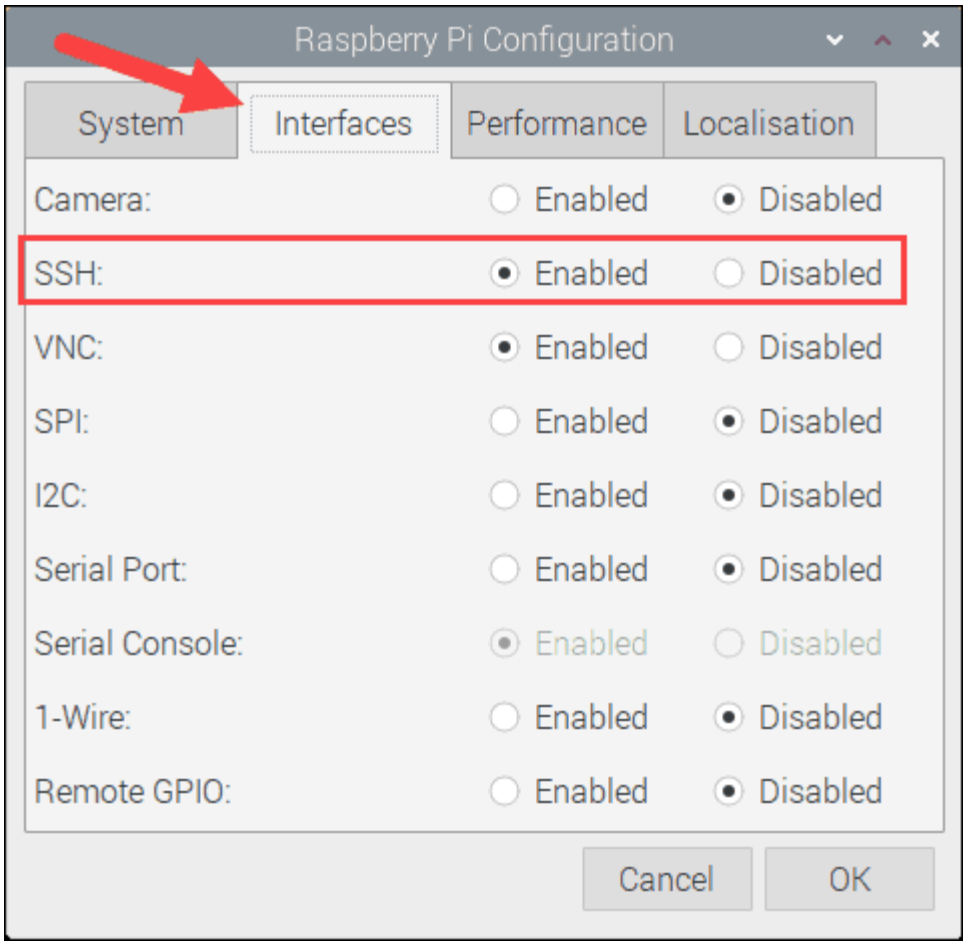



*Figure 2.7*

- Click OK to save the changes.

That's it. Your Raspberry Pi is now accessible via SSH. Make sure the device is connected to the internet before trying to establish an SSH session.

Second way is enabling SSH on raspberry pi in the terminal:

- Open the terminal on your Raspberry Pi and run the tool by typing:

```
sudo raspi-config
```

A BIOS-looking raspi-config tool loads.

- Use the arrows on your keyboard to select Interfacing Options.

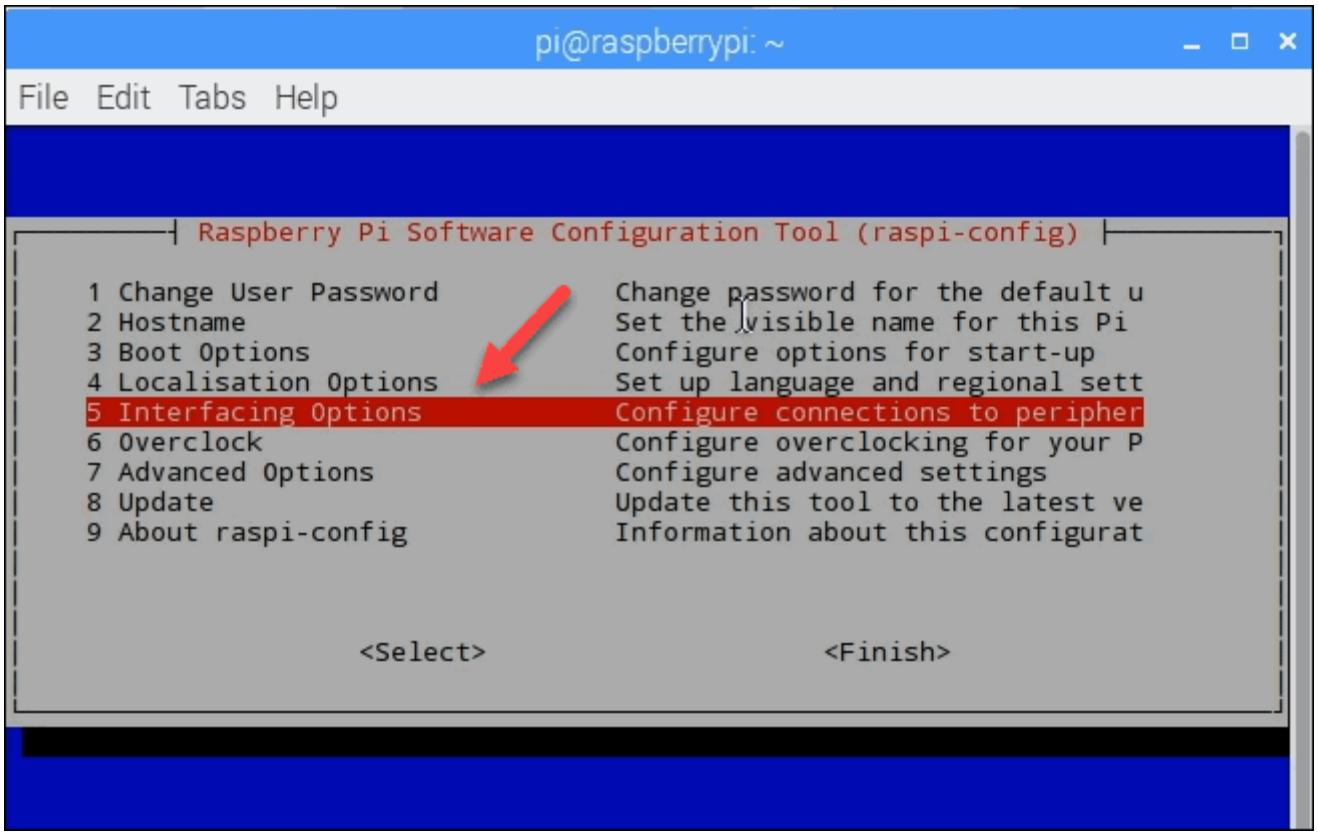



*Figure 2.8*

- Select the P2 SSH option on the list.

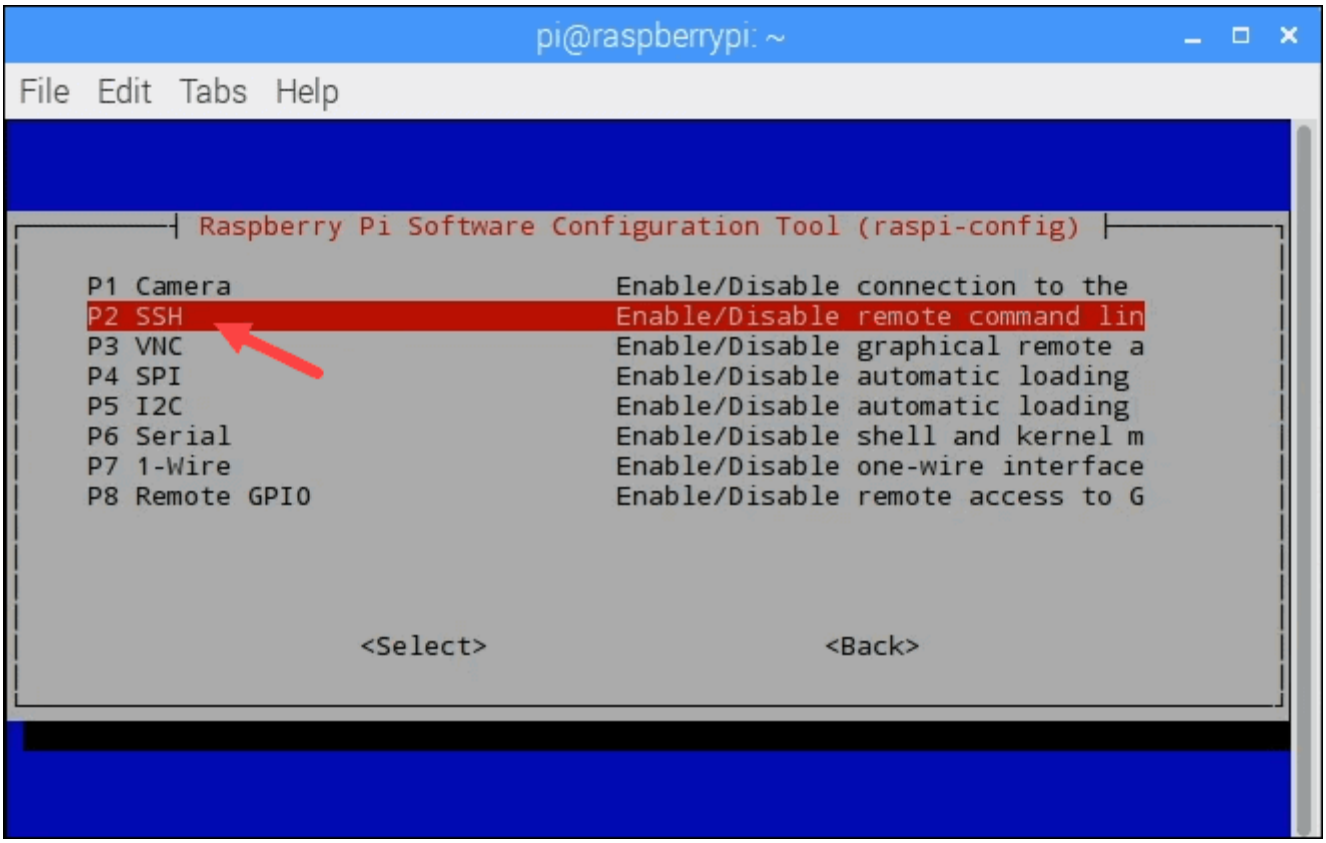



*Figure 2.9*

- Select <Yes> on the "Would you like the SSH server to be enabled?" prompt.
- Hit Enter on the "The SSH server is enabled" confirmation box.
- Navigate down and select Finish to close the raspi-config.

You can close the terminal window. Your device is now ready to accept SSH connections. [15]

Step 2: Install and Enable VNC

Working directly on the Raspberry Pi is not always convenient. Perhaps you'd like to work on it remotely from another device.

VNC is a graphical desktop sharing system that allows you to remotely control one computer's desktop interface (running VNC Server) from another computer or mobile device (running VNC Viewer). VNC Viewer sends keyboard and mouse or touch events to VNC Server and receives screen updates in return.

The Raspberry Pi's desktop will appear inside a window on your computer or mobile device. You'll be able to control it as if it were the Raspberry Pi itself.

RealVNC's NC Connect is included with the Raspberry Pi OS. It includes both VNC Server, which allows you to remotely control your Raspberry Pi, and VNC Viewer, which allows you to control desktop computers remotely from your Raspberry Pi if desired.

Before you can use VNC Server, you must first enable it; instructions for doing so are provided below. VNC Server, by default, provides remote access to the graphical desktop running on your Raspberry Pi, as if you were sitting in front of it. [16]

However, if your Raspberry Pi is headless or does not have a graphical desktop, you can use VNC Server to gain graphical remote access to it. For more information, see the section below titled Creating a Virtual Desktop.

- Installing VNC

VNC is already installed on the full Raspberry Pi OS image and can be installed via Recommended Software from the Preferences menu on other versions.

If you are not using a desktop, you can install it from the command line as follows:

```
sudo apt update
sudo apt install realvnc-vnc-server realvnc-vnc-viewer
```

Enabling VNC Server
You can do this graphically or at the command line.

Enabling VNC Server graphically

- On your Raspberry Pi, boot into the graphical desktop.
- Select Menu > Preferences > Raspberry Pi Configuration > Interfaces.
- Ensure VNC is Enabled.

- Enabling VNC Server at the command line

You can enable VNC Server at the command line using raspi-config:

```
sudo raspi-config
```

Now, enable VNC Server by doing the following:

- Navigate to Interfacing Options.

- Scroll down and select VNC > Yes.

Step 3: Enable Camera Option

A 15-way ribbon cable connects the camera board to the Raspberry Pi. There are only two connections to make: the ribbon cable must be connected to both the camera PCB and the Raspberry Pi. The cable must be inserted correctly, or the camera will not function. The blue backing on the cable should face away from the camera PCB, and the blue backing on the cable should face towards the Ethernet connection (or where the Ethernet connector would be if you're using a model A).

Despite the fact that the connectors on the PCB and the Pi are different, they function in the same way. Pull up the tabs on each end of the connector on the Raspberry Pi. It should be easy to slide up and pivot around slightly. Insert the ribbon cable completely into the slot, making sure it is straight, and then gently press down the tabs to clip it into place. You must also pull the camera PCB connector tabs away from the board, gently insert the cable, and then push the tabs back. The PCB connector is a little more difficult to use than the one on the Pi itself. [17]

- Setting up the camera software

Run the command line instructions below to download and install the most recent kernel, GPU firmware, and applications. For this to work properly, you'll need an internet connection.

```
sudo apt update
sudo apt full-upgrade
```

Now you need to enable camera support using the raspi-config program you will have used when you first set up your Raspberry Pi.

```
sudo raspi-config
```

Use the cursor keys to select and open Interfacing Options, and then select Camera and follow the prompt to enable the camera.

Upon exiting raspi-config, it will ask to reboot. The enable option will ensure that on reboot the correct GPU firmware will be running with the camera driver and tuning, and the GPU memory split is sufficient to allow the camera to acquire enough memory to run correctly.

To test that the system is installed and working, try the following command:

```
raspistill -v -o test.jpg
```

The display should show a five-second camera preview before taking a picture and saving it to the file test.jpg while displaying various informational messages.

## 2.3 Remote Session

At first, we will need PuTTY in Windows to Connect to Raspberry Pi we will talk about installation of PuTTY at this chapter in *"2.3 Programs Installation – 2.3.4 PuTTY"*

Windows users can SSH into Raspberry Pi using PuTTY.

Start the tool and enter the IP address of your device. Make sure SSH is selected, and the port set to 22.

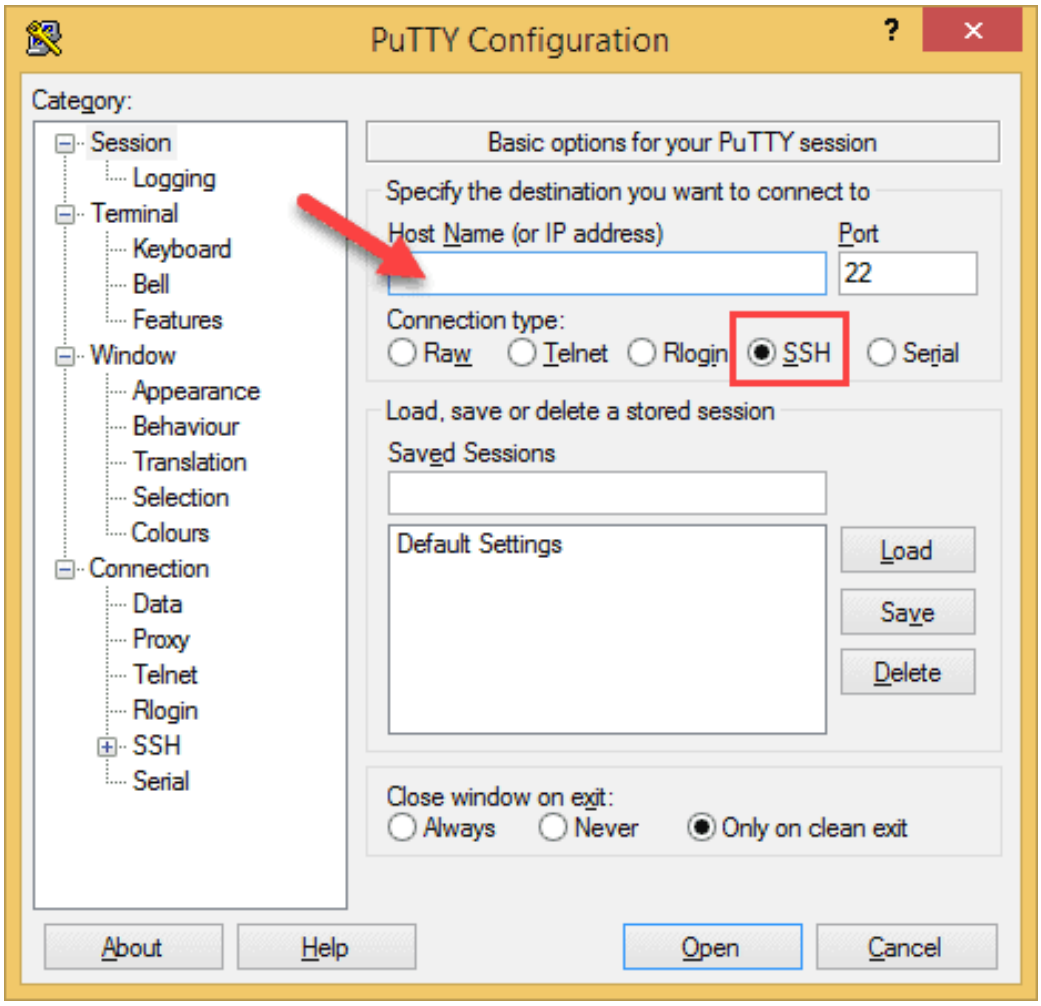



*Figure 2.10*

Click Open to start a new session. Enter your Raspberry Pi's account username and password.

Securing Raspberry Pi SSH

SSH was disabled by default on Raspbian in November 2016. This action was taken to prevent Raspberry Pi devices from becoming part of an IoT botnet.

We recommend that you change the default account password now that you've decided to use SSH to connect to your Raspberry Pi.

Because all Raspbian installations include a default account and password, hackers can easily log into your device. Change the default password on your Raspberry Pi to make it less vulnerable.

- To do so, run the raspi-config tool from the terminal on your device:

```
sudo raspi-config
```

- Select the Change User Password option.

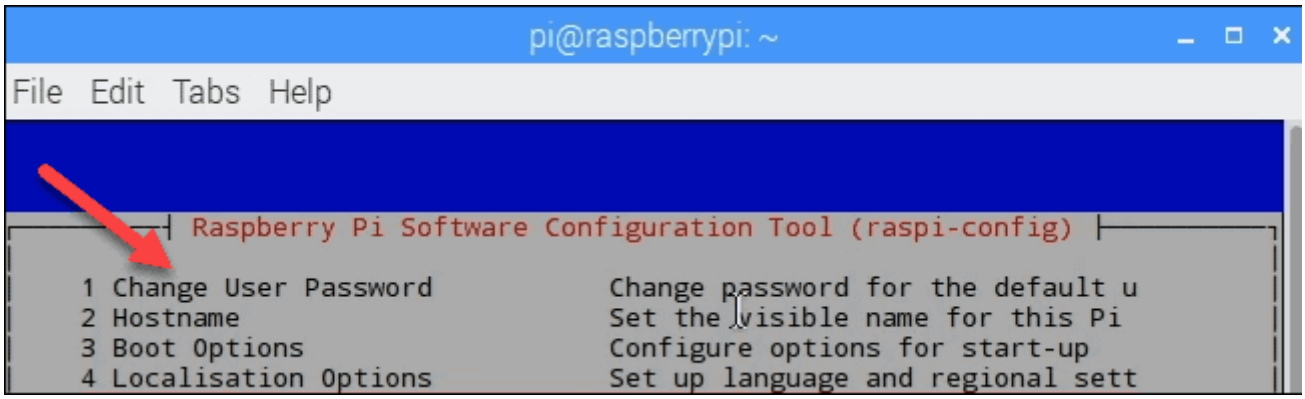



*Figure 2.11*

Follow the instructions to change the password. Your Raspberry Pi is now ready and more secure for SSH access. We recommend you take further steps to improve SSH security.

## 2.4 Remote Development

First, we need to install SublimeText3 on your windows you can check installation of it at this chapter in *"2.3 Programs Installation – 2.3.3 Sublime Text IDE"*

After applying steps at "2.2 Recommended Configurations" you will be able to access files of raspberry pi using SSH and then edit it with SublimeText3

Step 1: Install Package Control

- Open the command palette by pressing
  *Win/Linux: ctrl+shift+p, Mac: cmd+shift+p*

- OR Preferences >> Browse Packages

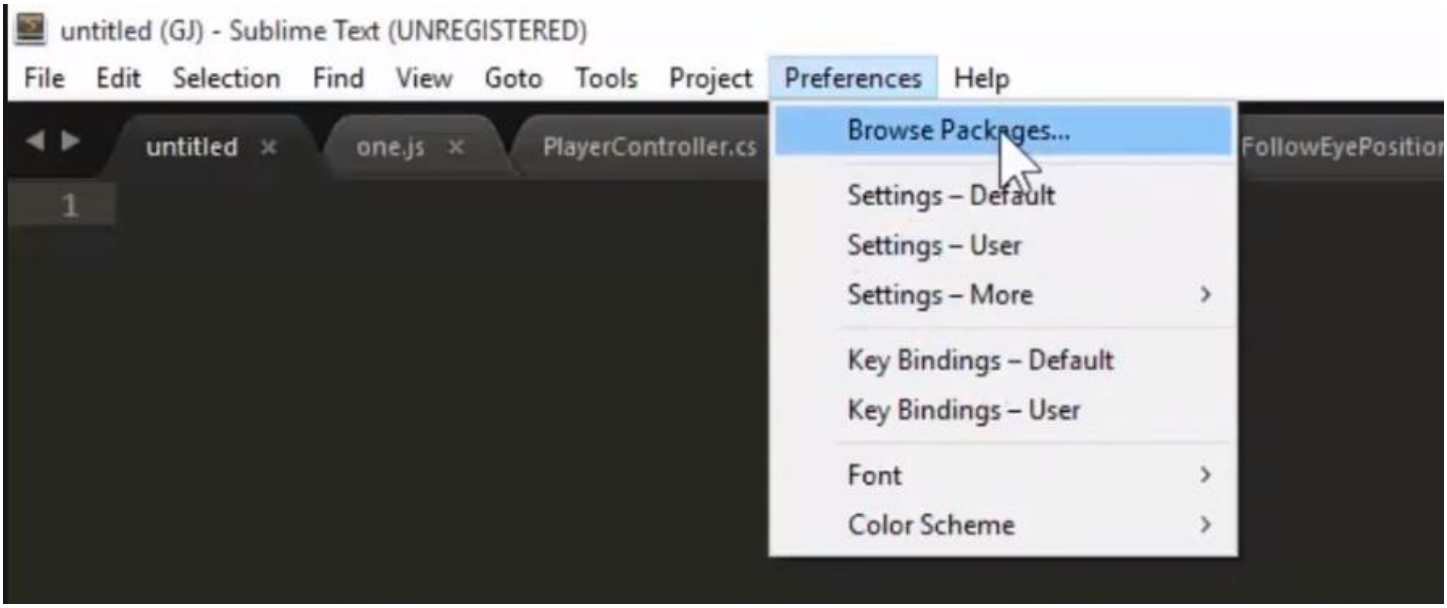



*Figure 2.12*

- Type Install Package Control, press enter
- Restart Sublime Text

Step 2: Install SFTP Package

- Preferences >> Package Control

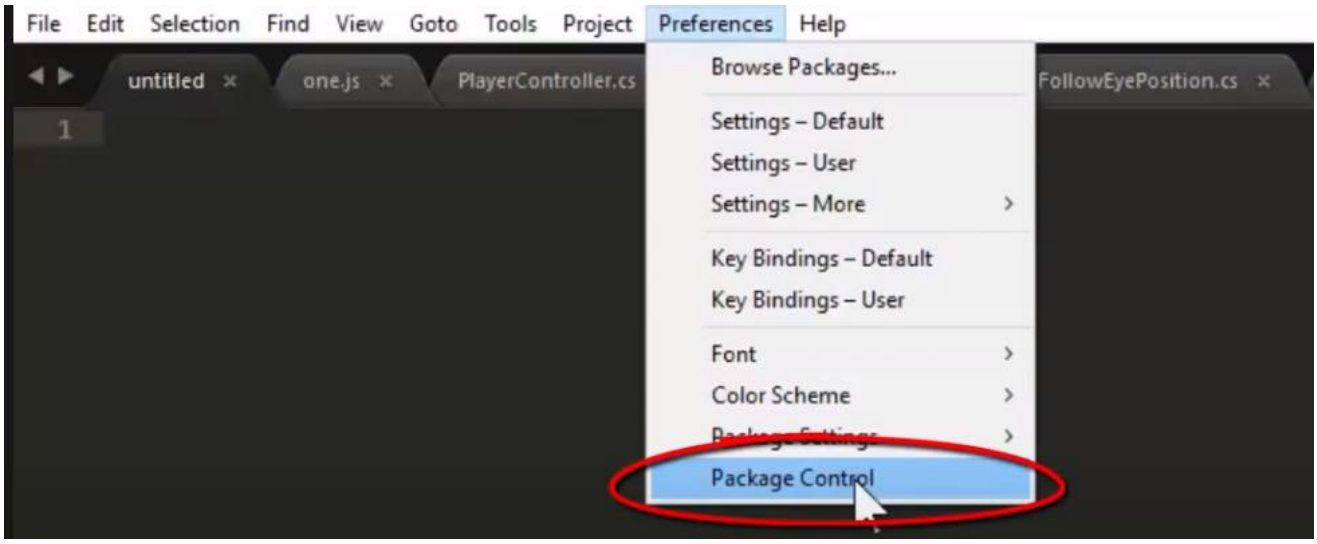



*Figure 2.13*

- Type Install Package

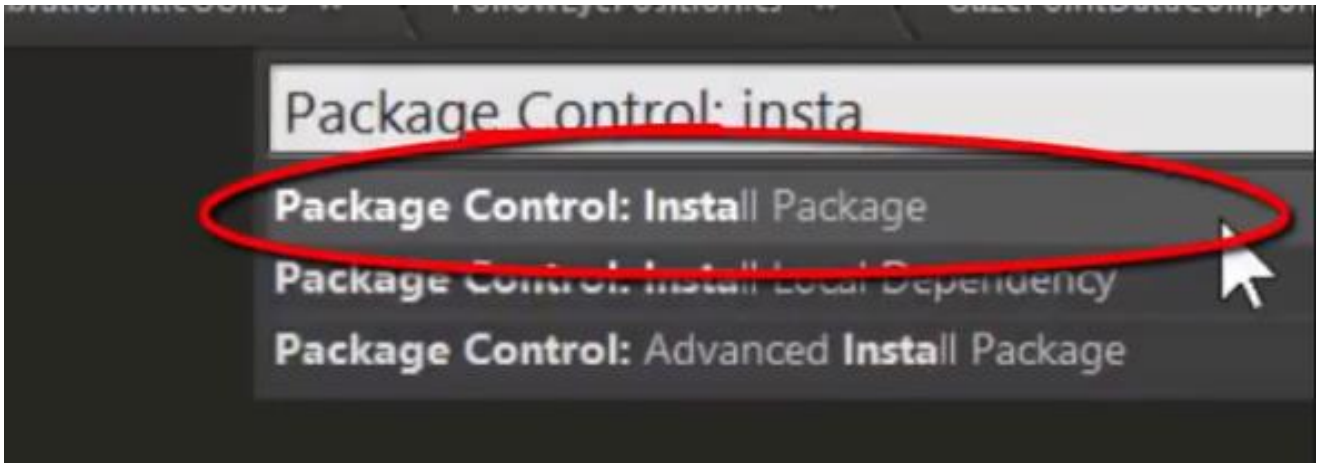



*Figure 2.14*

- Type SFTP

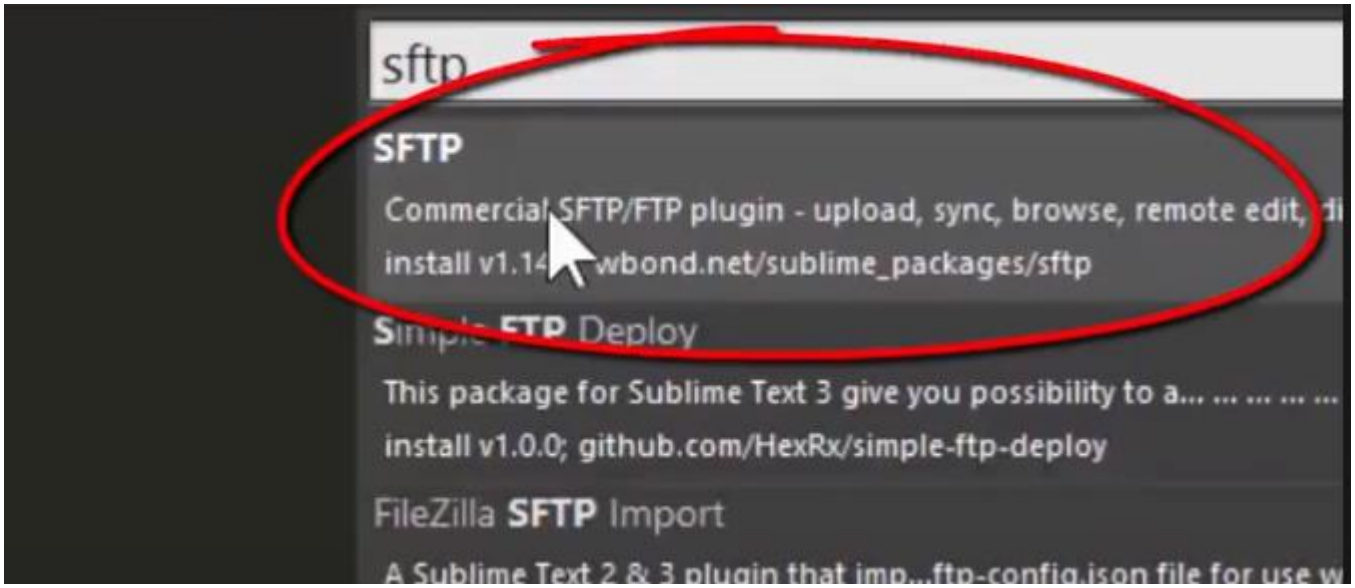



*Figure 2.15*

- Click Left on Folder/File and Choose SFTP/FTP >> Map to Remote

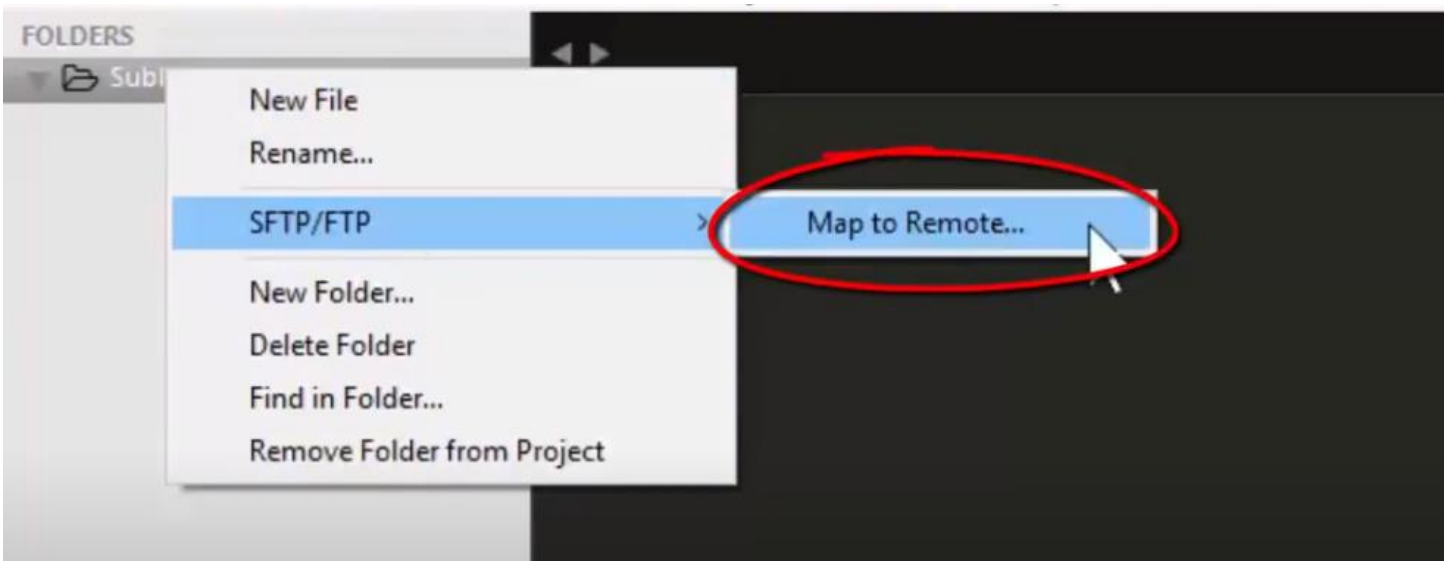



*Figure 2.16*

- sftp-config.json file will created then edit these data

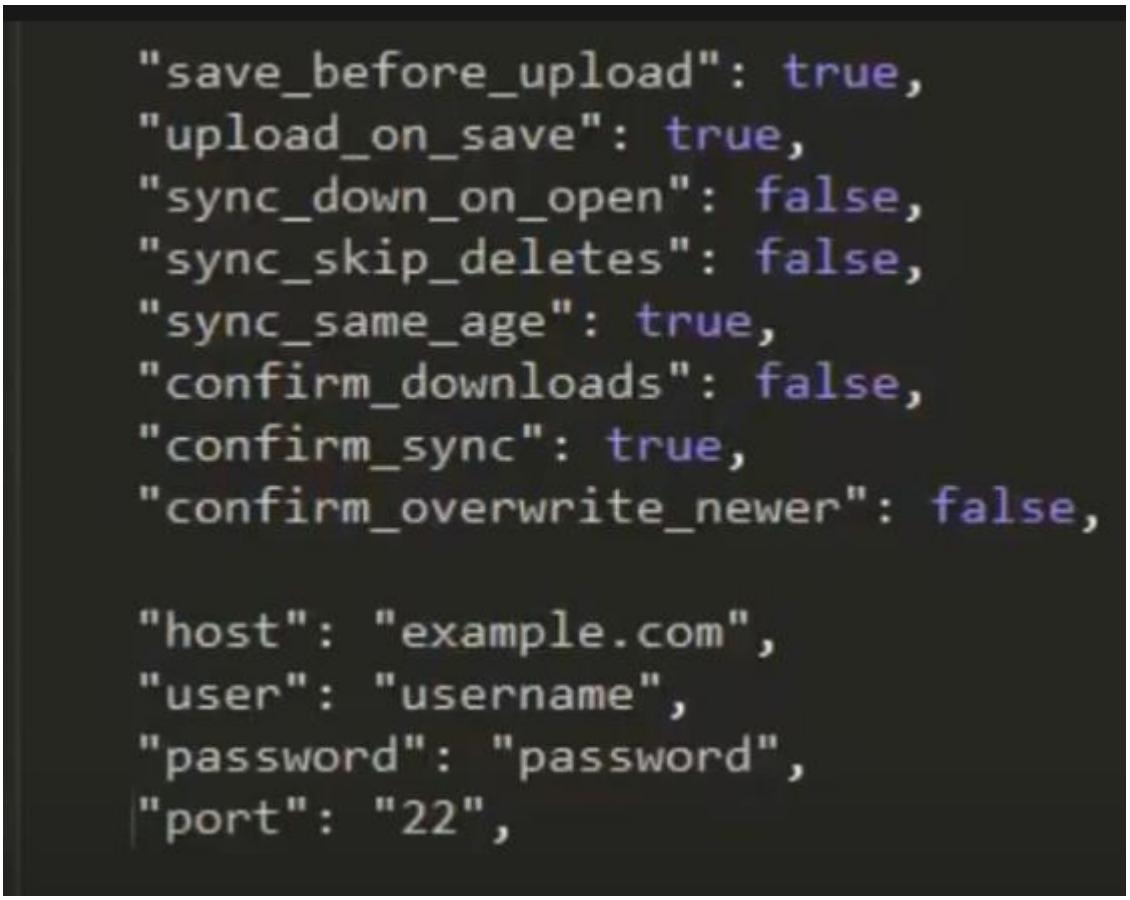



*Figure 2.17*

- After filling data all will be Done!

# 3. Programs Installation

We need to install programs in our windows that will help us to develop, remote, and controls raspberry pi easily

## 3.1 Python IDE

How to Install Python IDE
Below is a step-by-step process on how to download and install Python on Windows:

Step 1: To download and install Python, visit the official website of Python https://www.python.org/downloads/ and choose your version. We have chosen Python version 3.6.3 [18]

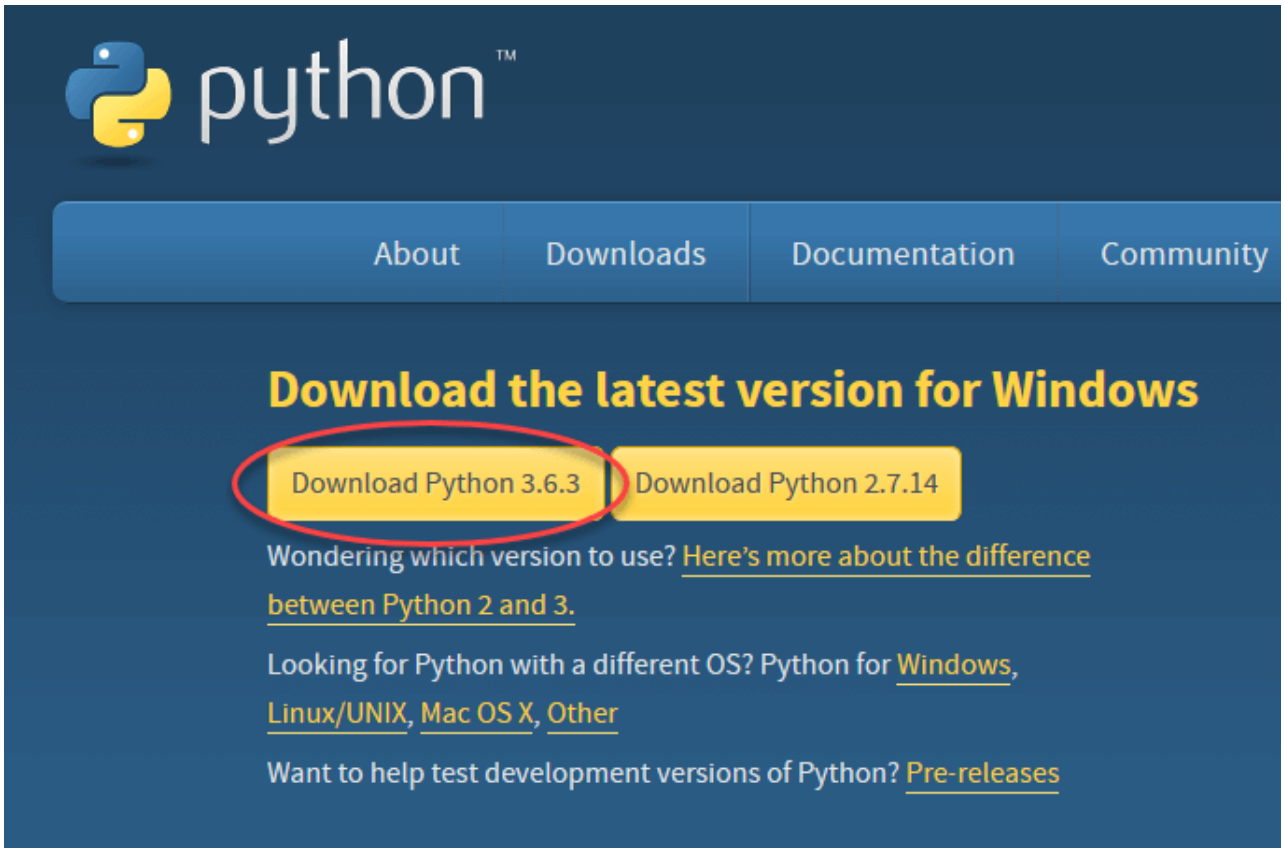



*Figure 2.18*

Step 2: Once the download is completed, run the .exe file to install Python. Now click on Install Now.

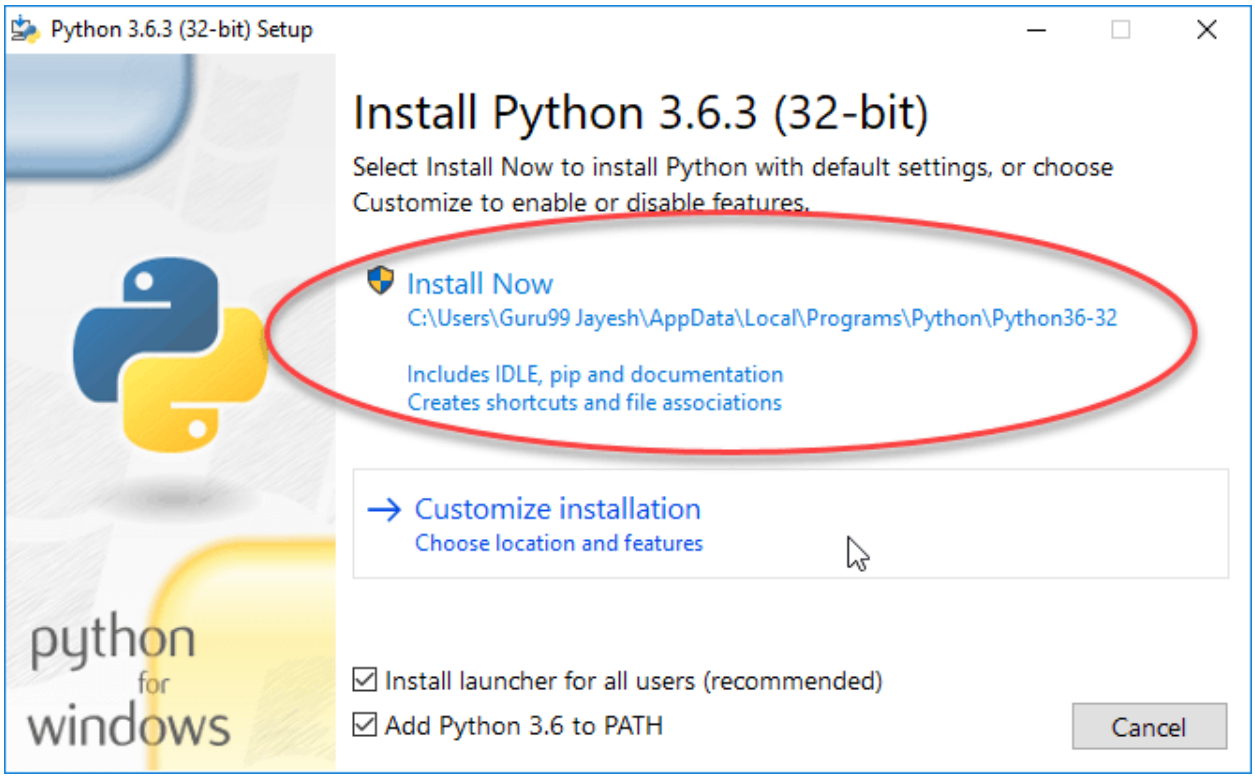



*Figure 2.19*

Step 3: You can see Python installing at this point.

Step 4: When it finishes, you can see a screen that says the Setup was successful. Now click on "Close".

## 3.2 PyCharm IDE

PyCharm is a cross-platform IDE that provides a consistent experience on Windows, macOS, and Linux.

PyCharm comes in three editions: Professional, Community, and Edu. The Community and Edu editions are free and open-source projects, but they have fewer features. PyCharm Edu offers courses and assists you in learning Python programming. The Professional edition is a paid version that includes an impressive set of tools and features. See the editions comparison matrix for more information.

# System requirements:

| Requirement | Minimum | Recommended |
|---|---|---|
| RAM | 4 GB of free RAM | 8 GB of total system RAM |
| CPU | Any modern CPU | Multi-core CPU. PyCharm supports multithreading for different operations and processes making it faster the more CPU cores it can use. |
| Disk space | 2.5 GB and another 1 GB for caches | SSD drive with at least 5 GB of free space |
| Monitor resolution | 1024x768 | 1920×1080 |

| Requirement | Minimum | Recommended |
| --- | --- | --- |
| Operating system | Officially released 64-bit versions of the following:<br>• Microsoft Windows 8 or later<br>• macOS 10.13 or later<br>• Any Linux distribution that supports Gnome, KDE, or Unity DE. PyCharm is not available for some Linux distributions, such as RHEL6 or CentOS6, that do not include GLIBC 2.14 or later.<br><br>Pre-release versions are not supported. | Latest 64-bit version of Windows, macOS, or Linux (for example, Debian, Ubuntu, or RHEL) |

How to Install PyCharm Here is a step-by-step process on how to download and install PyCharm IDE on Windows:

Step 1: To download PyCharm visit the following link: https://www.jetbrains.com/pycharm/download/

Step 2: Once the download is complete, run the exe for install PyCharm. The setup wizard should have started. Click “Next”.

Step 3: On the next screen, Change the installation path if required. Click “Next”.

Step 4: On the next screen, you can create a desktop shortcut if you want and click on “Next”.

Step 5: Choose the start menu folder. Keep selected JetBrains and click on “Install”.

Step 6: Wait for the installation to finish.

Step 7: Once installation finished, you should receive a message screen that PyCharm is installed. If you want to go ahead and run it, click the “Run PyCharm Community Edition” box first and click “Finish”. [19]

Step 8: After you click on "Finish," the Following screen will appear.

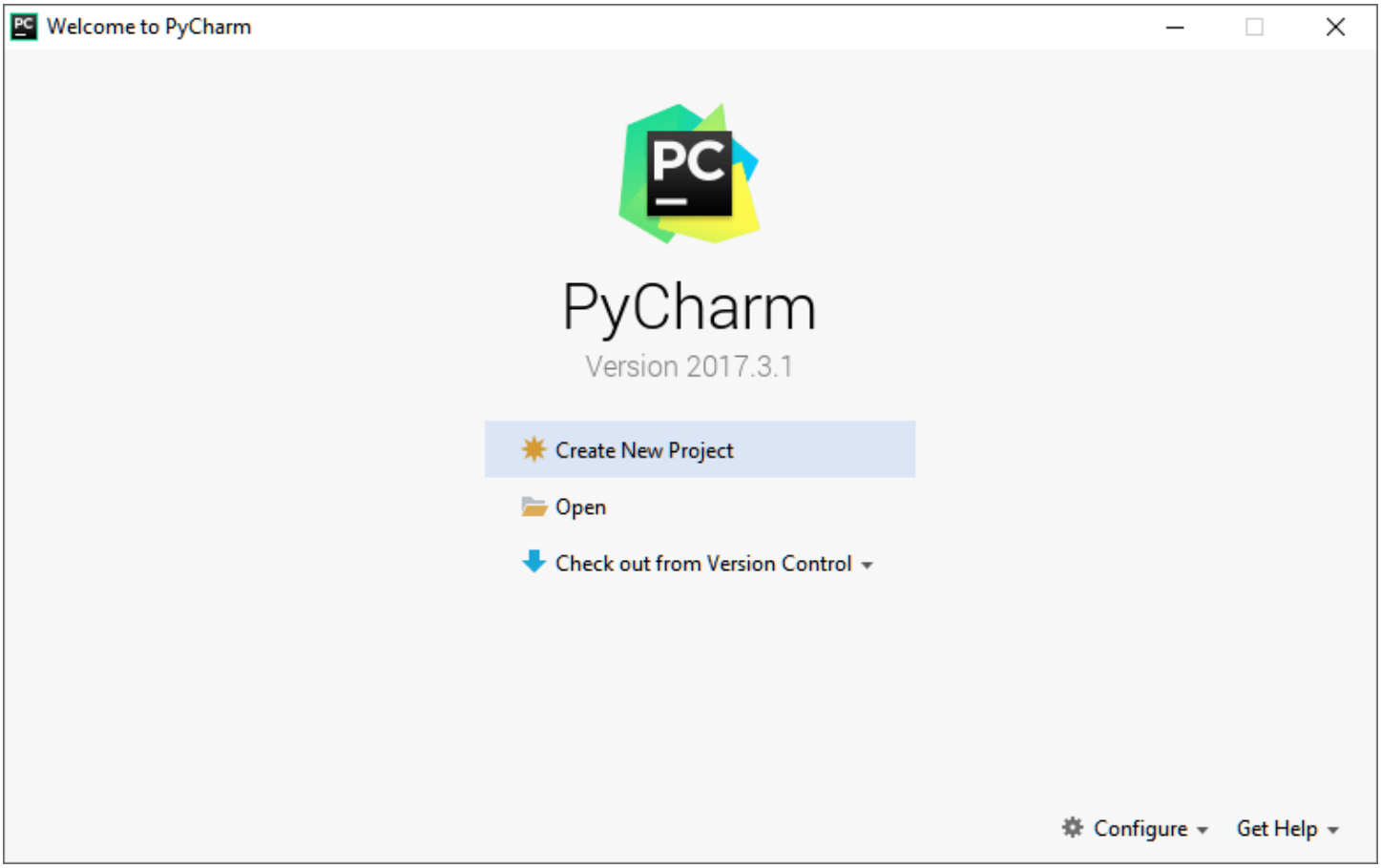



*Figure 2.20*

## 3.3 Sublime Text IDE

Written by a Google engineer sublime text is a cross-platform IDE developed in C++ and Python. It has basic built-in support for Python. Sublime text is fast, and you can customize this editor as per your need to create a full-fledged

Python development environment. You can install packages such as debugging, auto-completion, code linting, etc. There are also various packages for scientific development, Django, Flask, and so on.

Features of Sublime Text 3

- Go to anything for opening files with few clicks and can navigate to words or symbols.
- Python-based plugin API.
- Syntax highlighting and allows simultaneous editing (multiple selections)
- Command Palette implementation that accepts text input from users.
- High performance, block selection and simultaneous editing (multiple selections).

Downloading and Installation:
Sublime Text 3 can be downloaded from its official site sublimetext.com. To install sublime text 3 on Windows, go through the following steps:

Step 1: Open the downloaded .exe file from the downloads folder and begin with the installation process.

Step 2: Select the desired location and click on the next button to start installation

Step 3: If you want Sublime Text 3 to appear in your right-click menu, then mark the checkbox and click on the Next button.

Step 4: Press the install button.

Step 5: Finish with the installation process.

## 3.4 PuTTY

Download PuTTY:

Step 1: Open a web browser to visit the official PuTTY download site at www.chiark.greenend.org.uk/~sgtatham/putty/latest.html.

Step 2: Look for the Package files, MSI (Window Installer) for the 32-bit or 64-bit versions of the latest PuTTY release to download.

The 32-bit version of PuTTY [version] installer.msi will run on all 32-bit and 64-bit versions of PC and Windows editions that PuTTY supports.
The 64-bit version of PuTTY-64bit [version] installer.msi will provide better performance if your PC processor and Windows edition are both 64-bit (typically true for relatively new Windows PC).
If you are not sure which version to install, then select the 32-bit version. [21]

Install PuTTY:
Step 1: Open File Explorer (Windows 10) or Windows Explorer (Windows prior to 10). The Windows logo key + e shortcut works for Windows 10 and earlier versions. Navigate to the Downloads directory (or wherever you saved the installer) and double-click on the installer file name.

Step 2: The installer starts, showing the Welcome to the PuTTY Setup Wizard screen. Click Next.

Step 3: The installer next asks for the destination folder. Normally, it is fine to accept the default. Otherwise, enter your preferred destination folder. Click Next.

Step 4: The installer next asks you which PuTTY features to install.

Step 5: It is convenient to have a shortcut on your desktop. Change the Add shortcut to PuTTY on the Desktop from unavailable (red x) to Will be installed on a local hard drive. Click Install. If you are prompted to run the PuTTY Installer, then allow it to run.

Step 6: The installer will inform you when the install is complete. Click Finish.

Starting PuTTY SSH:
If you created the PuTTY desktop shortcut during the installation, then you have the PuTTY icon on your desktop. Double click the PuTTY icon to start the PuTTY SSH client.

If you did not create the desktop shortcut or cannot find it, then select the Windows Start button located on the bottom left of your desktop. Scroll to menu item PuTTY or PuTTY (64-bit), then select PuTTY.

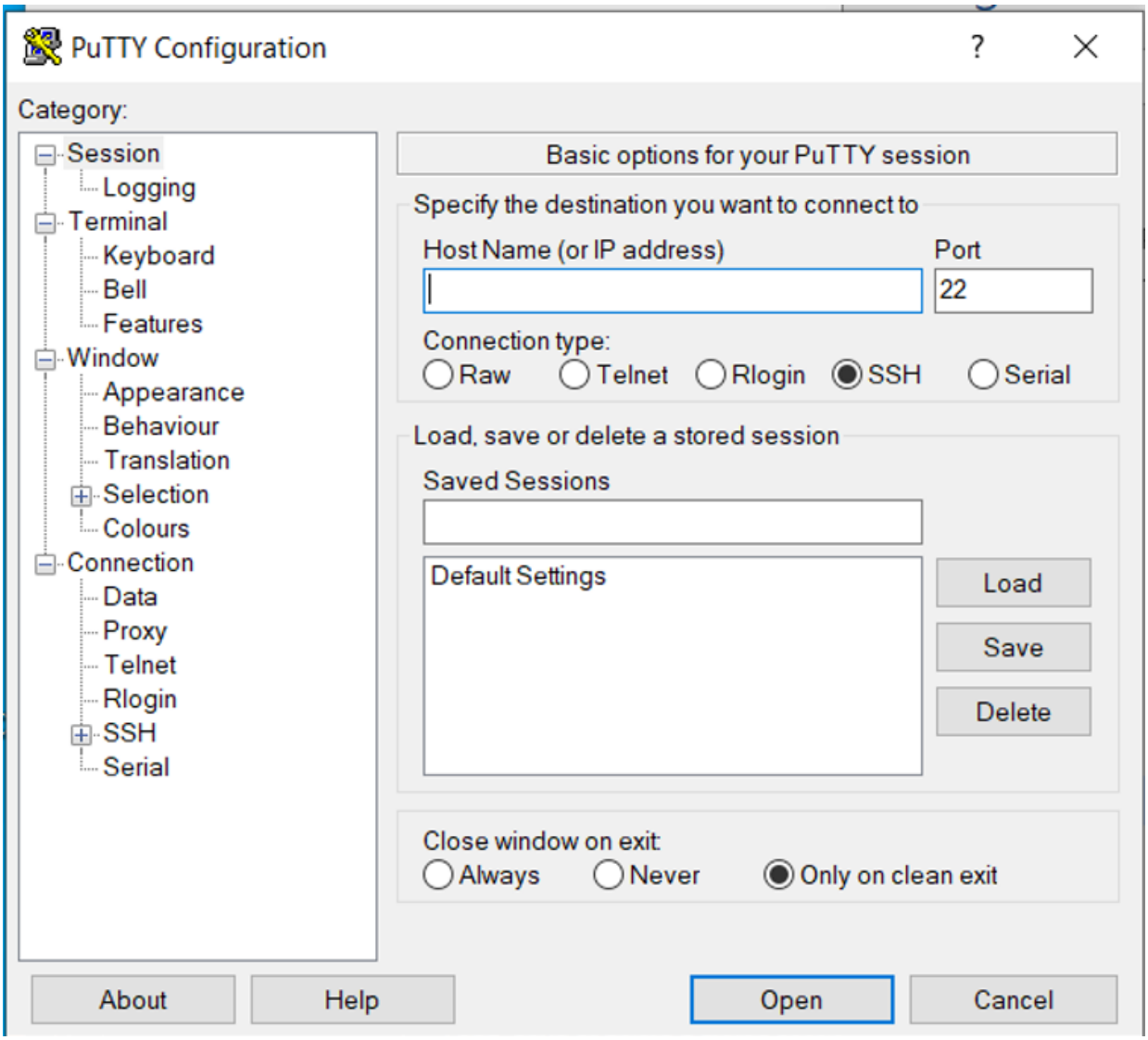



*Figure 2.21*

The PuTTY Configuration window will start. Enter the hostname or IP address of the remote host you want to connect in the “Host Name (or IP address)” field, then click Open.

## 3.5 VNC Viewer

Step 1: Activate your VNC subscription. RealVNC's VNC Server, the software you have to install on the computer that will be controlled, requires a license for use. To get this license, you'll need to create an account. Here are your options:

- A Home license is available for personal use at no cost and provides basic remote control of one computer. To get a free home license, visit https://www.realvnc.com/en/connect/home, and follow the on-screen instructions to create your account.
- If you need to control between 2 and 10 computers, you'll need a Professional or Enterprise account. These licenses cost money, but 30-day trials are available. To get a free trial of RealVNC Professional or Enterprise, visit https://www.realvnc.com/en/connect/trial and follow the on-screen instructions to create your account.

Step 2: Download VNC Server on the computer you want to control. The VNC Server software will need to be installed on any computer you want to operate remotely, while VNC Viewer will be installed on the computer, phone, or tablet you're using to access the server remotely.[2] To download VNC Server:

- Go to https://www.realvnc.com/en/connect/download/vnc on the computer you want to control.
- Click your operating system.
- Click the blue Download VNC Server [version] button.
- Save the installer to your computer.

Step 3: Run the VNC Server installer on the computer you want to control. Double-click the installer file you downloaded, then follow the on-screen

instructions to complete the setup process. During the setup, you'll be prompted to sign in with your RealVNC account—your Home, Professional, or Enterprise license is connected to this account.

- If you're installing the Enterprise version, enable cloud connectivity when prompted during setup.

Step 4: Download VNC Viewer to your local computer, phone, or tablet. VNC Viewer can be installed on a variety of operating systems, including Windows, Android, iOS, and ChromeOS.

- On a computer: Go to link below and select your operating system. Click the blue Download VNC Viewer link to download the installer to your computer.
  https://www.realvnc.com/en/connect/download/viewer
- Android: Open the Play Store app in your app drawer and search for vnc viewer. Tap INSTALL when you find it and follow the on-screen instructions.
- iPhone/iPad: Open the App Store app, tap Search, and search for vnc viewer. Tap GET once you find it and follow the on-screen instructions.

Step 5: Install VNC Viewer and log in. During the installation process, you'll need to sign in with the same account you used to sign in to the VNC Server on the computer you want to control. Once the process is complete, the VNC Viewer app will be ready to use.

- On a computer: Double-click the downloaded installer file and follow the on-screen instructions to install. When asked to sign in, use your RealVNC account information.
- On a phone or tablet: The app is already installed, so just tap the VNC Viewer icon on the home screen or in the app drawer, and then follow the on-screen instructions to log in with your RealVNC account.

Step 6: Launch VNC Server on the computer you want to control. The VNC Server application must be running when you attempt to make a connection.

Step 7: Select the computer to be controlled in VNC Viewer. Once the VNC Server is running on the computer to be controlled, it'll appear as a selectable option in VNC Viewer. A login screen will appear.

Step 8: Log in to the remote computer. This time you won't be using your RealVNC account—instead, use the login information you normally use to log in to the remote computer, such as a local user or network account. Once your password is validated, you will see the desktop of the remote computer in VNC Viewer on your computer, phone, or tablet.

# CHAPTER 3

## 1. Flowchart of Computer Vision

A flow chart, also known as a flow diagram, is a graphical representation of a process or system that details the steps required to produce output. A typical flow chart represents various functions with a set of basic symbols and shows the sequence and interconnection of functions with lines and arrows. Flow charts can be used to document almost any type of business system, from the movement of materials through machinery in a manufacturing operation to the flow of applicant information through a human resources department's hiring process. [22]

Each flow chart focuses on a single process or system. It starts with the input of data or materials into the system and proceeds through all of the procedures required to convert the input into its final output form. Flow chart symbols depict the processes that occur, the actions that are performed in each step, and the relationships between the various steps. [23] Flow charts can be as detailed as needed, ranging from a high-level overview of an entire system to a detailed diagram of one component process within a larger system. The flow chart, in any case, depicts the overall structure of the process or system, traces the flow of information and work through it, and highlights key processing and decision points.

Flow charts are an important tool for process improvement. They assist project teams in identifying the various elements of a process and understanding the interrelationships between the various steps by providing a graphical representation. Flow charts can also be used to collect information and data about a process as a decision-making or performance-evaluation tool. For example, the owner of a small advertising agency who wants to cut the time it takes to create a print ad might be able to use a process flow chart to identify and eliminate unnecessary steps. Despite the fact that flow charts are an old design tool, they are still popular among computer programmers

who work on system analysis and design. Many software programs have been developed in recent years to assist businesspeople in creating flow charts.

Constructing Flow Charts
Flow charts typically utilize specialized symbols. Some of the main symbols that are used to construct flow charts include:

- A round-edged rectangle to represent starting and ending activities, which are sometimes referred to as terminal activities.
- A rectangle to represent an activity or step. Each step or activity within a process is indicated by a single rectangle, which is known as an activity or process symbol.
- A diamond to signify a decision point. The question to be answered or decision to be made is written inside the diamond, which is known as a decision symbol. The answer determines the path that will be taken as a next step.
- Flow lines show the progression or transition from one step to another.

The following are the main steps in creating a flow chart:
- Define the process and identify the scope of the flow diagram.
- Identify project team members who will be involved in the process flow diagram construction.
- Define the various steps involved in the process and their interrelationships (all team members should contribute to developing and agreeing on the various steps for the process.
- Finalize the diagram, involving other stakeholders as needed and making any necessary changes.
- Use the flow diagram and continuously update it as needed. [24]

And at the following figures we will show our project flowchart which created by https://dashboard.visme.co/

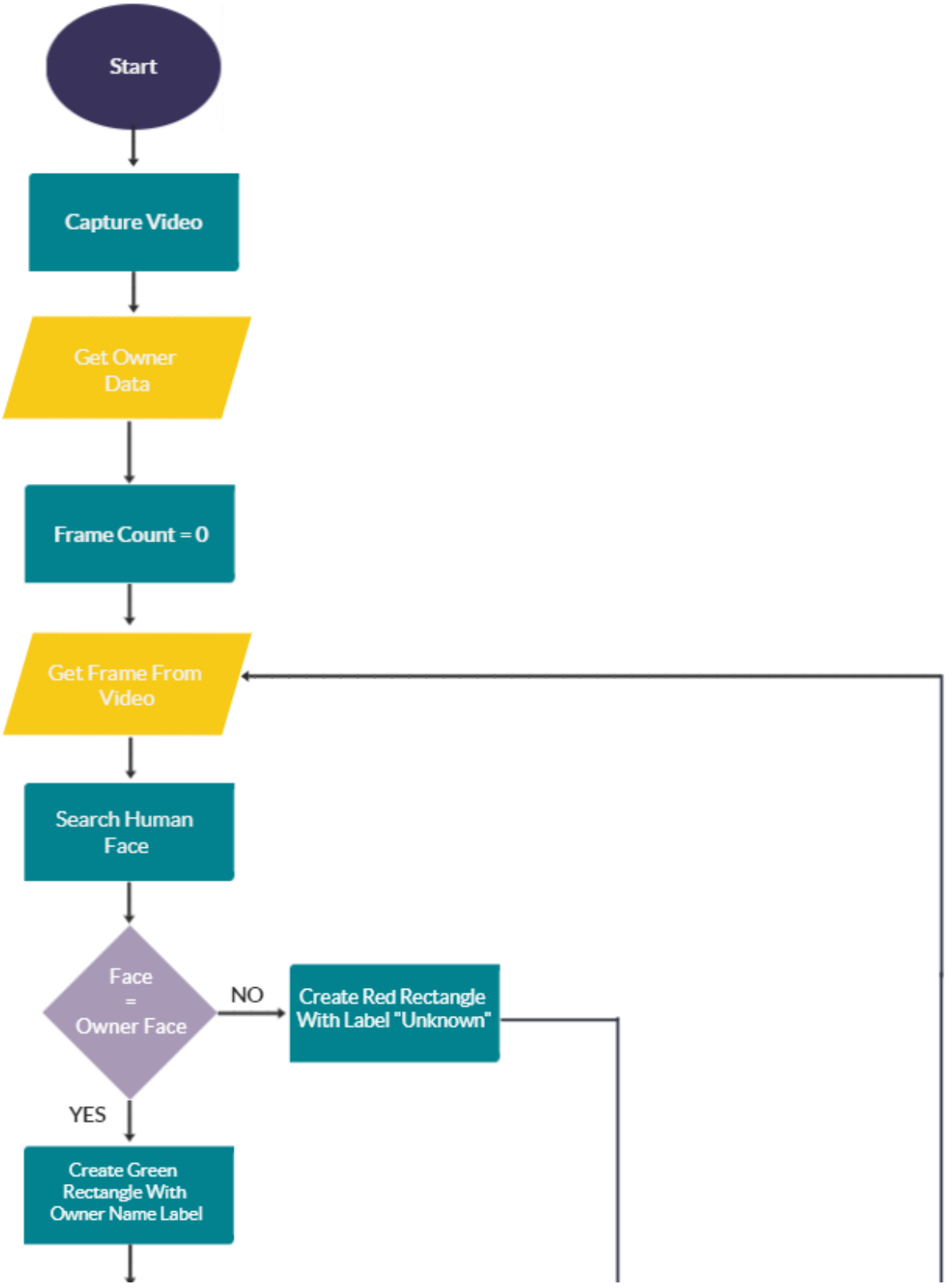

Start
Capture Video
Get Owner Data
Frame Count = 0
Get Frame From Video
Search Human Face
Face = Owner Face
NO
Create Red Rectangle With Label "Unknown"
YES
Create Green Rectangle With Owner Name Label

*Figure 3.1*

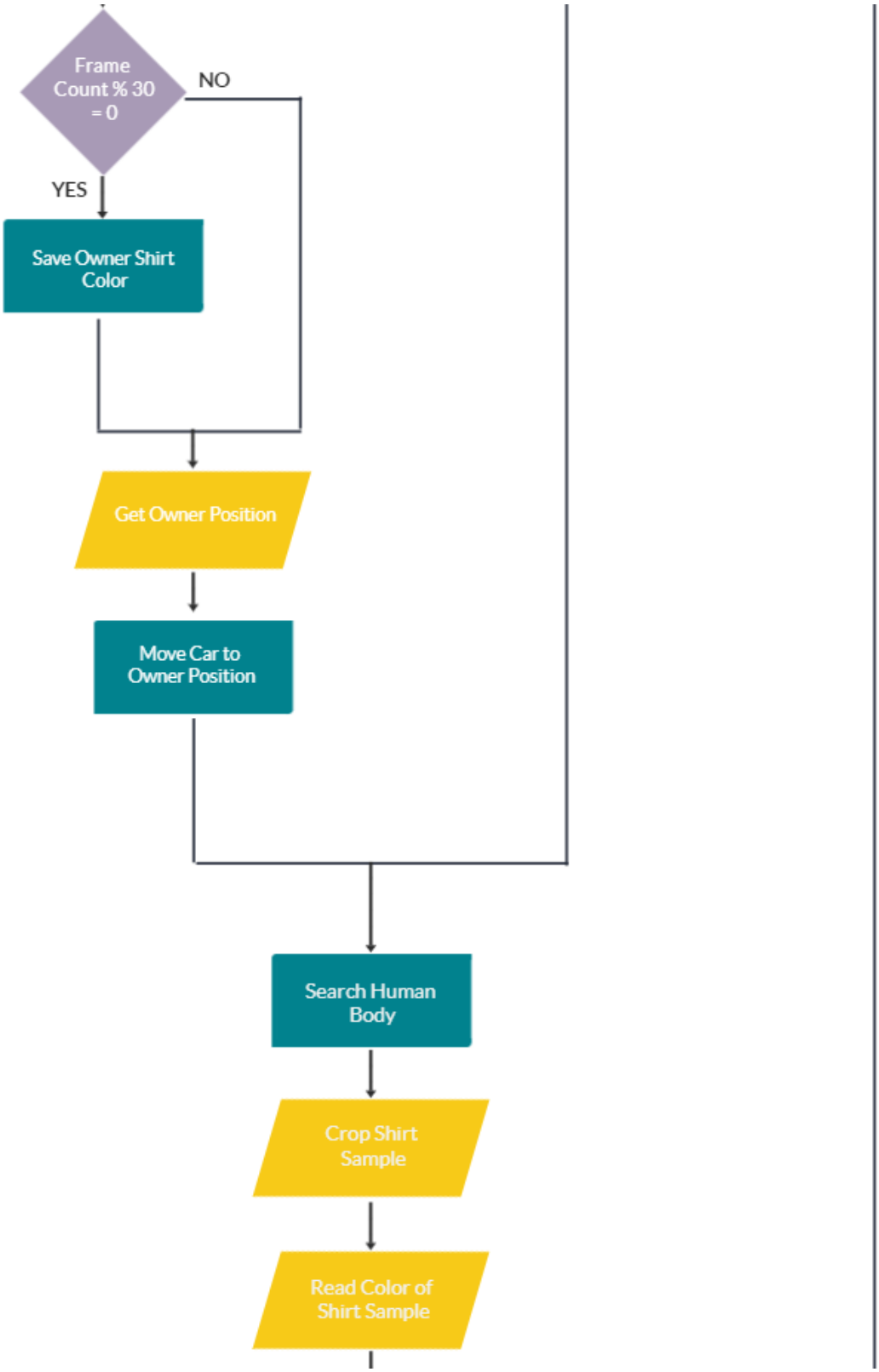

Frame Count % 30 = 0
NO
YES
Save Owner Shirt Color
Get Owner Position
Move Car to Owner Position
Search Human Body
Crop Shirt Sample
Read Color of Shirt Sample

*Figure 3.2*

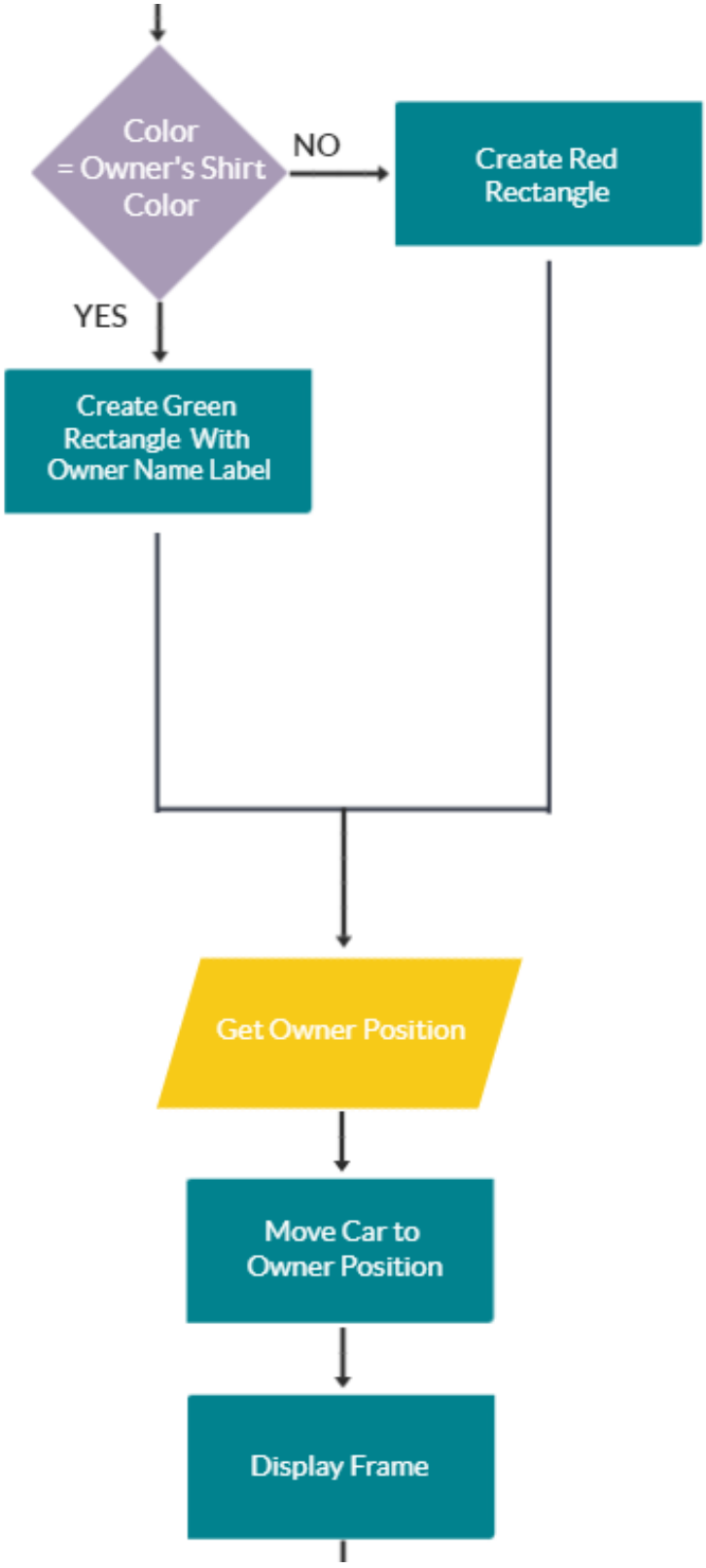

Color
= Owner's Shirt
Color
NO
Create Red
Rectangle
YES
Create Green
Rectangle With
Owner Name Label
Get Owner Position
Move Car to
Owner Position
Display Frame

*Figure 3.3*

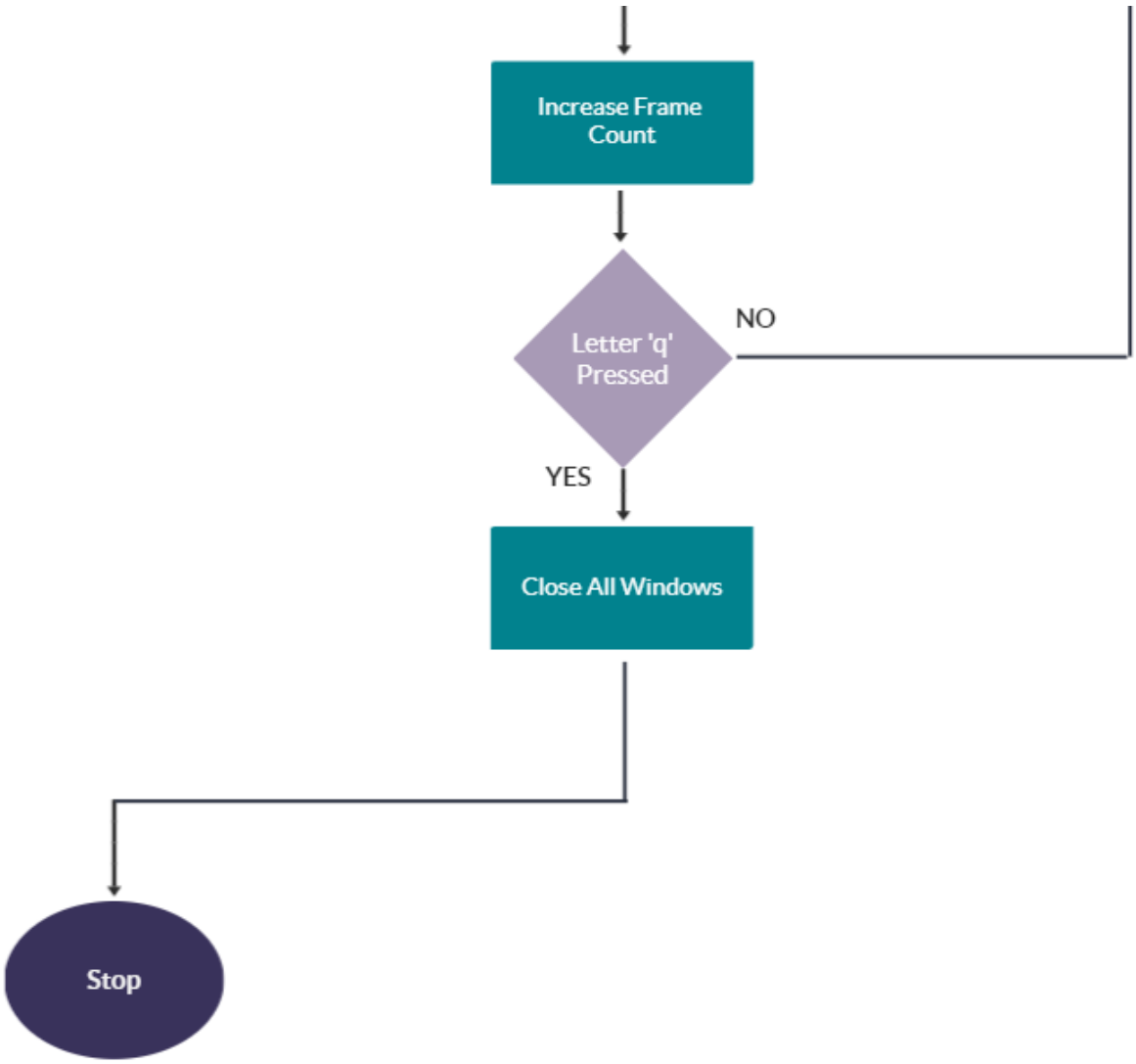

Increase Frame Count
Letter 'q' Pressed
NO
YES
Close All Windows
Stop

*Figure 3.4*

## 2. Main Diagram

As shown in our flowchart at the start of the program we need to setup our camera and capture a stream of video, so this is how we did it.

Step 1: We import necessary libraries – OpenCV

```
    import cv2 as cv
```

Step 2: Capture a frame from the camera by

```
def captureFromCamera():
    # Capture from camera or anything can get view
    return cv.VideoCapture(0, cv.CAP_DSHOW)
```

Step 3: Capture a stream of video from existence place

```
def captureFromVideo(directory: str):
    # Capture from existing sample directory
    return cv.VideoCapture(directory)
```

We need to do following processes on captured frame all of them in a Frame - Class:

- Display:

```
@staticmethod
def display(title: str, frame: cv.VideoCapture):
        # Display image
        cv.imshow(title,  # Window title
                  frame)  # frame
```

- Scale:

```
@staticmethod
def scale(frame, xy, x, y):
# Resize frame of video to 1/4 size for faster face reco
gnition processing
        return cv.resize(frame, xy, fx=x, fy=y)
```

- Crop:

```
@staticmethod
def crop(frame, x, y, width, height):
       return frame[int(y): int(y + height), int(x): int
(x + width)]
```

- Get Dimensions:

```
@staticmethod
    def getDimensions(captured_video: cv.VideoCapture):
        # Return Width, Height of frame
        return captured_video.get(cv.CAP_PROP_FRAME_WIDT
H), captured_video.get(cv.CAP_PROP_FRAME_HEIGHT)
```

- Close All Display Windows

```
@staticmethod
    def closeAllWindows(captured_video: cv.VideoCapture)
:
        # Closing video file or capturing device
        captured_video.release()
        cv.destroyAllWindows()
```

Step 4: After all of that we will use pervious methods to reference parameter

```
def main():
    # Initialize needed variables
    captured_video = captureFromCamera()
    camera_dimensions = Frame.getDimensions(captured_vid
eo)
```

Step 5: for a future purpose we need also to count processed frames or the frames that has been read, so we initialize a new variable:

```
# Count Frames
frame_counter = 0
```

Step 6:  also, Owners' data required to trace them but we before that we must define Owner as a new class type to make our computer understands what this mean but we will talk about this in details in a following parts:

```
owners_list = [
```

```
     Owner("Owner_Name_1", "owner_samples_directory_path
_1")
     Owner("Owner_Name_2", "owner_samples_directory_path
_2")
     ...
    Owner("Owner_Name_N", "owner_samples_directory_path_
N")
     ]
```

Step 7: We declare while loop to keep getting frames from camera and process every single frame so first thing after loop we get our frames

```
while True:
        # Grab a single frame from camera or video
        is_frame_exist, original_frame = captured_video.
read()
```

Step 8: but if we did a processes like scaling, coloring, cropping it will affect on other processes that's why we need to take a copy for processes to keep our frame pure if we needed it later

```
 processing_frame = copy(original_frame)
```

but why we need to copy a value?
The answer is because python working by something call pass by reference there is other technique like passing by value but what's the difference? And what passing by reference in first place?

Developers transitioning from other languages such as C++ and Java to Python are frequently perplexed by the Python process of passing arguments. The fundamental cause of the confusion is the object-centric data model and its treatment of assignment. In this article, we will look at the concept of passing arguments in Python and try to understand it through examples.

Is the Python Argument Passing model a “Pass by Value” or “Pass by Reference”?
You might want to punch something after reading ahead, so brace yourself. Python’s argument passing model is neither “Pass by Value” nor “Pass by Reference” but it is “Pass by Object Reference”.

The paradigms of “Pass by value”, “Pass by Reference” and “Pass by object Reference” can be understood by exploring the below example functions.
Look into the two functions defined below:

```
def set_list(list):
    list = ["A", "B", "C"]
    return list

def add(list):
    list.append("D")
    return list

my_list = ["E"]

print(set_list(my_list))

print(add(my_list))
```

Output:

```
['A', 'B', 'C']
['E', 'D']
```

Now, let's explore the above code,

The variable is not the object passed:
What if I told you... Tulsi Das did not write The Ramayana, but a man named Tulsi Das did write The Ramayana. Is that a reasonable distinction? No way, right?! But Python says it does, and it makes an important distinction. So, in Python and its PKDs, there is a significant distinction between the thing and the label we use to refer to it. "A man named Tulsi Das" is simply a man, and "Tulsi Das" is a name given to that man.
Consider making a list.

```
a = ["X", "Y"]
```

Here "a" is a variable that points to a list containing the element "X" and "Y". But "a" itself is not the list. Consider "a" to be a bucket that contains the object "X" and "Y".

In pass by reference the variable( the bucket) is passed into the function directly. The variable acts a Package that comes with its contents(the objects).

In the above code image, both "list" and "my list" are the same container variable and therefore refer to the exact same object in the memory. Any operation performed by the function on the variable, or the object will be directly reflected to the function caller. For instance, the function could completely change the variable's content, and point it at a completely different object:

Also, the function can reassign the contents of the variable with the same effect as below:

To summarize, in pass by reference the function and the caller use the same variable and object.

Pass by Value:
In pass by value the function is provided with a copy of the argument object passed to it by the caller. That means the original object stays intact and all changes made are to a copy of the same and stored at different memory locations.

The same is true for any operation performed by the function on the variable or the object

To summarize the copies of the variables and the objects in the context of the caller of the function are completely isolated.

Pass Object by Reference:
As python is different in this context, the functions in python receive the reference of the same object in the memory as referred by the caller. However, the function does not receive the variable(the bucket) that the caller is storing the same object in; like in pass-by-value, the function provides its own bucket and creates an entirely new variable for itself.

The same object in the memory is referred by both the caller and the function, so when the append function adds an extra element to the list, the caller object gets updated too. They have different names but are the same thing. Both the variables contain the same object. This is the meaning behind pass by object reference. The function and caller use the same object in memory but get them through different variables. Any changes made to the function variable(bucket) won't change the nature of the caller variable (bucket), only the content gets updated.

Let's back to the code now after copping noting left except, we need to detect our objects like face and body, but we discuss this in detail at this chapter – Object Detection Diagram but let's see how function works.

Step 9:  First, we need to get the positions of faces and bodies

```
# Detect objects
owners_faces_position = od.detectHumanFace(
            (original_frame, processing_frame, is_this_f
rame_processed, frame_counter),
            owners_list,
            default_border_color,
            camera_dimensions)
        owners_bodies_position = od.detectHumanBody(
            (original_frame, processing_frame),
            owners_list,
            default_border_color,
            camera_dimensions)
```

Second, we pass this value to structured function to trace our owner

```
# Trace owner if his face or body detected
traceOwner((owners_faces_position, owners_bodies_positio
n), owners_list[0].getName())
```

but how this function does this?

Step 10: To trace owners, we need couple of things:

- Owner positions attached with his name
- Targeted Owner: which car can't go to all of owners at a time, so the targeted owner is required to complete the target

```
def traceOwner(owners_positions: ([str, ObjectPositions]
), targeted_owner_name: str):
```

so according to the owner is face or body position we move the car towards them by sending signals to car and we will talk about this in a sperate chapter but let's dig into the process.

here we check if the targeted owner name equals the name of the passed data

```
if targeted_owner_name.lower().__eq__(owner_name.lower()
):
```

case true we act according to situation of position

```
if owner_position == ObjectPositions.CENTER.value:
    CarActions.moveForward(SpeedRanges.FAST)

elif owner_position == ObjectPositions.TOP_LEFT.value:
    CarActions.rotateRight(SpeedRanges.SLOW)

elif owner_position == ObjectPositions.TOP_RIGHT.value:
    CarActions.rotateLeft(SpeedRanges.SLOW)

elif owner_position == ObjectPositions.BOTTOM_LEFT.value
:
    CarActions.rotateRight(SpeedRanges.SLOW)

elif owner_position == ObjectPositions.BOTTOM_RIGHT.valu
e:
    CarActions.rotateLeft(SpeedRanges.SLOW)
```

```
else:
     CarActions.stop()
```

Step 11: display the frame

```
# Display detected objects
Frame.display("Objects Detectors", processing_frame)
```

Step 12: increase the count of frame

```
# Increase frame counter one
frame_counter += 1
```

Step 13: break the loop if ‘q’ pressed to abort the process

```
# Close window when ord(key) pressed
if cv.waitKey(1) & 0xFF == ord('q'):
     break
```

Step 14: close all display windows once loop break

```
Frame.closeAllWindows(captured_video)
```

## 3. Owner Declaration

To declare any new class, we need to determine what that class means in real world and what defines it by other meaning we what we need to know this is an Owner.

And this is it what we need:

- Name
- Shirt Color
- Face Samples

If we know these things, we could determine what we are looking for so this how we declare this

```
class Owner:
    # Owner Data
    __name = str
    __shirt_color = str
    __face_samples_path = str
    __owner_data_file = File
```

So, every time we call owner, we need to determine his name and some pictures from his face

```
def __init__(self, name: str, face_samples_path: str):
     self.__name = name.lower()
     self.__face_samples_path = "owners_data/owner_sampl
es/" + face_samples_path
     self.__samples_encodings_default_path = Directory.c
reate(self.__face_samples_path, "encodings")
```

```
    self.__saveData()
```

after that we need to encode those samples to make computer understands and recognizes every time camera look to the owner so first we need to import library

```
import face_recognition
```

then we encode the samples

```
face_recognition.face_encodings(face_recognition.load_im
age_file(self.__face_samples_path + sample_file))[0]
```

how can we save and get those encoding? The answer is by writing this data into a binary file that's why we create a customized Document class, but we will talk about it later

to save encodings

```
File("encoding_" + str(sample_counter), self.__samples_e
ncodings_default_path, ".dat").writeBinaryData(
                    sample_encodings)
```

to get encodings

```
def getFaceEncodings(self):
        samples_encodings = []
```

```
        for encoding_file in Directory.listFiles(self.__
samples_encodings_default_path):
            samples_encodings.append(
                File(encoding_file, self.__samples_encod
ings_default_path, ".dat").readBinaryData())
        return samples_encodings
```

also, we need to use the Document class to save and read our Owner shirt color

this how we save it

```
self.__owner_data_file.writeTextData(written_data + colo
r)
```

we read it by looping on the text file till we get the color line

```
# Detect color line in text
is_this_color_line = False
for text in self.__owner_data_file.readTextData():
    if ".Shirt Color:" in text:
        is_this_color_line = True
        continue
    if is_this_color_line:
        return text
```

# 4. Document Declaration

We all know that everything running on the computer store in two places first is the Storage which saved in memory card or hard disk second saved in RAM but the data stored in the last one doesn't save permanently but it saves till the computer release it or accidently power goes off so to keep accessing data all the time and not overloading on our RAM we make a class called Document his mission handling data text or binary and saves it in a directory or a file to specific path.

We need to determine which functions we need precisely:

- Directory
  - Create
  - Remove
  - List Contained Files
  - Count Contained Files

- File
  - Create
  - Remove
  - Write Text
  - Read Text
  - Write Binary
  - Read Binary

## 4.1 Directory

Let's start with the Directory class which handles everything in directory but to make this we need to import OS library

```
import os
```

then we should create our directory but to do that we need to know the name and the path of the Directory

```
@staticmethod
def create(path: str, name: str):
     # Create new dir by (name) if dir not exist
     if not os.path.exists(path + name):
          os.mkdir(path + name)
          print(name + " Directory Created")
     return path + name + '/'
```

remove function also needed in case we need to delete directory and it requires name and path too

```
@staticmethod
def remove(path: str, name: str):
     if os.path.exists(path + name):
     os.remove(path + name)
     print(name + " Directory Removed")
```

count files in the directory

```
    @staticmethod
    def countFiles(path: str) -> int:
        if os.path.exists(path):
            # Files in dir path
            return len(os.listdir(path))
```

```
        return -1
```

list files

```
    @staticmethod
    def listFiles(path: str):
        if os.path.exists(path):
            # Files in dir path
            return os.listdir(path)
        return None
```

## 4.2 File

We need to import OS library to but also, we need Pickle

```
import os
import pickle
```

Every file should have a name and path of it that's why we require it in our constructor

```
class File:
    __name = str
    __path = str

    def __init__(self, name: str, path: str, extension:
str):
        if name.__contains__(extension):
            self.__name = name
```

```
        else:
            self.__name = name + extension
        self.__path = path
        self.__create()
```

and this how we create it

```
def __create(self):
    # Create New text file by (name, path) if name not e
xist
    if not os.path.exists(self.__path + self.__name):
    open(os.path.join(self.__path, self.__name), 'x').c
lose()
    print(self.__name + " File Created
```

write text data and print if data written successfully or failed

```
def writeTextData(self, text: str):
    # 'w' overwrite
    # 'a' append
    is_opened, file = self.__open('w')
    if is_opened:
        file.write(text)
        file.close()
        print("Data Successfully Written")
    else:
        print("Failed To Write Data!")
```

write binary data and print if data written successfully or failed using pickle

```
def writeBinaryData(self, data):
     is_opened, file = self.__open('wb')
     if is_opened:
          pickle.dump(data, file)
          file.close()
          print("Data Successfully Written")
     else:
          print("Failed To Write Data!")
```

read text data and print if data fetched from the file or not

```
    def readTextData(self) -> str:
        is_opened, file = self.__open('r')
        data = []
        if is_opened:
            data = file.readlines()
            file.close()
            print("Data Fetched")
            return data
        else:
            print("Failed To Fetch Data!")
        return "None"
```

read binary data and print if data fetched from the file or not using pickle

```
def readBinaryData(self):
        is_opened, file = self.__open('rb')
        if is_opened:
            data = pickle.load(file)
            file.close()
```

```
            print("Data Fetched")
            return data
        else:
            print("Failed To Fetch Data!")
        return None
```

What is pickle?

Pickle is a Python module that implements binary protocols for serializing and de-serializing Python object structures. Pickling is the process of converting a Python object hierarchy into a byte stream, and unpickling is the reverse operation of converting a byte stream (from a binary file or bytes-like object) back into an object hierarchy. Pickling (and unpickling) is also known as "serialization," "marshalling," or "flattening"; however, to avoid confusion, the terms "pickling" and "unpickling" are used here.

## 5. Object Detection

In object detection we concern about objects and detection processes so we will need some basic library to keep going

```
import enum
import cv2 as cv
from colors import Colors
import face_recognition
from sklearn.cluster import KMeans
import numpy as np
from owner import Owner
```

first of all, when object detected we will pass a position of the object, so we need to set the positions that following function pass

```
class ObjectPositions(enum.Enum):
    NONE = -1
    CENTER = 0
    TOP_LEFT = 1
    TOP_RIGHT = 2
    BOTTOM_LEFT = 3
    BOTTOM_RIGHT = 4
```

To know that computer detected object successfully we need computer to cover our object with a rectangle something like this

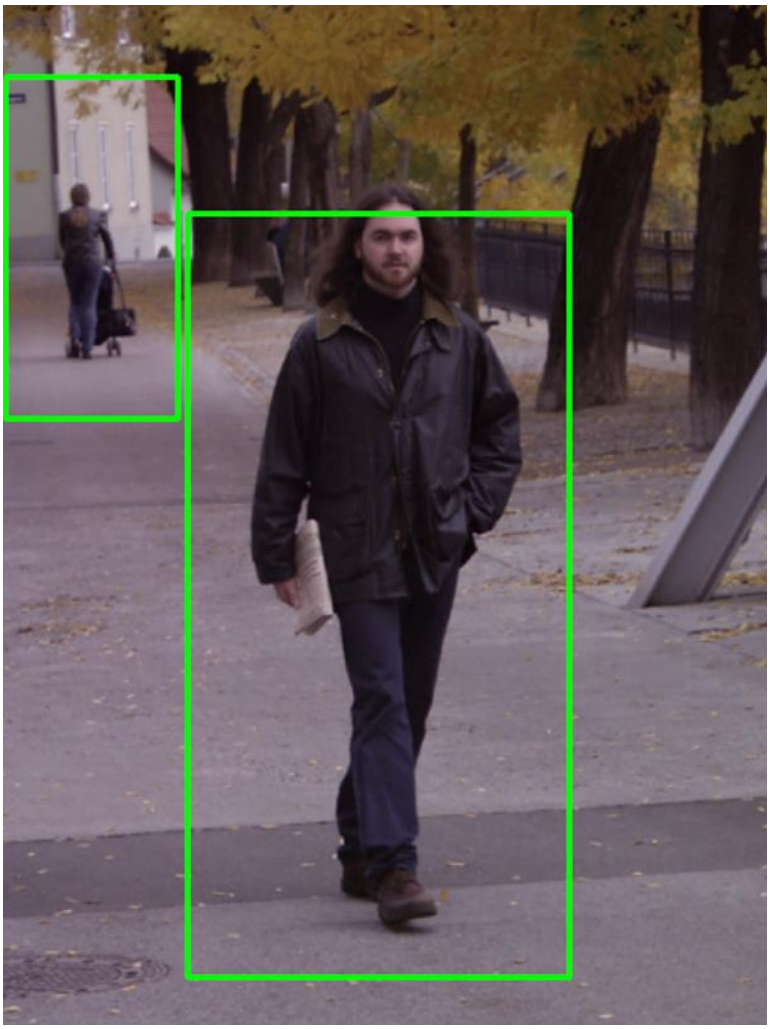


*Figure 3.5*

So, we do the pervious border by this function

```
@staticmethod
def __createObjectBorders(detected_object,
                          frame: cv.VideoCapture,
                          border_color: Colors.BGRColor,
                          border_thickness: int):
    # Display object in rectangle
    for (x, y, width, height) in detected_object:
        cv.rectangle(frame,  # Image color
                     (x, y),  # Start point
                     (width, height),  # End point
                      border_color.value,  # Rectangle c
olor
                      border_thickness)  # Rectangle th
ickness
```

and we add label under the border by this

```
self.__createObjectBorders([(x, height - background_heig
ht, width, height)], frame, border_color, cv.FILLED)
cv.putText(frame, object_name, (x + text_shift_x, height
- text_shift_y), cv.QT_FONT_NORMAL, text_size, text_colo
r.value, 1)
```

we detect shirt of the owner by cropping the frame under the face and pass it as a sample to this function then KMeans library called to list three of main colors of the image then color will be converted to HSV then to regular name like red, blue or white this process occurs in Color Class that we will talk about it later

```
@staticmethod
def __detectShirtColor(shirt_sample):
        # T-Shirt Color Detection Processes #
        # Frame.display("Taken Shirt Sample", shirt_samp
le) # Display shirt sample
        shirt_height, shirt_width, shirt_dim = shirt_sam
ple.shape
        k_means = KMeans(n_clusters=3)
        k_means.fit(np.reshape(shirt_sample, (shirt_heig
ht * shirt_width, shirt_dim)))
        unique_labels, counts_labels = np.unique(k_means
.labels_, return_counts=True)
        count_frame_clusters = 0
        for cluster_center in k_means.cluster_centers_[n
p.argsort(counts_labels)[::-1]]:
            count_frame_clusters += 1
            # Take the bottom cluster
            if count_frame_clusters == 1:
                # Captured colors
                hsv = Colors.convertRGBToHSV([(int(clust
er_center[2]),
                                               int(clust
er_center[1]),
                                               int(clust
er_center[0]))])
                return Colors.convertHSVToColorName(hsv)
```

then we need to know the position of object so we take the width and height of cam let's assume it's 640, 480 so the center of this frame is 320, 240 the half of height and width if object detected his y position less than 240 so it's in the top else it will be in the bottom if his x higher than 320 it will be right less will be left and so on, we implement this technique as following

```
if center_width_cam + 50 > center_width_object > center_
width_cam - 50 and center_height_cam + 50 > center_heigh
t_object > center_height_cam - 50:
            return ObjectPositions.CENTER
        elif center_width_object > center_width_cam and c
enter_height_object < center_height_cam:
            return ObjectPositions.TOP_LEFT
        elif center_width_object < center_width_cam and c
enter_height_object < center_height_cam:
            return ObjectPositions.TOP_RIGHT
        elif center_width_object < center_width_cam and c
enter_height_object > center_height_cam:
            return ObjectPositions.BOTTOM_RIGHT
        elif center_width_object > center_width_cam and c
enter_height_object > center_height_cam:
            return ObjectPositions.BOTTOM_LEFT
        else:
            return ObjectPositions.NONE
```

now we can detect object we will start with face the technique as following camera recognizes faces by

```
detected_face_locations = face_recognition.face_location
s(rgb_small_frame)
detected_face_encodings = face_recognition.face_encoding
s(rgb_small_frame, detected_face_locations)
```

then compare it with owner faces

```
matches = face_recognition.compare_faces(owners_face_enc
odings, face_encoding, tolerance=0.6)
```

if detected face is owner, we will create border with name of owner

```
self.__createObjectBorders(detected_face_location, proce
ssing_frame, Colors.BGRColors.GREEN, 2)
self.__createObjectNameLabel(detected_face_location,
     processing_frame,
     detected_face_name,
     text_size=0.8,
     text_font=cv.QT_FONT_NORMAL,
     text_thickness=1,
     text_color=Colors.BGRColors.White,
     border_color=Colors.BGRColors.GREEN)
```

if frame count % 30 equal zero the function that save shirt color will be called

```
# Run in every 30 frame
if (frame_counter % 30) == 0:
     # Crop frame to specific dimensions according to de
tected face dimensions
     shirt_sample = Frame.crop(original_frame, x * 2, y *
4, width * 4, height * 6)
     for owner in known_owners:
          if  owner.getName().lower().__eq__(detected_fac
e_name.lower()):
               # Detect shirt color and save it
               owner.saveShirtColor(self.__detectShirtCo
lor(shirt_sample))
```

```
                break
```

there some extra things like changing color of image to gray scale to optimize processes

```
# Scale frame and convert it to RGB color (which face_re
cognition uses) to optimize
rgb_small_frame = Colors.convertBGRToRGB(Frame.scale(pro
cessing_frame, (0, 0), 0.25, 0.25))
```

at last, return owners faces positions by

```
owners_faces_position.append(
     (detected_face_name,
      self.__detect_object_position(camera_width,
      camera_height,
     (x * 2, y * 2, width * 4, height * 4)).value))
return owners_faces_position
```

last function will detect of human body by comparing features of detected object to feature of human body features

```
detected_human_body = cv.CascadeClassifier("cv_objects_h
aarcascade/haarcascade_fullbody.xml") \
            .detectMultiScale(gray_frame,
                              1.1,  # Scale factor
                              5)  # Minimum neighbors
```

then we crop the shirt of sample

```
shirt_color = self.__detectShirtColor(shirt_sample)
```

if it's color equal owner shirt color it will surround owner's body with green border with label of his name

```
if owner.getShirtColor().__eq__(shirt_color):
     self.__createObjectBorders(detected_body_location,
processing_frame, Colors.BGRColors.GREEN, 2)
     self.__createObjectNameLabel(
          detected_body_location,
          processing_frame,
          owner.getName().capitalize(),
          text_size=0.5,
          text_font=cv.QT_FONT_NORMAL,
          text_thickness=1,
          text_color=Colors.BGRColors.White,
          border_color=Colors.BGRColors.GREEN)
else:
     self.__createObjectBorders(detected_body_location,
processing_frame, border_color, 2)
```

at last, return owners bodies positions by

```
owners_bodies_position.append((owner.getName().capitaliz
e(), self.__detect_object_position(camera_width, camera_
height, (x, y, width, height)).value))
```

```
        return owners_bodies_position
```

## 6.Colors Recognition

This class is responsible to convert colors from style to style and contains main colors that will we need.

First, we need to import some libraries

```
import enum
import cv2 as cv
```

Second, we need to define our colors that we will use by BGR

```
class BGRColors(enum.Enum):
    GREEN = (0, 230, 0)
    YELLOW = (0, 215, 220)
    RED = (0, 0, 240)
    WHITE = (255, 255, 255)
```

Convert frame colors to gray scale

```
@staticmethod
def convertToGrayScale(frame):
# Convert frame to gray scale to optimize processes
    return cv.cvtColor(frame, cv.COLOR_BGR2GRAY)
```

Convert RGB to HSV

What's the difference between RGB and HSV?

RGB is defined as the amount of red, green, and blue contained in a single value. It employs an additive method, which means that the more of each color added, the brighter it becomes.

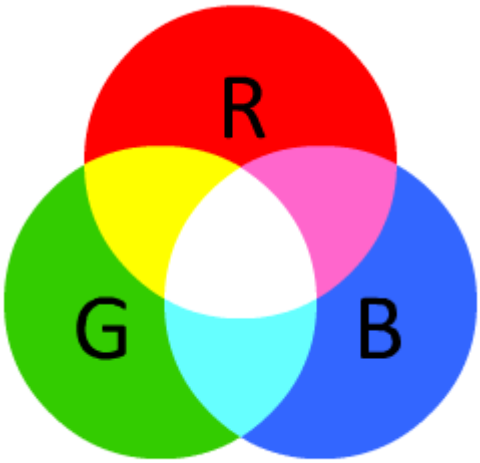



*Figure 3.6*

When looking at a specific color, it is extremely difficult to arbitrarily determine how much of each primary color composes it.

HSV on the other hand uses three parameters to describe color:

- Hue
- Saturation
- Value

With HSV with we now describe our color using a much more cement method as we only theoretically need to transform the Hue to capture the 'red' like color.

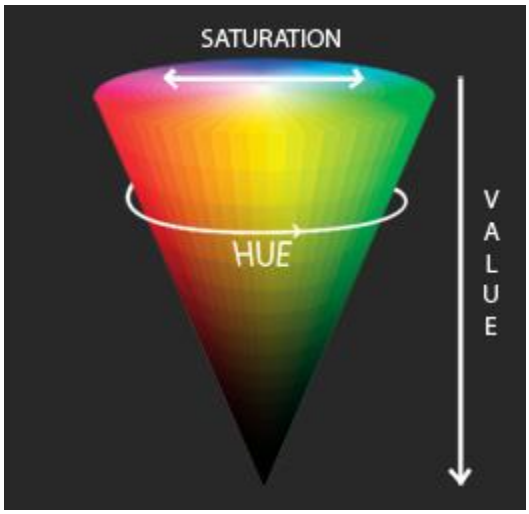



*Figure 3.7*

this how we implement the conversion

```
    @staticmethod
    def convertRGBToHSV(rgb):
        h, s, v = 0, 0, 0
        for r, g, b in rgb:
            r, g, b = r / 255.0, g / 255.0, b / 255.0
            mx = max(r, g, b)
            mn = min(r, g, b)
            df = mx - mn
            if mx == mn:
                h = 0
            elif mx == r:
                h = (60 * ((g - b) / df) + 360) % 360
            elif mx == g:
                h = (60 * ((b - r) / df) + 120) % 360
            elif mx == b:
                h = (60 * ((r - g) / df) + 240) % 360
            if mx == 0:
                s = 0
            else:
```

```
                s = (df / mx) * 100
            v = mx * 100
        return [(int(h), int(s), int(v))]
```

then we convert HSV to regular color name like red, blue or green by setting ranges

```
@staticmethod
    def convertHSVToColorName(hsv):
        for h, s, v in hsv:
            if v == 0:
                return 'Black'
            elif s == 0:
                if 0 < v < 50:
                    return 'Gray'
                elif 50 <= v < 75:
                    return 'Silver'
                elif 75 <= v < 100:
                    return 'White'
            elif s > 0:
                if 0 < h < 20:
                    return 'Red'
                elif 20 <= h < 45:
                    return 'Orange'
                elif 45 <= h < 65:
                    return 'Yellow'
                elif 65 <= h < 110:
                    return 'Lime'
                elif 110 <= h < 150:
                    return 'Green'
                elif 150 <= h < 200:
                    return 'Cyan'
                elif 200 <= h < 250:
```

```
                return 'Blue'
            elif 250 <= h < 300:
                return 'Purple'
            elif 300 <= h < 340:
                return 'Pink'
            elif 340 <= h < 360:
                return 'Marron'
        else:
            return 'Unknown'
```

## 7. Car Deceleration

In this class we aim to send speed ranges and car orders to the USBasp board.

First, import necessary libraries

```
import enum
import serial
from time import sleep
```

Second, we will create speed ranges

```
class SpeedRanges(enum.Enum):
    BREAK = 0
    SLOWEST = 20
    SLOW = 50
    NORMAL = 80
    FAST = 100
    FASTEST = 150
```

Third, UART orders that we will send to board we will send these data to ATmega32 by communication protocol, but we will discuss this later

```
class __UARTOrders(enum.Enum):
     FORWARD = 'A'
     FORWARD_RIGHT = 'Y'
     FORWARD_LEFT = 'E'
     BACKWARD = 'B'
     BACKWARD_RIGHT = 'Q'
     BACKWARD_LEFT = 'W'
     ROTATE_RIGHT = 'R'
     ROTATE_LEFT = 'L'
     STOP = 'S'
```

We move car according to MACRO that we will send to car like

```
@staticmethod
def moveForward(speed: SpeedRanges):
     CarActions.__transmitToATmega(CarActions.__UARTOrde
rs.FORWARD, speed)
```

# CHAPTER 4

## 1. Embedded Systems Components Selection

### 1. 1 Microcontroller VS Microprocessor

Choosing the right device to base your new design can be daunting. There are many implications for the need to balance the price, performance, and power consumption. First, there will be immediate technological considerations for the design that you can embark on. However, if there is a microcontroller (MCU) or a microprocessor (MPU), As the basis for a platform approach, a decision can have long lasting consequences.

The difference between the microprocessor and the microcontroller is becoming an important debate at this point. Typically, the MCU uses on-chip embedded Flash memory to store and chip embedded Flash memory to store and run its program. Storing the pro run its program.

Storing the program this way means that the MCU gram this way means that the MCU has a shorter start has a shorter start--up time and runs the code quickly. The only up time and runs the code quickly.

The only practical limitation to the use of embedded memory is that the total practical limitation to the use of embedded memory is that the total memory space available is finite. Most of the Flash MCU devices memory space available is finite. Most of the Flash MCU devices available on the market have a maximum of 2 Mbytes of program have a maximum of 2 Mbytes of program memory. Depending on the application, this may prove to be a memory. Depending on the application, this may prove to be a limiting factor.

MPUs do not have the same memory constraint. External memory is MPUs do not have the same memory constraint.

External memory is used to provide program and data storage. Typically, the program is used to provide program and data storage. Typically, the program is stored in stored in non-volatile memory, such as NAND or Serial Flash. volatile memory, such as NAND or Serial Flash. This is loaded into an external DRAM at start--up, and execution starts. up, and execution starts. This means that the MPU will not be up and running as quickly as the This means that the MPU will not be up and running as quickly as the MCU, but the amount of DRAM and NVM you can connect to the MCU, but the amount of DRAM and NVM you can connect to the processor is in the range of hundreds of Mbytes and even G processor is in the range of hundreds of Mbytes and even G Bytes for Bytes for NAND.NAND.

Power is another difference. By integrating its own power supply, the Power is another difference. By integrating its own power supply, the MCU needs only one single voltage power rail. By comparison, the MCU needs only one single voltage power rail.

By comparison, the MPU requires several differential voltage rails MPU requires several differential voltage rails for core, DDR, etc. For core, DDR, etc. The developer needs to cater for this with additional power on developer needs to cater for this with additional power on--board ICs / board ICs / converters.

Atmega32 and Arduino Differences:
In short, the ATmega328 is a microcontroller chip that is found on Arduino Uno boardsTheATmega328 microcontrollers come from the 8bit AVR family of microcontrollers. The picture below shows the AVR Atmega32 microcontroller.

Atmega32 microcontroller chip. The exact part number of this chip is ATMEGA328P PU as found on the top of the chip.

Atmega328 plugs into the Arduino Un socket as shown in the image below. There are some Arduino Uno boards with a surface mounted

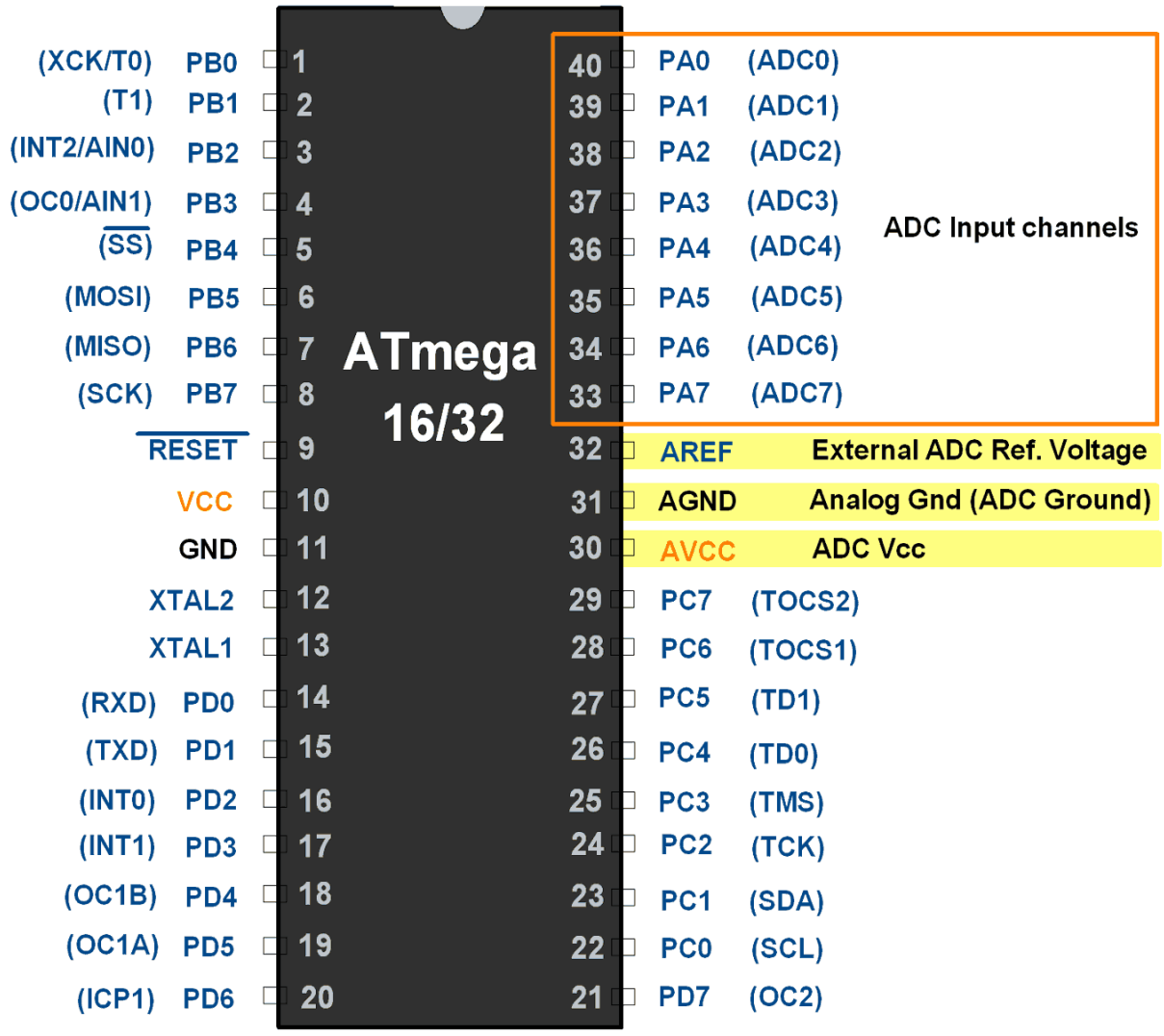



*Figure 4.1*

Atmega328 chip. In this case, a small square chip is soldered on top of the Arduino.

What the Microcontroller Does Inside Arduino on the Arduino, the microcontroller is the main component that does all the work. If the Arduino sketch is loaded to the Arduino, it is actually loaded to the memory inside the microcontroller chip. The microcontroller will then run or execute the sketch when the Arduino is powered up or reset after programming. AVR microcontrollers were previously manufactured by a company

called Atmel. Atmel was bought by Microchip, so AVR microcontrollers are no longer an Atmel product, but a Microchip product Shortly we can describe the difference between Arduino and the regular MCU in one point which is MCU for more customized projects which need more efficiency and performance in the other hand Arduino which need more efficiency and performance in the other hand Arduino for smaller or generalized purposes like for smaller or generalized purposes like students' activities.


*Figure 4.2*

## 1. 2 DC Motor

A gear motor is an all-in-one combination of a motor and gearbox. The addition of a gear head to a motor reduces the speed while increasing the torque output. The most important parameters in regard to gear motors are speed (rpm), torque (lb-in) and efficiency (%). In order to select the most suitable gear motor for your application you must first compute the load, speed, and torque requirements for your application. ISL Products offers a variety of Spur Gear Motors, Planetary Gear Motors and Worm Gear Motors to meet all application requirements. Most of our DC motors can be

complemented with one of our unique gearheads, providing you with a highly efficient gear motor solution.


Motor

| | | | |
|---|---|---|---|
| Voltage | DC 3V | DC 5V | DC 6V |
| Current | 100 MA | 100 MA | 120MA |
| Reduction rate | 48:1 | | |
| RPM (With tire) | 100 | 190 | 240 |
| Tire Diameter | 65mm | | |
| Car Speed(M/minute | 20 | 39 | 48 |
| Motor Weight (g) | 29 | | |
| Motor Size | 70mm*22mm*18mm | | |
| Noise | <65dB | | |

*Figure 4.3*

## 1. 3 PWM

Definition Of Pulse Width Modulation
Pulse Width Modulation (PWM) is a nifty current control technique that enables you to control the speed of motors, heat output of heaters, and much more in an energy-efficient (and usually quieter) manner. Existing applications for PWM include but are not limited to Variable speed fan controllers. VRF HVAC compressor drives. Hybrid and electric vehicle motor drive circuits. LED Dimmers. Pulse width modulation has changed the world by slashing the power consumption of appliances utilizing motors such as

inverter air conditioners [PDF], inverter refrigerators, inverter washing machines, among many others. For example, inverter air conditioners can consume less than half the energy that their non-inverter counterparts do in some cases. In this day and age, if a device is advertised as having a variable speed compressor or variable speed fan (this doesn't include two or three-speed fans), there is a significant chance that it is utilizing PWM!

Why Use PWM?

Aspiring electrical engineers may want to know why they should use pulse width modulation to control devices, and homeowners have similar question which has the same answer: Why use inverter air conditioners, or other variable speed appliances. The answer to both questions is: PWM varies the speed of the appliances' motors, so they only consume as much power as they need, but without the usual consequence of burning off unused current as heat. An example of an older alternative is a simple transistor circuit that varies the current passing through it by varying its resistance. The same efficiency rule that applies to resistors also applies to transistors — Their resistance results in wasted energy because they burn off some of it as heat. They act like heaters in that regard. Fortunately, these circuits were never mainstream. Appliances such as air conditioners and refrigerators just ran at full speed all the time, making lots of noise, and wasting lots of energy because they had to cycle on and off frequently. PWM does make use of transistors, but in a different way, as explained below.

PWM Motor Controller Example.

If you're looking to get started with PWM, a great entry point would be a 555 PWM circuit, Arduino PWM circuit (very convenient, as you can easily modify its behavior via a simple source code modification), or the MSP30 PWM circuit that I wrote about on Kompulsa.

How Does PWM Work?

PWM works by pulsating DC current and varying the amount of time that each pulse stays 'on' to control the amount of current that flows to a device such as an LED. PWM is digital, which means that it has two states: on and off (which correspond to 1 and 0 in the binary context, which will become more relevant to you if using microcontrollers). The longer each pulse is on, the brighter the LED will be. Due to the fact that the interval between pulses is so brief, the LED doesn't actually turn off. In other words, the LED's power source switches on and off so fast (thousands of times per second) that the LED actually stays on without flickering. This is called PWM dimming, and such as circuit is just called a PWM LED dimmer circuit. The squares in the PWM illustration below are the pulses which represent 'on' time, and the depressed areas represent the time that the power is 'off'. Both the squares and depressed areas are the same 'width', therefore the duty cycle is 50%. PWM signals are typically square waves, like the one in the illustration below.

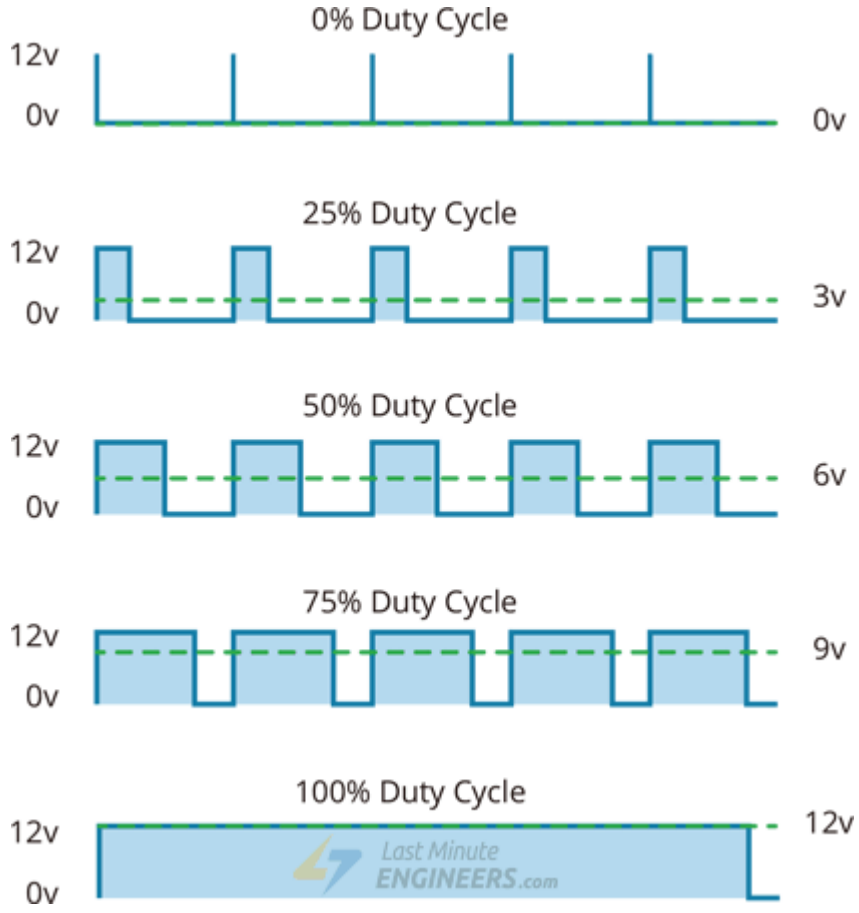



*Figure 4.4*

## 1. 4 L298N VS L293D Motor drivers

This L298N Motor Driver Module is a high-power motor driver module for driving DC and Stepper Motors. This module consists of an L298 motor driver IC and a 78M05 5V regulator. L298N Module can control up to 4 DC motors, or 2 DC motors with directional and speed control.

L298N Module Pin Configuration:

| Pin Name | Description |
|---|---|
| IN1 & IN2 | Motor A input pins. Used to control the spinning direction of Motor A |
| IN3 & IN4 | Motor B input pins. Used to control the spinning direction of Motor B |
| ENA | Enables PWM signal for Motor A |
| ENB | Enables PWM signal for Motor B |
| OUT1 & OUT2 | Output pins of Motor A |
| OUT3 & OUT4 | Output pins of Motor B |
| 12V | 12V input from DC power Source |
| 5V | Supplies power for the switching logic circuitry inside L298N IC |
| GND | Ground pin |

L298 Module Features & Specifications:

- Driver Model: L298N 2A
- Driver Chip: Double H Bridge L298N
- Motor Supply Voltage (Maximum): 46V
- Motor Supply Current (Maximum): 2A
- Logic Voltage: 5V
- Driver Voltage: 5-35V
- Driver Current:2A
- Logical Current:0-36Ma
- Maximum Power (W): 25W
- Current Sense for each motor
- Heatsink for better performance
- Power-On LED indicator

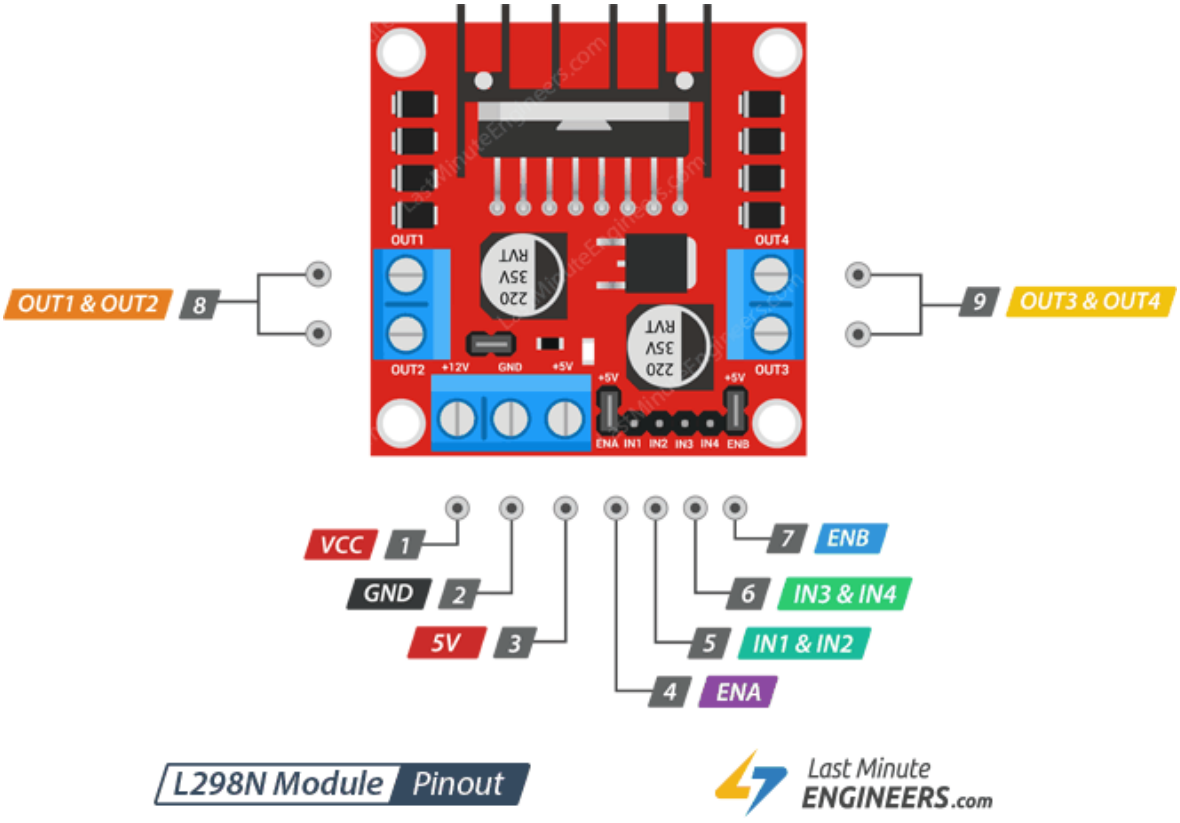



*Figure 4.5*

L293D Motor Driver

The L293D is a dual-channel H-Bridge motor driver capable of driving a pair of DC motors or one stepper motor. That means it can individually drive up to two motors making it ideal for building two-wheel robot platforms.

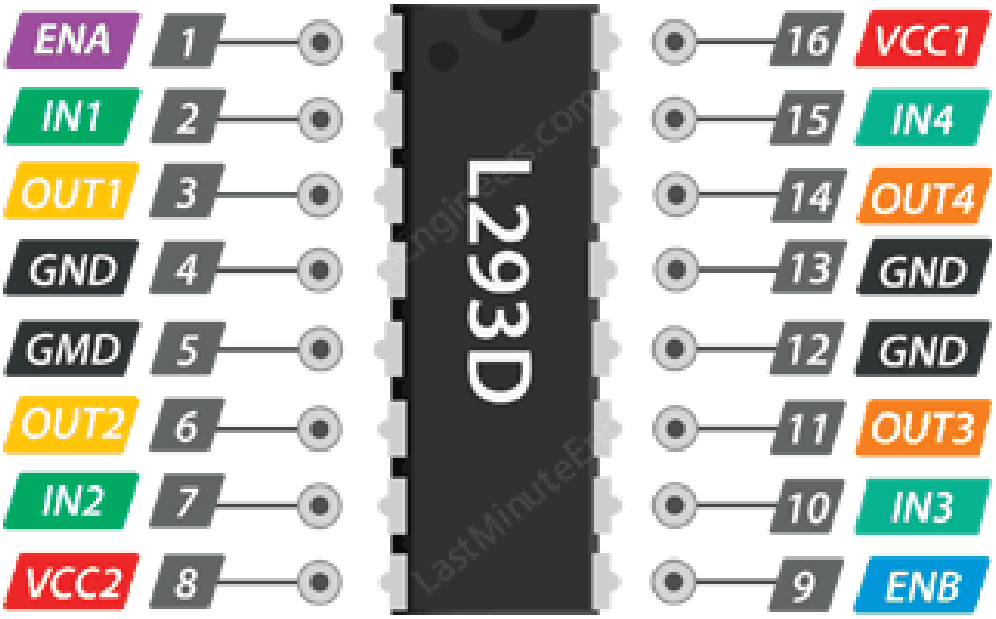



*Figure 4.6*

The spinning direction of a motor can be controlled by applying either a logic HIGH (5 Volts) or logic LOW(Ground) to these pins. The below chart illustrates how this is done.

| IN1 | IN2 | Spinning Direction |
|---|---|---|
| Low (0) | Low (0) | Motor OFF |
| High (1) | Low (0) | Forward |

| | | |
|---|---|---|
| Low (0) | High (1) | Backward |
| High (1) | High (1) | Motor OFF |

## 1. 5 Step Down Converter

The TPS53355 Inductor-On-Top step-down buck converter design enables high power density through reduction of X-Y PCB area by placing the output inductor on top of the IC and results in 92% minimum efficiency for 66W and 100W output power designs (3.3V and 5V output) requiring only 4x100µF ceramic output caps.


*Figure 4.7*

Features

- 94% Efficiency at 12VIN, 5Vout,20A output current, 650KHz
- 6W power loss at 100W Output Power
- 14Mv Output Voltage Ripple at 20A output current
- Only 4x100µF output ceramic capacitors

- 27Mv Output Voltage Over-shoot in response to 5A to a 10A load increase, 2.5A/µs
- 58.7°C IC case temperature at 5Vout and 20A output current (100W Output Power).

It's used to decrease the voltage that enter to the atmega32
MCU from 12V that comes from the motors to 5V to enter to atmega32 MCU.

## 1. 6 USBasp

USBasp is a USB in-circuit programmer for Atmel AVR controllers. It simply consists of an ATMega88 or an ATMega8 and a couple of passive components. The programmer uses a firmware-only USB driver, no special USB controller is needed. Features Works under multiple platforms. Linux, Mac OS X and Windows are tested. No special controllers or SMD components are needed. Programming speed is up to 5kBytes/sec. SCK option to support targets with low clock speed (< 1,5MHz).

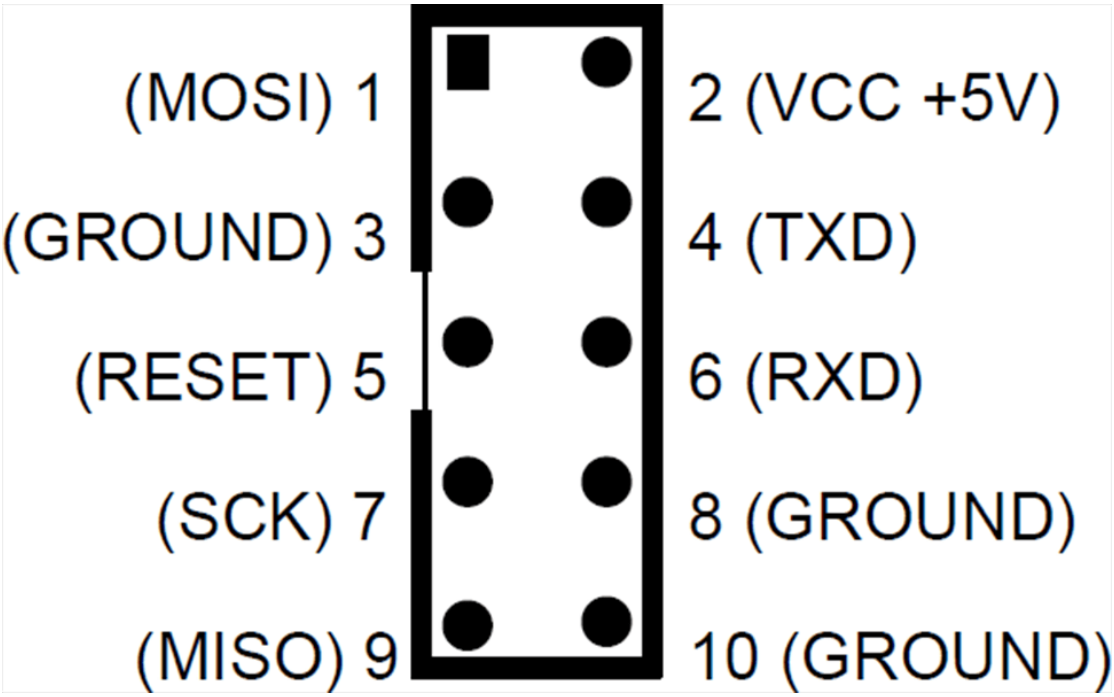



*Figure 4.8*

# 2.ATmega32 and Programs Installation

The ATmega32 is a low-power CMOS 8-bit microcontroller based on the AVR enhanced RISC architecture. By executing powerful instructions in a single clock cycle, the ATmega32 achieves throughputs approaching 1 MIPS per MHz allowing the system designer to optimize power consumption versus processing speed.

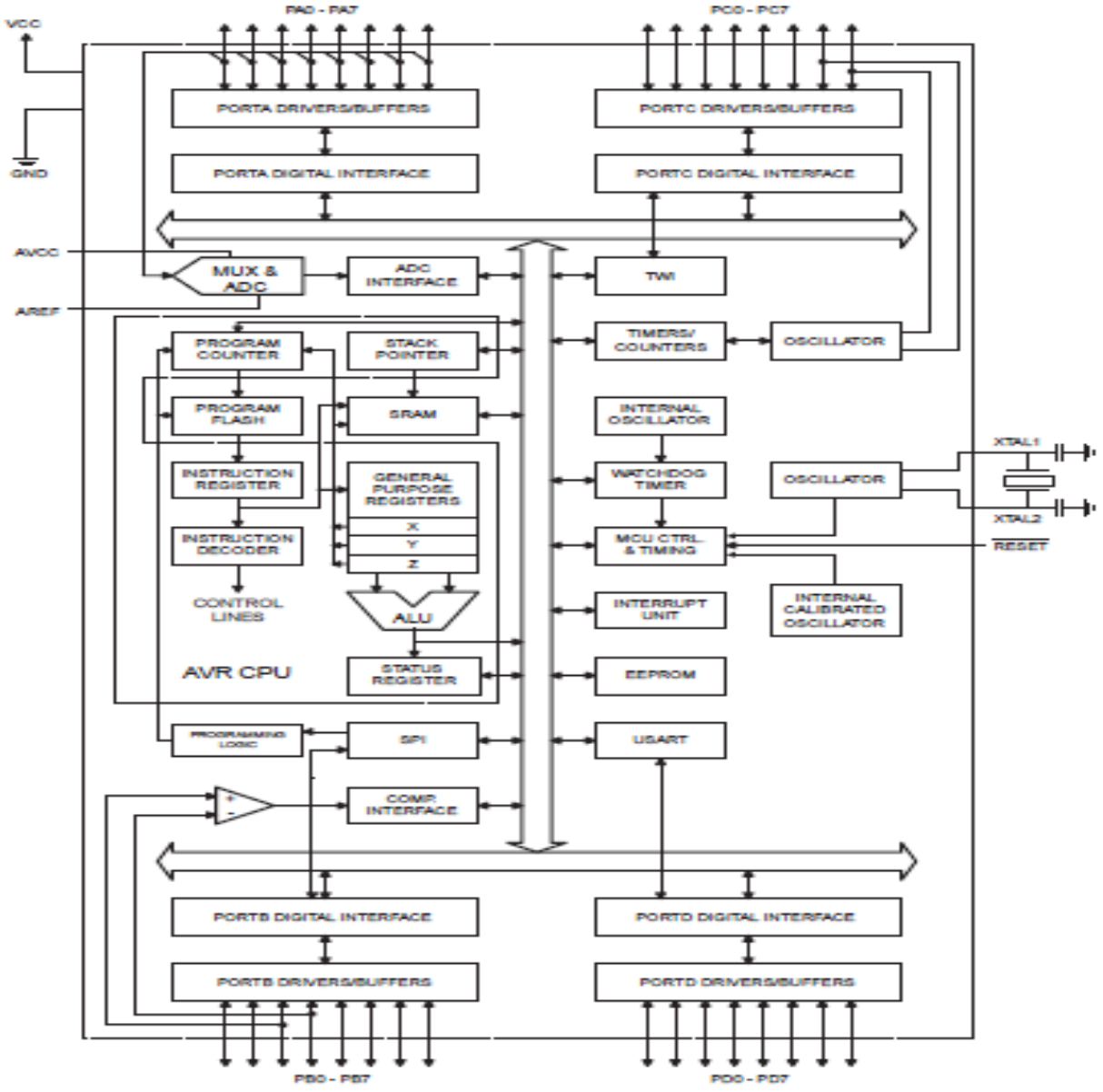



*Figure 4.9*

Features

1) High-performance, Low-power AVR® 8-bit Microcontroller
2) Advanced RISC Architecture
    - 131 Powerful Instructions – Most Single-clock Cycle Execution
    - 32 x 8 General Purpose Working Registers
    - Fully Static Operation
    - Up to 16 MIPS Throughput at 16 MHz
    - On-chip 2-cycle Multiplier
3) Nonvolatile Program and Data Memories
    - 32K Bytes of In-System Self-Programmable Flash
- Endurance: 10,000 Write/Erase Cycles
    - Optional Boot Code Section with Independent Lock Bits
- In-System Programming by On-chip Boot Program
- True Read-While-Write Operation
    - 1024 Bytes EEPROM
- Endurance: 100,000 Write/Erase Cycles
    - 2K Byte Internal SRAM
    - Programming Lock for Software Security
4) JTAG (IEEE std. 1149.1 Compliant) Interface
    - Boundary-scan Capabilities According to the JTAG Standard
    - Extensive On-chip Debug Support
    - Programming of Flash, EEPROM, Fuses, and Lock Bits through the JTAG Interface
5) Peripheral Features
    - Two 8-bit Timer/Counters with Separate Prescalers and Compare Modes
    - One 16-bit Timer/Counter with Separate Prescalers, Compare Mode, and Capture
- Mode
    - Real Time Counter with Separate Oscillator

    - Four PWM Channels
    - 8-channel, 10-bit ADC
- 8 Single-ended Channels
- 7 Differential Channels in TQFP Package Only
- 2 Differential Channels with Programmable Gain at 1x, 10x, or 200x
    - Byte-oriented Two-wire Serial Interface
    - Programmable Serial USART.

## 2.1 Extreme AVR burner SW

This software is used to upload the code on the chip of ATmega32. It's an easy way to burn your code and its fast too.

How to burn or upload your code
First there must be a connector to interface the atmega32 MCU with the computer to upload the code from the computer to the atmega32 like:

- USBasp (USB programmer for Atmel AVR controllers).

Steps to upload the code on ATmega23:

Step 1: Open Extreme AVR burner

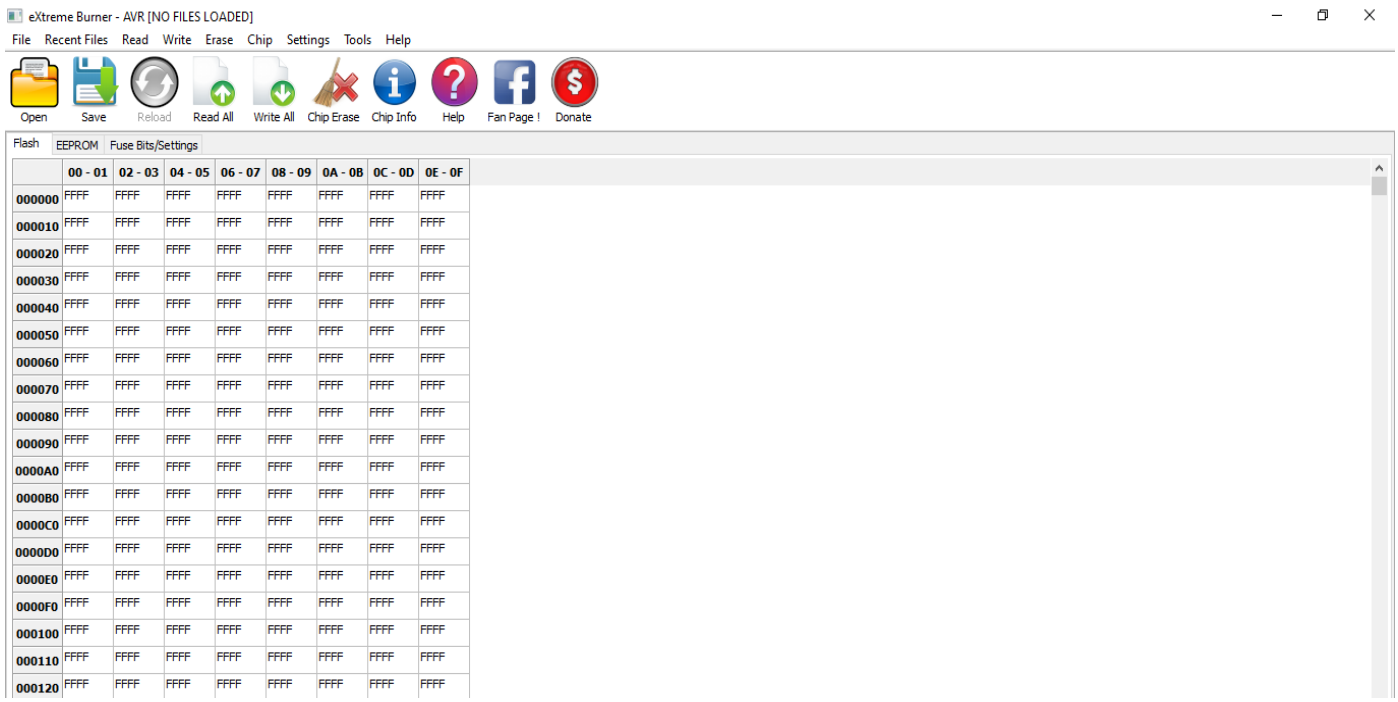



*Figure 4.10*

## Step 2: Click on open to browse your HEX file

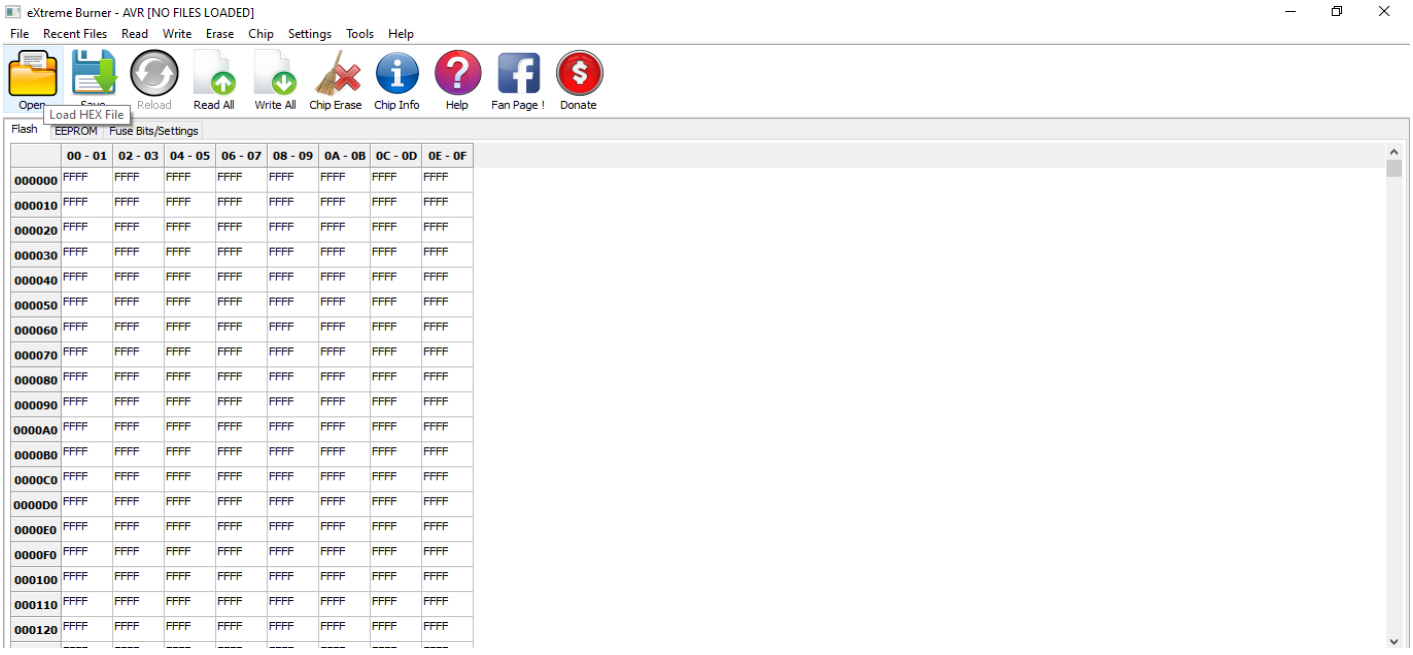



*Figure 4.11*

## Step 3: Choose the HEX. File

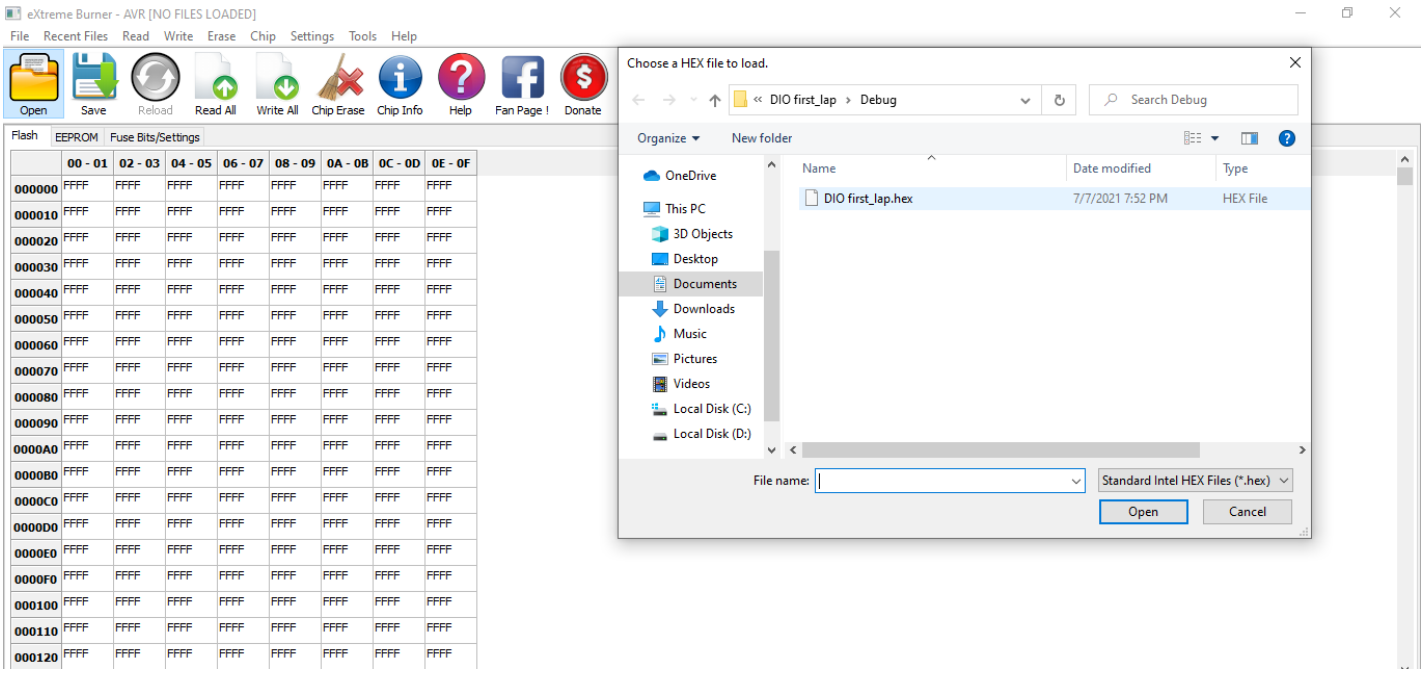
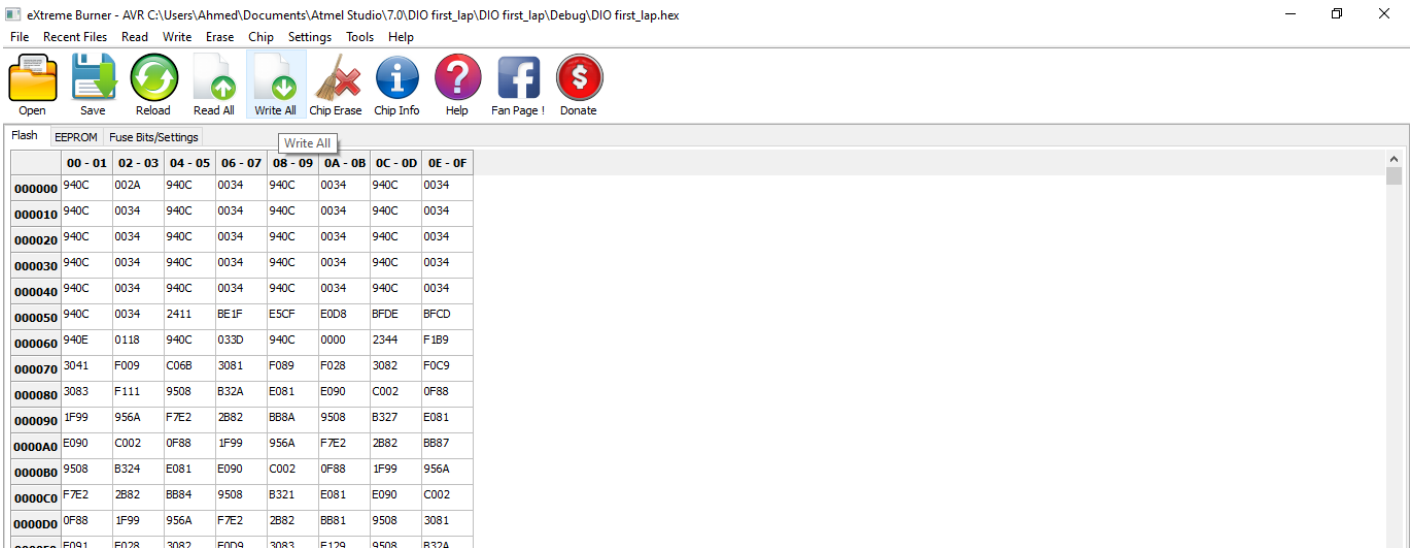



*Figure 4.12*

## Step 4: Click on write all.




| | 00 - 01 | 02 - 03 | 04 - 05 | 06 - 07 | 08 - 09 | 0A - 0B | 0C - 0D | 0E - 0F |
|---|---|---|---|---|---|---|---|---|
| 000000 | 940C | 002A | 940C | 0034 | 940C | 0034 | 940C | 0034 |
| 000010 | 940C | 0034 | 940C | 0034 | 940C | 0034 | 940C | 0034 |
| 000020 | 940C | 0034 | 940C | 0034 | 940C | 0034 | 940C | 0034 |
| 000030 | 940C | 0034 | 940C | 0034 | 940C | 0034 | 940C | 0034 |
| 000040 | 940C | 0034 | 940C | 0034 | 940C | 0034 | 940C | 0034 |
| 000050 | 940C | 0034 | 2411 | BE1F | E5CF | E0D8 | BFDE | BFCD |
| 000060 | 940E | 0118 | 940C | 033D | 940C | 0000 | 2344 | F1B9 |
| 000070 | 3041 | F009 | C06B | 3081 | F089 | F028 | 3082 | F0C9 |
| 000080 | 3083 | F111 | 9508 | B32A | E081 | E090 | C002 | 0F88 |
| 000090 | 1F99 | 956A | F7E2 | 2B82 | BB8A | 9508 | B327 | E081 |
| 0000A0 | E090 | C002 | 0F88 | 1F99 | 956A | F7E2 | 2B82 | BB87 |
| 0000B0 | 9508 | B324 | E081 | E090 | C002 | 0F88 | 1F99 | 956A |
| 0000C0 | F7E2 | 2B82 | BB84 | 9508 | B321 | E081 | E090 | C002 |
| 0000D0 | 0F88 | 1F99 | 956A | F7E2 | 2B82 | BB81 | 9508 | 3081 |

*Figure 4.13*

## 2.2 Atmel Studio IDE

Studio 7 is the integrated development platform (IDP) for all AVR® and SAM microcontroller applications. The Atmel Studio 7 IDP provides a streamlined and simple environment for writing, building, and debugging C/C++ or assembly code applications. It also communicates seamlessly with AVR and SAM debuggers, programmers, and development kits.

Downloading and Installing:

1) Download the latest Atmel Studio installer: Atmel Studio
    - The web installer is a small file (<10 MB) and will download specified components as needed.
    - The offline installer has all components embedded
2) Atmel Studio can be run side-by-side with older versions of Atmel Studio and AVR Studio®. Uninstallation of any previous versions is not required.
3) Verify the hardware and software requirements from the "System Requirements" section
4) Make sure your user has local administrator privileges
5) Save all your work before starting. The installation might prompt you to restart if required.
6) Disconnect all Atmel USB/Serial hardware devices
7) Double-click the installer executable file and follow the installation wizard
8) Once finished, the installer displays an option to Start Atmel Studio after completion. If you choose to open, then note that Atmel Studio will launch with administrative privileges, since the installer was either launched as administrator or with elevated privileges.
9) In Atmel Studio you may see an update notification (flag symbol) next to the Quick Launch field in the title bar. Here you may select and install updated components or device support.
    - Downloading Offline Documentation

- If you would like to work offline, it would be advisable to use the offline documentation for Studio 7.
- To do this, from the Studio 7 Start Page, click on Download documentation. When the help viewer pops up, first click the Online button and search for documentation of interest, such as data sheets, user manuals, and application notes (wait for the available documents to show up).
- In the example below, we are choosing to download the Power Debugger user manual, the ATtiny817 Xplained Pro user manual, as well as the ATtiny817 Complete data sheet. Clicking update will then initiate the download.

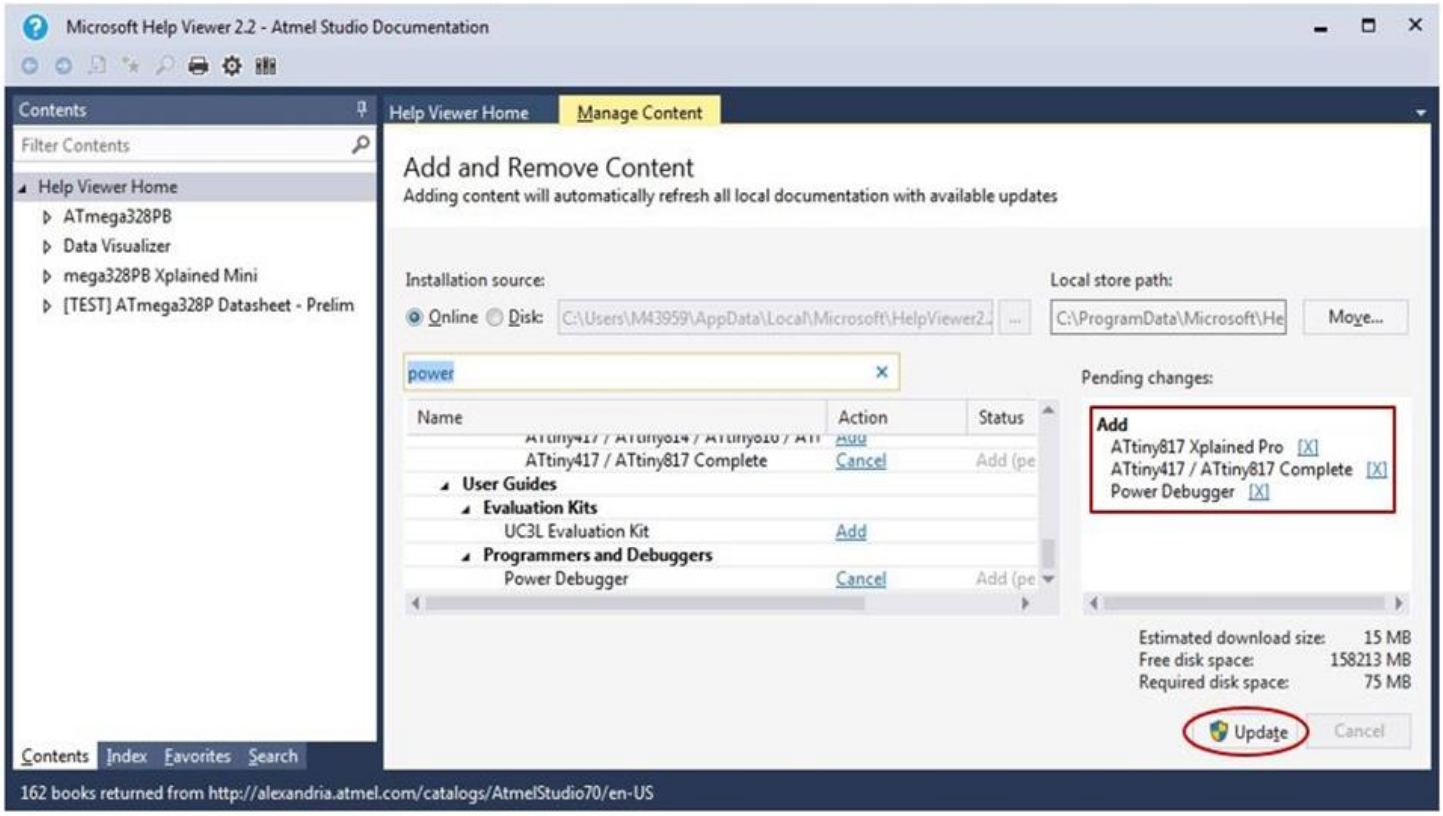



*Figure 4.14*

# CHAPTER 5

## 1. Flowchart of Embedded Systems

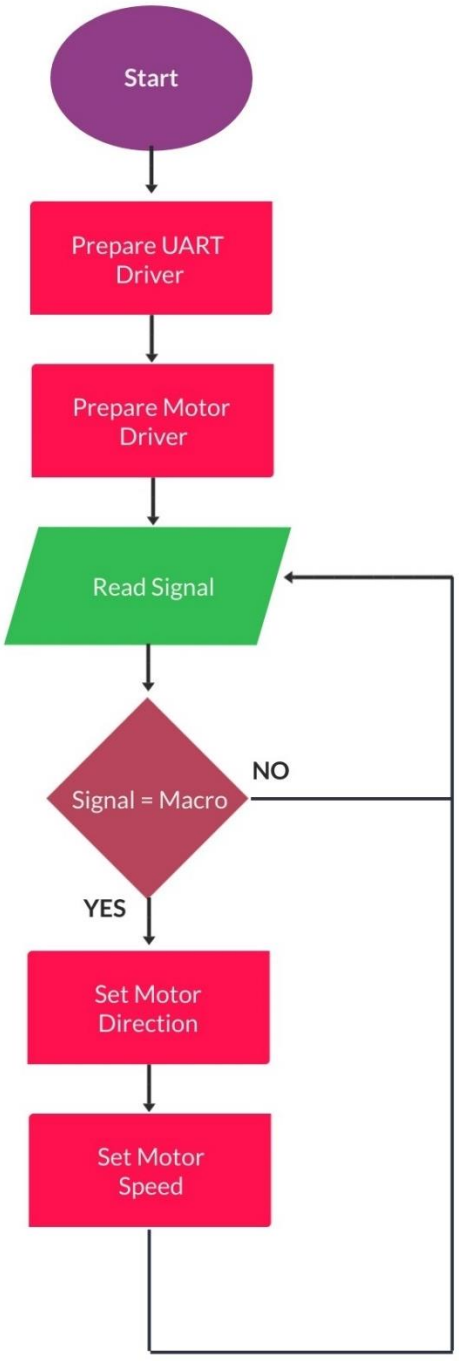



*Figure 5.1*

## 2.Macros

Initialization of macros to specify direction of motor these macros that already organized with Raspberry pi to receive if matches we will act with motors

Here we need to import some libraries

```
#define F_CPU 16000000
#include<util/delay.h>
#include "Motor.h"
#include "PWM.h"
#include "UART.h"
```

And this is the initialized macros

```
#define FORWARD 'A'
#define BACKWARD 'B'
#define RIGHT 'R'
#define LEFT 'L'
#define STOP 'S'
#define FORWARD_RIGHT 'Y'
#define FORWARD_LEFT 'E'
#define BACKWARD_RIGHT 'Q'
#define BACKWARD_LEFT 'W'
```

## 3.Motors

Initialization of motors and baud rate to drive electrical signal to right direction of motors

First, we need to set our baud rate to UART

```
UART_init (9600);
```

Define motors

```
MOTOR M_R,M_L;
Motor_init ( & M_R, DIO_PORTC , DIO_PIN2 , DIO_PORTD , D
IO_PIN3 , 1);
Motor_init ( & M_L, DIO_PORTC , DIO_PIN1 , DIO_PORTB , D
IO_PIN1 , 2); // c1= b2
```

And this how we set speed and direction

```
Motor_dir(&M_R , MOTOR_CCW );
Motor_setspeed(&M_R,100);
Motor_dir(&M_L , MOTOR_CW);
Motor_setspeed(&M_L ,200);
```

## 4. Main

```
while(1){
          ch = UART_RxChar();
          if(ch == FORWARD){
               //FORWARD
               Motor_dir(&M_R, MOTOR_CW);
               Motor_dir(&M_L, MOTOR_CCW);
               Motor_setspeed(&M_R, 100);
```

```
        Motor_setspeed(&M_L, 100);
    }
    else if(ch == BACKWARD){
        //BACKWARD
        Motor_dir(&M_R, MOTOR_CCW);
        Motor_dir(&M_L, MOTOR_CW);
        Motor_setspeed(&M_R, 100);
        Motor_setspeed(&M_L, 100);
    }
    else if(ch == RIGHT){
        //RIGHT
        Motor_dir(&M_R, MOTOR_CCW);
        Motor_dir(&M_L, MOTOR_CCW);
        Motor_setspeed(&M_R, 100);
        Motor_setspeed(&M_L, 100);
    }
    else if(ch == LEFT){
        //LEFT
        Motor_dir(&M_R, MOTOR_CW);
        Motor_dir(&M_L, MOTOR_CW);
        Motor_setspeed(&M_R, 100);
        Motor_setspeed(&M_L, 100);
    }
    else if(ch == STOP){
        //STOP
        Motor_dir(&M_R, MOTOR_CW);
        Motor_dir(&M_L, MOTOR_CCW);
        Motor_setspeed(&M_R, 0);
        Motor_setspeed(&M_L, 0);
    }
    else if(ch == FORWARD_LEFT){
        //FORWARD_LEFT
        Motor_dir(&M_R, MOTOR_CW);
        Motor_dir(&M_L, MOTOR_CCW);
```

```
                Motor_setspeed(&M_R, 100);
                Motor_setspeed(&M_L, 50);
            }
            else if(ch == FORWARD_RIGHT){
                //FORWARD_RIGHT
                Motor_dir(&M_R, MOTOR_CW);
                Motor_dir(&M_L, MOTOR_CCW);
                Motor_setspeed(&M_R, 50);
                Motor_setspeed(&M_L, 100);
            }
            else if(ch == BACKWARD_LEFT){
                //BACKWARD_LEFT
                Motor_dir(&M_R, MOTOR_CCW);
                Motor_dir(&M_L, MOTOR_CW);
                Motor_setspeed(&M_R, 100);
                Motor_setspeed(&M_L, 50);
            }
            else if(ch == BACKWARD_RIGHT){
                //BACKWARD_RIGHT
                Motor_dir(&M_R, MOTOR_CCW);
                Motor_dir(&M_L, MOTOR_CW);
                Motor_setspeed(&M_R, 50);
                Motor_setspeed(&M_L, 100);
            }

        }
}
```

# CHAPTER 6

## 1. Mechanical Design of 3D Printing

The Dynamic Motion Calculation:

$$F_c = m \times w^2 \times r$$

Where:

M: mass of the car
W: rotation of the wheels ($w = \frac{2 \times \Pi \times n}{60}$)
R: radius of curvature

$$F_c = 1kg * 9.81 * \frac{2 * \Pi * 70}{60} * 30 = 219.911\ N$$

## 2.The Static Motion of Car

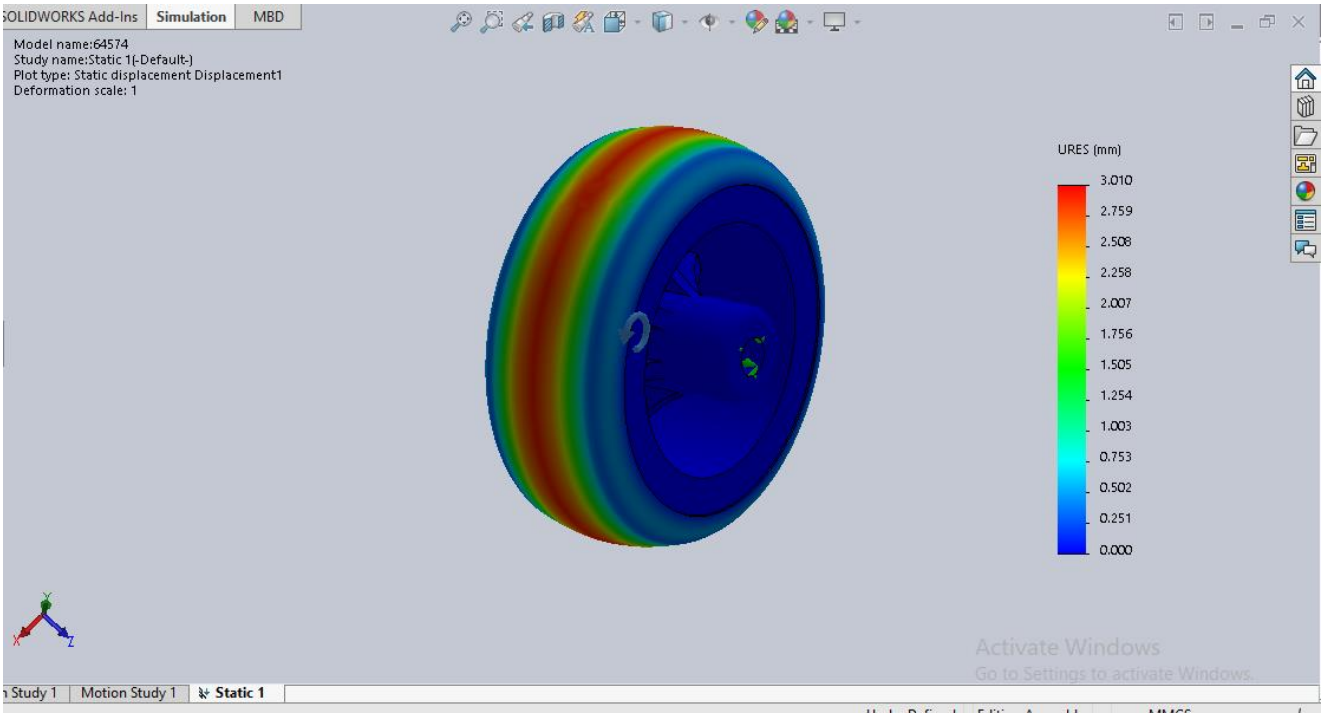



*Figure 6.1*

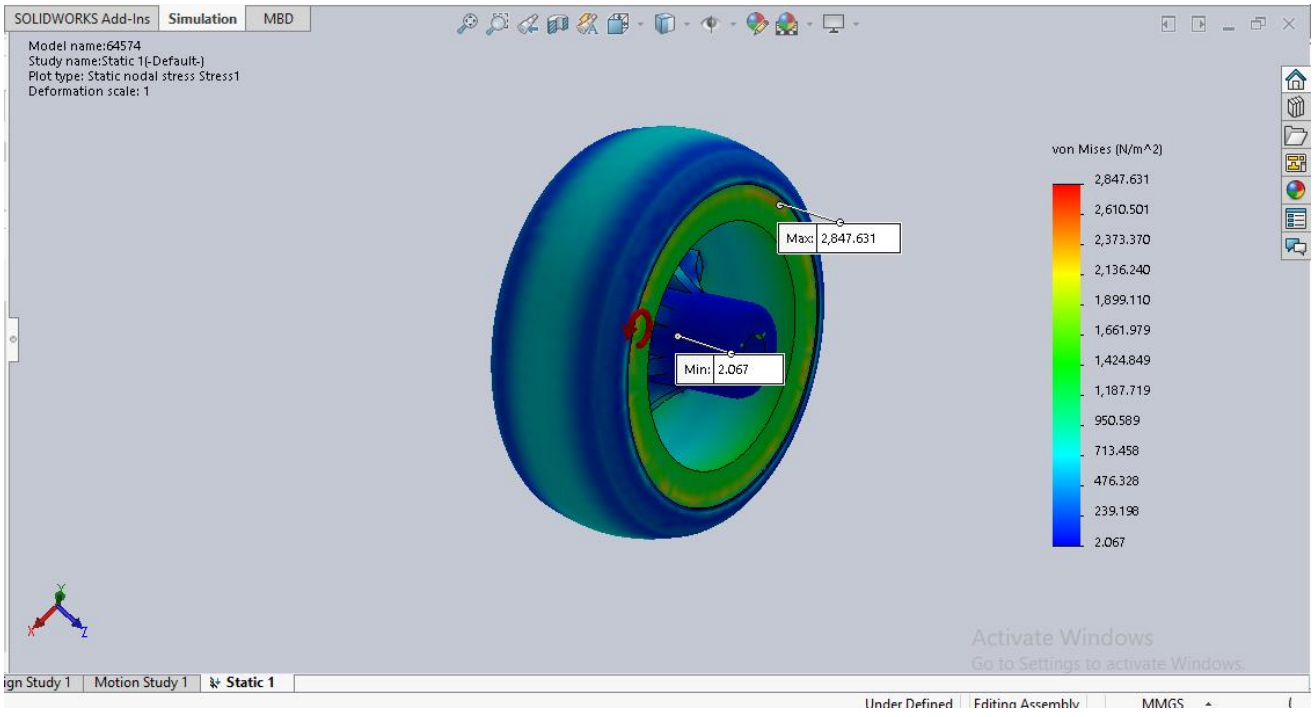



*Figure 6.2*

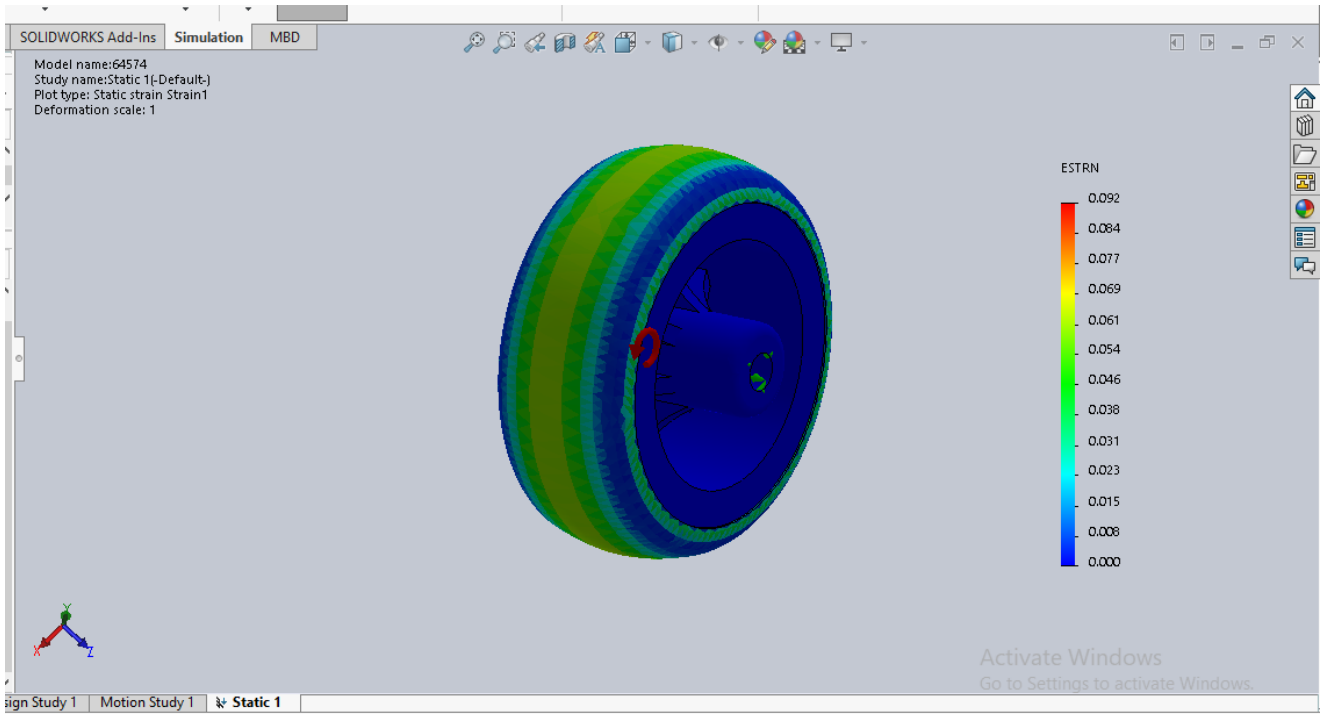



*Figure 6.3*

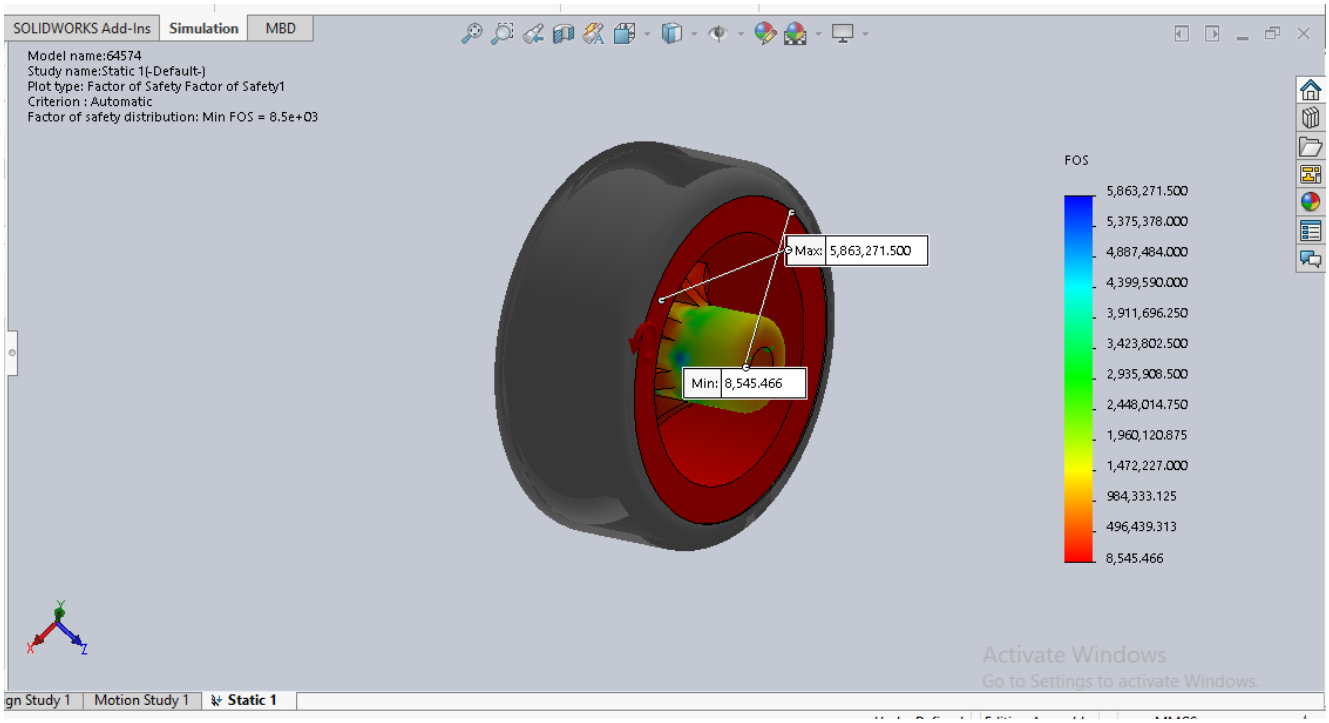



*Figure 6.4*

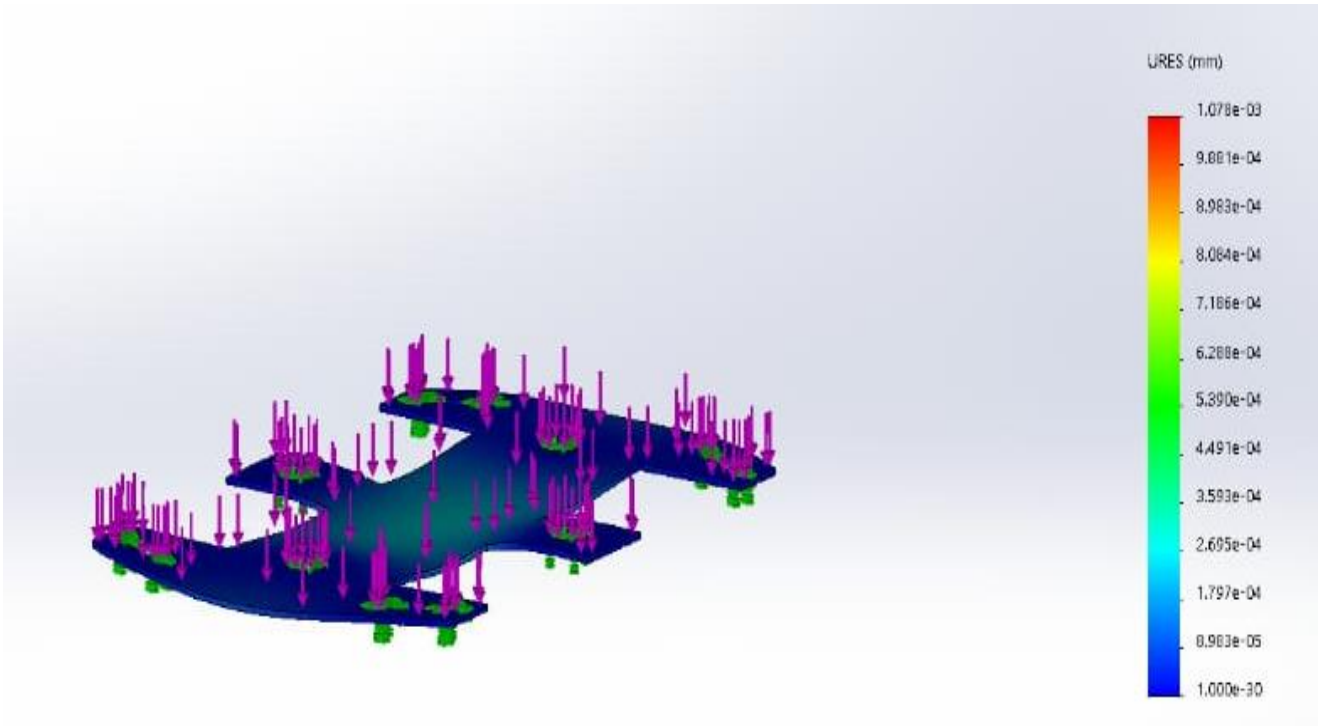



*Figure 6.5*

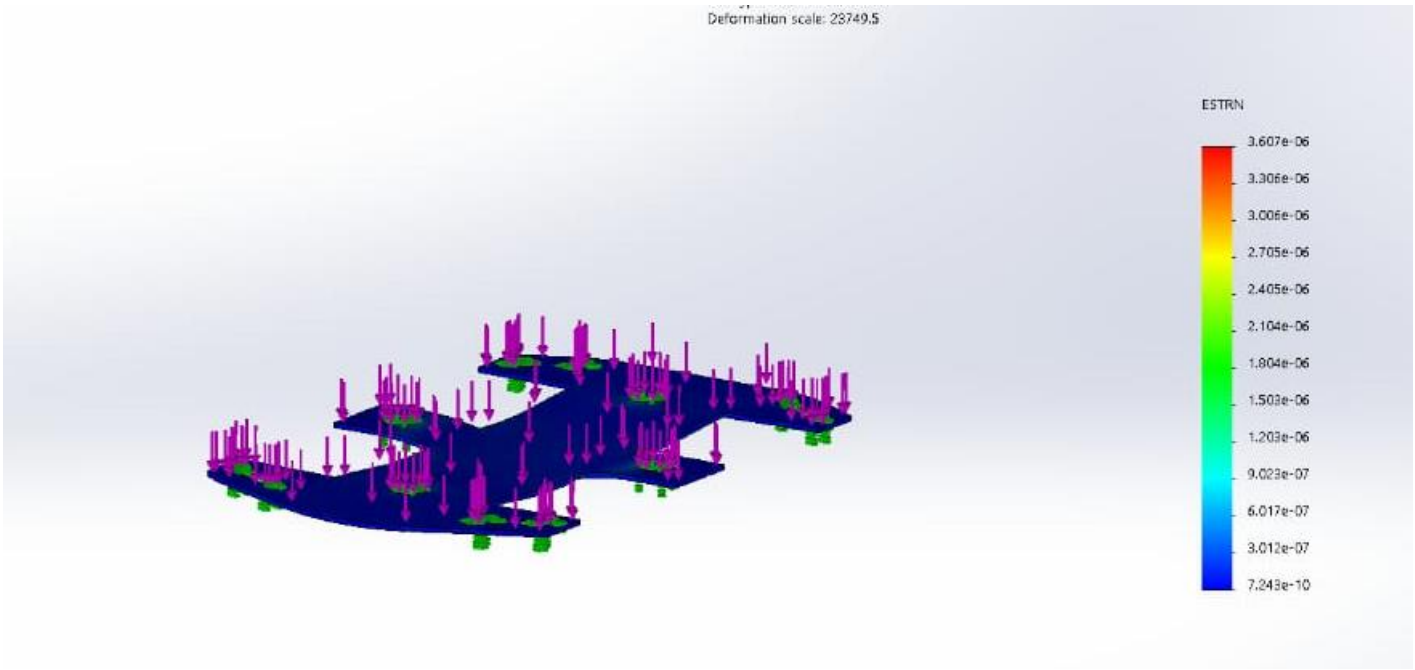



*Figure 6.6*

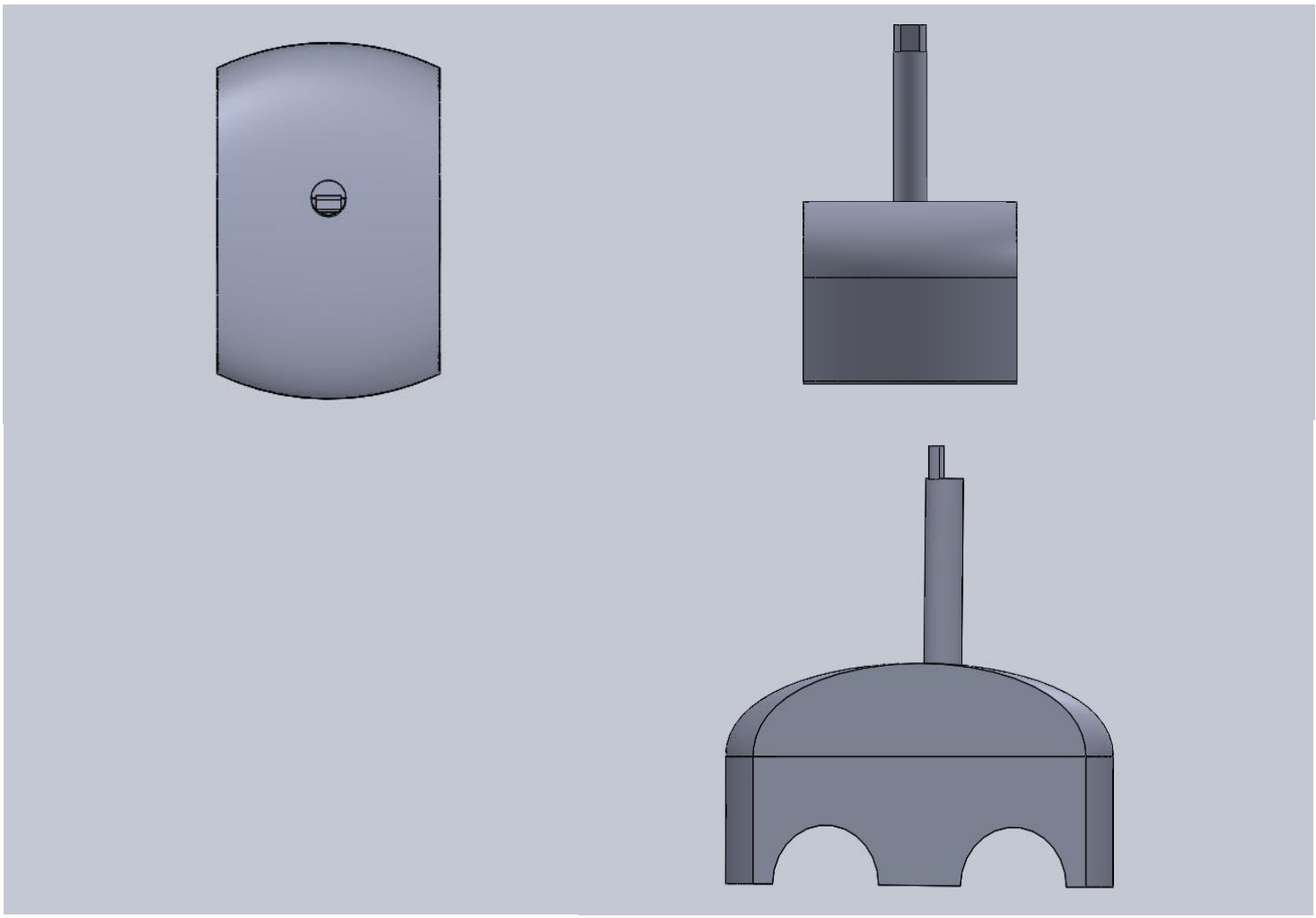


*Figure 6.7*

## 3. PCB Design

To design the PCB board, we use a ki cad software to make on it the schematic shape of the board and the layout. our PCB board is being divided to parts and it's a single layer PCB board.
Parts of the PCB:

- The power parts:
  We add all the parts of the power on one side of the board and make the two wires of the batteries (GND, +12V), connected to two Rosetta. And the upper and lower motors connected to another two Rosetta. We make

a parallel jack to connect the charger of the batteries to make it charging without taking off the batteries from the battery holders. And connect the switch to turn or off the motors with another Rosetta.

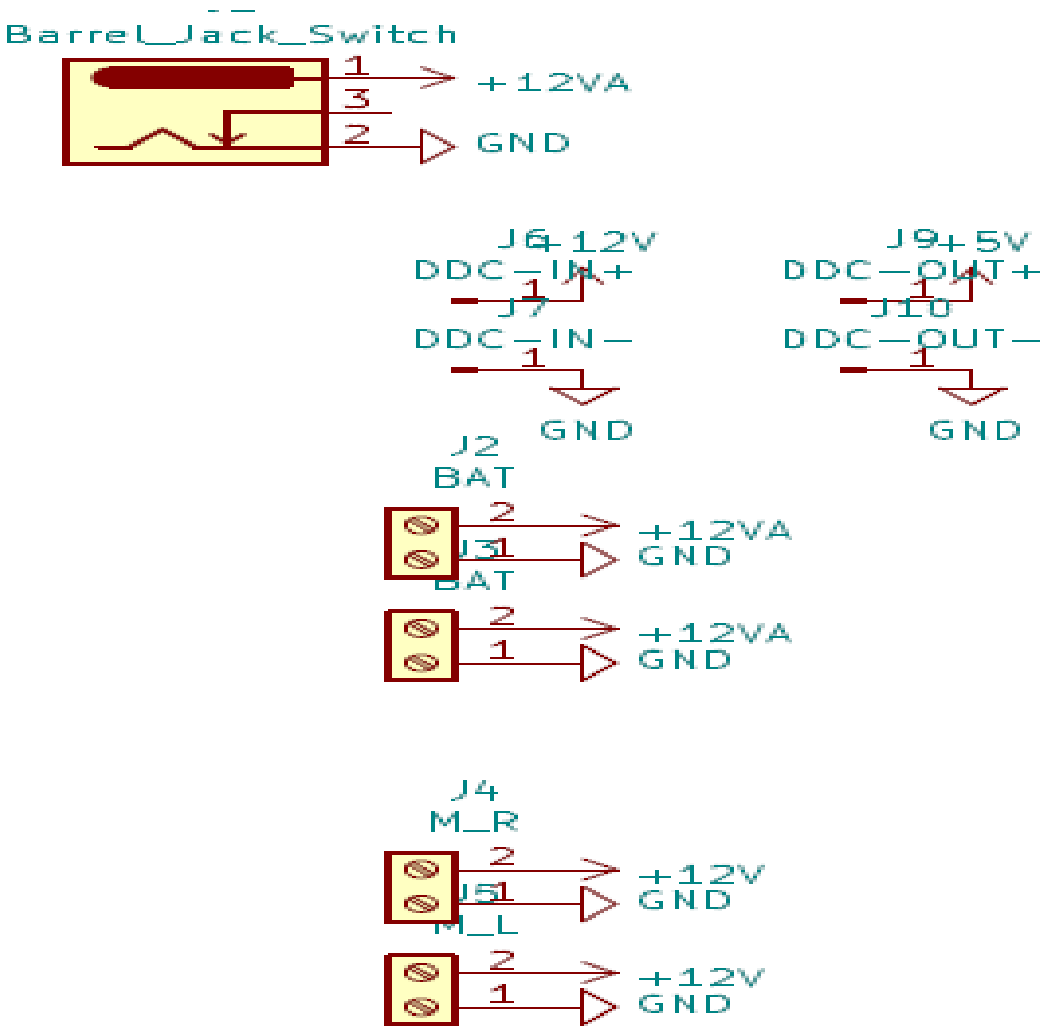



*Figure 6.8*

- The ATmega32 part:
  We use a 16MHZ crystal with the atmega32 chip and connect to it a resistance to lower the voltage. Then we put our atmega32 chip with 4 ports. And connect the reset pin of the MCU with a 10k resistance. The we make a lapel for the pins of the 4 ports as for emergency case.

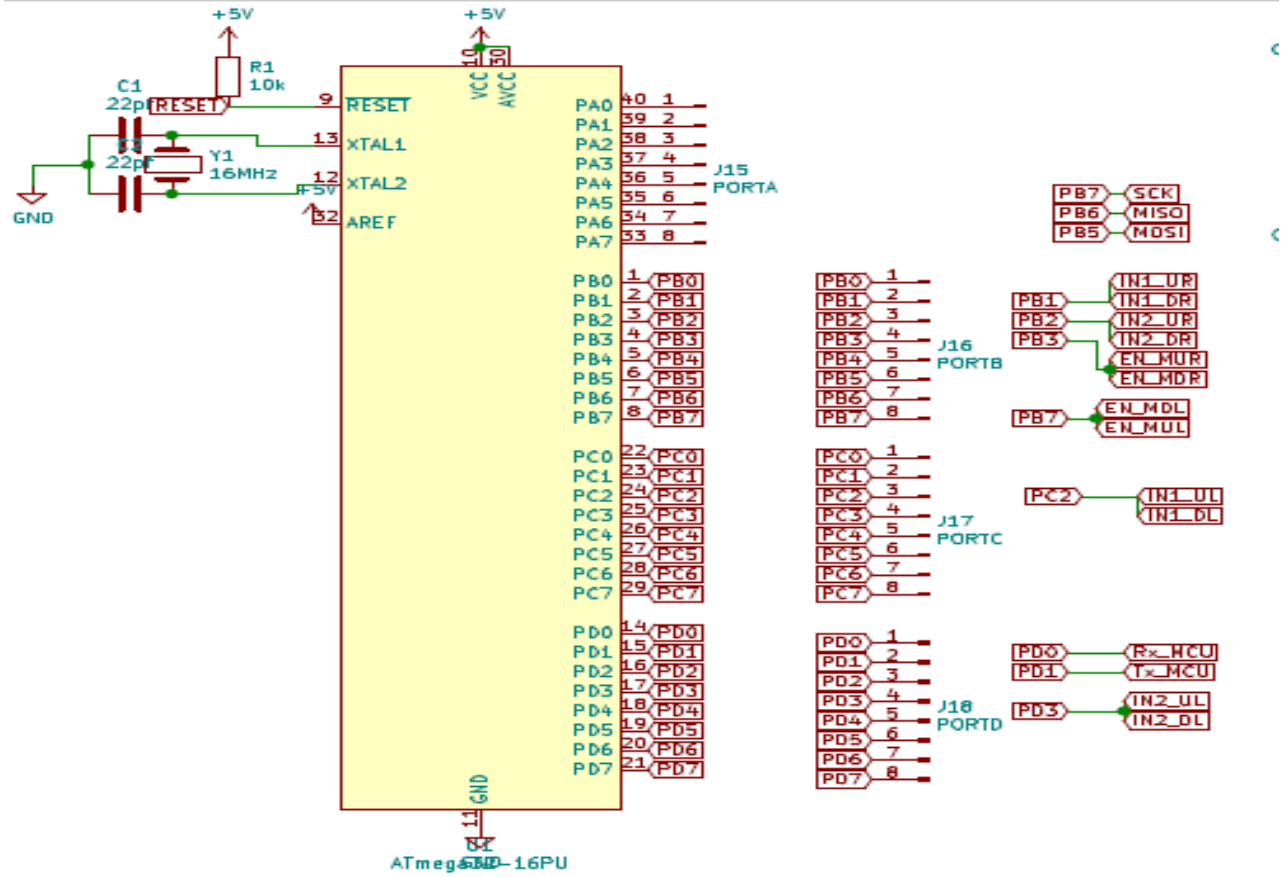



*Figure 6.9*

- The serial part:
  To make the raspberry pi interface with the atmega32 we have to use a communication protocol which as USART. But the problem is that the raspberry pi receives a 3.3V and the atmega32 transmit a signal with 5V, so we make a voltage divider using two resistance of 330 ohm and on resistance of 1K ohm to make the voltage drop from 5V to 3.3 V.

- The layout of the PCB:
  We faced a problem with the tracers of GND and 5V because our board is a single layer, so we make a via to overcome this problem.

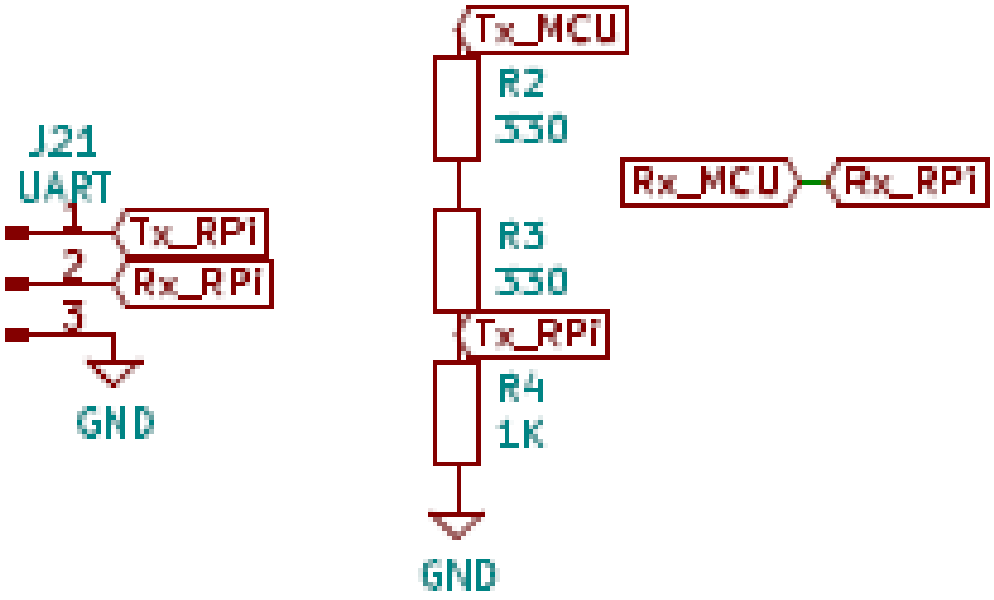



*Figure 6.10*

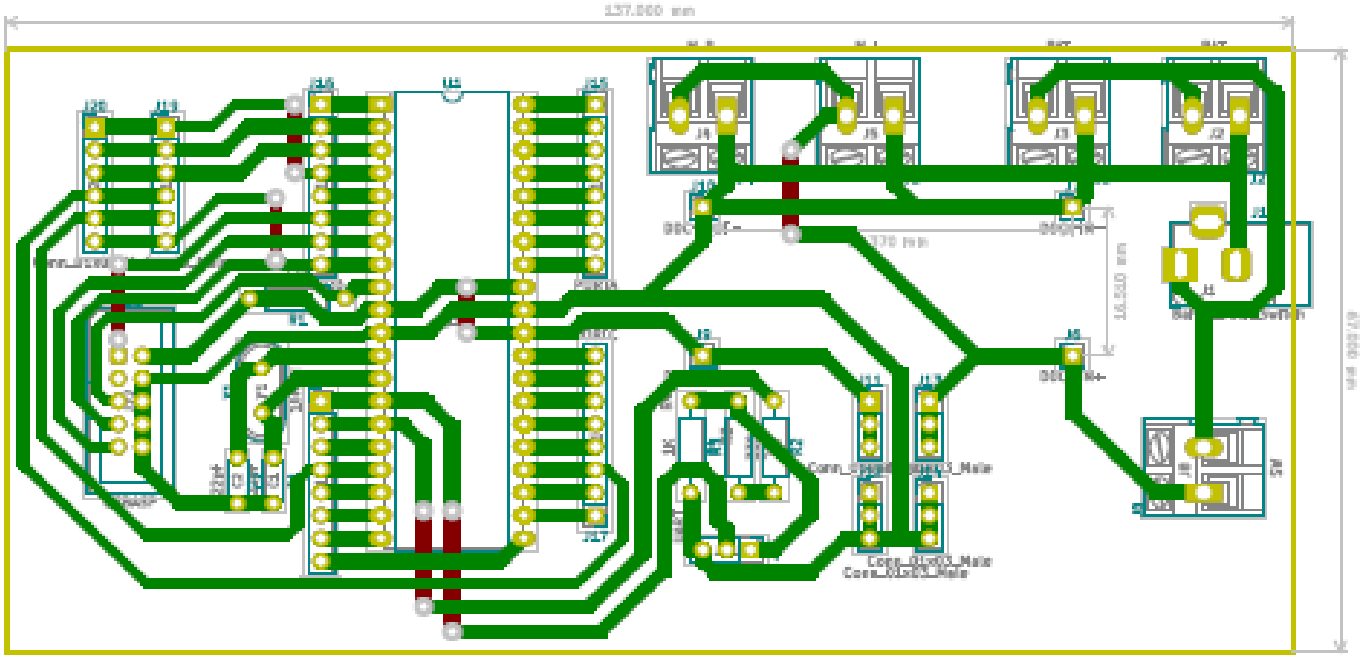


*Figure 6.11*

Final Shape of PCB

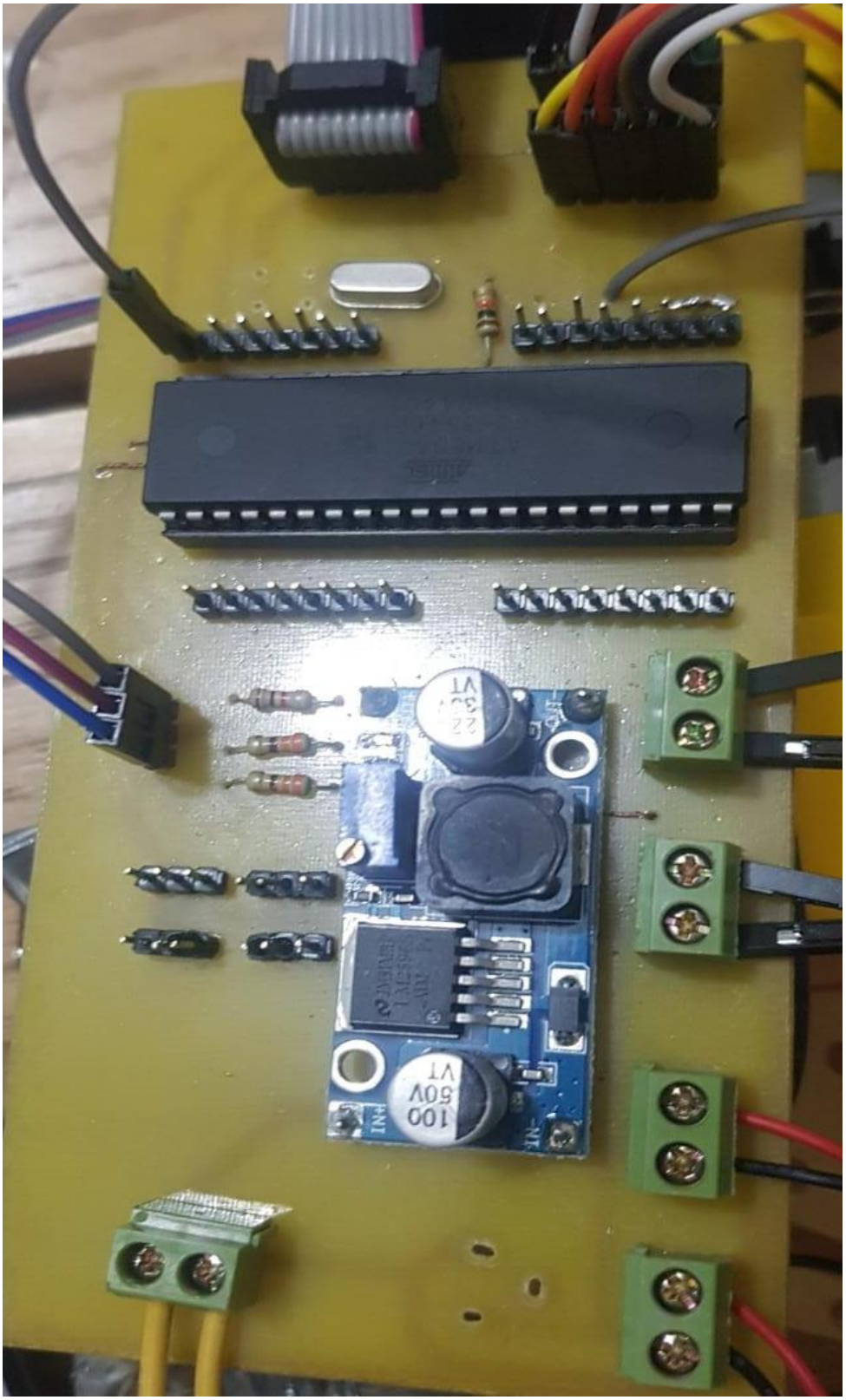



*Figure 6.12*

# CHAPTER 7

## 1. Flowchart of Integration

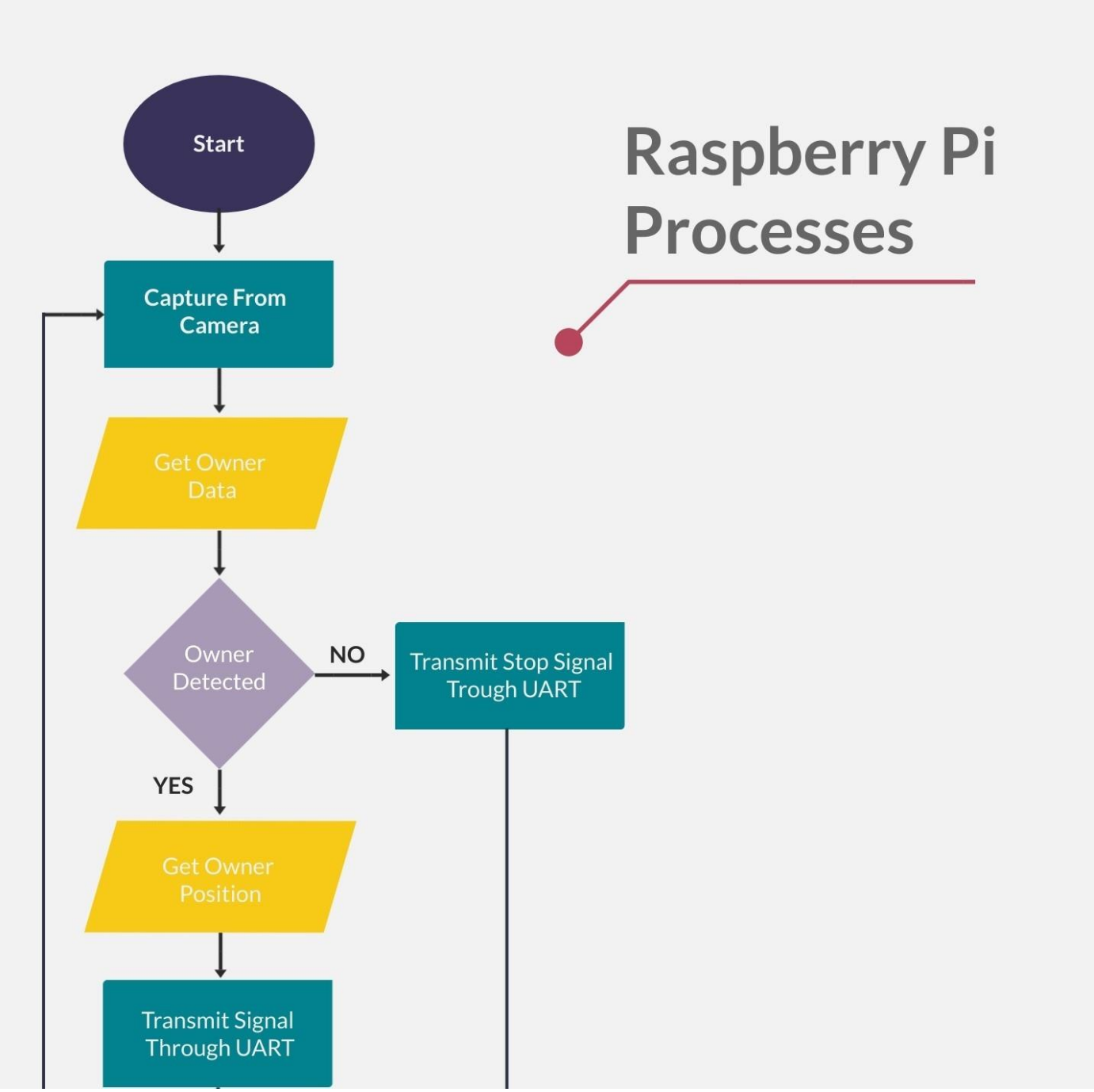



*Figure 7.1*

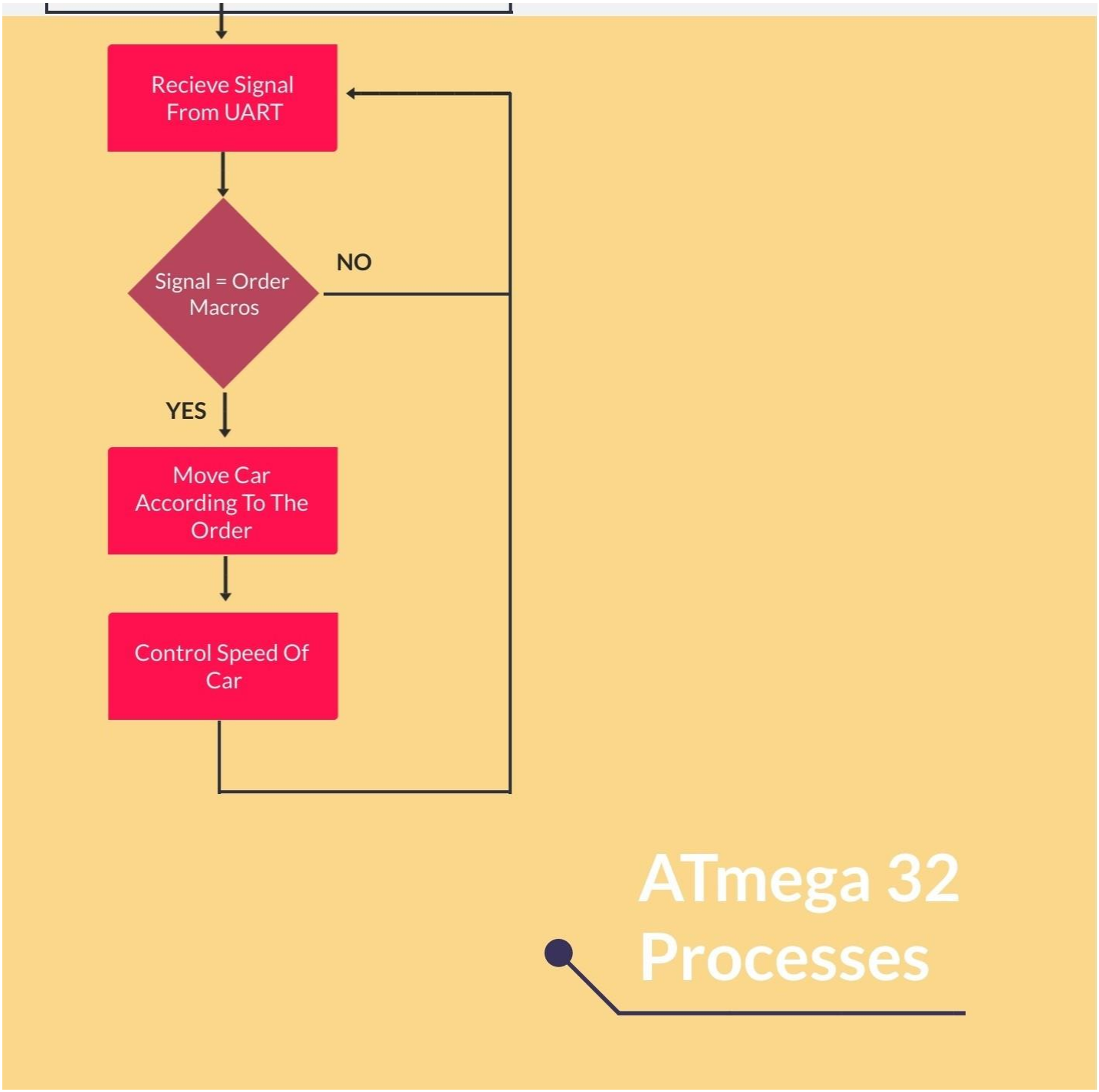

Recieve Signal From UART
Signal = Order Macros
NO
YES
Move Car According To The Order
Control Speed Of Car
ATmega 32
Processes

*Figure 7.2*

## 2. Communication Protocol

We are using USART as a serial communication between raspberry pi and the atmega32 MCU.

### 2.1 UART

USART (The Universal Synchronous and Asynchronous serial Receiver and Transmitter) Features:

is a highly flexible serial communication device. The main features are:

- Full Duplex Operation (Independent Serial Receive and Transmit Registers)
- Asynchronous or Synchronous Operation
- Master or Slave Clocked Synchronous Operation
- High Resolution Baud Rate Generator
- Supports Serial Frames with 5, 6, 7, 8, or 9 Data Bits and 1 or 2 Stop Bits
- Odd or Even Parity Generation and Parity Check Supported by Hardware
- Data Over Run Detection
- Framing Error Detection
- Noise Filtering Includes False Start Bit Detection and Digital Low Pass Filter

- Three Separate Interrupts on TX Complete, TX Data Register Empty, and RX Complete
- Multi-processor Communication Mode
- Double Speed Asynchronous Communication Mode.

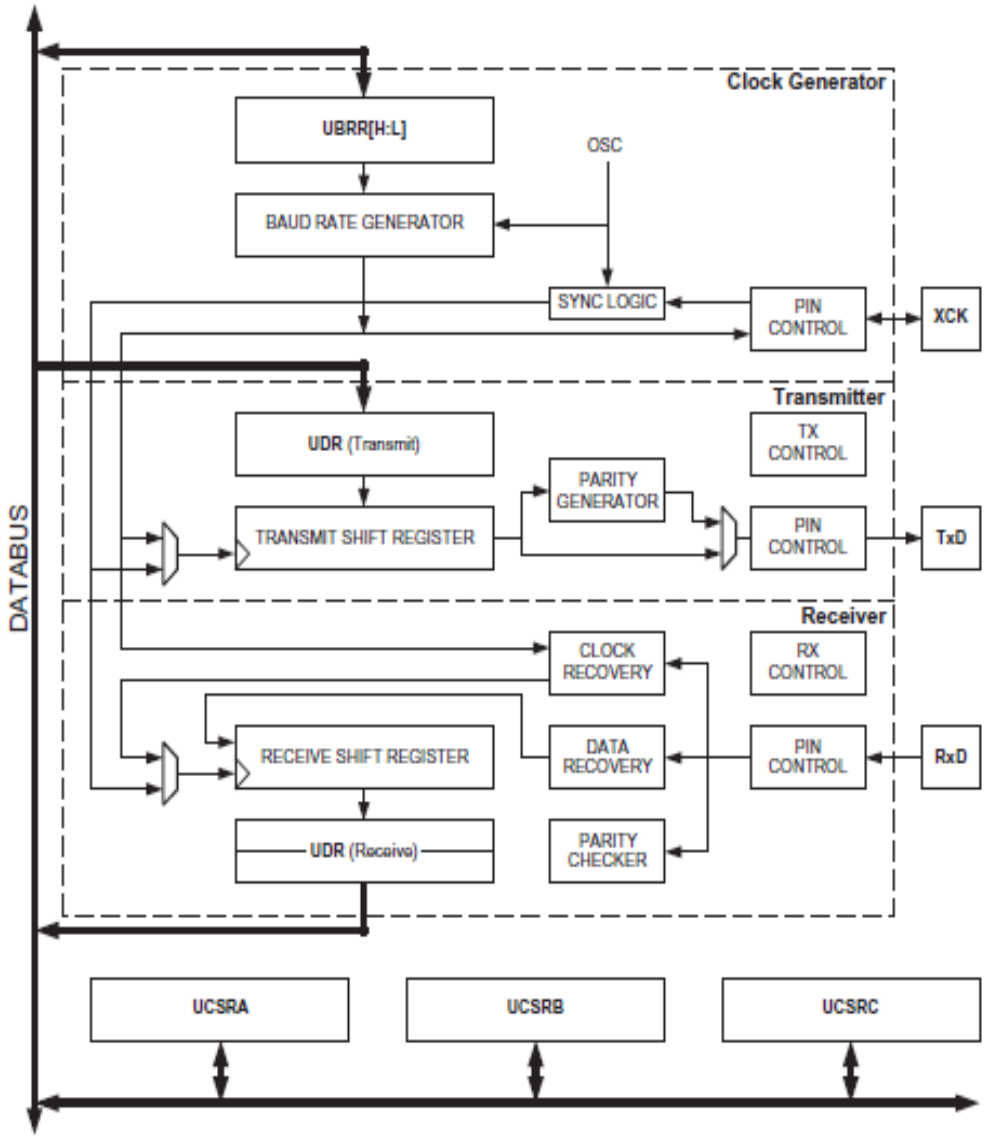



*Figure 7.3*

The dashed boxes in the block diagram separate the three main parts of the USART (listed from the top): Clock Generator, Transmitter and Receiver. Control Registers are shared by all units. The clock generation logic consists of synchronization logic for external clock input used by synchronous slave operation, and the baud rate generator. The XCK (Transfer Clock) pin is only used by Synchronous Transfer mode. The Transmitter consists of a single write buffer, a serial Shift Register, parity generator and control logic for handling different serial frame formats. The write buffer allows a continuous transfer of data without any delay between frames. The Receiver is the most complex part of the USART module due to its clock and data recovery units. The recovery units are used for asynchronous data reception. In addition to the recovery units, the receiver includes a parity checker, control logic, a Shift Register and a two level receive buffer (UDR). The receiver supports the same frame formats as the transmitter, and can detect frame error, data overrun and parity errors.

AVR USART vs. AVR UART – Compatibility:

- Bit locations inside all USART Registers
- Baud Rate Generation
- Transmitter Operation
- Transmit Buffer Functionality
- Receiver Operation.

We are connecting the RX pin of the atmega32 to the TX pin of the raspberry pi, and The TX pin of the atmega32 to RX pin of the raspberry pi, where the raspberry pi transmits a macro of the directions of the car and the atmega32 receive this signal and start to execute this function to make the car move towards the direction of human.

## 2.2 Transmission From RPi

Raspberry pi transmit data to ATmega32 through UART protocol and this how we implement it:

Import some packages:

```
import enum
import serial
from time import sleep
```

First, we need to determine location of transmission and baud rate of it

```
transmitter_location = serial.Serial(port="/dev/serial0"
, baudrate=9600)
```

Second, give it a chance to warm up by calling sleep library:

```
sleep(0.05)
```

third, we transmit direction or order data:

```
transmitter_location.write(order.value)
sleep(0.05)
```

at last, we transmit speed data:

```
transmitter_location.write(speed.value)
Print data that will be transmitted to ATmega32
```

Finally, we print transmitted data:

```
print("Order: ", order.value, " Speed: ", speed.value)
```

## 2.3 Receive From ATmega32

```
unsigned char UART_RxChar()
{
     /* Wait till data is received */
     while ((UCSRA & (1 << RXC)) == 0);
     return(UDR);        /* Return the byte */
}
```

# CONCLUSION AND FUTURE PERSPECTIVES

This project describes a way to make an autonomous robot follow a human person under specific constraints as a minimal CPU consumption.

Some applications could be a mobile robot in a specific environment as an o-ce, a hospital or in people home.

To use this robot in an environment as people home, future work could be to handle multiple people. For the moment, the robot is only able to follow one person chosen randomly. It takes the first person detected.

If another person come and cross the person followed there is a risk that the robot modifies its target and follow the other user. Things could be added to choose the person in order to keep all the time the same person. Concerning mapping and navigation, in this project, the first step is to create a map of the environment using a mapping algorithm.

To create the map, the robot moves thanks to keyboard teleoperation. An exploration algorithm could be added to the project so that the robot explores its environment autonomously to create the map.

Moreover, for mapping, 3D maps created thanks to the RGB-D sensor could be studied.

The advantages of 3D maps is that these maps contain more information about the environment which could be used for many applications, for example identify and put a label on each place of the house, and making an application on smart phone connected with the car when receive a signal from the bush button that someone want to take the car or stall it the camera of the raspberry pi take a photo of his face and send it to your smart phone.

# REFERENCES


[1]"Getting Started with Atmel Studio 7", *Atmel-studio-doc.s3-website-us-east-1.amazonaws.com*, 2021. [Online]. Available: http://atmel-studio-doc.s3-website-us-east-1.amazonaws.com/webhelp/GUID-54E8AE06-C4C4-430C-B316-1C19714D122B-en-US-1/index.html?GUID-8F63ECC8-08B9-4CCD-85EF-88D30AC06499&fbclid=IwAR2INVoijJo8gcbU6X3yyK9ixW4ZX0kpiNIYsi8qd61inf1RSObDbyHMVKU. [Accessed: 10- Jul- 2021].

[2]"USBasp - USB programmer for Atmel AVR controllers - fischl.de", *Fischl.de*, 2021. [Online]. Available: https://www.fischl.de/usbasp/?fbclid=IwAR2Jj7sbz6J8i9UDIZDIeMi_0MJzTMsdDqA3nUTTmeLIE8P8tP1x9FQs0c4. [Accessed: 10- Jul- 2021].

[3]"In-Depth: Control DC Motors with L293D Motor Driver IC & Arduino", *Last Minute Engineers*, 2021. [Online]. Available: https://lastminuteengineers.com/l293d-dc-motor-arduino-tutorial/?fbclid=IwAR3pLMXAmX8eJ5IhL3xgGRnNvr4xld1UcpkzqWe73hcQpJOcRnUpO20OvpI. [Accessed: 10- Jul- 2021].

[4]"What is Computer Vision?", *Ibm.com*, 2021. [Online]. Available: https://www.ibm.com/topics/computer-vision#:~:text=Computer%20vision%20is%20a%20field,recommendations%20based%20on%20that%20information. [Accessed: 10- Jul- 2021].

[5]I. Education, "What is Machine Learning?", *Ibm.com*, 2021. [Online]. Available: https://www.ibm.com/cloud/learn/machine-learning. [Accessed: 10- Jul- 2021].

[6]"What Is a Self-Driving Car?", *Car and Driver*, 2021. [Online]. Available: https://www.caranddriver.com/research/a31996016/what-is-a-self-driving-car/. [Accessed: 10- Jul- 2021].

[7]"The five levels of automation in self-driving cars", *Linkedin.com*, 2021. [Online]. Available: https://www.linkedin.com/pulse/five-levels-automation-self-driving-cars-naveen-joshi/. [Accessed: 10- Jul- 2021].

[8]"Top 8 Raspberry Pi Alternatives Available in 2021 | upGrad blog", *upGrad blog*, 2021. [Online]. Available: https://www.upgrad.com/blog/raspberry-pi-alternatives/. [Accessed: 10- Jul- 2021].

[9]"Pololu - Raspberry Pi Model B, Revision 2.0", *Pololu.com*, 2021. [Online]. Available: https://www.pololu.com/product/2750. [Accessed: 10- Jul- 2021].

[10]"How to install NOOBS on the Raspberry Pi - The Pi", *The Pi*, 2021. [Online]. Available: https://thepi.io/how-to-install-noobs-on-the-raspberry-pi/. [Accessed: 10- Jul- 2021].

[11]P. home, c. support and </form>, "How to Enable SSH on Raspberry Pi [Definitive Guide]", *Knowledge Base by phoenixNAP*, 2021. [Online]. Available: https://phoenixnap.com/kb/enable-ssh-raspberry-pi. [Accessed: 10- Jul- 2021].

[12]"VNC (Virtual Network Computing) - Raspberry Pi Documentation", *Raspberrypi.org*, 2021. [Online]. Available: https://www.raspberrypi.org/documentation/remote-access/vnc/. [Accessed: 10- Jul- 2021].

[13]"Camera configuration - Raspberry Pi Documentation", *Raspberrypi.org*, 2021. [Online]. Available: https://www.raspberrypi.org/documentation/configuration/camera.md. [Accessed: 10- Jul- 2021].

[14]"How to Install Python on Windows [Pycharm IDE]", *Guru99.com*, 2021. [Online]. Available: https://www.guru99.com/how-to-install-python.html#:~:text=Step%201)%20To%20download%20PyCharm,setup%20wizard%20should%20have%20started. [Accessed: 10- Jul- 2021].

[15]"How to install Sublime Text 3 in Windows? - GeeksforGeeks", *GeeksforGeeks*, 2021. [Online]. Available: https://www.geeksforgeeks.org/how-to-install-sublime-text-3-in-windows/#:~:text=Step%201%3A%20Open%20the%20downloaded,begin%20with%20the%20installation%20process.&text=Step%203%3A%20

If%20you%20want,Finish%20with%20the%20installation%20process. [Accessed: 10- Jul- 2021].

[16]"How to Install PuTTY SSH for Windows - Information Technology Services", *Information Technology Services*, 2021. [Online]. Available: https://its.gmu.edu/knowledge-base/how-to-install-putty-ssh-for-windows/. [Accessed: 10- Jul- 2021].

[17]"How to Use RealVNC: 8 Steps (with Pictures) - wikiHow", *Wikihow.com*, 2021. [Online]. Available: https://www.wikihow.com/Use-RealVNC. [Accessed: 10- Jul- 2021].

[18]D. Stammer, "Flow charts", *Nature*, vol. 408, no. 6809, pp. 153-153, 2000. Available: 10.1038/35041669.

[19]"Pass by reference vs value in Python - GeeksforGeeks", *GeeksforGeeks*, 2021. [Online]. Available: https://www.geeksforgeeks.org/pass-by-reference-vs-value-in-python/#:~:text=In%20pass%20by%20reference%20the,passed%20into%20the%20function%20directly.&text=To%20summarise%2C%20in%20pass%20by,to%20it%20by%20the%20caller. [Accessed: 10- Jul- 2021].

[20]C. Cheng, *Python*. Walker Books Australia, 2021.

[21]*8-bit AVR with 8k bytes in-system programmable flash*. San José, CA: Atmel, 2002.

[22]P. N R, "Smart pi cam based Internet of things for motion detection using Raspberry pi", *International Journal Of Engineering And Computer Science*, 2016. Available: 10.18535/ijecs/v5i5.15.

[23]F. Manganiello, *Computer Vision with Maker Tech*. .

[24]Y. Park, "Development of a guitar teaching Program using the ATmega 128", *Journal of the Korea Academia-Industrial cooperation Society*, vol. 13, no. 12, pp. 6001-6005, 2012. Available: 10.5762/kais.2012.13.12.6001.