İstanbul
Bilgi Üniversitesi

İstanbul Bilgi University

Electrical and Electronics Engineering

SpecialCase Robot

by

116202035 ÖZGE HÜLYA DURGUT

116202048 SERRA ECE DOĞAN

117202048 EGE ÖZMEN

116202026 BUSE ÇEVİK

Supervised by

Prof. Dr. M. N. Alpaslan Parlakçı

A report submitted for EEEN 492 Senior Design Project II

class in partial fulfillment of the requirements for the degree

of Bachelor of Science

(Department of Electrical and Electronics

Engineering) in Istanbul Bilgi University

June 15th, 2021

ACKNOWLEDGMENTS

Foremost, we would like to express our sincere gratitude to our advisor/rector Prof. Dr. M. N. Alpaslan Parlakçı for their continued support of our research, for their patience, motivation, and immense knowledge. Besides our advisor, we would also like to thank Dr. Mehmet Ayyıldız, Dr. Yeşim Öniz for their support, and Özgür Özdemir for his assistance. We would like to thank Dr. Okan Zafer Batur for his help with our writing process of the report. At last but not least, we would like to express our special thanks to the İstanbul Bilgi University and its staff for providing us such an environment to develop our project.

ABSTRACT

As the society living today is a consumer society, its needs are also increasing rapidly. With these increasing needs, many new areas have emerged where manpower cannot keep up. The work done to meet these needs is increasing every day and making life easier. One of the newly developed areas that have been set out to facilitate human life is the field of robotics. People are now resorting to the use of robots in places that require intensive labor, rather than endangering life safety, or where they think human life can be endangered. Robots are infrastructure systems that will determine their mobility according to the program applied to them. They operate in all kinds of different sectors.

The robot that will be made in the project is designed as a pandemic robot that can move on four wheels, simultaneously have capabilities such as face and mask detection, temperature measurement, distance measurement, and can also perform controls at the necessary times in the area where it is located. In this way, it is aimed that academicians, students and visitors can walk around the campus safely.

ÖZET

Bugün yaşayan toplum bir tüketim toplumu olduğu için ihtiyaçları da hızla artmaktadır. Artan bu ihtiyaçlarla birlikte insan gücünün yetişemeyeceği birçok yeni alan ortaya çıkmıştır. Bu ihtiyaçların karşılanması için yapılan çalışmalar her geçen gün artmakta ve hayatı kolaylaştırmaktadır. İnsan hayatını kolaylaştırmak için yola çıkan yeni geliştirilen alanlardan biri de robotik alanıdır. İnsanlar artık can güvenliğini tehlikeye atmaktan ziyade yoğun emek gerektiren yerlerde ya da insan hayatının tehlikeye girebileceğini düşündükleri yerlerde robot kullanımına başvuruyorlar. Robotlar, kendilerine uygulanan programa göre hareketliliklerini belirleyecek altyapı sistemleridir. Her türlü farklı sektörde faaliyet gösteriyorlar.

Projede yapılacak olan robot, dört tekerlek üzerinde hareket edebilen, aynı anda yüz ve maske algılama, sıcaklık ölçümü, mesafe ölçümü gibi yeteneklere sahip, aynı zamanda bölgede gerekli zamanlarda kontroller yapabilen pandemi robotu olarak tasarlandı. Bu sayede akademisyenlerin, öğrencilerin ve ziyaretçilerin kampüs içinde güvenli bir şekilde gezebilmeleri amaçlanmaktadır.

LIST OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF APPENDICES

LIST OF SYMBOLS/ABBREVIATIONS

ADC	Analog to Digital Converter
I/O	Input/Output
A*	A Star

1. INTRODUCTION

The robot is a virtual or mechanical man-made tool. In applications, the robot is usually an electromechanical device that gives the impression that it has its own unique will and decision-making ability with its movement and appearance. Although the word robot is used for both mechanical devices and software tools in virtual environments, when it comes to robots, electromechanical devices that usually have robot qualities come to mind. For a robot to return to an electromechanical device, it must have several or all of the following qualities; to be able to move, operate a mechanical arm, perceive its environment and use it following its requirements, to perform human or animal movements and to exhibit conceivable behaviors, etc.

The Covid-19 virus, which started in Wuhan, China, and affected the world, has caused millions of people worldwide to get sick and more than a million people to die. Globally, too many students have been unable to study in their schools, production has stalled in factories, workplaces have been unable to open and even people have been banned from going out. In trying to get rid of such a virus, certain measures have been taken all over the world in order to open schools, factories, workplaces, and return to everyday life. These measures are implemented to protect people's health. For example, wearing a mask, putting social distance between people, taking temperature measurements in different places, and constantly disinfecting the hands are just some of these applications. The aim of the project is to produce a robot that can provide warning and control for measures without endangering the lives of the people assigned to provide control of Covid-19 measures in places such as schools, workplaces, campuses, factories, shopping malls. There are many studies on robot design and designs for different purposes and shapes. While the purpose of such studies often depends on the areas where the robot will be used specifically, the common goal in all of them is to facilitate people's work. If it is necessary to evaluate projects with similar purposes to this project, Mitra Robot, a robot that measures fire and distance, is a humanoid robot designed and developed by the Indian company Invento Robotics. Fortis Hospital is using the Mitra robot for Covid-19 screening [1]. Checks are carried out on every visitor entering the hospital, including doctors, nurses, staff and patients. Autonomous navigation was used for the robot's facial recognition and speech features. It scans the visitor for symptoms of Covid- 19, namely fever, cough and colds. In this way, health workers will be able to be protected from

coronavirus, which has a very high contagion. In addition, studies similar to the obstacle detection and mapping features planned in this project have been carried out before. For example; Robots that can detect the obstacle information and find its way through the maze under the guidance of an appropriate algorithm and reach the exit or move from a certain location to another predetermined location without hitting the obstacles were made by others in previous projects. Another example, the M1 robot from NAVER LABS is a closed 3D/HD mapping robot that navigates independently indoors. The M1 robot automatically collects 3D spatial data via high-performance cameras [2]. The resulting HD maps provide the spatial data required for location-based services such as AR walking navigation and indoor autonomous service robots. For another feature, the Pepper robot from SoftBank Robotics is able to both control distance measurement between humans and check whether there are masks on. As people think, robots are not always human figures like in science fiction movies. In this way, it is aimed to minimize the rate at which people infect each other with Covid-19 virus in certain regions. General features of our robot are very important features such as temperature measurement, face detection, mapping, distance measurement, and avoiding obstacles. With the robot, the health check of the students on our campus can be carried out more often, without cost and without danger. In this way, a robot that is both useful and functional will be produced.

1.1. Robot Guarding the Campus Toll Gate against COVID-19 Virus

The robot is designed to prevent the coronavirus epidemic and will work at the campus gate. The robot will actively move and control people within a 10 x 10 meters area, including the campus toll booths. The robot will follow the person entering the campus from the toll booths and measure their body temperature using thermal cameras and sensors, the robot will warn people whose body temperature is dangerous. Social distance rules were introduced to prevent the spread of the virus. By using distance sensors, the social distance of people at the campus entrance can be measured, and people who are closer than 1.5 meters will be warned by the robot to follow the social distance rules. One of the most important points in combating the virus is wearing a mask. People need to wear masks as well . Using Python image processing, it can be checked whether people are wearing their masks properly, and people who do not wear masks will be warned to wear masks. These particular functions will use machine learning systems and deep learning algorithms.

1.2. What is a Robot?

Robots are machines built with the help of sensors, motors, circuits and software to perform human actions. As people think, robots are not always human figures like in science fiction movies. For example, from our daily lives, it is possible to describe even the blender device, where It can crush fruits, as a robot. Most of the robot designs currently used have a very different appearance. Most vehicles with systems designed in certain autonomies, written and designed in a certain order can be considered as robots. Robots can be used in different areas and sectors according to their structural purposes. [3]

1.3. Machine Learning

Machine learning is a sub-branch of computer science that was developed from the studies of numerical learning and model recognition in artificial intelligence in 1959. Machine learning is a system that can learn as a structural function and investigate the work and construction of algorithms that can make predictions over data. These types of algorithms work by building a model to make data-based predictions and decisions from sample inputs

rather than strictly following static program instructions. So machine learning is the idea of creating generic algorithms that can tell you interesting things about a particular dataset without having to write code. Instead of writing code, you feed this general algorithm with data, and that way the algorithm builds its logic based on that data.
For example, the classification algorithm is one of these algorithm types. The classification algorithm divides the data into different groups. The same classification algorithm can be used to identify handwritten numbers or to separate emails into spam or non-spam without changing a line of code. The algorithm is the same algorithm, but it creates a different classification logic because it is trained with different data. [4]

1.4. Deep Learning

The field of artificial intelligence encompasses machine learning, where machines can learn through experience and gain skills without human involvement. Deep learning, on the other hand, is a subset of machine learning in which artificial neural networks and algorithms inspired by the human brain learn from data. Similar to what people learn from their experience, the deep learning algorithm does a better job of making some changes each time to improve the result. Deep learning can realize the solution about any problem that requires thought. [4]

1.5. Euclidean Distance

Mathematics says that the Euclidean distance between two points in a Euclidean space gives you the line segment between the two points. This can also be calculated with the Cartesian coordinates of the points applying Pythagorean theorem, It is occasionally called the Pythagorean distance. Those two names come from the ancient Greek mathematician and philosopher Euclid and Pythagoras.
The Euclidean distance is used for measuring the length of a segment between two points in either the plane or 3-dimensional space. It is the most common way of showcasing the distance between two points . The distance in between two specified objects which are not exact points from two objects. Distance from a point to a line formulas are now for computing distances between different types of objects. In the subject of advanced maths, this idea of distance has been generalized to abstract metric spaces and other distances that

the Euclidean has studied before. Also in some various applications in statics, the square of the Euclidean distance is used instead of the distance itself. [5]

The Distance Formula

Commonly, especially when measuring the distance in a plane, the formula for the Euclidean distance is being used. According to the distance formula by Euclide [5], the distance between the two points in a wide plane with coordinates (x,y) and (a,b) is given by

$$\text{dist}((x, y), (a, b)) = \sqrt{[(x - a)^2 + (y - b)^2]}$$

For an example let it be considered the distance between points (2, -1) and (-2, 2):

$$\begin{aligned}\text{dist}((2, -1), (-2, 2)) &= \sqrt{(2 - (-2))^2 + ((-1) - 2)^2} \\ &= \sqrt{(2 + 2)^2 + (-1 - 2)^2} \\ &= \sqrt{(4)^2 + (-3)^2} \\ &= \sqrt{16 + 9} \\ &= \sqrt{25} \\ &= 5.\end{aligned}$$

Figure 1. Calculation Formula

Another way of displaying the formula is given below [6]:

$$\text{dist}((x, y), (a, b)) = |x - a| + |y - b|$$

3D Space

Let us consider that if p and q are points of R3, the Euclidean distance from p to q is the number.

$d(p, q)=\|p-q\|$ (Is a way showcasing the formula)

And since it can considered that

$$p-q=(p1-q1,p2-q2,p3-q3)$$

That expansion of the norm gives the graph below:

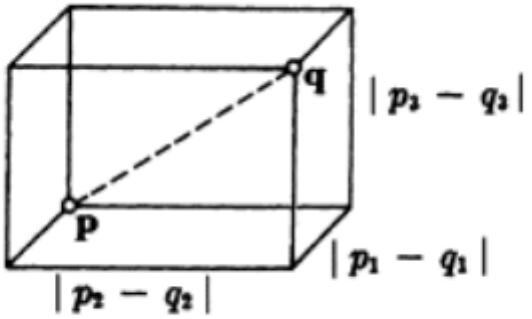
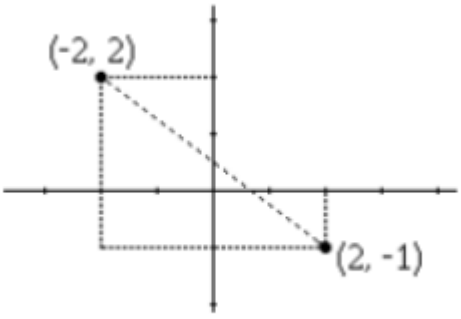


Figure 2. Distance Calculation in 3D Space

The Distance Formula is:

d(p,q)=((p1-q1)^2+ (p2-q2)^2+(p3-q3)^2)^(1/2)

All of the points above are almost enough to a point of an open set are also included in the set. This given definition is only valid with R3 replaced by Rn or I any set confirmed with a reasonable distance function. The Horizontal longitude is given 4 and the vertical longitude is given 3. Let it be introduced that one more point is (-2,-1). With a small addition a slightly right angled triangle with legs 3 and 4 will be obtained. Hence, by applying the Pythagorean theorem, the square of the hypotenuse is = 3^2 + 4^2… which gives 5 as the length of the hypotenuse same as distance between two points according to the distance formula.[6]



1.6. A* Path Finding

It's one of the algorithms used in computer science to find the shortest path. It is a common search algorithm used in topics such as path finding and data visualization. The algorithm

of a grid-the shortest distance between points A and B in the plane, find a way around the obstacles in between the frames around each point to own a scoring system based on a comparison with the scores of other squares, and check with to decide whether or not to continue processing it.

A * algorithm can be classified as an admissible heuristic algorithm. The reason for this is the function that his algorithm uses to calculate distance:

$$f(n) = g(n) + h(n)$$

Equation:

F (n) = Heuristic function that performs computation.

g (n) = Cost of coming from the initial node to the current node

h (n) = The estimated distance from the current node to the destination node.

As it should be noted, the reason why the function f(n) is intuitive is the heuristic function h(N), which is contained in this function and is based on estimation. The Grid / Node plane, which will be mentioned below, is a map, area consisting of squares. Grid / Node, one of the squares in the plan. [7]

Operation of The Algorithm

The algorithm has a fairly simple structure that uses the above addition process. In an algorithm that uses a priority queue as a data structure, the node with the highest priority is the node with the lowest value of f (n). The algorithm works as follows.

1) The point to be processed is determined as the starting point.

2) Define 8 points (up, down, right, left and 4 crosses) around the processed point. 8 grids may vary depending on the requirement.

3) Determined grids are processed.

4) Each point processed is calculated as the above equation and the value f(N) is generated. The scoring system in this step calculates the distance of a grid to the grids to the right, left, top, bottom of the grid as 10 units per grid. For example, if Grid A is to the left of

Grid B 3 units, their distance (if there are 3 more grids between them) is calculated as 3*10=30 points.

The distance of a grid to the grids on the Cross is calculated as 14 points. For example, if Grid C is 2 units across Grid D (if there are 2 more grids between them), the distance of the grids is calculated as 2*14=28 points.

5) According to The found F result, compare the values of F of all frames processed, re-determine the squares with the lowest value of F, and 2 for each. return to my name. When one of the processed grids becomes the destination grid, 5. step must be passed. No grid is reworked if its value does not equal the lowest value.

6) A loop occurs between items 2-5 until the target grid is reached. The algorithm is complete when the destination is reached. A grid defined as an obstacle is never processed.

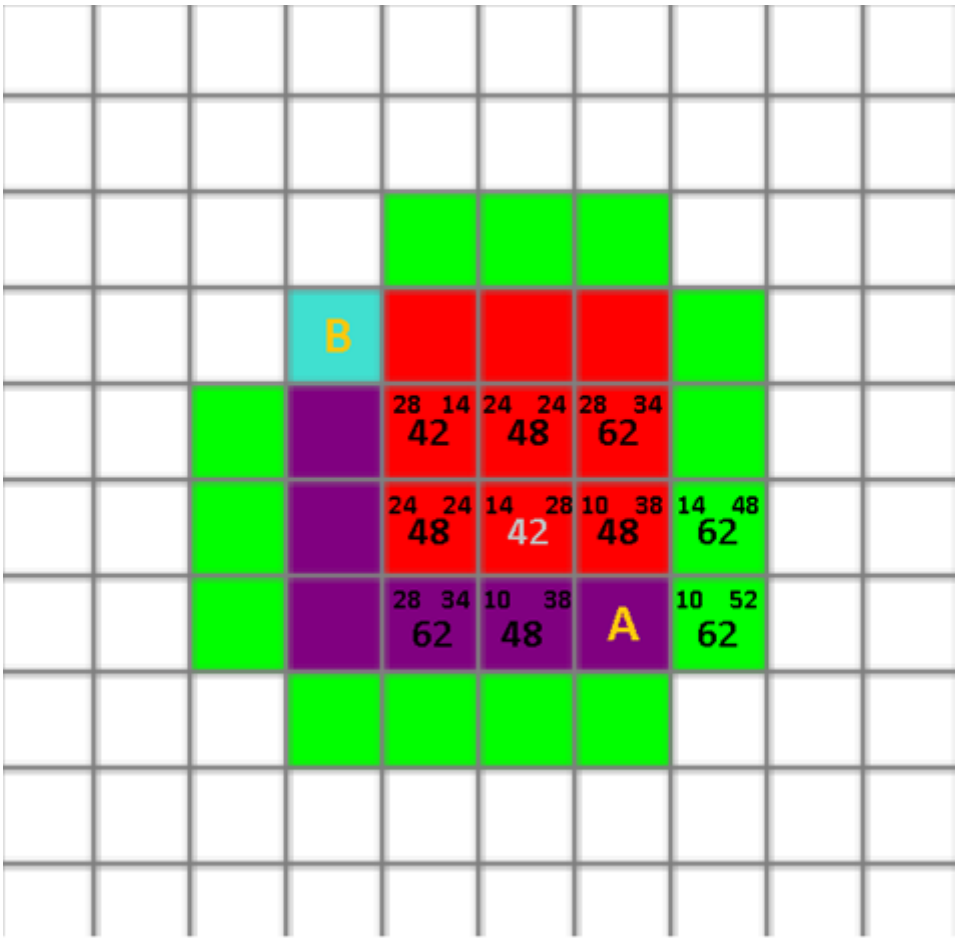


Figure 4. A* Path Finding Algorithm

In the image above, point A is designated as the beginning, point B is designated as the destination.

The distance of the grid to point A, the number written on the left (G),

The number written on the right of the grids is the distance to point B (H),

The number in the middle of the grids is defined as the value F (G+H).

8 grids around Point A are taken into account. A grid written in gray, which has the smallest value of F from 8 grids, was considered the center point because it has the smallest value (42), and this time the 8 grid around it was processed. If there was another grid with a value of 42, the grids around it would also be processed simultaneously. [7]

1.7. Image Processing

Image processing is a method developed to convert an image into a digital form and perform some operations, to obtain a specific image or extract some useful information from it. The input of this method is an image, such as a video cross-section or a photo. The output corresponds to the desired or required part of the image. Generally, the image processing system treats images as two-dimensional signals when applying preset signal Processing methods. [8]

Purpose of Image Processing

The purpose of image processing is divided into 5 groups.

1. Visualization-observing hard-to-see objects

2. Image sharpening and restoration-improve noisy images

3. Image retrieval-engaging and high-resolution image search

4. Pattern Recognition-identifying various objects in an image.

5. Image recognition-distinguish objects in an image.

1.8. Serial Communication

Serial communication means that the signals to be transmitted are sent sequentially over a single line. When the robot is implemented in real life, the hardware elements must communicate serially with each other. It was decided that elements such as Arduino Uno and Raspberry Pi should be used in the project. Raspberry Pi and Arduino can be communicated via serial port with RX and TX pins on both systems. Raspberry Pi works with 3.3V and Arduino with 5V. While communicating both systems, it is possible to solve this problem with a level shift circuit or a simple voltage divider. [9]

1.9. Hardware Environments

1.9.1. Arduino

Arduino is a physical programming platform consisting of an I/O board and a development environment that includes an implementation of the Processing/Wiring language.
The hardware of Arduino boards includes an Atmel AVR microcontroller (such as ATmega328, ATmega2560, ATmega32u4) and necessary peripherals for programming and connection to other circuits. Every Arduino board has at least one 5 volt regulated IC and a 16MHz crystal oscillator (some ceramic resonator). An external programmer is not needed for programming on Arduino boards, because a bootloader program is pre-written to the microcontroller on the board.[10]

To give information about the reason why Arduino is used so much today and about example Arduino Applications:

- Arduino is an open source microcontroller board.
- No microprocessor knowledge required.
- Open source means that the source code is shared with the user and changes are made.
- means that the right is given to the user.
- One of the reasons Arduino is so popular is because of its programming.
- That is easy.
- The Arduino development environment can be downloaded for free from its site.

- Many types and hardware add-ons are available. (Shield)
- The arduino you plug into the computer from the tools menu in the Arduino development environment.
- It's worth checking out the model.
- One of its most important features is its rich library support.

Arduino Uno

It is the basic board of Arduino. It has an ATMega328 microcontroller. It has 14 Digital I/O Pins, 6 PWM Outputs, 6 ADC Inputs. It has 32 KB Flash memory. [10]

Figure 5. Arduino Uno

Arduino Mega

Arduino Mega 2560 has ATMega2560 microcontroller. The Number of Inputs/Outputs is more than Arduino Uno. (54 Digital I/O Pins, 14 PWM Outputs, 16 ADC Inputs) Its memory is also higher than Arduino Uno. (256KB Flash memory)

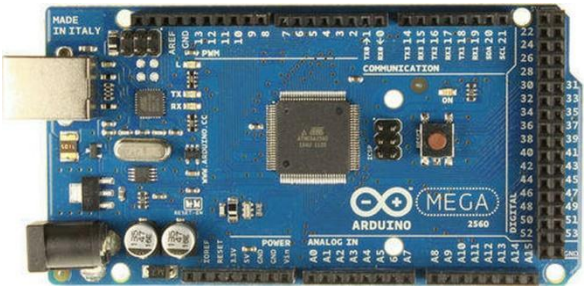


Figure 6. Arduino Mega

After choosing and purchasing the Arduino board suitable for you, you need some materials such as a computer, USB cable, various resistors, leds, buttons, breadboard, connection and jumper cables, various sensors, LCDs.

1.9.1.1. Arduino Components

The basic components of Arduino are: Arduino development environment (IDE), Arduino bootloader (Optiboot), Arduino libraries, AVR Dude (software to program microcontroller on Arduino) and compiler (AVR-GCC).
Arduino software consists of a development environment (IDE) and libraries. The IDE is written in Java and is based on the environment of the language called Processing. Libraries are written in C and C++ languages and include AVR-GCC and AVR Libc. compiled with.
The Optiboot component is the bootloader component of Arduino. This component is the component that enables the programming of the microcontroller on the Arduino boards.
The most important component that makes Arduino so popular is the Arduino libraries, which enable everyone to program without having to have detailed knowledge of the microcontroller. You can find a list of Arduino libraries here. Arduino libraries come with the development environment and are located under the "libraries" folder. By examining the codes, it is possible to see how microcontrollers are programmed and the structure of libraries. Finally, the AVRDude component is used to program the compiled code.

1.9.1.2. What can be done with Arduino?

You can easily program with Arduino libraries. You can receive and process analog and digital signals. Using signals from sensors, you can design robots and systems that interact with the environment. You can create reactions to the outside world, such as movement, sound and light, specific to the project you are designing.

Arduino has various cards and modules designed to produce solutions for different needs. You can develop your projects using these cards and modules.

1.9.1.3. How to Use an Arduino?

Despite all these advantageous features, Arduino is not a tool that you can do all your projects quickly with zero electronics and software knowledge. Using ready-made libraries and examples, you need to learn electronics and software along with Arduino in order not to get bogged down after a certain point.

As a result of the convenience of Arduino cards, you cannot use 100% performance of Atmega microcontrollers on Arduino boards.

Before you start working with Arduino, it will be helpful to learn the basics about Arduino. Then you should choose the Arduino board (Arduino Uno, Arduino Mega 250, Arduino Leonardo... etc) that is suitable for you and get one. All Arduino boards can be programmed in the same way, but different boards have different features and functions. For example, two frequently preferred Arduino boards; It is Arduino Uno and Arduino Mega.

1.9.2. Raspberry Pi 4 Model B

Raspberry Pi is a credit card sized single card computer developed by the Raspberry Pi Foundation in the UK to teach computer science in schools.

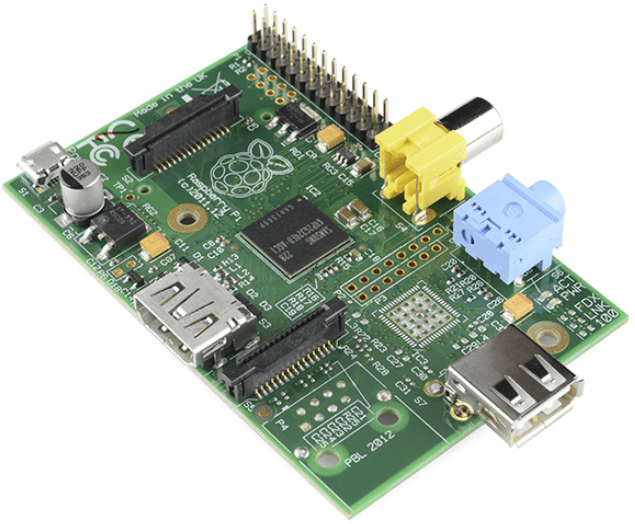


Figure 7. Raspberry Pi

A single-board computer is the name given to all of the small and low-power cards that have the necessary hardware for a computer to work on. These small cards usually have a processor (CPU), a graphics processor (GPU), and RAM, as well as connections (USB, Ethernet) to exchange information with the outside world. It consumes little electricity and does not contain mechanical/moving parts. [11]

1.9.2.1. What is Raspberry Pi?

Working on free software, this mini computer has the power to play even 1080p videos easily. This computer, which carries the operating system on the micro SD card installed on it, is highly portable thanks to its small dimensions. The Raspberry Pi, which works silently because it has no cooling fan or moving parts, meets most of the features expected from a personal computer. It can be used as a normal computer or as a platform to develop your own electronic solutions using the pins on it. [11]

1.9.3. Raspberry Pi Camera Board v2.1

Raspberry Pi Camera V2 is designed as an onboard for Raspberry Pi. This product, which shoots high-resolution photos and videos for projects that require a camera, draws attention with its 8 megapixel resolution and Sony IMX219 image sensor. Compatible with all Raspberry Pi models with a CSI connector, this camera can be directly connected and used with a short ribbon cable.

This camera model, which has a special fixed focus lens, can be used in projects such as motion detection and time-lapse photography, video shooting for security systems. Desired images can be captured with 3280 x 2464 pixel photos and 1080p30, 720p60 and 640x480p90 videos. [12]

Raspberry Pi Camera V2 Features

Fixed focus lens
8 megapixel native resolution sensor - can take 3280 x 2464 pixel photos.
It supports 1080p30, 720p60 and 640x480p90 video.
Size: 5mm x 23mm x 9mm

Weight: Only 3 grams

Raspberry Pi operating system is compatible with the latest version of Raspbian.

It can be directly connected to Raspberry Pi with a short ribbon cable. [12]

1.9.4. Direct Current (DC) Motors

A DC motor is a rotating electrical machine designed to operate with a direct voltage source. It is mostly used in industry. The circular speed of the motor is proportional to the applied voltage and the output torque is proportional to the coil current. Feedback is also used when motion is desired to be precisely controlled. Large motors have stators with coils, while smaller ones have permanent magnet stators (the stationary part). Permanent magnet motors have a multi-coil rotor (rotating part) and these coils are connected to the power supply by means of a commutator. The commutator is a cylinder with copper wires around it. The carbon brushes connect the power supply to the commutator to electrify one coil at a time. The magnetic field produced by the rotor coincides with the magnetic field produced by the stator and the resulting torque rotates the motor. As the rotor rotates, the commutator rotates causing the carbon brushes to feed another coil. In this way, continuous rotational movement is ensured. DC motors can be classified according to their structure and connection types. [13]

1.9.5. L293D and L293B Motor Driver ICs

L293D Motor Driver Integrated L293D and L293B motor driver integrated are 16-legged motor driver integrated with two H bridges. With L293D and L293B motor driver integrated motors, which are generally preferred in DC motor control, two motors can be controlled independently from each other in two directions. In addition, PWM control can be done by using the enabled legs of the L293 motor driver integrated.

L293D motor driver IC can be used in the range of 4.5 V to 36 V up to a maximum current limit of 600 mA. It is possible to use L293B motor driver IC in the same voltage range, up to a maximum current limit of 1 A.

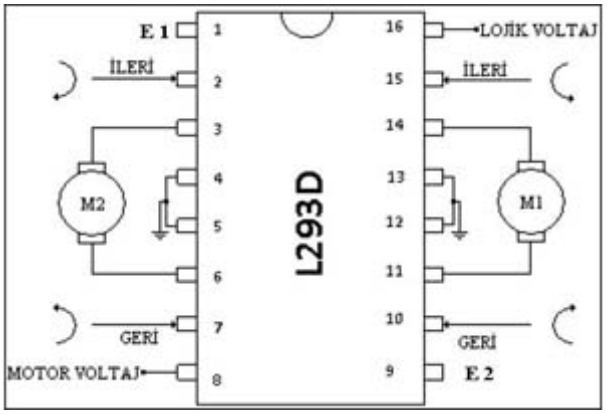


Figure 8. L293D Pin Diagram

The leg structure and connections of the L293 motor driver IC are given above. These features are the same for L293D and L293B. The diagram shows the necessary connections for the two motors to be driven independently of each other in two directions. Here, the logic voltage value is 5 V and the motor voltage is the voltage required to drive the motor used, and this value should not exceed 36 V. [14]

1.9.6. MLX 90614 Sensor

The MLX 90614 is an infrared thermometer for non-contact temperature measurements. Both the infrared sensitive thermocouple detector chip and the signal conditioning chip are integrated into the same TO-46 package. Thanks to its low-noise amplifier, 16-bit ADC and powerful DSP unit, the thermometer's high accuracy and resolution is achieved. The thermometer is calibrated from the factory with the digital interface with SMBus enabled. Reading resolution 0.02°C. [15]

Figure 9. MLX 90164 Sensor

Features:

Model MLX 90614 CJMCU

Temperature range: -70 - 380°C

Accuracy: 0.2°C

Resolution: 0.02°C

Output Type: I2C

Simple adaptation for applications at 8-16V

Size 18 x 11 mm

Usage areas:

High precision non-contact temperature measurements;

Handheld Thermometers

Ear Thermometers

Temperature-controlled household appliances;

Health care;

cattle monitoring;

Multi-zone temperature control - up to 100 sensors can be read via 2 common wires.

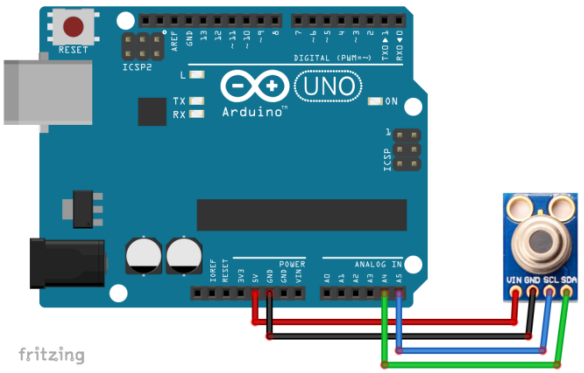


Figure 10. MLX 90164 Sensor Circuit Scheme with Arduino

MLX90614	Arduino Uno
VCC	5V
GND	GND
SCL	A5
SDA	A4

Table 1. MLX 90164 - Arduino Uno Connection Table

1.9.7. OLED Display I2C

OLED; It is the name given to screen technology, which is called organic LED or organic light emitting diode, in which the electroluminescent layer is an organic film that emits light in response to electric current. The OLED display works without a backlight because basically every pixel emits visible light. So it can show deep black levels and is thinner and lighter than a liquid crystal display (LCD) [16]

Figure 11. OLED

0.96 inch 128x64 Oled Display Technical Specifications

No backlight required, the product is self-luminous

High Definition: 128*64

Viewing Angle: > 160°

Widespread platform support: Fully compatible with Arduino, 51 Series, MSP430 Series, STM32/2, CSR IC and so on.

Ultra-low power consumption: Full screen bright: 0.08W

Voltage: 3V ~ 5V DC

Working Temperature: -30 to + 70C

Module Dimensions: 27.0mm * 27.0mm * 4.1mm

I2C/IIC Interface only requires 2 pins.

Driver IC: SSD1306

White color

Pin Wiring

Because the OLED display uses I2C communication protocol, wiring is very simple. You just need to connect to the Arduino Uno I2C pins as shown in the table below.

OLED	Arduino Uno
Vin	5V
GND	GND
SCL	A5
SDA	A4

Table 2. OLED - Arduino Uno Connection Table

Connect GND to common power/data ground.

Connect the PWR to the power supply, for a 3V sensor this is about 3.3V. For 5V version use about 5VDC.

Connect the SDA pin to the I2C data SDA pin on Arduino. On UNO & '328 based Arduino this is also known as A4, on Mega it is also known as digital 20 and on Leonardo/Micro it is also known as digital 2.

Connect the SCL pin to the I2C clock SCL pin on Arduino. On a UNO & '328 based Arduino, this is also known as A5, on a Mega it is also known as digital 21 and on a Leonardo/Micro, also known as digital 3.

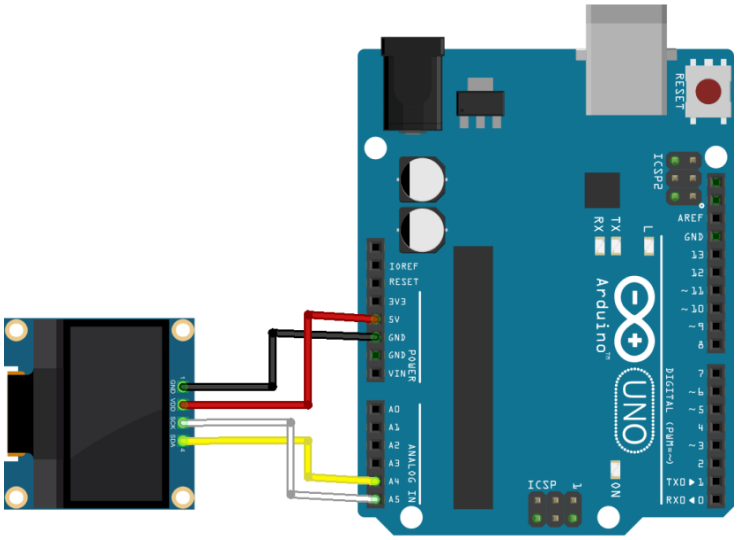


Figure 12. OLED Circuit Scheme with Arduino

1.9.8. RGB Led

RGB 4-Leg Common Cathode Led that can be lit in Red, Blue and Green colors. It is 10mm in size and has a 60 degree viewing angle. The longest leg is the minus (-) end. By giving + voltage to the other legs, the colors can be controlled one by one, that is, it is the common cathode. [17]

Figure 13. RGB Led

Technical Specifications

RGB LED package

10mm diameter

Red: 630 nm wavelength, Green: 525 nm, Blue: 460 nm

Red: 2-2.2V Voltage, at 20mA current, Green: 3.0-3.2V, Blue: 3.0-3.2V

Red: 3000 mcd typical brightness, Green: 5000 mcd, Blue: 900 mcd

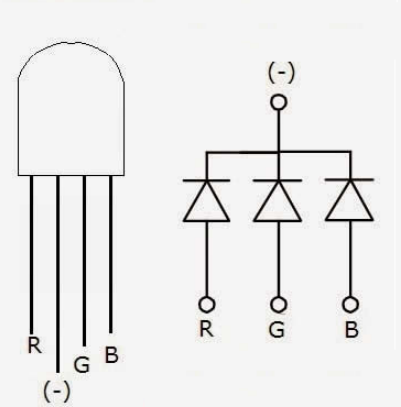


Figure 14. RGB Led Scheme

1.9.9. LIDAR Sensor

The word LIDAR is an abbreviation of "Light Detection and Ranging". It is sometimes referred to as “laser scanning” or “3D scanning”. LIDAR technology uses eye-safe laser beams to create a 3D representation of the environment being studied.

LIDAR shoots invisible beams of light from the light spectrum. This means that a LIDAR sensor can use infrared and ultraviolet light to map the surrounding environment. It can sense both the physical dimensions and movement (if any) of the objects around it. In other words, LIDAR is a way to understand what's going on around you using laser beams. [18,19]

Figure 15. LIDAR Sensor

1.9.9.1. How Does a Lidar Sensor Work?

A typical lidar sensor emits pulsed light waves into the surrounding medium. These pulses bounce off surrounding objects and return to the sensor. The sensor uses the time it takes for each pulse of light to return to the sensor to calculate the distance it has traveled. Repeating this process millions of times per second creates a precise, real-time 3D map of the

environment. An onboard computer can use this map for safe navigation. [18,19]

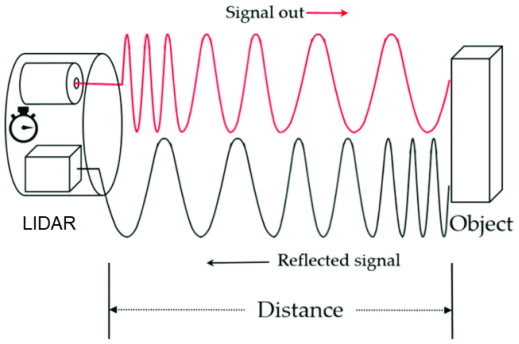


Figure 16. How Does a Lidar Sensor Work

1.10. Software Environments

1.10.1. Python

Python is the programming language that is required to be used. The reason why It's required is that the Python language was built for its readability and less complexity and such a complex design, a simple algorithm language tool is quite recommended. An integrated development environment is also needed like PyCharm, Xcode, VisualStudio etc.

1.10.2. Arduino IDE

The integrated development environment for Arduino is a cross-platform application written in C and C ++ languages. It is used to write and upload programs to Arduino compatible cards, but also can be used for 3rd party cores and vendors' development boards.

1.10.3. JavaScript

JavaScript is a scripting or programming language that permits you to put into effect complicated capabilities on net pages whenever an internet web page does extra than simply take a seat down there and show static facts to be able to appearance at showing well timed content material updates, interactive maps, lively 2D/3D graphics, scrolling video jukeboxes, etc. you may wager that JavaScript might be involved. It is the 1/3 layer of the layer cake of

widespread net technologies, of which (HTML and CSS)

1.10.4. Fritzing

Fritzing is an open source electronic drawing and simulation program. There are relays and switching elements in addition to the main boards and external equipment of various development boards, fixture components such as resistors, leds, and diodes. You can add your own library into the program, so you can use the development boards and external hardware that do not have the circuit elements you want and draw as you want.

1.10.5. What are Unreal Engine 4 and Animation in UE4?

Unreal Engine is a game engine developed by Epic Games and first used in a first-person shooter called Unreal, which was released in 1998. It was developed mainly for first-person shooters, but was later used in various types of games. Thanks to its code written in C++, it has a high degree of portability and has become a tool used by many game developers today.

The current version is Unreal Engine 4, DirectX for Microsoft Windows, Xbox One, Windows RT; OS X uses Linux, PlayStation 4, iOS, OpenGL for Android, and JavaScript/WebGL for internet browsers.

The animation system in Unreal Engine 4 (UE4) consists of several Animation Tools and Editors that confuse the skeletal-based deformation of cages with morph-based corner deformation to allow complex animation. This system can be used to make basic player movement look much more realistic by playing and blending between ready-made Animation Arrays, to create customized custom movements such as scaling ledges and walls using Animation Montages, to apply damage effects or facial expressions through Morph Targets, to control them directly. Create transformations of bones using Skeleton Controls or logic- based State Machines that determine what animation a character should use in a particular situation. [8]

1.10.5.1. Blueprint in Unreal Engine 4

The Blueprint Visual Scripting system in Unreal Engine is a complete gameplay scripting system based on the concept of using a node-based interface to create gameplay elements from within Unreal Editor. As with many common scripting languages, it is used to define object-oriented (OO) classes or objects in the engine. As you use UE4, you'll often find that objects defined using Blueprint are colloquially referred to as just "Blueprints."

This system is extremely flexible and powerful as it provides the ability for designers to use virtually the full range of concepts and tools generally only available to programmers. In addition, Blueprint-specific markup available in Unreal Engine's C++ implementation enables programmers to create baseline systems that can be extended by designers.

Blueprints are Unreal Engine 4's approach to visual scripting. This means that tasks generally reserved for programming scripts are instead created via a graph of nodes and connections, rather than having to type out any actual code. This enables artists and other non-coder type users to create intricate and sophisticated gameplay systems that were previously only available to programmers.

How Do Blueprints Work?

In their basic form, Blueprints are visually scripted additions to game. By connecting Nodes, Events, Functions, and Variables with Wires, it is possible to create complex gameplay elements.

Blueprints work by using graphs of Nodes for various purposes - object construction, individual functions, and general gameplay events - that are specific to each instance of the Blueprint in order to implement behavior and other functionality.

1.10.6. Mixamo

It is a web-based application for developing and editing 3D characters for animation and other hardware. Here, a character network can be created by the user. Positioners can be

placed for character movement. In this way, Mixamo's software is run. Mixamo is an application that can be used by anyone who uses Adobe applications. The Mixamo app is the first online character animation application developed by Stanford University. Mixamo has packages from avatar creation that can select characters suitable for animation. All the characters are fully equipped. In other words, they are suitable for immediate use in terms of hardware. If the desired character is loaded into Mixamo, a model suitable for the desired hardware is animated. Available characters are not limited to humans only. Animals are also involved in the application Characters that can be created with Mixamo are often used in illustration and graphics, 3D character creation, movie or video game production. [20]

1.10.7. 3Ds Max

The full name stands for 3D Studio max. Designed by Autodesk. 3D Studio Max is a program used for drawing. More 3D model designs are made. Besides modeling, it is also suitable for animation production. The graphic processing feature that can be done with this makes the program selectable. Although it is a program that is often used for advertising, it is a very suitable application for animation production.

1.10.8. Teachable Machine

Teachable Machine is a web tool that makes it fast and easy to create machine learning models for projects. Train a computer to recognize images, sounds, & poses, then export models for sites, apps, and more. [21]

1.11. Libraries

1.11.1. OpenCV

OpenCV is an open-source library that allows its users to capture real-time videos and images. The user may access a camera by using OpenCV and apply the needed algorithms to spot the humans in front of the camera and calculate their distances. [22,23]

1.11.2. Haar Cascade

Haar Cascade is a machine learning-based approach where a lot of positive and negative images are used to train the classifier.

1.11.3. NumPy

Numpy is a Python library required to run arrays. It is used to help make mathematical operations easy. The library has its own array. It is preferred because it is faster than Python's arrays. [24]

1.11.4. Adafruit

Adafruid is a company that manufactures additional products for processors such as Arduino. In functions that use attachments, different versions are run for different functions, working together with Arduino. [25]

1.11.5. Pygame

Pygame is a Python library built on top of the SDL library for creating interactive games in the Python programming language by Pete Shinners.[26]

2. FUNCTIONALITIES

FLOWCHART

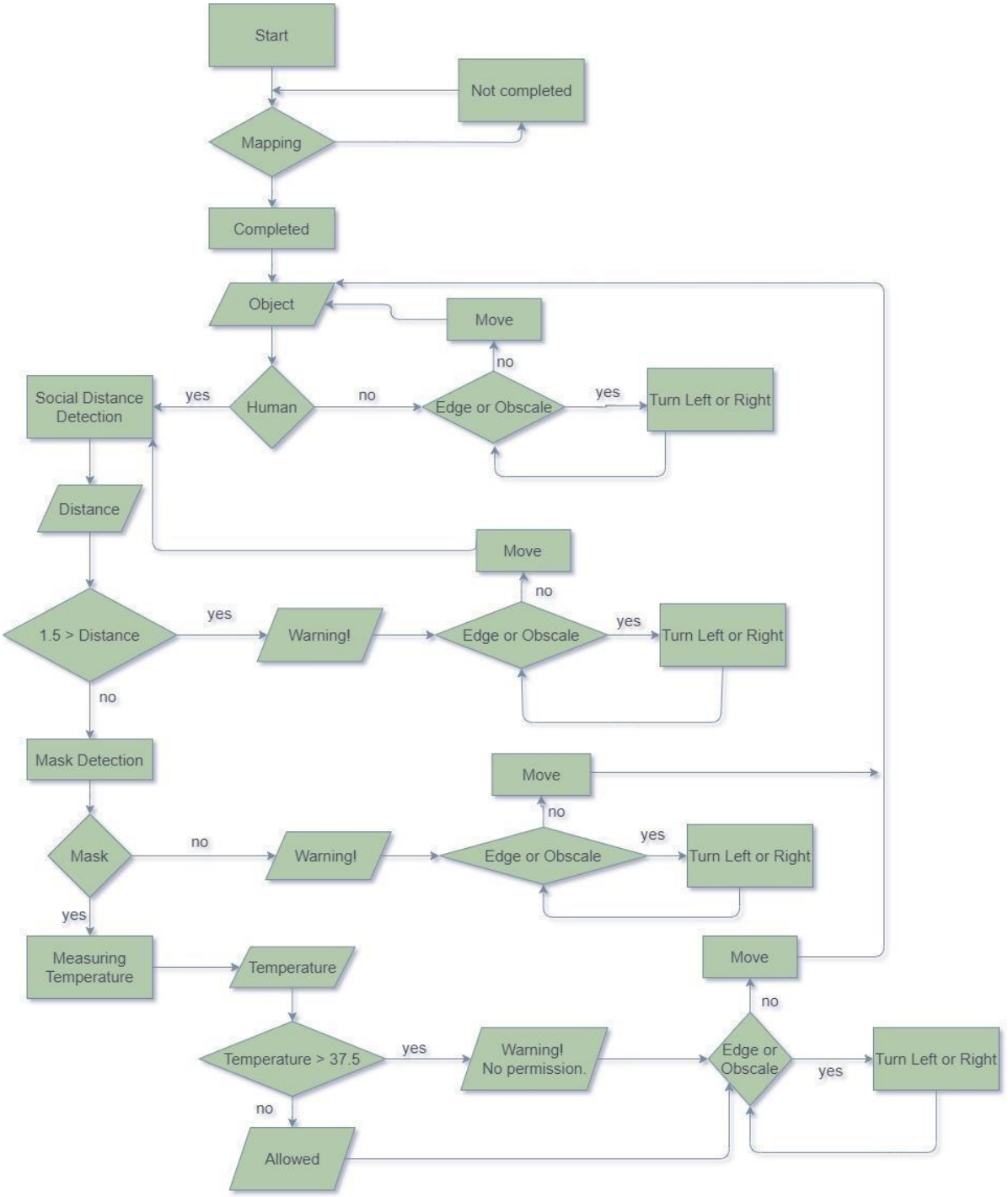


Figure . Flowchart of the Robots's Actions

2.1. Face and Mask Detection

Image processing is the name given to the technique that enables any process to be performed on images taken by any device. Image processing; It is used for many purposes such as increasing the clarity of an image, obtaining any object on the image or identifying objects. To be able to use any image via software, it must be digitized. Digitization; It is the expression of the colors in the picture with numerical values. Thanks to the rapid increase in technological developments, there is a rapid increase in security applications. Many personnel automation based on face, fingerprint and iris recognition are being developed. In the aforementioned applications, after any image is taken, image processing steps are applied on it to obtain the necessary data. Although there are standard methods of image processing, there are also ready-made libraries developed outside of these methods. OpenCV, EmguCV, AForge.NET are some of these libraries and they enable the operations to be done faster and with less command lines. Face detection is used to find and identify human faces in digital images and is an artificial intelligence (AI) based computer technology. Face detection technology can be applied to a variety of fields, including security, law enforcement, biometrics, entertainment, and personal security to enable people to be tracked and tracked in real time. [27,28,29]

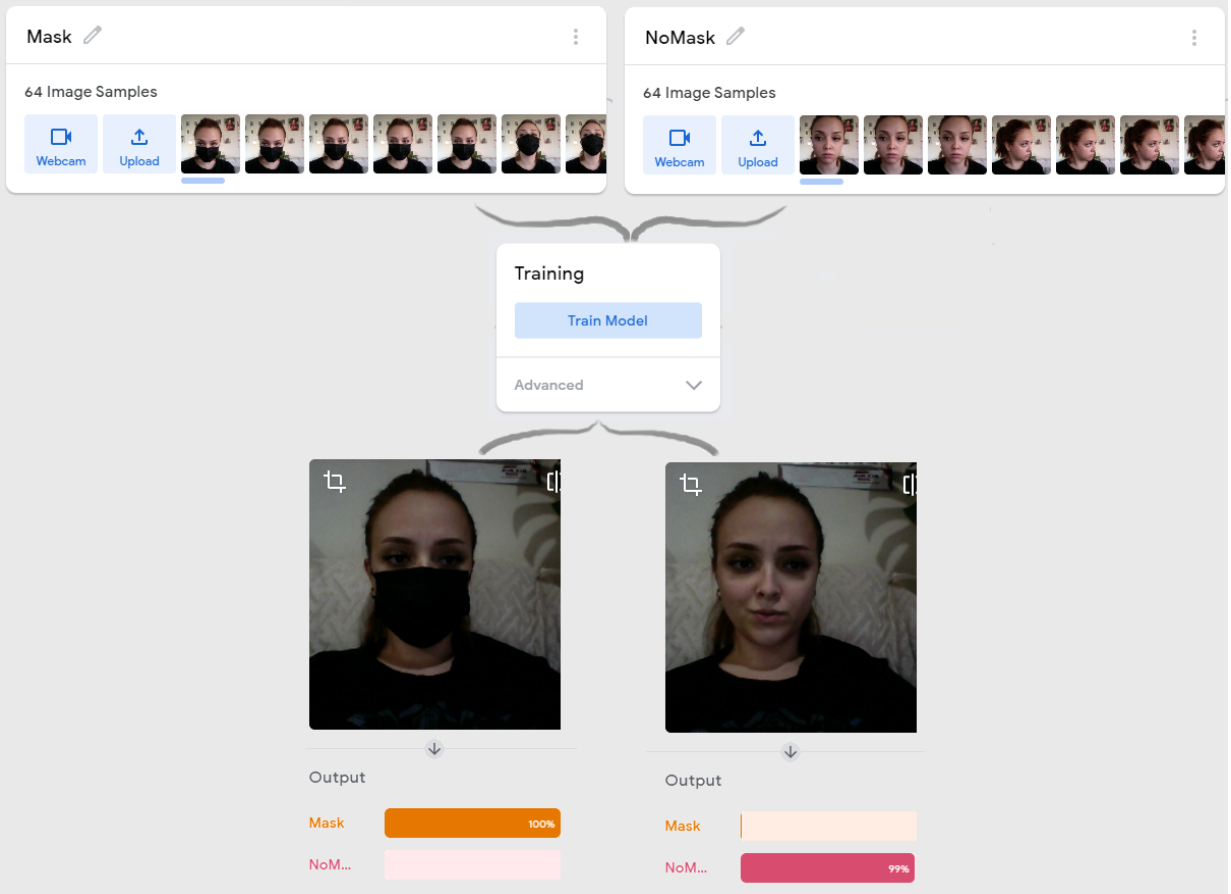


Figure 18. Data Set Training

As described above, datasets with masked and unmasked faces were created. Machine learning will be done using these datasets, allowing the system to distinguish between masked and unmasked faces. Thanks to the Teachable Machine, the data sets required to operate the mask and face detection function of the project were created with the masked and unmasked form of the group member.

2.1.1. Pseudo Code of the Mask Detection

```
data=0;
source=0; // If the camera captures an image; it puts it in the source variable.
isThereMask; //Variable that holds the return value from the mask check function.
humanFrame; //The frame where the robot sees people.
Start VideoCapture;
while(true){
    source=VideoCapture; //Captures from videocapture are thrown into the source variable.
    if(source==object){                    // If an object is thrown into the source variable.
```

```
            data=function detectHuman(source);          // Human recognition function is sent to
        see if the object is human.
            if (data == true){                          // If the object captured is human
                isThereMask= function deteckMask(source); //Call mask check function.
                if (isThereMask==true){ // If the mask exists and it is worn correctly.
                    humanFrame=green;
                    print("admission allowed"); // Access is granted.
            }
            else if (isThereMask==false){           // The mask is missing or inserted incorrectly.
                humanFrame=red;
                print("Warning, no entry allowed"); // Warning! No access allowed.
                    }
                }
            }
        else{           // Continue videocapture if an object is not captured.
                continue VideoCapture;
        }
}
Function bool detectHuman(object): {
        human=human;
        if (object == human){           //If the object is a human
            return true;
        }
        else{           // If the object is not human
            return false;
        }
}
Function detectMask(object): {
        If (there is a mask){ //If there is a mask
            If (the mask is worn correctly){ //If the mask is worn correctly.
                return true; }
            else{           // If the mask is present but not worn correctly.
                return false;
            }
        }
        else if (no mask){
            return false;
```

```
        }
    }
```

2.1.2. Results of Face and Mask Detection

Figure 19. Result of Face and Mask Detection

In this figure, the first image of the working phase of the face and mask detection function is shown. This is how the function starts to work.

Figure 20. Result of Face and Mask Detection with Mask

In this Figure 20, it is shown that the function studied performs mask detection correctly. The mask should be worn with the mouth and nose completely closed, and the function was operated in this way.

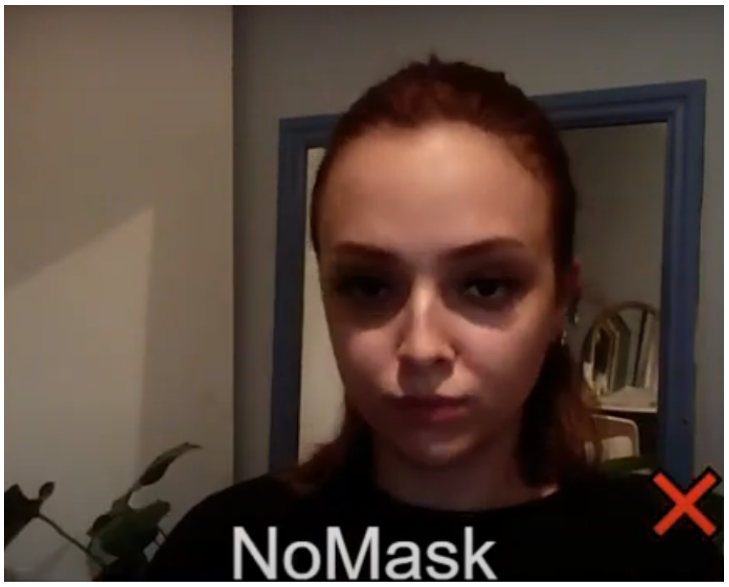


Figure 21. Result of Face and Mask Detection without Mask

The reaction of the program to the unmasked person is shown in figure 21. Although there is no mask on the detected face, the text “NoMask” appears with the X sign as seen in the figure.

Using the model trained on the Teachable Machine in JavaScript code, it can be determined whether the person in front of the live camera is with or without a mask. Thus, the face and mask detection function, which has an important place in the project, was activated.

Finally, the function that seen to work on the computer, when the robot is implemented in real life, needs to be installed on a mini computer such as a Raspberry Pi.

2.1.3. Simulation part of Face and Mask Detection

For the simulation part of this process the team used Unreal Engine 4 engine to create an environment for the robot to wander around and detect whether the humans (in this case

human models, not real humans) are wearing a mask or not. In order to execute this simulation the Actor and Blueprint tools were used.

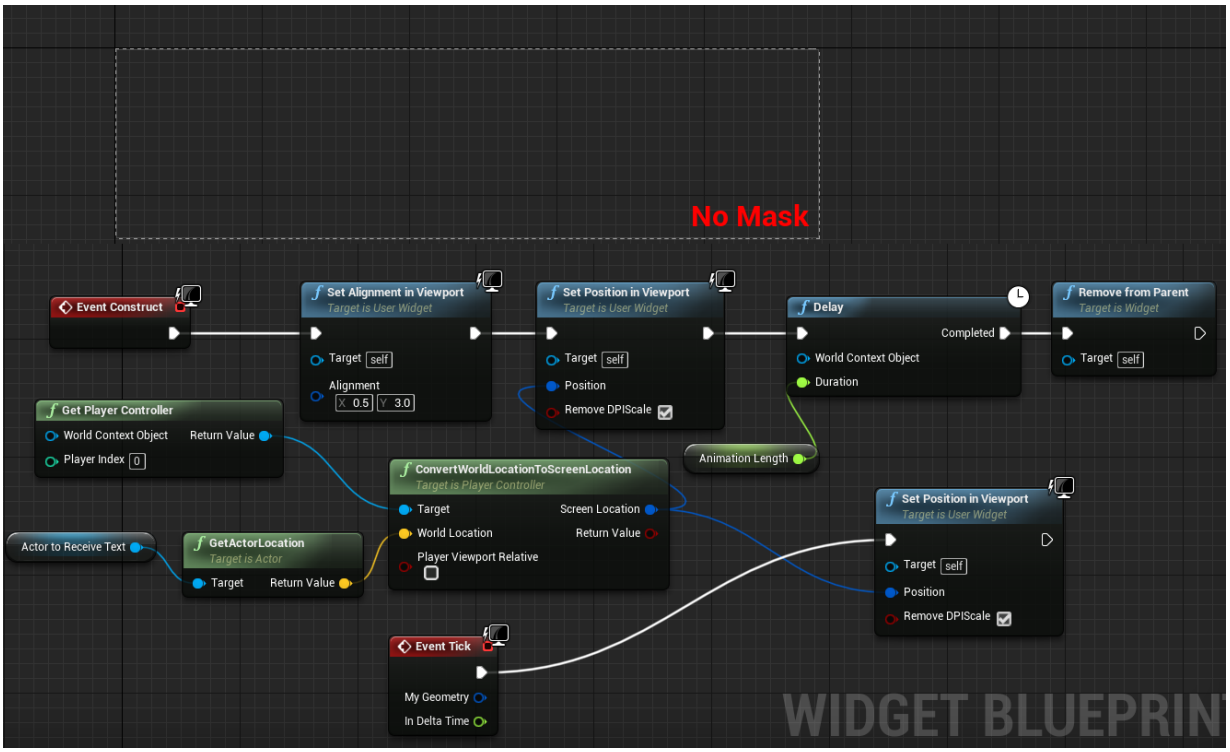


Figure 22. Blueprint of the Unreal Simulation of Mask Detection Function

The Figure above shows the blueprint scheme of the robots mask detection ability. With that scheme above the robot will be able to detect whether the models inside the simulations are wearing a mask or not.

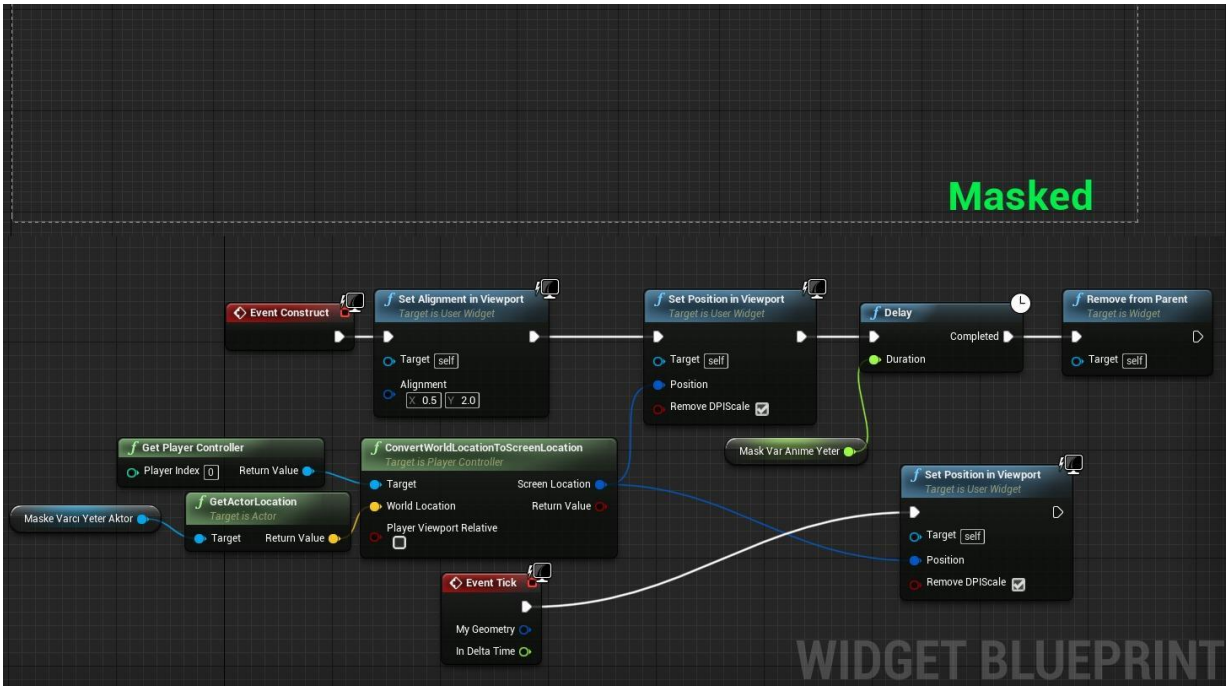


Figure 23. Blueprint of the Unreal Simulation of Mask Detection Function (With Mask)

Figure 24. Result of Face and Mask Detection Simulation

In the figure x it can be shown that, by applying the blueprint that has shown above, the face mask on the human model can easily be observed. Right above the green mask text the temperature widget (see: **Simulation part of Temperature Detection**) can also be seen as well.

2.2. Detecting Temperature

One of the most important symptoms of Covid-19 virus is elevated body temperature. Measuring fire with thermometers is one of the easiest methods to detect the virus. Therefore, it is aimed to keep healthy people away from the risk of Covid-19 virus by using fever measuring devices in places where people are together, such as schools, companies, factories, airports, hospitals, transportation vehicles, restaurants and cafes. Therefore, one of the characteristics of the robot is that it can measure temperature. The Robot is to measure the body temperatures of students, instructors, or civilians entering the campus. In the human body, the skin temperature is between 32 ° C (cheek) and 34 ° C (forehead). But it is below normal body temperature (36.5 ° C - 37.5 ° C) due to the heat exchange between the human body and the environment (loss of body temperature) [30]. The reason for measuring body temperature is that one of the most important and first indicators of the Covid-19 outbreak is high fever.

2.2.1. How Does It Work

In this section, an Arduino program designed to measure fever in a non-contact manner has been developed. In the developed program, people's temperature is expected to be below 37.5 degrees. If the body temperature of the person entering the campus is above 37.5 degrees, the red led will light up to indicate that this person does not have an entry permit, and if it is below 37.5 degrees, it will show that he has an entry permit with a green led.

Arduino is a kit developed for setting up electronic projects. With this software platform, circuits can be installed and operated using sensors and components. The Arduino programming language can be said to be almost identical to the C programming language. For this reason, it is possible to write this program with knowledge of the C programming language. The sensor and components used in the program developed for this part consist of the MLX90614 sensor, OLED and an Arduino Uno.

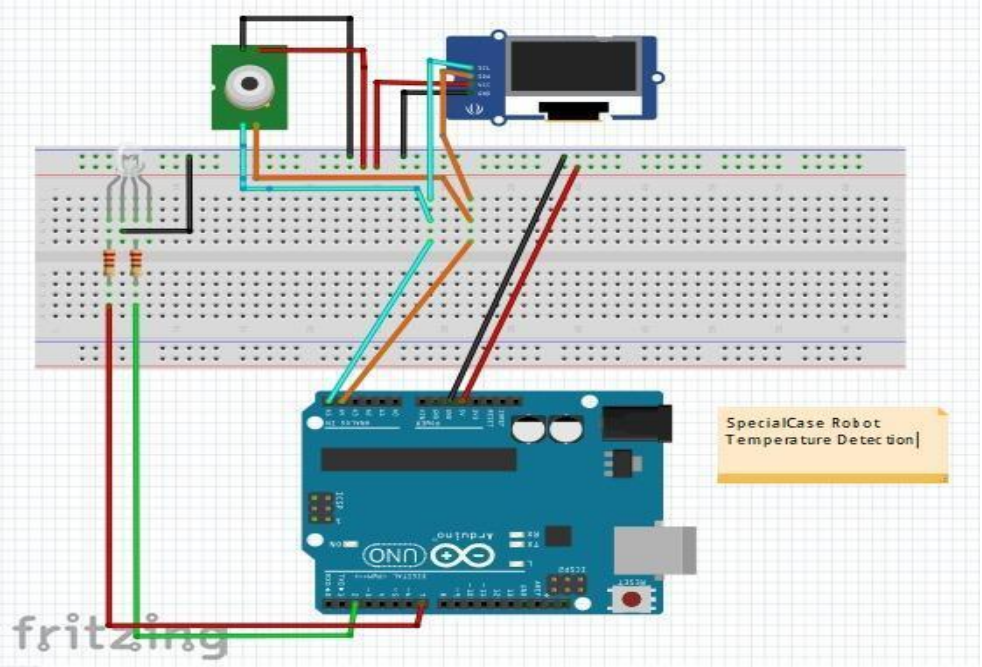


Figure 25. Temperature Detecting Circuit Representation and Sensor Communication

2.2.2. Pseudo-Code of Detecting Temperature

data=0;

source=0; // *If the camera captures an image; it puts it in the source variable.* temperatureValue // *Variable that holds the return value of body temperature.*

humanFrame; //*The frame where the robot sees people*.

Start VideoCapture;

while(true){

source=VideoCapture; //*Captures from videocapture are thrown into the source variable.*

if(source==object){ // *If an object is thrown into the source variable*.

data=function detectHuman(source); // *Human recognition function is sent to see if the object is human.*

if (data == true){ // *If the object captured is human.*

}

}

```
}
        temperatureValue= function detectTemperature(source); //The temperature is sent to the control function.
        if(temperatureValue <= 37.5 ){
                print("You can enter.") // If the temperature value is less or equal than 37 value, a person is allowed to enter the campus.
        }
        if(temperatureValue > 37.5){
                print("You cannot enter.") // If the temperature is higher than 37 value, a person is not allowed to enter the campus.
        }
        else{ // Continue videocapture if an object is not captured.
                continue VideoCapture;
        }
        human=human;
        if (object == human){ //If the object is a human
                return true;
        }
        else{ // If the object is not human
                return false;
        }
```

2.2.3. Results of Temperature Detection

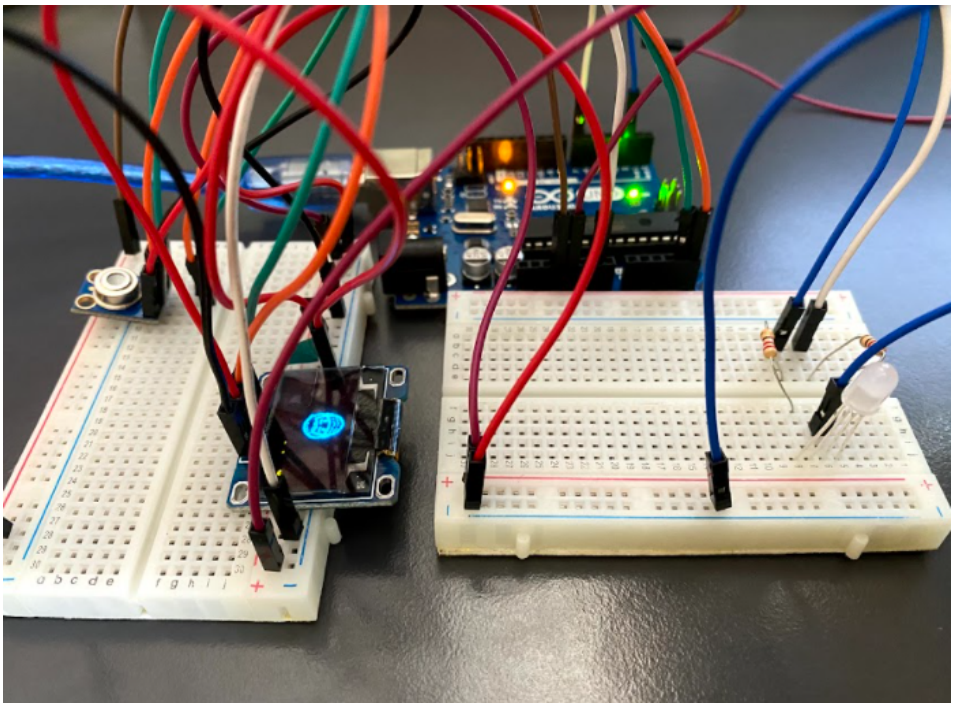


Figure 26. Temperature circuit built on Breadboard 1

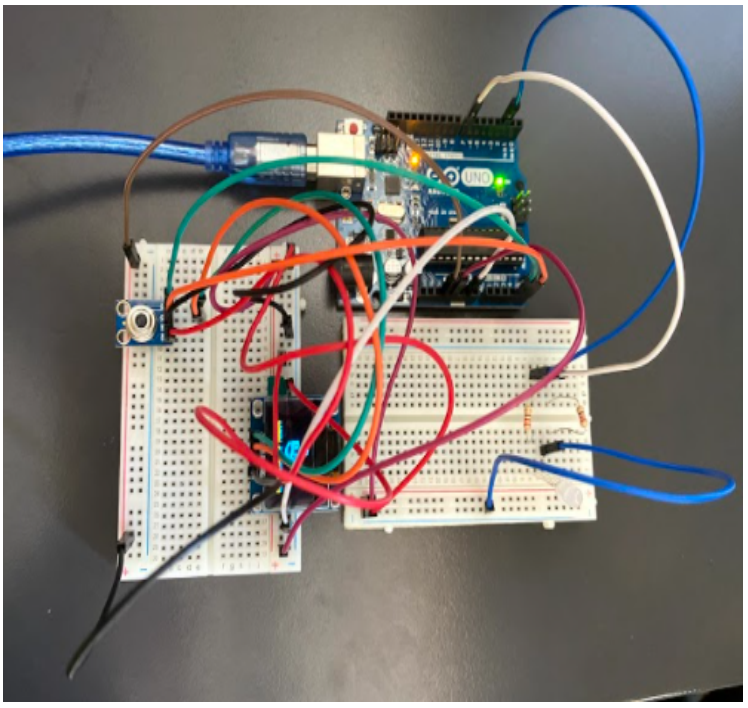


Figure 27. Temperature circuit built on Breadboard 2

In the two figures above, the temperature circuit connection is shown. After connecting the

circuit, the OLED becomes operational. But since there is no person approaching the sensor yet, no value is shown on the OLED.

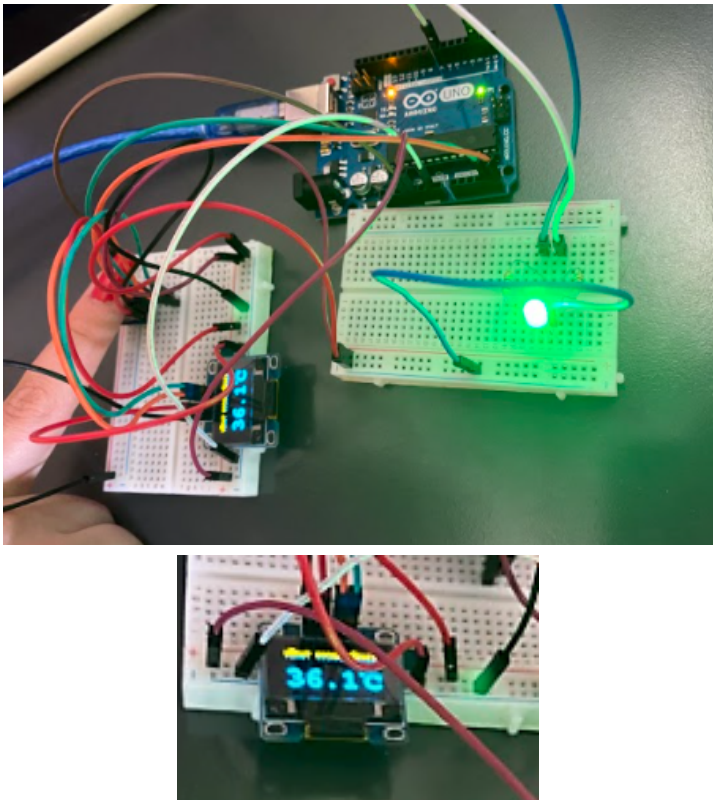


Figure 28.Circuit response at low temperature of 37.5 degrees

When someone with a body temperature below 37.5 degrees points their finger at the sensor, the data they receive from the sensor will be automatically written on the OLED and the green light will be turned on, allowing the person to enter.

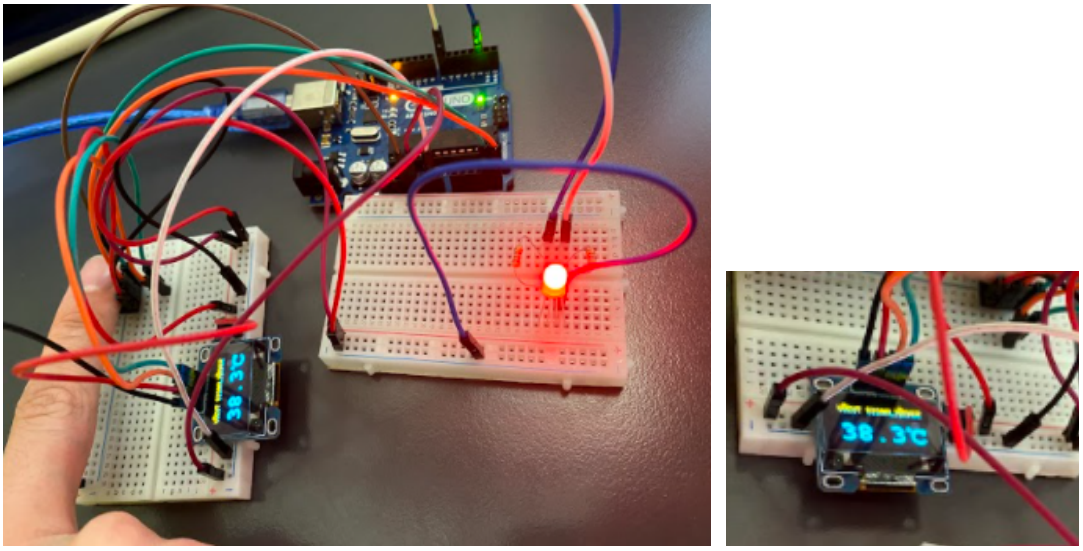


Figure 29. Circuit response at high temperature of 37.5 degrees

When a person with a body temperature above 37.5 degrees brings his finger closer to the sensor, the value will again appear in OLED, but the red light will light to inform the person that entry was not allowed.

2.2.4. Simulation part of Temperature Detection

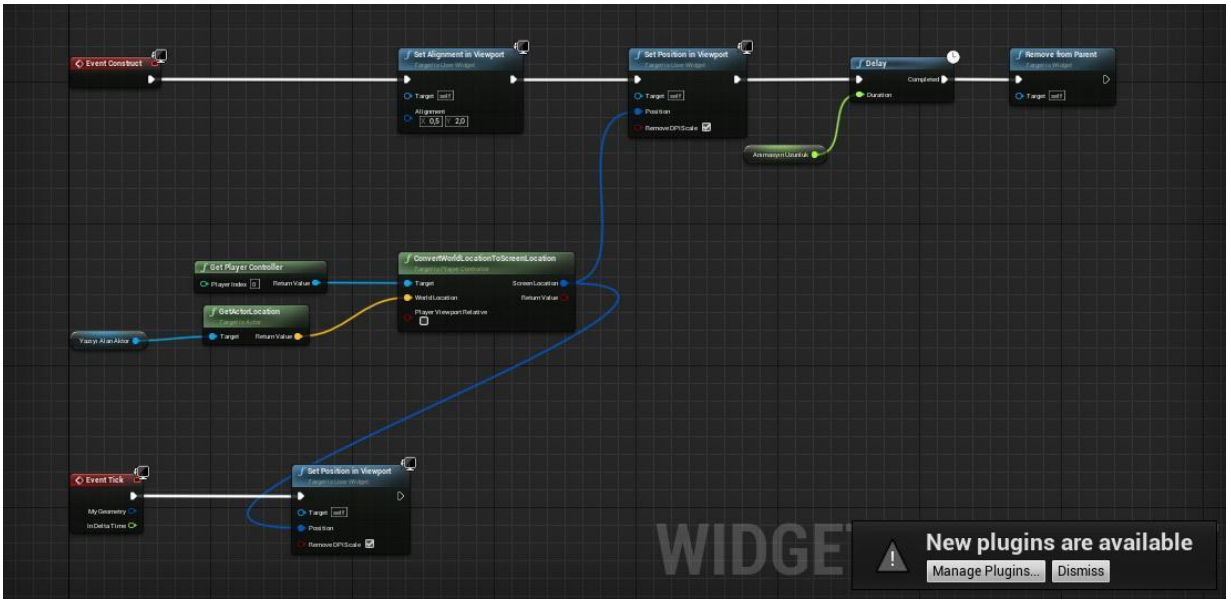


Figure 30. Blueprint Screen of Temperature Detection

The Blueprint scheme that is shown above can also be named as the event graph of

temperature detection. With this process above the temperature of the actor will be displayed right next to generated human model in Unreal Engine 4.

Figure 31. Result of Temperature Detection Simulation

After applying the blueprint scheme to the simulation segment of the Unreal Engine 4, The Following process will occur. The desired robot will advance in front of the human model inside the simulated map, and with the Blueprint schemes the robot will be allowed to see whether if the human is wearing a mask (see: Simulation part of Face and Mask Detection) and also will be able to see the temperature of the human model.

2.3. Distance Measurement

To make a robot calculate distances between the humans, the robot must use certain instruments as a webcam (the camera will detect the humans). The camera will detect the designated object and the distance between them.

2.3.1. Social Distance Functions and How To Execute Them

The robot (A machine) should know what is a human or what is the designated object

before even spotting the humans and calculating the distance between them. The wanted object, in this case, humans, must be identified to the robot via a dataset (Deep Learning Tools) and process the repository and after the robot ―learns‖ what to detect in front of itself, the distance calculation algorithm will be executed. [23]

- OpenCV
- Python

2.3.2. The Method

The required environment for such operation must be created, after the following tools are required, the following operations are listed below;

Step 1 - Defining the required objects (humans): As was explained previously, OpenCV will be the correct library for this function.

Step 2 - Detecting the objects in front of the camera: After the teaching process, the real-time video capture feature will be activated and the camera will detect the required objects In front of itself by of course following certain algorithmic operations.

Step 3 - Calculating the distance between people: For this very particular function, certain math tools and libraries must be included in the code. In a nutshell, a matrix-based and pixel- counting operation between objects will be executed to calculate the actual distance between humans.

Step 4 - If people move away from each other at a certain distance, the screen gives a green light, but if people do not follow the social distance rule and are very close to each other, "Keep Social Distance" will appear on the screen.

With the correct combination of the required tools which are listed above, It is possible to create such an environment to execute the right algorithm and code to detect the social distance between humans.

In order to perceive the social distance between people, it is first necessary to perceive the people on the camera. OpenCV already contains many pre-trained classifiers for face, eyes, smile etc. CascadeClassifier function of OpenCV is used for this. In this way, the

social distance between the detected people was measured. In the program developed by paying attention to the 1.5 meter social distance rule, when people are closer than 1.5 meters, the system makes an error sound and at the same time, "Keep Social Distance!" text appears. This process continues until the 1.5 meter social distance rule is followed. So, as soon as people follow this social distance rule, the error sounds stops and the text disappears.

Finally, when the robot is implemented in real life, The Social Distance Measurement Function, which is seen to work on the computer, needs to be installed on a mini computer such as a Raspberry Pi, as in The Face and Mask Detection part.

2.3.3. Pseudo Codes of Social Distance Measurement

```
while(true) {
        source = VidCapture;
        if(source==object) {

        data=functionDetectHuman(source)

        }

        if (data == true)

                        }
  //Upper part allows us to access the camera and make the algorithm to detect humans.
            Source.persona = a

            Source.personb = b
  //Grabbed people from the camera will be named in this case a and b are given.
            gcdData=gcd(a,b)

function gcd(a, b):

        if a == 0 :
                return b

        return gcd(b%a, a)      //In order to detect the distance between people, the Euclide theorem will be used. The Euclidean algorithm is named as function gcd.
             if { gcd >=1.5meters

                    Return true;

  }
```

2.3.4. Results of Distance Measurement

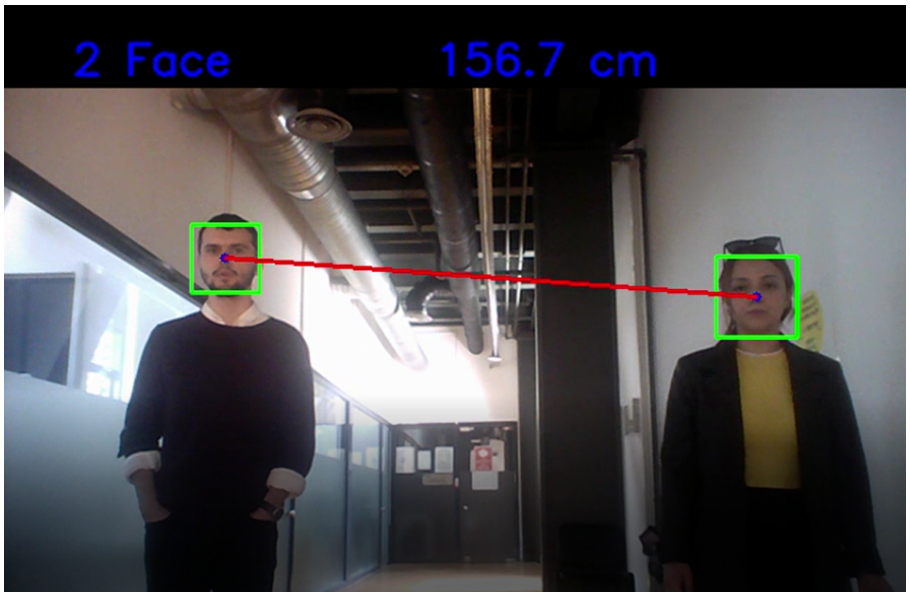


Figure 32. Result of Distance Measurement - Safe Distance

In Figure 32, the output of the Social distance measurement part, which is one of the functions of the robot, is shown. Two human faces are detected here. It is checked whether the distance between these human faces is suitable for social distance. As seen in this figure, the distance between two people is in accordance with social distance rules. Therefore, no warning text appeared on the screen. The instant distance between people is shown on the screen.

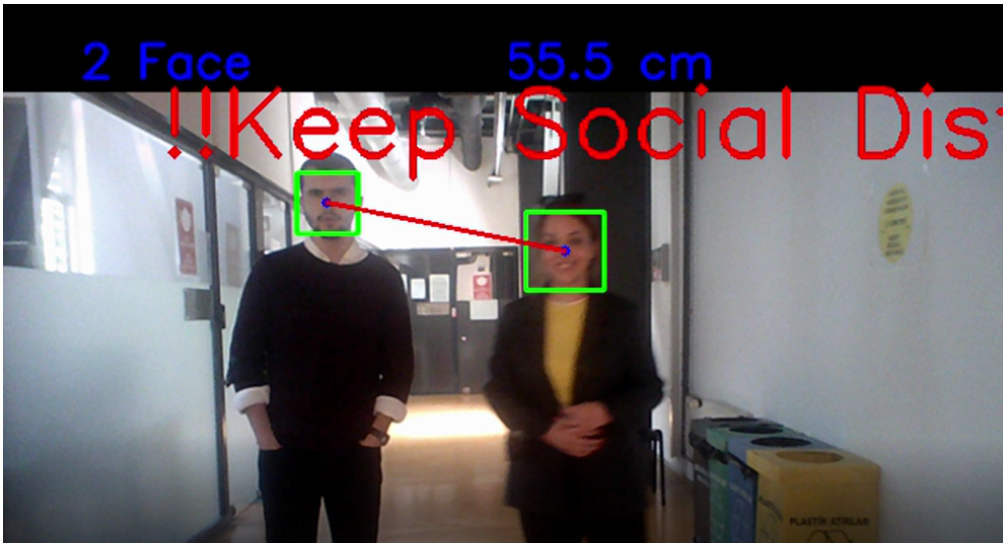


Figure 33. Result of Distance Measurement - Dangerous Distance

In Figure 33, it is seen that the two people in front of the camera do not follow the social distance rule and therefore the text "!!Keep Social Distance!!" appears on the screen.

2.3.5. Simulation part of Distance Measurement

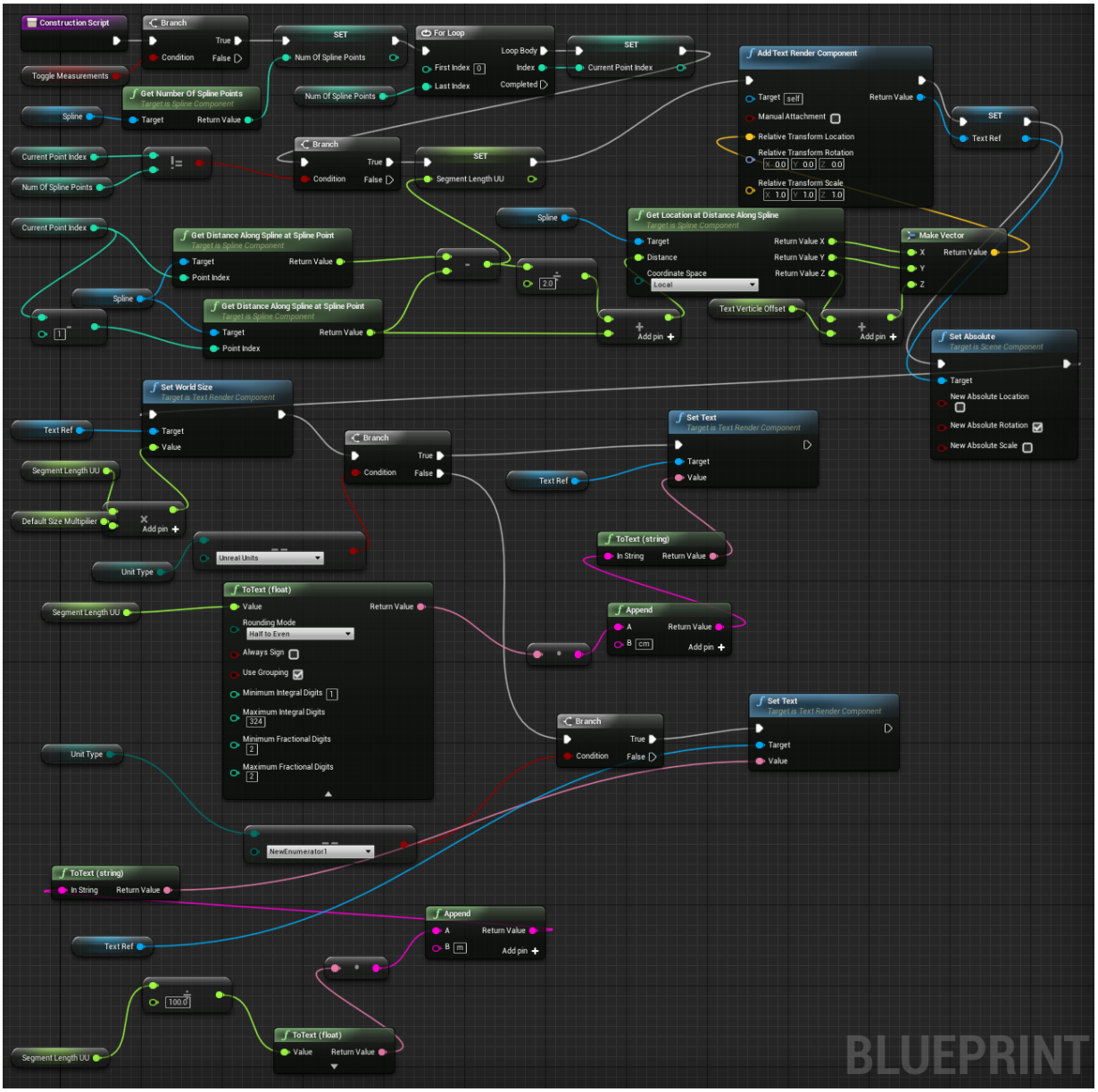


Figure 34. Blueprint Screen of Distance Measurement

It is shown how to create a measurement spline system using the Unreal Engine 4 Blueprints system. Thanks to the algorithm and the functions created over the blueprint, the distance between two objects or two people can be measured.

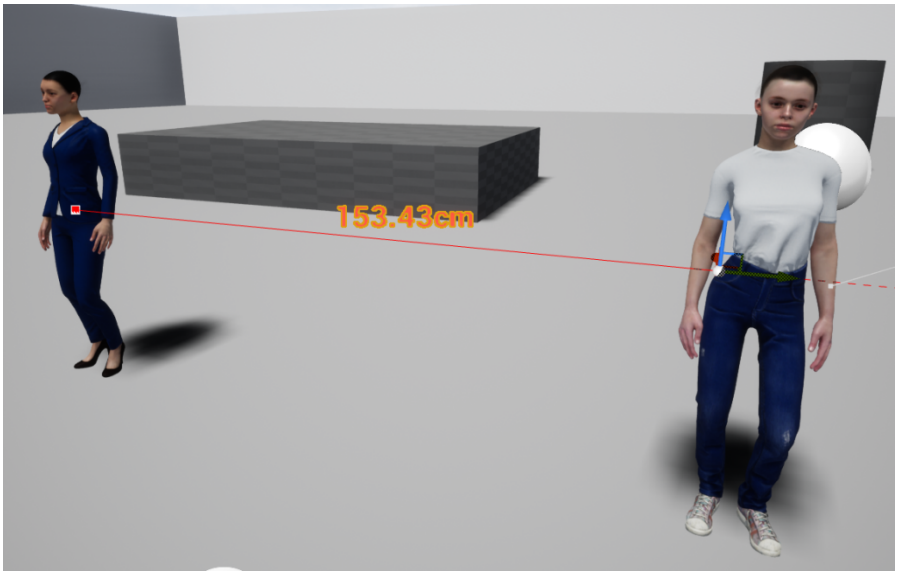


Figure 35. Result of Distance Measurement Simulation Part - 1

It is shown as the distance measurement required to apply the social distance rule is processed with the functions with the Unreal Engine 4 Blueprint. With the algorithm established in Blueprint, the social distance measurement is automatically detected by the program.

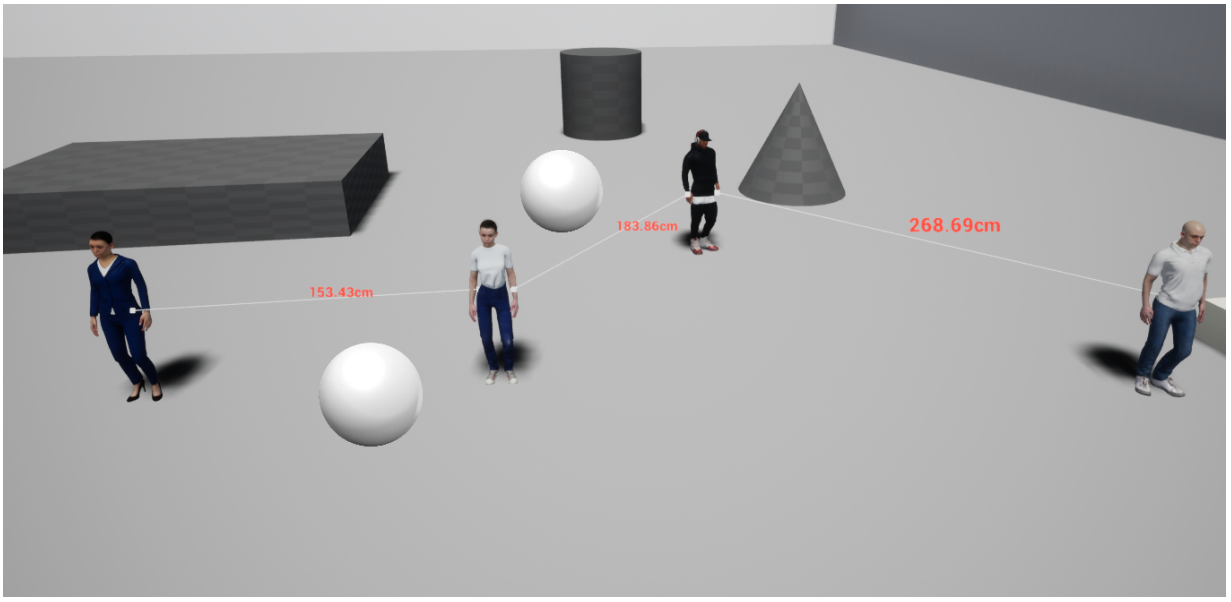


Figure 36. Result of Distance Measurement Simulation Part - 2

This figure shows the social distance measurement to be made when more than two people are in the same environment.

2.4. Mapping and Obstacle Detection

In order for robots to perform their wanderability, they must have the ability to change their movements according to changing situations. In short, it should have autonomous features. The ability to be autonomous is that the robot has the ability to make decisions in the face of a predetermined or undetermined situation thanks to the processor on the robot. The robot must collect data from its surroundings before deciding on its movement. Thanks to the total data obtained from the environment through the robot's sensors, the robot can decide its next movement.
In order for traveling robots to perform their tasks, they need to know or learn the environment they are in and know their own location. The robot must first avoid obstacles during its mobile in the environment.

Robots must have 3 abilities to be autonomous. These are walking, positioning and mapping functions.

There are two sources of information for robotic mapping. This is allothetic and idiothetic information. Idiothetic information sources receive information from odometry. Supports internal information about robot movement. And with this information, the dead reckoning approach can find the robot's position. Allothetic information is caused by perception and sensor. Provides external information about the environment. The information for the robot may be from laser, sonar, sensor or image.

A problem with robotic mapping is also known as measurement noise. This problem can be easily solved if the noise is in various statistically independent measurements.

Mapping can be done with the help of various sensors such as LIDAR, Laser or with image systems. This offers higher resolution than a low-resolution LIDAR. Mapping with the help of a camera is associated with image processing algorithms, which requires higher processing power. [18,19]

What all mapping algorithms have in common is that they are probable. Maps are divided into two, topographical and metric.
In metric maps, groups of objects with environmental coordinates are introduced as metrically formed in two-dimensional space. It's very similar to the architectural outline.

They are maps that many robots can use. It's easier to create than topographic maps.

In topographic maps, the environment is represented as the relationship between special places and these places. Topographic maps define the connection between various places and can be extracted automatically when creating a metric map. It increases geometric resolution and this information can be supportive for tasks such as planning, navigation, and so on. Used for large-scale environments. [18,19]

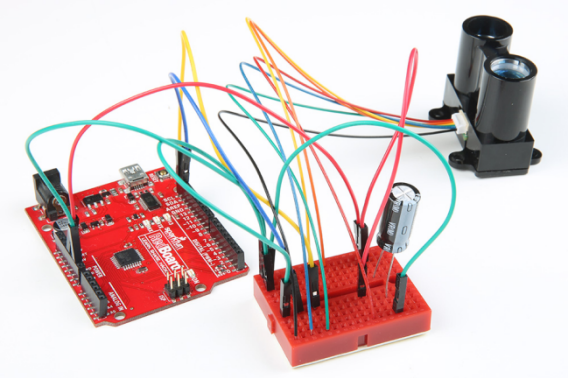


Figure 37. LIDAR Sensor Connection

The connection required for the use of The LIDAR Sensor, which has an important place for the mapping function, is made as shown in The Figure 37.

2.4.1. Pseudo Codes of Mapping

#include
"motion" Start
Motion; Start
lidarSensor;
capturedBySen
sor; source;
map <string, data>
environment; int

```
locationArray[5][5];
while(true){
    for(i=0; i<5; i++){
        for(int j=0; j<5; j++){
            changeLocation(locationArray[i][j]); // It is constantly traveling within
        a certain area.
            if (source==object){
                source=lidarSensor(capturedBySensor); // when the sensor
            captures something, throw it to source
                environment.insert(source); // adds objects captured in the map
            }
            else{
                continue Motion;
            }
        }
    }
```

2.4.2. Pseudo Codes of Obstacle Detection and Motion

```
#def
maxDistance
Start sensor;
Start VideoCapture; sensorSource; videoCaptureSource;Location currentLocation[2]=
{0,0}; // first value is horizontal, second vertical
int distance=0; //distance will be kept here
obj
ect
;
te
mp
;
```

```
data; // if 1; person, if 0; edge
while(true){
    source = sensor; //when sensor detect, take to source.
    if (source == object){
        data= checkObject(source); // if 0 there is an obstacle
        if (data == 0){
            checkEnvironment(currentLocation[2]); // If it sends 0 right, if 1 sends
          left, 2 forward, 3 back
            while(source is not human){
                data=checkObject(source);
                if (Movable function returned right){
                    temp= checkMovableDistance(currentLocation[2];
                    moveRight; currentLocation[0]+=currentLocation[0]+1;
                    // 1 meter
                    if (currentLocation[1]==5){   //the robot goes off the
                    map
                        Stop Motion;
                        break;
                    }
                }
                else if(Movable function returned left){ // move left
                    temp=checkMovableDistance(currentLocation[2]);
                    moveLeft;
                    currentLocation[0] = currentLocation[0]-1; // 1 meter
                    if (currentLocation == -5){ //the robot goes off the map
                        Stop Motion;
                        break;
                    }
```

```
                    }
                    else if(Movable function returned up){ // move left
                    temp= checkMovableDistance(currentLocation[2]);
                    moveForward;
                    currentLocation[1] = currentLocation[1]+1; // 1 meter
                        if (currentLocation[1] == 5){  //the robot goes off the map
                            Stop Motion;
                            break;
                            }
                    }
                    else{ // movable function returns back
                        turn around;
                        moveForward;
                        currentLocation[1] = currentLocation[1]-1; // 1 meter
                            if (currentLocation[1] == -5){ //the robot goes off the
                        map
                                    Stop Motion;
                                    break;  }
                            }
                    }
                }
            }
}
function checkObject(object){
    if (object==human){
            return human;
    }
```

```
    else if(object!=human){      //it is edge or obstacle
            return obstacle;
    }
}
function checkEnvironment(location[2]){
    start VideoCapture;
    if (right is movable){
            return right;
    }
    else if (left is movable){
            return left;
    }
    else if (forward is movable){
            return forward;
    }
    else if(backward is movable){
            return backward;
    }
```

2.4.3. Results of Mapping and Obstacle Detection

In this project, image processing was also used for pattern recognition. The aim of the desired recognition is to find and show the midpoints of the targeted objects. In the first stage, a drawing was made that can be representative of the starting point, goals and obstacles according to their colors.

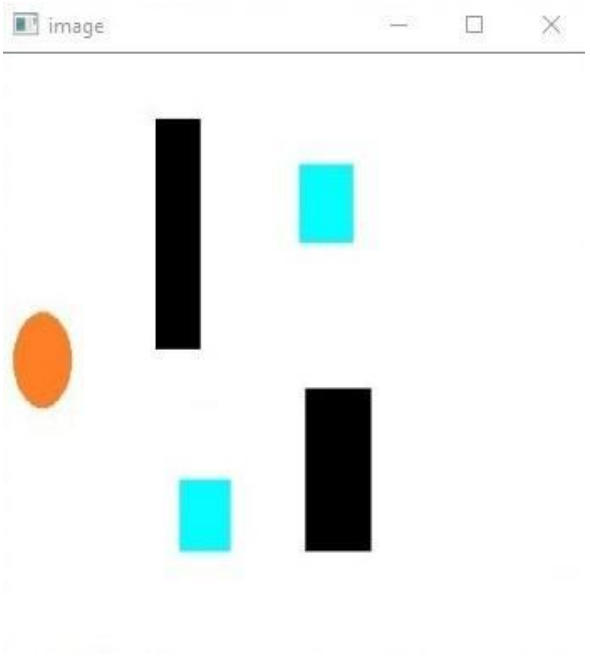


Figure 38. Original Image

In the second stage, the color of the people was introduced and only the masked image showing turquoise-colored targets was obtained.

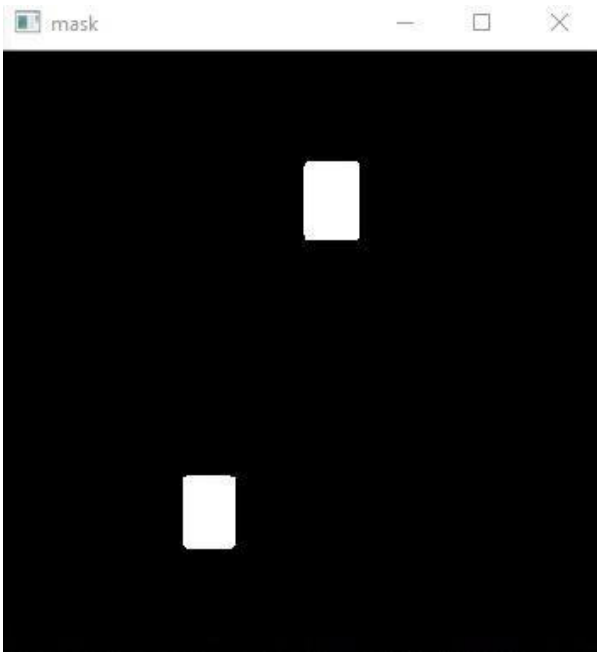


Figure 39. Masked Image

In the third stage, the coordinates of the target objects, i.e. people, were obtained.

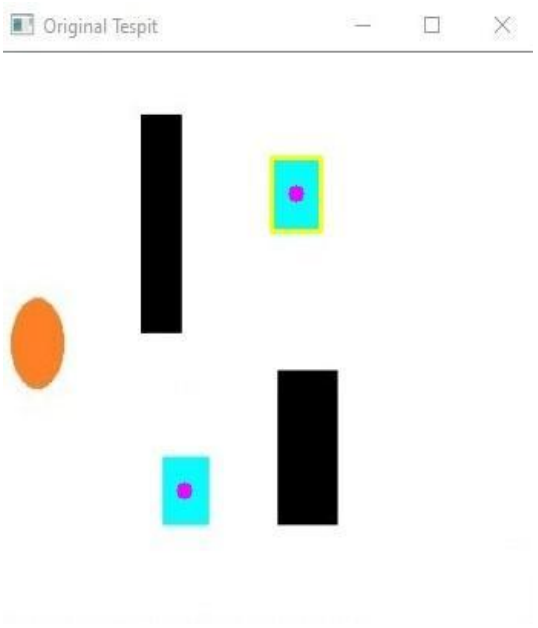


Figure 40. Midpoint Detected Target Objects

The robot, represented by the Orange figure in the created photo, must sense the coordinates of the people represented by the turquoise pentagons. Using the OpenCV and Numpy libraries, the generated code detects the exact points where people are and calculates the distance from the center chosen as the robot's midpoint to the midpoints that the program calculates. The reason for this is that the robot can go to that person, starting with the person closest to it, and then to another. In this way, a ranking is created from the first entrant to the last entrant.

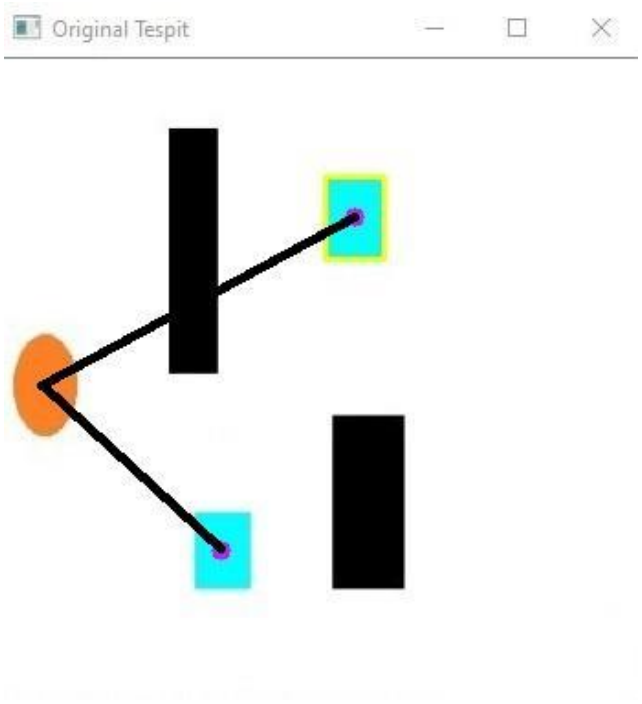


Figure 41. Shortest Path Calculation

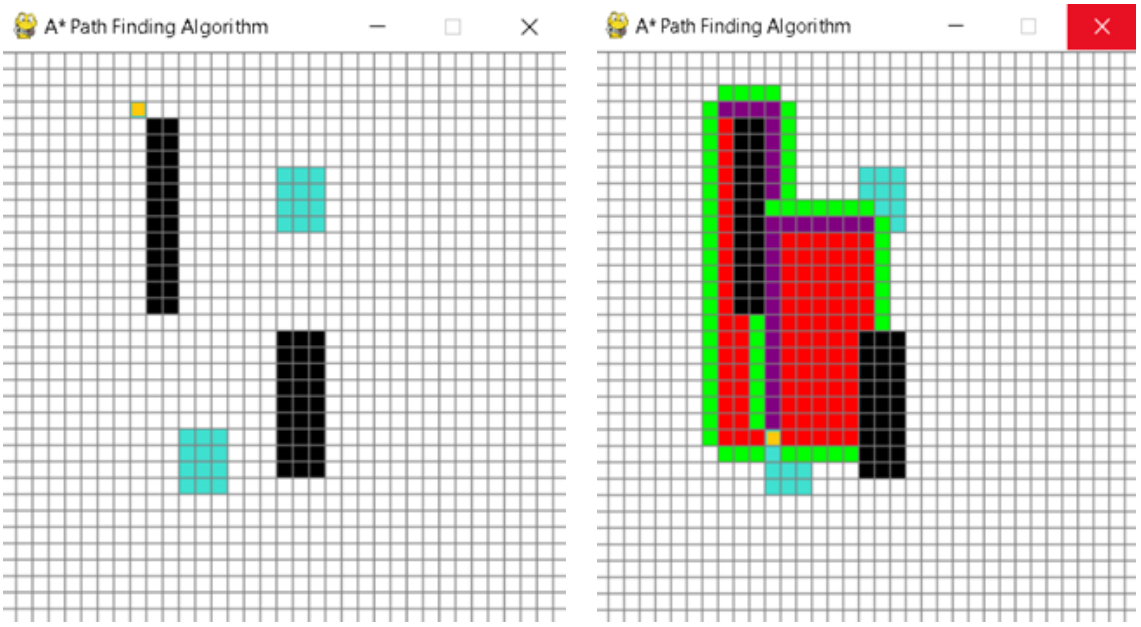


Figure 42. Example of mapping on A* Pathfinding

The above figure shows the function obtained from the A* Path Finding algorithm, aiming to reach the targets in the shortest distance on the map. Since the position of the robot may vary,

it can be placed in the desired position. After mapping, the robot detects the target at the shortest distance and then creates the purple squares, draws a path and reaches the target. Then, it aims to reach the other nearest target by calculating the shortest distance again. The green squares around the created map represent that the border areas outside the map are safe and unobstructed, and the red squares represent the empty areas of the map.

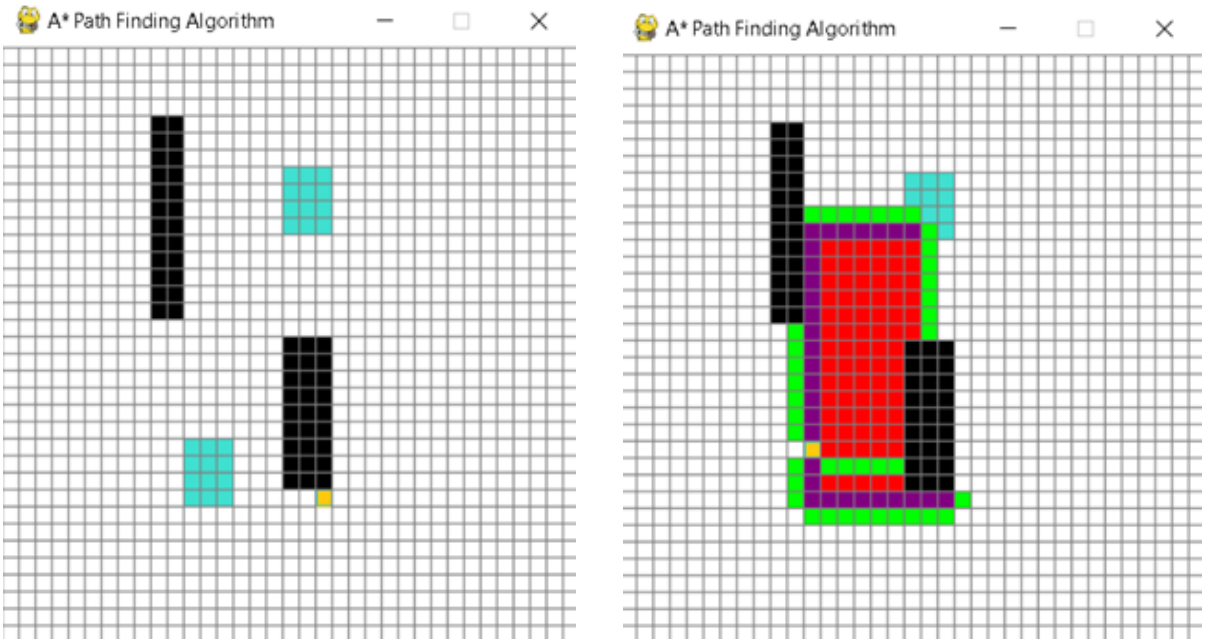


Figure 43. Other Example of mapping on A* Pathfinding

When the position is changed, the function changes and a new map is created again. In the figure above, the position of the robot is taken down. The first target to go to is on the left side. It primarily goes to the left side and then goes to the above destination and the function is completed. Because this algorithm can detect obstacles, it protects the robot from damage caused by impact. Thanks to its ability to measure the shortest distance, it saves time.

2.4.4. Simulation part of Mapping and Obstacle Detection

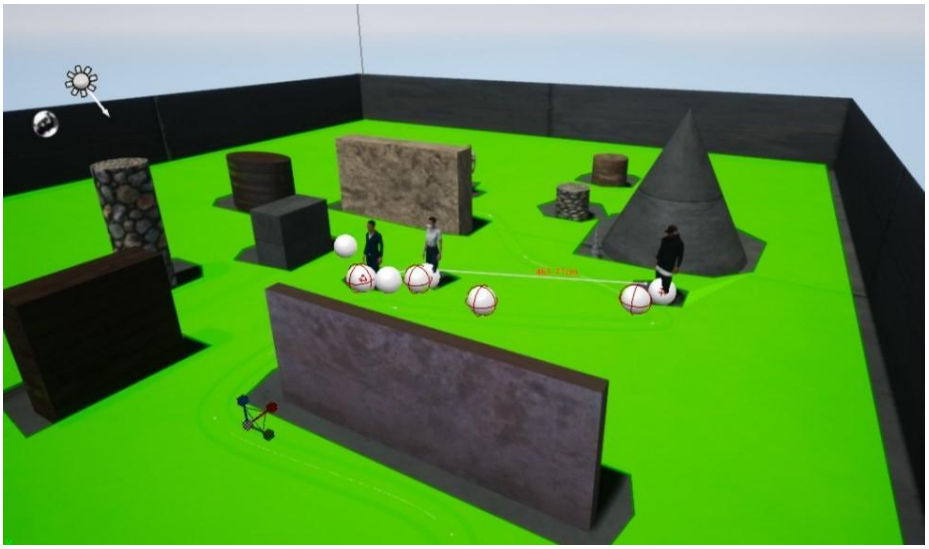


Figure 44. Simulation result of mapping and Obstacle Detection

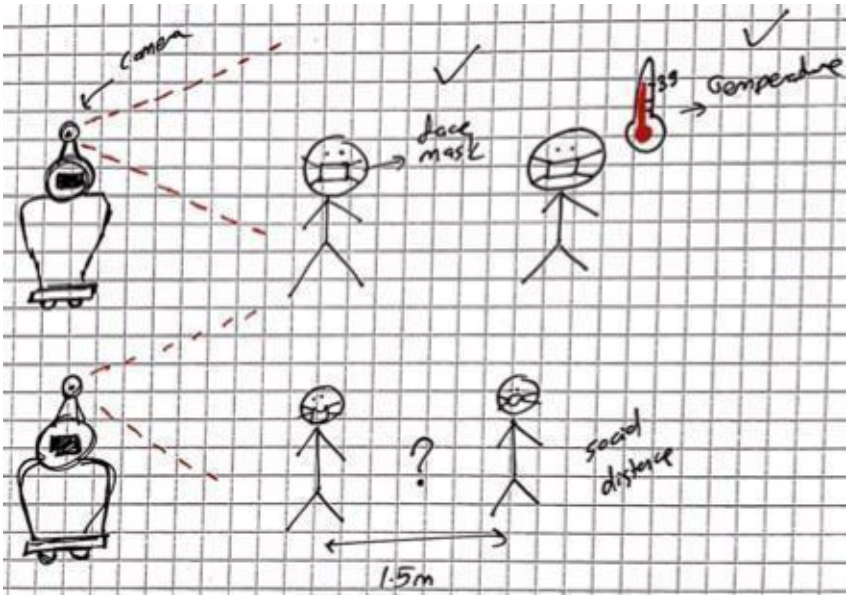
Nav Mesh Bounds Volume is an actor type, a special type of volume that defines the stage areas where navigation networks are created. NavMesh shows how an Actor can navigate through obstacles. The navigation system will usually list things as "simple" because an Actor is not receiving external data as if walking on ice or mud. However, it can easily move in three dimensions by sensing walls, obstacles and any protrusion along the way. The figure x shows that the mapping function of the simulation made in the Unreal Engine 4 environment is working. The robot detects obstacles and people in its working environment by mapping.

3. DESIGNING PROCESS

Before starting to showcase the team's design choices and process, why the conceptual design path was taken should be explained. Conceptual design is the starting point of a designing process. At this point the broad outlines of functions are articulated. The team designed the robot by considering the required functionalities which are social distance detection, facial recognition, mask detection, temperature detection and a moving body.

In order to execute the required functionalities we Raspberry Pi needs to be used and a Camera Module that will help the team to detect the numerical operations. The camera module must be at the tight of minimum 1m, because the average person height is around

1.70 m long and to detect a person's face the robot must be around 1.20 - 1.30 m long (including the height of the wheels)



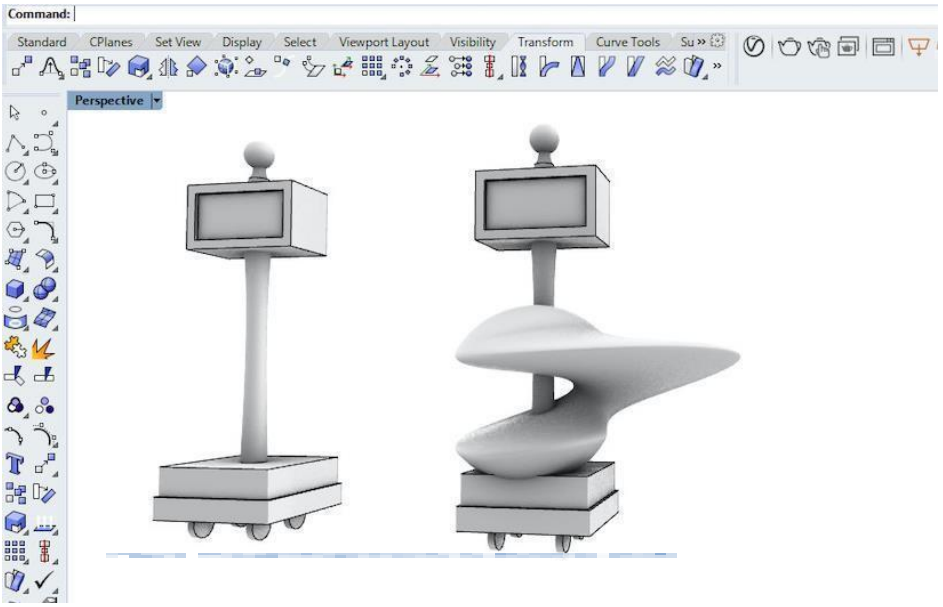
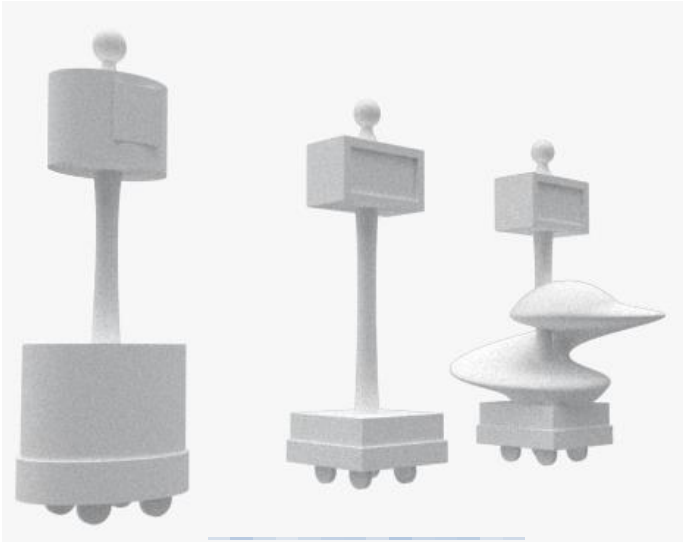


Figure 46. Robot 3D Model -1



3.1. Robot Model

The moving part of the robot; which is the base part; where the motor and the wheels will be; are going to be around 27cm x 15cm. The top of the base platform a cover module will be installed to protect the circuit from outside interventions. A skeleton design will be implemented but since the robot won't have any legs or arms, a simple spine to hold the base and head part will be sufficient (a long metal at the end of the metal rod, a platform the Raspberry Pi and the camera module will be installed.

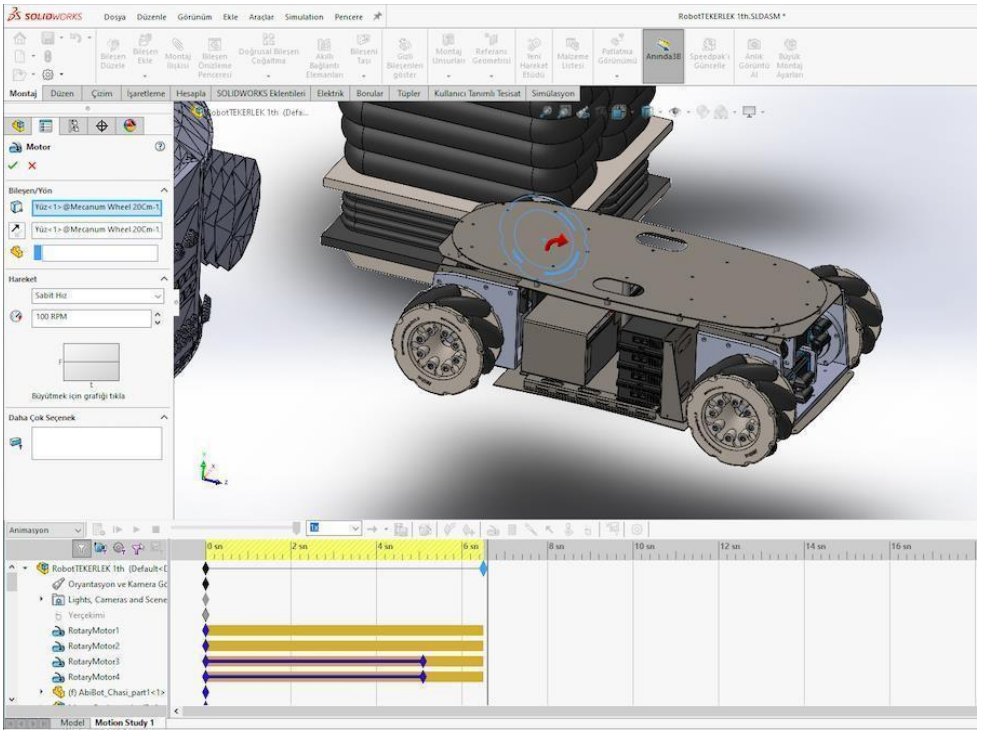


Figure 48. Base of the Robot with Wheels (Model from GrabCAD)

A skull for the highest platform will also be installed and this part needs to be protected very carefully, such electronics like Raspberry Pi are quite vulnerable for outside interactions so they need to be protected very carefully. Every part of the robot is significant and needs to be protected so a 3D scanned body or as an alternative a plastic hollow box that covers with a tarpaulin material can be used.

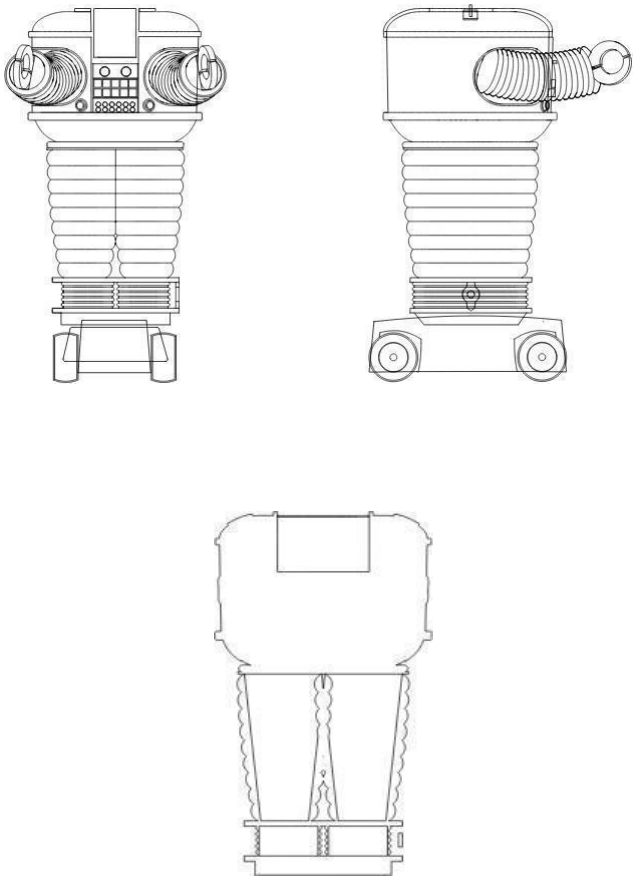
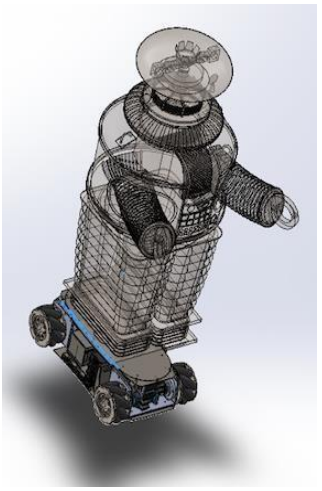


Figure 50. A Section View of the Model (Interior / Skeleton)



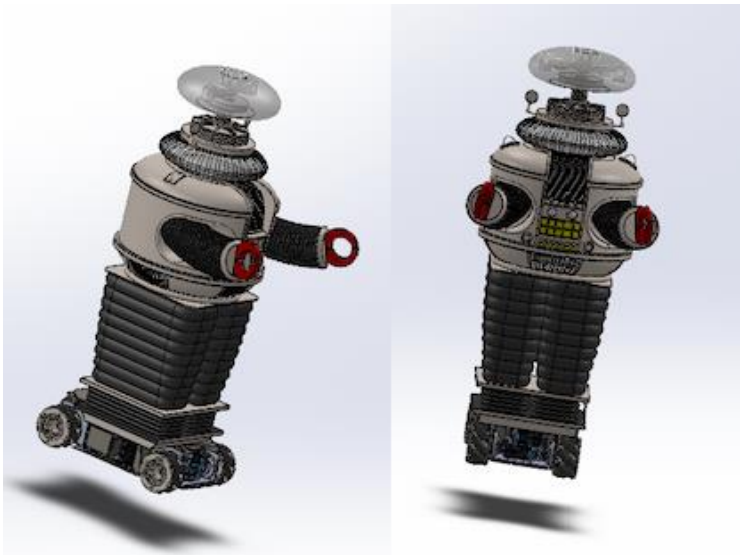


Figure . Body of the Robot Mounted Top of the Base Wheels Platform in Solidworks

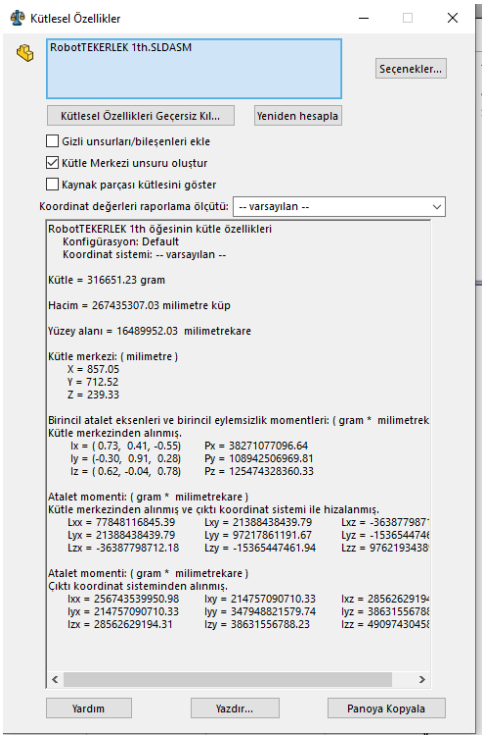


Figure 53. Mass Properties of the Robot

At this part, the center of gravity and other mass properties can be observed by simply implementing the robot model to Solid Works. The reason why the team used this method is that, It is quite hard labour to achieve such numbers by putting together an equation, therefore a computer simulation such as Solid Works needs to be used in order to see these numbers.

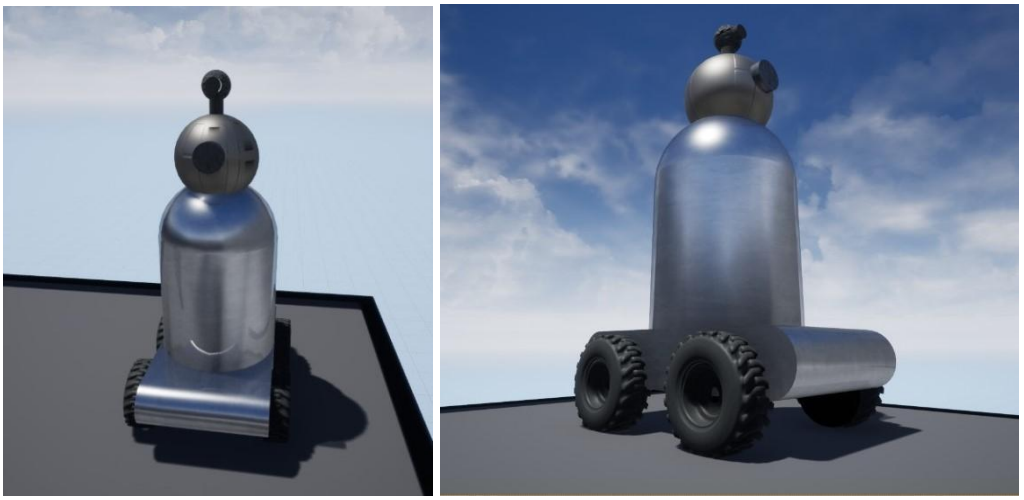


Figure 54. Final Version of the Robot Render

The two figures above are the final version of the robot. It is designed for completing all the tasks that are required with enough space within it. The Final version is more clean and has a lot of free space inside if in case an intervention to the circuits is required it can easily be done.

The position of the circuit boards and the circuit instruments were also quite important. Every single part needs to be maintained in a closed and cold environment. In order to access Raspberry Pi or Arduino directly, when the computer needs to be plugged in, there's a small hatch right behind the robot so with that IT can be accessed.

3.1.1. Electrical Simulation - The Base Section of the Robot

In order to run the robot on its wheels there needs to be an electrical infrastructure. Four DC motors and an Arduino are needed for this part of the project. Since the group couldn't manage to obtain the physical version of the required parts, certain alternatives were chosen. The most useful alternative for the group to run this segment was an online circuit building simulator which is called TinkerCad. With this tool the group was allowed to build the base circuit of the robot in order to make it move.

3.1.2. Electronics Parts

1- Arduino Uno

2- DC Motor

3- H Bridge Motor Drive (L293D) Scheme of the Circuit:

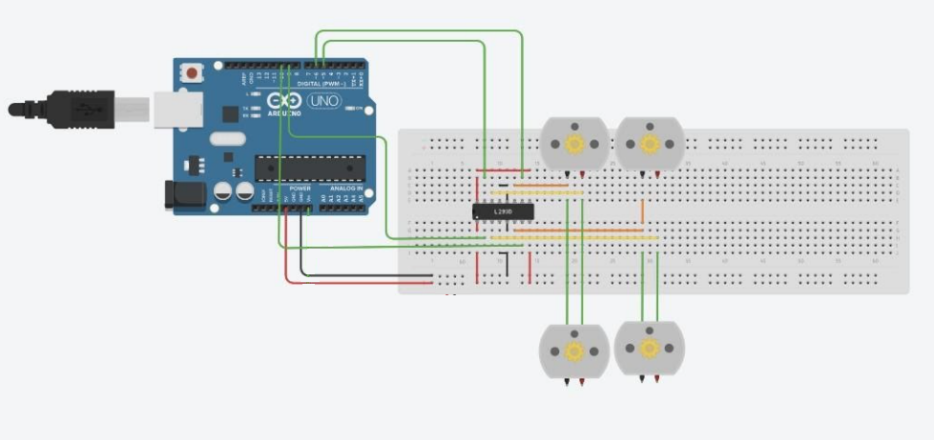
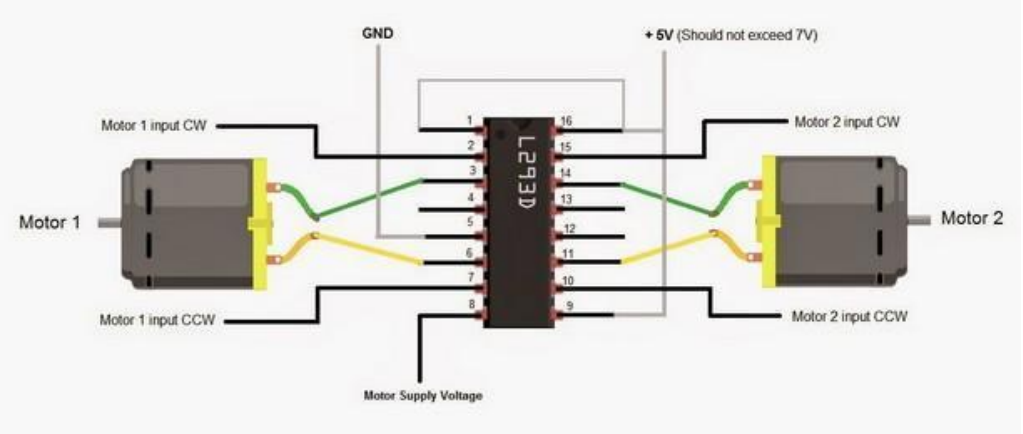


Figure 55. Circuit Scheme of DC Motors

The figure x. showcases the desired circuit for the motion function. It has four DC motors and an H-Bridge that connects them directly to the Arduino. With that circuit the robot will have the desired abilities which are moving in a direction and getting close to other people.

3.1.2.1. How Does It Work

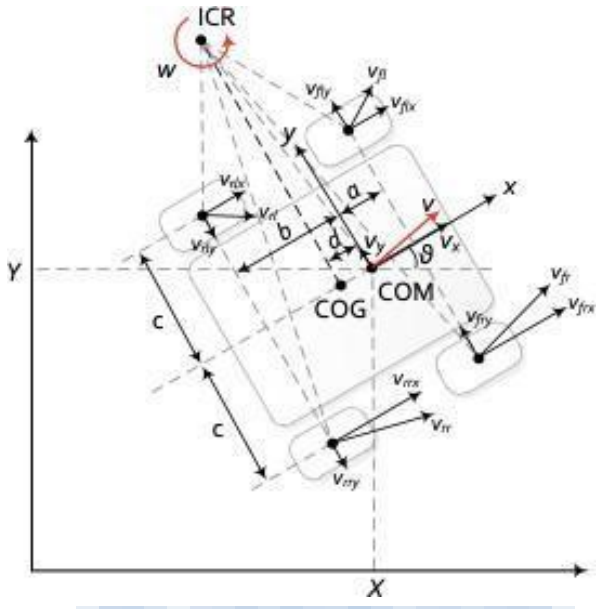
In order to move the base of the robot each wheel needs to be connected to the Dc Motor shown in figure below. Four wheels are equal to four DC Motors. After connecting them to a desired spot on the breadboard they need to be connected to a H Bridge Motor Drive.



3.2. System Dynamics and Control

While modeling the mathematical aspect of the robot, the calculations were considered on only the planer motor. Each wheel- ground contact is a single point and normal forces acting on the wheel-ground contact points are constant depending on the mass of vehicle and gravity. While planning the kinematics the slipping is neglected and the interaction forces between wheel and ground is represented with a conventional friction model. The free-body diagram is displayed below [see figure x] and the main body part which carries the electronic parts of the robot as well is represented with a point mass located at the center (Near the front side of the vehicle) Torque of the wheels are distributed equally on each side and servo drives and also the electric drives dynamics can be neglected.

3.2.1. Kinematics



The Kinematic scheme of the robot platform is shown above The robot configuration vector in coordinate frames is *X*, *Y* and *θ (angular)* which are the position and at the mean time orientation of the vehicle.

$$q = [X\ Y\ Q]^T$$

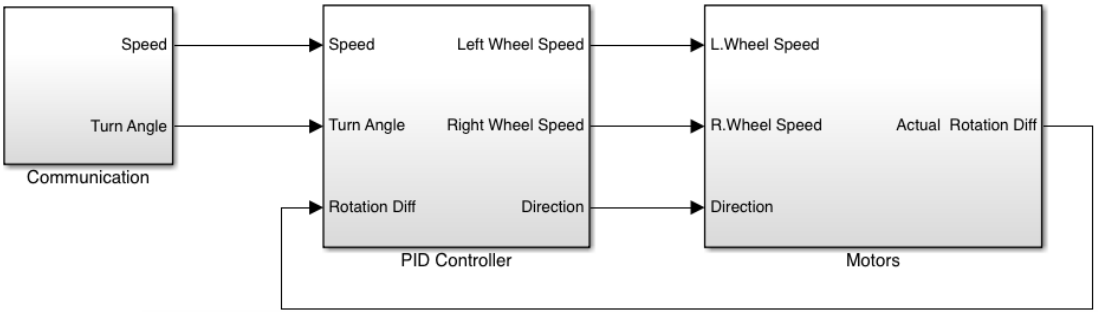
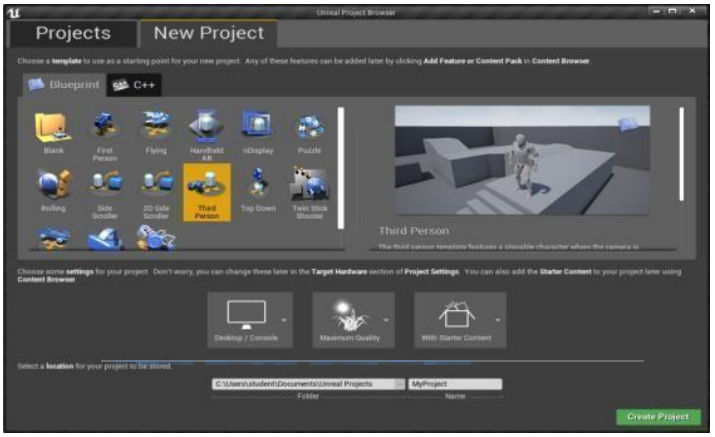


Figure 58. System Dinamics and Control Diagram

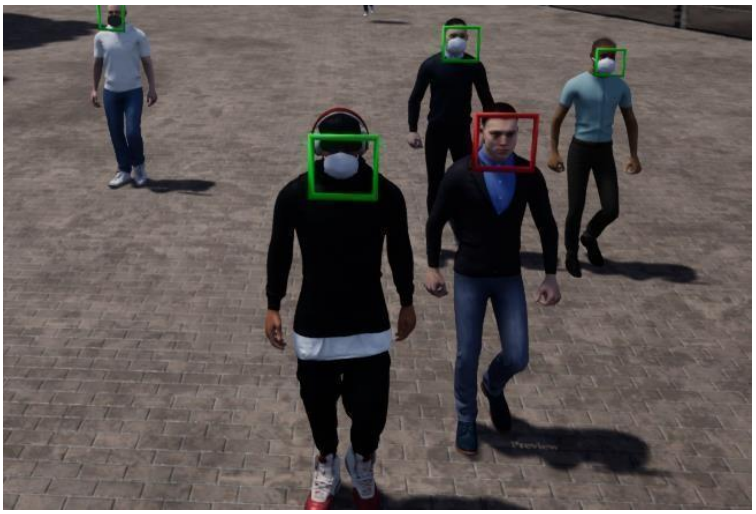
3.3. Creating the Animation



A new project file was created in the Unreal Engine 4 program. In order to show the animation process, an environment that is realistic and has the closest view to the school was designed.

After the environment was made completely suitable, the object with the representation of the robot previously drawn was added to the project. Various objects were used in the animation to show how the project works and what it does.

Human characters have been downloaded from the Mixamo to pass through the school gate turnstiles. The animations of these characters were also downloaded from the same site in accordance with the characters and imported into the project. Since there are no masks in the character models that are open on the internet, masks were placed on each character's face and these masks were fixed on the faces of the characters.



Since there are features such as mask identification, temperature measurement and distance measurement in the robot function, for the part where the animation is seen from the robot's eye;

A green frame was used for people wearing masks and a red frame was used for people not wearing masks.

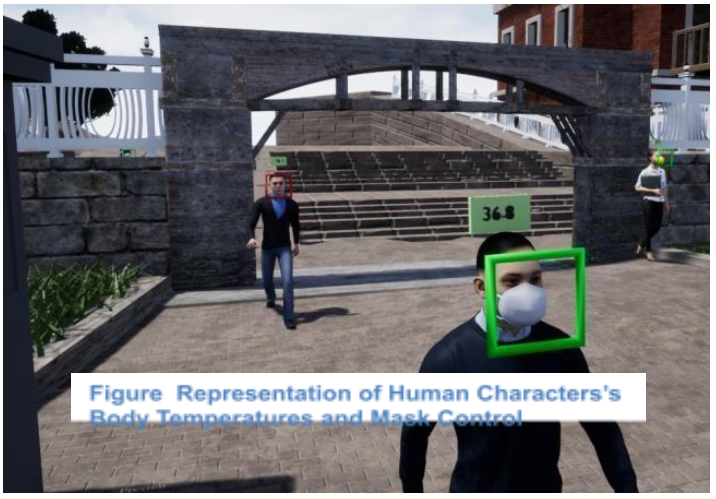
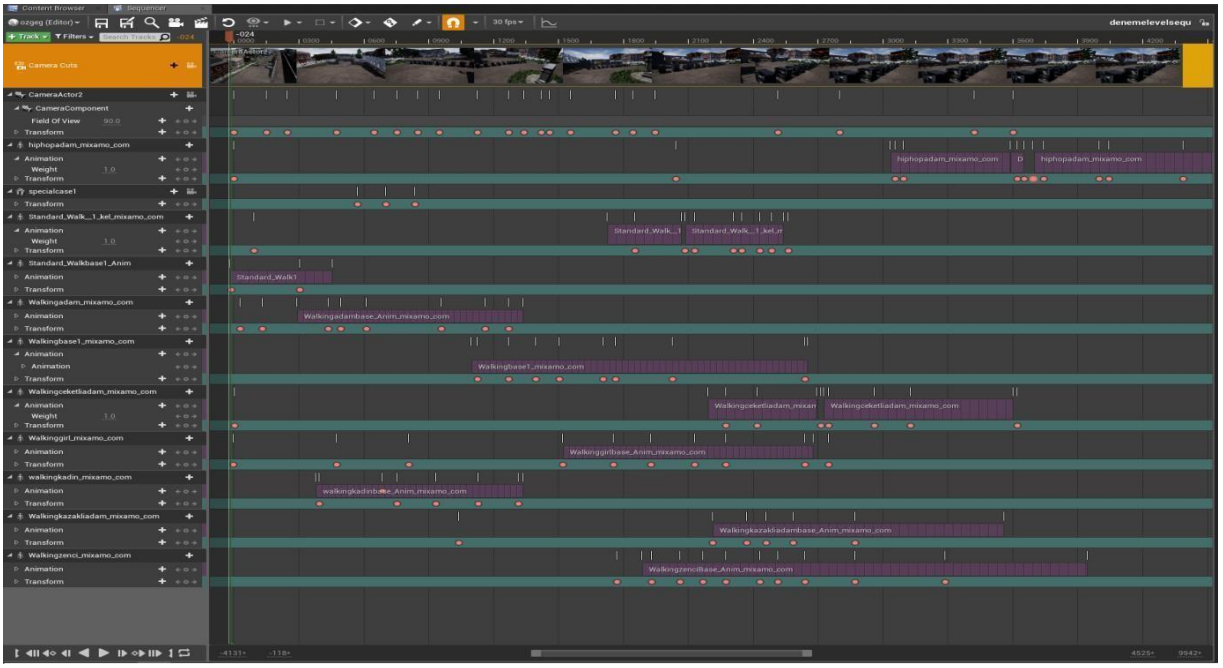


Figure Representation of Human Characters's Body Temperatures and Mask Control

Since the robot will measure the temperature of people, plates with temperature values were placed on people's heads.



The Sequencer Editor gives users the ability to create in-game cinematics with a specialized multi-track editor (similar to Matinee). By creating Level Sequences and adding Tracks, users can define the makeup of each Track, which will determine the content for the scene. Tracks can consist of things like Animations (for animating a character), Transformations (moving things around in the scene), Audio (to include music or sound effects), and several other Track types. — [32]

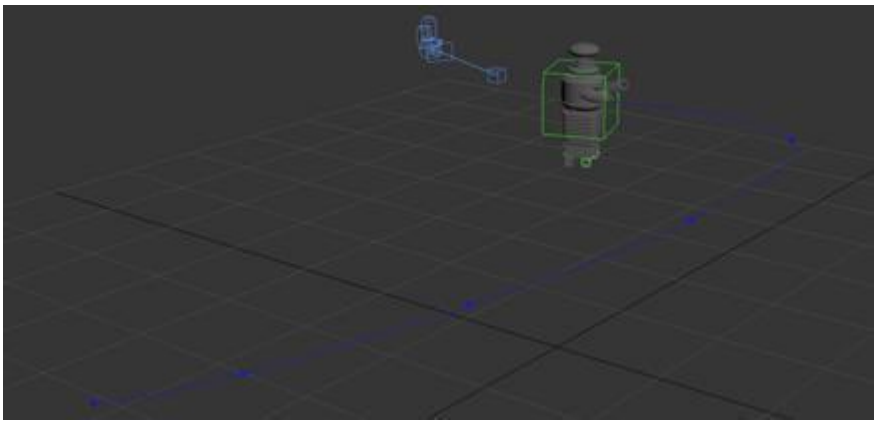
The movement of the camera, robot and people entering the school was designed to show the operation of the project. The positions, speed and movements of each character are designed. The animation was rendered after the editing was completely finished.

3.3.1. Animation of Robot

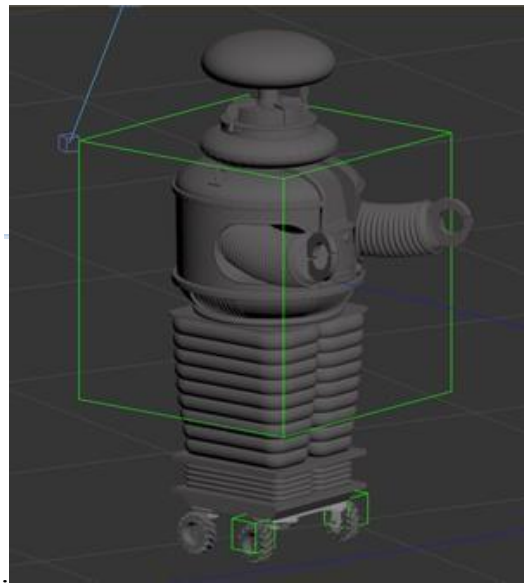
SpecialCase Robot's drawing with SolidWorks was thrown into the 3ds Max app for use in the robot's animation and simulation. 3ds Max is also intended to reflect the robot's step-by- step movements, which will be shown in simulation and animation. In fact, as with other libraries that It receives ready-made, the robot is expected to be mobile in the simulated environment.

The first action taken to get movement to the robot determines the path that the robot must go with the help of a line in advance. Here a blue line was made for the robot and the line representing the movement that the robot would make was attached to the robot. In this

way, the robot learned the way to go after the animation began.



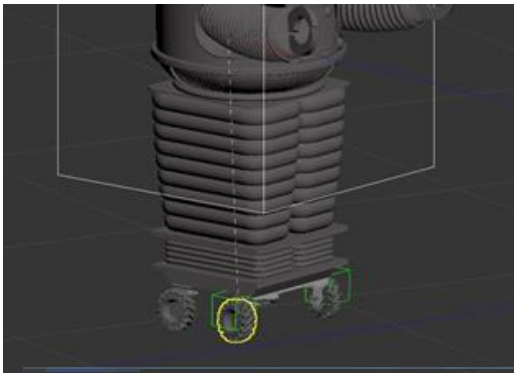
A "Dummy" was created so that it could help the front wheels and the robot's body. In this way, this method was preferred to be able to use separate parts together when moving the robot's body and wheels.



In general, two parts of the robot, one wheel and one robot's body, must have been moved. Since the body is attached with the "Path", the robot's feet can also move in synchrony in accordance with the body's helper when the wheels are attached to the body's helper. After the wheels were fixed into the body, the calculation was made of how much the wheels of the robot should turn in one round. With the calculation of this

(Diameter of the Robot's Wheel*Percent)/36

how many laps it should turn was automatically transferred to wheel movement.



After all the stages were completed, the robot was animated. In each time interval, the position of the robot was animated with the "Timeline" method.

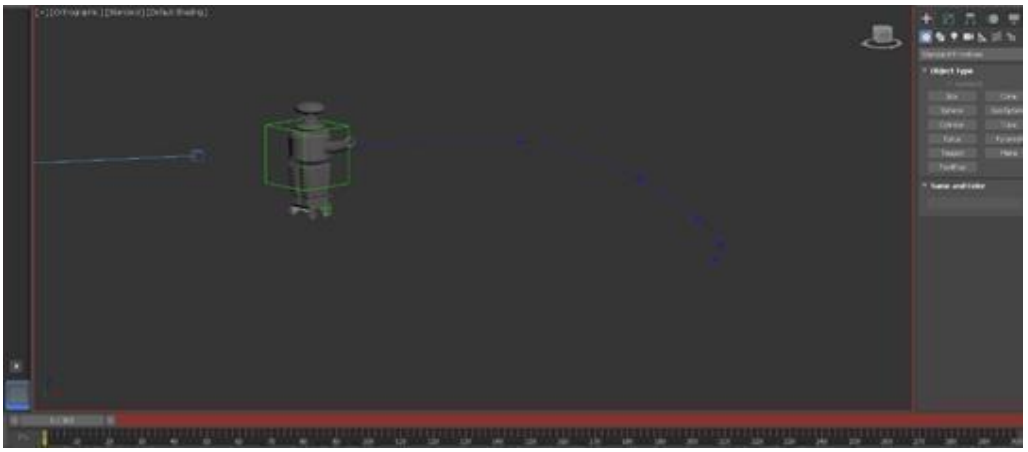
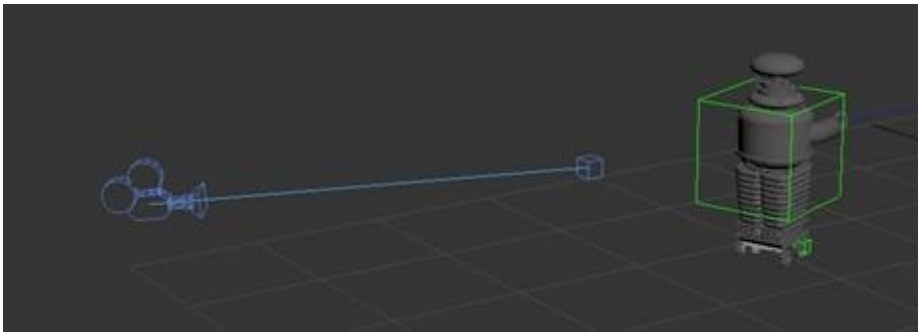


Figure 69. Animating Robot Step

Finally, a camera was installed to show the robot's movements. In this way, a close-up view of the robot's wheels and body was provided.



After the robot's wheels gained motion animation in 3Ds Max, the robot was transferred to Unreal Engine 4 to fully reflect the robot's function.

3.4. Motion Planning

Motion planning, also known as path planning, is a computational problem for finding a set of valid configurations that move an object from source to destination. The term is used in computational geometry, computer animation, robotics, and computer games.

For example, consider driving a mobile robot inside a building to a remote deconstructing point. This task should be done by avoiding walls and without falling down stairs. A motion planning algorithm takes the definition of these tasks as input and generates the speed and rotation commands that are sent to the wheels of the robot. Motion planning algorithms, more joints, more complex tasks, different constraints and uncertainty.

Motion planning has applications in CAD software such as autonomy, automation, and robotic design, as well as applications in other areas such as digital character revitalization, video games, architectural design, robotic surgery, and the study of biological molecules.

3.4.1. Motion of The Robot Under Various Scenarios

The Robot performs its movements on four main legs. Each of the feet is a wheel. The Robot is powered by a self-integrated DC motor. At the same time, it is able to map thanks to the established algorithm and written codes. The robot works with the A* algorithm. According to the algorithm, heights in front of the robot are detected, and when the robot detects these obstacles, it will go around them instead of jumping over them. But as a result of possible sudden changes, the paths that the robot constantly travels may change. One of

these possible situations is that one of the animals living on campus is on the route the robot is taking. In such a situation, if the animal is in motion, the robot may need to wait a few seconds, if the animal is not in motion, it may need to perceive the animal as an obstacle and create a new route around it. Another scenario is that too many people enter campus at the same time. Instead of scanning people who are near the Robot, this time it should look at all the people who come from a long distance and go to them one by one from the one closest to the field of view to the one far away. The Robot can maintain its functions in rainy weather, but can also detect snow masses as obstacles when the weather is snowy. In other words, the robot is designed to be ready for different movements in different scenarios at different times.

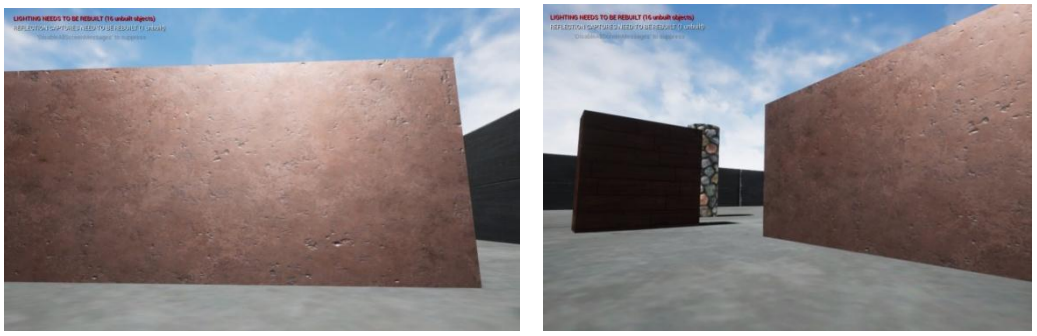


Figure 71. Detecting Obstacle Simulation

In the figures above, the path that the robot will follow in a possible scenario is shown. When the robot encounters a wall or an obstacle, it turns towards the non-obstructed side and continues to move forward.

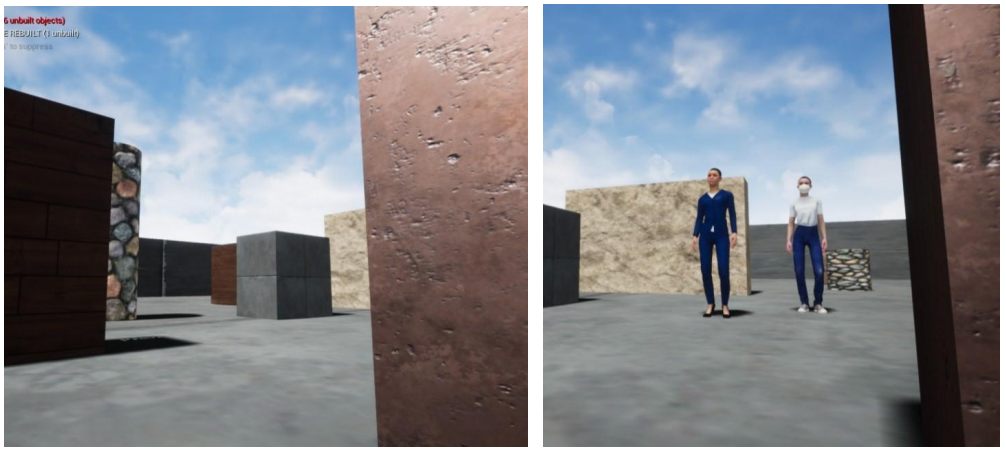


Figure 72. Moving Towards People

In Figure 72, it is shown that the robot moves forward without hitting obstacles and walls and turns towards people when it detects people.

3.5. Power Calculations

The design of mechanical systems also affects the efficiency of the working system. Efficiency is provided by the motor or energizers in the system. The Robot will carry no weight on it. For this reason, calculations will be made according to the robot itself. Calculations are mostly related to the system that will allow the robot to walk.

Arduino Uno: Arduino Uno can also receive power via USB or adapter input. As another option, it can feed from Vin (+) and GND (-) pins. It is aimed that the robot works with the help of adapters to be used. The recommended voltage for Arduino Uno is between 7V-12V. The reason is that the voltage below 7V cannot work stably and the voltage above 12V overheats. [10]

Raspberry Pi: The ideal power supply required for the Raspberry Pi is provided with the help of adapters. If you need to write the necessary amounts of power for Raspberry, it is possible to say values between 5V and 12V. [11]

Battery: The voltage value inside rechargeable batteries is 1.2 V. But using such batteries is not very efficient in terms of device performance. For this reason, the batteries to be used in the project will be batteries that cannot be charged and the voltage value is known as 1.5 V.

Brushed DC motors are the most basic of motor types. In some cases, they can be used together with a reducer, and in some cases they can be used without a reducer. They can be easily driven. These types of DC motors are 12V. [13]

Lidar Sensor: The Lidar sensor is used for self-driving vehicles. Because they are about 5V, it will be enough to use 4 batteries in proportion. [18]

Material	Voltage	Battery Amount
Brushed DC Motor	**12V**	**8**
Arduino	**7V-12V**	**6**
Lidar Sensor	**5V**	**4**
Raspberry Pi	**5V-12V**	**8**

Table 3. Number of Batteries to Meet Energy Needs

The voltages required for the materials to be used are as shown in the table. Assuming that when the amount of battery required for each material is collected, there will be a lot of batteries in it and this can lead to confusion in the internal Assembly, and the cost will increase with the constant renewal of the finished batteries, it was more convenient to use a powerbank instead of a battery.

4. CONCLUSION

The main purpose of the project is to collect all these necessary functions such as fever measurement, social distance measurement, or mask control performed by guards within the scope of covid-19 measures in a robot, thus reducing the likelihood of people assigned to get the virus. The main features of the robot are to detect possible Covid-19 patients and prevent them from entering Istanbul Bilgi University campuses, to measure high fever, which is the first known symptom of the disease, and then to make mask checks and to make distance measurements to keep people's health safe on campus at all times. As a result of the research made for the project, various algorithms that can provide these features of the robot have been established and these algorithms have been made to work in suitable environments. In order for the robot to move unmanned, the necessary algorithms have been established to map the environment in which it is located and to perceive the obstacles in front of it and provide its movement accordingly, and the necessary system has been established for the robot to move on its own. The ways necessary for the robot to attain the desired functions and features have been followed, and functions such as mask detection, temperature measurement, distance measurement, which are desired and also necessary, have been operated. It is obvious that a safer environment will be offered to people as a result of the continuity of the project. Therefore, the construction and implementation of the project is highly recommended.

4.1. Results

The main functions mentioned in the project are fever measurement, social distance measurement, mask control, face detection, obstacle detection and mapping. In real life, a number of basic algorithms were established first, after the necessary research and correct and incorrect attempts were made, scenarios were determined and pseudocodes with reality were written in the general framework, and then the codes to be implemented were tested in a computer environment and with physical components. As a result of the trials, a positive response was received and the most appropriate option was selected and integrated into the project. In the mapping section, the A star algorithm was considered the most appropriate option for the project. In this Section, people are perceived by the robot, the robot that chooses between the distances of people will first go to all the people

who enter the campus, including the nearest one, and will control the fire and mask while measuring the distance between the people who enter the campus. If there is no robot mask that goes next to people, it will also warn you to wear a mask with sound and writing. At the same time, he will not allow the person he is going with to enter the campus if he measures his temperature and has a temperature above 37.5 degrees. In this way, the risk of transmission within the campus will be minimized. This robot, whose scenario is described, has also been transferred to a simulation environment where all functions are shown in detail.

4.2. Discussion and Future Work

At this stage, we have made possible estimates for the part of the project if something does not go well. The first of these is that the wheels that can be detected by the eye may be small, and there may be a lot of torque due to the small wheel size. If there is too much torque, it may not comply with the working principle of the sensors. Secondly, it may be necessary to make the plastic accents metal, because the robot is a large robot, so it will be more resistant to any possible impact, such as falling. Third, the power of the processor may not be enough, so raspberry, at some points, some codes that we want to write our codes on may be too much for the system and may work very slowly or not work at all. We can use a mini PC to eliminate this problem. The fourth possible possibility is to put an aluminum or copper block on the robot or a fan to prevent the robot from overheating. Plus, the lithium batteries we plan to use in the robot can work up to a maximum of 0 degrees and not work at minus degrees. If we extend the copper pipes that will be in the robot's sensors and wrap them around the batteries, we may have heated the batteries. Our last possible guess is that we may need to set up a station to charge the robot. In this way, it can charge itself in the charging station.

4.3. Social, Environmental and Economical Impact

This project has an important social and economic position. This robot, which will be designed to manage the pandemic process that affects the health of the whole world more intelligently, will measure the distance required to prevent contamination between people and warn it. It will also be used for thermometry to enter shopping malls, a cafe, a business center, and many other areas. If the first of these two features is to be considered,

the social contribution of two creatures to stimulate each other may cause certain polemics in some cases, while being warned about distance by an inanimate robot may become more acceptable for society. In addition, rather than calculating the distance between two people by a person, it is processed by the robot, which is processed in its software and will do this task without error, which will ensure that the distance rule is fulfilled smoothly. The other effect is that it measures temperature when entering public spaces. It can be said that there is a negative situation in terms of their health, as both the employees and the security personnel performing this task are in contact with many people during the day to measure the fever. If these robots are started to be used for this purpose, the health of the employees will be protected. This situation can also be evaluated as an economic effect because a certain fee is paid to them to perform this task. If the functions that the robot is intended to perform are consistently performed by an officer, that employee must be paid a salary on a regular basis. Instead, the use of the robot purchased with a one-time budget to be created will be lower in cost. In short, thanks to this robot, it will have an important place in many aspects of the world in order to get through this process in a short time.

4.4. Cost Analysis

In this project, the daily income per person has been calculated as 150 TL with 4 full-day shifts per week. This is calculated as 2400 TL per person per month for four engineering students, and the total monthly cost is 9600 TL. The total salary of the four engineers for the project, which will last for 9 months, is 86400 TL. When 3739.21 TL material costs are added to this account, the budget allocated for the entire project becomes 90139.21 TL.

Material	Unit Price	Amount	Total Price
MLX 90614 Digital Non-Contact Infrared Thermometer	65 TL	1	65 TL
OLED Display I2C	35 TL	1	35 TL
Arduino Uno Clone	65 TL	2	130 TL
Breadboard	6 TL	4	24 TL
Mabuchi RS-380PH Gearless DC Motor	20.27 TL	4	81.08 TL
Raspberry Pi 4 Model B - 8GB	906 TL	1	906 TL
RGB Led	1 TL	1	1 TL
Raspberry Pi Camera Board V2.1 (8MP,1080P)	388.65 TL	1	388.65 TL
L293D Motor Driver Integrated Dip-16	9.12 TL	1	9.12 TL
Lidar Sensor	1651.59 TL	1	1651.59TL
20000 mAh Powerbank	147.59 TL	3	447.77 TL
TOTAL			3739.21TL

4.5. Standards

4.5.1. Raspberry Pi 4 Model B - Standards

The Raspberry Pi 4 Model B is the most recent addition to the popular Raspberry Pi computer line. When compared to the previous-generation Raspberry Pi 3 Model B+, it offers ground-breaking improvements in processing speed, multimedia performance, memory, and connection while maintaining backwards compatibility and similar power consumption.

A high-performance 64-bit quad-core processor, dual-display support at resolutions up to 4K via a pair of micro-HDMI ports, hardware video decode at up to 4Kp60, up to 8GB of RAM, dual-band 2.4/5.0 GHz wireless LAN, Bluetooth 5.0, Gigabit Ethernet, USB 3.0, and PoE capability are among the product's key features (via a separate PoE

HAT add-on).

The recommended operating system for routine use on a Raspberry Pi is Raspberry Pi OS (formerly known as Raspbian). Raspberry Pi Imager is a simple and quick way to install the Raspberry Pi OS and other operating systems to a microSD card that can then be used with users current Raspberry Pi. [33]

- Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz
- 2GB, 4GB or 8GB LPDDR4-3200 SDRAM (depending on model)
- 2.4 GHz and 5.0 GHz IEEE 802.11ac wireless, Bluetooth 5.0, BLE
- Gigabit Ethernet
- 2 USB 3.0 ports; 2 USB 2.0 ports.
- Raspberry Pi standard 40 pin GPIO header (fully backwards compatible with previous boards)
- 2 × micro-HDMI ports (up to 4kp60 supported)
- 2-lane MIPI DSI display port
- 2-lane MIPI CSI camera port
- 4-pole stereo audio and composite video port
- H.265 (4kp60 decode), H264 (1080p60 decode, 1080p30 encode)
- OpenGL ES 3.1, Vulkan 1.0
- Micro-SD card slot for loading operating system and data storage
- 5V DC via USB-C connector (minimum 3A*)
- 5V DC via GPIO header (minimum 3A*)
- Power over Ethernet (PoE) enabled (requires separate PoE HAT)
- Operating temperature: 0 – 50 degrees C ambient

Specifications:

Processor: Broadcom BCM2711, quad-core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz

Memory: 1GB, 2GB, 4GB or 8GB LPDDR4(depending on model) with on-die ECC

Connectivity: 2.4 GHz and 5.0 GHz IEEE 802.11b/g/n/ac wireless LAN, Bluetooth

5.0, BLEGigabit Ethernet 2 × USB 3.0 ports2 × USB 2.0 ports.

GPIO: Standard 40-pin GPIO header (fully backwards-compatible with previous boards)

Video & Sound: 2 × micro HDMI ports (up to 4Kp60 supported)2-lane MIPI DSI display port2-lane MIPI CSI camera port4-pole stereo audio and composite video port

Multimedia: H.265 (4Kp60 decode);H.264 (1080p60 decode, 1080p30 encode);OpenGL ES, 3.0 graphics

SD Card Support: Micro SD card slot for loading operating system and data storageInput power:5V DC via USB-C connector (minimum 3A1)5V DC via GPIO header (minimum 3A1) Power over Ethernet (PoE)–enabled(requires separate PoE HAT) Environment:Operating temperature 0–50ºC.

4.5.2. Arduino Uno Clone - Standards

The ATmega328-based Arduino Uno Clone is a microcontroller board. There are 14 digital input/output pins, 6 analog inputs, a 16 MHz ceramic resonator, a USB connector, a power jack, an ICSP header, and a reset button on this board. It comes with everything you'll need to get started with the microcontroller; simply plug it into a computer with a USB cable or power it with an AC-to-DC adapter or battery. [34]

Microcontroller: ATmega328P-AU

Operating Voltage: 5V

Input Voltage (Recommended): 7-12V

Input Voltage (Limits): 6-20V

Digital I/O Pins: 14 (of which 6 provide PWM output)

Analog Input Pins: 6

DC Current per I/O Pin: 40 mA

DC Current for 3.3V Pin: 50 mA

Clock Speed: 16 MHz

4.5.3. Unreal Engine - Standards

In Epic Games, there are few smooth coding necessities and conventions. Following the coding necessities is mandatory. Code conventions are critical to programmers for a few reasons: 80% of the lifetime rate of a piece of software program software goes to maintenance. Hardly any software program software is maintained for its entire lifestyles with the useful resource of the use of the proper author. Code conventions beautify the readability of the software program software, allowing engineers to apprehend new code greater, fast and thoroughly. If we're decided to show deliver code to mod community developers, we want it to be understood with out difficulty. Many of these conventions are sincerely required for cross-compiler compatibility. The coding necessities under are C++-centric, but the spirit of the necessities are expected to be located irrespective of which language is used. A section may additionally provide same guidelines or exceptions for particular languages in which it's far applicable. [35]

4.5.4. IFR (International Federation of Robotics)

International Standards on robotics are prepared with the International Organization for Standardization (ISO). Standards include safety, standard overall performance criteria, modularity, and vocabulary. The International Organization for Standardization (ISO) is the worldwide federation of national necessities organizations. Standards concerning robots are prepared with the useful resource of the usage of ISO Technical Committee 299 with the title “Robotics". Safety standardization is a completely critical trouble at ISO/TC 299. Safety necessities were developed for the economic robotics sector (ISO 10218-1, ISO 10218-2, ISO/TS 15066) further to for the non-industrial (service) robotics sector (ISO 13482) [36]

4.5.5. Ethical Standards for Robotics Engineers

There are many expert codes of ethics for plenty of special professions from many special expert groups. These codes of ethics for the maximum component offer very

comparable tips for the organization's contributors to follow. The groups maximum carefully associated with robotics engineering, the IEEE, ASME, and ACM, every have codes with many factors that are immediately relevant to robotics engineering. Robotics engineering, however, can even gift new and specific challenges, in large part because of the capacity autonomy of robots and the extent of interplay with people that robots will surely achieve. These problems are addressed withinside the outer edge in those different codes of ethics, however a code beneficial for robotics engineers must cope with those conditions extra immediately. Ideally, this code of ethics must additionally encompass factors extra precise to the conditions which a robotics engineer is in all likelihood to encounter, just like how the ACM code of ethics offers steering precise to laptop scientists. This enables to make clear the code for the meant target target market and makes the purpose at the back of the code extra clear. Specific examples of moral conditions precise to robotics engineers get up in most cases from the self reliant choice making that usually defines robots. The maximum arguable scenario that a robotics engineer can be faced with is the layout of absolutely self reliant robots with deadly capabilities. This scenario asks the engineer to make the moral choice of whether or not or now no longer to make a robot a good way to determine for itself if and whilst to take human life. A much less debatable example, despite the fact that nevertheless similarly precise to robotics engineering, is the manufacturing and layout of independent motor vehicles. Although independent vehicles had been in improvement for lots of years, they have now no longer but been located withinside the palms of the general public. This is due to the fact any independent entity will unavoidably be anticipated to make a mistake, whether or not in choice-making or merely mechanically. Such a mistake may be fatal, and as such the general public desires to recognise wherein the duty for a robot's choice will lie. A robotics engineer desires to determine whilst a product is secure and geared up to be utilized by the general public and what education or qualification the man or woman customers need to have. Another instance precise to robotics engineers is that of privateness problems and family robots with imaginative and prescient structures or microphones. Such robots should save nonpublic facts without the Code of Ethics for Robotics Engineers 21 Knowledge or consent of the owner, and these facts might be accessed through people with awful intentions. A robotics engineer should ensure that privateness rights are reputable and protected. [37

APPENDICES

A. Face and Mask Detection Function's Codes in JavaScript;

```
<div>Teachable Machine Image Model - p5.js and ml5.js</div>

<script
src="https://cdnjs.cloudflare.com/ajax/libs/p5.js/0.9.0/p5.min.js"></script>

<script
src="https://cdnjs.cloudflare.com/ajax/libs/p5.js/0.9.0/addons/p5.dom.min.js"
></script>

<script src="https://unpkg.com/ml5@latest/dist/ml5.min.js"></script>

<script type="text/javascript">

src="https://unpkg.com/ml5@latest/dist/ml5.min.js"

let video;

let label = "Checking!";

let classifier;

let modelURL = 'https://teachablemachine.withgoogle.com/models/MBXbP6W-9/';


//Loading the model

function preload() {

  classifier = ml5.imageClassifier(modelURL + 'model.json');

}

function setup() {

  createCanvas(500, 400);

  // Create the video

  video = createCapture(VIDEO);

  video.hide();


  // Starting classifying

  classifyVideo();

}

function classifyVideo() {
```

```
  classifier.classify(video, gotResults);
}

function draw() {
  background(0);
 // Draw the video
  image(video, 0, 0);

  textSize(50);
  textAlign(CENTER, CENTER);
  fill(255);
  text(label, width / 2, height - 16);

  // Picking an emoji, the "default" is train
  let simge = "⌚";
  if (label == "Mask") {
      simge = "✔";
  } else if (label == "NoMask") {
      simge = "❌";
  }

  // Drawing the emoji
  textSize(50);
  text(simge, width - 30, height - 50);
}
// Getting the classification
function gotResults(error, results) {

  if (error) {
    console.error(error);
```

```
    return;
  }
  // Storing the label and classifying again
  label = results[0].label;
  classifyVideo();
}
```

B. Social Distance Measurement Function's Codes in Python

```
import cv2
from scipy.spatial import distance as dist

cap = cv2.VideoCapture(0)
face_model = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')

while True:
      status , photo = cap.read()
      face_cor = face_model.detectMultiScale(photo)
      l = len(face_cor)
      photo = cv2.putText(photo, str(len(face_cor))+" Face", (50, 50),
cv2.FONT_HERSHEY_SIMPLEX,1, (255, 0, 0) , 2, cv2.LINE_AA)
      stack_x = []
      stack_y = []
      stack_x_print = []
      stack_y_print = []
      global D

      if len(face_cor) == 0:
      pass
      else:
```

```
	for i in range(0,len(face_cor)):
		x1 = face_cor[i][0]
		y1 = face_cor[i][1]
		x2 = face_cor[i][0] + face_cor[i][2]
		y2 = face_cor[i][1] + face_cor[i][3]

		mid_x = int((x1+x2)/2)
		mid_y = int((y1+y2)/2)
		stack_x.append(mid_x)
		stack_y.append(mid_y)
		stack_x_print.append(mid_x)
		stack_y_print.append(mid_y)

		photo = cv2.circle(photo, (mid_x, mid_y), 3 , [255,0,0] , -1)
		photo = cv2.rectangle(photo , (x1, y1) , (x2,y2) , [0,255,0] , 2)

	if len(face_cor) == 2:
		D = int(dist.euclidean((stack_x.pop(), stack_y.pop()),
(stack_x.pop(), stack_y.pop())))
		photo = cv2.line(photo, (stack_x_print.pop(),
stack_y_print.pop()), (stack_x_print.pop(), stack_y_print.pop()), [0,0,255],
2)
	else:
		D = 0

	if D<1500 and D!=0:
		import winsound
		frequency = 2500  # Set Frequency To 2500 Hertz
		duration = 1000  # Set Duration To 1000 ms == 1 second
		winsound.Beep(frequency, duration)

		photo = cv2.putText(photo, "!!MOVE AWAY!!", (100, 100),
```

```
cv2.FONT_HERSHEY_SIMPLEX,2, [0,0,255] , 4)


      photo = cv2.putText(photo, str(D/10) + " cm", (300, 50),
cv2.FONT_HERSHEY_SIMPLEX,

                  1, (255, 0, 0) , 2, cv2.LINE_AA)


      cv2.imshow('SpecialCase Robot' , photo)

      if cv2.waitKey(100) == 13:

            break


cv2.destroyAllWindows()
```

C. Temperature Detection Codes in Arduino

```
#include <Adafruit_GFX.h>

#include <Adafruit_SSD1306.h>

#include <Wire.h>

#include <Adafruit_MLX90614.h>

#include <Fonts/FreeMonoBold18pt7b.h>


Adafruit_SSD1306 display(128, 64);

Adafruit_MLX90614 mlx = Adafruit_MLX90614();


const unsigned char PROGMEM frame0 [] = {0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x07, 0xF8, 0x00, 0x00, 0x00, 0x00,
0x3C, 0x0E, 0x00, 0x00, 0x00, 0x00, 0x60, 0x43, 0x80, 0x00, 0x00, 0x01, 0x8A,
0x10, 0xC0, 0x00, 0x00, 0x03, 0x20, 0x00, 0x70, 0x00, 0x00, 0x06, 0x00, 0x00,
0x1C, 0x00, 0x00, 0x0D, 0x00, 0x00, 0x04, 0x00, 0x00, 0x18, 0x00, 0x00, 0x04,
0x00, 0x00, 0x30, 0x90, 0x00, 0x08, 0x00, 0x00, 0x20, 0xCA, 0x00, 0x0C, 0x00,
0x00, 0x21, 0x61, 0x70, 0xC4, 0x00, 0x00, 0x61, 0x1C, 0x07, 0xC4, 0x00, 0x00,
0x43, 0x07, 0xFC, 0x66, 0x00, 0x00, 0x42, 0x00, 0x00, 0x22, 0x00, 0x00, 0x46,
0x00, 0x00, 0x22, 0x00, 0x00, 0x44, 0x00, 0x00, 0x22, 0x00, 0x00, 0x44, 0xFC,
0x3F, 0x12, 0x00, 0x00, 0x64, 0x04, 0x01, 0x36, 0x00, 0x00, 0x2C, 0x00, 0x00,
0x14, 0x00, 0x00, 0x28, 0x30, 0x0C, 0x1C, 0x00, 0x00, 0x78, 0x30, 0x06, 0x1A,
0x00, 0x00, 0x58, 0x70, 0x0E, 0x1A, 0x00, 0x00, 0x4E, 0x20, 0x04, 0x76, 0x00,
0x00, 0x6B, 0x80, 0x01, 0xD4, 0x00, 0x00, 0x28, 0xF0, 0x0F, 0x14, 0x00, 0x00,
0x28, 0x1F, 0xF8, 0x14, 0x00, 0x00, 0x38, 0x00, 0x00, 0x1C, 0x00, 0x00, 0x19,
0x00, 0x00, 0x18, 0x00, 0x00, 0x08, 0x80, 0x01, 0x10, 0x00, 0x00, 0x08, 0x28,
0x10, 0x10, 0x00, 0x00, 0x08, 0x01, 0x40, 0x10, 0x00, 0x00, 0x0D, 0x00, 0x00,
```

```
0xB0, 0x00, 0x00, 0x04, 0x40, 0x02, 0x20, 0x00, 0x00, 0x02, 0x10, 0x14, 0x40,
0x00, 0x00, 0x03, 0x01, 0x40, 0xC0, 0x00, 0x00, 0x01, 0x80, 0x01, 0x80, 0x00,
0x00, 0x00, 0xC0, 0x03, 0x00, 0x00, 0x00, 0x00, 0x30, 0x0C, 0x00, 0x00, 0x00,
0x00, 0x1E, 0x38, 0x00, 0x00, 0x00, 0x00, 0x03, 0xC0, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00};

int GREEN=2;

int RED=7;

float reading[30];

float total ;

float temperature ;

void setup() {

   pinMode(GREEN,OUTPUT);

   pinMode(RED,OUTPUT);


  delay(100);

  display.begin(SSD1306_SWITCHCAPVCC, 0x3C);

  display.clearDisplay();

  display.setTextColor(WHITE);

  mlx.begin();

}

int xx =42;

int yy=20;

int tt=0;


void loop() {

  display.clearDisplay();

total=0;

for (int i=0; i<30; i++)

{

  float deger =mlx.readObjectTempC();

reading[i]= deger *1;

total = total + reading[i];
```

```
delay (5);
}
temperature = total / 30.0;

  if(temperature>30){
  display.clearDisplay();
  display.setFont();
  display.setCursor(10,10);  // (x,y)
  display.println("YOUR BODY TEMPERATURE");  // Text or value to print
  display.setCursor(14,2);  // (x,y)
  display.println(".");  // Text or value to print
  display.setCursor(17,2);  // (x,y)
  display.println(".");  // Text or value to print
  display.setCursor(88,4);  // (x,y)
  display.println("-");  // Text or value to print

  display.setFont(&FreeMonoBold18pt7b);  // Set a custom font
  display.setCursor(10,55);  // (x,y)
  display.println(temperature,1);
   display.setCursor(100,55);  // (x,y)
  display.println("C");  // Text or value to print
  display.setCursor(88,37);  // (x,y)
  display.println(".");  // Text or value to print

  }
  if(mlx.readObjectTempC()<=30){

display.clearDisplay();
  display.setFont();
  display.setCursor(10,10);  // (x,y)
```

```
  display.println("FEVER CONTROL");  // Text or value to print

//display.clearDisplay();

display.drawBitmap(xx, yy,frame0,48,48, 1);

display.display();

delay(tt);
```

D. Mapping and Obstacle Detection Codes in Python

A* Path Finding

```
import pygame

from queue import PriorityQueue

import ImageProcess2
import Numpy2

WIDTH = 350
WIN = pygame.display.set_mode((WIDTH,WIDTH))
pygame.display.set_caption("A* Path Finding Algorithm")

RED = (255, 0, 0)
GREEN = (0, 255, 0)
BLUE = (0, 255, 0)
YELLOW = (255, 255, 0)
WHITE = (255, 255, 255)
BLACK = (0, 0, 0)
PURPLE = (128,0,128)
ORANGE = (255, 165, 0)
GREY = (128, 128, 128)
TURQUOISE = (64,224,208)

class Spot:
    def __init__(self, row, col, width, total_rows):
        self.row = row
        self.col = col
        self.x = row * width
        self.y = col * width
        self.color = WHITE
        self.neighbors = []
        self.width = width
        self.total_rows = total_rows

    def get_pos(self):
        return self.row, self.col

    def is_closed(self):
        return self.color == RED

    def is_open(self):
        return self.color == GREEN
```

```
    def is_barrier(self):
        return self.color == BLACK

    def is_start(self):
        return self.color == ORANGE

    def is_end(self):
        return self.color == TURQUOISE

    def reset(self):
        self.color = WHITE

    def make_start(self):
        self.color = ORANGE

    def make_closed(self):
        self.color = RED

    def make_open(self):
        self.color = GREEN

    def make_barrier(self):
        self.color = BLACK

    def make_end(self):
        self.color = TURQUOISE

    def make_path(self):
        self.color = PURPLE

    def draw(self,win):
        pygame.draw.rect(win, self.color, (self.x, self.y, self.width,
self.width))

    def update_neighbors(self, grid):
        self.neighbors = []
        if self.row < self.total_rows - 1 and not grid[self.row +
1][self.col].is_barrier(): # DOWN
            self.neighbors.append(grid[self.row + 1][self.col])

        if self.row > 0 and not grid[self.row - 1][self.col].is_barrier(): #
UP
            self.neighbors.append(grid[self.row - 1][self.col])

        if self.col < self.total_rows - 1 and not grid[self.row][self.col +
1].is_barrier(): # RIGHT
            self.neighbors.append(grid[self.row][self.col + 1])

        if self.col > 0 and not grid[self.row][self.col - 1].is_barrier(): #
LEFT
            self.neighbors.append(grid[self.row][self.col -1])

    def _lt_(self, other):
        return False

def h(p1, p2):
    x1, y1 = p1
    x2, y2 = p2
    return abs(x1 - x2) + abs(y1 - y2)

def reconstruct_path(came_from, current, draw):
```

```
    while current in came_from:
        current = came_from[current]
        current.make_path()
        draw()

def algorithm (draw, grid, start, end):
    count = 0
    open_set = PriorityQueue()
    open_set.put((0, count, start))
    came_from = {}
    g_score = {spot: float("inf") for row in grid for spot in row}
    g_score[start] = 0
    f_score = {spot: float("inf") for row in grid for spot in row}
    f_score[start] = h(start.get_pos(), end.get_pos())

    open_set_has = {start}

    while not open_set.empty():
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                pygame.quit()

        current = open_set.get()[2]
        open_set_has.remove(current)

        if current == end:
            reconstruct_path(came_from, end, draw)
            end.make_end()
            return True

        for neighbor in current.neighbors:
            temp_g_score = g_score[current] + 1

            if temp_g_score < g_score[neighbor]:
                came_from[neighbor] = current
                g_score[neighbor] = temp_g_score
                f_score[neighbor] = temp_g_score + h(neighbor.get_pos(),
end.get_pos())
                if neighbor not in open_set_has:
                    count += 1
                    open_set.put((f_score[neighbor], count, neighbor))
                    open_set_has.add(neighbor)
                    neighbor.make_open()

        draw()

        if current != start:
            current.make_closed()

    return False

def make_grid(rows, width):
    grid = []
    gap = width // rows
    for i in range(rows):
        grid.append([])
        for j in range(rows):
            spot = Spot(i, j, gap, rows)
            grid[i].append(spot)

    return grid
```

```
def draw_grid(win, rows, width):
    gap = width // rows
    for i in range(rows):
        pygame.draw.line(win, GREY, (0, i * gap), (width, i * gap))
        for j in range(rows):
            pygame.draw.line(win, GREY, (j * gap, 0), (j * gap, width))



def draw(win, grid, rows, width):
    win.fill(WHITE)

    for row in grid:
        for spot in row:
            spot.draw(win)

    draw_grid(win, rows, width)
    pygame.display.update()



def get_clicked_pos(pos, rows, width):
    gap = width // rows
    y, x = pos

    row = y // gap
    col = x // gap

    return row, col



def main(win, width):
    ROWS = 35
    grid = make_grid(ROWS, width)

    start = None
    end = None
    end2 = None
    dist2 = 0
    x1 = 0
    x2 = 0
    y1 = 0
    y2 = 0

    endX = 0
    endY= 0
    startX = 0
    startY = 0

    dist=0

    run = True
    started = False
    spot = grid[ImageProcess2.a[0]][ImageProcess2.a[1]]
    spot.make_barrier()
```

```
spot = grid[ImageProcess2.a[2]][ImageProcess2.a[3]]
spot.make_barrier()
spot = grid[ImageProcess2.a[4]][ImageProcess2.a[5]]

spot = grid[value1][value2]
spot.make_end()
spot = grid[value1][value2]
spot.make_end()
end = spot

while run:
    draw(win, grid, ROWS, width)
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            run = False

        if started:

            continue

        if pygame.mouse.get_pressed()[0]: # LEFT
            pos = pygame.mouse.get_pos()
            row, col = get_clicked_pos(pos, ROWS, width)
            spot = grid[row][col]

            if not start:
                start = spot
                start.make_start()
                start.make_end()



            elif spot != start and spot !=end:

                start = spot
                start.make_end()


            elif spot !=end:

                start = spot
                spot.make_end()



        elif pygame.mouse.get_pressed()[2]: # RIGHT
            pos = pygame.mouse.get_pos()
            row, col = get_clicked_pos(pos, ROWS, width)
            spot = grid[row][col]
            spot.reset()
            if spot == start:
                start == None
            elif spot == end:
                end = None

        if event.type == pygame.KEYDOWN:
            if event.key == pygame.K_SPACE and not started:
                for row in grid:
                    for spot in row:
                        spot.update_neighbors(grid)
```

```
                    algorithm(lambda : draw(win, grid, ROWS, width), grid,
start, end)



    pygame.quit()

main(WIN, WIDTH)
```

Image Processing

```
from typing import List

import cv2

import numpy as np

img = cv2.imread('opencvpicture.png')

hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)

lower_range = np.array([84, 98, 0])

upper_range = np.array([179, 255, 255])

mask = cv2.inRange(hsv, lower_range, upper_range)

cv2.imshow('image', img)

cv2.imshow('mask', mask)

(contours,_) = cv2.findContours(mask.copy(), cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)

center = None

if len(contours) > 0:
   c = max(contours, key = cv2.contourArea)
   rect = cv2.minAreaRect(c)
   ((x,y), (width,height), rotation) = rect

   s = "x: {}, y: {}, width: {}, height:{}, rotation:{}".format(np.round(x),
np.round(y), np.round(width), np.round(height), np.round(rotation))

   print(s)

   box = cv2.boxPoints(rect)

   box = np. int64(box)

   #a = np.array([], np.int16)

   a: List[int] = [0]*6

   i = 0

   for c in contours:

      M = cv2.moments(c)
```

```
        cX = int(M["m10"] / M["m00"])

        cY = int(M["m01"] / M["m00"])

        a[i] = cX

        a[i+1] = cY

        cv2.drawContours(img, [box], 0, (0, 255, 255), 2)

        cv2.circle(img, (cX, cY), 5, (255, 0, 255), -1)

        #cv2.putText(img, s, (cX - 20, cY - 20),

        #cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 0, 0), 2)

        i = i + 2

        print(a)

    cv2.imshow("Original Tespit", img)

cv2.waitKey(0)
cv2.destroyAllWindows
```

Numpy

```
import ImageProcess2

def euclidian(x1, x2,y1,y2):
    xnew = (x1-x2)**2
    ynew = (y1-y2)**2
    sum_xy = xnew + ynew
    sonuc = (sum_xy)**(1/2)
    return sonuc

value1 =
euclidian(ImageProcess2.x,ImageProcess2.y,ImageProcess2.a[0],ImageProcess2.a[
1])
value2 =
euclidian(ImageProcess2.x,ImageProcess2.y,ImageProcess2.a[2],ImageProcess2.a[
3])

print("values of distances: ",value1,value2)
print(min(value1,value2))
minValue = min(value1,value2)
```

REFERENCES

[1] GERRETSEN, Isabelle, Robots are joining the fight against coronavirus in India (2020). Available: https://edition.cnn.com/2020/11/11/tech/robots-india-covid-spc-intl/index.html

[2] NAVER LABS, M1, 3D/HD Mapping Robot (07.01.2019). Available: https://www.youtube.com/watch?v=BYWbf1iZ7co

[3] DEMİRCİ, B.: Robot Nasıl Yapılır – 2 – Robot Mekaniği (04.06.2020). Available: https://blog.tekyaz.com/robot-2-robot-mekanigi/

[4] IBM CLoud Learner Hub; What is Machnie Learning; IBM; (15 July 2020) Available: https://www.ibm.com/cloud/learn/machine-learning

[5] WEISSTEIN, Eric W. "Euclid's Theorems." From MathWorld--A Wolfram Web Resource. Available: https://mathworld.wolfram.com/EuclidsTheorems.html

[6] WEISSTEIN, Eric W. "Pythagorean Theorem." From MathWorld--A Wolfram Web Resource. Available: https://mathworld.wolfram.com/PythagoreanTheorem.html

[7] ŞEKER, A Yıldız Arama Algoritması (A Star Search Algorithm, A* (March 2, 2009). Available: http://bilgisayarkavramlari.com/2009/03/02/a-yildiz-arama-algoritmasi-a-star-search-algorithm-a/

[8] DELİBAŞOĞLU, HSV renk uzayı, Renk filtreleme (November, 6 2016) Available: http://ibrahimdelibasoglu.blogspot.com/2016/11/hsv-renk-uzay.html

[9] PANDIT, Abhiemanyu; Serial Communication Protocols (April 29, 2019) Available: https://circuitdigest.com/tutorial/serial-communication-protocols

[10] SEMİZ, Tayfun Yağız, Arduino Uno Nedir? (2018). Available: https://maker.robotistan.com/arduino-uno/

[11] Samm Market, Orijinal Raspberry Pi Güç Adaptörü Özellikleri (2021). Available: https://market.samm.com/raspberry-pi-guc-adaptoru-lisansli-standart

[12] Camera Module V2 Specifications; Raspberry Organization (April 2016); Available: https://www.raspberrypi.org/products/camera-module-v2/

[13] İZGÖL, Kerem, DC Motor Çeşitleri Nelerdir? (2015). Available: https://maker.robotistan.com/dc-motor-cesitleri-nelerdir/

[14] L 293D Motor Driver IC ; Components 101; (8 Ctober 2017). Available: https://components101.com/ics/l293d-pinout-features-datasheet

[15] I2C Temassız IR Sıcaklık Sensörü - MLX90614-BCC - Gravity; Direnç.net. Available: https://www.direnc.net/i2c-temassiz-ir-sicaklik-sensoru-mlx90614-bcc-gravity

[16] ÇİFTÇİ, A; Arduino ile I2C Oled Ekran Kullanımı; RoboLink Teknoloji; Available: https://akademi.robolinkmarket.com/arduino-ile-i2c-oled-ekran-kullanimi/

[17] RGB Led Nedir?; Hayal Et Ve Yap; (29 May 2017). Available: https://hayaletveyap.com/rgb-led-nedir/

[18] DOĞAN, Göksel; LiDAR nedir? LiDAR Sistemleri Nasıl Çalışır? (December 25, 2020) Available: https://teknoloji.org/lidar-nedir-lidar-sistemleri-nasil-calisir/

[19] AKYOL S., UÇAR A., Rp-LIDAR ve Mobil Robot Kullanılarak Eş Zamanlı Konum Belirleme ve Haritalama (2019). Available: https://dergipark.org.tr/en/download/article-file/662030

[20] Adobe Creative Cloud (2020). Available: https://helpx.adobe.com/tr/creative-cloud/faq/mixamo-faq.html

[21] Teachable Machine; Available: https://teachablemachine.withgoogle.com/faq

[22] PARWIZ, OpenCV Python Color Detection Example (2018). Available: https://codeloop.org/opencv-python-color-detection-example/

[23] ANONYMOUS, Python OpenCV: Converting an image to HSV (2019). Available: https://techtutorialsx.com/2019/11/08/python-opencv-converting-image-to-hsv/

[24] GRANOVSKY, Python NumPy'de boyut ve eksen nedir? (2018) . Available: https://qastack.info.tr/programming/19389910/in-python-numpy-what-is-a-dimension-and-axis

[25] EARL, Bill; BARELA, Anne; Arduino Libraries (February 16, 2013) Available: https://learn.adafruit.com/adafruit-all-about-arduino-libraries-install-use

[26] FINCHER, Jon; PyGame: A Primer on Game Programming in Python (2016). Available: https://realpython.com/pygame-a-primer/

[27] PONNUSAMY, A., CNN Based Face Detector From Dlib (17.04.2018). Available:https://towardsdatascience.com/cnn-based-face-detector-from-dlib-c3696195e01c

[28] ROSEBROCK, ADRIAN, COVID-19: Face Mask Detector with OpenCV, Keras/TensorFlow, and Deep Learning (May 4 2020). Available: https://www.pyimagesearch.com/2020/05/04/covid-19-face-mask-detector-withopencv-keras- tensorflow-and-deep-learning/

[29] ROUSE, Margaret , Face Detection (2020). Available: https://searchenterpriseai.techtarget.com/definition/facedetection?_ga=2.267397260.362427 755.1607879949-563445389.1607879949

[30] YAO, Ye, Sites of Skin Temperature Measurement A Forehead B Cheek C chest D upper arm (2007). Available: https://www.researchgate.net/figure/Sites-of-skin-temperature-measurement-A-forehead-B-cheek-C-chest-D-upper-arm-E_fig1_247731876

[31] ROSEBROCK, Adrian, OpenCV Social Distance Detection (June 1, 2020). Available: https://www.pyimagesearch.com/2020/06/01/opencv-social-distancing-detector/

[32] Sequencer Overview, Unreal Engine , Epic Games. Available: https://docs.unrealengine.com/en-US/AnimatingObjects/Sequencer/Overview/index.html

[33] Raspberry PI Industry Standards; Raspberry Org; (April 2016) Available: https://www.raspberrypi.org/products/compute-module-4/?variant=raspberry-pi-cm4001

000

[34] Arduino Specifications ; Arduino Store; Available: https://store.arduino.cc/usa/arduino-uno-rev3

[35] Unreal Engine Coding Standards; Epic Games; Available: https://docs.unrealengine.com/4.26/en-US/ProductionPipelines/DevelopmentSetup/CodingStandard/

[36] IFR Standardization; International Standards on Robotics. Available: https://ifr.org/standardisation

[37] W. Allan; Ethical Standards in Robotics and AI, Nature Electronics, (February 2019) Available: https://www.researchgate.net/publication/331138667_Ethical_standards_in_robotics_and_AI