Palestine Polytechnic University

Collage of Information Technology and Computer Engineering

Sanitizer Spider Robot

Submitted By:

Bayan Bassam Karajat

Rana Mahmoud Awlad Mohammad

Supervisor:

Eng. Wael Takrouri.

Submitted to the Collage of Information Technology and Computer Engineering in partial fulfillment of the requirements for the degree of Bachelor degree in Computer Systems Engineering.

Jun, 2021

جامعة بوليتكنك فلسطين

الخليل – فلسطين

كلية تكنولوجيا المعلومات و هندسة الحاسوب

دائرة هندسة الحاسوب

لسم المشروع:

Sanitizer spider robot

أسماء الطلبة:

بيان بسام كرجات　　　　　　　　رنا محمود أولاد محمد

بناء على نظام كلية تكونولوجيا المعلومات و هندسة الحاسوب و إشراف و متابعة المشرف المبلشرة على المشروع و موافقة أعضاء اللجنة الممتحنة تم تقديم هذا المشروع إلى دائرة هندسة الحاسوب و ذلك للوفاء بمتطلبات درجة البكالوريوس في تخصص هندسة أنظمة الحالسوب.

توقيع المشرف

....

توقيع اللجنة الممتحنة

....　　　　..................................　　　　............................

توقيع رئيس الدائرة

....

Contents

Acknowledgement

In the name of “Allah “the most beneficent and merciful, who gave us strength and helpful us to get through this project.

We sincerely, thank our parents for their kind support to reach the end of point in our university life.

We would like to thank Dr. Hamzah Takrouri for his help to choose the YOLO algorithm and solving the problems we faced in implementation and thank Eng. Wael Takrouri for his support and assistance to complete this project.

Last but not least, we would like to thank our friends who went together step by step and hand by hand during the college years, and thanks to all teachers for giving us enough information, we are also thankful to everyone who supported us through many helpful comments.

Abstract

With the emergence of the Coronavirus that spread widely around the world. The importance of sterilization and virus-free surfaces have been one of the crucial subjects in this regard and due to the difficulty of performing periodic disinfection of surfaces by humans especially in public places. The need of an automated system has emerged to periodically perform an automated sterilization of the surfaces. Our project seeks to provide automatic sterilization of surface sterilization and guarantee periodic sterilization, by identifying possible infected objects such as tables, chairs or even doors in order to spray them with sterile material.

List of tables

List of figures

Chapter 1

Introduction

The following subsections have an overview, motivation, problem statement, system objectives and short description of the system.

1.1 Overview of the project

Sanitizer spider robot is an automated system designed to perform a periodically automated sterilization for the surface. Which is designed specially to function in public institutions. This robot can move easily with the help of its four legs. It can also detect the infected surfaces by its sensors and cameras that are fixed in the top of the robot. After detecting the surfaces for the sterilization process, the robot sprays them with sterilization material.

1.2 Motivation

The need for this robot was after the spread of the coronavirus and our need to sterilize the surfaces continuously and periodically to reduce the incidence of the virus transmitted to humans by contact with contaminated surfaces.

1.3 Problem statement

Surfaces are sterilized manually in institutions and public places. This would be loose; people might forget to sterilize all surfaces continuously and periodically during the day. This will lead to the accumulation of germs and viruses on the surfaces for a long time. Then it's transmitted to humans by contacting these contaminated surfaces. But after the emergence of the Coronavirus, the need to keep the surfaces sterile and free of viruses has become very important.

1.4 System objectives

The main objectives of this system are:

- The ability to detect indoor suspected infected objects such as tables, chairs in order to be sterilized.
- Guarantee periodic sterilization of surfaces.

1.5 Short description of the system

To ensure surface sterilization especially in public places, an automated system has been designed to perform a periodically automated sterilization of the surface. The system is a spider robot, which contains these components: four legs for movement and each leg has two servo motors to control the direction of movement and speed. The top of the robot contains sensors, a camera, Valve and pump. Raspberry-Pi is the controller of the system and battery to power it.

The robot moves in the place periodically and during movement detects possible infected objects. If these objects are considered infected, the robot will spray it with sterile material which is placed in a sterile container.

Figure 1.1 presents an overview of the system component and the relation between the robot and the external environment.

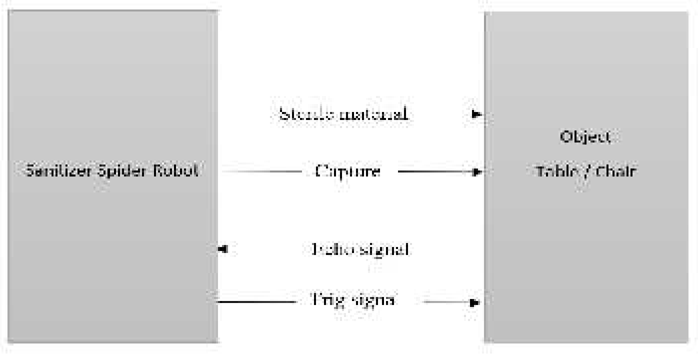

Figure 1.1: Context Diagram

1.6 Overview of the rest of the report sections

The major topics covered in this report are robotics and Machine learning which will be explained in chapter two. Chapter three will include the block diagram and design for the robot. Chapter four contains an introduction to the software and the platform used for the robots' programming. Chapter five will talk about the expected results and validation for the robot. Finally, the last chapter will provide the project's conclusions.

Chapter 2

Background

2.1 Overview

This chapter introduces the literature review of the project, a description of technologies used in the project and some description of the hardware and software components that are used in the system. At the end, some description of design specification and constraints.

2.2 Theoretical background

Robots are emerging in many areas. The researchers have almost daily read about them in the newspapers; the use of drones strikes in war, robots in nursing homes keeping the elderly company and about cars with the ability to park or even drive without human supervision. It seems like the world is becoming more and more automated along with the robot’s evolution. The research is based upon producing more sociable robots. Which can read facial expressions and hold conversations.[1]

The project's main idea is to design and implement a sanitizer spider robot using Raspberry Pi and based on a python environment. A special sensor is used to sense objects and send a signal to Raspberry Pi. Consequently, the camera captures an image for the suspected objects. If the object is one of the desired objects. Then the robot starts spraying it with sterile material.

2.3 Literature review

The researcher’s project is pretty much similar to the following projects:

1. Adaptive Intelligent Spider Robot project

Rahim Nakhli Mahal, Ng Keng Shen, and Kok Wai Chan researchers in University Kuala Lumpur, Malaysia in 2013 decided to design an Adaptive Intelligent Spider Robot which is able to monitor the environment wirelessly.

The main similarity between our project and the “Adaptive Intelligent Spider Robot” is its shape. Both of them contain four legs and each leg has two motors to move.

2. Spider Robot project

Mazen Alktebi a researcher at Evansville University, Indiana in 2019 decided to design a spider robot. Which able to do the following:

- Exploring dangerous areas instead of humans.
- Exploring war zones.
- Inspecting unstable buildings after natural disasters such as an earthquake.
- Defusing bombs such as land mines.

The main similarity between the researchers’ project and the “Spider Robot project” is its shape. But The researchers' project will insert eight servos for four legs instead of eighteen servos; the researchers preferred to use a smaller number of legs for the robot to to easily control its movement with less complexity. In addition, the researchers' project will be specialized in indoor places. So, there is no need to have a six-legged robot.

2.4 Technologies to be used in the project

This subsection illustrates the main technologies to be used in this project and what tasks they function.

1. **Python Programming Language**

Python is an open-source computer programming language that lets you work more quickly than other programming languages. Experienced programmers in any other language can pick up Python very quickly. Also, beginners find the clean syntax and indentation structure easy to learn. Furthermore, it was supported by the OpenCV library that provides object recognition features in the researchers' project.[2]

2. **OpenCV Library**

OpenCV is an open-source C++ library for image processing and computer vision. Originally developed by Intel and now supported by Willow Garage.

It is free for both commercial and non-commercial use. It is a library of many inbuilt functions mainly aimed at real-time image processing. Now it has several hundreds of image processing and computer vision algorithms that make developing advanced computer vision applications easy and efficient. [3]

2.5 Hardware components of the system

This part describes the components and devices used in the system.

The hardware of the spider robot consists of two major parts: the electrical parts and the Spider body panels. Below is a list of all electrical parts that have been used.

1. **Raspberry pi 3**

Raspberry Pi is a single-board computer. It is a credit-card sized computer with low cost. Which plugs into a computer monitor or TV. In order to operate it, a user have to use a standard keyboard and mouse.[4]

The single-board consists of a fully functional computer with its dedicated memory, processor and it runs an operating system. Figure 2.1 shows Raspberry Pi 3 mode B.

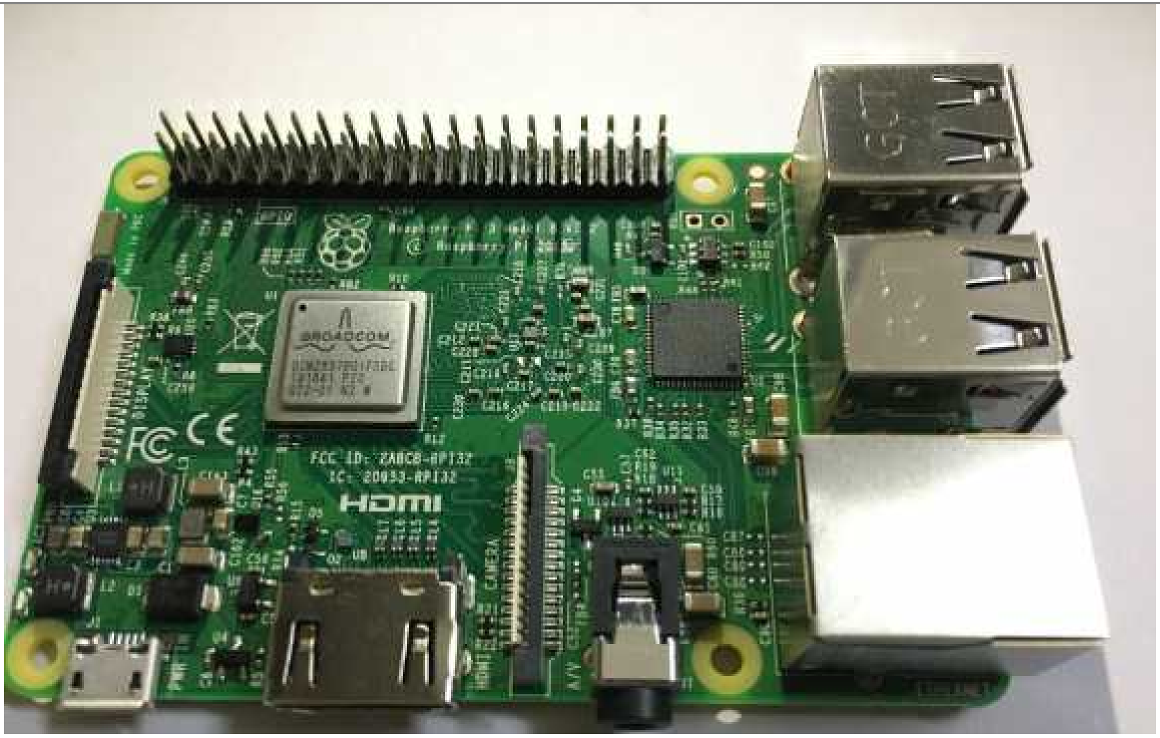

Figure 2.1: Raspberry Pi 3

2. Pi Camera

The Raspberry Pi Camera Modules are official products from the Raspberry Pi Foundation. It is easy to use for beginners but has plenty to offer advanced users if we're looking to expand our knowledge. There are lots of examples online of people using it for time-lapse, slow-motion and other video cleverness. We can also use the libraries we bundle with the camera to create effects. It attaches via a 15cm ribbon cable to the CSI port on the Raspberry Pi. There are numerous third-party libraries built for it, including the Pi Camera Python library. The camera module is very popular in a lot of applications and we can use it to take snapshots.[5]

The main function of our project is to detect the objects to be sterilized and this will never be achieved without taking an image of the objects. In Figure 2.2 appears Raspberry Pi Camera with ribbon cable.

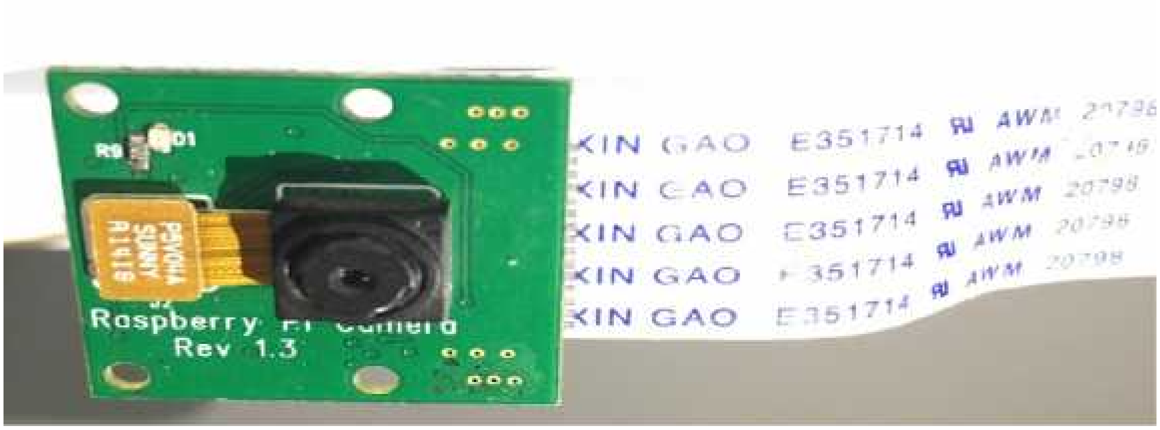

Figure 2.2: Pi-Camera

3. Ultrasonic Sensor

Ultrasonic sensor measures distance by using ultrasonic waves. The sensor head emits an ultrasonic wave and receives the wave reflected back from the target. Ultrasonic sensors measure the distance to the target by measuring the time between the emission and reception[6]. HC-SR04 Ultrasonic sensor illustrated in figure 2.3.

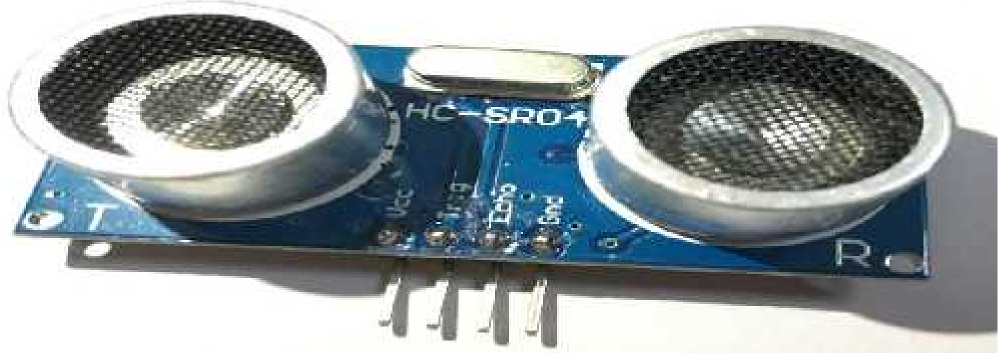

Figure 2.3: Ultrasonic Sensor.

4. Servo motors and motor driver

The servo motor is most commonly used for high technology devices in industrial applications like automation technology. It is a self-contained electrical device that rotates parts of a machine with high efficiency and great precision. The output shaft of this motor can be moved to a particular angle.[7]

In our project we will use servo motor mg 90S illustrated in figure 2.4; it has metal gears which means when the motor is jammed. The gears aren’t likely to strip and it will be more rigid to wear out and it uses a high-speed pulse train. The researchers’ robot is four-legged, each leg needs two servo motors to move. So, the researchers inserted eight servo motors.

To control the position of servo motors and drive smoothly. The robot needs a servo motor driver, which is used to control a single servo motor or even a group of servo motors. It is the mainstay of the task to be accomplished, the controller must drive more than one servo.

Figure 2.4: Servo motor MG90s.

5. **Sterile container**

   A sterile container is a container used to contain the sterile material. Is a plastic bottle connected to a DC motor in order to push on the top of the bottle to spray the sterile.

6. **Lithium-Ion Battery**

   This battery used to supply the Raspberry Pi and motor driver. [8]

7. **Dc motor and DC motor driver**

   The DC motor used to connect to the sterile container to spray the sterile and to control DC motor driving we need (L293D) IC.

8. **Spider robot 3D printed parts**

   The robot design were selected from Thingiverse website [9] and printed using a 3D printer. The resulted parts are shown in figure 2.5.

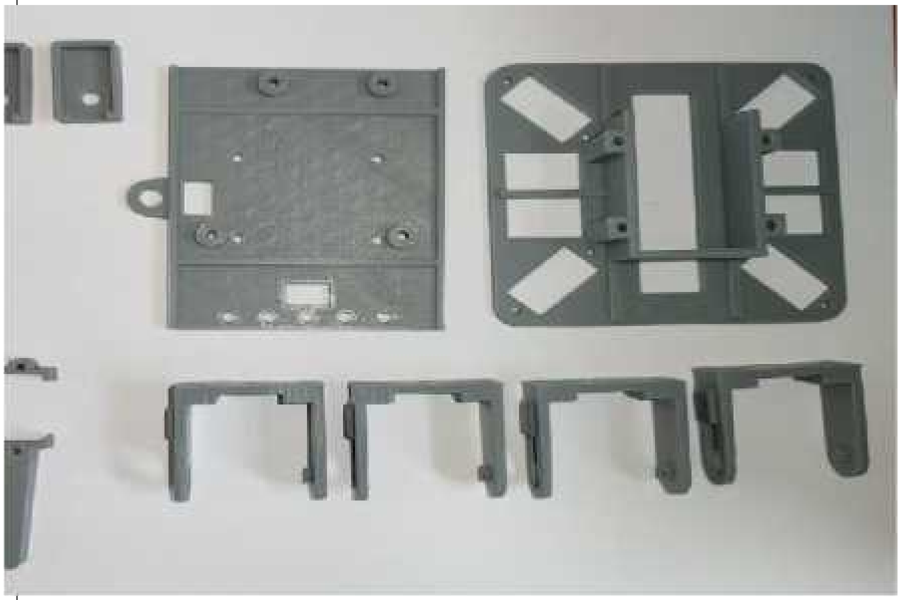

Figure 2.5: Spider robot 3D printed parts

2.6 Design alternatives

This subsection illustrates the alternative for the hardware components of the system and explains why we don't use it.

1- Microcontroller of the system:

ASUS Tinker Board

Tinker Board is a Single Board Computer (SBC) in an ultra-small form factor that offers class-leading performance while leveraging outstanding mechanical compatibility. The Tinker Board offers makers, IoT enthusiasts, hobbyists, PC DIY enthusiasts, and others a reliable and extremely capable platform for building and tinkering their ideas into reality.[9]

Table 2.1 presents the main difference in specifications for Raspberry Pi 3 and ASUS Tinker Board.[10]

	Raspberry Pi 3	**ASUS Tinker Board**
CPU	Cortex-A53 Quad Core @ 1.4GHz	Cortex A17 Quad Core @ 1.8GHz
RAM	1 GB LPDDR2 (900 MHz)	2 GB LPDDR3 (dual channel)
GPU	Broadcom VideoCore IV @ 400 MHz	Mali-T764 @ 600MHz
Network (LAN)	10/100/1000 Mbps (~230 Mbps real world)	10/100/1000 Mbps
Network (Wi-Fi)	2.4GHz or 5GHz 802.11n/ac	2.4GHz 802.11n
Audio interface	48kHz / 16 bit	192kHz / 16 bit
GPIO	40 pin header (not color coded)	40 pin header (color coded)
Price	35$	49.99$

Table 2.1: Comparator between Raspberry Pi 3 and ASUS Tinker Board specifications

ASUS Tinker Board is more powerful but it costs more and this board not available in the local market. And the Raspberry Pi will fulfill the requirement to be done. So, we prefer to use Raspberry Pi 3 because it is cheaper and available in markets. Also, Raspberry Pi specifications are enough to perform the main functions in our project.

2- Distance measurement sensor

IR sensor:

IR sensor is an electronic device that uses an infrared transmitter and receiver to emit and detect objects.

The biggest difference between IR sensor and Ultrasonic sensor is the way in which the sensor works. Ultrasonic sensor use sound waves to measure how far away you are from an object. On the other hand, IR sensor use infrared light to determine whether or not an object is present. To measure the distance with greater accuracy and reliability, the researchers decided to use the Ultrasonic sensor.

3- Servo motor driver:

8 channel servo controller (SC08):

8 Channel Servo Controller SC08 offers reliable yet user-friendly RC Servo motor controllers to hobbyists and students. This SC08A can control 8 channels of servo simultaneously. The advantages of this SC08A over SC16A are the user is able to deactivate any servo channel at any time and the user can even set the initial position of any channel at the next startup. Each servo signal pin is able to generate servo pulses from 0.5 ms to 2.5 ms, which is greater than the range of most servos, further allowing for servos to operate 180 degrees. SC08A comes with a small size and different protocol compared to SC16A. Through serial communication, SC08A can be daisy chain in 2 boards to offer independent control over 16 RC servo motors simultaneously. The host of SC08A can either be a PC desktop/Laptop with USB to serial (UART) converter, or microcontroller with UART interface. UART interface presents a flexible, fast and easy to use a feature.[11]

But this type of servo driver is not available in the local markets. Because of that, the researchers prefer to use PCA9685 rather than SC08A.

4- Servo motor:

Servo motor (SG90):

Micro servo is tiny and lightweight with high output power. Servo with three horns. It can be converted into continuous rotation servo and can rotate approximately 180.

The main difference between the MG90S and SG90 is the material used in the manufacture of the motor. SG90 has a plastic gears servo motor. On other hand, MG90S has a metal gears servo motor which guarantees that the gears aren’t damaged during rotation. So the researchers prefer to use the MG90S.

Chapter 3

System Design

3.1 Overview

The following section has a description of the system, detailed design, and necessary information about the design.

3.2 Brief description of the system

The system of the project will be designed to track a specific path in the room and detect an infected object in order to be satirized. When the system is activated the Raspberry-Pi sends a signal to the servo motor driver, which controls the position of the servo motor and drives it.

After the robot starts to track the path, it will view all objects around the path by using the camera which detects the objects in the area. Then if the detected object is a chair or table, the robot calculates the distance between him and the first object in front of by using ultrasonic. When the distance equals five centimeters, the robot will rotate left or right or move straight depending on the position of the object. and it sends a signal to the raspberry pi board. Then the raspberry pi sends a signal to the pi camera to capture an image. After that, it compares the image with known objects to be satirized. If the object is identified as an infected surface the raspberry pi sends a signal to the sterile container to spray the object with the sterile material. Otherwise, the system will keep moving on its path. Figure 3.1 illustrates how the robot moves in a room.

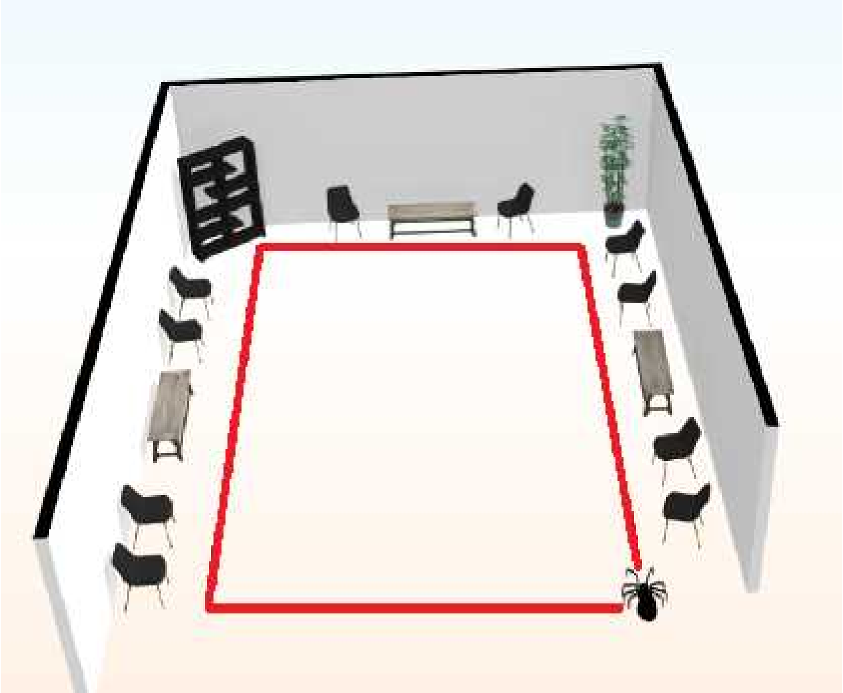

Figure 3.1: System demo

3.3 System diagrams

In this subsection, three diagrams are represented for the understanding of the project concepts and design.

3.3.1 System block diagram

Figure 3.1 is the general block diagram of the project. As illustrated below, the raspberry pi if the system activated its control on the servo motor driver which controls the position of servo motors and runs it. The ultrasonic sensor detects the object and sends a signal to the raspberry pi. Consequently, Raspberry pi sends a signal to the camera to capture an image and save it in raspberry pi. Raspberry pi compares it with known objects to satirize if the object is identified as a surface to satirize the raspberry pi sends a signal to the sterile container to spray the object with the sterile material, Otherwise the system will return to move in the room.

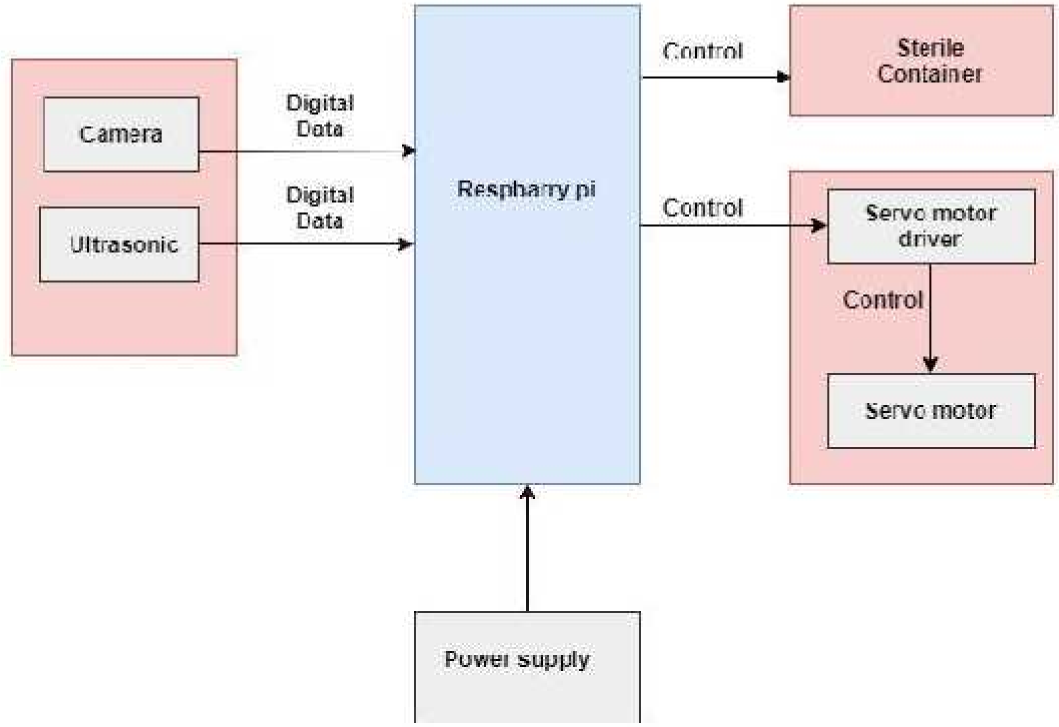

Figure 3.2: Block diagram

3.3.3 Schematic diagram

Figure 3.4 shows how hardware components connect with each other's:

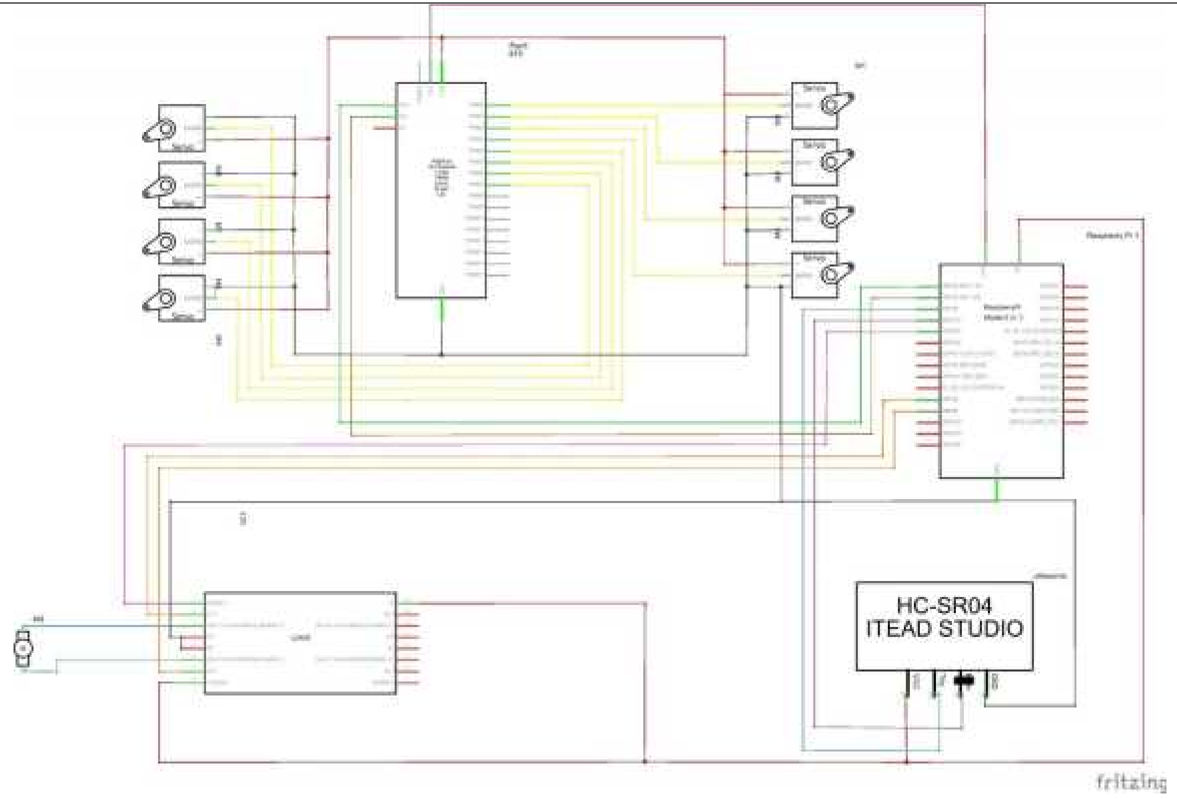

Figure 3.4: schematic diagram

1- **Raspberry Pi:** The main component in the system, which will be linked to the other components. Represent the inlet and outlet of the signals and information.
2- **Ultrasonic sensor:** Sensing the there is any object in front of the system in the range of about 5 centimeters, then send a signal to raspberry pi to give an order to the camera to capture an image.
3- **Camera:** Used for image input that captures the object after taking an order from Raspberry Pi.

4- **Servo motors:** Used to control the movement of the system. In our system, we have four legs and each leg requires two servo motors to move.
5- **Servo motor driver:** Used to control the position of servo motors and drive it.
6- **DC motor:** Used for push on the top of the bottle to spray sterile.
7- **DC motor driver (L293D IC):** Used for control on dc motor.

First, the eight servo motors connect to channels pin from zero to seven in the servo motor driver (PCA9685). Then connect the I2C data pin (SDA) and I2C clock pin (SCL) in the driver to GPIO 2 and GPIO 3 pins in Raspberry Pi.

Ultrasonic pins connect to Raspberry Pi pins as following: Vcc pin to 5V pin, GND to GND pin, Trig pin to GPIO 4 and Echo pin to GPIO 5. Finally, the enable pin in L293D connects to GPIO, input1 and input2 connect to GPIO 6 and GPIO 7 then the output1 and output2 from L293D connect to DC motor.

3.4 System flowchart

Figure 3.5 shows the flow chart of the system:

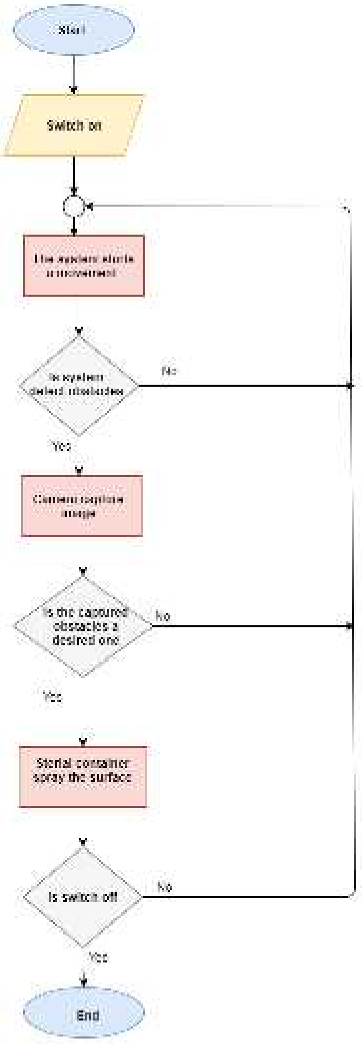
Start
Switch on
The system starts a movement
Is system detect obstacles
No
Yes
Camera capture image
Is the captured obstacles a desired one
No
Yes
Sterial container spray the surface
Is switch off
No
Yes
End

Figure 3.5: flowchart

Chapter 4:

Software and hardware implementation

4.1 Overview

This chapter covers the software and hardware implementation of the project, as well as the various components and tools needed to construct the robot.

4.2 Raspberry pi microcomputer

To function, the Raspberry Pi requires an operating system. The researchers used Raspberry Pi imager v1.6 to install Raspberry Pi OS (Raspbian). Installing the Raspberry Pi OS and other operating systems to a microSD card is simple and quick. As a result, the Raspberry Pi OS is stored on a 16 GB class10 SD card. After selecting an operating system, write and verify the SD card. After you've inserted the SD card and connected it to the power source, the Raspbian operating system will be ready to use in our system.

4.3 Installing needed packages

Raspbian comes with many useful pre-installed packages such as JDK, python, and others, but we need to install some needed packages for our project. The most important one is the OpenCV library that will be used to provide object detection features. Many essential pre-installed packages, such as JDK, Python, and others, come with Raspbian, but we'll need to install a few more for our project. The OpenCV library, which will be utilized to offer object detection characteristics, is the most significant.

4.3.1 OpenCV implementation

OpenCV is a Python binding library for solving computer vision problems. The researchers follow the directions in [1] to install this library.

4.3.2 PCA9685 and ServoKit libraries

With the PCA9685 module, it's simple to develop Python code that controls servos and PWM. As a result, the researchers will need to install these libraries in order to control the robot's eight servo motors. The researchers follow the instructions in [12] to install the servoKit library.

4.4 Object detection implementation

Object detection is one of the most important capabilities of most computers and robot vision systems. We need an algorithm to implement object detection in our robot so that it can detect the tables and chairs that need to be disinfected.

Many object identification methods exist in computer science, including Regions based Convolutional Neural Networks (R-CNN), Fast R-CNN, Faster R-CNN, and the You Only Look Once (YOLO) algorithm. As for this research paper, the YOLO algorithm was chosen for this project.

YOLO is a distinct kind of object detection algorithm than region-based algorithms. The bounding boxes and class probabilities for these boxes are predicted by a single neural network in YOLO. YOLO algorithm is important because of these reasons:

- Speed: it can forecast objects in real time, it can boost detection speed.
- High precision: it can deliver precise results with minimum background noise.
- Learning capabilities: it can learn object representations and use them to detect items.
- It is an open source neural network framework that uses the YOLO algorithm to implement object detection; it's a pre-trained classifier. [13]

4.5 Hardware implementation

This subsection provides some information about the hardware implementations done through our project.

In short, the connection of the electronic components was as the following: Each leg in a quadruped robot contains two servo motors. The first motor represents the arm and the second motor represents the foot.

To build the robot we calibrate the motors at 90. Then we put each motor in its position. Then we connect it to the PSA9685 and write the Python script to control its rotation. The script runs the motors as the following:

1- footA rotates up and armA rotates forward then footA rotates down.

2- footC rotates up and armC rotates backward then footC rotates down.

3- footB rotates up and armB rotates backward then footB rotates down.

4- footD rotates up and armD rotates forward then footD rotates down.

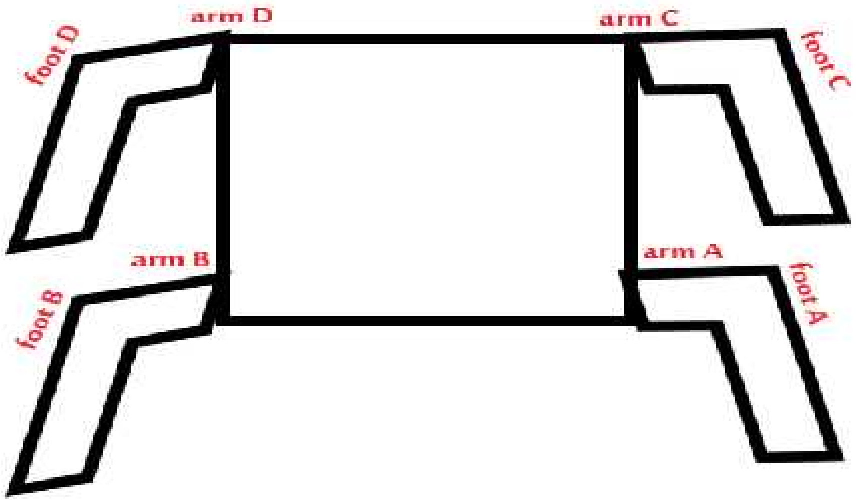

Figure 4.1: Arms and foots labeling

After motor connection and the robot movement were done. We connect the camera, ultrasonic, and L293D to the Raspberry Pi. Then write a Python script to run each of them.

4.5.1 Electronic units implementation

4.5.1.1 Power configuration

There are eight MG90S servo motors, each of which requires 5V and 175 mA to move [14]. In addition, the ultrasonic sensor HC-SR04 requires 5V and a 15-mA operating current [15]. The Pi camera requires 5V and 225mA of operational current [16]. Also, the Raspberry Pi contains a cooling fan that runs on 5V and 200 mA.

The robot will dissipate roughly 1800 mA based on the mentioned components. So we'll use 2X900 mAH Lithium-ion batteries, which will discharge in about an hour.

4.5.1.2 Raspberry pi camera configuration

The camera was needed to capture the objects in the room. The raspberry pi camera is connected to the camera module port of the Raspberry Pi.

4.5.1.3 PCA9685 module configuration:

To connect the PCA9685 module to Raspberry Pi we enable I2C, and then connect the pins as the following:

- Pi 3V3 to module VCC.
- Pi GND to module GND.
- Pi SCL to module SCL.
- Pi SDA to module SDA.

To run the PCA9685 module we need an external 5V power supply for servos.

4.5.1.4 Ultrasonic sensor configuration

The ultrasonic sensor was required to calculate the distance between the robot and the targeted object in order to determine whether or not it was within the sanitizing range.

To connect the Ultrasonic sensor to Raspberry Pi we put it on breadboard, and then connect the pins as the following:

- VCC to pi 5V pin.
- GND to pi GND pin.
- TRIG to GPIO18 pin.
- ECHO to GPIO24 pin.

4.5.1.5 DC Motor and Motor driver configuration

To spray the sanitizer into the sterile container, the DC motor was required. We used a motor driver IC to power this motor (L293d).

L293d is an IC used to control driving one DC motor or two motors. Because we used it to control on one dc motor, we connect it to the Raspberry Pi as the following:

1- Enable 1 pin in the IC connected to GPIO 4 pin in raspberry pi.
2- Input 1 and input 2 pins in the IC connected to GPIO 5 and GPIO 6 in raspberry pi.
3- GND pins in IC connected to ground pin in raspberry.
4- Vs and Vss pins in IC connected to 5 volt pin.
5- Output 1 and output 2 in IC connected to ends of the motor.

4.5.1.6 3D-printed holder

Both the sanitizing spray and the motor are held in place by a 3D printed holder portion. When the motor rotates, it pulls the spray head down, allowing the sanitizer to reach the targeted object.

Figure 4.5 shows the 3D- printed holder.

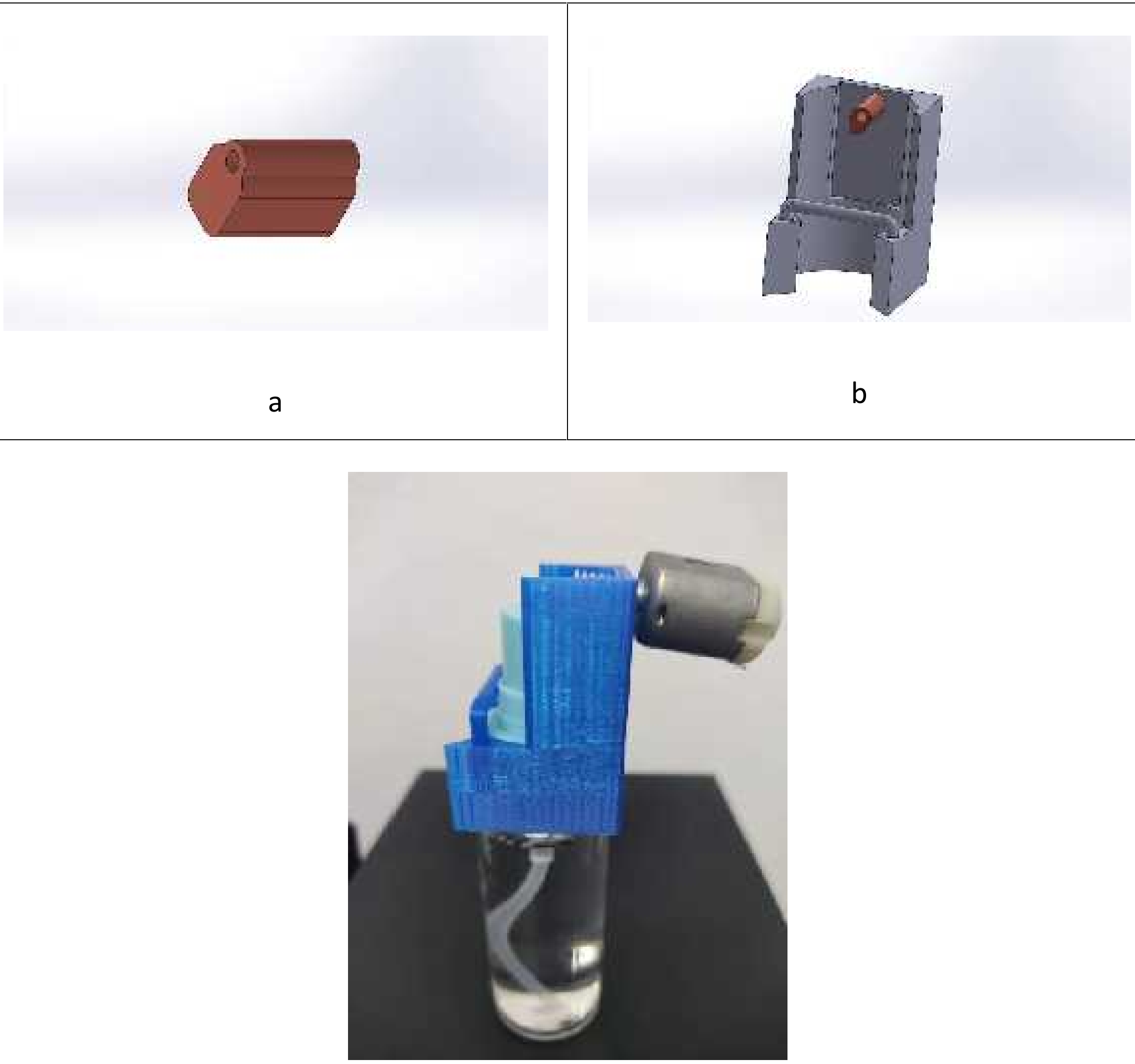

Figure 4.2: 3D-printed holder

Chapter 5

Validation and testing

5.1 Overview

This chapter explains the project component testing methodology and displays the project system implementation outcomes.

5.2 Motors testing

In this project, the researchers used two different types of motors, each with its own set of tests to see if they perform properly. As a result, the next subsection explains how the researchers tested the motors.

5.2.1 Servo motors testing

Each servo motor is powered individually from the Raspberry Pi, which then executes the servo motor control python software. After ensuring that all motors are spinning properly and that the servo motor driver can operate several motors at the same time. The eight servo motors are connected to the PCA9685 module. After that, import the necessary Python libraries, such as ServoKit and time. Run the python script to control many servo motors.

5.2.2 DC motor testing

The dc motor was tested by connecting it to the power supply. Then the L293D IC was connected to a Raspberry Pi and DC motor. And run the python script that was written for drive DC motor.

5.3 Camera testing and connection

The Pi camera was attached to the camera module port on the Raspberry Pi using a camera module ribbon cable that fit the Pi camera module port to see if it was working properly. The camera was then enabled in the terminal window via Raspberry Pi configuration. The (raspistill -o Desktop/image.jpg) command was written to take the image and save it to the Desktop in a jpg format.

5.4 Ultrasonic testing and connection

The ultrasonic was attached to the Raspberry Pi as explained in the last chapter to see if it was working properly. The distance between the ultrasonic and the item, assuming one exists, was then measured using a Python script.

5.5 OpenCV testing

To test if the OpenCV library was installed correctly. We tried the below commands in the python terminal.

```
$python3

>>> import cv2

>>> cv2.__version___

‘4.1.1’
```

5.6 YOLO algorithm testing

YOLO algorithm that will be used to provide object detection. We downloaded a Github project of the YOLO algorithm. It contains files such as yolov3.cfg, yolov3.weights ,and coco.names. Then we connect the pi camera to the Raspberry Pi and run a python script to detect objects that exist in front of the camera. Then detected objects were bound by a box and above each box there is a label which is the name of detected object figure 5.1 shows the output of YOLO algorithm.

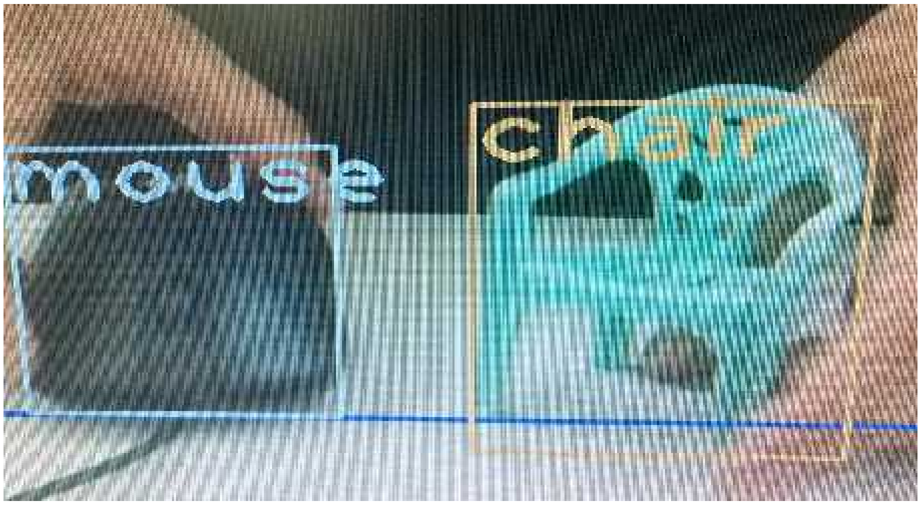

Figure5.1: YOLO test result

5.7 System testing

After connecting each component separately to the Raspberry Pi and testing it. We connect all parts of the robot and run the Python code.

5.8 System validation

The system was able to move forward in the room, and it can detect if there object and check if the object is a chair or table then if the object is in spray range the robot drives the DC motor to push on the sterile spray.

Chapter 6

Conclusion and future work

6.1 Overview

This chapter shows the project conclusion, the main achieved objectives, and will propose future works that can be developed on the system.

6.2 Conclusion

In conclusion, the two main objectives of this project were achieved. The robot can move in the infected room and it can measure the distance from the objects then it sprays sterilization material on the object chair or table. This was achieved by using the appropriate hardware components, as well as installing the required packages. The project's hardware and software have been successfully integrated and work to meet the requirements.

6.3 Future work

Some future works are suggested and recommended to improve the project:

1- Our robot doesn't know the start and end point in the room nor its current location, the system can be improved by adding localization and mapping to the robot.

2- Improve computational time by training the system other than using the pre-trained model.

3- Adding additional objects to be sterilized.

Summary

In this report, the researchers explain the idea of the project including the preparation process and the tools used to design the robot. In Chapter one the researchers introduced a brief introduction about the project, which included the following topics: overview, motivation, problem statement, system objectives, and a short description of the system. In chapter two, the project talked about the background of the project; the researchers introduced the Literature Review of the project, a description of technologies to be used in the project and some description of hardware and software components used in the system. In chapter three, a brief description of the design specification and constraints, contained the design of the project. The researcher's description of the system and exposed the system design in detail. In chapter four, a brief description of the software and hardware implementation. In chapter five, the researchers explain the test process of the project components. Finally, the last chapter contains the project conclusion and proposes future works that can be developed on the system.

Appendixes

Appendix A

Sanitizer spider robot Python script:

```
import cv2
import numpy as np
import RPi.GPIO as GPIO
from adafruit_servokit import ServoKit
from time import sleep

TRIG = 21 # Trigger
ECHO = 20 # Echo
GPIO.setmode(GPIO.BCM) # GPIO Numbering
GPIO.setwarnings(False)       # Stop warning msgs
GPIO.setup(TRIG,GPIO.OUT)  # Trigger pin as output
GPIO.setup(ECHO,GPIO.IN)    # Echo pin as input
GPIO.output(TRIG,False)
GPIO.setup(Motor1A,GPIO.OUT) # Spary motor pin1
GPIO.setup(Motor1B,GPIO.OUT) # Spary motor pin2
GPIO.setup(Motor1E,GPIO.OUT)
kit = ServoKit(channels = 16)# Servo kit 16 channels
Fup = 140     # Foot up angle
Fdw = 100     # Foot down angle
Abw = 120     # Arm backword angle
Afw = 160     # Arm forword angle
spd = 0.01    # speed delay


def initailize_yolo():

"""
Initailze_yolo loads/reads weight file and configration files.
Return a boolean value indicating that desired object exist or not
"""

   net = cv2.dnn.readNet("yolov3-tiny.weights", "yolov3.cfg")
   classes = []
   with open("coco.names", "r") as f:
      classes = [line.strip() for line in f.readlines()]
   layer_names = net.getLayerNames()
   output_layers = [layer_names[i[0] - 1] for i in
net.getUnconnectedOutLayers()]
   colors = np.random.uniform(0, 255, size=(len(classes), 3))


def object_detection():
"""
ObjectDetect function detect type of object read by camera.
Return a boolean value indicating that desired object exist or not
"""

   found = False
   # Loading video
   fourcc = cv2.VideoWriter_fourcc(*"XVID")
   out = cv2.VideoWriter("the_new_video_is.avi", fourcc , 25, (852, 480))

   # repalce the test.mp4 with an video of your own
   camera = cv2.VideoCapture(0)


   _,img = camera.read()
   height, width, channels = img.shape

   # Detecting objects
   blob = cv2.dnn.blobFromImage(img, 0.00392, (320, 320), (0, 0, 0), True,
crop=False)
   net.setInput(blob)
   outs = net.forward(output_layers)

   # Showing informations on the screen
   class_ids = []
   confidences = []
   boxes = []
   for out in outs:
      for detection in out:
         scores = detection[5:]
         class_id = np.argmax(scores)
         confidence = scores[class_id]
         if confidence > 0.5:
            # Object detected
            center_x = int(detection[0] * width)
            center_y = int(detection[1] * height)
            w = int(detection[2] * width)
            h = int(detection[3] * height)
            # Rectangle coordinates
            x = int(center_x - w / 2)
            y = int(center_y - h / 2)
            boxes.append([x, y, w, h])
            confidences.append(float(confidence))
            class_ids.append(class_id)

   indexes = cv2.dnn.NMSBoxes(boxes, confidences, 0.5, 0.4)

   font = cv2.FONT_HERSHEY_PLAIN
   for i in range(len(boxes)):
      if i in indexes:
         x, y, w, h = boxes[i]
         label = str(classes[class_ids[i]])
         if label == 'chair':
            found = True
         color = colors[i]
         cv2.rectangle(img, (x, y), (x + w, y + h), color, 2)
         cv2.putText(img, label, (x, y + 30), font, 3, color, 3)
      #cv2.imshow("Image", img)
      # key = cv2.waitKey(1)
      # if key == 27:
      # break
   camera.release()
   cv2.destroyAllWindows()


def read_distance():
"""
read_distance function dertermines distacne between robot and
obsticales distance value are in cm
"""
    time.sleep(0.2)
    GPIO.output(TRIG,True)
    time.sleep(0.00001)
    GPIO.output(TRIG,False)
    while GPIO.input(ECHO)==0:
        pulse_start=time.time()
    while GPIO.input(ECHO)==1:
        pulse_end=time.time()
    pulse_duration=pulse_end-pulse_start
    distance=pulse_duration*17150
    distance=round(distance,2)
    #print("distance:",distance,"cm")
    time.sleep(2)
    return distance

def spary_motor():
"""
spary_motor moves the DC motor of santizer spary
"""

    GPIO.output(Motor1A,GPIO.HIGH)
    GPIO.output(Motor1B,GPIO.LOW)
    GPIO.output(Motor1E,GPIO.HIGH)
    sleep(5)   # wait for 5 seconds
    GPIO.output(Motor1A,GPIO.LOW)
    GPIO.output(Motor1B,GPIO.LOW)
    GPIO.output(Motor1E,GPIO.HIGH)

def foot_up_A():
"""
Moves A foot upward
"""
    max = 80
    min = 40
    for i in range(min, max, +1):
        kit.servo[1].angle = i
        time.sleep(spd)

def foot_down_A():
"""
Moves A foot downward
"""
    max = 80
    min = 40
    for i in range(max, min, -1):
        kit.servo[1].angle = i
        time.sleep(spd)

def arm_forword_A():
"""
Moves A arm forward
"""
    min = 80
    max = 140
    for i in range(min, max, +1):
        kit.servo[0].angle = i
        time.sleep(spd)

def foot_up_B():
"""
Moves B foot upward
"""
    max = Fup + 5
    min = Fdw + 5
    for i in range(min, max, +1):
        kit.servo[3].angle = i
        time.sleep(spd)

def foot_down_B():
"""
Moves B foot downward
"""
    max = Fup + 5
    min = Fdw + 5
    for i in range(max, min, -1):
        kit.servo[3].angle = i
        time.sleep(spd)

def arm_backward_B():
"""
Moves B arm backward
"""
    min = 120
    max = 180
    for i in range(max , min , -1):
        kit.servo[2].angle = i
        time.sleep(spd)
def foot_up_C():
"""
Moves C foot upward
"""
    max = Fup + 5
    min = Fdw + 5
    for i in range(min, max, +1):
        kit.servo[7].angle = i
        time.sleep(spd)

def foot_down_C():
"""
Moves C foot downward
```

```
"""
    max = Fup + 5
    min = Fdw + 5
    for i in range(max , min , -1):
        kit.servo[7].angle = i
        time.sleep(spd)

def arm_backward_C():
"""
Moves C arm backward
"""
    min = Abw + 1
    max = Afw + 1
    for i in range(max , min , -1):
        kit.servo[6].angle = i
        time.sleep(spd)


def foot_up_D():
"""
Moves D foot upward
"""
    max = Fup + 5
    min = Fdw + 5
    for i in range(min , max , +1):
        kit.servo[5].angle = i
        time.sleep(spd)

def foot_down_D():
"""
Moves D foot downward
"""
    max = Fup + 5
    min = Fdw + 5
    for i in range(max, min, -1):
        kit.servo[5].angle = i
        time.sleep(spd)

def arm_forword_D():
"""
Moves D foot forward
"""
    min = 80
    max = 120
    for i in range(min , max , +1):
        kit.servo[4].angle = i
        time.sleep(spd)



def go_ahead():
"""
Go a head function will do the sequence to move forward
"""
    for i in range(140, 100, -1):
        kit.servo[0].angle = i
        kit.servo[4].angle = i
        y = 100 + 40 - i
        kit.servo[6].angle = y
        kit.servo[2].angle = y
        time.sleep(spd)

def walk():
    go_ahead()
    foot_up_A()
    arm_forword_A()
    foot_down_A()
    foot_up_C()
    arm_backward_C()
    foot_down_C()
    foot_up_B()
    arm_backward_B()
    foot_forkward_B()
    foot_up_D()
    arm_forword_D()
    foot_down_D()


def main():
"""
main is the main entry point to the code
"""
   initailize_yolo()
    while 1 :
      readDistance() # Read distnace from object
      if distance < 30:
        if object_detection() == True:
           spary_motor()
           else:
        walk()
```

References

1. “http://study.com/directory/category/Engineering/Electrical_Engineering_and_Electronics/Robotics_Technologies.html#:~:text=Robotics%20technologies%20consist%20of%20all,spacecraft%20without%20direct%20human%20intervention.” [Accessed 12 11 2020].

2. Python. [Online]. Available: https://www.python.org/about/. [Accessed 12 11 2020].

3. “ http://www.opencv.org, “[Online]. Available: https://docs.opencv.org/master/d0/de3/tutorial_py_intro.html. [Accessed 12 11 2020].

4. w. i. raspberry-pi. [Online]. Available: https://www.raspberrypi.org/help/what-is-a
raspberry-pi/. [Accessed 13 11 2020].

5. "raspberrypi.org," [Online]. Available: https://www.raspberrypi.org/products/cameramodule/. [Accessed 13 11 2016].

6. “https://www.keyence.com/ss/products/sensor/sensorbasics/ultrasonic/info/”. [Accessed 13 11 2016].

7. “ https://www.electrical4u.com/servo-motor-controller-or-servo-motor-driver/#:~:text=Servo%20motor%20driver%20may%20be,airplane%2C%20which%20uses%20many%20servos.&text=Connectors%2C%20wires%20etc.” [Accessed 14 11 2020].

8. “ https://www.banggood.com/Geekworm-UPS-HAT-Board-2500mAh-Lithium-Battery-For-Raspberry-Pi-3-Model-B-Pi-2B-B-A-p-1105260.html?cur_warehouse=CN” [Accessed 16 11 2020].
9. “ https://www.thingiverse.com/thing:3572854 “. [Accessed 15 10 2020].

10. “ http://www.asus.com”[Online]. Available: https://www.asus.com/us/Single-Board-Computer/Tinker-Board/. [Accessed 14 11 2020].

11. “https://www.jeffgeerling.com/blog/2018/asus-tinker-board-compelling-upgrade-raspberry-pi-3-b”. [Accessed 14 11 2020].
12. “ https://pysource.com/2018/10/31/raspberry-pi-3-and-opencv-3-installation-tutorial/”. [Accessed 2 3 2021].

13. “https://learn.adafruit.com/16-channel-pwm-servo-driver/python-circuitpython “.[Accessed 11 4 2021].
14. “ https://pjreddie.com/darknet/yolo/ “. [Accessed 3 5 2021].
15. “https://learn.adafruit.com/adafruit-arduino-lesson-14-servo-motors/if-the-servo-misbehaves” . [Accessed 28 5 2021].
16. “https://www.youtube.com/watch?v=OTBIXnzcI34”. [Accessed 28 5 2021].
17. “https://raspberrypi.stackexchange.com/questions/80300/camera-gets-hot-how-much-power-should-it-dissipate “.[Accessed 28 5 2021].