Hochschule Reutlingen
Reutlingen University

TEC
Technik

EV3 Hardware and Programming

International Project Engineering

Computer Science for Engineers
IPEB3 WS22/23

by

Encarnacion Ortega - 808412
Ferlando Mkiva - 808238
Laura Orozco - 808415

03 November 2022 - 19 January 2023

Declaration

We have read and understand the **Allgemeine Studien- und Prüfungsordnung für das Bachelor- und Masterstudium der Hochschule Reutlingen - § 13 Täuschung, Ordnungsverstoß, Plagiat** [1]

Therefore, we hereby declare that the work contained within this document is of our own and that all sources and their authors have been listed and appropriately recognized for their contribution in the completion of this document.

STUDENT SIGNATURES

Encarnacion Ortega (808412) **Ferlando Mkiva (808238)** **Laura Orozco (808415)**

[1] https://www.reutlingen-university.de/fileadmin/user_upload/2022_05_23_StuPro_AllgemTeil_WiSe_22-23_2022-05-11.pdf

Executive Summary

Agricultural intensification, mechanization, and automation have all led to major increases in agricultural productivity throughout time [1]. Robots are intelligent machines that may be trained to carry out particular jobs, make choices, and take immediate action. They are needed in a variety of industries where less labor is often needed, and they function best in stable environments where consistent accuracy and high productivity are required [2].

LEF BOTS GmbH is a robot specialising company, the name of the firm is created from the initials of the stakeholders: **L**aura; **E**ncarnacion; **F**erlando **(LEF)**. In this text, the firm presents the development of its major product; the fruit harvesting robot. The robot is given an intuitive name **Fruta Oes**, which is a combination of two languages Spanish and Afrikaans. Fruta means fruits in Spanish and Oes means to harvest in Afrikaans, thus, the name of the robot **Fruta Oes** means fruits harvester. One might be asking themselves why Spanish and Afrikaans, well the answer is simple, the stakeholders of the firm are from Spain (Laura and Encarnacion) and South Africa (Ferlando). Figure 1 Presents **Fruta Oes**.

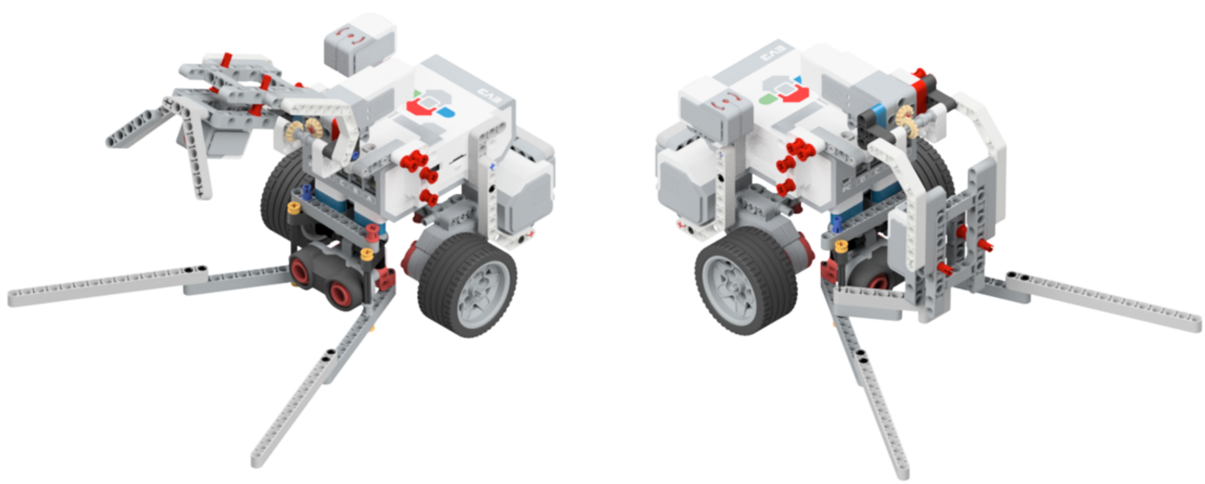


Figure 1: Fruta Oes

The robot mechanical structure is the most important part of this project therefore, this document will discuss in detail all the sub-components of the mechanical design in section 2 on page 2. Following by the software design where a full discussion on all the algorithms used, activity diagrams and testing will be provided in section 3 on page 6. Following will be the management of the software development, section 4 on page 8 highlights the Agile Software engineering (SCRUM) procedure followed in this project. The SCRUM section will briefly discuss the project setup in section 4.1 on page 8, then section 4.2, section 4.3,section 4.4 present the role and progress achieved in sprint 1, sprint 2 and sprint 3 respectively. Finally, recommendations will be detailed in section 5 on page 11 then in the same section this document will go into a concluding discussion.

Table of Contents

List of Figures

List of Tables

1 Introduction

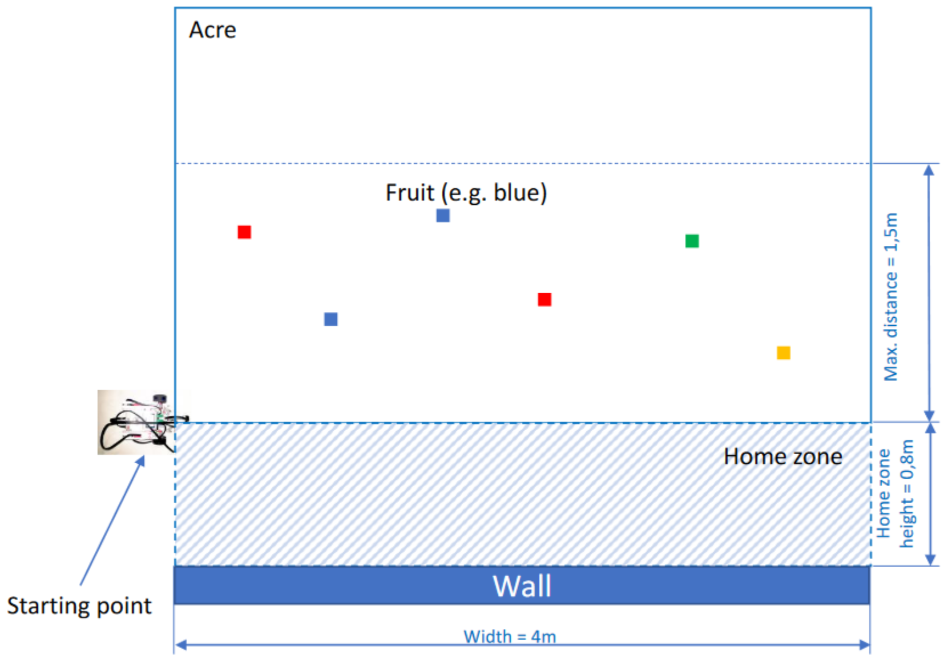


Figure 2: Field [3]

2 Mechanical Design

This section presents the mechanical design of **Fruta Oes** robot. The section will start presenting how the sub-components fit and work together, then proceed by discussing each component in detail. Figure 3 depicts the five main components of **Fruta Oes**.

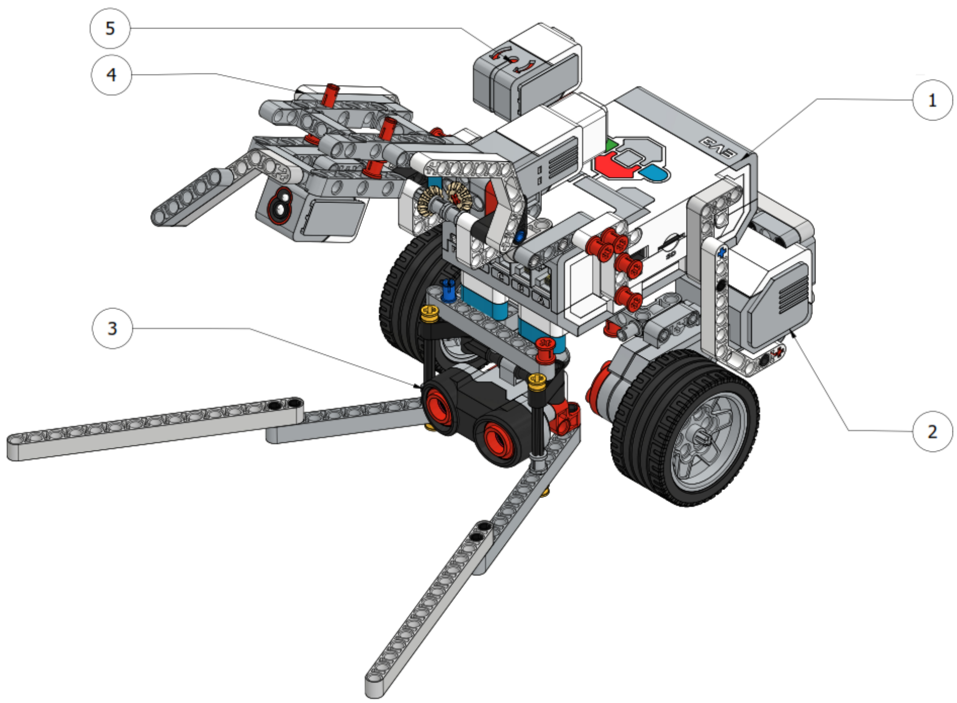


Figure 3: Main Components of Fruta Oes

The five main components of this robot are listed in table 1. **Item 1-EV3 Brick** is the heart and brain of the robot, the processing of the software takes place in this module. **Item 2-Drive Base** serves two purposes; providing stability and mobility, the in-depth design of this module will be discussed in section 2.3 on page 5.

Table 1: Main Components

Item	Component
1	EV3 Brick
2	Drive Base
3	Object Detection and Manipulator Module
4	Gripper and Colour Sensing Module
5	Gyro-Sensor

Item 3-Object Detection and Manipulator Module not only enable object detection capabilities but also makes sure that the object is manipulated to be in a good position for the gripping action. Object Detection and Manipulator Module will be fully discussed in section 2.1 on the next page. **Item 4-Gripper and Colour Sensing Module** grips the object and makes sure it is in a good position for the colour sensor.

2.1 Object Detection and Manipulator Module

This section discusses the object detection mechanism. This module not only enable object detection capabilities but also makes sure that the object is manipulated to be in a good position for the gripping action.

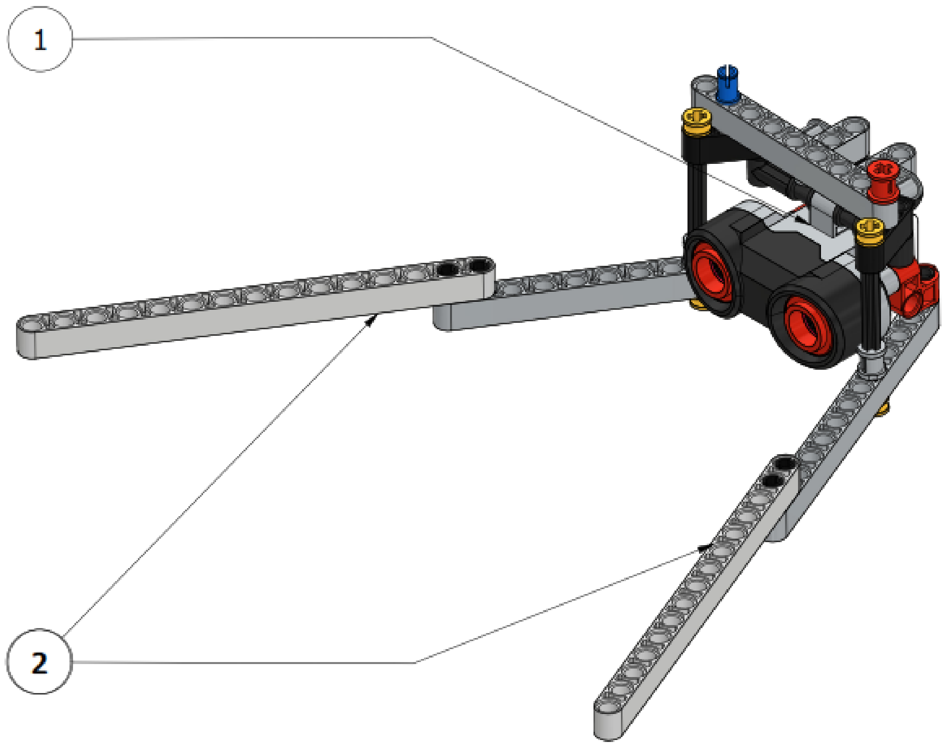


Figure 4: Object Detection and Manipulator Module

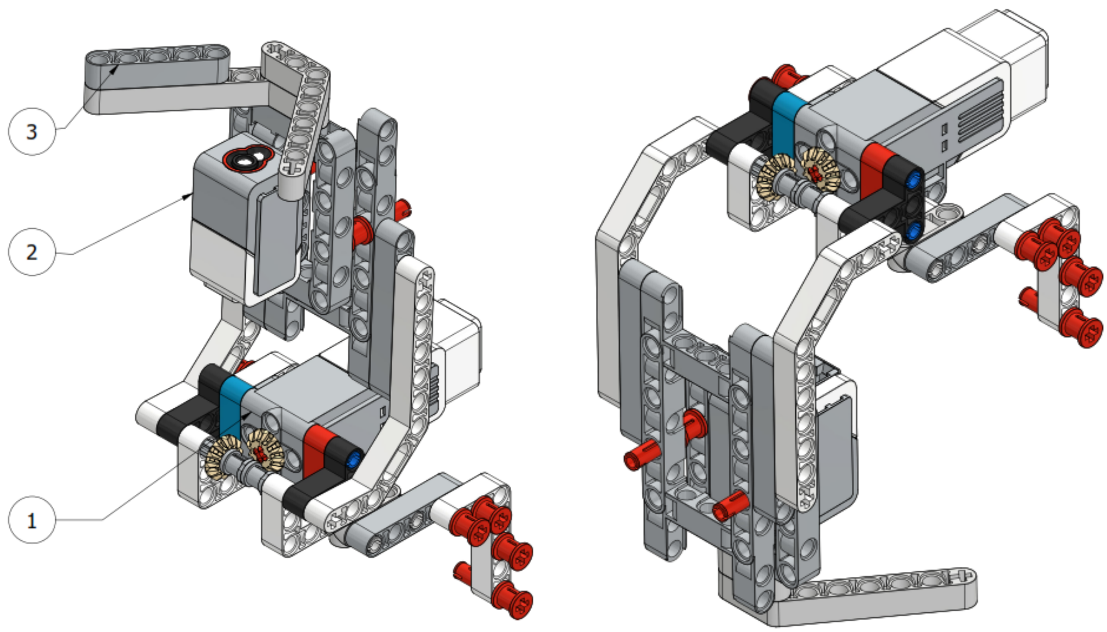
Table 2: Object Detection and Manipulator Module Components

Item	Component
1	Ultra Sonic Sensor
2	Manipulator Fangs

Figure 4 shows the object detecting manipulator, it is composed of two main parts listed in table 2: the ultrasonic Sensor which enables the fruit detection capability for the robot and the manipulator fangs. The ultra-sonic sensor is positioned as low as possible, because the objects to be detected are of 2.5cm height. This sensor is also used to provide distances to the robot as it is also capable of computing the distance to the detected object. The EV3 sensor has a measuring range of 250 cm and an accuracy range of $\pm$ 1cm. During the testing phase of the sensors, it was noted that the sensor is sometimes losing the object when the robot is approaching the detected object, this lead to the introduction of the manipulator fangs which are just LEGO beams positioned in such a way that they manipulate the object to slide and ends up in-front of the sensor. This mechanism improved the object detection and made the gripping operation and colour sensing easier. The latter will be discussed later in the document.

2.2 Gripper and Colour Sensing Module

This section presents the design of the colour sensing and gripping mechanism. This module is also equipped with manipulating fangs. This module is designed to with two states, opened and closed, the module must be fully opened during the searching of fruits in the acre to avoid distracting the signal of the ultrasonic sensor module that was discussed in section 2.1 on the preceding page.



(a) Opened Gripper (b) Closed Gripper

Figure 5: Gripper and Colour Sensing Module

Table 3: Gripper and Colour Sensing Module Components

Item	Component
1	EV3 medium servo motor
2	EV3 Colour Sensor
3	Gripper Manipulator fangs

Figure 5 depicts te gripper and the colour sensing module. figure 5a shows the gripper in opened mode and figure 5b shows the gripper in closed mode. This module comprises of three major components: EV3 medium servo motor, this motor is a great solution when the design of the robot requires shorter response times like the harvesting robot designed in this project. The motor is used for opening and closing mechanism of the gripper, it is equipped with 240 to 250 rpm and 0.08 Nm of running torque. The next component is an EV3 Colour sensor, the main function of this component is to enable the robot to distinguish between the required fruits and non-required fruits. The EV3 Colour Sensor is capable of detecting seven colours plus the absence of colour. It can tell the difference between colour or black-and-white or among blue, green, yellow, red, white, and brown. This sensor samples at a rate of 1 kHz, it must be positioned 1cm away from the object for better results.

2.3 Drive Base

In this section, the assembling of the drive base for the robot is discussed. it is made up of two parts: EV3 large motors and the wheels, also listed in table 4. there is also a balancing wheel positioned underneath the base close to the rear but this wheel is not shown in figure 6.

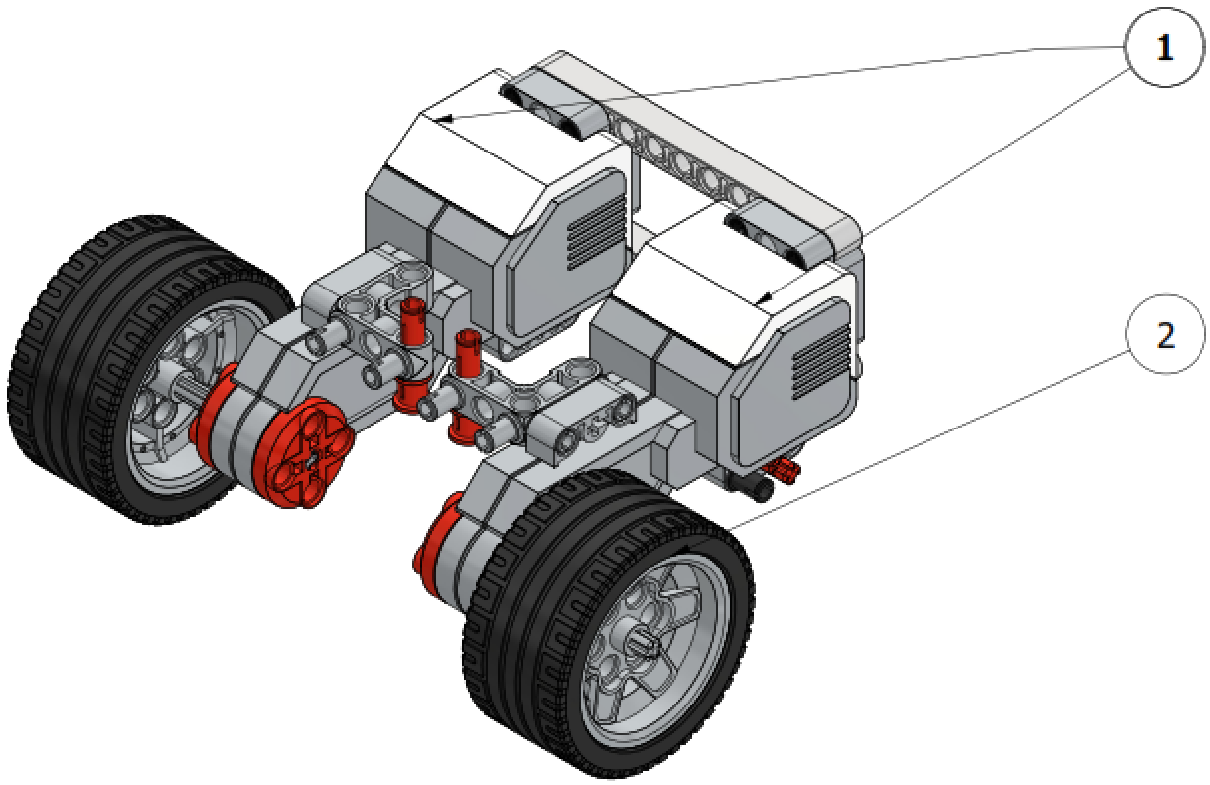


Figure 6: Drive Base

Table 4: Drive Base Components

Item	Component
1	EV3 Large Motors
2	Wheels

The EV3 Large Servo Motor uses tacho feedback for accurate control to within one degree [4].

3 Software Design

3.1 Main Code

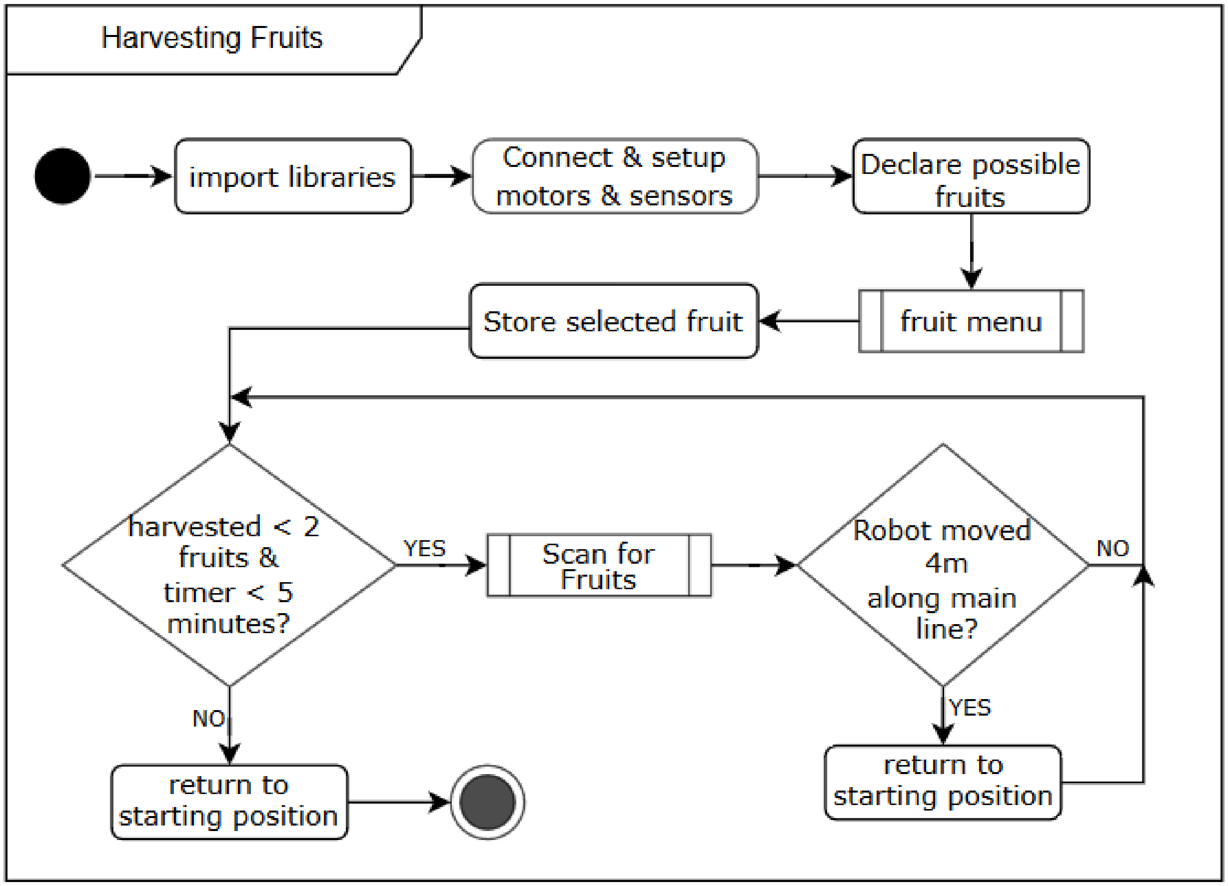


Figure 7: Main Code Activity Diagram

3.2 User Menu for Selecting the Colour (fruit)

As it can be seen in figure 1 on page ii, the EV3 brick buttons are coloured with all possible colours of the project: Left button for Blue, Up button for Red and right button for green. The user must wait for the beep after the robot asked them to select the colour from the buttons. Based on the pressed button, the robot will confirm the chosen colour and proceed to search for that specific fruit in the acre. Figure 8 presents the activity diagram for User Menu function, the corresponding source code is attached on appendix B.

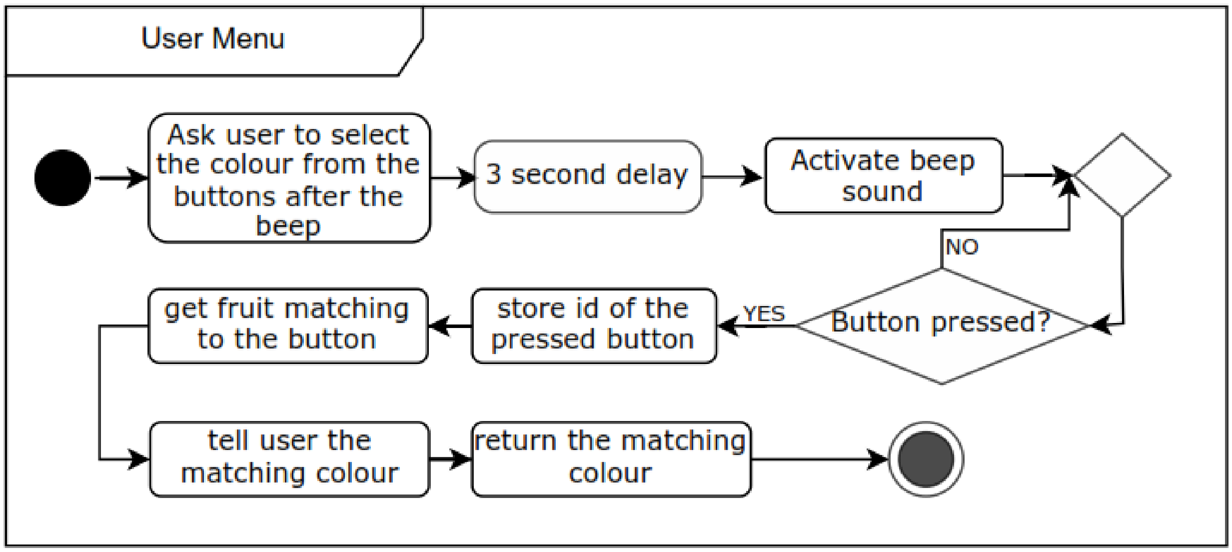


Figure 8: Menu Code Activity Diagram

Algorithm 1 presents a pseudo code for this user menu method. A five minute timer is started at the beginning of the function, because one of the requirements is that the robot must return to the starting position after five minutes and this time must be measured as soon as the robot starts interacting with the user. This algorithm is constructed from the activity diagram above.

Algorithm 1 : User Menu Method Pseudo Code

Start the 5 minutes timer;
Use EV3 Speaker function to ask the user to select the colour from the buttons after the beep;
sleep for 3 seconds to allow the sound command to complete;
Activate the EV3 Beep Sound;
while *True* **do**
    if Any button is pressed **then**
        Store the id of the pressed button;
        break out of the while loop;
    end if
end while
Use the id of the pressed button to get matching fruit from the fruit dictionary;
Use EV3 speaker to tell the user the matching colour;
Return the matching colour to the rest of the code;

4 Agile Software Engineering

4.1 Project Setup

LEF BOTS GmbH is a robot specialising company, the name of the firm is created from the initials of the stakeholders: **L**aura; **E**ncarnacion; **F**erlando **(LEF)**. In this text, the firm presents the development of its major product; the fruit harvesting robot. The robot is given an intuitive name **Fruta Oes**, which is a combination of two languages Spanish and Afrikaans. Fruta means fruits in Spanish and Oes means to harvest in Afrikaans, thus, the name of the robot **Fruta Oes** means fruits harvester. One might be asking themselves why Spanish and Afrikaans, well the answer is simple, the stakeholders of the firm are from Spain (Laura and Encarnacion) and South Africa (Ferlando). Figure 1 on page ii Presents **Fruta Oes**.

4.1.1 Project Vision

"For harvesters who care for the environment. We believe in a world where harvesters can harvest being eco-friendly"

4.1.2 SCRUM Team

Figure 9 presents the scrum team: Encarnacion Nunez-Ortega is the **Product Owner** and responsible for determining what needs to be done. Laura Ponce-Orozco is the team's **Scrum master** and responsible for removing all impediments. The **development team** consist of Ferlando Mkiva, Encarnacion Nunez-Ortega and Ponce-Orozco together will determine how to deliver chunks of work in frequent increments.

Figure 9: SCRUM Team

4.1.3 Product Backlog

Table 5: Product Backlog

Priority	User Story	Effort
1	As a user, I want the robot to be completely autonomous.	1
2	As a user, I want the robot to ask me for the correct fruit and to confirm that it understood me.	2
3	As a developer, I want the robot to be able to detect a 2.5cm cube.	1
4	As a user, I want the robot to look for the specified fruit and check if the fruit is correctly grabbed.	5
5	As a user, I don't want the robot to return to the home zone without the fruit grabbed.	4
6	As a user, I want the robot to place the right fruit in the home zone.	3
7	As a developer, I want the robot to keep count of the harvested fruits	1
8	As a user, I want the robot to immediately return to the starting point after collecting two fruits within 5 minutes.	4
9	As a user, I don't want the robot to search for fruits outside the acre.	3
10	As a developer, I want the code to be efficient.	4

4.2 Sprint 1

4.2.1 Goals of the sprint

The goal of this sprint was to tackle the following user stories. In total five cups of coffees must be consumed as the effort of this sprint.

Priority 2

As a **user**, I want the robot to ask me for the **correct fruit** and to confirm that it understood me. 2

Priority 3

As a **developer**, I want the robot to be able to **detect** a 2.5cm cube. 1

Priority 4

As a **user**, I want the robot to look for the **specified fruit** and check if the fruit is correctly grabbed. 2

4.2.2 Burn-down Chart

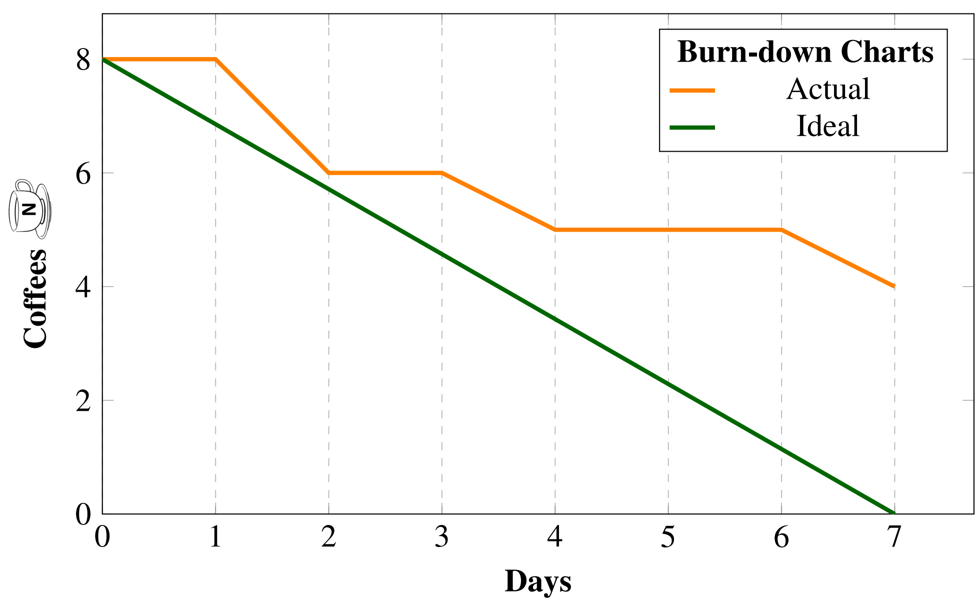


Figure 10: Sprint 1 Burnd-own Chart

4.3 Sprint 2

4.3.1 Goals of the sprint

The goal of this sprint was to tackle the following user stories. In total five cups of coffees must be consumed as the effort of this sprint.

Priority 2

As a **user**, I want the robot to place the right fruit in the home zone. 3

Priority 3

As a user, I don't want the robot to return to the home zone without the fruit grabbed. 1

Priority 4

As a user, I want the robot to look for the specified fruit and check if the fruit is correctly grabbed. 2

Priority 4

As a user, I want the robot to look for the specified fruit and check if the fruit is correctly grabbed. 2

4.4 Sprint 3

5 Recommendations and Conclusion

References

[1] Nof SY, Nof SY. Springer handbook of automation. Springer; 2009.

[2] Nof SY. Emerging trends and industry needs handbook of industrial robotics. John Wiley & Sons; 1999.

[3] Danner M, Wachter M. Computer Science for Engineers: Phase 3 Challenge - Harvester. Reutlingen University - International Project Engineering. 2022;IPEB3 - WS22/23.

[4] Robots LR. Raising Robots MINDSTORMS EV3 Large Servo Motor; 2023. Available from: `https://raisingrobots.com/product/large-servo-motor/`.

Note: diagrams and graphs are works of the report authors. Specifically, drawings are created using Microsoft Viso Professional 2019 and Autodesk Inventor Student version 2023, algorithms are constructed using the LaTeX "algpseudocode" package, and Code appendices are constructed using the LaTeX "listings" package (with data obtained from MATLAB® R2016a Simulink). No work done by other persons is presented as that of the report author.

Appendices

A Library importing and Device Setup

Code 1: Library importing and Device Setup

```
#!/usr/bin/env python3
import ev3dev.ev3 as ev3
from ev3dev.ev3 import * #to allow the use of brick buttons
import math #importing Math Library to allow usage of Mathematical functions
import ColorSensor2 #importing Color sensor library
from time import sleep #importing time library
import time
#-------------------------------------------------------------------
#Device Connections start
# Connect the outputs to the motor
motor_left = ev3.LargeMotor('outA')
motor_right = ev3.LargeMotor('outD')
gripper_motor = ev3.MediumMotor('outC')

#Gyro Sensor initialisation
gy = ev3.GyroSensor()

#Colour Sensor initialisation
cl= ColorSensor2.ColorSensor2()
cl.mode='COL-COLOR'

#Ultra-Sonic Sensor initialisation
us = ev3.UltrasonicSensor()
us.mode='US-DIST-CM'

btn = Button() #activate the use of brick buttons

#Global Variables (we know use of global variables is not a good practice, but in this case it is necessary)
harvested = 0  #keeping count of the harvested fruits
total_distance = 0  #keeping track of the distance travelled by the robot
gripper_opened = True   #keeping track of the state of the gripper
thetha = None   #To avoid the scanning angle from exceeding the limits
start = 0   #Variable to start the 5 minute timer
end = 0 #Variable to end the 5 minute timer
current_time = 0    #Variable to track the current time
#Device Connections ends
```

B User Menu for Selecting the Colour

Code 2: User Menu for Selecting the Colour

```
def menu():
#Function to select the desired fruit using the buttons from the Brick
global start
ev3.Sound.speak("Select the colour from the buttons after the beep").
    wait()
sleep(3)
Sound.beep().wait()
start = time.time() #Starting the timer
while True:
if btn.any():# Checks if any button is pressed.
button_id = btn.buttons_pressed
print(button_id[0])
break
else:
sleep(0.01)

option_text = " I will search for {0}" #Robot confirming the colour
ev3.Sound.speak(option_text.format(fruit_dict.get(button_id[0]))).wait
    ()
return fruit_dict.get(button_id[0]) #Returning the desired fruit
```

C Function for turning left

Code 3: Function for turning left

```
def turn_left(speed,angle):
#Function that turns left, it requires the turning speed and the angle
    it must turn
gy.mode = 'GYRO-RATE'
gy.mode = 'GYRO-ANG'
sleep(1)
while gy.value()>-angle:
motor_left.run_forever(speed_sp=-speed, stop_action = 'hold')
motor_right.run_forever(speed_sp=speed, stop_action = 'hold')
#print(gy.value())
motor_left.stop()
motor_right.stop()
```

D Function for turning right

Code 4: Function for turning right

```
def turn_right(speed,angle):
#Function that turns right, it requires the turning speed and the angle
    it must turn
gy.mode = 'GYRO-RATE'
gy.mode = 'GYRO-ANG'
sleep(1)
while gy.value()<angle:
motor_left.run_forever(speed_sp=speed, stop_action = 'hold')
motor_right.run_forever(speed_sp=-speed, stop_action = 'hold')
#print(gy.value())
motor_left.stop()
motor_right.stop()
```

E Function for moving the robot forward

Code 5: Function for moving the robot forward

```
def move_distance_in_mm(distance_mm):
#Function to move a specified distance, it transforms rotation of the
    wheels to distance.
dist = (distance_mm/1000) * (360/0.176)
motor_left.run_to_rel_pos(position_sp = dist, speed_sp=400, stop_action
    = "brake")
motor_right.run_to_rel_pos(position_sp = dist, speed_sp=400,
    stop_action = "brake")
motor_left.wait_while('running')
motor_right.wait_while('running')
motor_left.stop()
motor_right.stop()
```

F Function for moving the robot from Acre to Home-Zone

Code 6: Function for moving the robot from Acre to Home-Zone

```
def drive_to_home_zone(hypotenuse, current_angle, add):
#Function to drive to the home zone from the Acre with a correct fruit.
global total_distance, harvested, start, end, current_time
distance = abs(hypotenuse*(math.sin(math.radians(current_angle)))) # Using Pythagorean theorem to calculate the distance from the fruit to the home-zone
total_distance += abs(hypotenuse*(math.cos(math.radians(current_angle)))) #Adding the distance moved forward to the total for returning to the starting point
turn_right(100,current_angle + 85) #turning right 90 degrees (we used 85 because the Gyro-sensor was not accurate)
gy.mode = 'GYRO-RATE'
gy.mode = 'GYRO-ANG'
sleep(1)
move_distance_in_mm(abs(distance + add))
gripper("open") #Open the gripper to release the fruit in the home-zone
harvested += 1  #keeping count of the harvested fruits
move_distance_in_mm(-add) #returning to the main-line
turn_left(200,85) #Turning left to continue searching
end = time.time()
current_time += (end - start) #Checking the current time
```

G Function for moving the robot from Acre to Main-Line

Code 7: Function for moving the robot from Acre to Main-Line

```
def drive_from_acre_to_line(hypotenuse, current_angle):
#Function to drive to the main-line from the Acre without any fruit.
global total_distance, start, end, current_time
distance = abs(hypotenuse*(math.sin(math.radians(current_angle))))
print(distance)
turn_left(1000, 80) #moving a wrong fruit away so that it cannot be detected again.
sleep(1)
turn_right(200,current_angle) #turning the robot perpendicular to the main line after moving a wrong fruit.
total_distance += abs(hypotenuse*(math.cos(math.radians(current_angle)))) #Adding the distance moved forward to the total for returning to the starting point
gy.mode = 'GYRO-RATE'
```

```
gy.mode = 'GYRO-ANG'
sleep(1)
move_distance_in_mm(-abs(distance)) #Reversing the robot to the main-
    line
```

H Function for operating the Gripper

Code 8: Function for operating the Gripper

```
def gripper(command):
#Function to open and close the gripper using the Medium Motor
global gripper_opened
if (command == "close" and gripper_opened == True):
gripper_motor.run_to_rel_pos(speed_sp=150, position_sp=230, stop_action
     = 'brake') # closing
sleep(2)
gripper_opened = False
elif (command == "open" and gripper_opened == False):
gripper_motor.run_to_rel_pos(speed_sp=250, position_sp=-210,
    stop_action = 'brake') #opening
gripper_motor.wait_while('running')
gripper_opened = True
```

I Function for Searching the fruits

Code 9: Function for Searching the fruits

```
def scanning():
#Function for scanning the acre for nearby fruits
global total_distance, start, end, current_time, thetha
gy.mode = 'GYRO-RATE'
gy.mode = 'GYRO-ANG'
sleep(1)
search_space = 1500 #scanning for fruits withing 1.5m (Width of the
    acre)
scanning_speed = 50
dist = us.value() #Storing the distance given by the Ultra-Sonic sensor
while(dist > search_space or thetha == None): #Only detect fruits which
     are withing the search space
while(dist > search_space and gy.value()>-85): #It scans turning left
    90 degrees
motor_left.run_forever(speed_sp=-scanning_speed, stop_action = 'hold')
```

```
motor_right.run_forever(speed_sp=scanning_speed, stop_action = 'hold')
dist = us.value() #Distance at which the fruit is detected
motor_left.stop()
motor_right.stop()

while(dist > search_space and gy.value()<-5): #it scans turning right 90 degrees
motor_left.run_forever(speed_sp=scanning_speed, stop_action = 'hold')
motor_right.run_forever(speed_sp=-scanning_speed, stop_action = 'hold')
dist = us.value() #Distance at which the fruit is detected
motor_left.stop()
motor_right.stop()

thetha = gy.value() #Angle at which the fruit is detected
dist =us.value() #Distance at which the fruit is detected

if(dist > search_space): # If no fruits detected, advance the robot 20 cm
move_distance_in_mm(200)
total_distance += 200 #Adding the distance moved forward to the total for returning to the starting point

end = time.time()
current_time += (end - start) #Checking the current time

while(us.value() < search_space): #optimizing the exact angle at which the fruit is detected, turn the robot until the fruit is lost/no_longer_detected
motor_left.run_forever(speed_sp=-50, stop_action = 'hold')
motor_right.run_forever(speed_sp=50, stop_action = 'hold')
motor_left.stop()
motor_right.stop()
alpha = gy.value() - thetha #Angle at which the fruit is lost when turning

while (gy.value()< (thetha + (alpha/2))): #Aligning the robot with the center of the fruit
motor_left.run_forever(speed_sp=50, stop_action = 'hold')
motor_right.run_forever(speed_sp=-50, stop_action = 'hold')
motor_left.stop()
motor_right.stop()

if(dist > 500): #If the fruit is more than 0.5m away, the robot must stop halfway and scan again.
```

```
move_distance_in_mm(dist/2)
while(us.value() < search_space):
motor_left.run_forever(speed_sp=-50, stop_action = 'hold')
motor_right.run_forever(speed_sp=50, stop_action = 'hold')
motor_left.stop()
motor_right.stop()
alpha = gy.value() - thetha

while (gy.value()< (thetha + (alpha/2))):
motor_left.run_forever(speed_sp=50, stop_action = 'hold')
motor_right.run_forever(speed_sp=-50, stop_action = 'hold')
motor_left.stop()
motor_right.stop()
move_distance_in_mm(dist/2+10)
else:
move_distance_in_mm(dist+10) #Moving forward 1cm more to force the fruit to be below the colour sensor

sleep(1)
gripper('close') #Closing the gripper to allow colour sensing function

if (cl.getCalibratedColorString() == fruit): #if the sensored fruit is correct
ev3.Sound.speak("Right")
drive_to_home_zone(dist, abs(gy.value()), 300) #Drive it to the Home-zone
else:
ev3.Sound.speak("wrong") #if the sensored fruit is wrong
gripper('open') #open the gripper
drive_from_acre_to_line(dist, abs(gy.value())) #move the fruit away
turn_right(200,85)
end = time.time()
current_time += (end - start) #Checking the current time
```

J Main Code

Code 10: Main Code

```
fruit_dict ={   #Declaring the available fruits
  #Dictionary of all possible fruits (Coloured cubes) in the competition and corresponding selection button on the EV3 brick
  'up':"Red",
  "right":"Green",
  "left":"Blue",
```

```
  "down":"Yellow",
}

fruit = menu() #Storing the desired fruit

while (harvested<2  and current_time<300): #when it reaches 300s (5 minutes) or when it harvested 2 fruits, it must come back to the starting point
scanning()
if(total_distance >= 4000): #If the robot drove the whole length of the acre, it must return to the starting point and scan again
move_distance_in_mm(-total_distance)
total_distance = 0

move_distance_in_mm(-total_distance) #Reversing the robot back to the starting Position
```