UNIVERSIDAD DE TARAPACÁ

FACULTAD DE INGENIERÍA

DEPARTAMENTO DE INGENIERÍA CIVIL EN COMPUTACIÓN E INFORMÁTICA

Ingeniería@
Computación e Informática

Plan de Proyecto
"Wheatley Robotics"

Alumno(os): Renato Almeyda
Martín Castillo
Osvaldo Costagliola
Adiel Espinoza
Nicolás Zarzuri

Asignatura: Proyecto I

Profesor: Humberto Urrutia López

12 – 2024

Historial De Cambios

Fecha	Versión	Descripción	Autor(es)
09/08/2024	1.0	Formulación del Informe 1	Renato Almeyda Martín Castillo Osvaldo Costagliola Adiel Espinoza Nicolás Zarzuri
13/11/2024	2.0	formulación del informe 2	Martín Castillo Osvaldo Costagliola
9/12/2024	3.0	Formulación del informe final	Martín Castillo

Tabla de Contenidos

1. Panorama General

1.1. Introducción

El presente informe tiene como objetivo documentar el desarrollo y los resultados obtenidos durante la ejecución del proyecto “Wheatley Robotics”, realizado como parte del curso Proyecto I. Este proyecto consiste en la construcción y programación de un robot basado en la plataforma Lego Mindstorms EV3, diseñado para desplazarse y manipular una pelota de ping pong mediante una interfaz gráfica de usuario.

A lo largo de este informe, se detallarán los objetivos generales y específicos del proyecto, las metodologías utilizadas, la planificación de las actividades y los recursos empleados. Además, se presentarán los desafíos enfrentados durante el desarrollo, las soluciones implementadas y los resultados obtenidos en cada una de las etapas.

Este trabajo no solo ha permitido el desarrollo de competencias técnicas en programación, diseño y control de robots, sino también habilidades de trabajo en equipo, gestión de proyectos y resolución de problemas, aspectos fundamentales en la formación de futuros profesionales en ingeniería informática.

Finalmente, se analizarán las conclusiones derivadas del proceso, identificando las lecciones aprendidas y las oportunidades de mejora que podrán ser aplicadas en futuros proyectos.

1.2. Objetivos

1.2.1. Objetivo General

Construir y programar un robot con Lego Mindstorms Ev3 capaz de levantar una pelota, desplazarla y dejarla en el suelo nuevamente a través de una interfaz de usuario.

1.2.2. Objetivos Específicos

- Desarrollar un robot desde la problemática entregada.
- Comprender la conectividad que necesita el robot para ser controlado.
- Diseñar soluciones eficientes para la problemática entregada.
- Utilizar la librería Tkinter de Python para crear una GUI fácil de usar.

1.3. Restricciones

- Disponibilidad limitada del robot.
- Límite de tiempo en las fases de construcción, desarrollo y documentación.
- Se debe usar solo Python como lenguaje de programación.
- Se debe utilizar la plataforma Redmine para documentar el proyecto.

1.4. Entregas

Bitácoras: Son informes semanales con las actividades, problemas y avances que se tuvieron durante la elaboración del proyecto, donde también se asignan temas a tratar en futuras reuniones.

Carta Gantt: Es un método de organización de proyectos en una línea de tiempo, donde se separan y asignan las tareas de un proyecto, marcando avance en un determinado tiempo.

Informe de Formulación: Un informe que muestre la organización del equipo, avances en el proyecto, problemas que se presentaron y el proceso que se tuvo.

Manual de Usuario: Un manual que presentará de manera clara e intuitiva cómo controlar el robot y sus funciones.

Presentaciones: En las presentaciones se muestran los avances del proyecto, además de mostrar los objetivos y organización del personal.

Interfaz del robot: Una interfaz creada en Pyhton con la librería Tkinter para controlar el robot de manera remota.

Robot: El diseño y construcción de un robot Lego MindStorms ev3.

2. Organización del Personal

2.1. Descripción de los Roles

Jefe de proyecto: Responsable de organizar y supervisar el avance del proyecto y las tareas del equipo.

Ensamblador: Encargado de diseñar y probar piezas para el robot, cuidando la funcionalidad de este.

Programador: Encargado de codificar y crear la conectividad necesaria para controlar el robot.

Documentador: Responsable de crear documentación para el seguimiento del proyecto y su organización.

Diseñador: Responsable de diseñar el logotipo, estética e interfaz de la documentación del proyecto.

2.2. Personal que Cumplirá los Roles

Rol	Responsable	Involucrados
Jefe de proyecto	Martín Castillo	Martín Castillo
Ensamblador	Nicolás Zarzuri	Nicolás Zarzuri Osvaldo Costagliola
Diseñador	Martín Castillo	Martín Castillo
Programador	Adiel Espinoza	Adiel Espinoza Renato Almeyda
Documentador	Martín Castillo	Martín Castillo Osvaldo Costagliola

2.3. Métodos de Comunicación

El principal método de comunicación y traspaso de información ha sido Whatsapp, ya que todo el avance del proyecto mayormente se ha realizado en clases.

3. Planificación del Proyecto

3.1. Actividades

Fase I:

Nombre	Descripción	Responsables	Producto
Conteo de piezas	Se cuentan las piezas del robot.	Nicolás Zarzuri	Análisis y reconocimiento de piezas.
Creación del prototipo 1 del robot	Se construye el primer prototipo del robot.	Nicolás Zarzuri, Osvaldo Costagliola	Prototipo funcional del robot para hacer pruebas.
Conectividad con el robot	Se conecta el robot a Wifi y se vincula con el computador para poder controlarlo.	Adiel Espinoza, Renato Almeyda	Posibilidad de controlar al robot.

Modificaciones en el robot	Se buscan nuevos diseños para la garra del robot.	Nicolás Zarzuri, Osvaldo Costagliola	Nuevas formas de lograr que el robot levante la pelota
Conectividad del robot	Se crean los archivos y librerías necesarias para poder controlar el robot de forma remota.	Adiel Espinoza, Renato Almeyda	Posibilidad de mover el robot de forma remota.
Pruebas con el robot	Se testea si el robot es capaz de moverse con comandos a tiempo real.	Adiel Espinoza, Renato Almeyda	Se comprueba si el robot es capaz de moverse y se analizan posibles problemas.
Primer informe de avance	Se crea el informe según el avance del proyecto.	Martín Castillo	Informe de avance entregable
Presentación del avance del proyecto	Se crea la presentación para mostrar los avances del proyecto.	Martín Castilo	Presentación de avance del proyecto.

Fase II:

Nombre	Descripción	Responsables	Producto
Redacción de las bitácoras	Documento donde se registra lo trabajado en la semana	Martin Castillo Osvaldo Costagliola	Entrega de bitácoras semanales
Videos y fotos	Registro visual del avance del proyecto	Martin Castillo	Fotos o videos del proyecto para luego utilizarlo en la wiki
Modificación de la arquitectura del robot	Se realizan cambios por problemas con el diseño.	Nicolás Zarzuri	Nuevas formas de lograr que el robot levante la pelota

Codificación del servidor	Encargado de la comunicación entre la interfaz y el robot.	Adiel Espinoza, Renato Almeyda	Enlace entre la interfaz y el robot
Actualización de la wiki	Se agregan nuevos aspectos desarrollados en la 2da fase del proyecto	Osvaldo Costagliola	Actualización de la wiki.
Conexión	Establecer la conexión con la interfaz y el robot.	Adiel Espinoza, Renato Almeyda	Encargado de generar conexión entre robot y el interfaz
Elaboración del 2do informe	Se agregan aspectos trabajados en la segunda fase.	Martín Castillo Osvaldo Costagliola	Informe de avance entregable
Creación de la 2da presentacion	Presentación que muestra los puntos más importantes en esta fase.	Martín Castilo	Presentación de avance del proyecto.

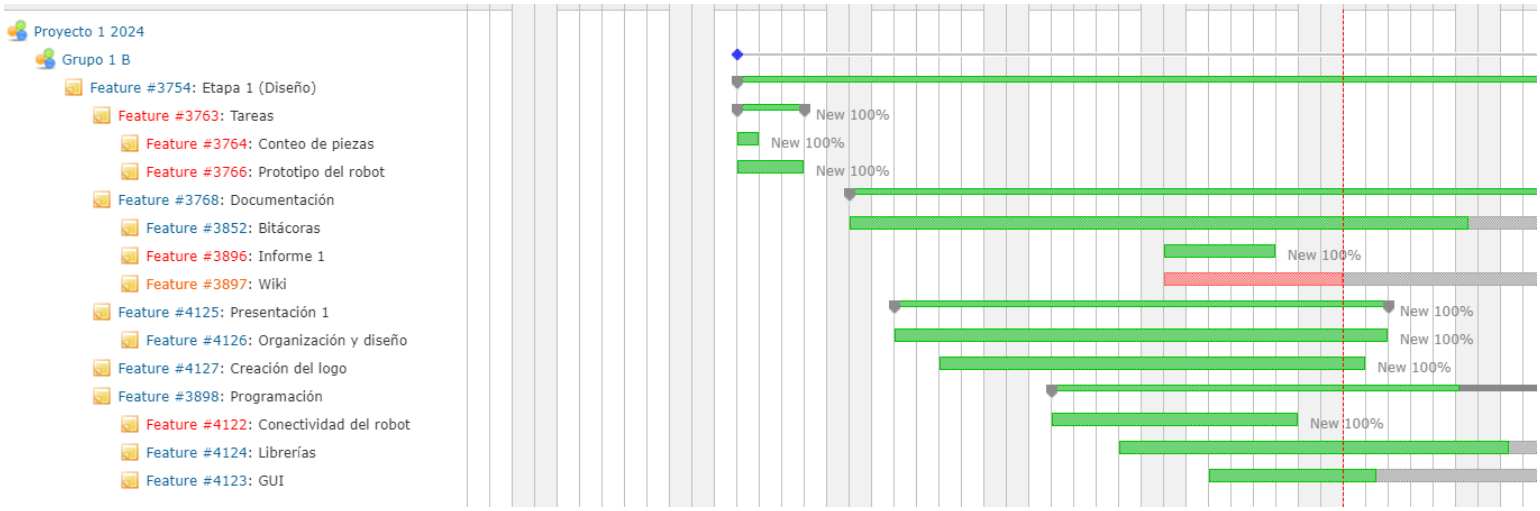
Fase III:

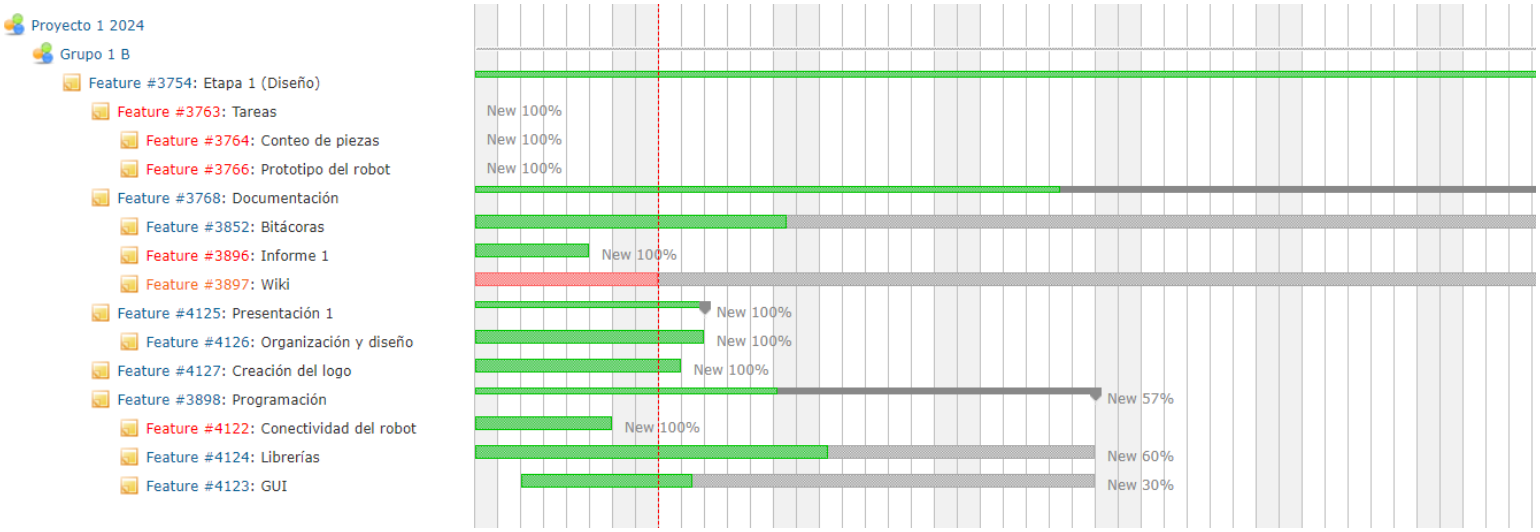
Nombre	Descripción	Responsables	Producto
Redacción informe final	Documento donde se registra el avance del proyecto y su proceso.	Martin Castillo	Informe final del proyecto.
Manual de usuario	Documento que describe la interfaz y la forma de usar el robot.	Adiel espinoza, Martín Castillo	Manual de usuario para el robot.
Pruebas con el robot	Se hacen pruebas para probar la funcionalidad del robot y realizar cambios.	Nicolás Zarzuri	Pruebas con el robot.

Creación de la 3ra presentación	Se hace una síntesis del proyecto para ser presentado.	Martín Castillo	Presentación final del proyecto.

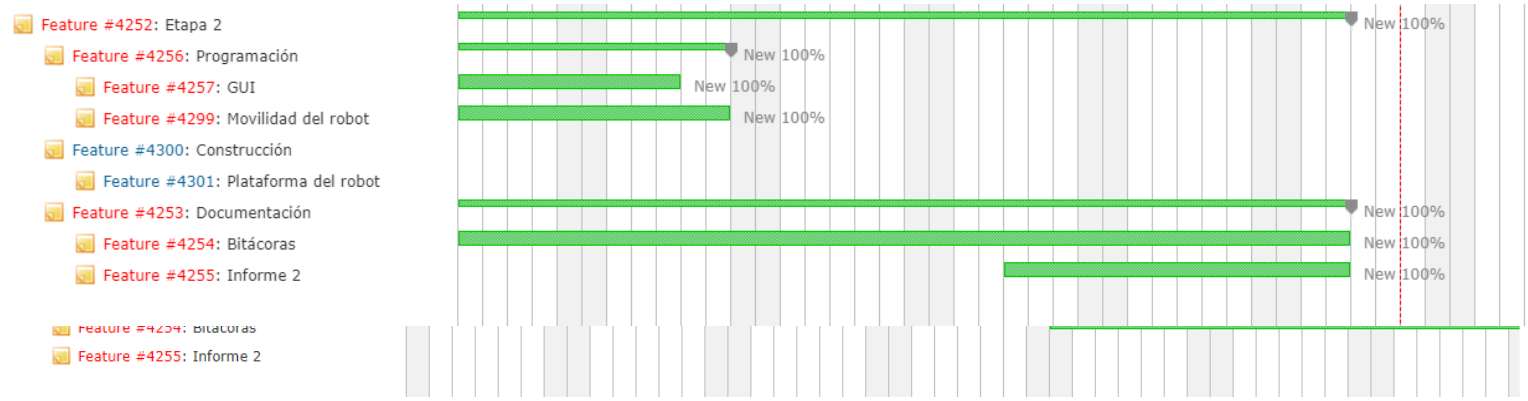
3.2. Carta Gantt

Fase I:

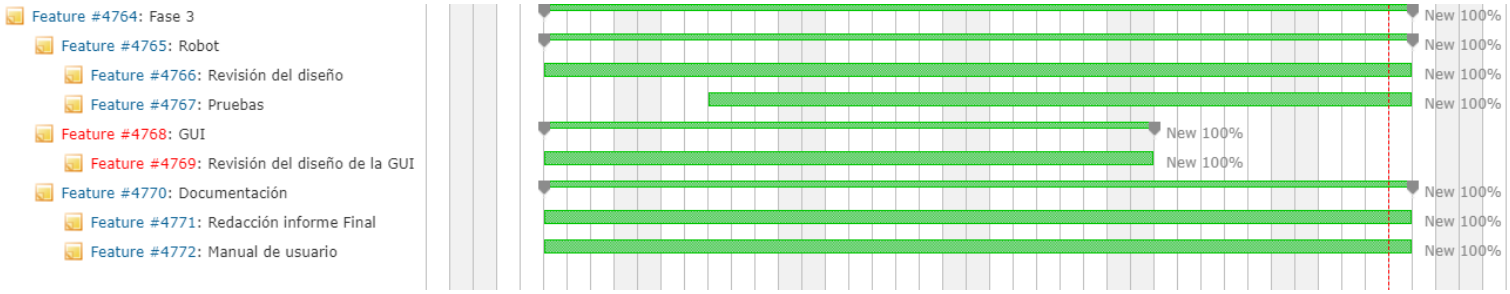




Fase II:


Feature #4252: Etapa 2
Feature #4256: Programación
Feature #4257: GUI
Feature #4299: Movilidad del robot
Feature #4300: Construcción
Feature #4301: Plataforma del robot
Feature #4253: Documentación
Feature #4254: Bitácoras
Feature #4255: Informe 2
New 100%
New 100%
New 100%
New 100%
New 100%
New 100%
New 100%
Feature #4255: Informe 2

Fase III:


Feature #4764: Fase 3
Feature #4765: Robot
Feature #4766: Revisión del diseño
Feature #4767: Pruebas
Feature #4768: GUI
Feature #4769: Revisión del diseño de la GUI
Feature #4770: Documentación
Feature #4771: Redacción informe Final
Feature #4772: Manual de usuario
New 100%
New 100%
New 100%
New 100%
New 100%
New 100%
New 100%
New 100%
New 100%

3.3. Gestión de Riesgos

En esta sección se mostrarán los distintos problemas a los que se ha enfrentado el proyecto, además de la clasificación de su riesgo y cómo lo resolvimos.

1. *Daño catastrófico*
2. *Daño crítico*
3. *Daño circunstancial*

Riesgo	Probabilidad de Ocurrencia	Nivel de Impacto	Acción Remedial
Desarme del robot por mala manipulación	30%	3	El ensamblador arregla el problema rápidamente volviendo a armar la sección desarmada.
Enfermedad o inconveniente de personal	50%	3	Se cubren o se reorganizan las tareas del integrante para minimizar retrasos.
Horas de trabajo fuera de clases escasas	10%	3	Se buscan horarios comunes para planificar y organizar al equipo.
Error en la codificación	60%	3	Los desarrolladores se dedican a este problema e investigan para encontrar una solución.
Retraso en las tareas pautadas	30%	2	Se hace un enfoque especial en esta tarea y entre todo el equipo se avanza con esta.
Incumplimiento de tareas	40%	2	Reorganización para priorizar tareas atrasadas y volver a un curso normal.

Interrupción por choque de horarios	10%	2	Seguir con el avance en la medida de lo posible, estableciendo horarios para el desarrollo del proyecto.
Demora con la construcción del robot	50%	3	Los Ensambladores se dedican a este problema e investigan para encontrar una solución.

4. Planificación de los Recursos

4.1. Hardware

- Set Lego Mindstorm EV3.
- Micro SD.
- Computador con los programas necesarios para programar.

4.2. Software

- Visual Studio Code, editor de código.
- krita, software de dibujo.
- libresprite, software de dibujo a pixeles.
- Canva, presentaciones.
- google docs, documentación.

4.3. Estimación de Costos

Costo de Hardware:

Producto	Precio
Set Lego Mindstorms (EV3)	$ 1.229.000
3 Notebooks (700.000 c/u)	$ 2.100.000

Micro SD	$ 12.990
Total:	$ 3.341.990

Costo de Software:

Producto	Precio
Visual Studio Code	$ 0
Krita	$ 0
libresprite	$ 0
Canva	$ 0
Google Docs	$ 0
Total :	$ 0

Costo de Trabajador:

Rol	Horas	Horas Extra	Precio / Hora
Jefe de proyecto	40 horas	3 horas	$ 30.000
Programador	40 horas	8 horas	$ 25.000
Ensamblador	40 horas	8 horas	$ 25.000
Documentador	40 horas	3 horas	$ 25.000
Total:	-	-	$ 4.765.000

Total de Costo:

Costo Hardware	$ 3.341.990
Costo Software	$ 0
Costo Empleados	$ 4.765.000
Total:	$ 8.106.990

5. Análisis – Diseño

5.1. Especificación de requerimientos

5.1.1 Requerimientos funcionales

- Desarrollar un robot que se comunique vía wifi y permita al usuario controlarlo mediante una interfaz gráfica en Python

- Capacidad para moverse en direcciones hacia adelante, atrás, izquierda, derecha y capaz de levantar la pelota de ping pong

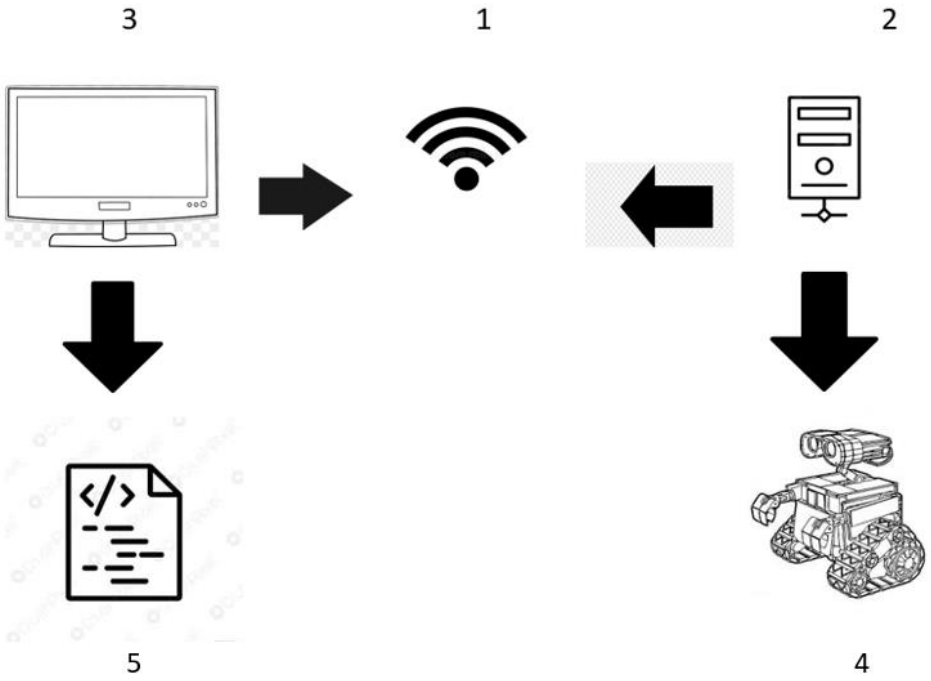
- La interfaz gráfica debe ofrecer opciones específicas para acciones como desplazarse, mover la pala de la bola y realizar el levantamiento.

5.1.2 Requerimientos no funcionales

- El proyecto debe incluir un manual detallado con instrucciones completas sobre el funcionamiento integral del robot.

- La interfaz gráfica debe contar con botones específicos para controlar el desplazamiento del robot, una sección para ajustar la pala para levantar la pelota

5.2 Arquitectura



1. Ambos dispositivos deben estar conectados a la misma red wifi para establecer la comunicación.

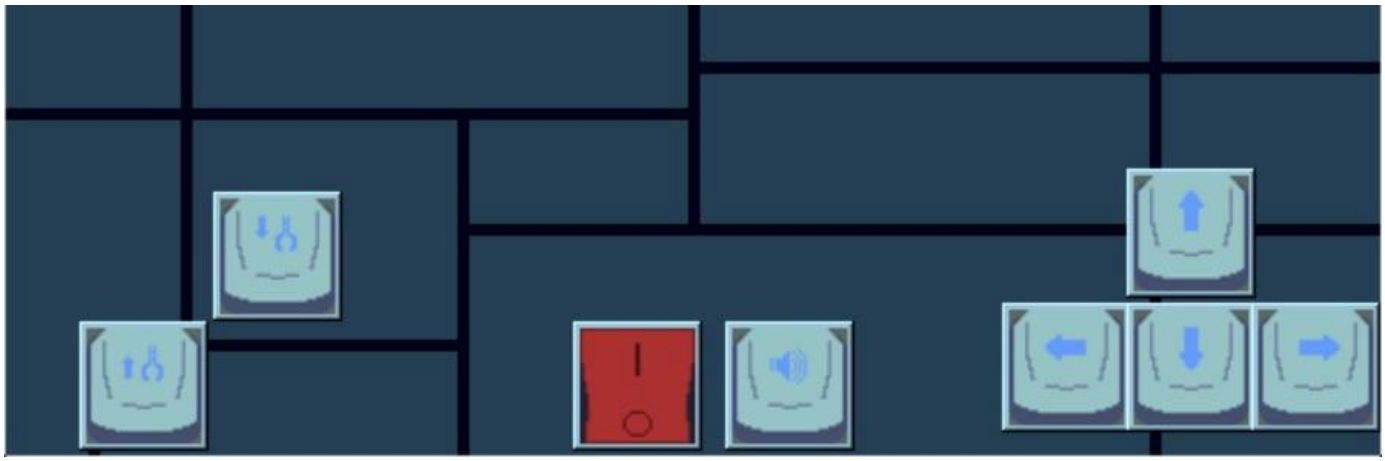
2. Encargado de la conexión remota del cliente con el robot el cual permanece en espera de alguna conexión entrante.

3. Mediante una PC la interfaz se conectará al servidor del robot y el usuario podrá controlarlo.

4. El robot ejecuta la acción recibida por parte del usuario y procede a realizarla de acuerdo con las instrucciones recibidas.

5. Interfaz gráfica que el usuario usará para controlar al robot.

5.3 Interfaz de usuario



6. Implementación

6.1. Fórmulas utilizadas:

Cálculo del Torque Necesario para Levantar la Pelota de Ping Pong

Para calcular el torque necesario que debe ejercer el motor del robot, se considera que la garra tiene una **longitud de 80 mm** (L = 0,08 m) y que la fuerza gravitacional de la pelota actúa al final de la garra.

---

1. Datos iniciales

- Masa de la pelota: m = 2,7 g = 0,0027 kg
- Aceleración gravitacional: g = 9,81 m/s²
- Longitud de la garra: L = 80 mm = 0,08 m
- Centro de masa de la garra: ubicado en la mitad de la longitud de la garra (L/2 = 0,04 m)
- Masa de la garra: m_garra = 10 g = 0,01 kg

---

2. Fuerza gravitacional de la pelota y la garra

La fuerza gravitacional se calcula con la fórmula:

F = m * g

- Fuerza de la pelota:
  F_pelota = 0,0027 kg * 9,81 m/s² = 0,0265 N
- Fuerza de la garra:
  F_garra = 0,01 kg * 9,81 m/s² = 0,0981 N

---

3. Cálculo del Torque

El torque se calcula con la fórmula:

T = F * d

Donde:

- F es la fuerza aplicada.
- d es la distancia desde el eje de rotación hasta el punto donde actúa la fuerza.

- Torque debido a la pelota (fuerza aplicada al final de la garra):
  T_pelota = F_pelota * L
  T_pelota = 0,0265 N * 0,08 m = 0,0021 Nm

- Torque debido a la garra (considerando su centro de masa a la mitad de la longitud):
  T_garra = F_garra * (L / 2)
  T_garra = 0,0981 N * 0,04 m = 0,0039 Nm

---

4. Torque Total

El torque total requerido es la suma de ambos torques:

T_total = T_pelota + T_garra
T_total = 0,0021 Nm + 0,0039 Nm = 0,0060 Nm

---

5. Ajuste por Eficiencia

Considerando un **20% adicional** para compensar posibles pérdidas por fricción e ineficiencia mecánica:

T_ajustado = T_total * 1,2
T_ajustado = 0,0060 Nm * 1,2 = 0,0072 Nm

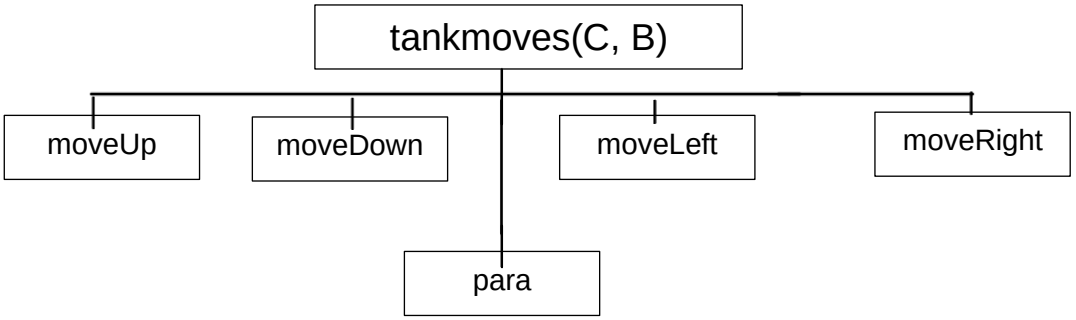
Resultado Final

El **torque ajustado** que el motor del robot debe proporcionar para levantar la pelota, considerando el peso de la garra y las pérdidas de eficiencia, es aproximadamente:

0,0072 Nm

Este es el valor mínimo de torque requerido para garantizar el funcionamiento adecuado del sistema.

6.2. Diagramas:



6.2. Server:

Server.py > ...

```python
#!/usr/bin/env python3
import socket
from librari import *

s = socket.socket()
print("Socket creado")
port = 8080
s.bind(('', port))
print("El socket se creo con puerto:{}".format(port))
s.listen(5)
print("EL socket is listening....")
connect, addr = s.accept()
print("Se conecto a {}".format(addr))
conectado()

while True:
    rawByte = connect.recv(1)
    char = rawByte.decode('utf-8')
    if (char == 'w'):
        moveUp()
    if (char == 's'):
        moveDown()

    if (char == 'd'):
        moveRight()

    if (char == 'a'):
        moveLeft()

    if (char == 'g'):
        print("Terminando la sesion....")
        break

    if (char == 'p'):
        para()

    if(char == 'q'):
        levantarGarra()

    if(char == 'r'):
        bajarGarra()

    if(char == 'b'):
        bocina()
apagado()
```

6.3. Librerías:

```python
#!/usr/bin/env ev3dev2
from ev3dev2.motor import LargeMotor,MediumMotor, OUTPUT_C, OUTPUT_B,SpeedPercent, MoveTank,OUTPUT_A
from ev3dev2.sound import Sound
from time import sleep

ma = LargeMotor(OUTPUT_B)
mb = LargeMotor(OUTPUT_C)
mg = MediumMotor(OUTPUT_A)
tankmoves = MoveTank(OUTPUT_C,OUTPUT_B)
sound = Sound()


Codeium: Refactor | Explain | Generate Docstring | ×
def moveUp():
    print("Moving up...")
    tankmoves.on(50,50)
Codeium: Refactor | Explain | Generate Docstring | ×
def moveDown():
    print("Moving down...")
    tankmoves.on(-50,-50)
Codeium: Refactor | Explain | Generate Docstring | ×
def moveRight():
    print("Moving right...")
    tankmoves.on(-5,50)
Codeium: Refactor | Explain | Generate Docstring | ×
def moveLeft():
    print("Moving left...")
    tankmoves.on(50,-5)
Codeium: Refactor | Explain | Generate Docstring | ×
def levantarGarra():
    print("Moving garra...")
    mg.on_for_seconds(5,1)
Codeium: Refactor | Explain | Generate Docstring | ×
def bajarGarra():
    print("Moving garra...")
    mg.on_for_seconds(-5,1)
Codeium: Refactor | Explain | Generate Docstring | ×
def bocina():
    print("Bocina...")
    sound.beep()
Codeium: Refactor | Explain | Generate Docstring | ×
def para():
    print("Stopping...")
    tankmoves.on(0,0)
Codeium: Refactor | Explain | Generate Docstring | ×
def conectado():
    print("Conectado")
    sound.speak("on")
    sound.beep()
Codeium: Refactor | Explain | Generate Docstring | ×
def apagado():
    print("Apagado")
    sound.speak("off")
    sound.beep()
```

6.4. Interfaz de usuario:

```
def send_command(command):
    try:
        client_socket.send(command.encode())
        print(f"Comando enviado: {command}")
    except Exception as e:
        print(f"Error al enviar comando: {e}")
Codeium: Refactor | Explain | Generate Docstring | ×
def coneccion():
                global g
                if g== False:
                  try:
                      print("paso")
                      client_socket.connect(('192.168.71.22', 8080))
                      print("Conectado al servidor.")
                      g = True
                      print("Encender")
                      btn_g.config(image=img_On)
                  except Exception as e:
                      print("fallo es posible que su servidor no este activo")
                      print(f"Error al conectar: {e}")
                      g = False

                else:
                  print("True")
                  send_command("g")
                  g = False
                  print("apagar")
                  client_socket.close()
                  btn_g.config(image=img_Off)

Codeium: Refactor | Explain | Generate Docstring | ×
def on_key_press(event):
    global client_socket
    global g
    key = event.char
    if key in keys_pressed :
        keys_pressed[key] = True  # Actualizar el estado a presionado
        # Manejo de botones visuales...
        print(key)
        print("key es true")
        if key == "w":
            btn_w.config(image=img_up_pressed)
        elif key == "a":
            btn_a.config(image=img_left_pressed)
        elif key == "s":
            btn_s.config(image=img_down_pressed)
        elif key == "d":
            btn_d.config(image=img_right_pressed)
        elif key == "q":
            btn_q.config(image=img_Up_garra_pressed)
        elif key == "r":
            btn_r.config(image=img_down_garra_pressed)
        elif key == "b":
            btn_b.config(image=img_bocina_pressed)

```

```python
def on_key_release(event):

    key = event.char
    if key in keys_pressed :
        keys_pressed[key] = False  # Actualizar el estado a no presionado
        # Manejo de botones visuales...
        send_command('p')
        if key == "w":
            btn_w.config(image=img_up)
        elif key == "a":
            btn_a.config(image=img_left)
        elif key == "s":
            btn_s.config(image=img_down)
        elif key == "d":
            btn_d.config(image=img_right)
        elif key == "q":
            btn_q.config(image=img_Up_garra)
            send_command('q')
        elif key == "r":
            btn_r.config(image=img_down_garra)
            send_command('r')
        elif key == "b":
            btn_b.config(image=img_bocina)
            send_command('r')



Codeium: Refactor | Explain | Generate Docstring | ×
def send_command_continuously():
    for key, pressed in keys_pressed.items():
        if pressed and key not in keys_pressed_once:
            send_command(key)  # Enviar comando al EV3
            print(f"Enviando comando: {key}")

    root.after(15, send_command_continuously)  # Llama esta función cada 15 ms

```

7. Resultados

7.1. Estado actual del proyecto

Hasta este momento, el proyecto en base al robot Lego Mindstorms Ev3 ha tenido los siguientes resultados:

- Ya se ha generado un diseño satisfactorio para el robot y su garra.
- El robot es totalmente capaz de moverse a tiempo real con el teclado a través de un servidor y librerías creados en Python.
- Actualización de la Wiki hasta la actualidad del proyecto.
- Se ha creado una interfaz gráfica con diseños propios.
- Se generaron bitácoras durante la duración de todo el proyecto.

- Se logró que el robot elevara la pelota, tiene algunos problemas para soltarla.

7.2 Problemas encontrados

Problema	Solución
El robot no se movía de manera fluida y a tiempo real con el teclado.	Los encargados de la programación del robot investigaron y buscaron la lógica necesaria para que el robot cumpliera con el movimiento requerido.
La plataforma del robot no podía cumplir con el requerimiento de la problemática.	El encargado de la construcción del robot se encargó de diseñar una garra que sea capaz de cumplir con la problemática.
La Wiki no avanzaba en el tiempo establecido.	Los documentadores repartieron su trabajo para dar suficiente prioridad a la wiki.
La garra no fue capaz de manipular correctamente la pelota de ping pong.	Buscar nuevos diseños para el robot o un mecanismo distinto.

8. Conclusión

A lo largo del desarrollo del proyecto “Wheatley Robotics”, se logró cumplir con gran parte de los objetivos planteados inicialmente, destacando la construcción y programación de un robot basado en la plataforma Lego Mindstorms EV3, capaz de desplazarse en tiempo real mediante una interfaz gráfica intuitiva creada en Python con la librería Tkinter.

El trabajo en equipo permitió superar desafíos técnicos importantes, como la implementación de la conectividad del robot y la optimización de sus movimientos en las cuatro direcciones. Sin embargo, no se logró implementar de manera satisfactoria el mecanismo de la garra para levantar y dejar la pelota de ping pong en el suelo, debido a dificultades en el diseño mecánico. Esta situación resalta la necesidad de seguir explorando y optimizando soluciones para la manipulación precisa de objetos livianos.

A pesar de este desafío, el proyecto fue exitoso en demostrar la capacidad del equipo para integrar soluciones de software y hardware, así como en la creación de un servidor funcional y una interfaz de usuario amigable. Además, el proceso permitió desarrollar habilidades clave en áreas como programación, diseño de sistemas y trabajo colaborativo.

En conclusión, el proyecto “Wheatley Robotics” alcanzó resultados significativos en términos de movilidad y control del robot, aunque deja como desafío pendiente la mejora del mecanismo de la garra. Las lecciones aprendidas durante el desarrollo servirán como base para futuras implementaciones, reforzando la importancia de la iteración, el análisis de problemas y la búsqueda continua de soluciones innovadoras.

9. Referencias

Documentación de la librería Pybricks:
https://docs.pybricks.com/en/stable/

Página de Compra de Lego Mindstorm EV3
“Lego Mindstorm EV3” mercadolibre.cl Disponible: https://articulo.mercadolibre.cl

Página de Compra de tarjeta MicroSd
“tarjeta Microsd 8gb” mercadolibre.cl Disponible: https://articulo.mercadolibre.cl