



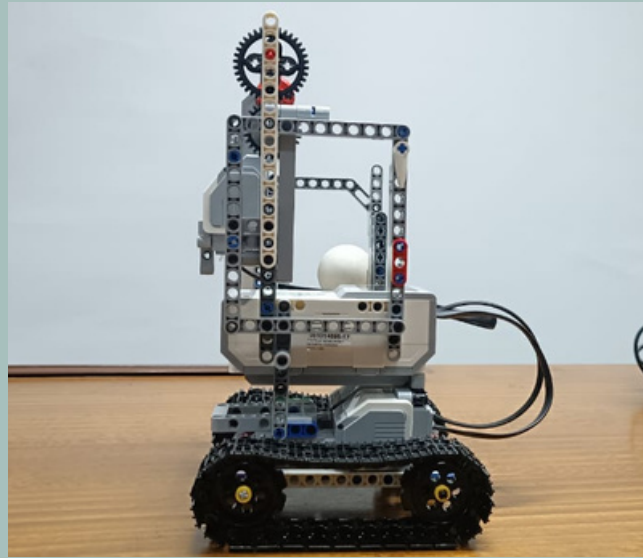
UNIVERSIDAD DE TARAPACÁ
Universidad del Estado

Ingeniería@
Computación e Informática

KIKI'S PROYECT

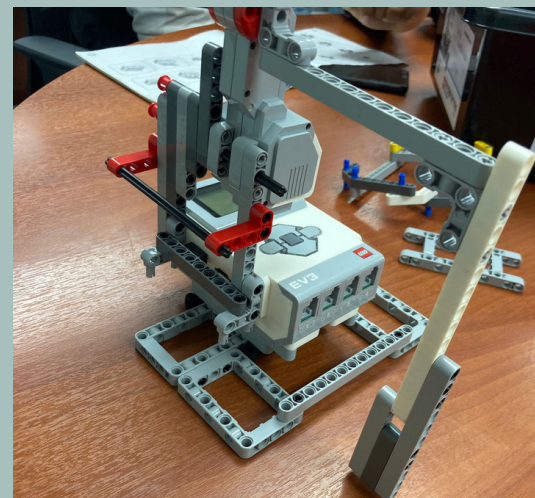
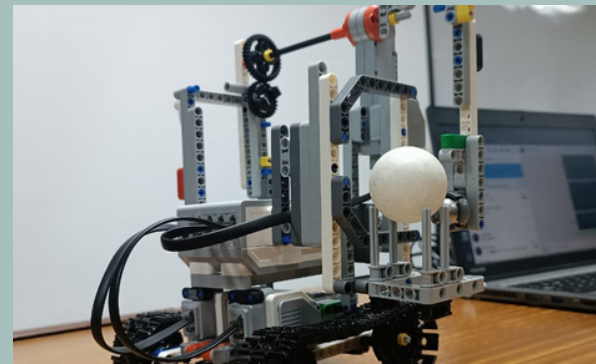
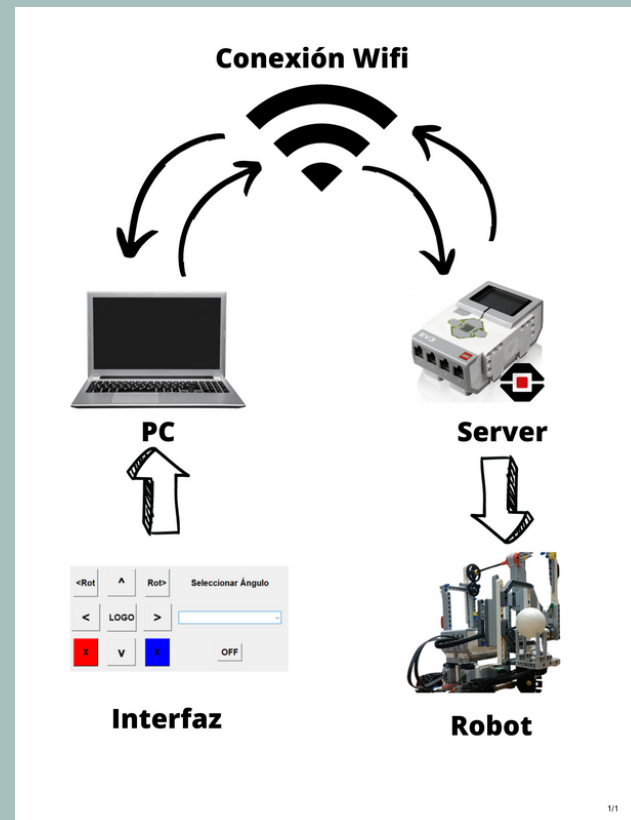
*Integrantes: Juan Casilla.
Fabian Diaz.
Jordan Nina.
Andrea Navia.*





```
# Importar modulos necesarios
import evdev
import ev3dev.auto as ev3
import time
import socket
from ev3dev.ev3 import *
from ev3dev2.motor import Large
from ev3dev2.motor import Large

motorA=LargeMotor("outA")
motorD=LargeMotor("outD")
motorB=LargeMotor("outB")
```

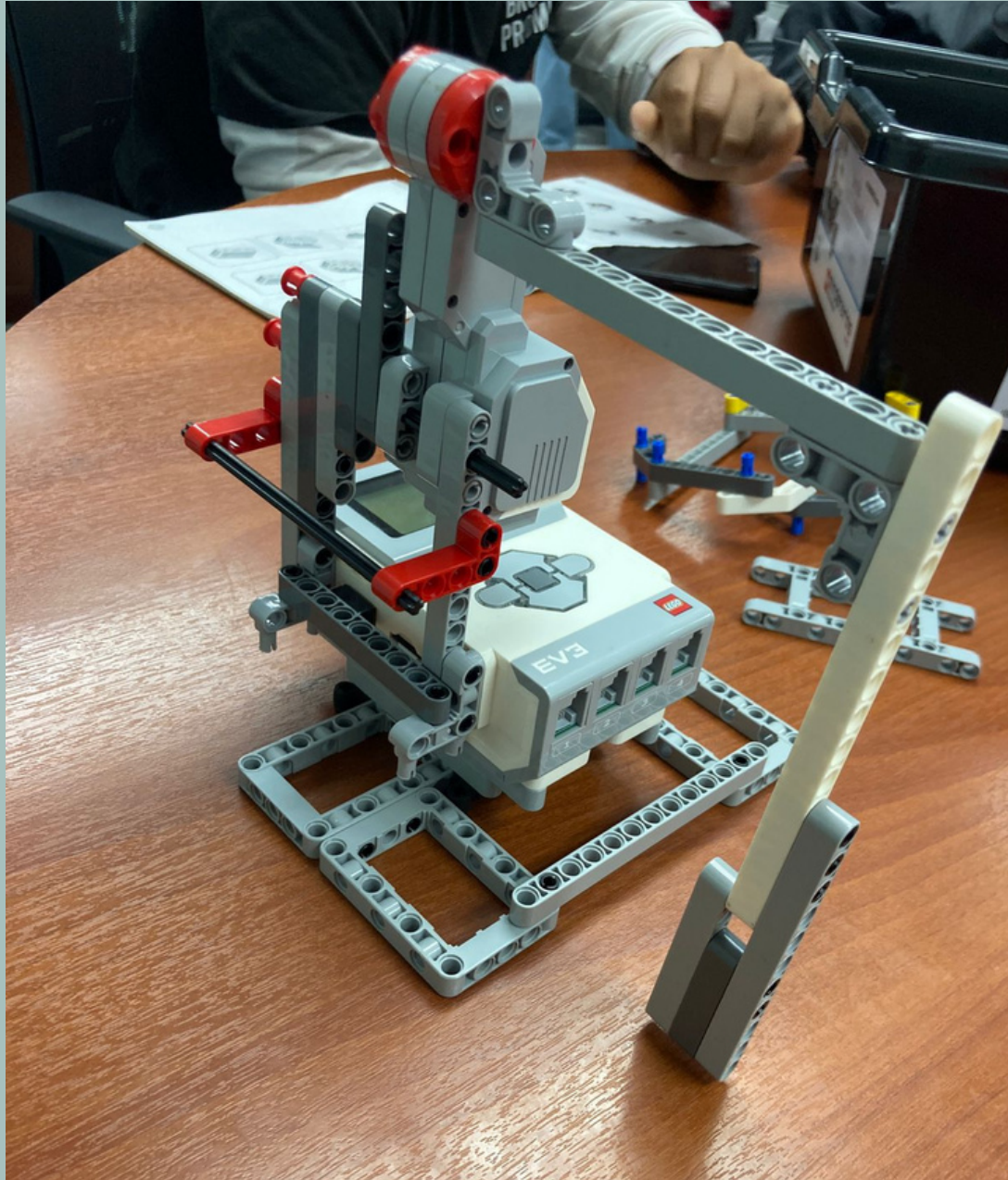


Contenido

01. Evolución.
08. Requerimientos.
09. Arquitectura.
10. Interfaz.
11. Fundamentos del proyectil.
13. Descripción del programa.
19. Estado actual.
20. Problemas y soluciones.

EVOLUCIÓN

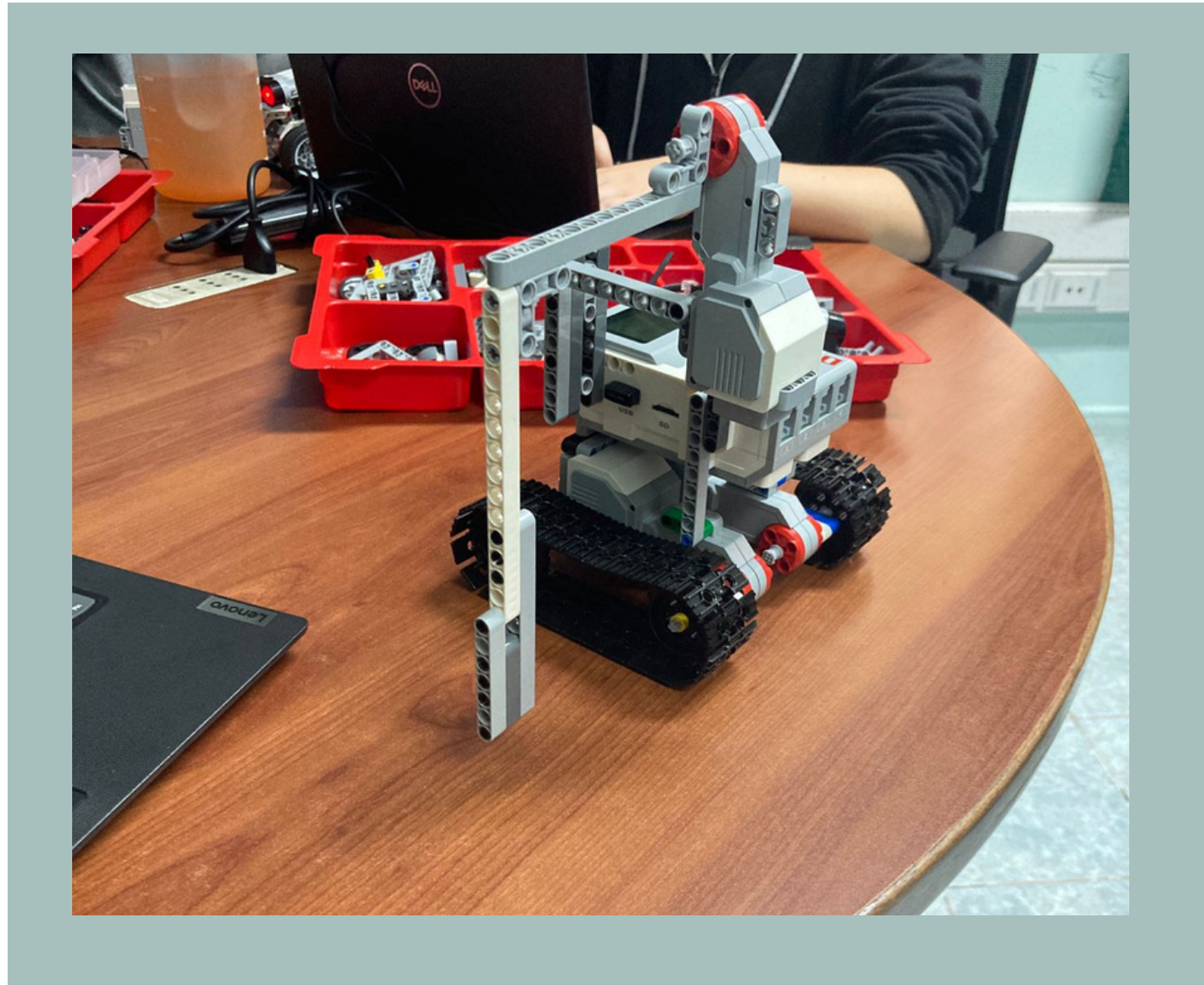
K.1



- No puede desplazarse.
- El palo de golf era inestable.

EVOLUCIÓN

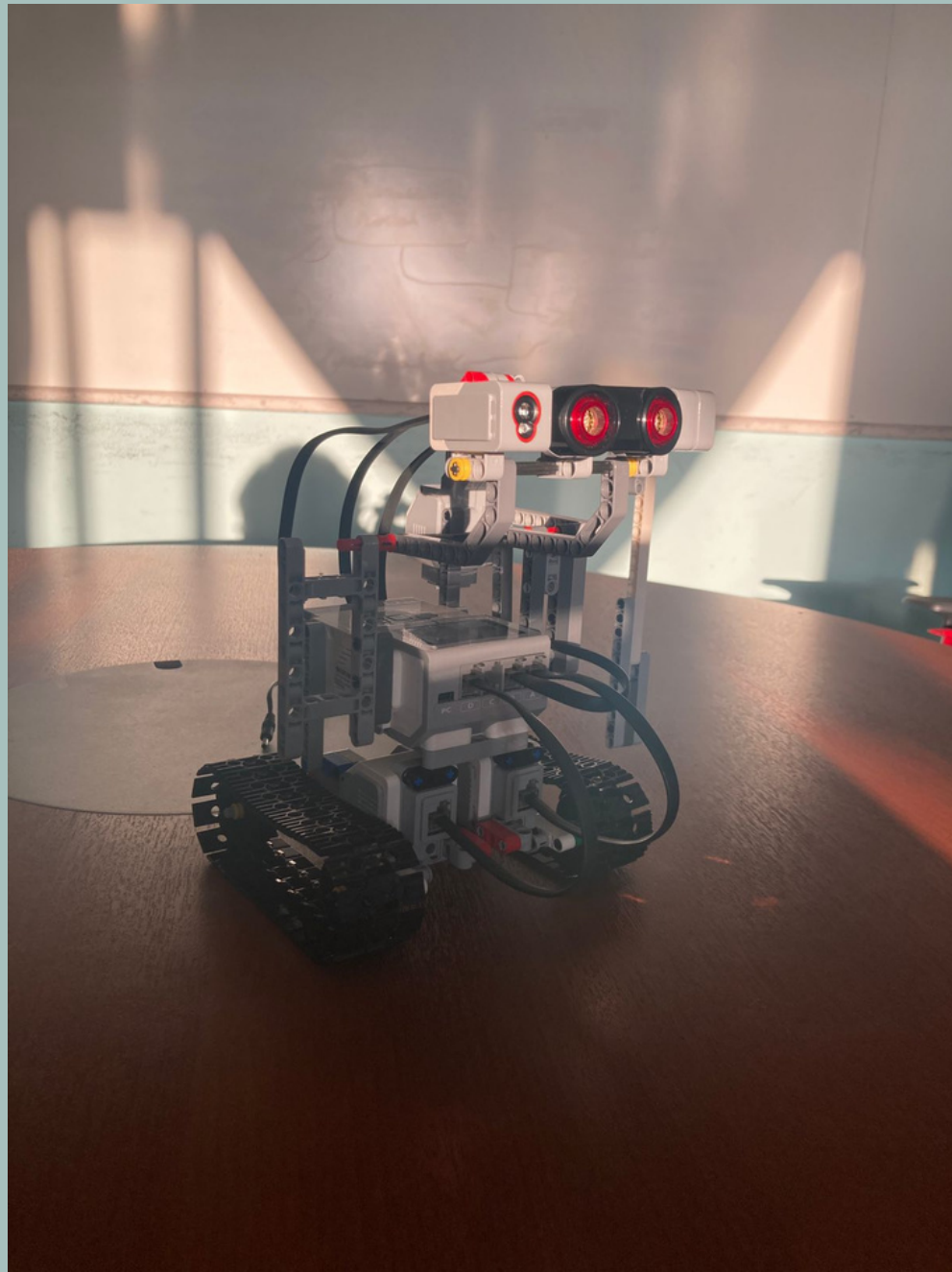
K.2



- Se le agrego movilidad.
- El palo de golf sigue siendo inestable.

EVOLUCIÓN

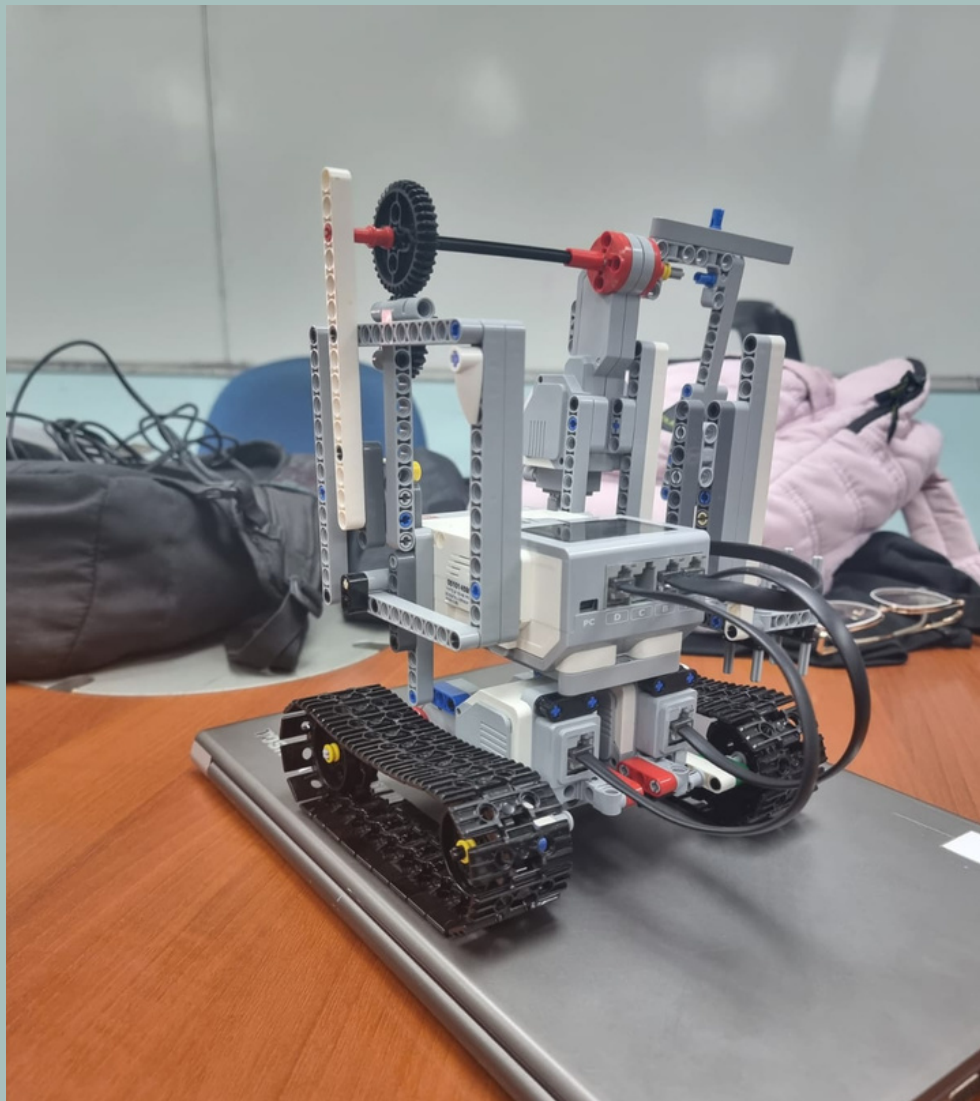
K.3



- El palo de golf sigue siendo inestable.
- Se le agrego cosas innecesarias, pero contribuian a su aspecto.

EVOLUCIÓN

K.4

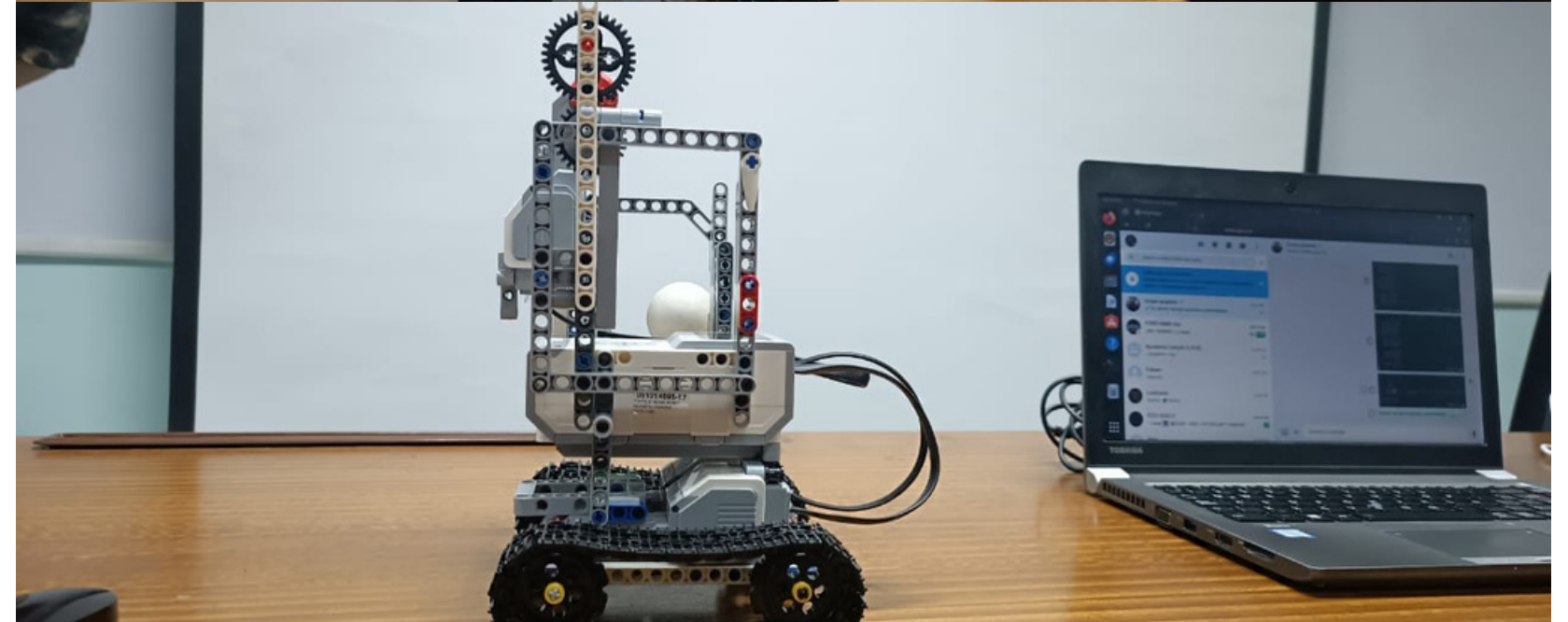
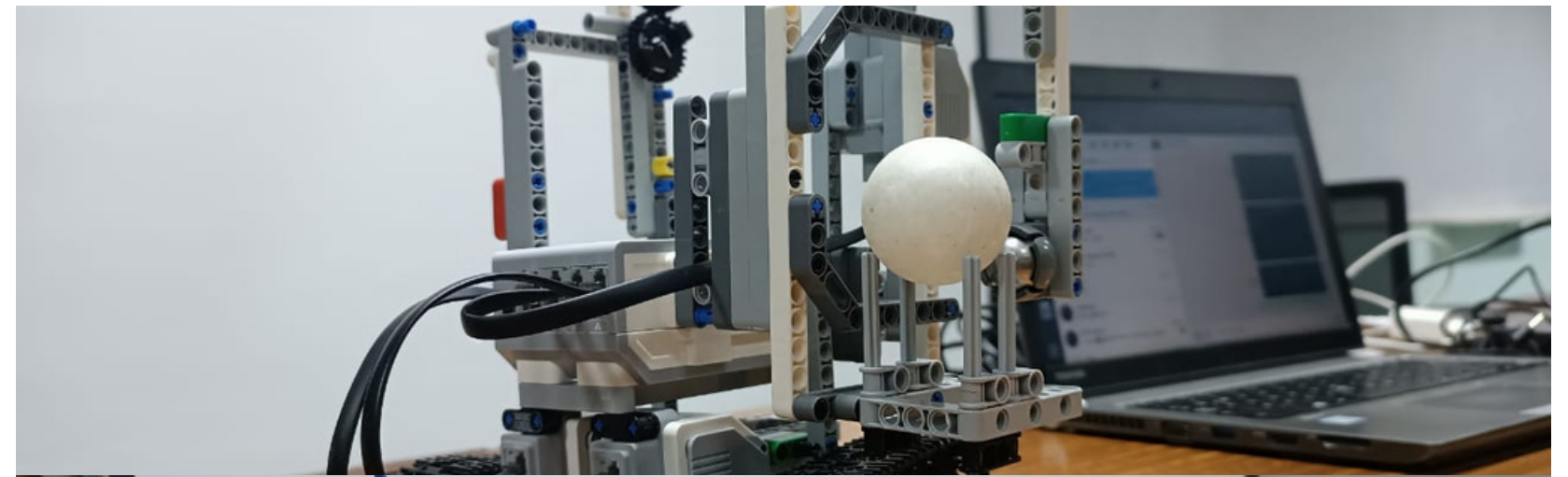


- El palo de golf es estable, se reforzó.
- Se le retiró todo lo innecesario.
- El palo de golf era demasiado pequeño.
- Se le agrego una base para la pelota.

EVOLUCIÓN FINAL

K.5

- Se alargó el palo porque estaba muy corto.
- Se reforzó la estructura que sostiene al motor unido al palo.



REQUERIMIENTOS

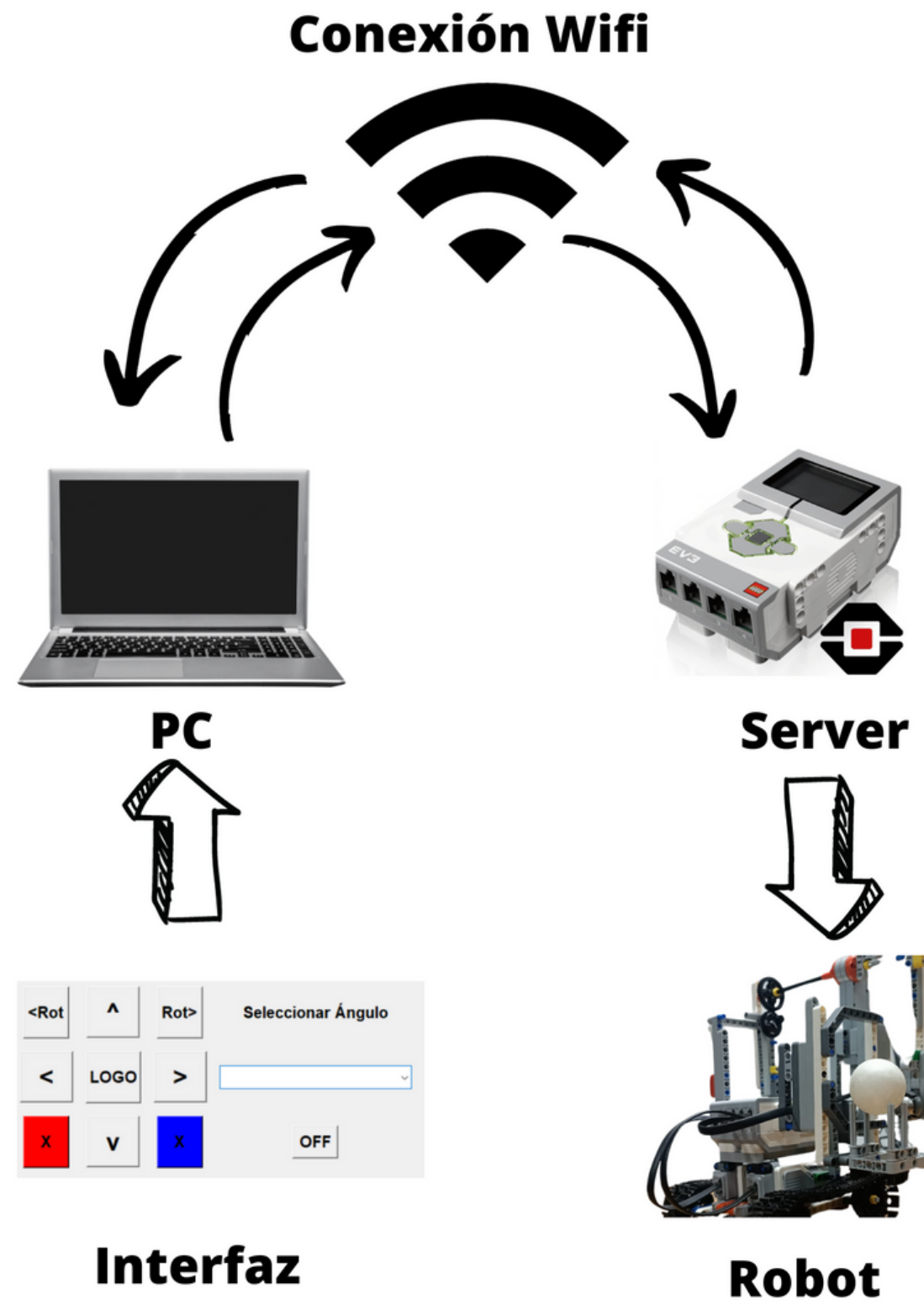
☆ FUNCIONALES

- El robot podrá hacer diferentes funciones, siendo controlado por una interfaz gráfica.
- El robot tiene la función de avanzar, retroceder, girar hacia la derecha e izquierda.
- El robot tiene la función de golpear una pelota de golf.

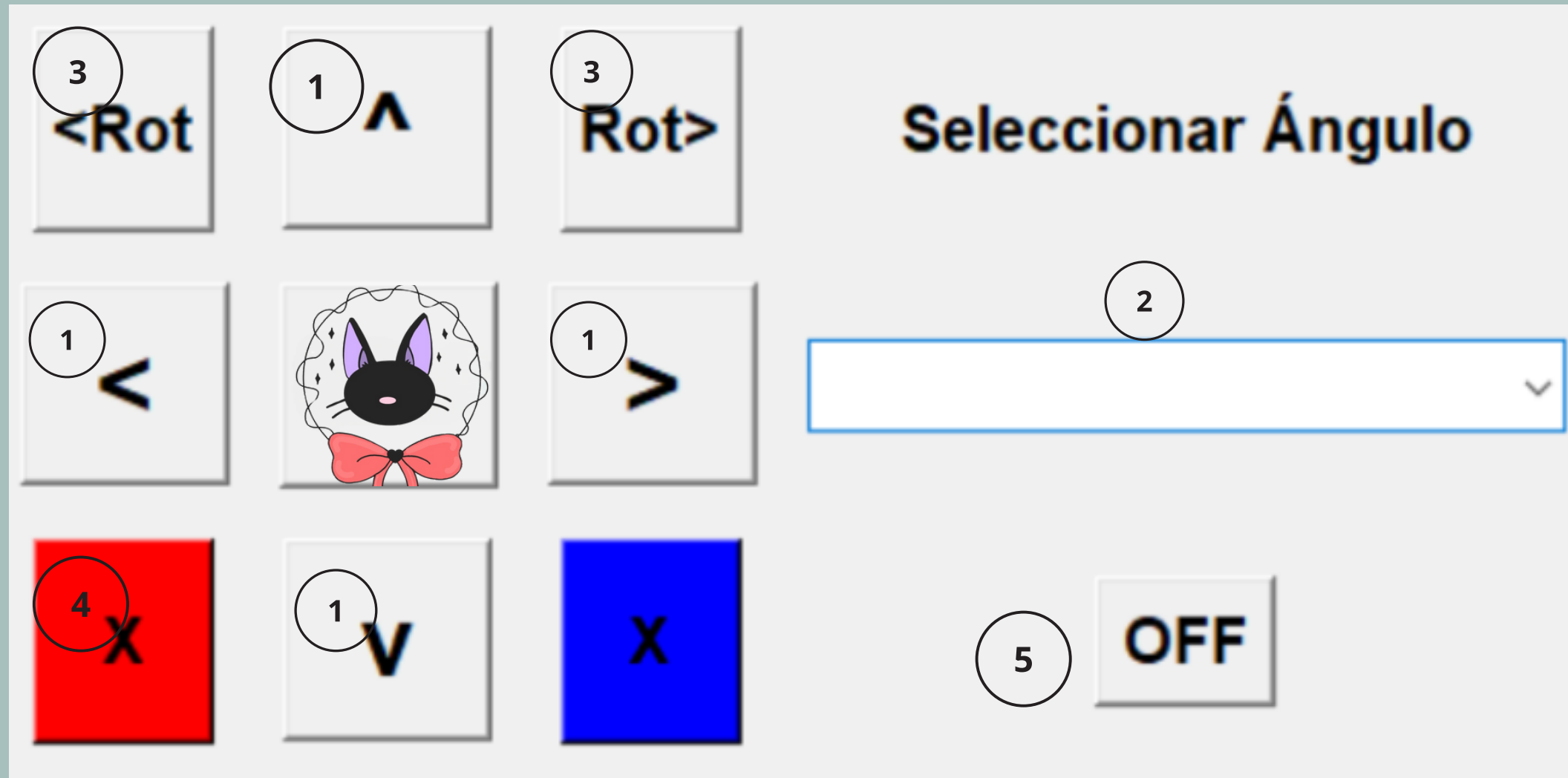
☆ NO FUNCIONALES

- Cada movimiento del robot se realizará en un tiempo y velocidad estimados.
- El diseño del robot debe ser estable y eficiente.
- Se utilizara kit Lego Mindstorm EV3 Education en la construcción.
- El lenguaje de programación para hacer este proyecto es Python.
- El robot será manejado por un sistema operativo de Linux.

ARQUITECTURA



INTERFAZ



01. Botones de movimiento del robot.

02. Ángulo.

03. Reposicionar del palo.

04. Golpe

05. Desconectar.

Fundamentos de proyectil

Calcular V_0 y t conociendo que la aceleración de gravedad es igual 9.8 m/s^2 , y los siguiente datos:

$$\theta = 60^\circ$$

$$X = 0.5 \text{ m}$$

$$X_0 = 0 \text{ m}$$

$$Y = 0 \text{ m}$$

$$Y_0 = 0.11 \text{ m}$$

$$V_0 = ?$$

$$t = ?$$

$$v_{ox} = v_o \cos(60^\circ) = v_o \cdot 1/2$$

$$v_{oy} = v_o \sin(60^\circ) = v_o \cdot \sqrt{3}/2$$

$$x = x_o + v_{ox} \cdot t \Rightarrow x = v_o/2 \cdot t \Rightarrow v_o = 2x/t$$

$$y = y_o + v_{oy} \cdot t - 4.9 \cdot t^2 \Rightarrow 0 = y_o + v_o \cdot (\sqrt{3}/2) \cdot t - 4.9 \cdot t^2$$

reemplazando v_o en la ecuación de y :

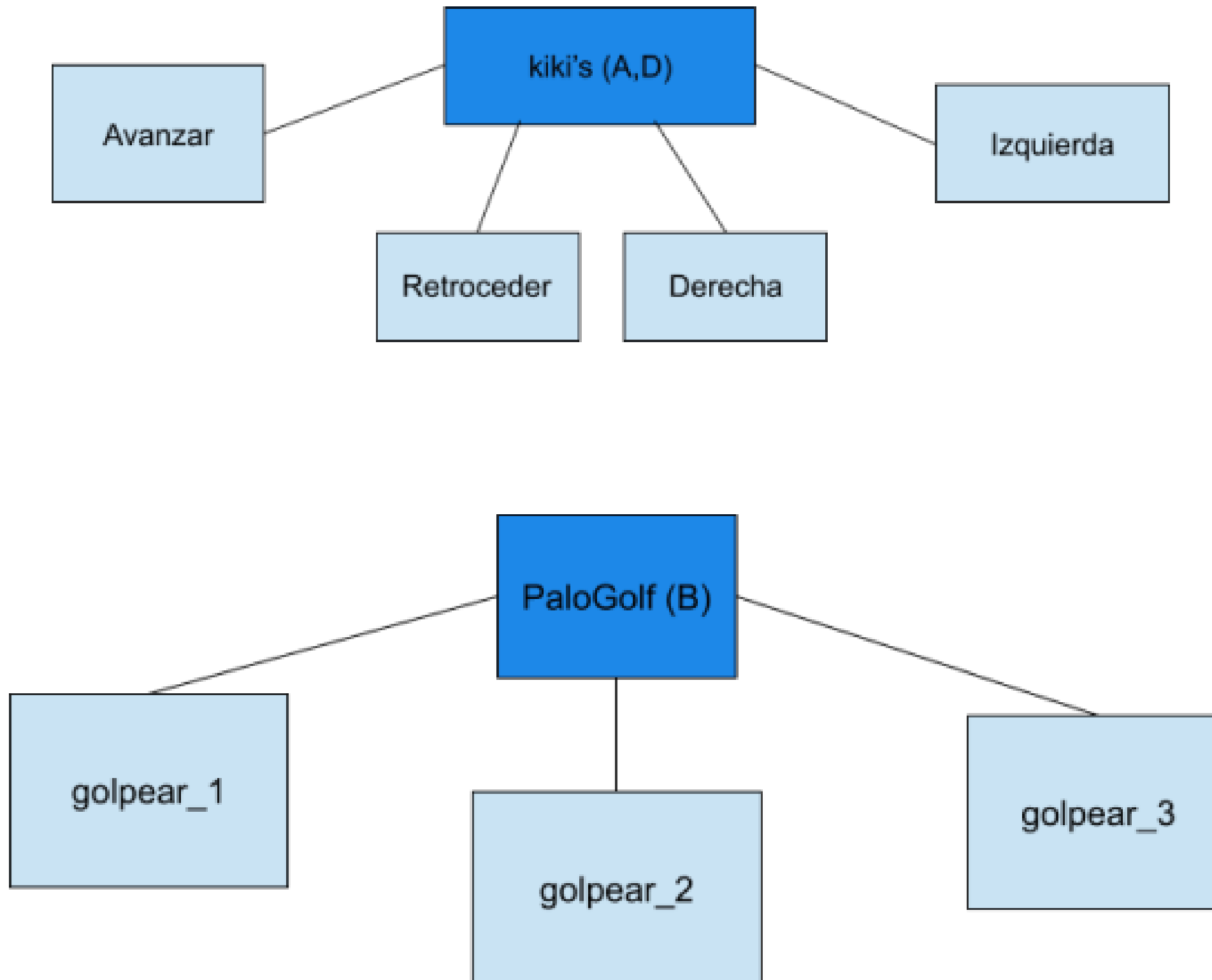
$$\Rightarrow 0 = y_o + (2x\sqrt{3} \cdot t)/(2 \cdot t) - 4.9 \cdot t^2$$

$$\Rightarrow t = \sqrt{\frac{y_o + x \cdot \sqrt{3}}{4.9}} = 0.44 \text{ s}$$

reemplazando t en v_o :

$$\Rightarrow v_o = 2x/t = 2.27 \text{ m/s}$$

Diagrama



Feature #2407: Bitácoras

Feature #2410: Bitácora 1

Feature #2462: Bitácora 2

Feature #2542: Bitácora 3

Feature #2707: Bitácora 4

Feature #2840: Bitácora 5

Feature #3181: Bitácora 6

Feature #3182: Bitácora 7

Feature #3262: Bitacora 8

Feature #3343: Bitacora 9

Feature #2710: Documentación del proyecto

Feature #2838: Fotos y videos

Feature #2951: Carta Gantt

Feature #2832: Wiki

Feature #2996: Etapa 1

Feature #2997: Formulación del proyecto

Feature #2999: Construcción del robot

Feature #2998: Estudio del sistema operativo

Feature #3001: Informe 1

Feature #3000: Estimación de costos

Feature #3002: Presentación 1

Bug #2983: Etapa 2

Feature #2985: Estudio de librería ev3dev lang

Feature #2984: Estudio de lenguaje Python

Feature #2986: Programación del robot

Feature #2987: Programar movimientos del ...

Feature #2988: Programación de interfaz

Feature #3454: Implementación de la Interfaz

Bug #2989: Informe 2

Feature #3457: Calculo de proyectiles

Feature #3458: Rediseño del robot

Feature #3459: Actualización de la Wiki

Feature #2990: Presentación 2

New 89%

New 100%

New 100%

New 100%

New 100%

New 100%

New 100%

New 100%

New 100%

New 0%

New 55%

New 80%

New 50%

New 80%

New 100%

New 100%

New 100%

New 100%

New 100%

New 100%

New 100%

New 77%

New 100%

New 90%

New 96%

New 90%

New 100%

New 100%

New 80%

New 100%

New 50%

New 0%

Carta GANTT

Descripcion de programas

Interfaz

```
1 import tkinter as tk
2 import tkinter.font as font
3 import socket
4
5 PORT = 12345
6 SERVER = '192.168.71.1'
7
8 AVANZAR = 'avanzar'
9 RETROCEDER = 'retroceder'
10 DETENER = 'detener'
11 CLOSE = 'close'
12 DERECHA='derecha'
13 IZQUIERDA='izquierda'
14 GOLPEAR='golpear'
15 ATRAS='atras'
16 ADELANTE='adelante'
17
18 if __name__ == '__main__':
19     client = socket.socket()
20     client.connect((SERVER, PORT))
21     print(client.recv(1024).decode())
22     def avanzar():
23         client.send(AVANZAR.encode())
24         print("Avanzando")
25     def retroceder():
26         client.send(RETROCEDER.encode())
27         print("Retrocediendo")
```

Interfaz

```
28     def girar_der():
29         client.send(DERECHA.encode())
30         print("Girando derecha")
31     def girar_izq():
32         client.send(IZQUIERDA.encode())
33         print("Girando izquierda")
34     def rotar_brazo_Atras():
35         client.send(ATRAS.encode())
36         print("Rotando brazo hacia la derecha")
37     def rotar_brazo_Adelante():
38         client.send(ADELANTE.encode())
39         print("rotando brazo hacia la izquierda")
40     def parar_robot():
41         client.send(DETENER.encode())
42         print("Se detuvo el robot")
43     def golpear():
44         client.send(GOLPEAR.encode())
45         print("golpeando...")
```


Interfaz

```
46 ventana = tk.Tk()
47 ventana.title("Control de Robot con Palo de Golf")
48 botonFont = font.Font(family='Helvetica', size=24, weight='bold')
49 botonFontSec = font.Font(family='Helvetica', size=15, weight='bold')
50
51 boton_avanzar = tk.Button(ventana, text="^", command=avanzar, height=1, width=3, font=botonFont)
52 boton_retroceder = tk.Button(ventana, text="v", command=retroceder, height=1, width=3, font=botonFont)
53 boton_izquierda = tk.Button(ventana, text="<", command=girar_izq, height=1, width=3, font=botonFont)
54 boton_derecha = tk.Button(ventana, text=">", command=girar_der, height=1, width=3, font=botonFont)
55
56 boton_rotar_atras = tk.Button(ventana, text="Rot>", command=rotar_brazo_Atras, height=2, width=4, font=botonFontSec)
57 boton_rotar_adelante = tk.Button(ventana, text="<Rot", command=rotar_brazo_Adelante, height=2, width=4, font=botonFontSec)
58
59 boton_stop = tk.Button(ventana, text="X", command=golpear, height=2, width=4, font=botonFontSec, bg='red')
60 boton_apagar = tk.Button(ventana, text="OFF", command=parar_robot, height=2, width=6, font=botonFontSec)
61 logo = tk.Button(ventana, text="LOGO", height=2, width=5, font=botonFontSec)
62
63 boton_rotar_adelante.grid(column=0,row=0, padx=8, pady=8)
64 boton_avanzar.grid(column=1,row=0, padx=8, pady=8)
65 boton_rotar_atras.grid(column=2,row=0, padx=8, pady=8)
66
67 boton_izquierda.grid(column=0,row=1, padx=8, pady=8)
68 logo.grid(column=1,row=1, padx=8, pady=8)
69 boton_derecha.grid(column=2,row=1, padx=8, pady=8)
70
71 boton_stop.grid(column=0,row=2, padx=8, pady=8)
72 boton_retroceder.grid(column=1,row=2, padx=8, pady=8)
73 boton_apagar.grid(column=2,row=2, padx=8, pady=8)
74
75 ventana.mainloop()
76 client.close()
```

Servidor

```
4 import evdev
5 import ev3dev.auto as ev3
6 import time
7 import socket
8 from ev3dev.ev3 import *
9 from ev3dev2.motor import LargeMotor, OUTPUT_A, OUTPUT_B, SpeedPercent, MoveTank
10 from ev3dev2.motor import LargeMotor
11
12 motorA=LargeMotor("outA")
13 motorD=LargeMotor("outD")
14 motorB=LargeMotor("outB")
15 ## Funciones
16 def avanzar():
17     motorA.on(-50)
18     motorD.on(-50)
19     time.sleep(1)
20     motorA.stop()
21     motorD.stop()
22 def golpear_1():
23     motorB.on(-10)
24     time.sleep(0.5)
25     motorB.stop()
26 def golpear_2():
27     motorB.on(-70)
28     time.sleep(0.1)
29     motorB.stop()
```

Servidor

```
26 def golpear_2():
27     motorB.on(-70)
28     time.sleep(0.1)
29     motorB.stop()
30 def golpear_3():
31     motorB.on(10)
32     time.sleep(0.5)
33     motorB.stop()
34 def retroceder():
35     motorD.on(50, False)
36     motorA.on(50, False)
37     time.sleep(1)
38     motorA.stop()
39     motorD.stop()
40 def derecha():
41     motorA.on(-20)
42     motorD.on(20)
43     time.sleep(0.5)
44     motorA.stop()
45     motorD.stop()
46 def izquierda():
47     motorD.on(-20)
48     motorA.on(20)
49     time.sleep(0.5)
50     motorD.stop()
51     motorA.stop()
```

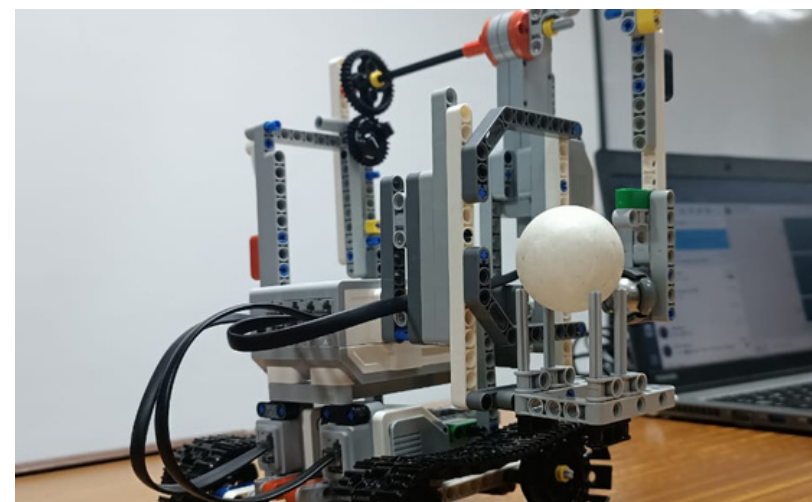
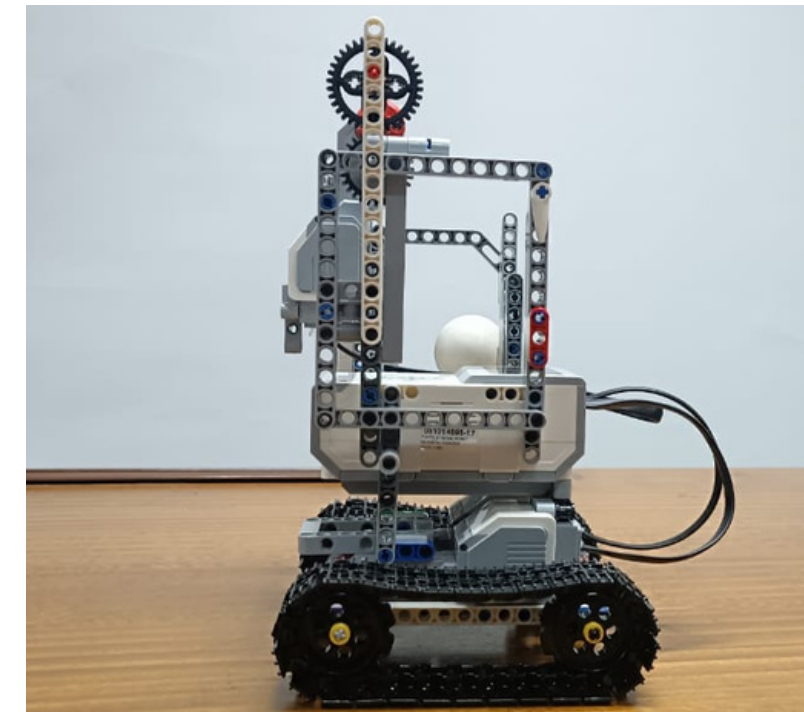
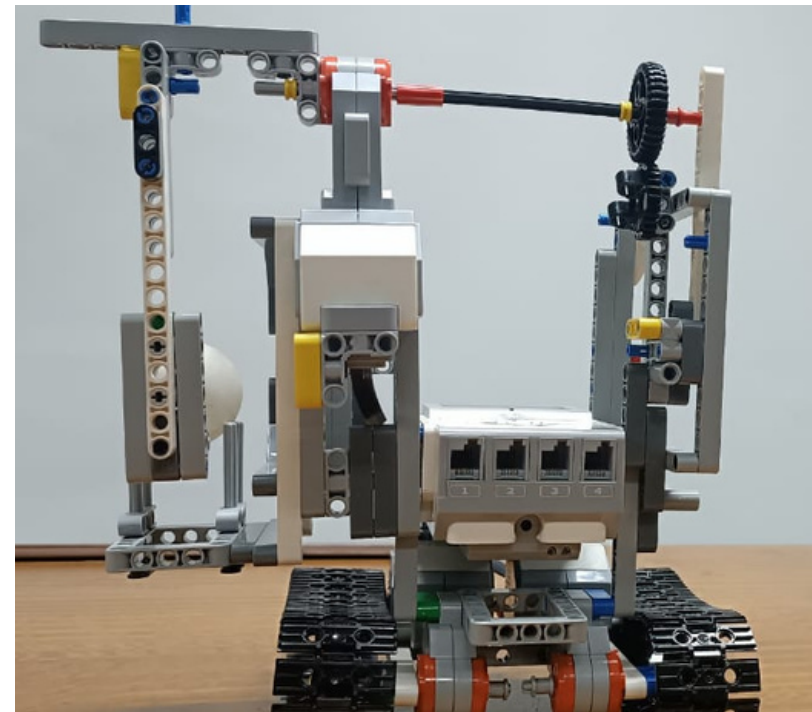
Servidor

```
52 host = "192.168.71.1"
53 port = 12345
54
55 server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
56 server_socket.bind((host, port))
57
58 server_socket.listen(1)
59 print("Listening at port 12345")
60
61 client_socket, addr = server_socket.accept()
62 print("Connection from: " + str(addr))
63 client_socket.send("Bienvenido".encode())
64 while True:
```

```
65     data = client_socket.recv(1024).decode()
66     if data=="avanzar":
67         avanzar()
68     elif data == "golpear":
69         golpear_2()
70     elif data=="atras":
71         golpear_3()
72     elif data=="adelante":
73         golpear_1()
74     elif data=="retroceder":
75         retroceder()
76     elif data=="derecha":
77         derecha()
78     elif data=="izquierda":
79         izquierda()
80     elif data=="sensor":
81         sensor()
```


Estado Actual

- Ensamblaje básico. completamente implementado.
- Programación avanzada.
- Conexión al robot.
- Carta Gantt.
- Redmine wiki
- Bitácoras semanales



Problemas y soluciones

Problemas encontrados	Soluciones
Fallas en la comunicación con el robot debido a una conexión a internet débil en ciertas circunstancias	Para probar el robot, se recomienda hacerlo en momentos en los que no haya otros grupos conectados a la misma red inalámbrica. Si esto no es posible, se debe generar una conexión con un dispositivo móvil.
Inestabilidad a la hora de golpear la pelota debido al diseño mal balanceado el robot se balancea hacia un lado en específico	Rediseño de la estructura del robot ,agregando un contra peso en el lado opuesto

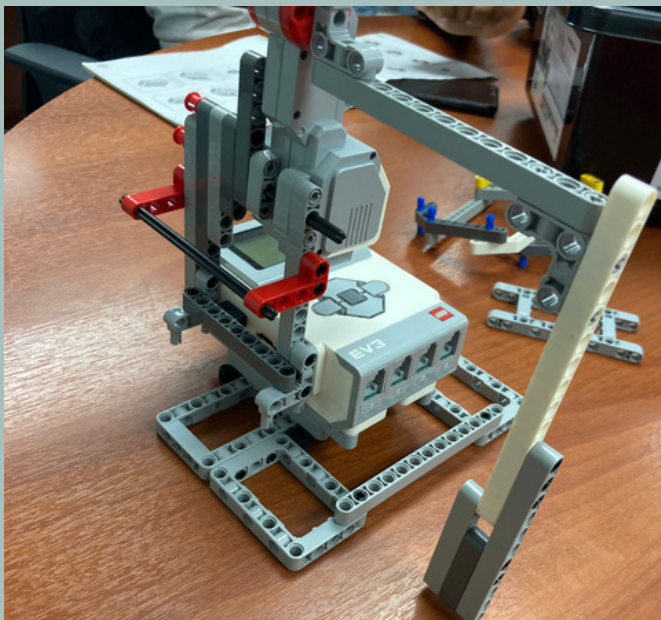
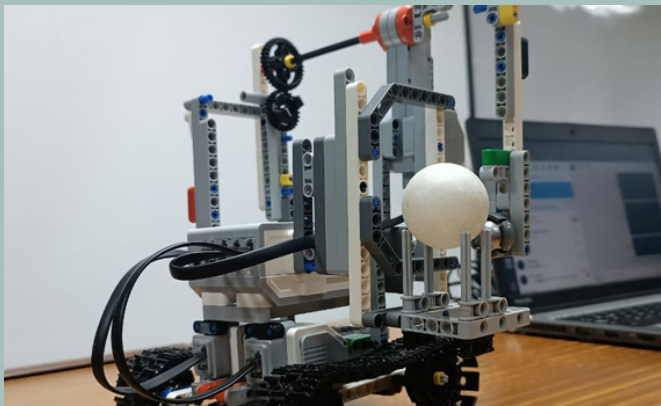
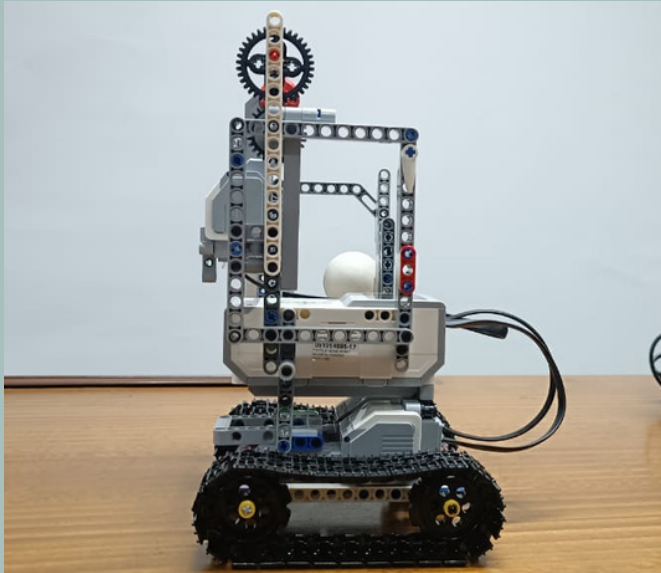
Problemas y soluciones

Problemas encontrados	Soluciones
<p>Fallas a la hora de golpear , la estructura para golpear una pelota no se posiciona de manera correcta provocando tiros en posiciones aleatorias</p>	<p>Crear funciones en el programa que nos permita dirigir el palo a voluntad.</p>
<p>Fallas del robot a la hora de girar hacia los lados, provocando saltos innecesarios a la hora de girar, pudiendo provocar el desgaste del motor</p>	<p>Usar 2 motores, uno que avanza hacia adelante y otro hacia atrás, para que este pueda girar sin complicaciones.</p>

Conclusión

Durante el desarrollo de este proyecto se pudieron apreciar dificultades, principalmente en la arquitectura del robot, debido a que antes de llegar a construir el modelo que se tiene actualmente se armó un modelo distinto estéticamente. Actualmente se cuenta con un diseño estable, en el cual el usuario puede controlar sus movimientos a través de una interfaz.

Por otro lado, debido a los inconvenientes que se han presentado, como la salida de uno de los integrantes de equipo, se han reasignado las tareas a cada uno de los integrantes para poder sobrellevar la situación y reforzar la organización del equipo.



¡Muchas
gracias!