UNIVERSIDAD DE TARAPACÁ
Universidad del Estado

Ingeniería
Computación e Informática

Proyecto I

Hole In One

Integrantes:
Tihare Cabello, Melisa Huanca, Cristian Huanca, Liliana Gálvez,
Byron Santibáñez

CONTENIDO

PERSONAL

Líder

Melisa Huanca

Diseñador

Liliana Gálvez

Ensamblador

Byron Santibáñez

Programador

Cristian Huanca

Documentador

Tihare Cabello

Líder

Melisa Huanca

Diseñador

Byron Santibáñez

Ensamblador

Liliana Gálvez

Programador

Cristian Huanca

Documentador

Tihare Cabello

Líder

Melisa Huanca

Diseñador

Byron Santibáñez

Ensamblador

Liliana Gálvez

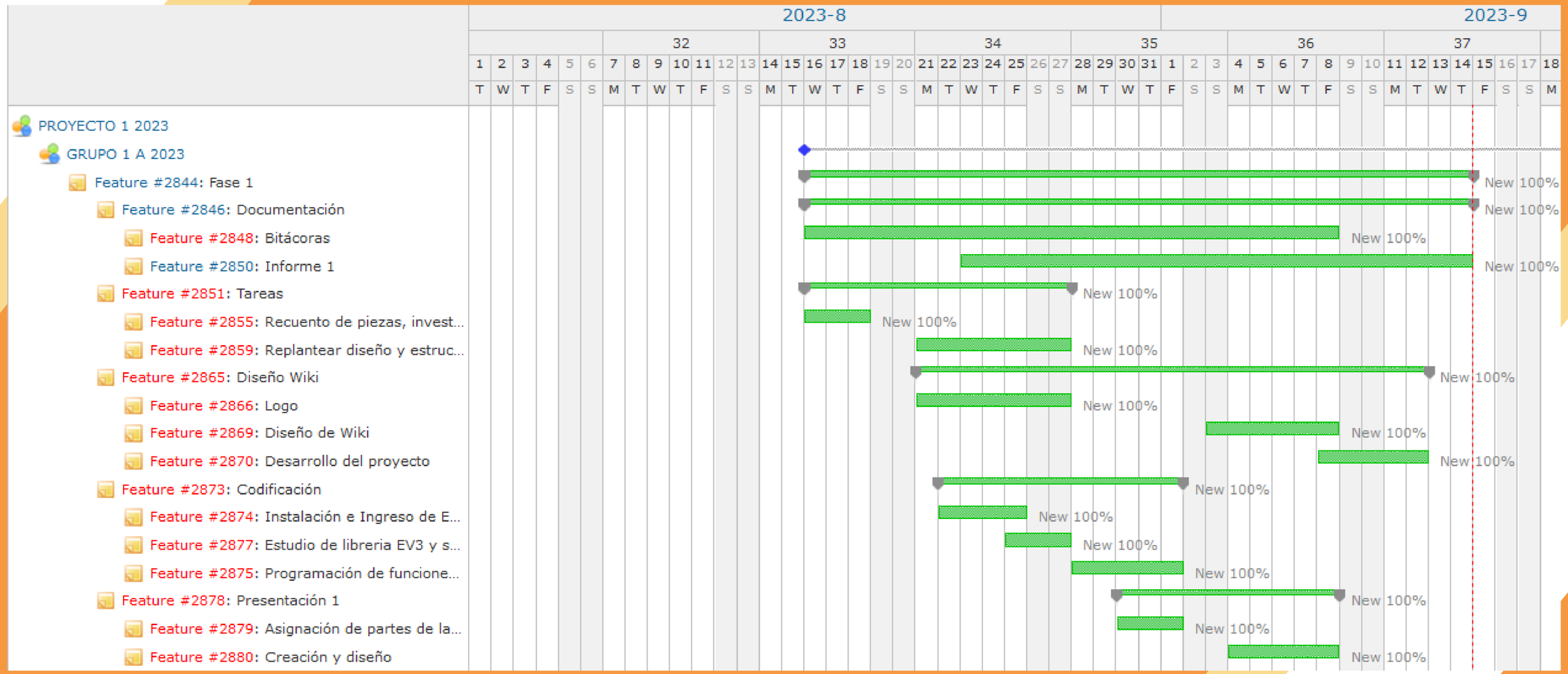
Programador

Tihare Cabello

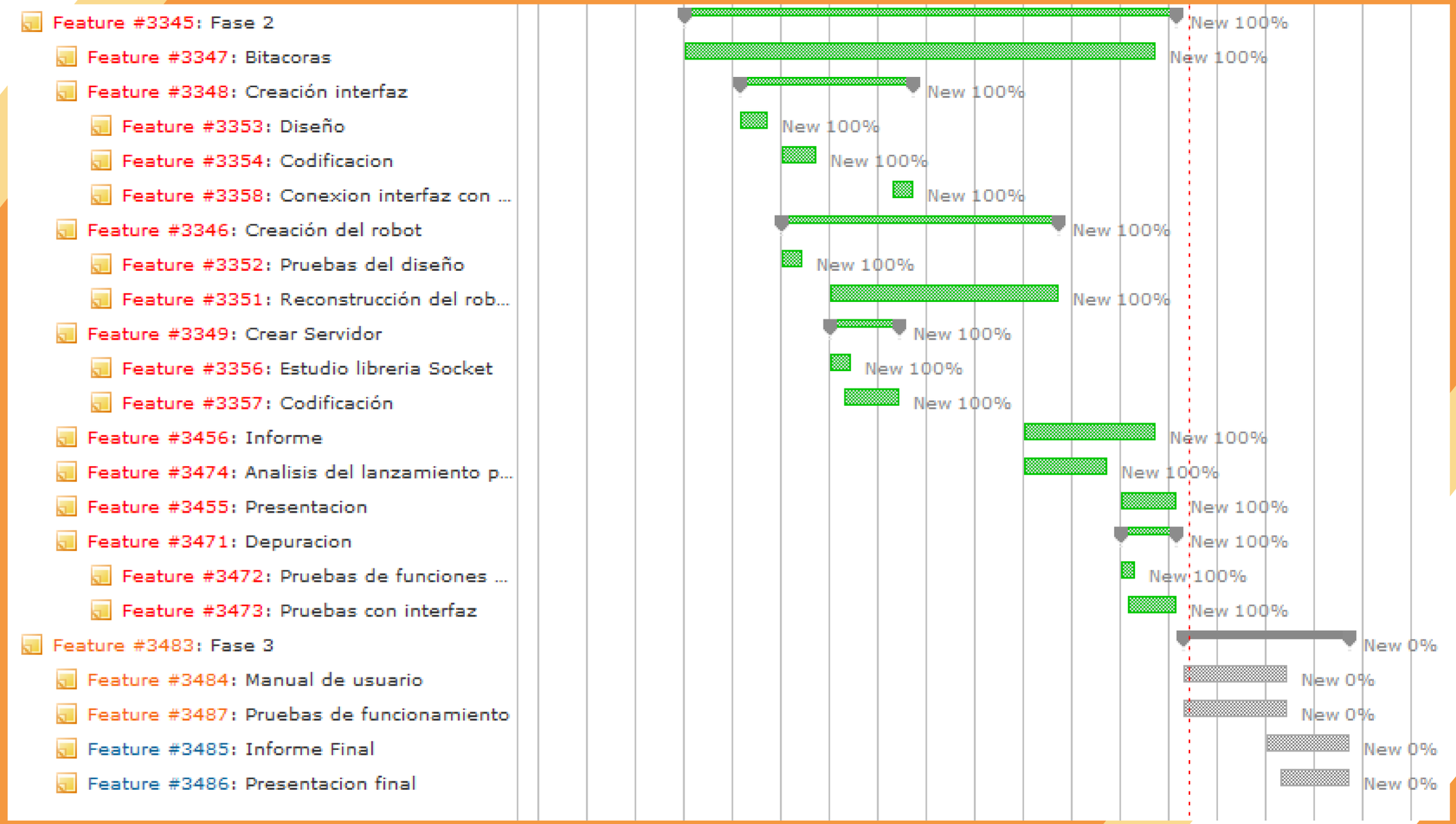
Documentador

Cristian Huanca

CARTA GANTT


2023-8
2023-9
32
33
34
35
36
37
PROYECTO 1 2023
GRUPO 1 A 2023
Feature #2844: Fase 1
Feature #2846: Documentación
Feature #2848: Bitácoras
Feature #2850: Informe 1
Feature #2851: Tareas
Feature #2855: Recuento de piezas, invest...
Feature #2859: Replantear diseño y estruc...
Feature #2865: Diseño Wiki
Feature #2866: Logo
Feature #2869: Diseño de Wiki
Feature #2870: Desarrollo del proyecto
Feature #2873: Codificación
Feature #2874: Instalación e Ingreso de E...
Feature #2877: Estudio de libreria EV3 y s...
Feature #2875: Programación de funcione...
Feature #2878: Presentación 1
Feature #2879: Asignación de partes de la...
Feature #2880: Creación y diseño
New 100%

CARTA GANTT


Feature #3345: Fase 2
Feature #3347: Bitacoras
Feature #3348: Creación interfaz
Feature #3353: Diseño
Feature #3354: Codificacion
Feature #3358: Conexion interfaz con ...
Feature #3346: Creación del robot
Feature #3352: Pruebas del diseño
Feature #3351: Reconstrucción del rob...
Feature #3349: Crear Servidor
Feature #3356: Estudio libreria Socket
Feature #3357: Codificación
Feature #3456: Informe
Feature #3474: Analisis del lanzamiento p...
Feature #3455: Presentacion
Feature #3471: Depuracion
Feature #3472: Pruebas de funciones ...
Feature #3473: Pruebas con interfaz
Feature #3483: Fase 3
Feature #3484: Manual de usuario
Feature #3487: Pruebas de funcionamiento
Feature #3485: Informe Final
Feature #3486: Presentacion final
New 100%
New 0%

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

NO FUNCIONALES

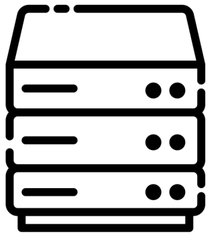
- El proyecto debe incluir un manual detallado con instrucciones completas sobre el funcionamiento integral del robot

- La interfaz gráfica debe contar con botones específicos para controlar el desplazamiento del robot, una sección para ajustar la rotación del palo de golf y un botón para controlar la posición de la base que sostiene la bola.

ARQUITECTURA

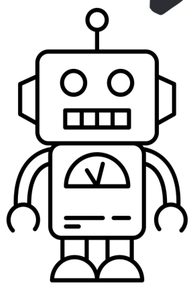
Cliente

Internet



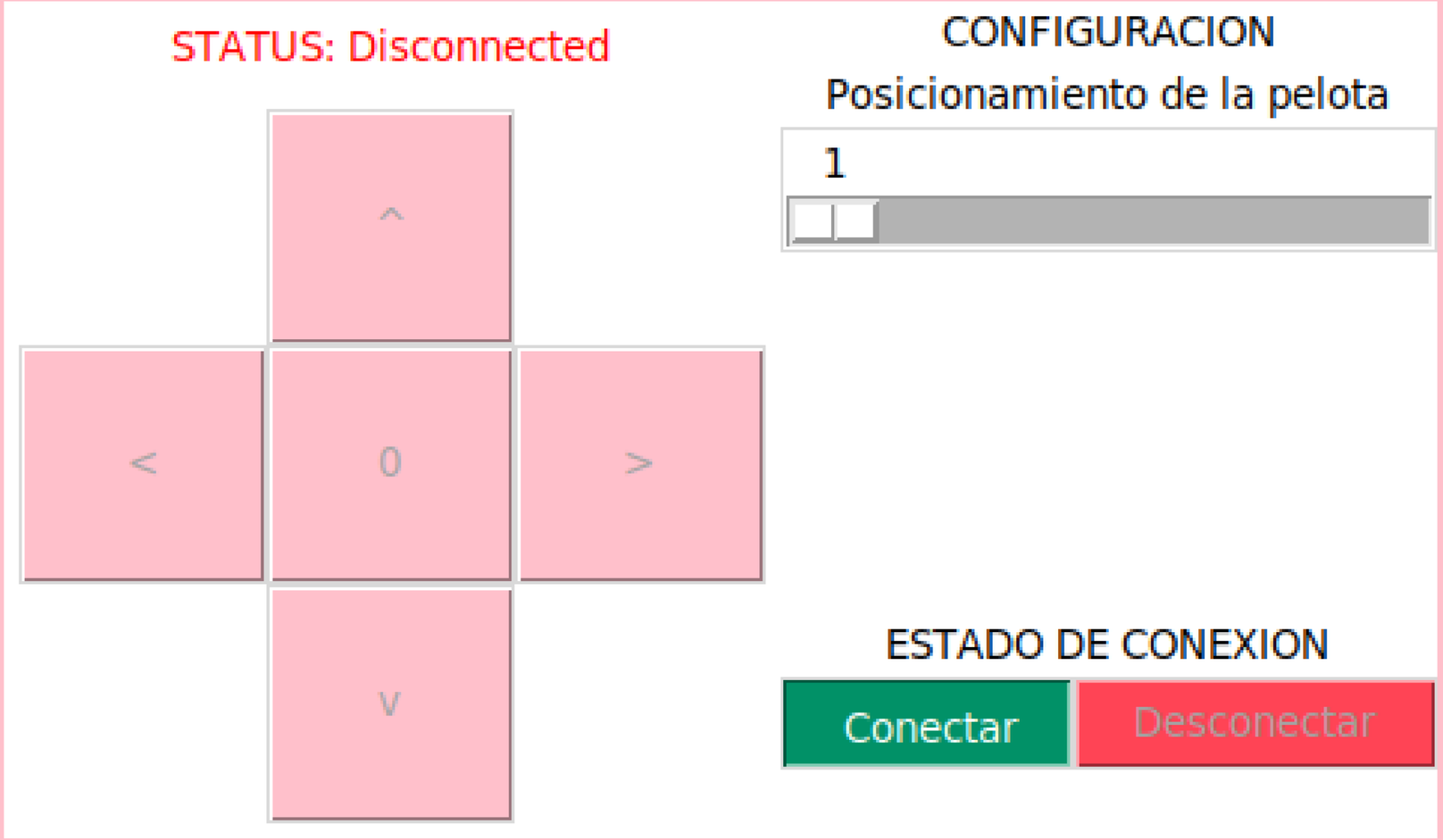
Servidor

Interfaz

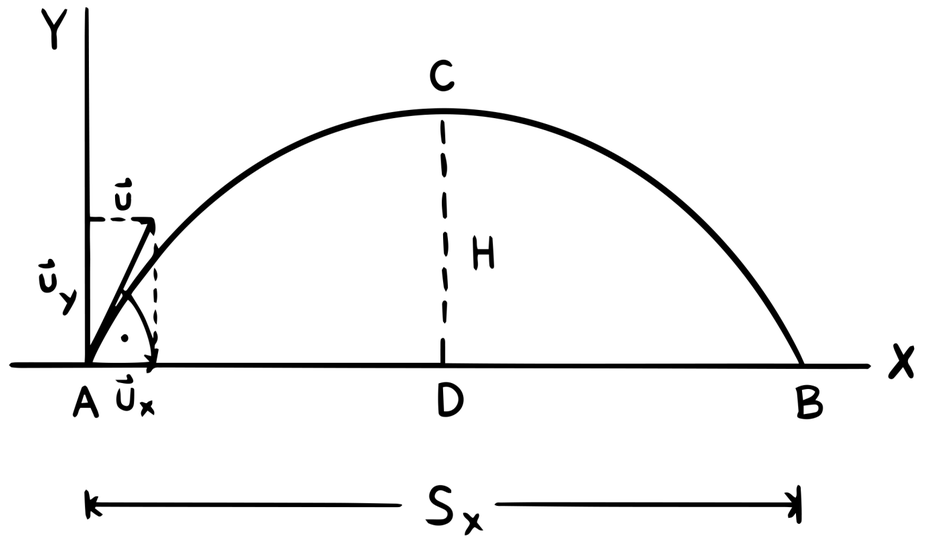


Robot

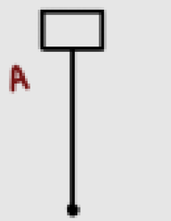
Interfaz


STATUS: Disconnected
^
<
0
>
v
CONFIGURACION
Posicionamiento de la pelota
1
ESTADO DE CONEXION
Conectar
Desconectar

Fundamentos de Proyectiles



Energía



$Em_A = Em_B$

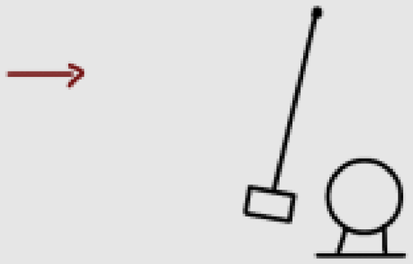
$U_{gA} = K_B$ → $m = 0.1\,kg$ $h = 25\,cm = 0.25\,m$

$mgh = \frac{1}{2} m V_B^2$

$0.245 = 0.05\, V_B^2$

$V_B = \sqrt{4.9}$

$V_B = 2.213$



Choques

$\vec{P_i} = \vec{P_f}$

m_p: masa palo
m_{pe}: masa pelota
V_p: : velocidad palo

$m_p V_{pi} + m_{pe} V_{pei} = m_{pf} V_{pf} + m_{pe} V_{pef}$ $(V_{pef} = V_{pf}) = V_f$

$(0.1)(2.213) = (0.1 + 0.07) V_f$

$0.2213 = 0.17\, V_f$

$V_f = 1.25$

Cinemática

$x_o = 0\,m$ $V_o = 1.25\,m/s$ $t = 1\,(s)$

$y_o = 0.2\,m$ $x_f = 1.2\,m$

$X = x_o + V_o \cos\alpha\, t$

$V_x = V_o \cos\alpha$

$Y = y_o + V_o \operatorname{sen}\alpha\, t - 4.9t^2$

$V_y = V_o \operatorname{sen}\alpha - 9.8t$

★ $X_f = 1.2\,m$

$1.2 = 1.25 \times 1 \times \cos\alpha$

$\cos\alpha = 0.96$

$\alpha = 16.26°$

★ $V_y = 0$ (Altura máxima)

$0 = 1.25 \times \operatorname{sen}(16.26°) - 9.8t$

$t = \dfrac{1.25\,\operatorname{sen}(16.26°)}{9.8} = 0.036\,(s)$

★ y_F (Altura máxima)

$t = 0.036$

$y_F = 0.2 + 1.25\,\operatorname{sen}16.26°\,(0.036) - 4.9\,(0.036)^2$

$y_F = 0.3\,m$

Descripción de los Programas

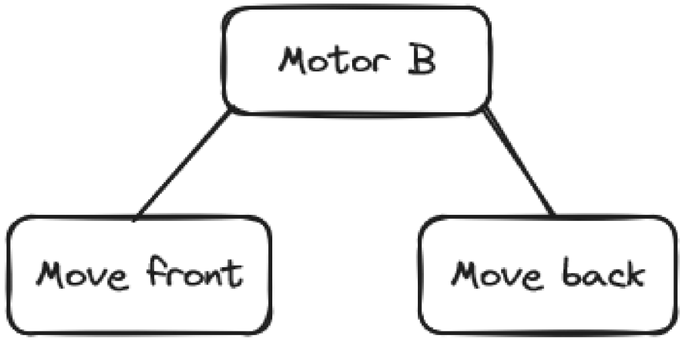
```python
import socket
from robot import Robot

server = socket.socket()

PORT = 8800

server.bind(('',PORT))
server.listen(1)


connection, address = server.accept()


robot = Robot()

actions = {
    "w" : robot.move_front,
    "a" : robot.move_left,
    "d" : robot.move_right,
    "s" : robot.move_back,
    "e" : robot.move_punch
}

while True:
    data = connection.recv(1)
    keyword = data.decode("utf-8")

    if keyword in actions:
        actions[keyword]()

    elif keyword == "q":
        break
```

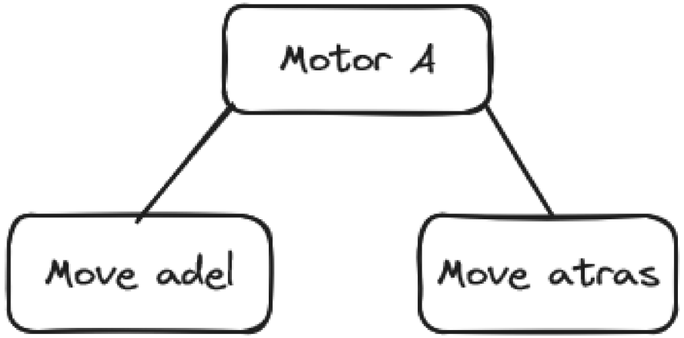
```python
from ev3dev2.motor import (LargeMotor, MoveTank, MediumMotor, OUTPUT_A, OUTPUT_B, OUTPUT_C, OUTPUT_D)
from time import sleep


class Robot:

    eje = 0
    base = -0.58
    MAX_VALUE_EJE = 140
    MIN_VALUE_EJE = -140
    MAX_VALUE_BASE = 0
    MIN_VALUE_BASE = -0.58
    SPEED = 100
    contaux1 = 0
    contaux2 = 0
    cont = 1


    def __init__(self):
        self.out_a = LargeMotor(OUTPUT_A) # movimiento pelota
        self.out_b = LargeMotor(OUTPUT_B) # ruedas de atras, movimiento
        self.out_c = MediumMotor(OUTPUT_C) # ruedas de rotacion, giro izquierda/derecha
        self.out_d = MediumMotor(OUTPUT_D) # lanzamiento de pelota
```

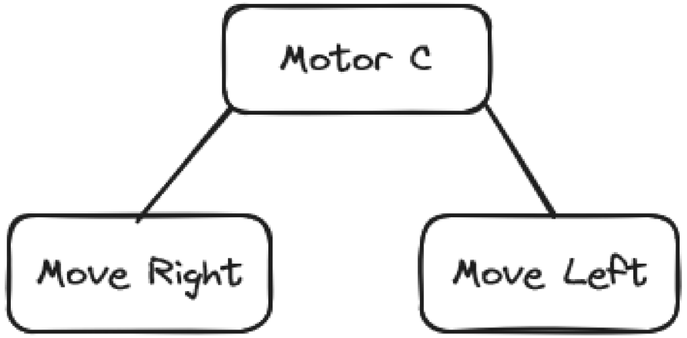
```
    # Función para desactivar completamente el robot
    def off_robot(self):
        self.out_a.stop()
        self.out_b.stop()
        self.out_c.stop()
        self.out_d.stop()

    # Función para mover el robot hacia adelante
    def move_front(self):
        self.out_b.on_for_rotations(speed = 100, rotations= -2, brake = True, block = True)
        self.out_b.stop()

    # Función para mover el robot hacia atras
    def move_back(self):
        self.out_b.on_for_rotations(speed = 100, rotations= 2, brake = True, block = True)
        self.out_b.stop()

    # Función para mover las ruedas del robot hacia la derecha
    def move_right(self):
        if(self.MIN_VALUE_EJE < self.eje): # Se comprueba el eje de las ruedas para no provocar una falla en el robot / para que no se queden pegadas
            self.out_c.on_for_degrees(speed = 100, degrees = -1.4, brake = True, block = True)
            self.eje-=1.4
            self.contaux1-=1
        print(self.eje)
        self.out_c.stop()

    # Función para mover las ruedas del robot hacia la izquierda
    def move_left(self):
        if(self.MAX_VALUE_EJE > self.eje): # Se comprueba el eje de las ruedas para no provocar una falla en el robot / para que no se queden pegadas
            self.out_c.on_for_degrees(speed = 100, degrees = 1.4, brake = True, block = True)
            self.eje+=1.4
            self.contaux2+=1
        print(self.eje)
        self.out_c.stop()
```

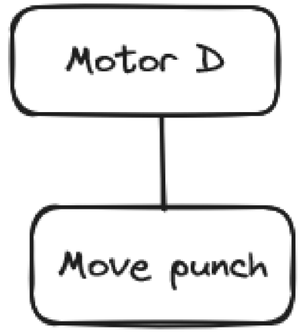
```
    # Función para ingresar los grados para hacer el golpe
    def move_punch(self, degrees):
        self.degrees = degrees
        self.out_d.on_for_degrees(speed = 100, degrees=degrees)
        self.out_d.stop()

    # Función para mover la base de la pelota, hacia adelante
    def mov_adel(self):
        if(self.MAX_VALUE_BASE>self.base): # Se comprueba la base de la pelota para que no falle el robot / en este caso para que no se golpee hacia atras
            self.out_a.on_for_rotations(speed = 100, rotations=-0.0116, brake = True, block = True)
            self.base += 0.0116
        self.out_a.stop()

    # Función para mover la base de la pelota, hacia atras
    def mov_atras(self):
        if(self.MIN_VALUE_BASE<self.base): # Se comprueba la base de la pelota para que no falle el robot / en este caso es para que no se pase hacia adelante
            self.out_a.on_for_rotations(speed = 100, rotations=0.0116, brake = True, block = True)
            self.base -= 0.0116
        self.out_a.stop()
    def off(self):
        if(self.base>-0.58):  # Se comprueba si esta en la posicion original la base de la pelota
            while(True):
                if(50 == self.cont):
                    return False # Se rompe el ciclo solo si la base tiene 4, donde estara en posicion original
                else:
                    self.out_a.on_for_rotations(speed = 100, rotations = 0.0116, brake = True, block = True)
                    self.base+=0.0116
                    self.cont = self.cont + 1
                    print(self.cont)
                    self.out_a.stop()
```

Diagrama


Motor B
Move front
Move back


Motor A
Move adel
Move atras


Motor C
Move Right
Move Left


Motor D
Move punch

Estado Actual

- Finalización de la versión del robot
- Funciones de movimiento de implementacion
- Interfaz grafica desarrollada con Tkinter
- Implementación del servidor
- Conexión remota establecida
- Documentación del proyecto
- Finalización de la versión del robot
- Registro de actividades informe y presentación

Problemas Encontrados
y
Soluciones Propuestas

Problemas	Soluciones
Falta de materiales	Comunicarse con el responsable de la distribución para obtener un mayor suministro.
Reconstrucción frecuente del robot	Llegar a una idea precisa que cumpla con todas las características y a su vez complete los objetivos establecidos.
Complicaciones con la estabilidad de la estructura en el diseño del robot	Reestructuración del diseño mediante ajustes en la base y fortalecimiento en los puntos débiles del robot.
Modificación frecuente de la interfaz gráfica y funciones debido a la reconstrucción del robot.	Terminar las modificaciones del robot para dar avance al desarrollo de la interfaz y funciones.

CONCLUSIÓN

UNIVERSIDAD DE TARAPACÁ
Universidad del Estado

Ingeniería
Computación e Informática

Proyecto I

Hole In One

Integrantes:
Tihare Cabello, Melisa Huanca, Cristian Huanca, Liliana Gálvez,
Byron Santibáñez