



UNIVERSIDAD DE TARAPACÁ
Universidad del Estado

Ingenierí@
Computación e Informática



Presentación avance del proyecto

Integrantes: Fabian Quezada, Diego Ferrada
Maykol Bravo, Joshua Jara, Javier Huanca.

Contenidos a tratar

01. Cambio de roles

02. Análisis y diseño

- Especificaciones de requerimiento

- Arquitectura

- Interfaz

03. Implementación

- Fundamentos de proyectiles

- Diagramas

- Descripción de programas

04. Avance del robot

05. Conclusión





01

Cambio de roles



Roles antiguos.

Jefe de grupo	Diego Ferrada
Programador	Joshua Jara
Documentador(es)	Maykol Bravo Javier Huanca
Ensamblador(es)	Maykol Bravo Fabian Quezada

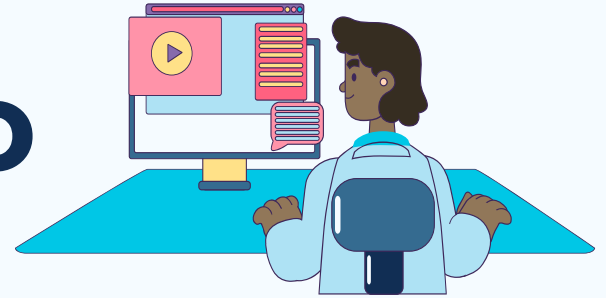
Roles actuales.

Jefe de grupo	Fabian Quezada
Programador	Diego Ferrada Maykol Bravo
Documentador(es)	Joshua Jara Fabian Quezada
Ensamblador(es)	Javier Huanca Maykol Bravo



02

Análisis y diseño



Requerimientos funcionales y no funcionales

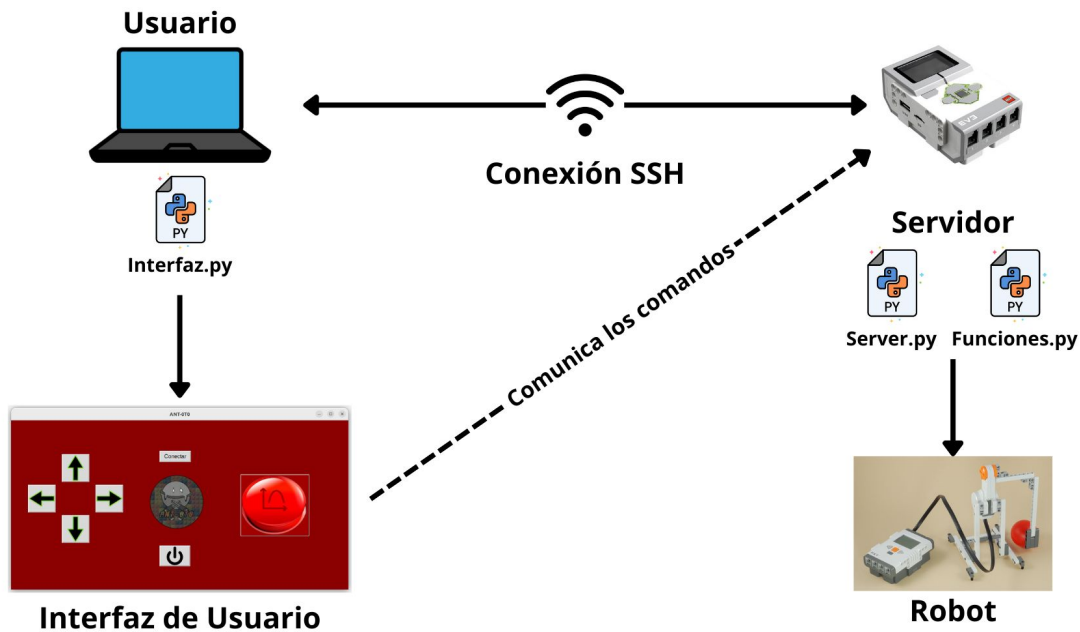
Funcionales

- El robot debe tener la capacidad de moverse
- El robot debe tener la capacidad de golpear una pelota de golf en una trayectoria específica.
- El robot debe ser controlado remotamente mediante una interfaz de usuario.

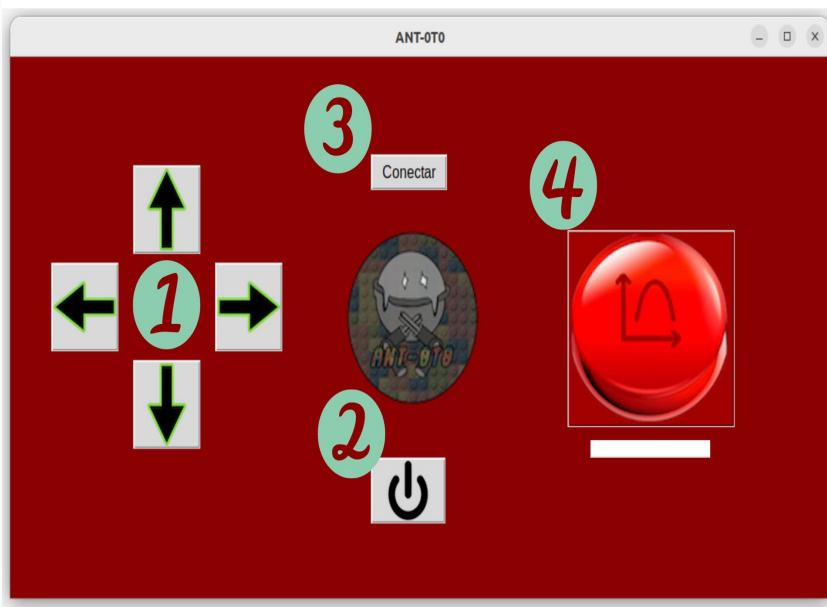
No funcionales

- Interfaz gráfica que contenga los controles necesarios para poner al robot en movimiento, conectarse al dispositivo controlador y golpear la pelota.
- El programa debe ser desarrollado con el lenguaje de programación Python en conjunto con la biblioteca ev3dev.
- El robot debe ser ensamblado utilizando piezas del kit Lego Mindstorms.

Arquitectura



Interfaz

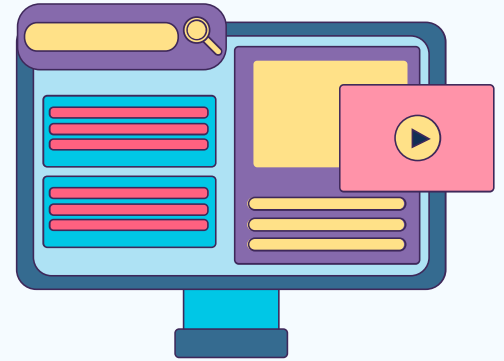


La interfaz del robot incluye un menú con varias opciones que nos permiten enviar instrucciones al robot directamente desde la interfaz.

- 1) Cada flecha representa un botón que activa el movimiento del robot en la dirección indicada.
- 2) Al presionar este botón, se finaliza el programa.
- 3) Al presionar este botón, permite la conexión del robot.
- 4) Al presionar este botón, lanza la pelota a una distancia proporcionada.

03

Implementación



Fundamentos de proyectiles



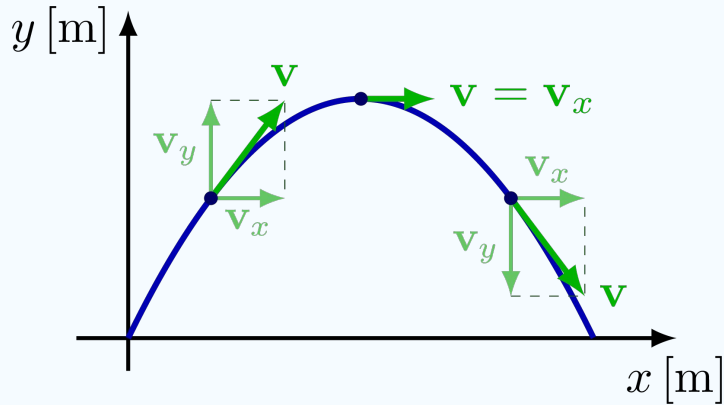
3.1 Configuración inicial

Ángulo de lanzamiento: 45°

Distancia entre la pelota y el objetivo: 1 [m]

Altura inicial de la pelota: 0,08 [m]

Fundamentos de proyectiles



3.2 Descomposición de velocidad inicial

Eje x $v_x = v_i \cos\theta$

Eje y $v_y = v_i \operatorname{sen}\theta$

Fundamentos de proyectiles

3.3 Ecuaciones de posición

$$* x(t) = x_0 + v_0 * t + \frac{1}{2}at^2$$

$$\text{Eje x} \quad x = v_i \cos\theta * t$$

$$\text{Eje y} \quad y = y_0 + v_i \text{sen}\theta * t - \frac{1}{2}gt^2$$

3.4 Condiciones para el encuentro

$$x_f = 1 ; y_f = 0$$

Fundamentos de proyectiles

$$1 \quad 1 = v_i \cos \theta * t \quad \theta = 45^\circ$$

$$2 \quad 0 = 0,08 + v_i \sin \theta * t - 4,9 * t^2$$

Fundamentos de proyectiles

$$1 \quad 1 = v_i \cos \theta * t \quad \theta = 45^\circ$$

$$2 \quad 0 = 0,08 + v_i \sin \theta * t - 4,9 * t^2$$

Despejando

$$1 \quad v_i = \frac{1}{\cos \theta * t}$$

$$\theta = 45^\circ$$

$$1 \quad 0 = 0,08 + \frac{\text{sen}\theta}{\text{cos}\theta} - 4,9t^2$$

$$2 \quad t = \sqrt{\frac{0,08 + \text{tan}\theta}{4,9}} \approx 0,58 \text{ s}$$

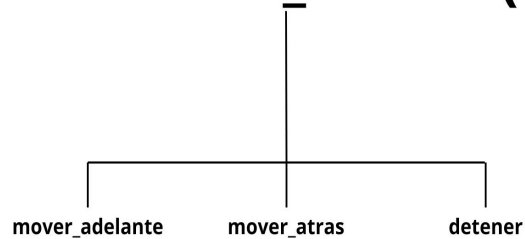
$$3 \quad v_i = \frac{1}{\text{cos}\theta * t} \approx 3,23 \text{ m/s}$$

Tiempo de vuelo: 0.58 s

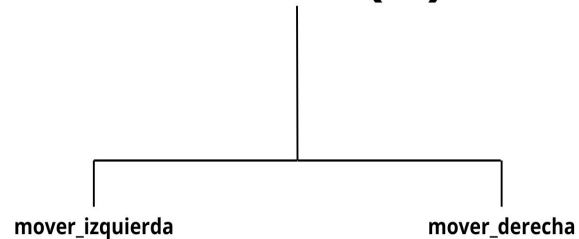
Velocidad inicial: 3.23m/s

Diagramas

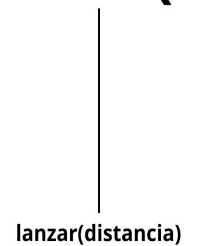
movimiento_ruedas(A,B)



mRotar(D)



mPalo(C)



Descripción del programa

Funciones

```
from ev3dev2.motor import LargeMotor, MediumMotor, OUTPUT_A, OUTPUT_B, OUTPUT_C, OUTPUT_D, SpeedRPM, MoveTank
from ev3dev2.sound import Sound
import math
import os
import sys
from time import sleep

#movimiento simultáneo de las ruedas(rueda izquierda, rueda derecha)
movimiento_ruedas = MoveTank(OUTPUT_A, OUTPUT_B)
#movimiento del palo del robot
mPalo = LargeMotor(OUTPUT_C)
#movimiento de rotación de las ruedas delanteras
mRotar = MediumMotor(OUTPUT_D)

#función para lanzar un objeto con el palo
def lanzar(distancia):
    pi = math.pi
    vel_necesaria = math.sqrt((4.9*distancia)/math.cos(pi/4)*math.sin(pi/4))
    vel_rpm = (60*vel_necesaria)/(2*pi*0.189)
    # Rotar el motor en -180 grados al 6% de velocidad
    mPalo.on_for_degrees(6,-180)
    # Rotar el motor en 225 grados al porcentaje de velocidad de vel_rpm
    mPalo.on_for_degrees(SpeedRPM(round(vel_rpm)),225)
    # Rotar el motor en -45 grados al 6% de la velocidad
    mPalo.on_for_degrees(6,-45)

#función para mover el robot hacia adelante
def mover_adelante():
    movimiento_ruedas.on(-100, -100)

#función para mover el robot hacia atras
def mover_atras():
    movimiento_ruedas.on(100, 100)

#función para mover el robot hacia la izquierda
def mover_izquierda():
    mRotar.on(50)

#función para mover el robot hacia la derecha
def mover_derecha():
    mRotar.on(-50)

#función para detener el robot
def detener():
    movimiento_ruedas.stop()
```

Descripción del programa

Servidor

```
import socket
from funciones import *

#se crea un socket
s = socket.socket()
s.settimeout(5)
print("Socket creado")
port = 2222
#se asocia una dirección IP y un número de puerto al socket
s.bind(("", port))
print("El socket se creo con puerto:{}".format(port))
#se comienza a escuchar conexiones entrantes (máximo 5 conexiones pendientes)
s.listen(5)
print("The socket is listening...")
#se acepta la conexión, creando un nuevo socket a partir de la conexión entrante y la dirección del cliente
connect, addr = s.accept()
print("Se conecto a {}".format(addr))
while True:
    #se define el carácter que el socket del cliente recibe como dato
    rawByte = connect.recv(1)
    #se traduce el carácter recibido
    char = rawByte.decode('utf-8')
    if (char == 'w'):
        mover_adelante()
    if (char == 's'):
        mover_atras()
    if (char == 'a'):
        mover_izquierda()
    if (char == 'd'):
        mover_derecha()
    if (char == ' '):
        detener()
    if (char == 'l'):
        lanzar(2)
```

Descripción del programa

Interfaz

```
import tkinter as tk
from tkinter import ttk
from tkinter import messagebox
from tkinter import *
import socket
import sys

ev3_ip = "192.168.70.147"

#Ventana
root = tk.Tk()
root.config(width=1000, height=500)
root.config(bg="darkred")
root.title("ANT-0T0")

#Acciones del robot
def irAdelante():
    clientSocket.send(bytes([ord('w')]))

def irAtras():
    clientSocket.send(bytes([ord('s')]))

def irIzquierda():
    clientSocket.send(bytes([ord('a')]))

def irDerecha():
    clientSocket.send(bytes([ord('d')]))

def inicioLanzar():
    clientSocket.send(bytes([ord('l')]))

def soltar(event):
    clientSocket.send(bytes([ord(' ')]))

def conectar(address):
    try:
        clientSocket.connect((address,port))
        messagebox.showinfo("Mensaje Servido","Cliente conectado al robot: {0} : {1}".format(address,port))
    except socket.error:
        messagebox.showwarning("Conexión errónea","No se ha logrado al conexión, verifique la ip {0}".format(address))
    clientSocket.close()
```

Interfaz

```
#Imágenes
imgLogo = tk.PhotoImage(file="logo_ant0t0.png")
imgFlechaIzq = tk.PhotoImage(file="flecha_izquierda.png")
imgFlechaSup = tk.PhotoImage(file="flecha_superior.png")
imgFlechaDer = tk.PhotoImage(file="flecha_derecha.png")
imgFlechaInf = tk.PhotoImage(file="flecha_inferior.png")
imgApagar = tk.PhotoImage(file="Boton_Apagar.png")
imgLanzar = tk.PhotoImage(file="imgLanzar.png")

etiqueta_logo = tk.Label(root, image=imgLogo, bg = "darkred").place(x=350,y=150)
#Boton Flecha Izquierda
Bt_FlechaIzq = Button(root, image=imgFlechaIzq, command=irIzquierda)
Bt_FlechaIzq.place(x=50,y=190)
#Boton Flecha Superior
Bt_FlechaSup = Button(root, repeatdelay=50, repeatinterval=50, image=imgFlechaSup, command=irAdelante)
Bt_FlechaSup.place(x=150,y=100)
Bt_FlechaSup.bind('<ButtonRelease-1>', soltar)
#Boton Flecha Derecha
Bt_FlechaDer = Button(root, image=imgFlechaDer, command=irDerecha)
Bt_FlechaDer.place(x=250,y=190)
#Boton Flecha Inferior
Bt_FlechaInf = Button(root, repeatdelay=50, repeatinterval=50, image=imgFlechaInf, command=irAtras)
Bt_FlechaInf.place(x=150,y=280)
Bt_FlechaInf.bind('<ButtonRelease-1>', soltar)

#Boton Apagado
Bt_Apagar = Button(root, image=imgApagar, command=root.destroy).place(x=440,y=370)

#Boton Lanzar
Bt_Lanzar = Button(root, image=imgLanzar, bg = "darkred", command=inicioLanzar)
Bt_Lanzar.place(x=670, y=160)

#Boton Conectar
button_connect = tk.Button(root, text="Conectar", command= lambda: {conectar(ev3_ip)}, font=("Arial",12)).place(x=440,y=90)

if len(sys.argv) > 2:
    print("usage:client-laptop.py [IP-addr-of-robot]")
    sys.exit(1)

elif len(sys.argv) == 2:
    ipAddress = sys.argv[1]
    print("using specified IP address: {}".format(ipAddress))

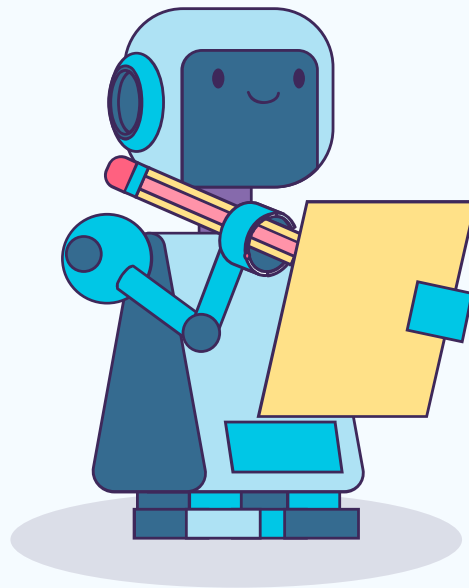
else:
    print("using default IP address: {}".format(ev3_ip))

clientSocket = socket.socket()
port = 2222

root.mainloop()
```

05

Resultados



Estado Actual del Proyecto

- Versión del robot “ANT-0T0” semi finalizada (Estructura final susceptible a pequeñas modificaciones).
- Sistema de movimiento terminado, giro en direcciones laterales, aceleración y reversa finalizados.
- Sistema de disparo casi finalizado, palo de golf terminado, base para la pelota a modificar.
- Informe corregido en su totalidad según la retroalimentación del informe de planificación.
- Conexión al servidor del robot finalizada.
- Código del robot finalizado, funciones de movimiento y giro listas para la fase de pruebas.
- Interfaz gráfica finalizada, realizada con la librería tkinter en Python.
- Actualización de archivos y wiki en redmine.
- Mejora a la Carta Gantt.



Problemas Encontrados y Solución Propuesta

Problema Encontrado	Solución Propuesta
Problemas de estabilidad en la base del robot	Quitar piezas innecesarias del robot, re-distribuir el peso y añadir estructuras que mejoren la estabilidad
Problemas con el sistema de giro del robot	Reemplazar el sistema de giro que utilizaba dos motores para redireccionar las ruedas por uno de un motor que utiliza engranajes para realizar el giro
Base inadecuada para la pelota de golf	Reconstruir la base para la pelota por una que no interfiera con el paso del palo que golpea la pelota
Problemas de conectividad con el servidor	Realizar apropiadamente la conexión mediante ssh y ejecutar el archivo que contiene el servidor directamente desde el robot
Adaptabilidad a los nuevos roles	Realizar una reunión de retroalimentación sobre el progreso de cada integrante en su anterior rol y establecer las tareas futuras en los nuevos roles



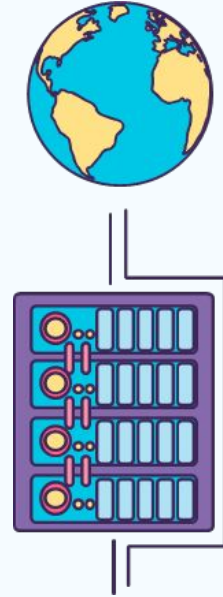
06

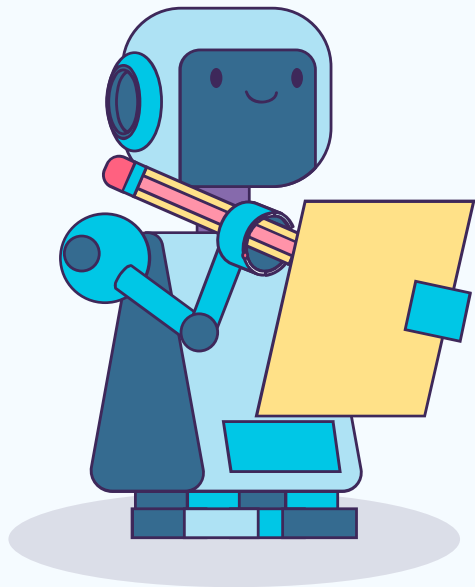
Conclusión



Conclusión

El proyecto "ANT-0T0" ha sido un logro integral que destaca la aplicación efectiva de conceptos teóricos en la construcción de un robot de minigolf utilizando el kit LEGO Mindstorms EV3. La utilización de la conexión SSH facilitó la conexión remota con el robot, mientras que el desarrollo de la interfaz gráfica permitió un control intuitivo. La planificación meticulosa, expresada en la carta Gantt y la plataforma Redmine, demostró ser esencial para una gestión efectiva del tiempo y los recursos. En resumen, "ANT-0T0" ha representado un desafío integral que ha permitido integrar teoría y práctica, fortalecer habilidades técnicas y de gestión de proyectos, y alcanzar con éxito los objetivos propuestos.





**Gracias por su
atención**