**UNIVERSIDAD DE TARAPACÁ**



**FACULTAD DE INGENIERÍA**

Departamento de Ingeniería en Computación e Informática



**Avance de Proyecto  
Robot “CL4W-T4NK3R”**

**Autor(es): Marco Chayo.**

**Patricio Gutiérrez.**

**Katia Layi.**

**Adolfo Navea.**

**Sebastián Torres.**

**Asignatura: Proyecto 1**

**Profesor(es): Ricardo Valdivia Pinto**

ARICA, 17 de octubre 2019

# Historial de Cambios

|  |  |  |  |
| --- | --- | --- | --- |
| **Fecha** | **Versión** | **Descripción** | **Autor(es)** |
| 04/09/2019 | 1.0 | Informe sobre la formulación del proyecto. | Marco Chayo, Patricio Gutiérrez, Katia Layi, Adolfo Navea y Sebastián Torres. |
| 24/09/19 | 2.0 | Corrección de la formulación del proyecto. | Katia Layi y Sebastián Torres. |
| 26/09/19 | 2.1 | Inicio de informe de avance de proyecto. | Katia Layi y Sebastián Torres. |
| 08/10/19 | 2.2 | Agregación del código final y de la interfaz. | Katia Layi y Sebastián Torres. |
| 15/10/19 | 2.3 | Revisión y corrección de últimos detalles del avance de proyecto. | Katia Layi y Sebastián Torres. |
| 17/10/2019 | 3.0 | Informe de avance del proyecto. | Katia Layi y Sebastián Torres. |

Tabla de contenido

[Historial de Cambios 2](#_Toc22154374)

[1. Panorama General 4](#_Toc22154375)

[1.1. Introducción 4](#_Toc22154376)

[1.2. Objetivo General 4](#_Toc22154377)

[1.3. Objetivos Específicos 4](#_Toc22154378)

[1.4. Restricciones 5](#_Toc22154379)

[1.5. Entregables 5](#_Toc22154380)

[2. Organización del Personal 5](#_Toc22154381)

[2.1. Descripción de Roles 5](#_Toc22154382)

[2.2. Personal que cumplirá los Roles 6](#_Toc22154383)

[2.3. Mecanismos de Comunicación 7](#_Toc22154384)

[3. Planificación del Proyecto 7](#_Toc22154385)

[3.1. Actividades 7](#_Toc22154386)

[3.2. Asignación de tiempo 9](#_Toc22154387)

[3.3. Gestión de Riesgos 9](#_Toc22154388)

[4. Planificación de Recursos 10](#_Toc22154389)

[4.1. Recursos Hardware-Software requeridos 10](#_Toc22154390)

[4.2. Estimación de Costos 10](#_Toc22154391)

[5. Análisis y diseño 11](#_Toc22154392)

[5.1. Especificación de requerimientos (Funcionales y no funcionales) 11](#_Toc22154393)

[5.2. Arquitectura propuesta 11](#_Toc22154394)

[5.3. Diseño de la interfaz de usuario 12](#_Toc22154395)

[6.  Implementación 12](#_Toc22154396)

[7. Resultados 13](#_Toc22154397)

[7.1. Estado actual del proyecto 13](#_Toc22154398)

[7.2. Problemas encontrados y soluciones propuestas 14](#_Toc22154399)

[7.3. Conclusiones 15](#_Toc22154400)

[7.4. Trabajo a futuro 15](#_Toc22154401)

[8. Referencias 15](#_Toc22154402)

[Anexo 16](#_Toc22154403)

1. **Panorama General**
   1. **Introducción**

Mediante el presente informe, daremos a conocer el proceso inicial de planeación y construcción de un robot Lego Mindstorms [1] además de la programación del “Brick” en MobaTextEditor y Visual Studio Code en el lenguaje de programación Python.

La construcción del robot está adaptada para que participe en una competencia de Flip Tac Toe [2] contra otros robots similares. El juego consiste en hacer una línea horizontal, vertical o diagonal con piezas (latas de bebidas en este caso) del mismo color y cada robot cuenta con una única oportunidad de hacer girar la pieza (cada lata tendrá dos colores diferentes).

Se mostrará la organización y distribución de los roles que desempeñará cada integrante del equipo tanto en la programación como en la documentación. Además, se enseñarán las actividades que se realizarán durante el desarrollo del proyecto.

* 1. **Objetivo General**

El objetivo general de nuestro proyecto es poder realizar la construcción y programación de un robot a base de piezas Lego Mindstorms EV3, el cual sea capaz de poder competir contra otros robots en un juego llamado Flip Tac Toe.

* 1. **Objetivos Específicos**

Los objetivos específicos que se nos presentaron a lo largo de la creación del proyecto son:

1. Buscar un diseño [3] apropiado para el “CL4W-T4NK3R” para lo cual se llevó a cabo una búsqueda de diseños y llegamos a la decisión de crear un robot estilo tanque. La pinza [4] que utilizará el robot para poder agarrar los objetos fue creado a partir de las ideas de los integrantes del grupo.
2. Programar e implementar los movimientos [5] básicos para que el robot pueda moverse hacia adelante y hacia atrás.
3. Implementación de instrucciones y movimientos complementarios, es decir, programar el robot para que pueda seguir las instrucciones dadas y que, además, pueda abrir, cerrar y girar la pinza.
4. Hacer experimentos para probar si funcionan correctamente todas las funciones del robot para que pueda participar en la competencia de Flip Tac Toe.
5. Desarrollo de la interfaz de usuario, en el cual se tendrá una comunicación remota con el robot.
   1. **Restricciones**

En el proceso de construcción del robot nos encontramos con restricciones tales como:

1. La falta de motores y piezas que necesitábamos para el diseño por lo que tuvimos que ocupar y mezclar otras piezas para poder reemplazar las faltantes.
2. Problemas al momento de programar debido a que no todos los del grupo sabían exactamente cómo programar en Python por lo que fue necesario una búsqueda de información y de comandos para realizar la programación.
3. Restricción de tiempo, debido a que el proyecto debe estar finalizado al término de este semestre.
4. Falta de algún integrante del grupo, para solucionar este problema hay que ajustar las actividades de dicha semana para no retrasar el progreso.
   1. **Entregables**

Al momento de finalizar este proyecto se entregarán los siguientes artículos:

1. El robot ya completado y funcionando, demostrando que puede realizar las funciones como moverse en diferentes sentidos, agarrar objetos, voltearlos y luego volverlos a dejar en su sitio.
2. Un manual de usuario en el cual se especificará el modo de uso y las instrucciones que se le puede dar al robot.
3. Un video promocional donde se mostrará al robot funcionando y sus características.
4. Un informe en el cual se especificará todo el proceso de la creación y construcción del robot. Este informe estará constituido por informes parciales previos hechos durante el semestre.
5. Una wiki, en donde se mostrará la información del proyecto.
6. **Organización del Personal**
   1. **Descripción de Roles**

Durante todo el proceso se necesitó que cada integrante cumpliera ciertos roles para llevar a cabo una mejor realización y comunicación logrando así completar las tareas semanales en el tiempo especificado. Los roles que se llevaron a cabo durante todo el proceso fueron: Encargado del diseño del robot, encargado de la búsqueda de diseños, encargado de la construcción de la base del robot, encargado de la construcción de la pinza del robot, encargado de la decoración, encargado de la programación, encargado de las pruebas de funcionamiento del robot, encargado de la creación del Power Point, encargado de la creación del informe, encargado de la creación del manual de usuario, encargado de la creación del video, encargado de las bitácoras, encargado de la carta Gantt, encargado de las actividades y finalmente el jefe de grupo.

Los roles generales que cumplirán los integrantes del equipo son:

|  |  |  |
| --- | --- | --- |
| **Roles** | **Encargado(s)** | **Descripción** |
| Jefe de proyecto | Sebastián Torres | Encargado de representar al equipo de trabajo, de la organización y de la toma de decisiones. |
| Diseñador(es) y constructor(es) | Katia Layi y Adolfo Navea | Encargados de diseñar el modelo del robot y construirlo de tal manera que el robot pueda moverse en todas las direcciones y pueda tomar objetos, voltearlos y soltarlos. |
| Programador(es) | Marco Chayo, Patricio Gutiérrez y Adolfo Navea | Encargados de desarrollar e implementar el código en Python logrando así que el robot pueda ejecutar las acciones solicitadas. |
| Documentador(es) | Katia Layi y Sebastián Torres | Encargados de realizar los informes, presentaciones, bitácoras, video, manual de usuario y wiki del proyecto. |

* 1. **Personal que cumplirá los Roles**

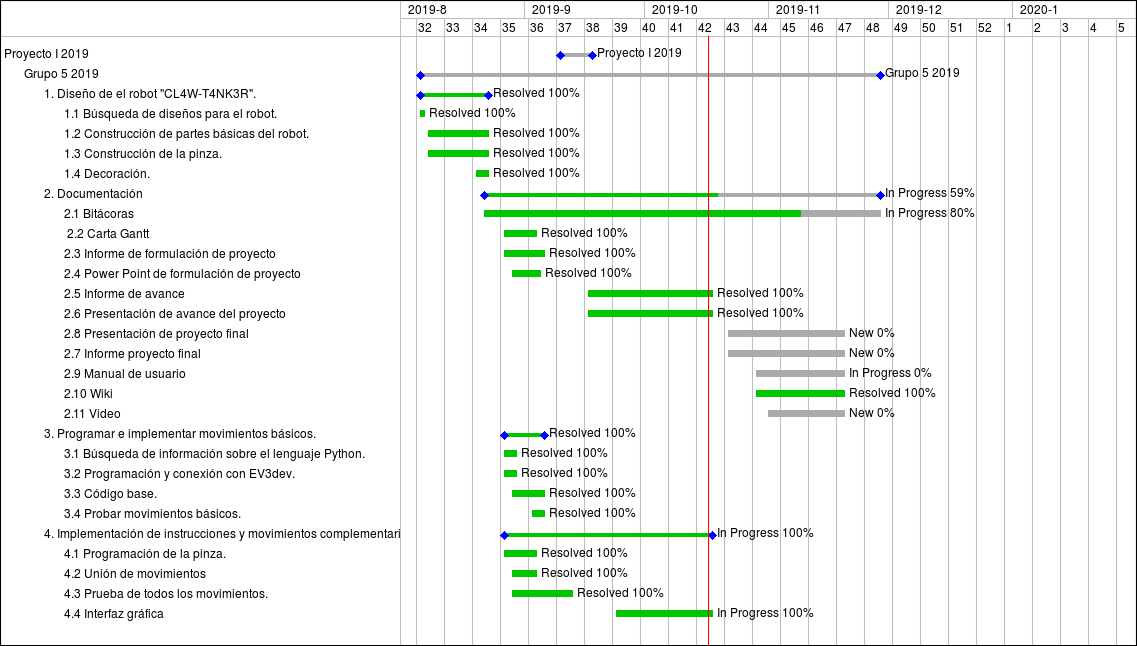
|  |  |
| --- | --- |
| **Roles** | **Encargado(s)** |
| Diseño del robot. | Katia Layi, Adolfo Navea y Patricio Gutiérrez. |
| Búsqueda de diseños. | Adolfo Navea y Patricio Gutiérrez. |
| Construcción de la base del robot. | Katia Layi. |
| Construcción de la pinza del robot. | Marco Chayo, Adolfo Navea y Patricio Gutiérrez. |
| Decoración. | Katia Layi y Adolfo Navea. |
| Programación. | Marco Chayo, Adolfo Navea y Patricio Gutiérrez. |
| Probar funcionamiento del robot. | Adolfo Navea, Patricio Gutiérrez y Marco Chayo. |
| Creación del Power Point. | Patricio Gutiérrez, Marco Chayo, Katia Layi, Adolfo Navea y Sebastián Torres. |
| Creación del informe. | Katia Layi y Sebastián Torres. |
| Creación del manual de usuario. | Katia Layi y Sebastián Torres. |
| Creación del video. | Patricio Gutiérrez, Marco Chayo, Katia Layi, Adolfo Navea y Sebastián Torres. |
| Creación de la interfaz. | Adolfo Navea, Patricio Gutiérrez y Marco Chayo. |
| Creación de las bitácoras. | Sebastián Torres. |
| Carta Gantt. | Sebastián Torres y Katia Layi. |
| Creación de la Wiki. | Katia Layi. |
| Jefe de grupo. | Sebastián Torres. |

* 1. **Mecanismos de Comunicación**

Para realizar el proyecto, la mejor manera de comunicarnos es en clases, pero para trabajar a distancia utilizamos la plataforma “Discord” [6] el cual es un medio muy eficiente de comunicación, además nos comunicamos a través de “WhatsApp” para enviarnos documentos, enlaces y mensajes de manera rápida.

1. **Planificación del Proyecto**
   1. **Actividades**

|  |  |  |  |
| --- | --- | --- | --- |
| **Actividad** | **Descripción** | **Responsable** | **Producto** |
| Diseño del robot. | Búsqueda de diseños y elección del más óptimo. | Adolfo Navea. | Elección del modelo tipo “Tanque”. |
| Construcción base. | Construcción de la base del robot. | Katia Layi. | Robot con base “Tanque”. |
| Construcción pinza. | Construcción de la pinza del robot. | Adolfo Navea. | Pinza que se abre, cierra y gira. |
| Decoración. | Decoración y agregación de la pinza a la base. | Katia Layi. | Robot tipo “Tanque” con pinza. |
| Programación de movimientos. | Programación de movimientos del robot. | Marco Chayo. | Robot que se mueve hacia adelante y atrás. |
| Programación de pinza. | Programación de los movimientos de la pinza. | Patricio Gutiérrez. | La pinza se abre, cierra y gira sobre su eje. |
| Bitácoras. | Creación de las bitácoras de cada semana. | Sebastián Torres. | Bitácoras semanales. |
| Informe. | Creación del informe del proyecto. | Katia Layi. | Informe. |
| Manual de usuario. | Creación del manual de usuario y sus instrucciones. | Sebastián Torres. | Manual de usuario. |
| Power Point. | Creación del Power Point para la exposición | Sebastián Torres. | Power Point. |
| Carta Gantt. | Creación de la Carta Gantt con las actividades programadas. | Sebastián Torres. | Carta Gantt. |
| Creación del video. | Creación y diseño del video sobre el robot. | Marco Chayo. | Video promocional del robot. |
| Creación de la Wiki. | Creación y diseño de la Wiki del proyecto. | Katia Layi. | Wiki. |
| Programación de la comunicación remota. | Programación para comunicarse a través del pc con el robot. | Patricio Gutiérrez. | Comunicación remota. |
| Creación de la interfaz. | Diseño de la interfaz para mover el robot. | Marco Chayo. | Interfaz de usuario. |

* 1.  **Asignación de tiempo**
  2. **Gestión de Riesgos**

|  |  |  |  |
| --- | --- | --- | --- |
| **Riesgos** | **Probabilidad de ocurrencia** | **Nivel de impacto** | **Acción remedial** |
| Piezas defectuosas. | 40% | 3 | Reemplazar las piezas defectuosas por otras nuevas |
| Daño de la tarjeta SD. | 15% | 1 | Comprar una nueva tarjeta SD y usar una copia del código |
| Ausencia de un integrante. | 10% | 2 | Redistribuir las tareas asignadas para ese día |
| Falta de tiempo. | 30% | 1 | Hacer una adaptación con lo anteriormente trabajado |
| Daño del robot por terceros. | 40% | 1 | Copia del diseño y fotos |
| Daño del robot en el testeo. | 60% | 1 | Seguir testeando para poder hacer un programa óptimo y seguro para el robot |

1. **Planificación de Recursos**
   1. **Recursos Hardware-Software requeridos**

Para la construcción y programación del robot “CL4W-T4NK3R” se utilizó:

1. Hardware: Tarjeta SD de 8 GB, laptops, “Lego Mindstorms EV3”, teléfonos celulares, EV3DEV, dongle wifi tenda.
2. Software: “MobaTextEditor”, “Visual Studio Code”, lenguaje de programación “Python”, librerías (rpyc, msvcrt, os), conexión con el robot vía Putty.
   1. **Estimación de Costos**

Recursos utilizados:

|  |  |  |  |
| --- | --- | --- | --- |
| **Recurso(s)** | **Valor de recurso** | **Cantidad** | **Valor total** |
| Lego Mindstorm Ev3 Core set | $671.257 | 1 | $671.257 |
| Tarjeta SD de 8 GB | $3.000 | 1 | $3.000 |
| Computadores | $660.000 | 3 | $1.980.000 |
| Dongle wifi tenda | $7.334 | 1 | $7.334 |
| Microsoft Office Home & Student | $35.841 | 3 | $143.364 |

Monto total de recursos utilizados: $2.804.955.

Mano de obra:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tarea** | **Valor mano de obra** | **Cantidad de trabajadores** | **Horas trabajadas** | **Valor total** |
| Programador | $5.000 por hora | 3 | 22 | $330.000 |
| Diseñador y constructor | $6.000 por hora | 2 | 5 | $60.000 |
| **Tarea** | **Valor mano de obra** | **Cantidad de trabajadores** | **Total de documentos** | **Valor total** |
| Documentación | $6.000 por documento | 2 | 6 | $36.000 |

Monto total de mano de obra: $426.000.

Monto total (Total de recursos + Total de mano de obra): $3.230.955.

# Análisis y diseño

## **5.1.** **Especificación de requerimientos (Funcionales y no funcionales)**

Requerimientos funcionales:

* Se construirá un robot el cual se comunica vía Wifi con el computador, logrando así que el usuario pueda elegir los movimientos que realizará el robot mediante la interfaz gráfica que está desarrollada en el robot.
* La interfaz gráfica contiene los movimientos necesarios para poder avanzar, tomar un objeto con la pinza, voltear este objeto y finalmente, soltar el objeto.

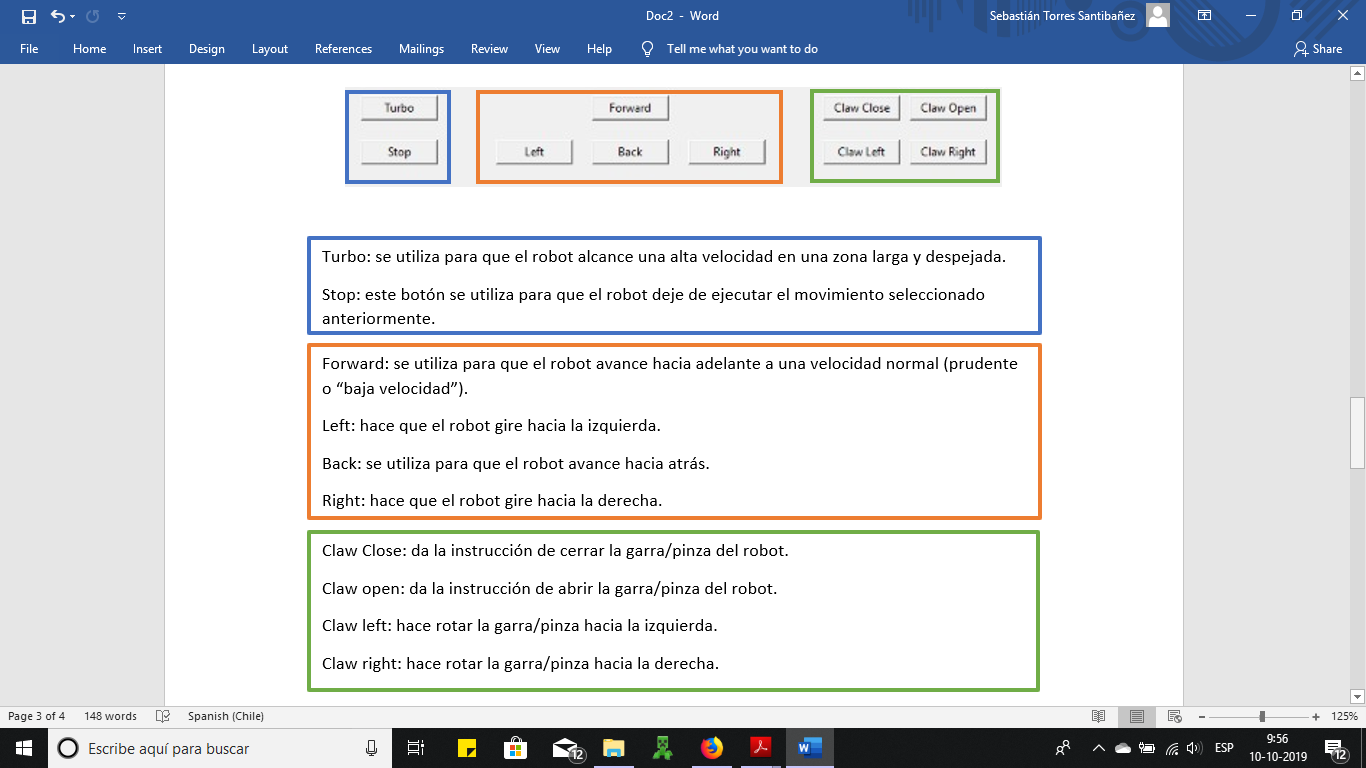
Requerimientos no funcionales:

* Cada movimiento del robot se realizará en un tiempo y velocidad estimados debido a que en la interfaz se puede ajustar la velocidad de movimiento.
* El proyecto contará con un manual de usuario estructurado para el buen entendimiento de sus funciones.
* El proyecto poseerá una interfaz gráfica adecuada y fácil de entender.

## **5.2.** **Arquitectura propuesta**

1. Ambos dispositivos (robot y computador) conectados a la misma red Wifi.
2. Se inicia el servidor del robot y su respectiva conexión con el computador del usuario.
3. El cliente abre la interfaz de usuario, la cual tendrá los controles/botones utilizados para que el robot realice los movimientos (dichos movimientos se envían al servidor para que el Brick los lea).
4. El robot realiza los movimientos enviados desde el cliente en simultáneo.

## **5.3. Diseño de la interfaz de usuario**

La interfaz de usuario consta de una pantalla principal con botones de instrucciones para el robot.

Turbo: Se utiliza para que el robot alcance una alta velocidad en una zona larga y despejada.

Stop: Este botón se utiliza para que el robot deje de ejecutar el movimiento seleccionado anteriormente.

Forward: Se utiliza para que el robot avance hacia adelante a una velocidad normal (prudente o “baja velocidad”).

Back: Se utiliza para que el robot avance hacia atrás a una velocidad normal.

Left: Se utiliza para que el robot gire hacia la izquierda.

Right: Se utiliza para que el robot gire hacia la derecha.

Claw Close: Da la instrucción de cerrar la garra/pinza del robot.

Claw Open: Da la instrucción de abrir la garra/pinza del robot.

Claw Left: Da la instrucción de rotar la garra/pinza hacia la izquierda.

Claw Right: Da la instrucción de rotar la garra/pinza hacia la derecha.

# 6.  Implementación

Para la implementación utilizamos la clase MyService la cual contiene las acciones en funciones que controlan los motores del robot.

Los métodos de la clase son:

* def forward(): Esta funcion acciona los LargeMotor, permitiendo al robot avanzar.
* def parar(): Esta funcion detine la accion de todos los motores (LargeMotor y MediumMotor).
* def back(): Esta funcion acciona los LargeMotor, permitiendo al robot retroceder.
* def giroizq(): Esta funcion acciona el MediumMotor conectado a la salida B, permitiendo al robot girar la garra hacia la izquierda.
* def giroder(): Esta funcion acciona el MediumMotor conectado a la salida B, permitiendo al robot girar la garra hacia la derecha.
* def left(): Esta funcion acciona los LargeMotor con velocidades inversas, permitiendo al robot virar hacia la izquierda.
* def right(): Esta funcion acciona los LargeMotor con velocidades inversas, permitiendo al robot virar hacia la derecha.
* def nitro(): Esta funcion acciona los LargeMotor, permitiendo al robot avanzar a una velocidad mayor.
* def abrir(): Esta funcion acciona el MediumMotor conectado a la salida B, permitiendo al robot abrir la garra.
* def cerrar(): Esta funcion acciona el MediumMotor conectado a la salida B, permitiendo al robot cerrar la garra.

# 7. Resultados

**7.1. Estado actual del proyecto**

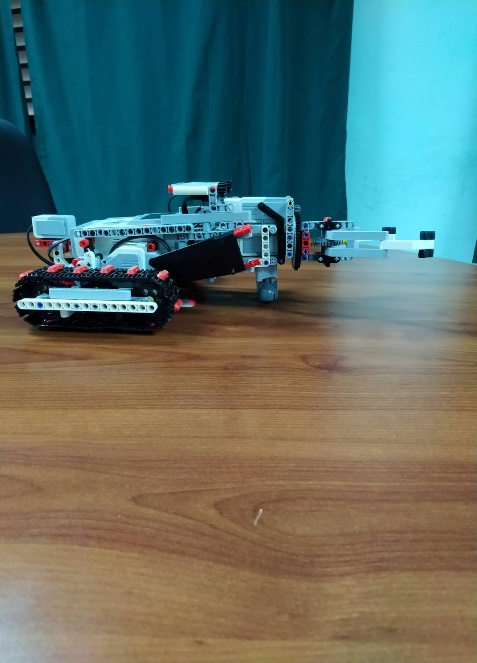
A estas alturas del proyecto, la construcción del robot está completa, incluyendo las acciones y movimientos que este puede realizar mediante la comunicación a través de la interfaz.

Además, la interfaz de usuario está en su primera versión, lo que falta es darle un aspecto más llamativo pero los botones necesarios para mover el robot están funcionando correctamente. Además, ya está completa la Wiki y la carta Gantt.

Finalmente, lo que nos falta realizar es el manual de usuario y el video.

Fotos del estado actual del robot:

****

****

## **7.2. Problemas encontrados y soluciones propuestas**

|  |  |
| --- | --- |
| **Problemas encontrados** | **Soluciones propuestas** |
| Fallo durante el flasheo en la tarjeta SD. | Utilizar otro dispositivo para formatear la tarjeta SD. |
| Conexión del robot con el computador | Cambiar la versión de Python, de la versión 3.7.0 a la versión 3.4.0 |
| Destrucción de la pinza al momento de hacer una prueba. | Reconstruirla y cambiar el código para que no cierre completamente y se vuelva a destruir. |

**7.3. Conclusiones**

Durante el desarrollo del proyecto, hemos adquirido mejores técnicas al momento de programar, pero no todo fue correcto debido a que tuvimos que modificar muchas veces el código para que funcionaran correctamente todas las funciones del robot y no ocurrieran errores tales como, cerrar en totalidad la pinza generando que se destruyera esta misma.

Cabe destacar que a medida que hemos desarrollado este proyecto se han presentado situaciones buenas como malas, pero, con paciencia y confianza en el equipo se han podido solucionar todos los problemas y hemos ganado mucha experiencia para futuros proyectos.

**7.4. Trabajo a futuro**

Como equipo nos gustaría implementar a futuro que el robot pueda identificar mediante sensores, los colores de los objetos, en este caso latas con dos colores (uno de cada equipo) logrando así que el robot pueda saber cuál objeto hay que voltear para poder ganar la competencia. Además, nos gustaría implementar que el robot al momento de identificar el objeto a voltear, pueda moverse en esa dirección, pero para lograr eso, necesitamos conocimientos más avanzados en el ámbito de la programación con sensores.

# 8. Referencias

|  |  |
| --- | --- |
| [1] | «Diseños Lego Mindstorms,» [En línea]. Available: https://www.lego.com/es-ar/themes/mindstorms/buildarobot. |
| [2] | «Referencia del juego Flip Tac Toe,» [En línea]. Available: https://www.youtube.com/watch?v=nvRmpKmmcPc. |
| [3] | «Diseño de la pinza para el robot,» [En línea]. Available: https://www.youtube.com/watch?v=VEsLKZAAoSc. |
| [4] | «Modelo de la base del robot,» [En línea]. Available: https://le-www-live-s.legocdn.com/sc/media/lessons/mindstorms-ev3/building-instructions/model-expansion-set/ev3-model-expansion-set-tank-bot-006a7f22d89c631c1d49fa27eccaf290.pdf. |
| [5] | «Programación de movimientos/pinza del robot,» [En línea]. Available: https://sites.google.com/site/ev3python/. |
| [6] | «Discord,» [En línea]. Available: https://discordapp.com/. |

# Anexo

Códigos Implementados

from tkinter import \*

from tkinter import scrolledtext

import rpyc

import os

conn = rpyc.connect('192.168.43.63', port=54321)

def test(x):

"""x: char

Envia todas las interacciones del usuario mediante el teclado o mouse al servidor del robot"""

conn.root.main(x)

def key\_press(event):

"""event: char

Lee lo presionado en el teclado para luego enviarlo a la funcion test"""

key = event.char

print(key, 'presionado')

test(key)

def key\_down(event):

"""event: char

Llama a la funcion test con "p" lo cual detiene al robot"""

test('p')

""""""

"""Creacion de la interfaz"""

root = Tk()

root.title('KL4W-T4NK3R')

root.geometry('1280x720')

photo = PhotoImage(file = "C:\\Users\\adolf\\Desktop\\FondosRobot\\faker.ppm")

fondo = Label(root, image=photo)

fondo.place(x=0, y=0, relwidth=1, relheight=1)

frame = Frame(fondo)

frame.grid(row=0, column=0)

frame.pack()

root.bind("<KeyPress>", key\_press)

root.bind("<KeyRelease>", key\_down)

buttonForward = Button(frame, text = "Forward", width = 10, command = lambda: test('w'), bd= 4)

buttonForward.grid(row = 1, column = 3, padx = 10, pady = 10)

buttonLeft = Button(frame, text = "Left", width = 10, command = lambda: test('a'), bd= 4)

buttonLeft.grid(row = 2, column = 2, padx = 10, pady = 10)

buttonBack = Button(frame, text = "Back", width = 10, command = lambda: test('s'), bd= 4)

buttonBack.grid(row = 2, column = 3, padx = 10, pady = 10)

buttonRight = Button(frame, text = "Right", width = 10, command = lambda: test('d'), bd= 4)

buttonRight.grid(row = 2, column = 4, padx = 10, pady = 10, sticky=W+E+N+S)

buttonClawLeft = Button(frame, text = "Claw Left", width = 10, command = lambda: test('q'), bd= 4)

buttonClawLeft.grid(row = 2, column = 6, padx = (50, 0), pady = 10)

buttonClawRight = Button(frame, text = "Claw Right", width = 10, command = lambda: test('e'), bd= 4)

buttonClawRight.grid(row = 2, column = 7, padx = 10, pady = 10)

buttonClawClose = Button(frame, text = "Claw Close", width = 10, command = lambda: test('k'), bd= 4)

buttonClawClose.grid(row = 1, column = 6, padx = (50, 0), pady = 10)

buttonClawOpen = Button(frame, text = "Claw Open", width = 10, command = lambda: test('j'), bd= 4)

buttonClawOpen.grid(row = 1, column = 7, padx = 10, pady = 10)

buttonTurbo = Button(frame, text = "Turbo", width = 10, command = lambda: test('n'), bd= 4)

buttonTurbo.grid(row = 1, column = 1, padx = 50, pady = 10)

buttonStop = Button(frame, text = "Stop", width = 10, command = lambda: test('p'), bd= 4)

buttonStop.grid(row = 2, column = 1, padx = 50, pady = 10)

buttonForward.bind("<Key>", lambda parar : key\_press())

help(test)

help(key\_press)

help(key\_down)

root.mainloop()

Código dentro del robot

#!/usr/bin/env python3

import rpyc

from rpyc.utils.server import ThreadedServer

from ev3dev.ev3 import \*

class MyService(rpyc.Service):

"""La clase que contiene las acciones en funciones que controlan los motores del robot

"""

motor\_left = LargeMotor('outD')

motor\_right = LargeMotor('outA')

garra = MediumMotor('outC')

giro = MediumMotor('outB')

def forward():

MyService.motor\_left.run\_forever(speed\_sp=-500)

MyService.motor\_right.run\_forever(speed\_sp=-500)

def parar():

MyService.motor\_left.run\_forever(speed\_sp=0)

MyService.motor\_right.run\_forever(speed\_sp=0)

MyService.garra.run\_forever(speed\_sp=0)

MyService.giro.run\_forever(speed\_sp=0)

def back():

MyService.motor\_left.run\_forever(speed\_sp=700)

MyService.motor\_right.run\_forever(speed\_sp=700)

def giroizq():

MyService.giro.run\_forever(speed\_sp=-1000)

MyService.garra.run\_forever(speed\_sp=150)

def giroder():

MyService.giro.run\_forever(speed\_sp=300)

def left():

MyService.motor\_left.run\_forever(speed\_sp=-400)

MyService.motor\_right.run\_forever(speed\_sp=400)

def right():

MyService.motor\_left.run\_forever(speed\_sp=400)

MyService.motor\_right.run\_forever(speed\_sp=-400)

def nitro():

MyService.motor\_left.run\_forever(speed\_sp=-1000)

MyService.motor\_right.run\_forever(speed\_sp=-1000)

def abrir():

MyService.garra.run\_forever(speed\_sp=300)

def cerrar():

MyService.garra.run\_forever(speed\_sp=-300)

def exposed\_main(self, valor):

"""valor: char

Esta funcion es el main el cual recibe un valor, dependiendo del valor accionara la funcion correspondiente"""

if valor == 'q':

MyService.giroizq()

if valor == 'e':

MyService.giroder()

if valor == 'n':

MyService.forward()

if valor == 'p':

MyService.parar()

if valor == 's':

MyService.back()

if valor == 'a':

MyService.left()

if valor == 'd':

MyService.right()

if valor == 'j':

MyService.abrir()

if valor == 'k':

MyService.cerrar()

if valor == 'w':

MyService.nitro()

else:

pass

if \_\_name\_\_ == '\_\_main\_\_':

s = ThreadedServer(MyService, port=54321)

s.start()

help(MyService)

help(exposed\_main)