**UNIVERSIDAD DE TARAPACÁ**

**FACULTAD DE INGENIERÍA**

**DEPARTAMENTO DE INGENIERÍA CIVIL EN COMPUTACIÓN E INFORMÁTICA**



**Parte l**

**“Gorilla-Tank MK ll”**

**Alumno(os): Luis Huanca Calle**

 **Vranika Santiago Yovich**

 **Gabriel Pailamilla Peréz**

 **Brayan García Arancibia**

 **Luciano Vera Norambuena**

**Asignatura: Proyecto l**

**Profesor: Humberto Urrutia López**

**SEPTIEMBRE – 2022**

**Historial de Cambios**

|  |  |  |  |
| --- | --- | --- | --- |
| **Fecha** | **Versión** | **Descripción** | **Autor(es)** |
| 04/10/2021 | 1.0 | Formulación del Proyecto | **Vranika Santiago** |
| 02/11/2021 | 1.1 | ﻿﻿Desarrollo del plan de avance | **Vranika Santiago****Gabriel Pailamilla** |

Tabla 1: Historial de cambios.

Tabla de Contenidos

[1. Índice de tablas 5](#_Toc118482312)

[2. Índice de figuras 5](#_Toc118482313)

[3. Panorama General 6](#_Toc118482314)

[3.1. Introducción 6](#_Toc118482315)

[3.2. Objetivos 6](#_Toc118482316)

[3.2.1. Objetivo General 6](#_Toc118482317)

[3.2.2. Objetivos Específicos 6](#_Toc118482318)

[3.3. Restricciones 6](#_Toc118482319)

[3.4. Entregables 7](#_Toc118482320)

[4. Organización del Personal 8](#_Toc118482321)

[4.1. Descripción de los Roles 8](#_Toc118482322)

[4.2. Personal que cumplirá los Roles 8](#_Toc118482323)

[4.3. Mecanismos de Comunicación 8](#_Toc118482324)

[5. Planificación del Proyecto 9](#_Toc118482325)

[5.1. Actividades 9](#_Toc118482326)

[5.2. Asignación de Tiempo 12](#_Toc118482327)

[5.3. Gestión de Riesgos 12](#_Toc118482328)

[6. Planificación de los Recursos 14](#_Toc118482329)

[6.1. Hardware 14](#_Toc118482330)

[6.2. Software 14](#_Toc118482331)

[6.3. Estimación de Costos 14](#_Toc118482332)

[6.3.1. Costos de Hardware 14](#_Toc118482333)

[6.3.2. Costos de Software 15](#_Toc118482334)

[6.3.3. Costos de Gestión 15](#_Toc118482335)

[6.3.4. Costos totales del proyecto 16](#_Toc118482336)

[7. Análisis y Diseño 17](#_Toc118482337)

[7.1. Especificación de Requerimiento 17](#_Toc118482338)

[7.2. Arquitectura 17](#_Toc118482339)

[7.3. Interfaz 18](#_Toc118482340)

[8. Implementación 20](#_Toc118482341)

[8.1. Fundamentos de Proyectiles 20](#_Toc118482342)

[8.2. Descripción de los programas 21](#_Toc118482343)

[8.3. Diagramas 25](#_Toc118482344)

[9. Resultados 27](#_Toc118482345)

[9.1. Estado Actual del Proyecto 27](#_Toc118482346)

[9.2. Problemas Encontrados y Solución Propuesta 27](#_Toc118482347)

[10. Conclusión 28](#_Toc118482348)

[11. Referencias 29](#_Toc118482349)

# Índice de tablas

[Tabla 1: Historial de cambios. 2](#_Toc118481038)

[Tabla 2: Roles. 8](#_Toc118481039)

[Tabla 3: Actividades. 11](#_Toc118481040)

[Tabla 4: Gestión de riesgos. 13](#_Toc118481041)

[Tabla 5: Costos de Hardware. 14](#_Toc118481042)

[Tabla 6: Costos de Software. 15](#_Toc118481043)

[Tabla 7: Costos de Gestión. 15](#_Toc118481044)

[Tabla 8: Costos totales. 16](#_Toc118481045)

# Índice de figuras

[Ilustración 1: Carta Gantt. 12](file:////Users/vrasantiago/Documentos/Proyecto%20l/Informe%20ll.docx#_Toc118481046)

[Ilustración 2: Arquitectura. 17](file:////Users/vrasantiago/Documentos/Proyecto%20l/Informe%20ll.docx#_Toc118481047)

[Ilustración 3: Interfaz. 18](#_Toc118481048)

[Ilustración 4: Movimiento parabólico. 20](#_Toc118481049)

[Ilustración 5: Diagrama l. 25](file:////Users/vrasantiago/Documentos/Proyecto%20l/Informe%20ll.docx#_Toc118481050)

[Ilustración 6: Diagrama ll. 26](file:////Users/vrasantiago/Documentos/Proyecto%20l/Informe%20ll.docx#_Toc118481051)

# Panorama General

## Introducción

Ole Kirk Christiansen nacido en Dinamarca, creó la compañía llamada “LEGO” (proveniente del danés *leg godt*, es decir, “jugar bien” en español) en 1932. Ésta se encargaba de producir juguetes de madera para infantes. Sin embargo, no fue hasta 1949 que comenzaron a fabricarlos en plástico, donde se encontraba su producto más famoso: los cubos entrelazados, el cual este emprendedor danés los apodada como “ladrillos de unión automática”, debido a que estos podían ser construidos y luego, ser des-construidos por la persona sin ningún inconveniente.

El fundador de esta famosa compañía jamás imaginó el impacto que tendría en el ámbito de la robótica. Verbigracia, en 2013 se lanzó el pack educativo LEGO® MINDSTORMS® Education EV3, que se caracteriza por su pequeña computadora programable (EV3) y que tiene como objetivo principal, controlar motores y recopilar información de sensores.

El proyecto “Gorilla-Tank MK ll” creado para la asignatura Proyectos l, fue armado con piezas del kit nombrado en el párrafo anterior. Por consiguiente, en el presente documento se expondrá el proyecto tecnológico trabajado en clases.

Cabe señalar que, para este trabajo, se usarán tanto los conocimientos que se han adquirido en la formación de los estudiantes de la carrera, como también aplicando la materia de Taller de Programación l y ll.

## Objetivos

### Objetivo General

El objetivo general de la actividad es desarrollar y construir un robot EV3 que tiene la funcionalidad de disparar proyectiles a través de una interfaz gráfica programada en Python.

### Objetivos Específicos

* Estudiar la librería Tkinter que ofrece Python para hacer interfaces gráficas de usuarios.
* Crear una interfaz gráfica de usuario que esté capacitada para enviar instrucciones remotamente hacia el robot EV3.
* Aplicar los contenidos vistos el semestre pasado en la asignatura Mecánica clásica acerca del movimiento de proyectiles y luego hacer predicciones en base a las fórmulas dadas.

## Restricciones

Existe un conjunto de limitaciones para que el desarrollo del robot sea exitoso, las cuales se pueden observar en la siguiente tabla:

* **Tiempo:** fecha límite establecida para hacer el proyecto.
* **Set LEGO EV3:** Utilización de kit LEGO MINDSTORM EV3.
* **Temática:** debe tener un cañón que realice lanzamiento de proyectiles.
* **Control:** el robot debe ser controlado vía remota.
* **Plataforma de documentación:** todos los archivos redactados, serán subidos a Redmine.

## Entregables

* Informes y presentaciones:
* Formulación del proyecto.
* Avance del proyecto l.
* Informe y presentación final.
* Bitácora semanal.
* Wiki y manual de usuario.
* Producto final.

# Organización del Personal

A cada integrante del grupo se le designó una responsabilidad, donde estarán a cargo de cumplir en el tiempo estimado. Cabe recalcar que por mucho que existan encargados para cada eje, todos los integrantes deben velar por la realización tanto de sus tareas como la de sus compañeros.

## Descripción de los Roles

* **Jefe de grupo:** encargado de representar al equipo de trabajo, de la organización y de la toma de decisiones.
* **Programador:** encargado de desarrollar e implementar el código en Python logrando así que el robot pueda ejecutar las acciones solicitadas.
* **Ensamblador:** encargado de diseñar y armar el robot de tal manera que pueda moverse en todas las direcciones y lance un proyectil.
* **Diseñador:** encargado de la estética de la interfaz gráfica.
* **Documentador:** encargado de realizar los informes, presentaciones, bitácoras, video, manual de usuario y wiki del proyecto.

## Personal que cumplirá los Roles

|  |  |  |
| --- | --- | --- |
| **Rol** | **Responsable** | **Involucrados** |
| Jefe de grupo | Vranika Santiago | Vranika Santiago |
| Programador | Luis Huanca | Luis Huanca y Brayan García |
| Ensamblador | Gabriel Pailamilla | Gabriel Pailamilla y Luciano Vera |
| Diseñador | Brayan García | Brayan García y Luciano Vera |
| Documentador | Vranika Santiago | Vranika Santiago y Luciano Vera |

Tabla 2: Roles.

## Mecanismos de Comunicación

Para tener una mejor organización, se creó un grupo de Whatsapp donde podremos establecer y avisar los horarios de las reuniones. También sirve para dejar registro de nuestra asistencia por día y para mandar archivos (informes, enlaces, vídeos, fotos, entre otros). Allende a lo anterior, se utilizó la plataforma social Discord, en la cual cada vez que exista una propuesta para mejorar en el proyecto, sea discutida y llegar a un consenso. Por otro lado, el uso de Google Drive tiene como función principal respaldar cada archivo redactado y además, los demás integrantes pueden tener una segunda vía de conocimiento acerca de los archivos subido en Redmine.

Cabe agregar que, la comunicación con el docente a cargo será por medio del sistema oficial de la UTA; Intranet o por vía email.

# Planificación del Proyecto

## Actividades

|  |  |  |
| --- | --- | --- |
| **Actividad** | **Descripción** | **Responsable** |
| Redacción de bitácoras | Registro de todas las actividades que se desarrollan semanalmente. | Vranika Santiago |
| Vídeos y fotos | Registro visual de los avances del proyecto. | Luciano Vera |
|  Redacción de Carta Gantt | Planificación de las actividades a lo largo del semestre. | Vranika Santiago |
| Wiki | Se capturan y comparten ideas e información del proyecto. | Vranika Santiago |
| Organización | Designación de la actividad que estará encargado cada integrante. | Vranika Santiago |
| Contabilizar piezas | Entrega del kit EV3 donde luego se contaron las piezas que habían. | Gabriel Pailamilla  |
| Búsqueda de ideas | Indagación de ideas que se podrían llevar a cabo | Luciano Vera |
| Construcción del robot | Armado de la base del robot y el ángulo de inclinación en él. | Gabriel Pailamilla |
| Instalar SO en la tarjeta SD | - | Luis Huanca |
| Administrar Redmine | Conocer la forma en la que se usa, sus funcionalidades y, además, sus características. | Vranika Santiago |
| Informe l | Crear informe para la parte l: formulación del proyecto. | Vranika Santiago |
| Estimación de costos | Calcular el presupuesto del proyecto.  | Luciano Vera |
| Presentación l | Elaborar presentación de la formulación del proyecto y luego exponerlo. | Vranika Santiago |
| Estudiar y realizar pruebas en Python | Estudio del lenguaje de programación Python y de la librería ev3dev-lang-python. | Luis Huanca |
| Análisis del esqueleto del robot | Analizar y revisar la estructura del robot para ver si cabe la posibilidad de alguna modificación. | Gabriel Pailamilla |
| Estudiar bibliotecas | Estudiar las diferencias entre EV3DEV-Lang y microypthon. | Luis Huanca |
| Programación de los movimientos | Aplicar los conocimientos estudiados anteriormente en Python | Luis Huanca |
| Crear interfaz gráfica | Aplicar los conocimientos estudiados anteriormente en las librerías Tkinter. | Brayan García |
| Realizar predicciones con respecto al lanzamiento de proyectil | Estudiar sobre lanzamiento de proyectiles para luego, realizar predicciones. | Gabriel Pailamilla |
| Implementar interfaz gráfica en el servidor | El control del robot podrá ser de manera remota. | Brayan García |
| Informe ll  | Redacción del informe del avance del proyecto. | Vranika Santiago |
| Presentación ll | Creación del material de apoyo sobre el avance del proyecto. | Luciano Vera |
| Depuración código | Proceso en el cual se identificarán y corregirán errores en el algoritmo. | Luis Huanca |
| Pruebas de funcionamiento | Chequear que el robot este cumpliendo su propósito. | Brayan García |
| Implementar efectos sonoros | - | Luciano Vera |
| Documentación del código | Añadir información para explicar lo que hace cada parte del código. | Luis Huanca |
| Informe final | Redacción del informe final. | Vranika Santiago |
| Presentación final | Creación del material para la presentación final. | Luciano Vera. |

Tabla 3: Actividades.

## Asignación de Tiempo

La Carta Gantt fue creada con el propósito de planificar y gestionar de manera más eficaz nuestras actividades a lo largo del semestre. Además, se puede registrar el tiempo dedicado para cada tarea y compararlo con el tiempo estimado.

Ilustración 1: Carta Gantt.

## Gestión de Riesgos

**Niveles de impacto:**

1. Catastrófico
2. Crítico
3. Circunstancial
4. Irrelevante

|  |  |  |  |
| --- | --- | --- | --- |
| **Riesgos** | **Probabilidad de concurrencia** | **Nivel de impacto** | **Acción remedial** |
| Desarme del robot a causa de una caída. | 40% | 2 | Volver a construir el robot. |
| Daño o pérdida de la tarjeta SD.  | 10% | 1 | Comprar otra tarjeta SD y respaldar la información. |
| Escasez de piezas. | 80% | 3 | Ir a buscar las piezas faltantes o reemplazarla por una similar. |
| Enfermedad o inconveniente del personal. | 40% | 3 | Un integrante del grupo se encargará temporalmente de las tareas del personal faltante. |
| Reconstrucción total del robot por incompatibilidad de lo requerido en el proyecto. | 25% | 1 | Buscar nuevas ideas acorde a lo pedido y llevarlas a cabo. |
| Incumplimiento de tareas. | 20% | 2 | Reorganizar los roles del grupo para asignar al más optimo y establecer fechas límites de trabajo. |
| Pocas horas autónomas. | 10% | 2 | Mejorar la comunicación y motivar a los integrantes del grupo buscando horarios disponibles para todos. |
| Descarga de batería del EV3. | 35% | 4 | Cargar la batería, mientras el EV3 no sea utilizado. |

Tabla 4: Gestión de riesgos.

# Planificación de los Recursos

## Hardware

* Notebook Msi modern b5
* Samsung Galaxy s6 10.4” 64 gb
* Macbook pro M1 13” 256gb
* Macbook pro M1 13” 512gb
* iPad 8va generación
* Kit Lego MINDSTORMS (EV3)
* Micro SD (8 GB)
* Dongle USB Wifi
* Router

## Software

* Software de desarrollo Phyton
* Visual Studio Code
* Licencia de Microsoft Office
* Licencia de Windows 11
* Gmail
* Whatsapp
* Discord

## Estimación de Costos

### Costos de Hardware

|  |  |  |  |
| --- | --- | --- | --- |
| **Productos** | **Cantidad** | **Coste** | **Coste total** |
| Notebook Msi modern b5 | 1 unidad | $649.990 | $649.990 |
| Samsung Galaxy s6 10.4” 64 gb | 1 unidad | $379.990 | $379.990 |
| Macbook pro M1 13” 256gb | 1 unidad | $1.449.990 | $1.449.990 |
| Macbook pro M1 13” 512gb | 1 unidad | $1.649.990 | $1.649.990 |
| iPad 8va generación | 1 unidad | $399.990 | $399.990 |
| Kit Lego MINDSTORMS (EV3) | 1 unidad | $450.000 | $450.000 |
| Micro SD (8 GB) | 1 unidad | $5.000 | $5.000 |
| Dongle USB Wifi | 1 unidad | $7.000 | $7.000 |

Tabla 5: Costos de Hardware.

|  |  |  |  |
| --- | --- | --- | --- |
| Router | 1 unidad | $15.000 | $15.000 |

|  |  |
| --- | --- |
| Subtotal | $5.006.950 |

### Costos de Software

|  |  |  |  |
| --- | --- | --- | --- |
| **Productos** | **Cantidad** | **Coste** | **Coste total** |
| Software de desarrollo Phyton | 1 unidad | Gratuito | $0 |
| Visual Studio Code | 1 unidad | Gratuito | $0 |
| Licencia de Microsoft Office | 1 unidad | $30.000 | $30.000 |
| Licencia de Windows 11 | 1 unidad | $10.000 | $10.000 |
| Gmail | 1 unidad | Gratuito | $0 |
| Discord | 1 unidad | Gratuito | $0 |
| Whatsapp | 1 unidad | Gratuito | $0 |

|  |  |
| --- | --- |
| Subtotal | $40.000 |

Tabla 6: Costos de Software.

### Costos de Gestión

* Según la planificación entregada por el docente, son 154 horas totales (72 presenciales y 82 autónomas). En total son 16 semanas de trabajo y al dividir las horas totales por las semanas de trabajo, aproximándolo, se debería trabajar 9 horas por cada semana.
* Para calcular el valor total de cada encargado, se debe multiplicar el valor mensual por las 16 semanas de trabajo, quedando de esta forma:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Encargado** | **Valor hora** | **Horas semanales** | **Valor mensual** | **Valor total** |
| Horas trabajadas PROGRAMADOR | $7.000 | 9 | $63.000 | $1.008.000 |
| Horas trabajadasENSAMBLADOR | $5.000 | 9 | $45.000 | $720.000 |
| Horas trabajadas JEFE DE GRUPO | $6.000 | 9 | $54.000 | $864.000 |
| Horas trabajadas DOCUMENTADOR | $4.000 | 9 | $36.000 | $576.000 |
| Horas trabajadas DISEÑADOR | $4.000 | 9 | $36.000 | $576.000 |

|  |  |
| --- | --- |
| Subtotal | $3.744.000 |

Tabla 7: Costos de Gestión.

### Costos totales del proyecto

|  |  |
| --- | --- |
| **Recurso** | **Costo total** |
| Hardware | $5.006.950 |
| Software | $40.000 |
| Gestión | $3.744.000 |
| Total proyecto | $8.790.950 |

Tabla 8: Costos totales.

En resumen, la estimación del proyecto tendrá un presupuesto de $$8.790.950.

# Análisis y Diseño

## Especificación de Requerimiento

**Requerimientos funcionales:**

* El robot debe tener la capacidad de moverse.
* El robot debe tener la capacidad de disparar proyectiles.
* El robot deberá ser controlado por el usuario a través de una interfaz gráfica en un dispositivo tecnológico, tal como un notebook.

**Requerimientos no funcionales:**

* La interfaz gráfica contendrá los movimientos necesarios para poder avanzar, retroceder, girar la izquierda y girar a la derecha. Además, disparar proyectiles.
* La programación de cada código debe estar escrita en Python.
* El robot debe ser capaz de disparar un elástico (en nuestro caso, es el tipo de proyectil que será usado).
* El robot debe ser construido con piezas del kit Lego Mindstorm EV3 Education.
* La estructura del robot debe ser estable y así mismo, sus movimientos.
* La interfaz gráfica debe poseer un diseño cómodo y fácil de usar para el usuario.
* El software para que el robot pueda ser controlado podrá ser utilizado en los sistemas operativos Windows, Linux y macOS.

## Arquitectura



Ilustración 2: Arquitectura.

1. El robot y el notebook deben estar conectados a la misma red Wifi.
2. Iniciar el archivo **Server.py** para así obtener la conexión remota con el notebook.
3. El usuario debe iniciar la interfaz gráfica para controlar el robot a distancia.
4. La interfaz se conectará al servidor del robot y el usuario podrá controlarlo.
5. Robot que realizará los movimientos enviados por el usuario gracias al servidor establecido entre sí mismo y el notebook del usuario.

## Interfaz



Ilustración 3: Interfaz.

El usuario dispondrá de una interfaz, como la que se puede apreciar en la imagen (Ilustración 3), la cual será la pantalla principal y en ella se encontrarán:

* Flechas para indicar las direcciones en las que el robot se desplazará, es decir, ir para adelante, para atrás, girar a la izquierda y girar a la derecha.
* Botón “Ingresar” que tiene como función ingresar el ángulo del cañón, en el cual en la parte de arriba se puede encontrar el símbolo de un ángulo.
* Botón “Disparar” que logra disparar el proyectil.
* Botón “Conectar” el cual vincula al cliente a través de la IP del servidor que está en el EV3.
* Logo característico del robot construido junto a su nombre.
* Botón con el símbolo de on – off que permite salir del programa, en cualquier momento.

# Implementación

## Fundamentos de Proyectiles

Tenemos conocimiento que el proyectil (en este caso, un elástico) que está siendo lanzado por un cañón, realiza un movimiento parabólico de caída libre. A continuación, se utilizarán las fórmulas correspondientes para calcular el tiempo de vuelo del proyectil. Cabe recalcar, que para este ejercicio, la aceleración de gravedad tiene un valor aproximado de 10 m/s, la velocidad inicial ( es 8/m y forma un ángulo de 35°.



tb

ts

Ilustración 4: Movimiento parabólico.

Sabemos que , por lo que los datos hasta el momento son:

Usando la fórmula:

 10t = 50 t = 5s

Ahora sabemos que el tiempo de subida es el mismo que el tiempo de bajada, por lo que sería 5s. Entonces para calcular el tiempo de vuelo, tenemos que sumar el tiempo de subida con el tiempo de bajada obteniendo 10s.

## Descripción de los programas

* **Funciones:**

from ev3dev2.motor import LargeMotor, MediumMotor, OUTPUT\_A, OUTPUT\_B, OUTPUT\_C,

OUTPUT\_D, SpeedPercent, MoveTank

from ev3dev2.sensor import INPUT\_1

from ev3dev2.sensor .lego import TouchSensor

from ev3dev2.led import Leds

motor\_angle= LargeMotor(OUTPUT\_A)

cannon = MediumMotor(OUTPUT\_D)

tankmoves = MoveTank(OUTPUT\_B,OUTPUT\_C)

def moveUp():

 print("Moving up...")

 tankmoves.on(SpeedPercent(100),SpeedPercent(100))

def moveDown():

 print("Moving down...")

 tankmoves.on(SpeedPercent(-100),SpeedPercent(-100))

def moveRight():

 print("Moving right...")

 tankmoves.on(SpeedPercent(100),SpeedPercent(-100))

def moveLeft():

 print("Moving left...")

 tankmoves.on(SpeedPercent(-100),SpeedPercent(100))

def Inclinacion0():

 motor\_angle.on\_for\_degrees(10,55,brake=True)

def Inclinacion45():

 motor\_angle.on\_for\_degrees(10,-55,brake=True)

def Disparar():

 cannon.on(SpeedPercent(100))

def Stop():

 tankmoves.stop()

 cannon.stop()

import socket

from Function import \*

s = socket.socket()

print("Socket creado")

port = 19999

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))

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 == 'p'):

 Disparar()

 if (char == 'i'):

 Inclinacion45()

 if (char == 'o'):

 Inclinacion0()

 if (char=='u'):

 Inclinacion90()

 if (char == ' '):

 Stop()

* **Cliente:**

import tkinter as tk

from tkinter import ttk

from tkinter import \*

import socket

#Ventana

ventana = tk.Tk()

ventana.config(width=900, height=700)

ventana.title("Interfaz -> Gorilla-Tank")

ventana.configure(bg="gray25")

#FUNCIONES

def Arriba(event= None):

 clientSocket.send(bytes([ord('w')]))

def Abajo(event=None):

 clientSocket.send(bytes([ord('s')]))

def Izquierda(event=None):

 clientSocket.send(bytes([ord('a')]))

def Derecha(event=None):

 clientSocket.send(bytes([ord('d')]))

def Disparar(event=None):

 clientSocket.send(bytes([ord('p')]))

def Angulo():

 if(opcion.get()=="45"):

 clientSocket.send(bytes([ord('i')]))

 elif(opcion.get() == "0"):

 clientSocket.send(bytes([ord('o')]))

def on\_realese(event):

 print("click realese")

 clientSocket.send(bytes([ord(' ')]))

def getAddres():

 w\_ip = Tk()

 w\_ip.geometry("300x100")

 dato = StringVar(w\_ip)

 w\_ip.title("Configurar Ip")

 ip = ttk.Entry(w\_ip,textvariable=dato).place(x=10,y=10)

 button = Button(w\_ip,text =" Aplicar",command=lambda:[conectar(dato.get()),w\_ip.destroy()]).place(x=170,y=9)

 print(dato.get())

def conectar(adress):

 port = 19999

 try:

 clientSocket.connect((adress,port))

 messagebox.showinfo("Mensaje Servido","Cliente conectado al robot: {0} : {1}".format(adress,port))

 except socket.error:

 messagebox.showwarning("Conexión erronea","No se ha logrado al conexíon, verifique la Ip {0}".format(adress))

 getAddres()

 clientSocket.close()

#ETIQUETAS

img2 = tk.PhotoImage(file="gorilla.png")

lnl\_img2 = tk.Label(ventana,image = img2, bg = "gray25").place(x=80,y=0)

#Angulo

img3 = tk.PhotoImage(file="Angulo.png")

lnl\_img3 = tk.Label(ventana,image = img3,bg = "gray25").place(x=600,y=100)

opcion=StringVar()

angulo = Spinbox(ventana,state="readonly",values=("0","45"),textvariable=opcion).place(x=600,y=270)

boton\_angulo = Button(ventana, text="Ingresar", command=Angulo,background = "gray25",fg="white",font=("Arial Black",10)).place(x=600, y=300)

#Botones

#Boton Detener

img4 = tk.PhotoImage(file="ONOFF.png")

boton\_salir = Button(ventana,image=img4, command=ventana.destroy,bg = "gray25").place(x=600, y=440)

#Boton Arriba

img\_boton\_arri = tk.PhotoImage(file="Arriba.png")

boton\_arriba = tk.Button(repeatdelay=50,repeatinterval=50,image=img\_boton\_arri,command=Arriba, background = "gray25")

boton\_arriba.place(x=170, y=340)

boton\_arriba.bind('<ButtonRelease-1>',on\_realese)

#Boton Abajo

img\_boton\_abaj = tk.PhotoImage(file="Abajo.png")

boton\_abajo = tk.Button(repeatdelay=50,repeatinterval=50,image=img\_boton\_abaj,command=Abajo, background = "gray25")

boton\_abajo.place(x=170, y=450)

boton\_abajo.bind('<ButtonRelease-1>',on\_realese)

#Boton Izquierda

img\_boton\_izq = tk.PhotoImage(file="Izquierda.png")

boton\_izq = tk.Button(repeatdelay=50,repeatinterval=50,image=img\_boton\_izq,command=Izquierda, background = "gray25")

boton\_izq.place(x=60, y=400)

boton\_izq.bind('<ButtonRelease-1>',on\_realese)

#Boton Derecha

img\_boton\_derech = tk.PhotoImage(file="Derecha.png")

boton\_derech = tk.Button(repeatdelay=50,repeatinterval=50,image=img\_boton\_derech,command=Derecha, background = "gray25")

boton\_derech.place(x=280, y=400)

boton\_derech.bind('<ButtonRelease-1>',on\_realese)

#Boton Disparar

boton\_disparar = tk.Button(repeatdelay =50,repeatinterval = 50,text="Disparar",command=Disparar, background = "gray25",fg="white",font=("Arial Black",10))

boton\_disparar.place(x=175,y=600)

boton\_disparar.bind('<ButtonRelease-1>',on\_realese)

#Boton Conectar

button\_connect = tk.Button(ventana,text = "Conectar",command=getAddres,bg = "gray25",fg="white",font=("Arial Black",10)).place(x=600,y=600)

clientSocket = socket.socket()

port = 1999

ipAddress = "192.168.221.160"

ventana.mainloop()

## Diagramas

TankMoves(B,C)

Ilustración 5: Diagrama l.

MoveUp

Movedown

moveleft

moveright

Ilustración 6: Diagrama ll.

Motor\_Angle(A)

Inclinacion45

Inclinacion0

Cannon(D)

Disparar

# Resultados

## Estado Actual del Proyecto

En estos momentos, el proyecto cuenta con:

* La versión finalizada del robot “Gorilla Tank-MK ll”, es decir, el cuerpo del robot ya está construido en su totalidad.
* Las funciones de movimiento. Esto significa que, el robot ya se desplaza en distintas direcciones y a su vez, dispara proyectiles.
* La programación de la interfaz gráfica. Para la manipulación de la interfaz se utilizó la librería “Tkinter”, la cual permite que el cliente ejecute las funciones desde un dispositivo que pueda operar con Python. Actualmente, es completamente funcional.
* La creación del servidor. Para esto se utilizó la librería de Python “socket”, la cual habilita la conexión entre el host y el cliente, Este se encuentra programado dentro el EV3DEV con el nombre de Server.py. Está terminado y ya es completamente funcional.
* La conexión vía remota.
* La wiki del proyecto.
* Carta Gantt actualizada.
* Bitácoras, informes y presentaciones concretadas.

## Problemas Encontrados y Solución Propuesta

|  |  |
| --- | --- |
| **Problemas encontrados** | **Soluciones** |
| Falló de la micro SD durante la fase de pruebas. | Se cambió la micro SD y formatearla. |
| El ángulo de inclinación no respondía de manera correcta a la hora de enviarle instrucciones. | Para arreglar este error, se chequeó el código para ver si había que reforzar la función del ángulo.  |
| Para insertar el ángulo en el cañón, fue complejo, debido a que era inestable. | Se modificaron las fuerzas del motor y se agregaron piezas para reforzar.  |
| Los tiempos en la Carta Gantt no eran realistas, por lo que muchas de las actividades no fueron realizadas cuando correspondían. | Se reajustó el tiempo teniendo el cuenta las horas trabajadas en clases, autónomas y los días feriados. |
| La red wifi del departamento es deficiente, siendo este un problema al momento de trabajar con Putty. | Se utilizó una red propia del equipo compartida por uno de sus integrantes, usando su plan de datos.  |
| Conocimientos débiles para las predicciones del lanzamiento de proyectiles. | Retroalimentación de materia vista en el semestre pasado en Mecánica Clásica y utilizar herramientas para medir la distancia y la velocidad de forma exacta. |

Tabla 8: Problemas encontrados y solución propuesta.

# Conclusión

A lo largo del proyecto hemos tenido dificultades, principalmente en el diseño final del robot, debido a que antes de llegar a construir “Gorilla Tank-MK ll”, se armaron 2 modelos totalmente distintos estéticamente. No obstante, el día de hoy contamos con un diseño estable, en el cual podemos programar para que tenga la capacidad de moverse en distintas direcciones controladas por el usuario. También agregar que dispara proyectiles y asu vez, cambia el ángulo de inclinación del proyectil. Todo esto es posible gracias a nuestros conocimientos adquiridos en Python y que fueron trabajados arduamente en clases y en el hogar.

Por otro lado, pese a todos los inconvenientes que se nos han presentado tanto de manera personal como grupal, hemos aprendido a sobrellevar cada obstáculo y trabajar en equipo de manera adecuada y respetuosa. Cabe destacar que, en este tiempo, ha existido una mejora en nuestra capacidad de organización, la cual nos será útil en los próximos proyectos.

# Referencias

[1] *La historia de LEGO | ROBOTIX Hands-on Learning*. (s. f.). Blog de ROBOTIX. <https://www.robotix.es/blog/historia-lego/>

[2] (s. f.). Welcome to Joliet Public Schools | Joliet School District 86. <https://www.joliet86.org/assets/1/6/Historia_de_Legos.pdf>

[3] *LEGO*. (s. f.). Lego Enciclopedia. <https://lego.fandom.com/es/wiki/LEGO>

[4] *LEGO Mindstorms EV3. Historia, detalles y precios*. (s. f.). WWWhat's new. <https://wwwhatsnew.com/2017/03/12/lego-mindstorms-ev3-historia-detalles-y-precios/>