

UNIVERSIDAD DE TARAPACÁ



FACULTAD DE INGENIERÍA

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



**Plan de Proyecto
“McQueen”
“robot de transporte de materiales”**

**Alumnos:
Renato chacon
Bruno Rojas
Milton Porlles**

**Profesor:
Baris Klobertanz**

| | |
|---|-----------|
| 1. Planteamiento del problema y objetivos | 3 |
| 1.2.1 Objetivo general | 4 |
| 1.2.2 Objetivos específicos | 5 |
| 2. Organización del Personal | 5 |
| 2.1 Descripción de los roles definidos | 5 |
| 2.2 Asignación de roles | 6 |
| 2.3 Canales de comunicación | 6 |
| 3. Planificación del Proyecto | 6 |
| 3.1 Actividades definidas | 6 |
| 3.2 Carta Gantt | 7 |
| 3.3 Gestión de Riesgos | 8 |
| 4. Identificación de los recursos y costos asociados | 9 |
| 4.1 Hardware | 9 |
| 4.2 Software | 9 |
| 4.3 Recursos humanos | 10 |
| Notebook Gamer Asus TUF Gaming A15 Ryzen 7 8GB RAM 512GB SSD 15,6" NVIDIA RTX 2050 | 10 |
| 5. Análisis y diseño | 12 |
| 5.1 Especificación de requerimientos | 12 |
| 5.1.1 Requerimientos funcionales | 12 |
| 5.1.2 Alcance incluido | 12 |
| 5.1.3 Fuera del alcance | 13 |
| 5.1.2 Requerimientos no funcionales | 13 |
| 5.2 Arquitectura del software | 13 |
| 5.3 Diseño inicial de la interfaz gráfica de usuario (GUI) | 14 |
| 6. Implementación | 14 |
| 7. Resultados | 15 |
| 7.1 Estado actual del proyecto | 15 |
| 7.2 Problemas encontrados y solucionados | 15 |
| 8. Conclusión | 16 |
| 9. Referencias | 17 |

1. Planteamiento del problema y objetivos

1.1 Problema

El proyecto consiste en la simulación de la fase de transporte de minerales de un proceso minero, para esto se construirá y programará un camión transportador utilizando LEGO Spike, con el objetivo de mejorar la seguridad para los trabajadores. Se utilizarán piezas mecánicas con sensores y motores, creando un vehículo funcional que pueda desplazarse hacia adelante, retroceder y girar con precisión.

Además, se desarrollará un programa que permita manejar el camión de manera remota, aprovechando las herramientas de LEGO Spike. En este proyecto se busca aplicar conocimientos de robótica, programación y trabajo en equipo, mientras se diseña un sistema práctico y eficiente que simula un vehículo real en miniatura.

1.2 Objetivos

1.2.1 Objetivo general

Construir un robot de transporte minero el cual transportará material de construcción hacia un punto estimado, el robot se usará en un lugar plano con distintos obstáculos simulando una pista minera y programar un robot de lego spike para que sea capaz de moverse a través de una interfaz gráfica programada la cual hace la simulación de control remoto.

1.2.2 Objetivos específicos

- Aprender y dar uso del set de Lego Spike para la creación de un vehículo minero, pasando por distintos prototipos para seleccionar el diseño más óptimo y darlo como producto final mediante los prototipos de vehículos creados.
- Construir un vehículo con el set Lego Spike el cual pueda transportar una pieza de Lego de un punto a otro.
- Desarrollar y codificar las secuencias de control remoto en el programa lego Spike para permitir que el vehículo minero realice las maniobras de movimiento.
- Programar mediante Python una interfaz gráfica que permita la movilidad del robot la cual hacia adelante, atrás y los lados.
- Lograr que el robot Spike cumpla con los obstáculos de la pista, la cual nos permitirá ver la firmeza del robot y la programación de este .

2. Organización del Personal

2.1 Descripción de los roles definidos

Jefe de proyecto: La persona que representa al equipo, revisa bitácoras, Redmine, entre otros, y organiza el proyecto.

Programador: Es la persona encargada de la planificación del código para que el robot pueda cumplir las funciones pedidas por el profesor.

Documentador: Encargado de la realización de informes, bitácoras, redmine, etc.

2.2 Asignación de roles

Responsable del rol: Persona encargada de la realización del trabajo propuesto por el grupo

Involucrados: Personas secundarias las cuales ayudarán al responsable para la realización de su trabajo propuesto

| Rol | Responsable | Involucrados |
|------------------|----------------|-----------------------------|
| Jefe de proyecto | Bruno Rojas | Bruno Rojas |
| Programador | Renato Chacón | Renato Chacón |
| Documentador | Milton Porlles | Milton Porlles, Bruno Rojas |

2.3 Canales de comunicación

Los métodos que se utilizaron fueron los siguientes:

Whatsapp: Este método sirvió para ver y hablar cosas rápidas, ya sea alguna modificación de último minuto o proponer alguna reunión.

Discord: Método que se utiliza para realizar modificaciones y hacer llamadas en grupo o simplemente para llevar a cabo las actividades del equipo, también para resolver dudas.

Clases: Método tradicional para hacer reuniones y donde más se nota avance ya que se logra que el grupo completo este en persona

3. Planificación del Proyecto

3.1 Actividades definidas

| Nombre | Descripción | Responsable | Resultado |
|--|--|---------------------------------|---|
| Planificación del Proyecto | Se realiza la investigación sobre lego spike | Todo el grupo | Lograr tener conocimiento previo antes de trabajar |
| Investigación sobre software para uso | Búsqueda sobre software de lego spike es el mejor para la programación del robot | Bruno Rojas Renato Chacon | Tener el mejor software a nuestro gusto que nos deje hacer lo que queramos respecto a modificaciones y nuevos códigos |
| Organización del proyecto | Dividir roles, saber quien va a hacer cada cosa y determinar un jefe de grupo | Todo el grupo | Orden a la hora de trabajar |
| Revisión de piezas | Conteo y vistazo de piezas entregadas | Todo el grupo | Determinar el material que tenemos para poder trabajar a gusto |
| Prototipo número 1 del robot | Realización del prototipo número 1 del robot | Renato Chacon Bruno Rojas | Prototipo de robot funcional |
| Revisión del prototipo | Ver detalles y ver errores | Milton Porlles | Verificar si el robot tiene algún detalle |
| Programación de movimientos | Se usa el software elegido para la primera programación del robot lego spike | Bruno Rojas | Robot completamente programado para poder usarse y hacer movimientos |
| Uso del robot con movimientos programados | Probar robot con los movimientos programados | Renato Chacon | Robot funcional y prueba de todos los movimientos para ver si hay algún detalle el cual revisa |
| Prototipo número 2 del robot | Realización de cambios al prototipo 1 | Renato Chacon | Cambios como eliminación de piezas, cambio de formas, etc. |
| Programación de movimientos nueva | Nuevos movimientos en el robot | Bruno rojas | Aplicación de movimientos nuevos en el prototipo número 2 |
| Investigación lenguaje para crear joystick | Buscar la correcta herramienta que nos ayude para programar un joystick | Todo el grupo | Encontrar solución a nuestro problema |
| Confección del primer informe sobre Lego spike | Realización de informe sobre robot | Renato Chacon Milton Porlles | Entrega de informe número 1 |
| Término de bitácoras de la etapa 1 | Bitácoras terminadas y subidas al redmine | Renato Chacon | Entregas de bitácoras |

Proyecto I 2025

GRUPO 2

- Feature #5669: fase 1**
 - Feature #4865:** 1ra reunion de planificacion sobr... (New 100%)
 - Feature #4870:** Investigar opciones de codigo d... (New 100%)
 - Feature #4872:** contruir un prototipo inicial (New 100%)
 - Feature #4866:** Revision del prototipo (New 100%)
 - Feature #4867:** Investigar medios para la creaci... (Resolved 100%)
 - Feature #5240:** programacion de movimientos (New 100%)
 - Feature #5252:** uso de robot con los movimientos (New 100%)
 - Feature #5253:** Prototipo numero 2 (Resolved 100%)
 - Feature #5254:** confeccion de informe (Resolved 100%)
 - Feature #5634:** Desarrollo del prototipo de Cont... (New 100%)
 - Feature #5635:** Segunda revision del Prototipo A... (New 100%)
- Feature #5674: fase 2**
 - Feature #5683:** investigacion de software para c... (New 95%)
 - Feature #5679:** Creacion de app para celular (New 100%)
 - Feature #5684:** uso de app inventor para prototi... (New 100%)
 - Feature #5680:** codificacion de codigo para la app (New 100%)
 - Feature #5685:** investigacion numero 2 de softw... (New 100%)
 - Feature #5681:** avance de informe numero 2 (New 60%)
 - Feature #5733:** Visualizacion de la plantilla ... (New 100%)
 - Feature #5819:** Creacion y realizacion del in... (New 20%)
 - Feature #5727:** utilizacion de pybricks para la pr... (New 100%)
 - Feature #5728:** Diseño del sistema de contr... (New 100%)
 - Feature #5729:** Programación de movimient... (New 100%)
 - Feature #5730:** programacion de interfaz grafica (New 100%)
 - Feature #5731:** realizacion de botones para ... (New 100%)
 - Feature #5732:** testeo de interfaz grafica (New 100%)
- Feature #6144: Parte 2**
 - Feature #6145:** Exposición numero 2 (New 100%)
 - Feature #6146:** Entrega de informe nuemero 2 (New 100%)
 - Feature #6147:** avance informe numero 3 (New 100%)
 - Feature #6148:** testeto de pista y obstaculos (New 100%)
- Feature #6149: Finalizacion**
 - Feature #6150:** Exposicion numero 3 (New 50%)
 - Feature #6151:** entrega de informe numero 3 (New 50%)
 - Feature #6152:** manual de usuario (New 50%)

3.3 Gestión de Riesgos

Se muestra la creación de una tabla la cual muestra los problemas que se pueden presentar durante la creación de la primera fase del proyecto. Resume y compara los tipos de daños los cuales puede presentar el robot

1. Daño catastrófico: hay que actuar de inmediato, porque puede detener el proyecto o hacer que tengamos que empezar todo de nuevo.
2. Daño crítico: es necesario tomar medidas para solucionarlo, ya que puede causar retrasos importantes en varias partes del proyecto.
3. Daño circunstancial: se debe atender cuando ocurra, porque puede retrasar una parte importante del proyecto.
4. Daño irrelevante: no es algo grave, solo un detalle que se puede arreglar en cualquier momento sin afectar mucho el proyecto.
5. Daño recurrente: no es serio, pero sucede varias veces y puede retrasar un poco las sesiones de trabajo, aunque no afecta las etapas principales del proyecto.

4. Identificación de los recursos y costos asociados

4.1 Hardware

- Set LEGO spike.
- Computador para la realización del sistema.
- Celular para uso como controlador del robot.

4.2 Software

- Sistema operativo Windows para programar las funciones del robot.
- Redmine, página para la organización del proyecto.
- ApplInventor es una página que nos ayuda a crear el joystick para las funciones del robot.
- Github para almacenar y guardar algunos datos.
- Documentos de Google para la realización del informe.

4.3 Recursos humanos

Costo del hardware:

| Producto | Precio (CLP) |
|--|--------------|
| Set a LEGO spike | \$772.306 |
| Notebook Gamer Asus TUF Gaming A15 Ryzen 7 8GB RAM 512GB SSD 15,6" NVIDIA RTX 2050 | \$619.000 |
| Notebook Gamer Nitro V15 ANV15-51-53W1-1 / Intel® Core™ i5 8 Núcleos / NVIDIA® GeForce® RTX 2050 / 16GB RAM / 512GB SSD / 15,6" FHD IPS | \$699.000 |
| Cougar PC (ryzen 5 5600x ,rtx 3060 zotac gaming, 16gb de ram a 3200mhz asus am4 tuf gaming x570-plus) | \$1.400.000 |
| Total | \$3.490.306 |

Costo del Software:

| Producto | Precio (CLP) |
|---------------------------------|--------------|
| Licencia Software LEGO Spike | \$26.700 |
| Lenguaje de programación python | \$0 |
| Total | \$26.700 |

Costo del Trabajador:

| Rol | Horas | Horas Extra | Precio/Hora |
|------------------|-------|-------------|-------------|
| Jefe de proyecto | 54 | 9 | \$35.000 |
| Programador | 60 | 5 | \$24.000 |
| Documentador | 51 | 1 | \$25.000 |
| Total: | - | - | \$3.661.000 |

En costos del trabajador para el conteo de las horas indicamos desde la fecha de inicio que fue 09-08-2025 hasta el día de la entrega del informe, respecto al precio de hora establecido nos guiamos gracias a algunos colegas que nos cooperaron con esta información respecto a cuánto podíamos cobrar

Total de costo:

| | |
|-----------------|-------------|
| Costo Hardware | \$3.490.306 |
| Costo Software | \$26.700 |
| Costo Empleados | \$3.661.000 |
| Total: | \$7.178.006 |

5. Análisis y diseño

5.1 Especificación de requerimientos

5.1.1 Requerimientos funcionales

Actores del sistema:

- **Usuario:** Mineros encargados del transporte de materiales.
- **Cliente:** Dueño de la empresa minera.

RF1. Entrega del paquete:

- El robot debe llegar a destino sin perder la carga.

RF2. Cualidades de carga:

- El robot debe tener una zona de carga con capacidad de 96 cm³ como mínimo.
- El robot debe poder moverse con 50 gramos de carga de 24 gramos.

RF3. Movilidad:

- El robot debe poder rodear obstáculos en su camino.
- El robot debe poder pasar sobre rampas con una inclinación de 4° aprox.
- El robot debe poder desplazarse de un punto X a un punto Y.

RF4. Comunicación:

- La aplicación cliente debe disponer de un panel de control con botones virtuales que traduzcan las pulsaciones del usuario en instrucciones de movimiento para el servidor

5.1.2 Alcance incluido

- El robot va a recorrer una pista definida, no una pista libre.
- Tendrá obstáculos para dificultar su paso.
- Solo hará una navegación de un punto x a un punto y y su base.
- Incluye un controlador simple.

5.1.3 Requerimientos no funcionales

| Atributo | Requerimiento | Métrica |
|----------------|--|---|
| Robustez | El robot debe soportar el entorno simulado(la pista) manteniendo su estructura | el robot debe operar si fallas estructurales durante al menos 5 minutos |
| Rendimiento | la máquina debe ser eficiente | el dispositivo debe lograr atravesar el recorrido en menos de 2 minutos |
| Usabilidad | La interfaz gráfica debe ser fácil de usar | Un usuario no entrenado debe lograr realizar un movimiento con el robot en menos de 10 segundos |
| Disponibilidad | La conexión debe ser estable mientras se use el robot | La conexión debe mantenerse por el 98% del tiempo de uso |

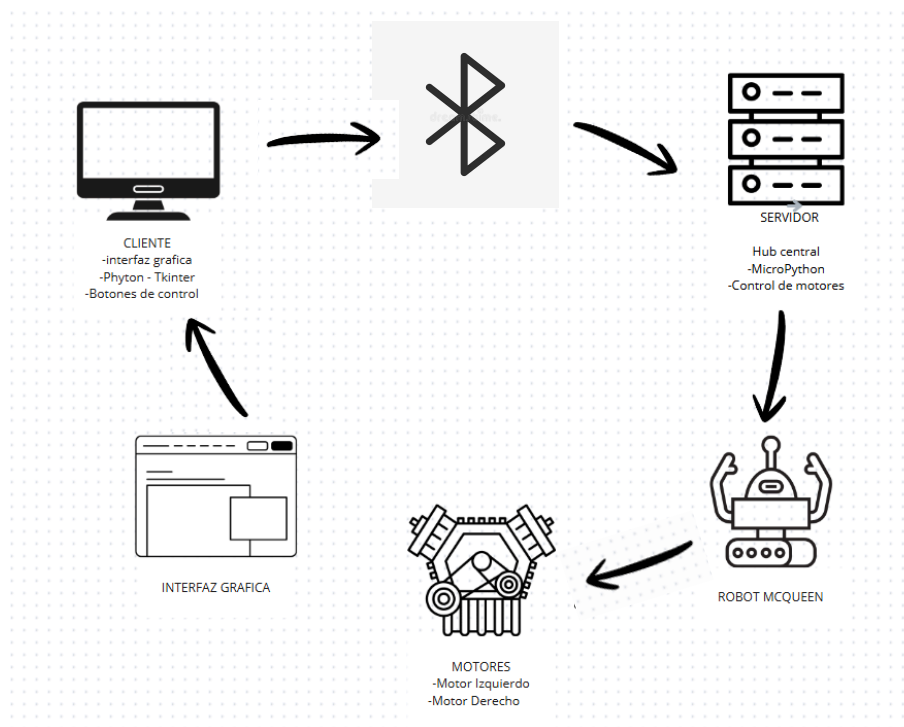
5.2 Arquitectura del software

Cliente (Interfaz de Control): Gestiona la GUI, captura las órdenes del operador y emite las peticiones de movimiento.

Canal (Bluetooth/BLE): Establece el enlace inalámbrico para el intercambio de datos en tiempo real entre ambos puntos.

Servidor (Hub LEGO Spike): Procesa las instrucciones recibidas, ejecuta la lógica de control y coordina los actuadores.

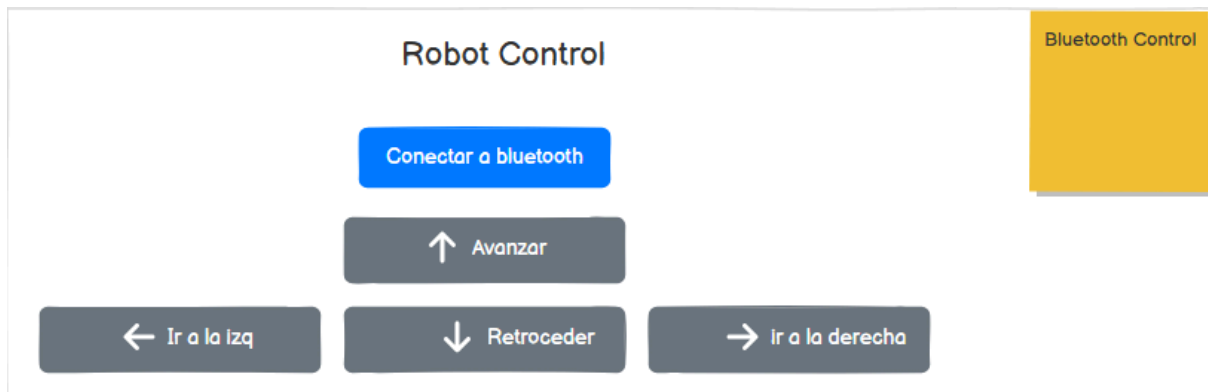
Robot McQueen: Entidad física que integra motores para el desplazamiento y sensores para la detección de obstáculos.



En el diagrama se puede observar como desde la GUI se envía un evento hacia el controlador del cliente y este envía un carácter a través de bluetooth y este llega al “servidor” que está en el hub y este decodifica el mensaje y lo envía a los motores

5.3 Diseño inicial de la interfaz gráfica de usuario (GUI)

Nos permite conectar el robot vía bluetooth y controlarlo mediante botones direccionales de forma simple



6. Implementación

En esta sección se presentan los resultados obtenidos, una justificación mediante la física de la configuración del robot, y un desglose de la interfaz gráfica.

6.1 Fundamentos del Movimiento

Para garantizar que el diseño es viable, debemos confirmar que el motor puede generar una **Fuerza Neta Positiva** que acelere el robot de 0 a su velocidad de crucero en un tiempo razonable. Para esto utilizaremos los datos de Lego.

Peso del vehículo = 0.8kg.

Diametro de Rueda = 0.028m.

Torque máximo de un motor = 0.25Nm.

Torque máximo de McQueen = 0.50Nm (dos motores).

Con la siguiente fórmula calculamos la fuerza de propulsión:

$$F_{\text{prop}} = \frac{\tau_{\text{total}}}{r_{\text{rueda}}} = \frac{0.50 \text{ N} \cdot \text{m}}{0.028 \text{ m}} \approx 17.86 \text{ Newtons}$$

por lo que podemos decir que el vehículo presenta la capacidad de desplazamiento dado que cumple con tener una fuerza neta positiva

6.2 Descripción del sistema.

Repositorio: [McQueen. Robot de transporte.](#)

6.2.1 Cliente.

En la siguiente figura se muestra la función runner la más importante del cliente

Esta función se encarga de buscar el hub, conectarse, crear e inyectar el archivo temporal y ejecutarlo, cargar el teclado para recibir los inputs y por último entrar en un loop infinito que recibe las peticiones de la GUI.

```
async def _runner(self):
    temp_path = None
    try:
        self.log("Buscando Hub...")
        device = await find_device()
        if not device:
            self.log("Error: No se encontró Hub.")
            return

        self.hub = PybricksHubBLE(device)
        await self.hub.connect()

        with tempfile.NamedTemporaryFile(mode='w', suffix='.py', delete=False, encoding='utf-8') as tf:
            tf.write(HUB_GATEWAY_CODE)
            temp_path = tf.name

        self.log("Conectado. Cargando teclado...")
        asyncio.create_task(self.hub.run(temp_path))
        self.running.set()

        while True:
            char_cmd = await self.queue.get()
            await self.hub.write(char_cmd.encode())

    except Exception as e:
        self.log(f"Error: {e}")
    finally:
        if temp_path and os.path.exists(temp_path): os.unlink(temp_path)
        if self.hub: await self.hub.disconnect()
        self.running.clear()
```

6.2.2 Servidor

Aquí se muestra lo que representa de manera lógica el servidor

Aquí se utiliza una poll para detectar la llegada de inputs desde la GUI para luego procesar esos inputs con un ciclo while infinito que decodifica el carácter recibido en comandos de pybricks que el hub puede entender.

```
# ----- HUB GATEWAY CODE -----
HUB_GATEWAY_CODE = """
from pybricks.hubs import PrimeHub
from pybricks.pupdevices import Motor
from pybricks.parameters import Port
from pybricks.tools import wait
import uselect
import usys

hub = PrimeHub()
motor_l = Motor(Port.D)
motor_r = Motor(Port.B)

poll = uselect.poll()
poll.register(usys.stdin, uselect.POLLIN)

hub.display.char('K') # 'K' for Keyboard Mode

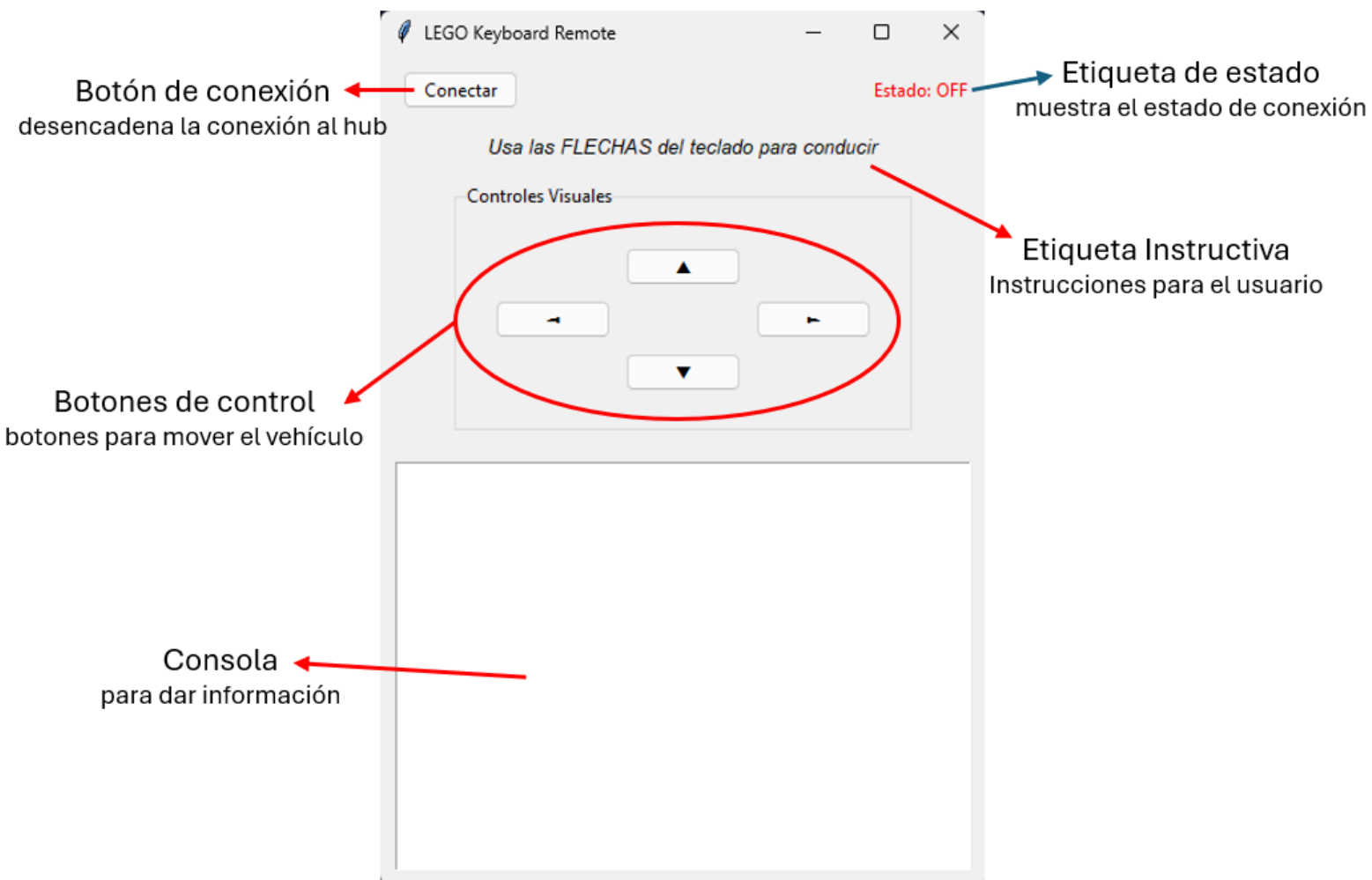
while True:
    if poll.poll(10):
        cmd = usys.stdin.read(1)
        if cmd == 'F':
            motor_l.run(800)
            motor_r.run(-800)
        elif cmd == 'B':
            motor_l.run(-800)
            motor_r.run(800)
        elif cmd == 'L':
            motor_l.run(-500)
            motor_r.run(-500)
        elif cmd == 'R':
            motor_l.run(500)
            motor_r.run(500)
        elif cmd == 'S':
            motor_l.stop()
            motor_r.stop()
        wait(10)
    """
```

6.2.3 GUI

Esta es una función para crear los botones de la GUI se solicita el texto para el botón, el comando que se enviará cuando se presiona y su posición en una cuadrícula. Luego le asignamos funciones en los eventos de press y release con el fin de que el robot se mueva solo mientras se presiona la tecla o el botón.

```
self.btns = {}  
def create_btn(text, cmd, row, col):  
    btn = ttk.Button(ctrl_frame, text=text)  
    btn.grid(row=row, column=col, padx=5, pady=5)  
    btn.bind("<ButtonPress>", lambda e: self.worker.send_command(cmd))  
    btn.bind("<ButtonRelease>", lambda e: self.worker.send_command("stop"))  
    self.btns[cmd] = btn  
  
create_btn("▲", "fwd", 0, 1)  
create_btn("◀", "left", 1, 0)  
  
create_btn("▶", "right", 1, 2)  
create_btn("▼", "bwd", 2, 1)
```

La GUI cuenta con varios componentes aquí los detallamos



7.Resultados

7.1 Estado actual del proyecto

El proyecto va en un estado bueno, ya que tenemos el 90% de sus funcionalidades hechas, debemos mejorar lo que es la programación del robot, afinar detalles que nos pueden retrasar o limitar un poco más lo que sería nuestra programación. Tenemos los obstáculos listos, la pista lista, solamente faltaría probar el robot en nuestra pista y terminar lo que sería la programación del robot.

7.2 Problemas encontrados y solucionados

Un problema principal fue el ser pocas personas en el grupo a la hora de realizar el trabajo pedido, cuesta mucho más realizar las cosas, también es muy notoria la falta de un integrante, la solución que le dimos fue hacer trabajo doble utilizando la plataforma de Discord.

Tuvimos un problema con la programación y es un problema que todavía tenemos, pero estamos cerca de la solución, una solución que tenemos y en la que estamos trabajando es pasar de la programación normal en Lego Spike a hacer la combinación con Python, cosa que está funcionando como lo tenemos planeado y tenemos propuesto tenerlo completo muy pronto. Son dos problemas que se nos presentaron dentro del proyecto; estos son los más importantes, ya que los demás pueden ser irrelevantes para mencionar, ya que la solución se encontró dentro de una charla o solamente una investigación previa.

8. Prueba de funcionamiento del sistema

8.1 Descripción de la prueba de funcionamiento

La prueba consiste en operar el robot de transporte minero sobre una pista simulada, la cual incluye un recorrido de un punto x a un punto y, incorporando obstáculos y variaciones dentro del terreno, el robot debe ser capaz de ser controlado por una interfaz gráfica desarrollada por la empresa minera recibiendo los comandos de avance, retroceso y giro a través de una conexión bluetooth.

Durante la presentación de la prueba, el robot debe transportar una carga sin perderla, manteniendo una estabilidad y superando los obstáculos propuestos en la pista, los motores deben responder correctamente a las instrucciones enviadas y la comunicación del cliente y servidor debe mantenerse estable durante toda la operación.

Esta prueba nos va a permitir verificar si nuestro robot cumple con todas las pruebas propuestas definidas en el proyecto, como las funciones de movilidad, transporte y de control.

8.2 Resultados observados para la prueba de funcionamiento



IMG_0049.mp4



IMG_0051.mp4

9. Conclusión

El desarrollo del proyecto McQueen nos permitió cumplir los objetivos propuestos, logrando una construcción y programación de un robot de transporte minero funcional, capaz de desplazarse sobre una pista simulada para superar obstáculo y transportar una carga controlado mediante una interfaz gráfica, en este proyecto aprendimos mucho sobre funcionamientos los cuales no teníamos conocimiento alguno y nos nutrimos de muchos conocimientos

Durante el desarrollo se presentaron dificultades, sentí la más grande la falta de integrantes y problemas con el desarrollo de la programación del robot, pudimos controlar estas situaciones buscando mejores organizaciones y más trabajo como equipo, más investigaciones y aprender más sobre un nuevo entorno de trabajo esto nos permitió avanzar de forma eficiente.

Como trabajo a futuro queremos hacer mejoras al proyecto como cambios estructurales para que se vea más estético, ampliar sus capacidades, mejor programación y mejorar lo que sería su interfaz para que se vea más estético.

10. Referencias

LEGO Education: SPIKE Prime Set (45678) Building Kit 5702016376319

EBay.(2025, 22 marzo). eBay. <https://www.ebay.com/itm/115737438795>

Notebook Gamer Asus TUF Gaming A15 Ryzen 7 8GB RAM 512GB SSD 15,6" NVIDIA

RTX 2050 | ABC. (s. f.). Abc.

<https://www.abc.cl/notebook-gamer-asus-tuf-gaming-a15-ryzen-7-8gb-ram-512gb-ssd-156-nvidia-rtx-2050/28419970.html>

Wikipedia contributors. (2025, 25 octubre). *List of system quality attributes*.

Wikipedia. [List of system quality attributes - Wikipedia](#)

LEGO Education. (s. f.). SPIKE Prime Technical Fact Sheet 45602. Recuperado de https://assets.education.lego.com/v3/assets/blt293eea581807678a/bltb9abb42596a7f1b3/5f8801b5f4c5ce0e93db1587/le_spike-prime_tech-fact-sheet_45602_1hy19.pdf