Modelo escala Lego
Vehículo Minero

Integrantes:

Cristofer Lazaro

Francisca Albornoz

Brayan Cahuachia

Ruth Huanca

Abraham Canaviri

Asignatura: Proyecto I

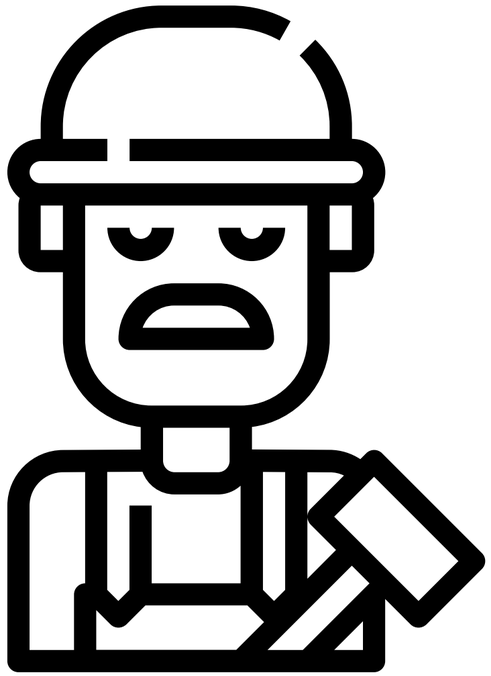
Profesor: Baris Klobertanz

ÍNDICE

Introducción

La minería representa el motor fundamental de la economía chilena y posiciona al país como líder mundial en la exportación de cobre y litio. Sin embargo, en la minería existen múltiples procesos en la extracción subterránea de minerales, el proyecto se basará en desarrollar un robot que replique el proceso de traslado, este será capaz de movilizarse con el material de carga que simulará los minerales extraídos, transportandolos de forma eficaz, asegurando la integridad del personal de trabajo y del material de carga de forma simulada.

PROBLEMA

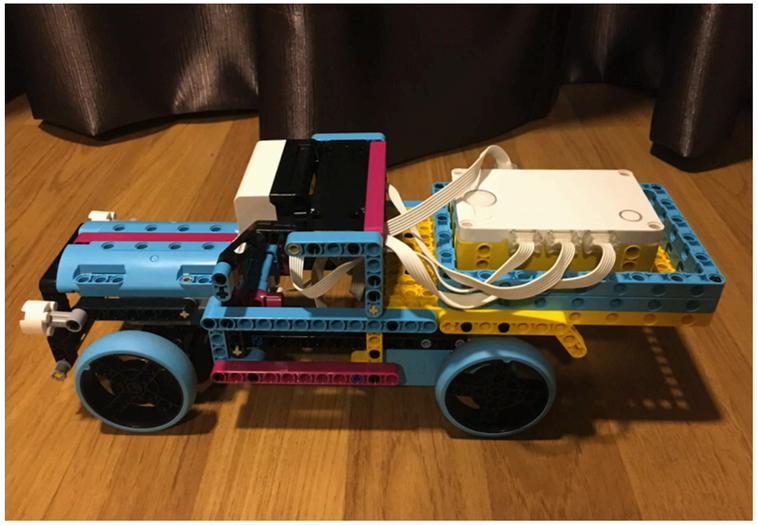


- Seguridad de los trabajadores.
- Traslado de materiales.

OBJETIVOS

Objetivo General:

- Desarrollar un modelo a escala de un vehículo minero utilizando el set LEGO SPIKE Prime, para simular el transporte de carga, evaluando su movilidad y control, con el fin de proponer una solución tecnológica que garantice la seguridad de los trabajadores frente a los desafíos del entorno subterráneo.



OBJETIVOS

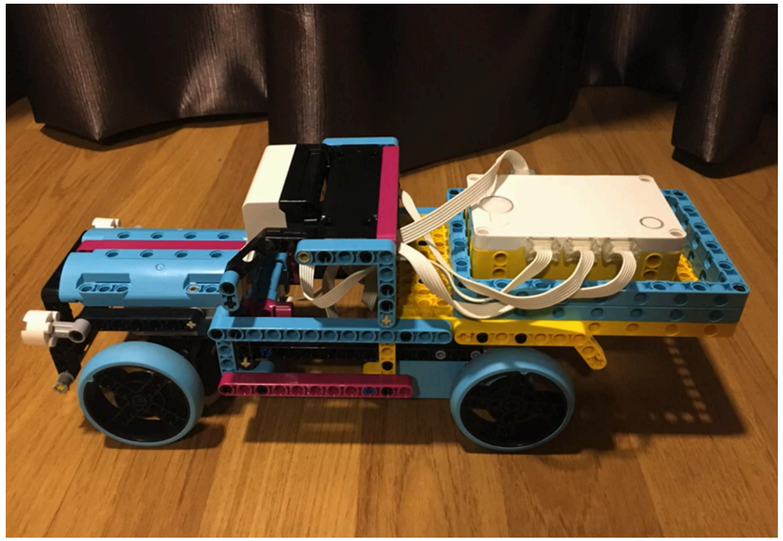
Objetivos Específicos:

- Analizar las capacidades técnicas del set LEGO SPIKE Prime, identificando la configuración de motores más adecuada para simular la tracción y dirección de un vehículo minero.

- Investigar y aplicar las librerías de Python (como Pybricks y Bleak) que permitan establecer el protocolo de comunicación inalámbrica necesario para el control remoto del Hub.

- Diseñar e implementar la estructura mecánica del vehículo, asegurando una configuración robusta que optimice la estabilidad durante el transporte de la carga.

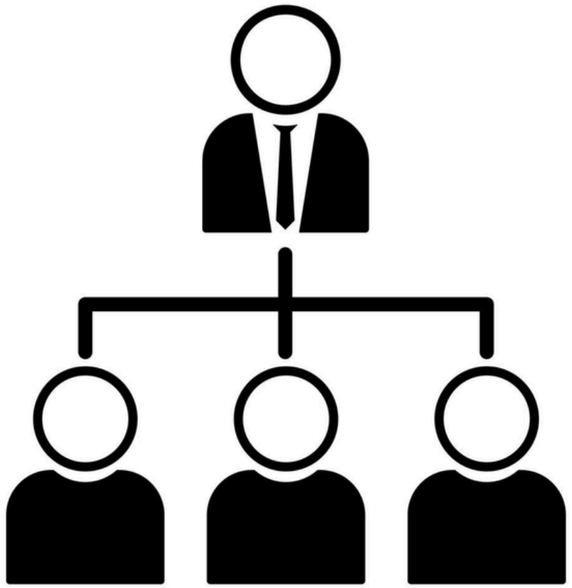
- Implementar una interfaz gráfica de usuario usando Tkinter que sea sencilla de usar para el usuario.



ESTRUCTURA ORGANIZACIONAL

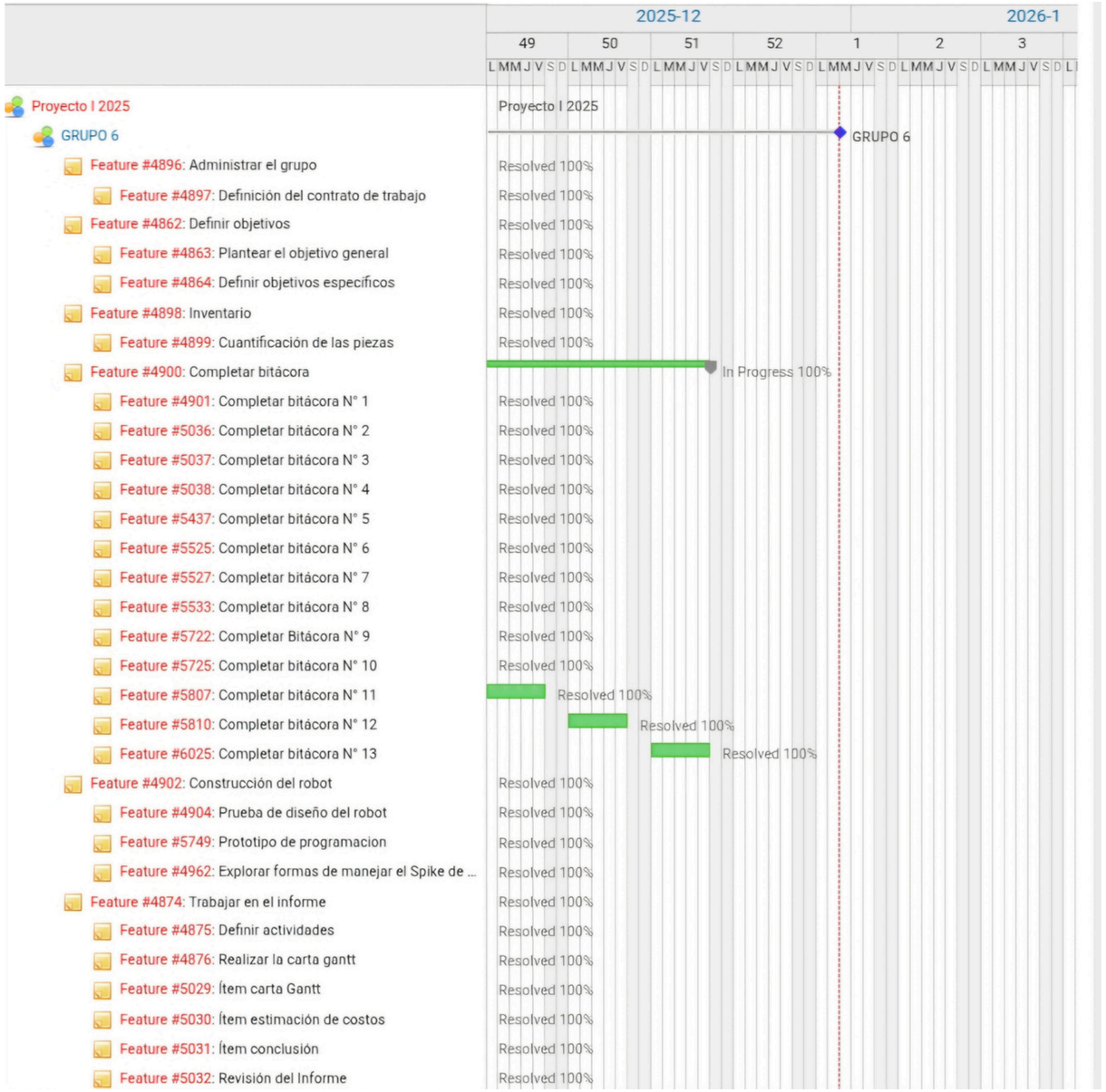
- Asignación clara de funciones
- Roles definidos por área de trabajo
- Coordinación entre integrantes

- Jefe de proyecto
- Documentador
- Ensamblador
- Programador

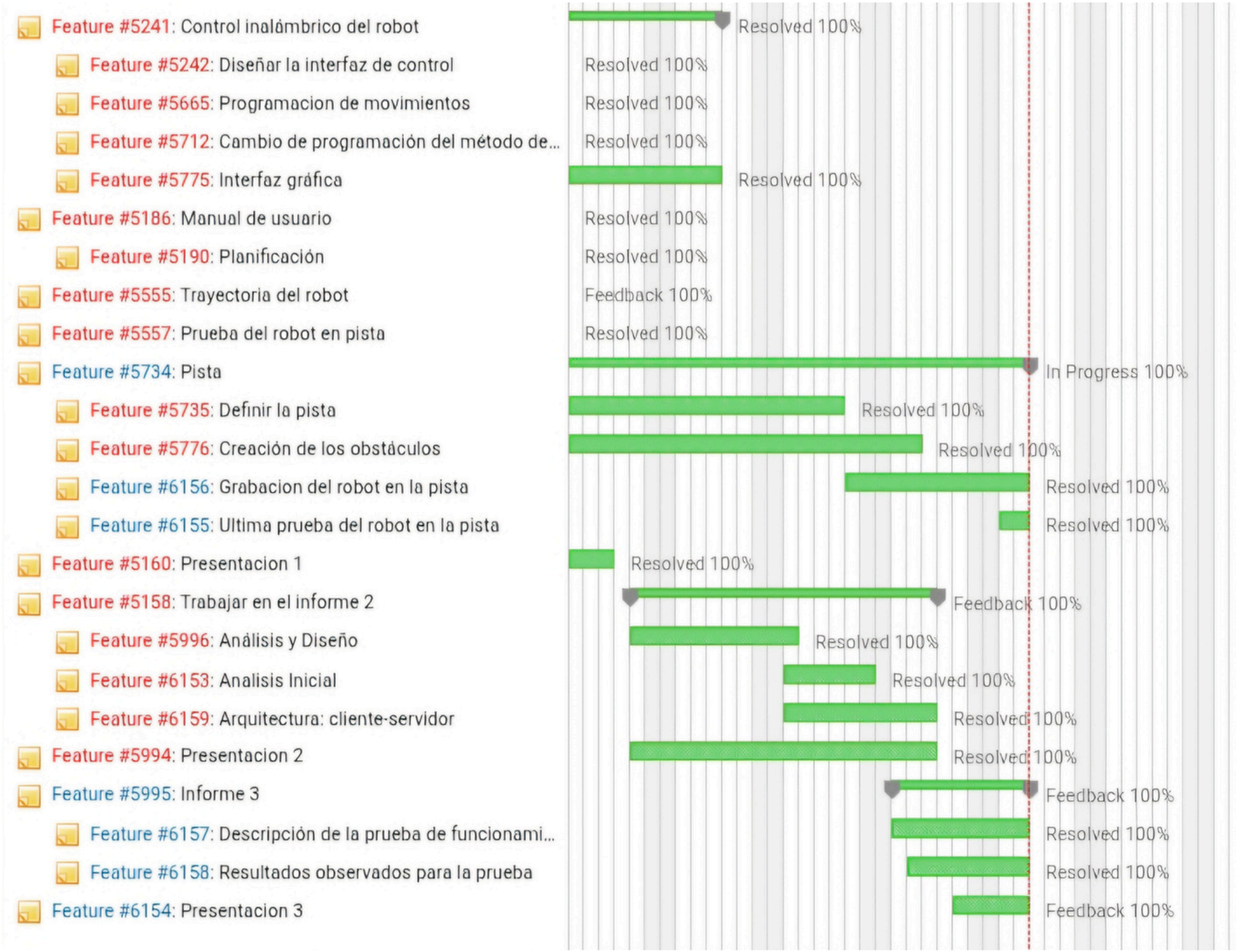


CARTA GANTT

Un medio visual para mostrar las actividades y el desarrollo de un proyecto , permitiendo tener una visión clara del avance de cada tarea dentro de un lapso de tiempo predeterminado.



CARTA GANTT


Feature #5241: Control inalámbrico del robot
Feature #5242: Diseñar la interfaz de control
Feature #5665: Programacion de movimientos
Feature #5712: Cambio de programación del método de...
Feature #5775: Interfaz gráfica
Feature #5186: Manual de usuario
Feature #5190: Planificación
Feature #5555: Trayectoria del robot
Feature #5557: Prueba del robot en pista
Feature #5734: Pista
Feature #5735: Definir la pista
Feature #5776: Creación de los obstáculos
Feature #6156: Grabacion del robot en la pista
Feature #6155: Ultima prueba del robot en la pista
Feature #5160: Presentacion 1
Feature #5158: Trabajar en el informe 2
Feature #5996: Análisis y Diseño
Feature #6153: Analisis Inicial
Feature #6159: Arquitectura: cliente-servidor
Feature #5994: Presentacion 2
Feature #5995: Informe 3
Feature #6157: Descripción de la prueba de funcionami...
Feature #6158: Resultados observados para la prueba
Feature #6154: Presentacion 3
Resolved 100%
Resolved 100%
Resolved 100%
Resolved 100%
Resolved 100%
Resolved 100%
Resolved 100%
Feedback 100%
Resolved 100%
In Progress 100%
Resolved 100%
Resolved 100%
Resolved 100%
Resolved 100%
Resolved 100%
Feedback 100%
Resolved 100%
Resolved 100%
Resolved 100%
Resolved 100%
Feedback 100%
Resolved 100%
Resolved 100%
Feedback 100%

GESTIÓN DE RIESGOS

Se han tomado medidas de los riesgos que podríamos encontrar en este proyecto, siendo estos 4 niveles:

1. Daño catastrófico.
2. Daño crítico.
3. Daño circunstancial.
4. Daño irrelevante.

FUNDAMENTOS DE MOVIMIENTO

FI 035

"Determinar la aceleración media necesaria para que el vehículo recorra una distancia de 1 metro en el menor tiempo posible."

Meta: 2.0 s
Modelo ideal

Distancia a recorrer (d): 1 m.
Tiempo objetivo (t): 2.0 s.
Velocidad inicial (v0): 0 m/s (el robot parte del reposo).
Incógnita: Aceleración media (a).

$$d = v_0 \cdot t + \frac{1}{2} \cdot a \cdot t^2$$

$$d = \frac{1}{2} \cdot a \cdot t^2$$

$$a = \frac{2d}{t^2}$$

$$a = \frac{2 \cdot 1m}{(2s)^2} = 0.5\ m/s^2$$

REQUERIMIENTOS FUNCIONALES

Hardware	Requerimiento	Descripción
RF1	Desplazamiento Multidireccional	El robot debe utilizar sus motores de tracción para moverse en 4 direcciones: avanzar, retroceder, y girar a izquierda o derecha.
RF8	Restablecer Trayectoria	El sistema debe accionar el motor de dirección para reubicar las ruedas a su ángulo cero al recibir la orden.

Software	Requerimiento	Descripción
RF2	Iniciar conexión	El sistema debe permitir al usuario establecer una conexión inalámbrica entre el Cliente y el Servidor desde la interfaz.
RF3	Finalizar conexión	La GUI debe permitir una desconexión manual de la comunicación inalámbrica mediante un control específico.
RF4	Visualización de Estado	La Interfaz Gráfica debe indicar visualmente el estado actual de la conexión.
RF5	Historial de acciones	La GUI debe mostrar en pantalla un registro de los últimos comandos ejecutados por el usuario.
RF6	Control de Dirección	La Interfaz debe desplegar visualmente los botones o controles necesarios para operar el movimiento del robot.
RF7	Realineación de dirección	La GUI debe proporcionar un botón específico que envíe la señal lógica para centrar la dirección del robot.

REQUERIMIENTOS NO FUNCIONALES

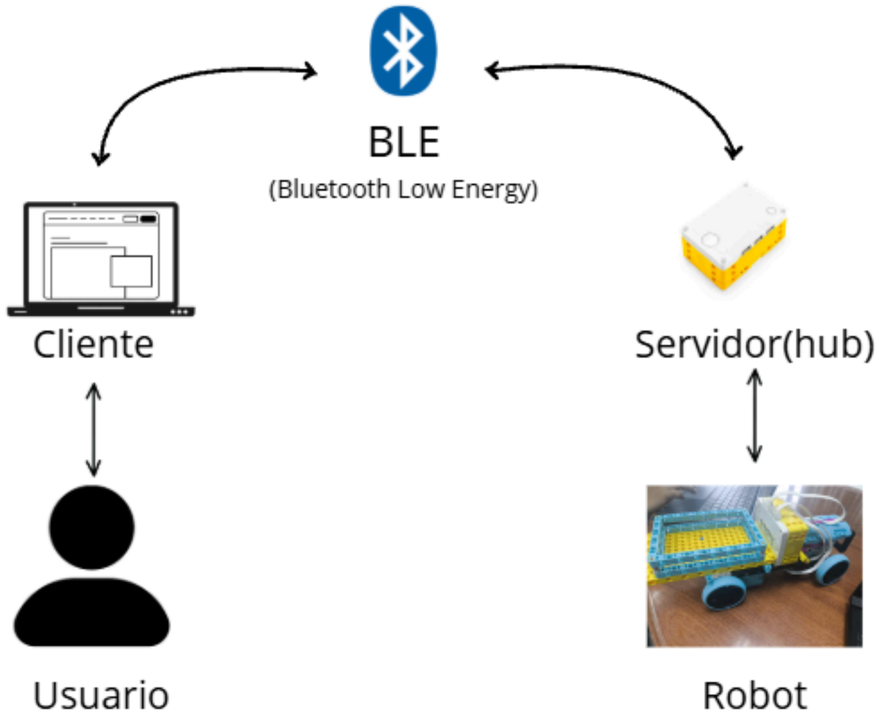
Hardware	Requerimiento	Descripción
RNF2	Disponibilidad y Energía	El sistema debe garantizar una disponibilidad continua de aproximadamente 10 minutos, condicionado a la carga de batería del Hub y el alcance físico.
RNF5	Restricción de componentes	El sistema está obligado físicamente a utilizar los motores de tracción del kit Lego para cumplir con el desplazamiento.

Software	Requerimiento	Descripción
RNF1	Uso práctico	La interfaz gráfica debe ser comprensible y sencilla de usar.
RNF3	Eficiencia	El software debe procesar y enviar la señal en menos de 1 segundo desde el clic en la interfaz hasta la reacción del robot.
RNF4	Comunicación	La arquitectura del software debe implementar estrictamente el protocolo Bluetooth Low Energy (BLE) para la transmisión de datos.

ARQUITECTURA DE SOFTWARE

Se ha diseñado una arquitectura cliente-servidor.

- Cliente: Desarrollado en Python, se encarga de capturar los eventos que el usuario quiera hacer.
- Comunicación (Bluetooth): Es el medio de comunicación entre el cliente y el servidor
- Servidor (Hub): Se mantiene a la espera de los scripts enviados por el usuario a través del cliente.



IMPLEMENTACIÓN

BOTON “AVANZAR”

```
# Avanzar
self.btn_avanzar = self.create_momentary_btn("▲ AVANZAR", "F", 1, 1)

def create_momentary_btn(self, text, cmd_char, r, c):
    """Crea un botón que envía comando al presionar y Stop al soltar"""
    btn = ctk.CTkButton(self.control_frame, text=text, height=45)
    btn.grid(row=r, column=c, padx=5, pady=5, sticky="ew")

    # Vincular eventos de mouse
    btn.bind("<ButtonPress-1>", lambda e: self.worker.send_command(cmd_char))
    btn.bind("<ButtonRelease-1>", lambda e: self.worker.send_command("S"))
    return btn
```

CLIENTE - ENVÍO POR BLE (COLA + WRITE)

```python
def send_command(self, char):
    if self.running.is_set() and self.queue:
        # 1. Enviar el comando al loop asyncio
        self.loop.call_soon_threadsafe(self.queue.put_nowait, char)

        # 2. Traducir y mostrar en GUI
        desc = self.CMD_DESC.get(char, f"Comando: {char}")
        self.log(f"Acción: {desc}")
```

```python
async def _runner(self):
    temp_path = None
    while True:
        await self._connect_request.wait()
        try:
            ...
            self.queue = asyncio.Queue()
            self.run_task = asyncio.create_task(self.hub.run(temp_path))
            ...
            while self.running.is_set():
                try:
                    cmd = await self.queue.get()
                    if self.hub:
                        await self.hub.write(cmd.encode())
                except asyncio.CancelledError:
                    break
                except Exception as e:
                    if "disconnected" in str(e):
                        break
                    self.log(f"Error enviando: {e}")
        ...
```

SERVIDOR - INTERPRETACIÓN DE COMANDOS

```
while True:
    if poll.poll(10):
        cmd = usys.stdin.read(1)
        if cmd == 'F':
            motorB.run(-1100)
            motorF.run(1100)
        elif cmd == 'T':
            motorB.run(-1100)
            motorF.run(1100)
        elif cmd == 'B':
            motorB.run(800)
            motorF.run(-800)
```

```
    elif cmd == 'L':
        motorD.run_target(500, -25, wait=False)
    elif cmd == 'R':
        motorD.run_target(500, 25, wait=False)
    elif cmd == 'C':
        motorD.run_target(500, 0, wait=True)
        motorD.stop()
    elif cmd == 'S':
        motorB.stop()
        motorF.stop()
    elif cmd == 'X':
        motorB.stop()
        motorF.stop()
        motorD.stop()
        raise SystemExit
wait(10)
```

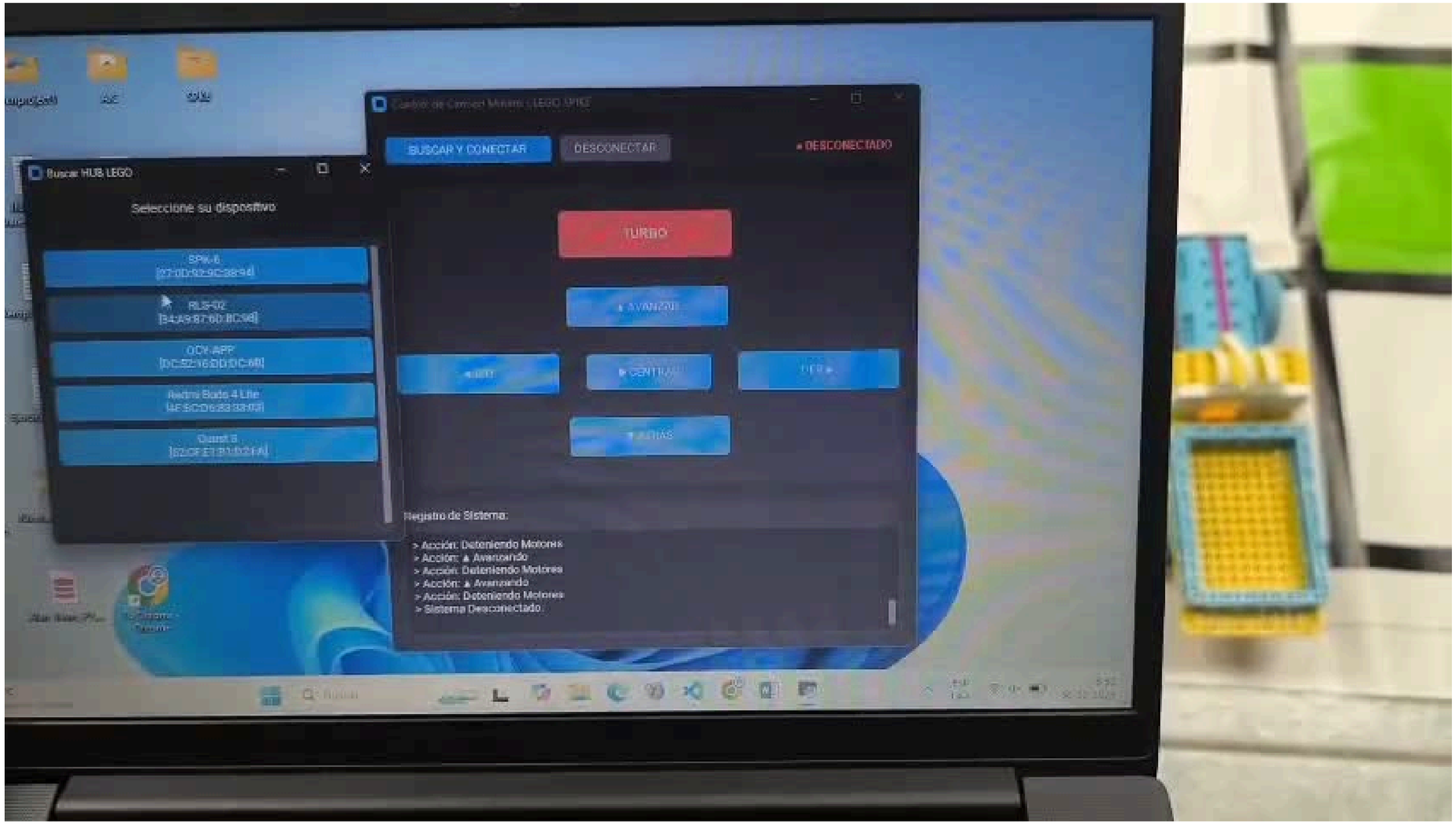
PRUEBA MÍNIMA FUNCIONAL

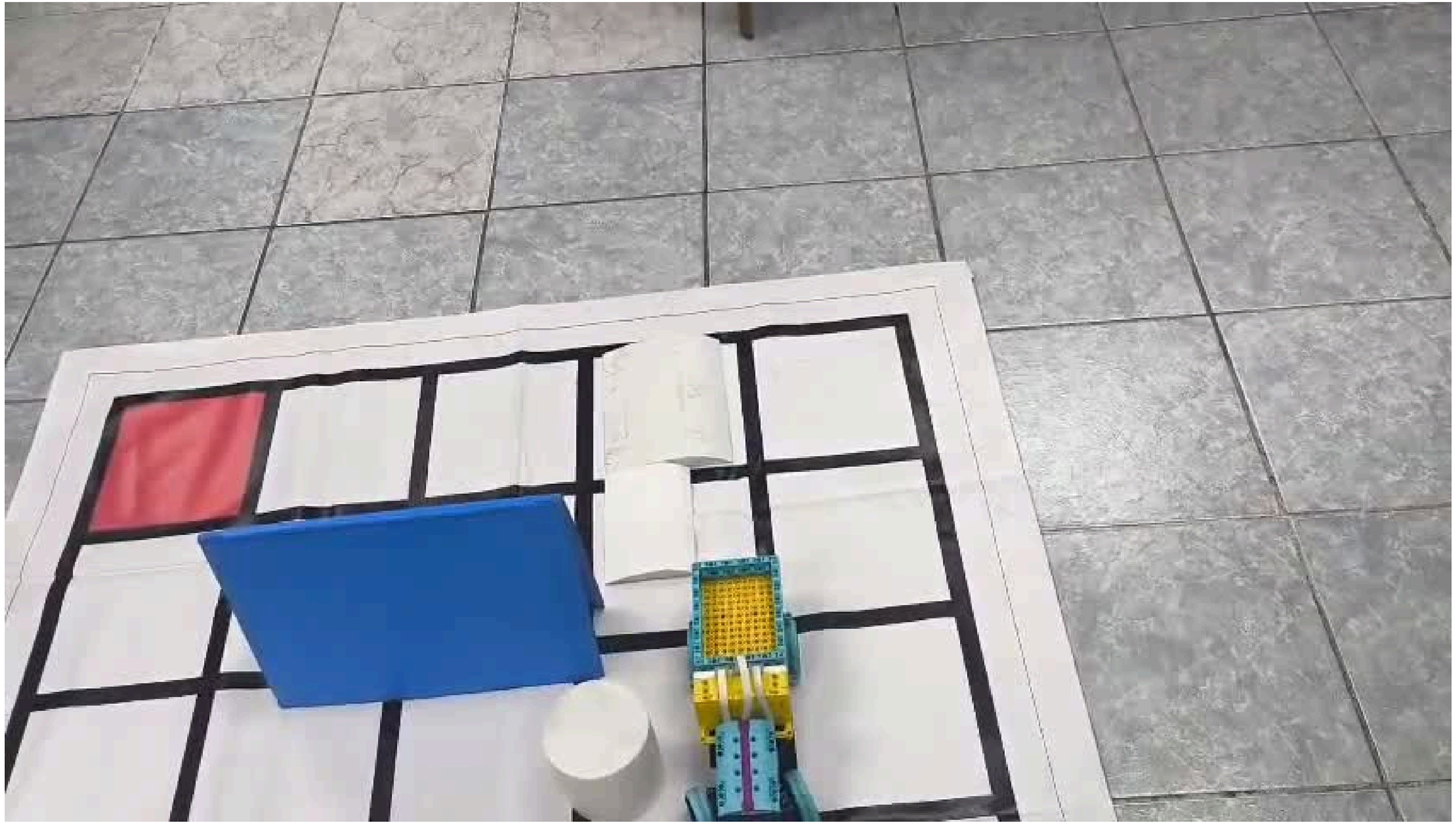
Verificar que el robot responda correctamente a los comandos de movimiento básicos:

- avanzar
- retroceder
- girar a la izquierda
- girar a la derecha

Se requiere establecer la comunicación entre el cliente y el servidor.

Corresponde a los requerimientos funcionales RF1 y RF2.


Buscar HUB LEGO
Seleccione su dispositivo
SPIK-6
RLS-02
OCY-APP
Redmi Buds 4 Lite
Quest 3
BUSCAR Y CONECTAR
DESCONECTAR
DESCONECTADO
TURBO
AVANZAR
CENTRAR
Registro de Sistema:
> Acción: Deteniendo Motores
> Acción: Avanzando
> Acción: Deteniendo Motores
> Acción: Avanzando
> Acción: Deteniendo Motores
> Sistema Desconectado



MANUAL DE USUARIO

Actualmente, se posee un repositorio de GitHub del proyecto con un archivo llamado README.md

Nombre del Proyecto.

Características:

- Conexión Inalámbrica
- Control Dual
- Feedback Visual
- Modo Turbo
- Logs en Tiempo Real

Instalación

Uso Básico

Capturas

CONCLUSIÓN

- El robot desarrollado simula la etapa de transporte.
- Se implementó con éxito diseño del modelo, el control del robot, la conexión inalámbrica y el restablecimiento de dirección
- Se identificaron y solucionaron problemas relacionados con el control del robot.

CONCLUSIÓN

Aprendizajes:

- La gestión de riesgos es un recurso útil.
- Uso de GitHub para controlar versiones.

Mejora futura:

- Extender el manejo de errores con detección de desconexión y reconexión Bluetooth automática.

GRACIAS