SORTING MINING:
CLASIFICADOR REMOTO DE MINERALES PARA LA OPTIMIZACIÓN Y SEGURIDAD EN MINERIA

Alumnos:

- Camilo Geraldo
- Ignacio Cuevas
- Maximiliano Burgos
- Jose Quispe
- Matias Sagredo

Asignatura:

- Proyecto I

Profesor:

- Baris Nikolai Klobertanz Quiroz

CONTENIDO

INTRODUCCIÓN

La industria minera es un pilar económico en Chile, representando más 10% del PIB del país. Con el avance tecnológico se introduce la Minería 4.0, una transición entre de mineria tradicional a una más digitalizada y automatizada.

No obstante, la seguridad de los trabajadores persiste como un problema que las empresas mineras deben avarcar con seriedad.

Es en este sentido que el proyecto tiene la finalidad de desarrollar e implementar un gemelo físico de una maquinaria empleada en la industria minera y automatizarla para que un usuario pueda operar dicho prototipo de forma remota, simulando así una etapa del proceso minero.

ORGANIZACIÓN PERSONAL

Rol	Responsable
Jefe de proyecto	Maximiliano Burgos
Ensamblador	Matias Sagredo
Diseñador	Jose Quispe
Programador	Camilo Geraldo
Documentador	Ignacio Cuevas

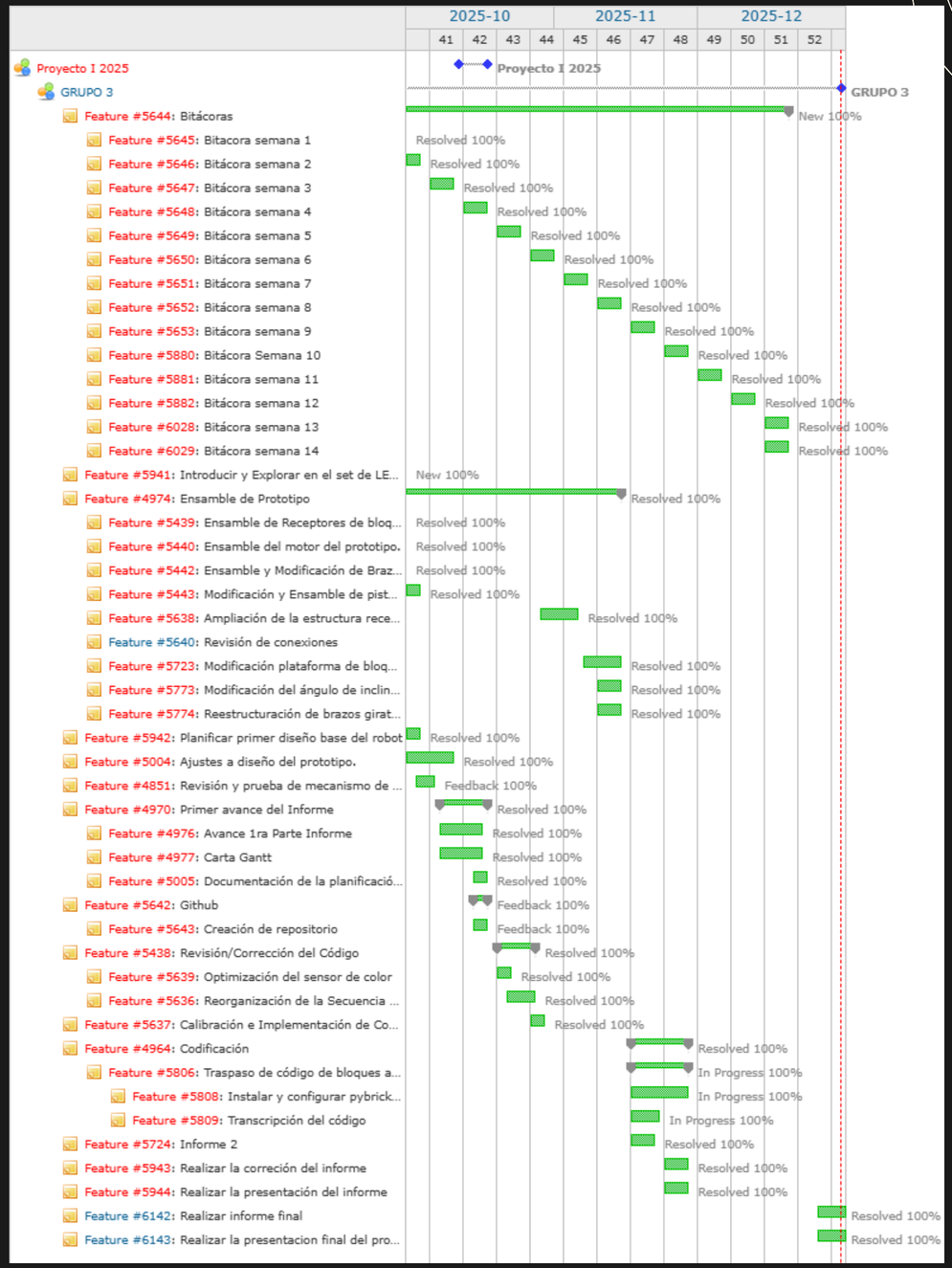
OBJETIVO GENERAL

Implementar un prototipo robótico utilizando el kit LEGO Spike Prime, que simule el proceso de clasificación de materiales en la industria minera siendo capaz de identificar diferentes tipos de materiales representados por piezas de colores y ordenarlos en contenedores designados de manera autónoma o manual por los trabajadores mineros.

OBJETIVOS ESPECÍFICOS

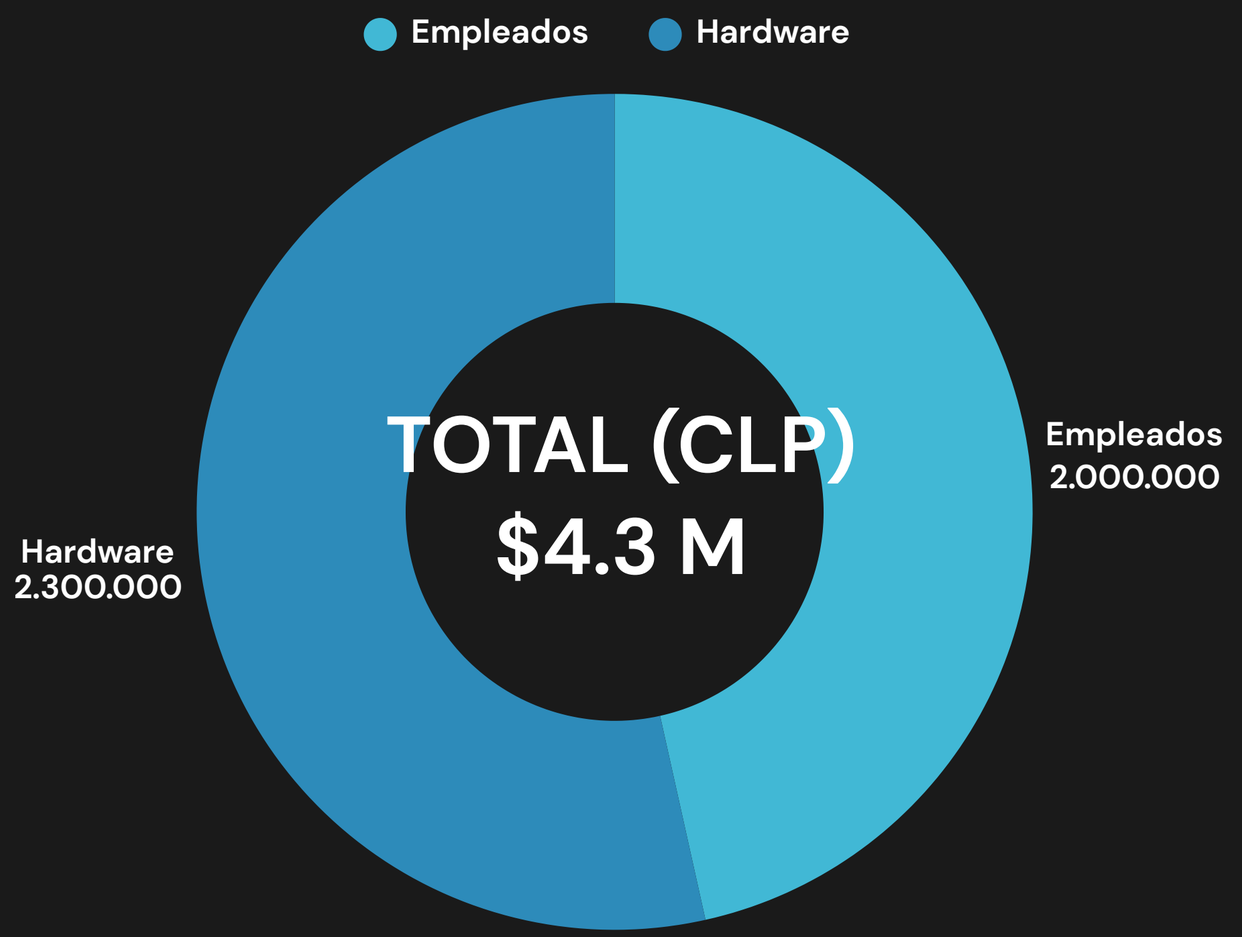
1. Diseñar y ensamblar en 3 semanas una estructura de robot con LEGO Spike Prime que incluya mecanismos de entrada y salida de piezas.
2. Programar el robot para identificar y clasificar piezas por color de forma autónoma a través del sensor de color del set LEGO Spike Prime en un lapso de 5 semanas.
3. Diseñar una interfaz de usuario tkinter capaz de enviar comandos inalámbricos al robot, permitiendo al menos 4 acciones de control manual.
4. Optimizar el prototipo robótico, realizando mejoras en software y estructura, hasta lograr al menos un 98% de precisión en pruebas controladas tanto del modo automático como manual.
5. Elaborar en 2 meses un informe técnico de al menos 10 páginas y una presentación preliminar de mínimo 10 diapositivas que expliquen el diseño, funcionamiento y resultados del robot.

CARTA GANTT


2025-10
2025-11
2025-12
41
42
43
44
45
46
47
48
49
50
51
52
Proyecto I 2025
GRUPO 3
Feature #5644: Bitácoras
Feature #5645: Bitacora semana 1
Feature #5646: Bitácora semana 2
Feature #5647: Bitácora semana 3
Feature #5648: Bitácora semana 4
Feature #5649: Bitácora semana 5
Feature #5650: Bitácora semana 6
Feature #5651: Bitácora semana 7
Feature #5652: Bitácora semana 8
Feature #5653: Bitácora semana 9
Feature #5880: Bitácora Semana 10
Feature #5881: Bitácora semana 11
Feature #5882: Bitácora semana 12
Feature #6028: Bitácora semana 13
Feature #6029: Bitácora semana 14
Feature #5941: Introducir y Explorar en el set de LE...
Feature #4974: Ensamble de Prototipo
Feature #5439: Ensamble de Receptores de bloq...
Feature #5440: Ensamble del motor del prototipo.
Feature #5442: Ensamble y Modificación de Braz...
Feature #5443: Modificación y Ensamble de pist...
Feature #5638: Ampliación de la estructura rece...
Feature #5640: Revisión de conexiones
Feature #5723: Modificación plataforma de bloq...
Feature #5773: Modificación del ángulo de inclin...
Feature #5774: Reestructuración de brazos girat...
Feature #5942: Planificar primer diseño base del robot
Feature #5004: Ajustes a diseño del prototipo.
Feature #4851: Revisión y prueba de mecanismo de ...
Feature #4970: Primer avance del Informe
Feature #4976: Avance 1ra Parte Informe
Feature #4977: Carta Gantt
Feature #5005: Documentación de la planificació...
Feature #5642: Github
Feature #5643: Creación de repositorio
Feature #5438: Revisión/Corrección del Código
Feature #5639: Optimización del sensor de color
Feature #5636: Reorganización de la Secuencia ...
Feature #5637: Calibración e Implementación de Co...
Feature #4964: Codificación
Feature #5806: Traspaso de código de bloques a...
Feature #5808: Instalar y configurar pybrick...
Feature #5809: Transcripción del código
Feature #5724: Informe 2
Feature #5943: Realizar la correción del informe
Feature #5944: Realizar la presentación del informe
Feature #6142: Realizar informe final
Feature #6143: Realizar la presentacion final del pro...
Proyecto I 2025
GRUPO 3
New 100%
Resolved 100%
Feedback 100%
In Progress 100%

GESTIÓN DE RIESGOS

PLANIFICACIÓN DE RECURSOS



REQUERIMIENTOS FUNCIONALES

RF1 **Identificación de materiales:** El prototipo debe identificar cuatro tipos de materiales representados por piezas de LEGO de los siguientes colores: rojo, azul, amarillo y verde.

RF2 **Clasificación física:** El prototipo debe ser capaz de mover un bloque de color identificado hacia su cuadrante o contenedor correspondiente.

RF3 **Modo de operación automático:** El sistema debe contar con una función autónoma que ejecute cíclicamente la identificación (RF1) y la clasificación (RF2) de las piezas sin intervención del operador.

RF4 **Modo de control manual:** El prototipo debe permitir la ejecución de las acciones de clasificación (RF2) mediante comandos enviados por un operador desde la Interfaz Gráfica de Usuario (GUI).

REQUERIMIENTOS FUNCIONALES

RF5 **Visualización de datos:** La interfaz gráfica debe mostrar al operador el color detectado por el sensor del prototipo.

RF6 **Eliminación de Materiales:** El sistema debe permitir al operador eliminar cualquier material no deseado desde la interfaz gráfica.

RF7 **Control de contenedores:** El prototipo debe permitir el giro de los contenedores de depósito en ángulos de 0° y 180∘ para la recepción de piezas.

REQUERIMIENTOS NO FUNCIONALES

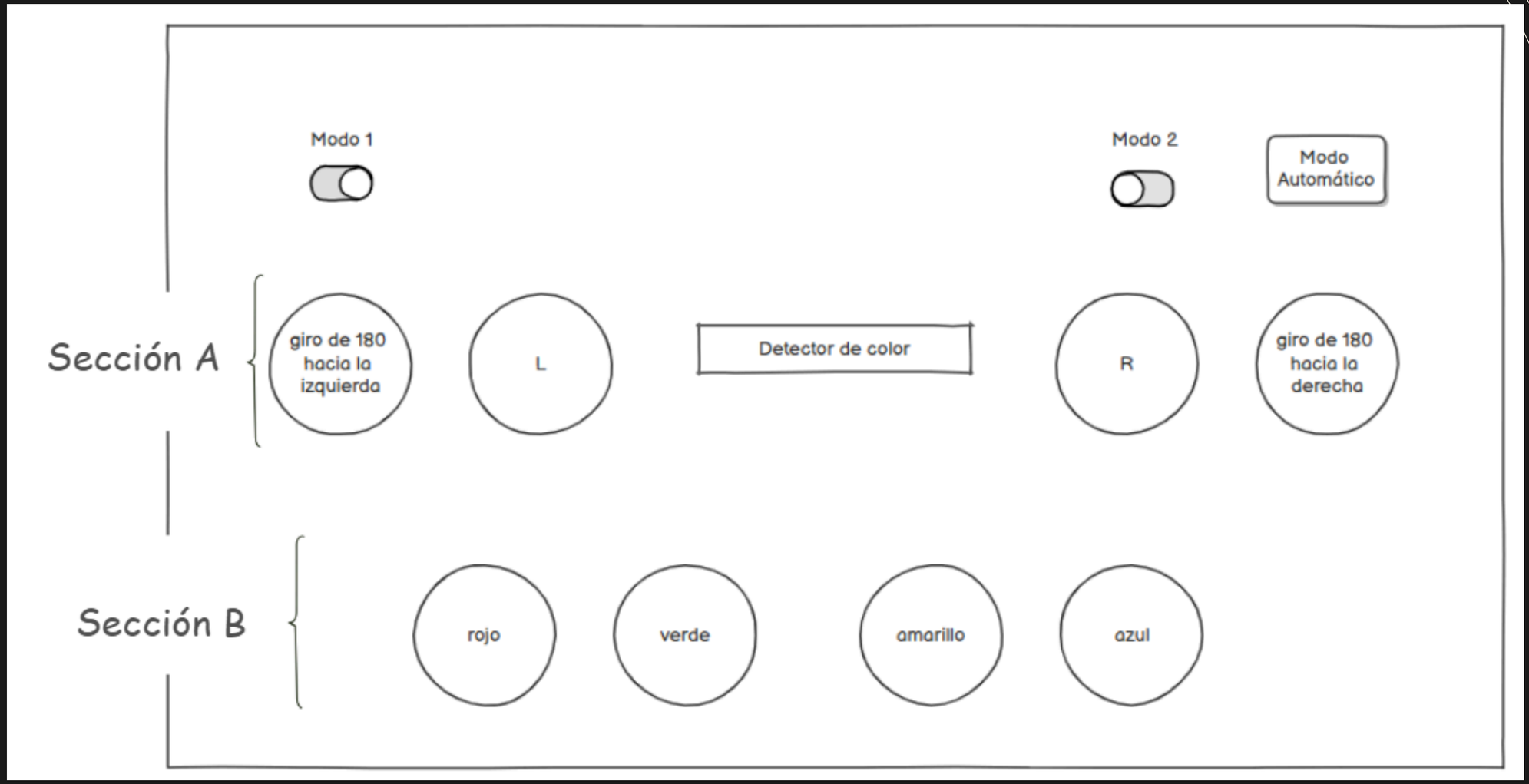
- **Disponibilidad:** El prototipo debe garantizar una disponibilidad operativa del 100% durante el horario de la demostración. Durante operaciones mineras, debe operar por encima del 99.5% permitiendo realizar mantenimiento y cambio de las baterías.
- **Robustez:** El prototipo debe manejar de forma elegante situaciones inesperadas como es el caso de detectar piezas de material que no puede clasificarse, moviendo la pieza a una zona de rechazo sin interferir en el proceso general de clasificación de las piezas.
- **Usabilidad:** La interfaz gráfica de usuario debe ser diseñada bajo principios de diseño universal, debiendo obtener un puntaje mayor a 70 en el cuestionario SUS (System Usability Scale) y al menos el 80% de los usuarios o trabajadores deben completar la tarea de clasificación sin asistencia.

REQUERIMIENTOS NO FUNCIONALES

- **Rendimiento:** El tiempo de latencia entre el envío de un comando manual desde la interfaz de usuario y el inicio de la ejecución física del comando por el prototipo no debe superar los 500 milisegundos. El ciclo completo de clasificación no debe superar los 2 segundos.

- **Seguridad:** La interfaz gráfica debe contar con un botón virtual de parada de emergencia que al ser activado corte instantáneamente la potencia a todos los actuadores.

WIREFRAME


Modo 1
Modo 2
Modo Automático
Sección A
giro de 180 hacia la izquierda
L
Detector de color
R
giro de 180 hacia la derecha
Sección B
rojo
verde
amarillo
azul

INTERFAZ GRÁFICA DE USUARIO

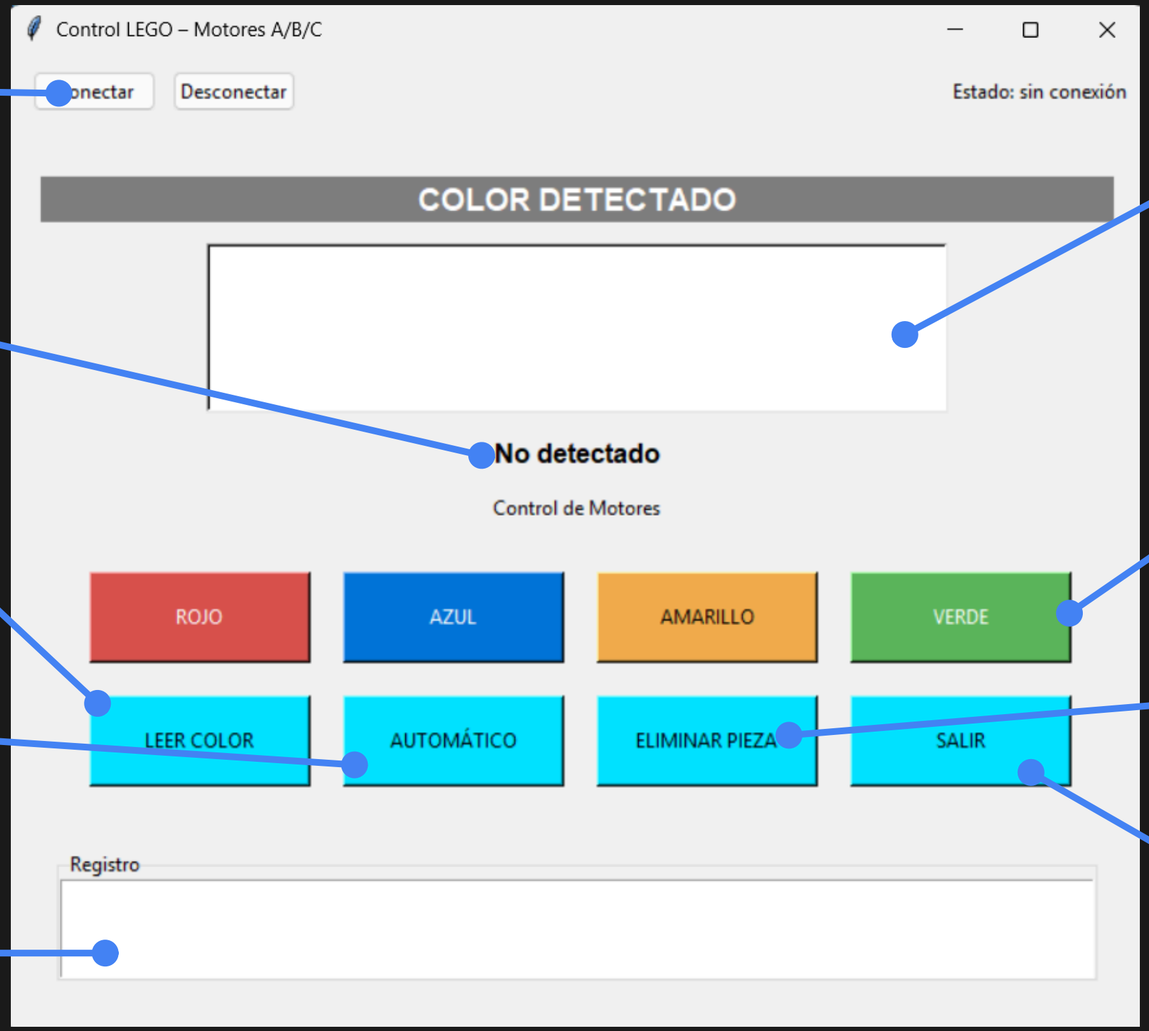
Inicia/Finaliza la conexión vía bluetooth entre el sistema en el computador y el hub Spike Prime.

Visualizador que muestra el nombre del color detectado; en caso de no haber piezas muestra “No detectado”

IInicia el sensor de colores para luego mostrar el color detectado en el display de colores.

Inicia el prototipo robótico en modo automático, clasificando los bloques de forma automática.

Visualizador que muestra los comandos ejecutados por el sistema y los errores que pueden producirse.



Visualizador que muestra el color detectado por el sensor de color.

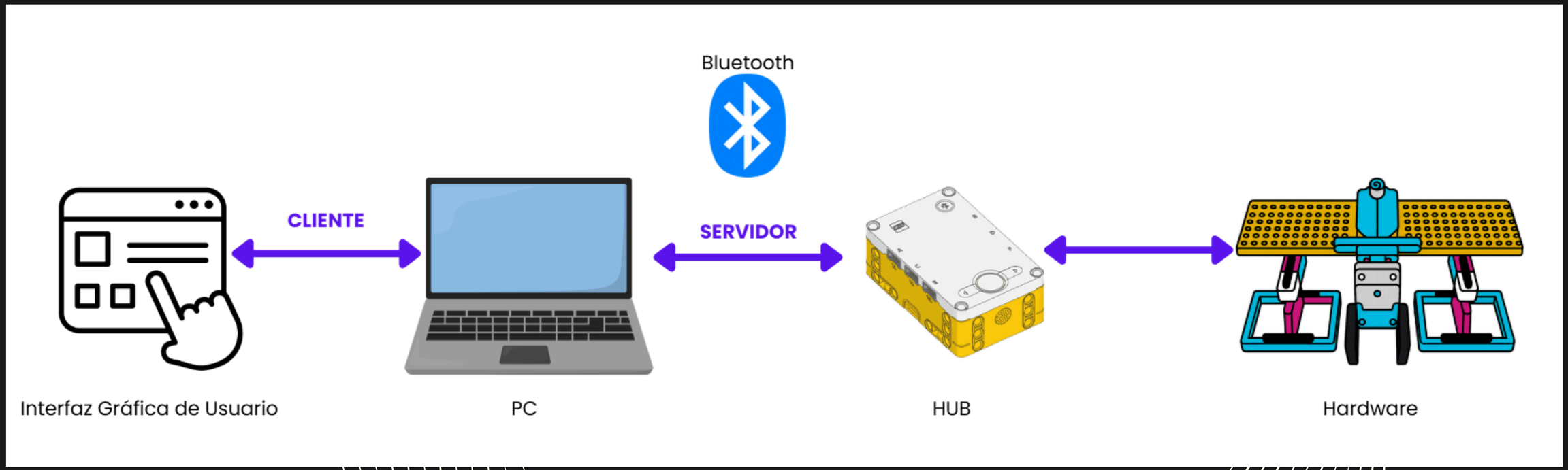
Botones que clasifican las piezas en los contendores correspondientes.

Elimina la pieza deslizando lentamente con el fin de que no se ingrese junto a las demás piezas

Finaliza el código, terminando todas las acciones y cerrando la interfaz gráfica.

ARQUITECTURA DE SOFTWARE

La arquitectura del sistema se basa en un modelo Cliente-Servidor, diseñado para permitir el control remoto y la supervisión del proceso de clasificación de minerales. Esta estructura separa la gestión del usuario de la ejecución física de las tareas en el prototipo.



INTERFAZ

```
Frame_btn = ttk.Frame(body, padding=20)
Frame_btn.pack(fill='both', expand=True)
btn_coman=[
    ("ROJO","#d9534f","white","#b94a42",lambda: self.worker.send_command("rojo")),
    ("AZUL","#0275d8","white","#025aa5",lambda: self.worker.send_command("azul")),
    ("AMARILLO","#f0ad4e","black","#d08b36",lambda: self.worker.send_command("amarillo")),
    ("VERDE","#5cb85c","white","#4cae4c",lambda: self.worker.send_command("verde")),
    ("LEER COLOR","#00e1ff","black","#0093a7",lambda: self.worker.send_command("Leer")),
    ("AUTOMÁTICO","#00e1ff","black","#0093a7",lambda: self.worker.send_command("auto")),
    ("ELIMINAR PIEZA","#00e1ff","black","#0093a7",lambda: self.worker.send_command("Eliminar")),
    ("SALIR","#00e1ff","black","#0093a7",self.root.quit)
]
```

```
#botones de ventana menu
fila=0
columna=0
for texto, fondo, clrFuente, fondoActivado, accion in btn_coman:
    if columna==4:
        columna=0
        fila+=1
    btn = tk.Button(Frame_btn, text=texto, width=18, height=3,
                    bg=fondo, fg=clrFuente, activebackground=fondoActivado,
                    command=accion)
    btn.grid(row=fila, column=columna, padx=10, pady=10, sticky="nsew")
    columna+=1
```

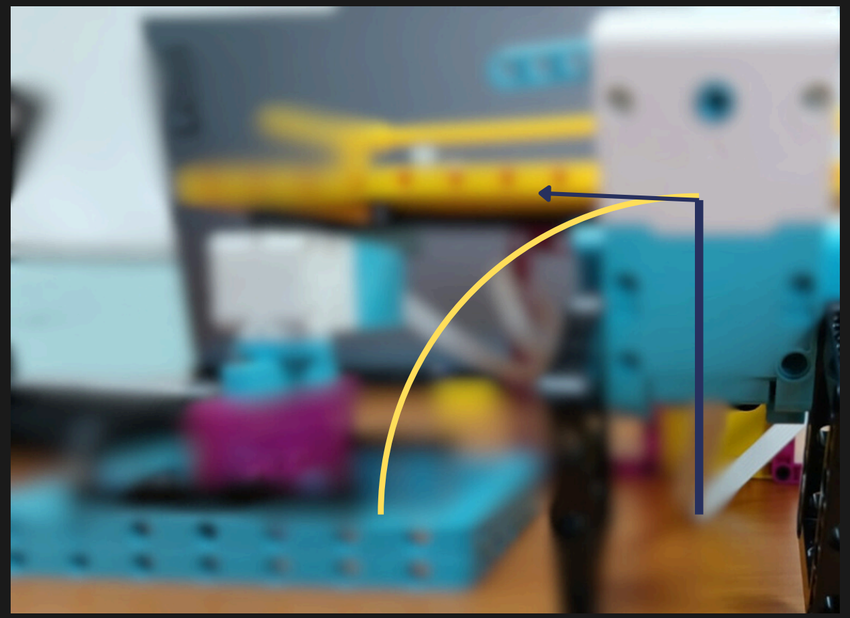
BLEWORKER

```python
async def execute_command(worker, comando: str, log_cb=None):
    program = create_program(comando)

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

    try:
        if log_cb:
            log_cb(f"Ejecutando: {comando}")
        # Para recibir datos, capturar el output impreso
        if "Leer" in comando:···
        else:
            await worker.hub.run(temp_path, wait=True, print_output=True)
            worker.color_canvas.configure(bg="#ffffff")
            worker.color_label.configure(text=f"Color detectado: No detectado")
```

```python
        "rojo": """
target_C = 0
current_C = motorC.angle()
diff_C = (target_C - current_C) % 360
if diff_C > 180:
    diff_C = diff_C - 360
if abs(diff_C) > TOLERANCE:
    motorC.run_target(600, target_C)
motorA.run_angle(600, 180)
""",

        "Leer": """···

        "Eliminar": """···
    }
    if comando == "auto":···
    else:
        drive_code = actions.get(comando, "")
        code= f"""
from pybricks.hubs import PrimeHub
from pybricks.pupdevices import Motor, ColorSensor
from pybricks.parameters import Port
from pybricks.tools import wait

hub = PrimeHub()

motorA = Motor(Port.A)
motorB = Motor(Port.F)
motorC = Motor(Port.E)
TOLERANCE = 20
{drive_code}

wait(500)
print("Comando '{comando}' ejecutado.")
        """
        return code
```

FUNDAMENTOS DE MOVIMIENTO

Movimiento Rotación de Clasificador:



Tiempo de caida:

$$t = \sqrt{\frac{2h}{g}} = \sqrt{\frac{0.16}{9.8}} \approx \sqrt{0.01633} \approx 0.1278\ s$$

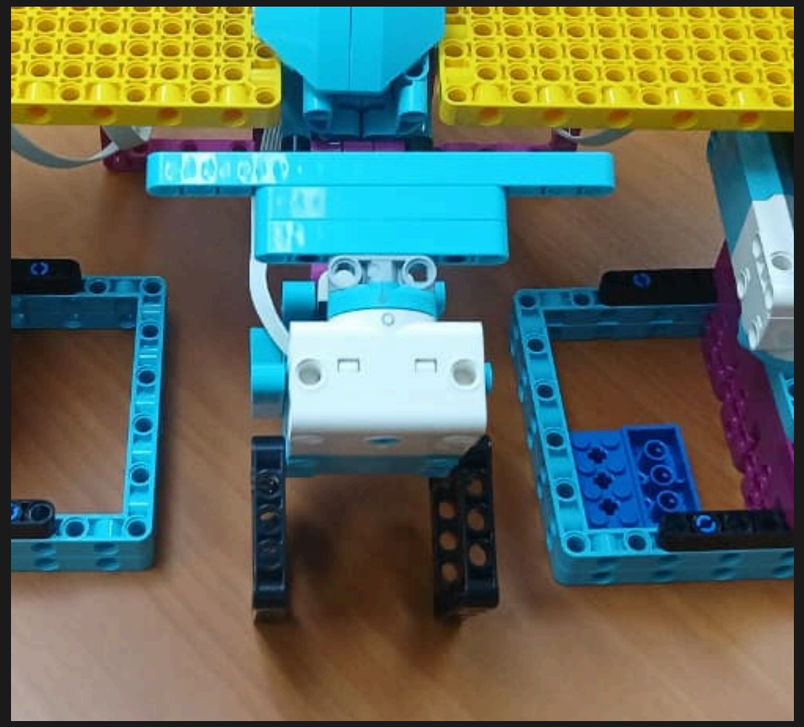
Velcidad mínima y máxima para llegar al intervalo de 4cm - 7cm

$$v_b = \frac{x_{min}}{t} \approx \frac{0.04}{0.1278} \approx 0.313\ m/s$$

$$v_b = \frac{x_{max}}{t} \approx \frac{0.07}{0.1278} \approx 0.547\ m/s$$

FUNDAMENTOS DE MOVIMIENTO

Movimiento Rotación de Clasificador:



Velocidad del momento antes del impacto

$$v_b = \frac{2M_{eff}}{M_{eff} + m} v_i$$

Velcidad mínima y máxima de contacto entre la barra y el bloque para alcanzar la distancia de 4cm y 7 cm

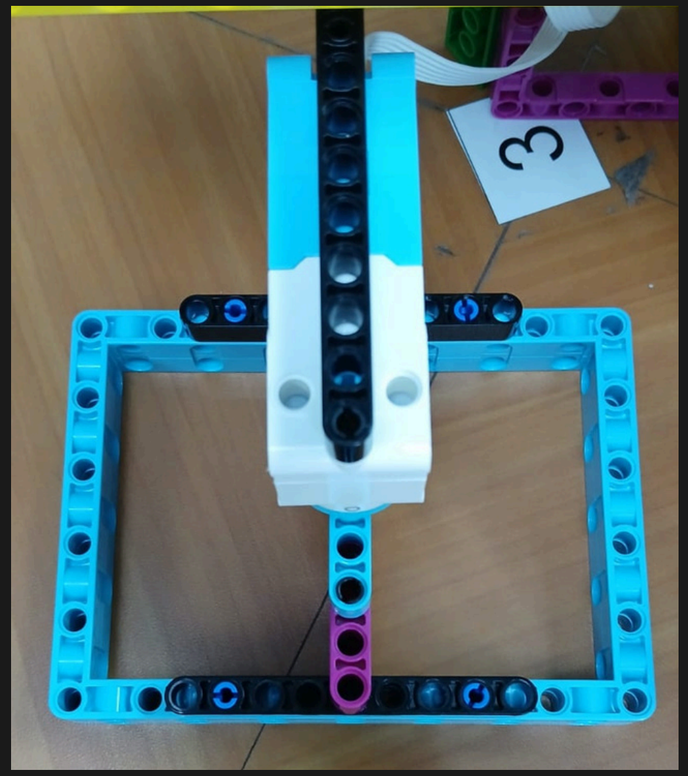
$$v_i^{min} \approx \frac{0.313}{0.669} \approx 0.468\ m/s$$

$$v_i^{max} \approx \frac{0.547}{0.669} \approx 0.818\ m/s$$

Velocidad aplicada actualmente 600 grados/ segundos aproximadamente 0.534 m/s

FUNDAMENTOS DE MOVIMIENTO

Movimiento Rotación de Contenedores:



Velocidad máxima del motor

$$\omega_{max} = 135\ RPM = 14.137\ rad/s$$

Velocidad angular máxima

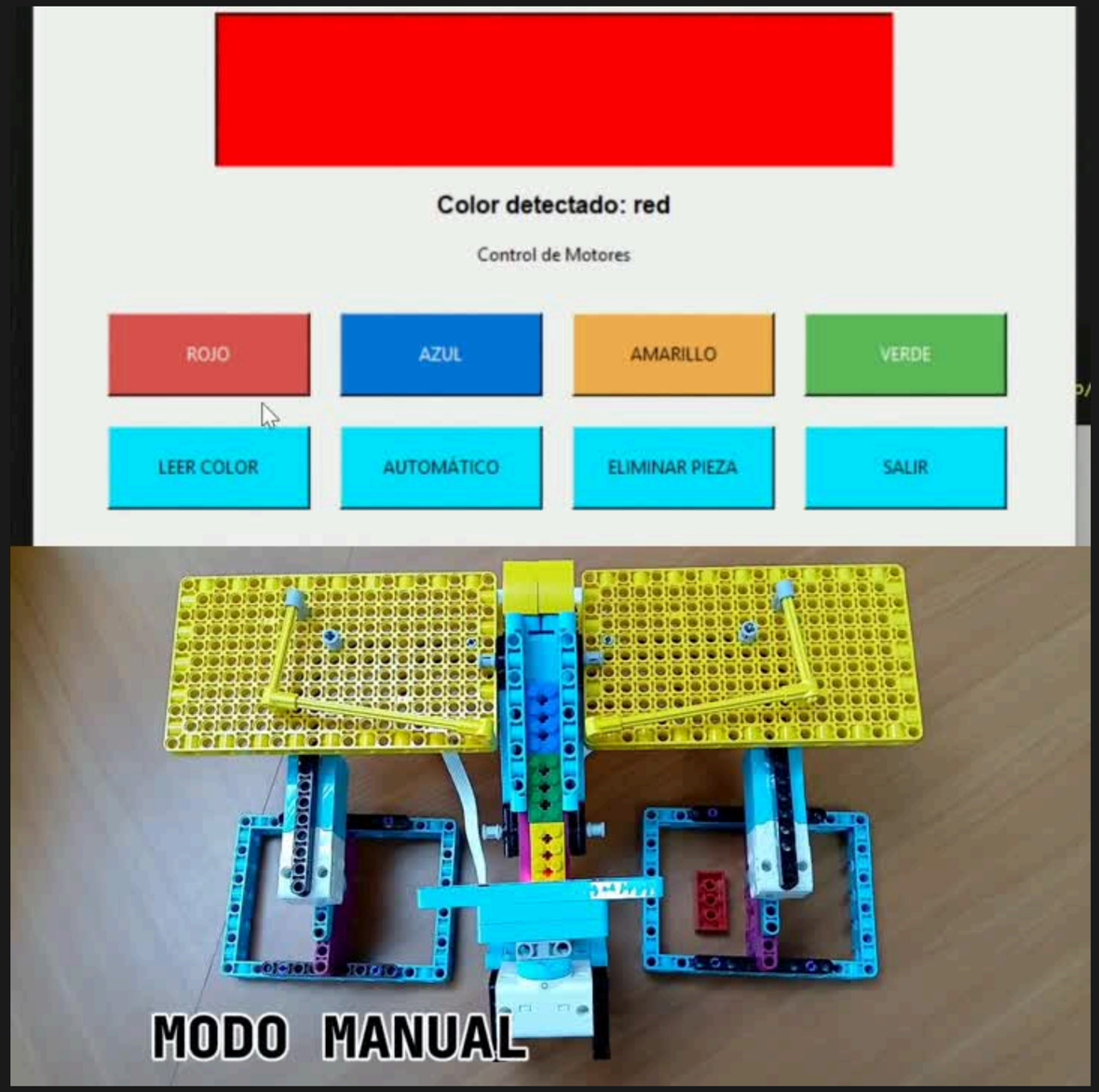
$$\alpha = \frac{\tau}{I} = \frac{3.5}{5.763\times10^{-5}} \approx 60750\ rad/s^2$$

Velocidad óptima de giro del motor

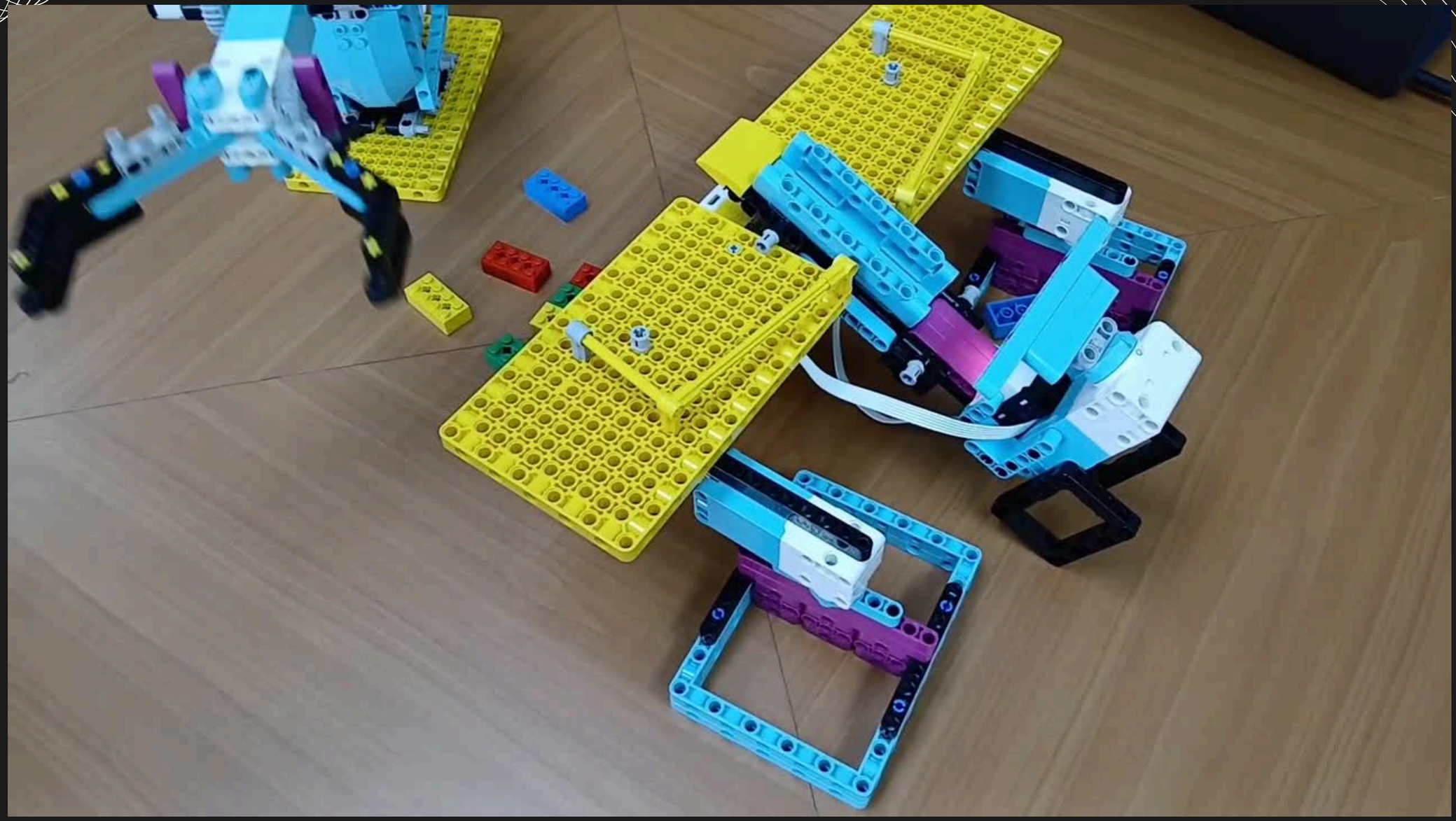
$$v_{max} = \omega_{max} * r = 14.137 * 0.44 \approx 0.622\ m/s$$

Velocidad aplicada actualmente 600 grados/ segundos aproximadamente 0.534 m/s

VIDEO DEMO DEL PROTOTIPO



VIDEO DEMO DEL PROTOTIPO



RESULTADOS

FINALIZADAS LA PRUEBAS SE DETECTARON DOS FALLOS PRINCIPALES:

Cuando las piezas son depositadas y llegan a la parte inferior, si presentan un tono oscuro, el sensor no las detecta adecuadamente y las interpreta como color negro, el cual no está considerado entre los colores definidos (rojo, amarillo, azul y verde). Esto ocurre porque el sensor requiere una distancia mínima de 3 mm para que la luz proyectada se refleje y permita detectar el color correcto.

FALLA 1

Al finalizar la operación o si se deja de usar por un prolongado periodo de tiempo, solo se desconecta el computador, mientras que el hub permanece en un estado que requiere ser reiniciado para poder reconectarse.

FALLA 2

Exceptuando estos errores, se observa un funcionamiento óptimo del prototipo tanto en la clasificación manual como automática, aunque esta última está limitada a cinco iteraciones. Se cumplió con los requisitos definidos, logrando detectar las piezas mediante el sensor de color y depositarlas en sus contenedores correspondientes, todo ello a través de una interfaz de usuario que permite el control remoto del prototipo.

MANUAL DE USUARIO

README

Manual de Usuario

Sorting Mining: Clasificador Remoto de Minerales para la Optimización y Seguridad en Minería

Descripción

Sorting Mining es un prototipo diseñado para clasificar minerales representados por piezas de colores mediante un sensor de color. El sistema busca optimizar la clasificación, mejorar la precisión y aumentar la seguridad operacional a través del control remoto y la automatización.

Características

- Detecta piezas por color a través de un **sensor de color**.
- Clasifica las piezas en **contenedores específicos** según el color detectado.
- Permite **control remoto** del prototipo vía **Bluetooth**.
- Dispone de un **modo automático** de clasificación de piezas.
- Incluye una **interfaz gráfica de usuario (GUI)** para el manejo del prototipo.
- Soporta **casos particulares de piezas de colores** fuera del rango estándar.

Requerimientos necesarios

- **Sistema operativo:** Windows 11
- **Conectividad:** Bluetooth

Pasos de instalación y configuración

1. **Instalar dependencias**
   - Descargar e instalar **Python 3.12.4.** https://www.python.org/downloads/release/python-3124/
   - Descargar e instalar la **última versión de Visual Studio Code (VS Code).** https://code.visualstudio.com/download
   - Descargar e instalar la **última versión de PyBricks** y sus extensiones correspondientes en VS Code. https://pybricks.com/learn/getting-started/install-pybricks/
2. **Preparar el entorno del dispositivo**
   - Formatear el **Spike Prime HUB** utilizando PyBricks.
3. **Obtener y abrir el código fuente**
   - Descargar el **código fuente** desde el repositorio de GitHub.
   - Guardar el proyecto en una carpeta de fácil acceso.
   - Abrir esa carpeta dentro de **Visual Studio Code**.
4. **Establecer conexión Bluetooth**
   - Activar la **conexión Bluetooth** en el PC.
   - Encender el **HUB** y verificar que el **botón de Bluetooth** esté **encendido y parpadeando**.
5. **Ejecutar la interfaz gráfica**
   - Compilar el archivo correspondiente a la **interfaz gráfica**.
   - Dentro de la GUI, **conectarse con el HUB** mientras el botón de Bluetooth siga parpadeando.
6. **Pruebas y ejecución**
   - Realizar las **pruebas funcionales necesarias.**
   - Iniciar la **ejecución de las tareas de clasificación automática.**

CONCLUSIÓN

Se implementó con éxito un prototipo robótico funcional (Lego Spike Prime) capaz de identificar y clasificar materiales por color, cumpliendo con el 100% de los objetivos iniciales.

LOGRO DEL OBJETIVO

El sistema permite la clasificación remota mediante una arquitectura Cliente-Servidor, cumpliendo el propósito de preservar la integridad del trabajador y digitalizar procesos mineros.
Se integraron funciones avanzadas de descarte de materiales no deseados y una interfaz gráfica intuitiva para el control total del dispositivo.

SEGURIDAD Y ROBUSTEZ

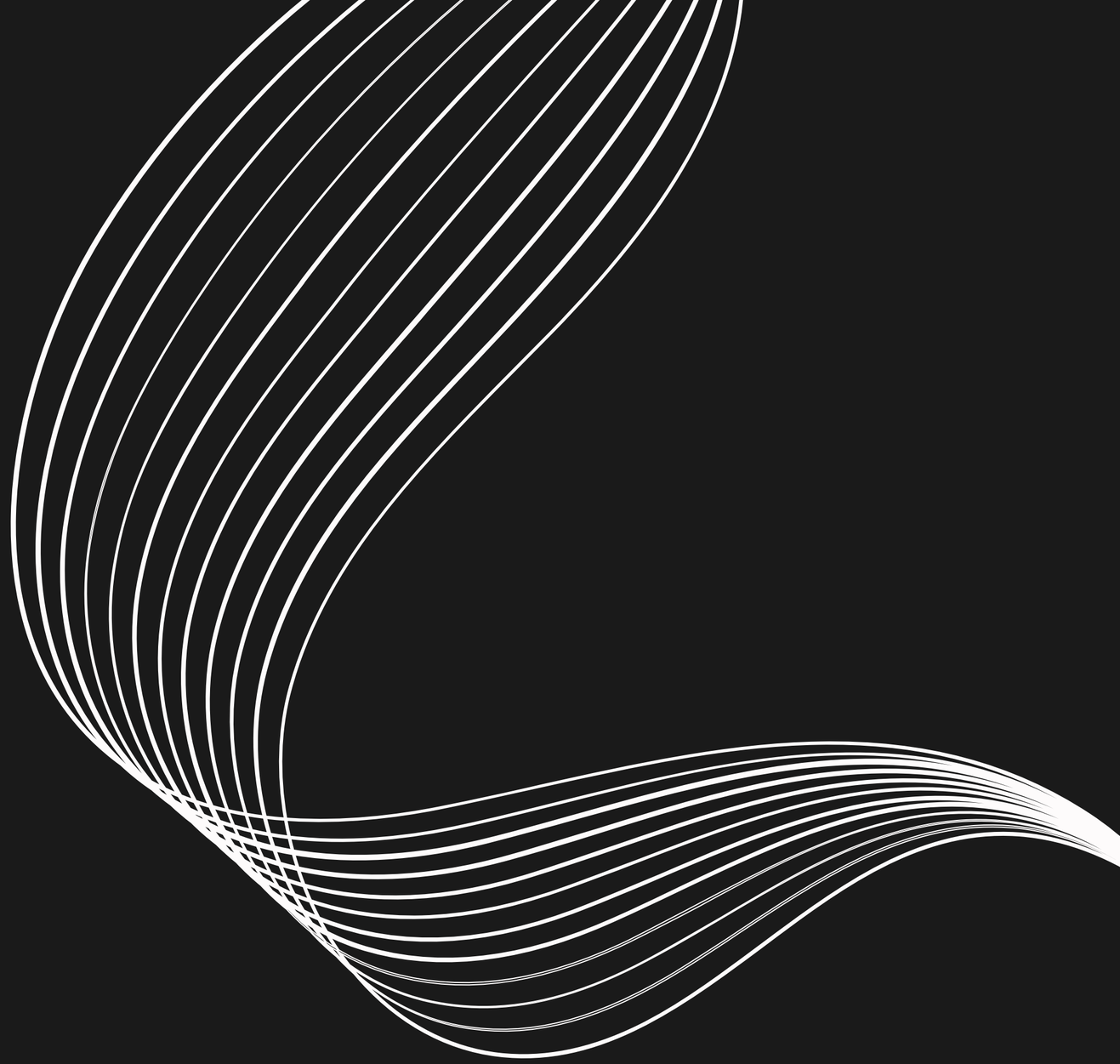
La arquitectura soporta flujos de trabajo masivos.
Las limitaciones actuales (como el ciclo de 5 piezas) son parámetros de seguridad, fácilmente ajustables para una operación continua.

ESCALABILIDAD

VISIÓN A FUTURO

El proyecto consolidó competencias críticas en trabajo en equipo, gestión de proyectos formales y resolución de problemas técnicos complejos.
El camino queda abierto para expansiones, como el control manual individualizado de cada motor o la implementación de asistencia por voz en la interfaz, etc.

MUCHAS GRACIAS



SORTING MINING:
CLASIFICADOR REMOTO DE MINERALES PARA LA OPTIMIZACIÓN Y SEGURIDAD EN MINERIA

Alumnos:

- Camilo Geraldo
- Ignacio Cuevas
- Maximiliano Burgos
- Jose Quispe
- Matias Sagredo

Asignatura:

- Proyecto I

Profesor:

- Baris Nikolai Klobertanz Quiroz