



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





INDICE

01

**Análisis y
Diseño**

02

Implementación

03

Resultados

01

ANÁLISIS Y DISEÑO

REQUERMIENTOS 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).

01

ANÁLISIS Y DISEÑO

REQUERMIENTOS FUNCIONALES

RF5

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

RF6

Gestión de parada: El sistema debe permitir al operador detener la operación del prototipo de forma inmediata 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.

01 ANÁLISIS Y DISEÑO

REQUERMIENTOS 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.

01 ANÁLISIS Y DISEÑO

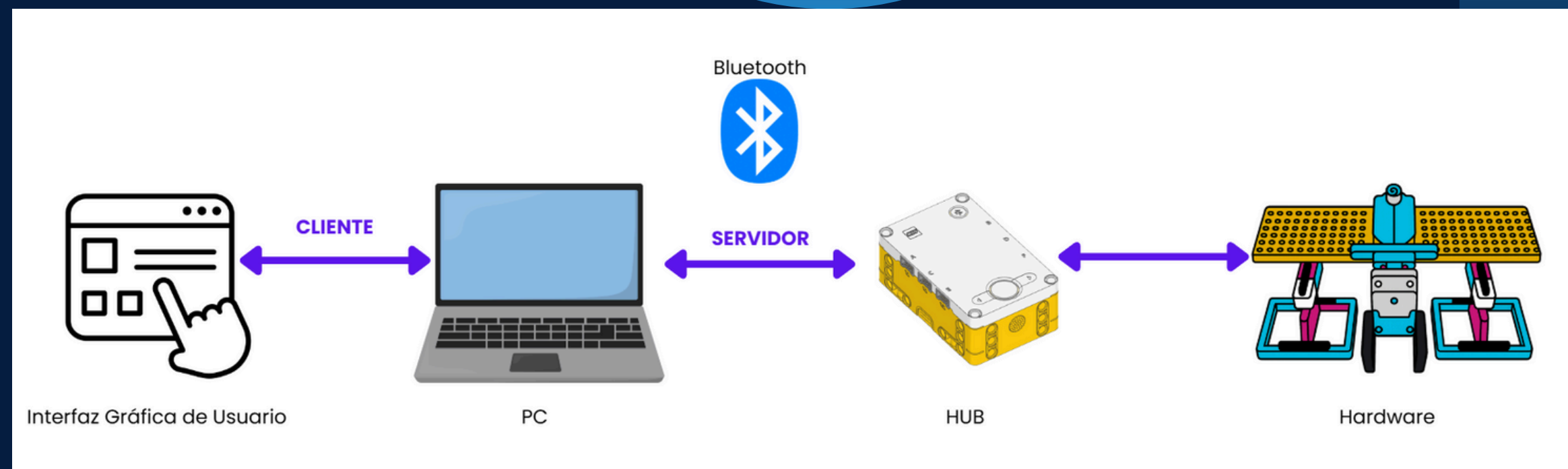
REQUERMIENTOS NO FUNCIONALES

- **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.

01 ANÁLISIS Y DISEÑO

ARQUITECTURA DE SOFTWARE

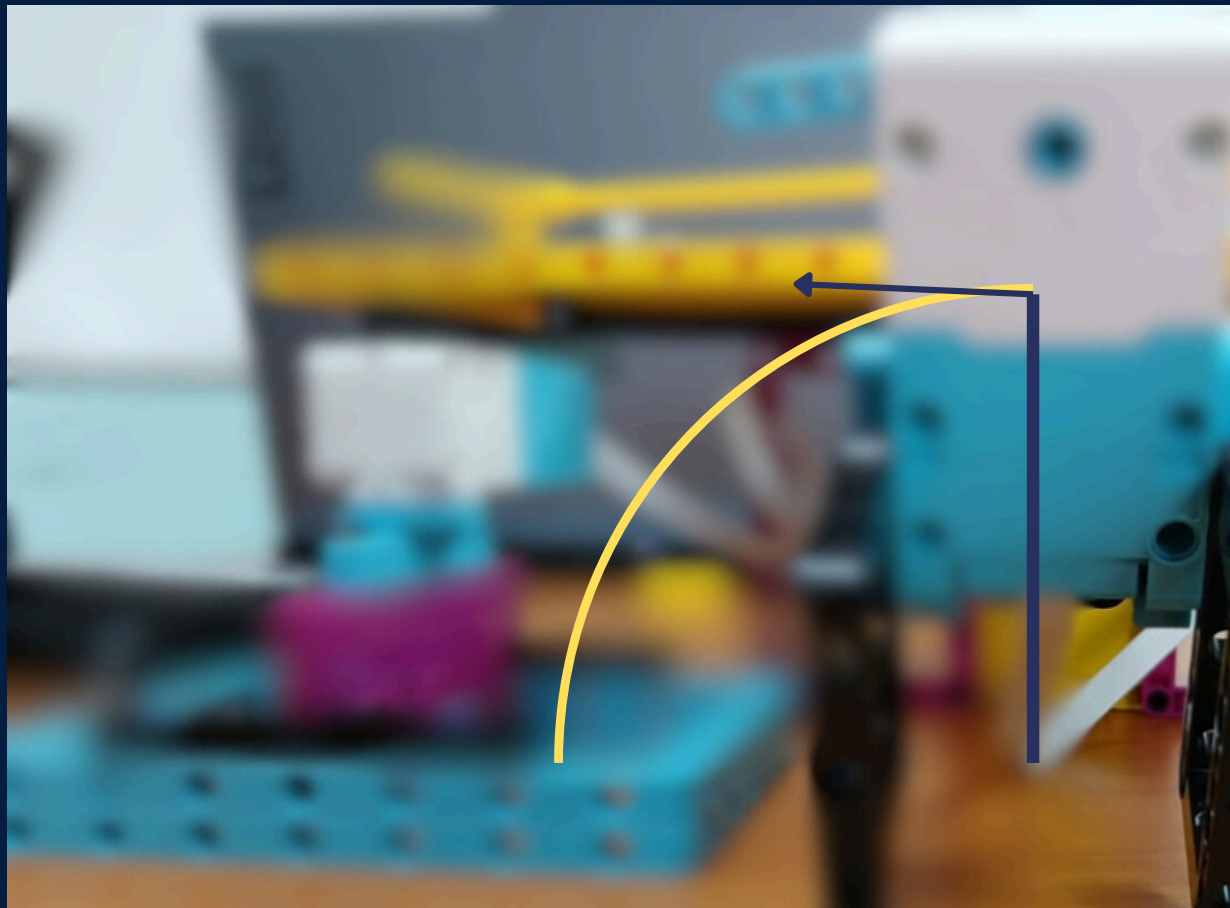
La arquitectura del sistema se basa en un modelo Cliente-Servidor, diseñado para permitir el control remoto y la supervisión del proceso de clasificación de minerales. Esta estructura separa la gestión del usuario de la ejecución física de las tareas en el prototipo.



02 IMPLEMENTACIÓN

FUNDAMENTOS DE MOVIMIENTO

Movimiento Rotación de Clasificador:



Tiempo de caída:

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

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

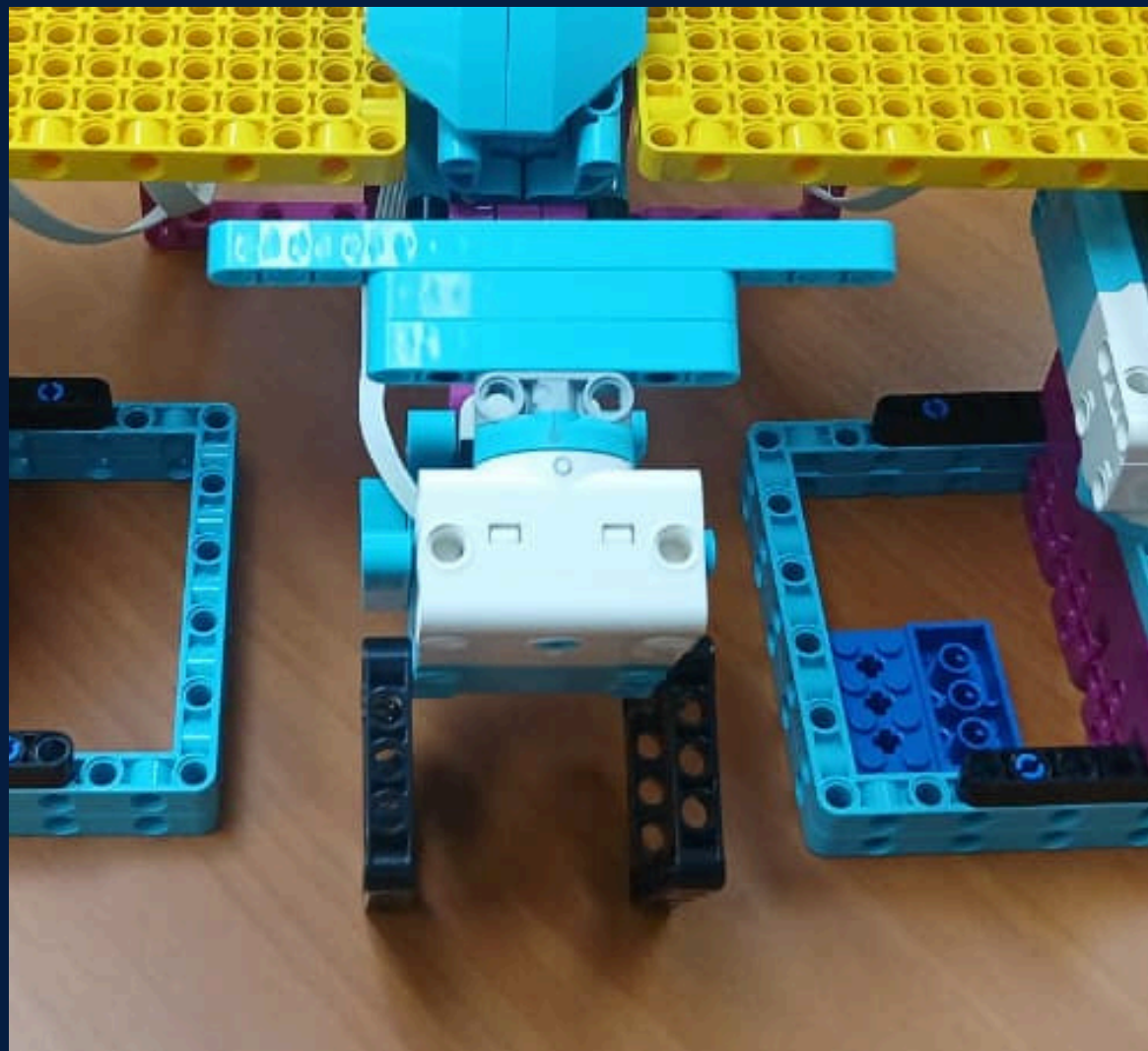
$$v_b = \frac{x_{min}}{t} \approx \frac{0.04}{0.1278} \approx 0.313 \text{ m/s}$$

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

02 IMPLEMENTACIÓN

FUNDAMENTOS DE MOVIMIENTO

Movimiento Rotación de Clasificador:



Velocidad del momento antes del impacto

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

$$M_{eff} = \frac{M}{3} = \frac{0.0035}{3} \approx 0.0011667 \text{ kg}$$

$$m = 2.32 \text{ g} = 0.00232 \text{ kg}$$

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

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

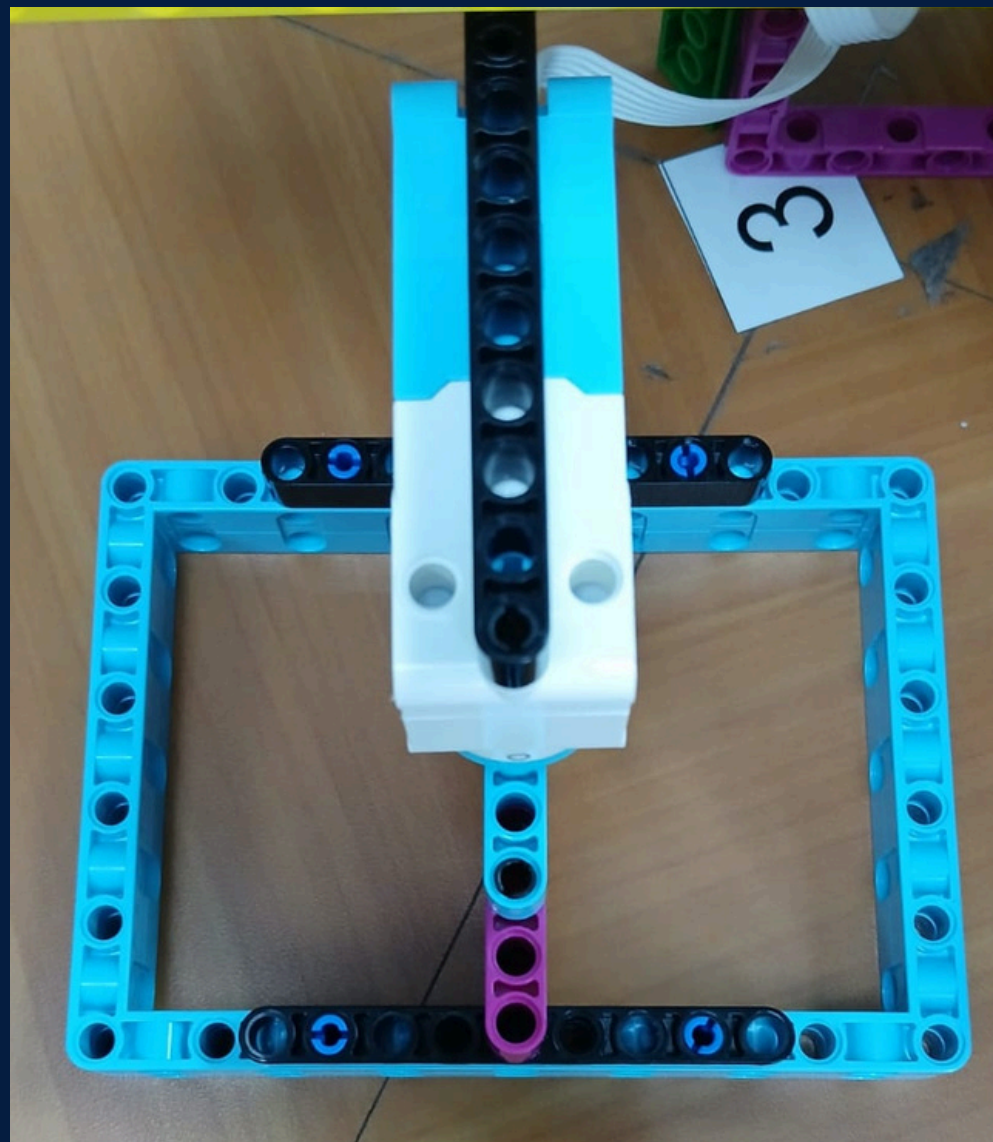
$$v_i^{max} \approx \frac{0.547}{0.669} \approx 0.818 \text{ m/s}$$

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

02 IMPLEMENTACIÓN

FUNDAMENTOS DE MOVIMIENTO

Movimiento Rotación de Contenedores:



Velocidad máxima del motor

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

Velocidad angular máxima

$$\alpha = \frac{\tau}{I} = \frac{3.5}{5.763 \times 10^{-5}} \approx 60750 \text{ rad/s}^2$$

Velocidad óptima de giro del motor

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

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

02 IMPLEMENTACIÓN CLIENTE

El cliente es la aplicación de control utilizada por el operador minero, ejecutada en un PC. Está desarrollada en Python con una interfaz gráfica en Tkinter, y permite la interacción directa con el sistema de clasificación. A través de esta interfaz, el operador puede realizar las siguientes funciones:

- Gestión de conexión: Iniciar y finalizar la comunicación inalámbrica con el servidor (HUB).
- Modo automático (RF3): Activar o desactivar la clasificación autónoma de piezas.
- Control manual (RF4): Operar remotamente el motor principal y los contenedores.
- Monitoreo de sensores (RF5): Visualizar el color detectado por el sensor del prototipo.
- Parada de emergencia (RF6): Detener de forma inmediata cualquier operación en ejecución.

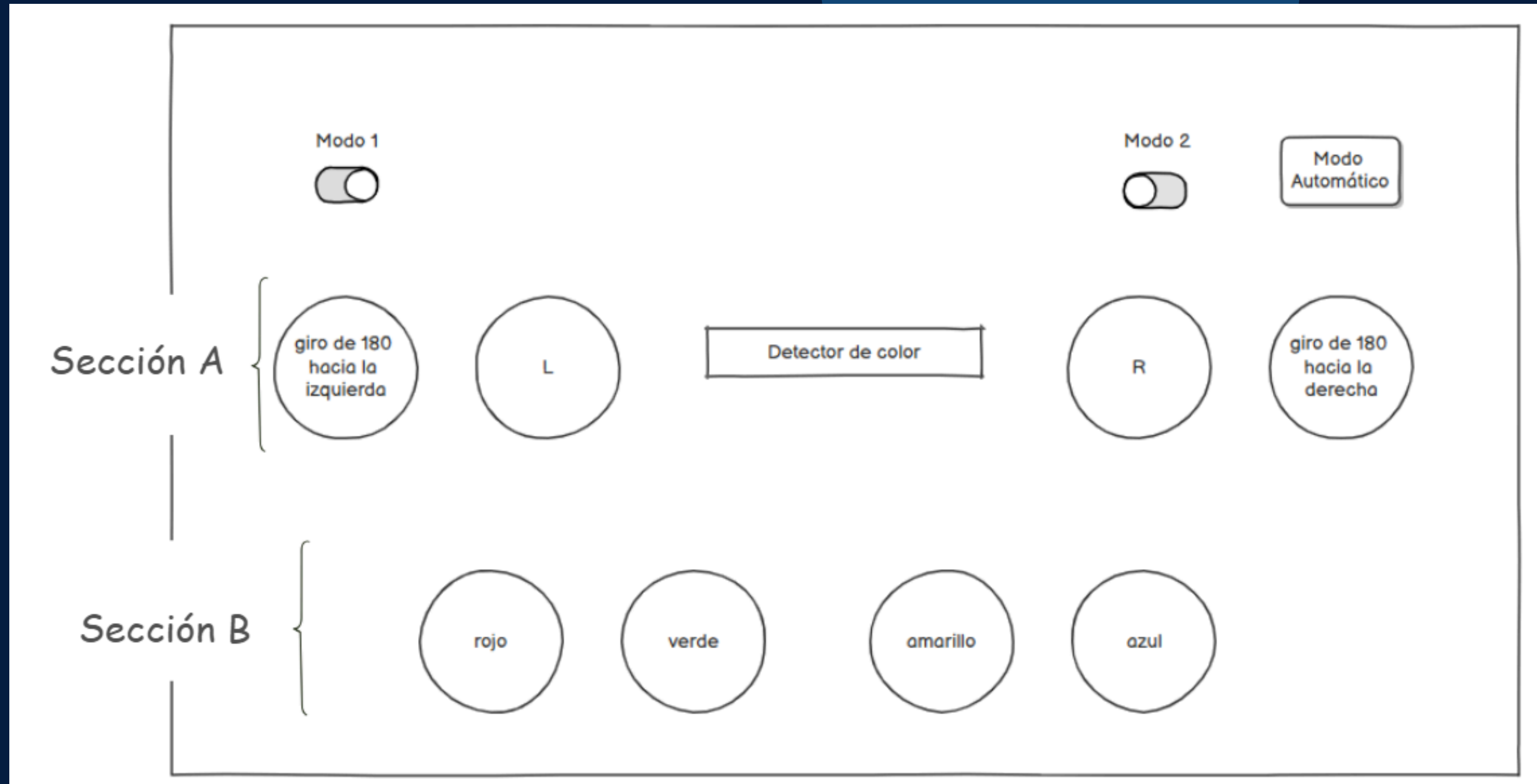
02 IMPLEMENTACIÓN SERVIDOR

En esta arquitectura, el servidor se analiza desde dos perspectivas complementarias que permiten la ejecución de las tareas de clasificación:

- Servidor Físico: Corresponde al LEGO Spike Prime HUB. Es el hardware que ejecuta el firmware de Pybricks para ofrecer los servicios de control de motores y lectura de sensores.
- Servidor Lógico: Es el componente de software encargado de representar al hardware y gestionar la comunicación técnica con él. Este reside parcialmente en el computador del cliente y utiliza la librería `pybricksdev` para enviar instrucciones al HUB mediante la conexión Bluetooth establecida.

02

IMPLEMENTACIÓN WIREFRAME



02 IMPLEMENTACIÓN

GUI

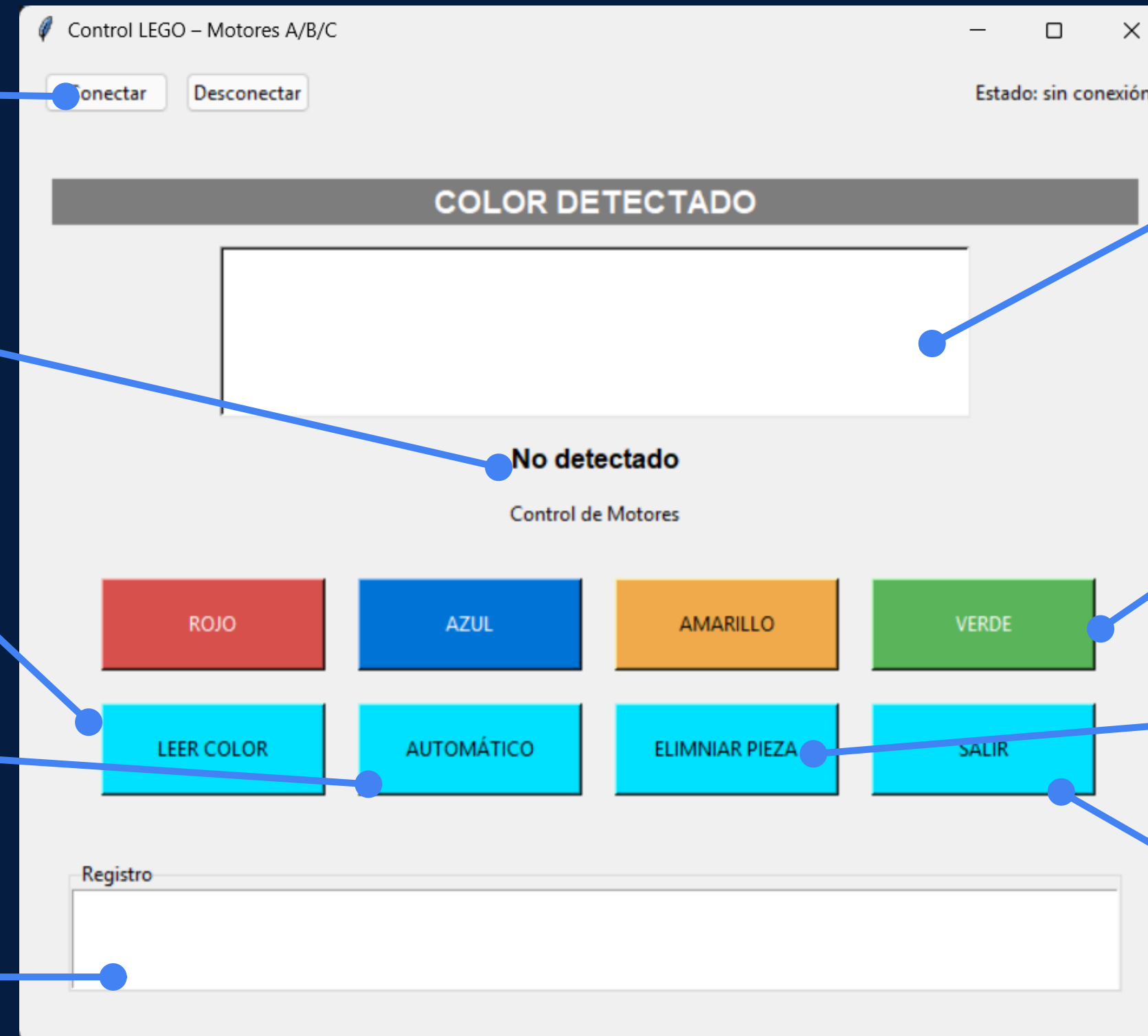
Inicia/Finaliza la conexión vía bluetooth entre el sistema en el computador y el hub Spike Prime.

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

Inicia 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.

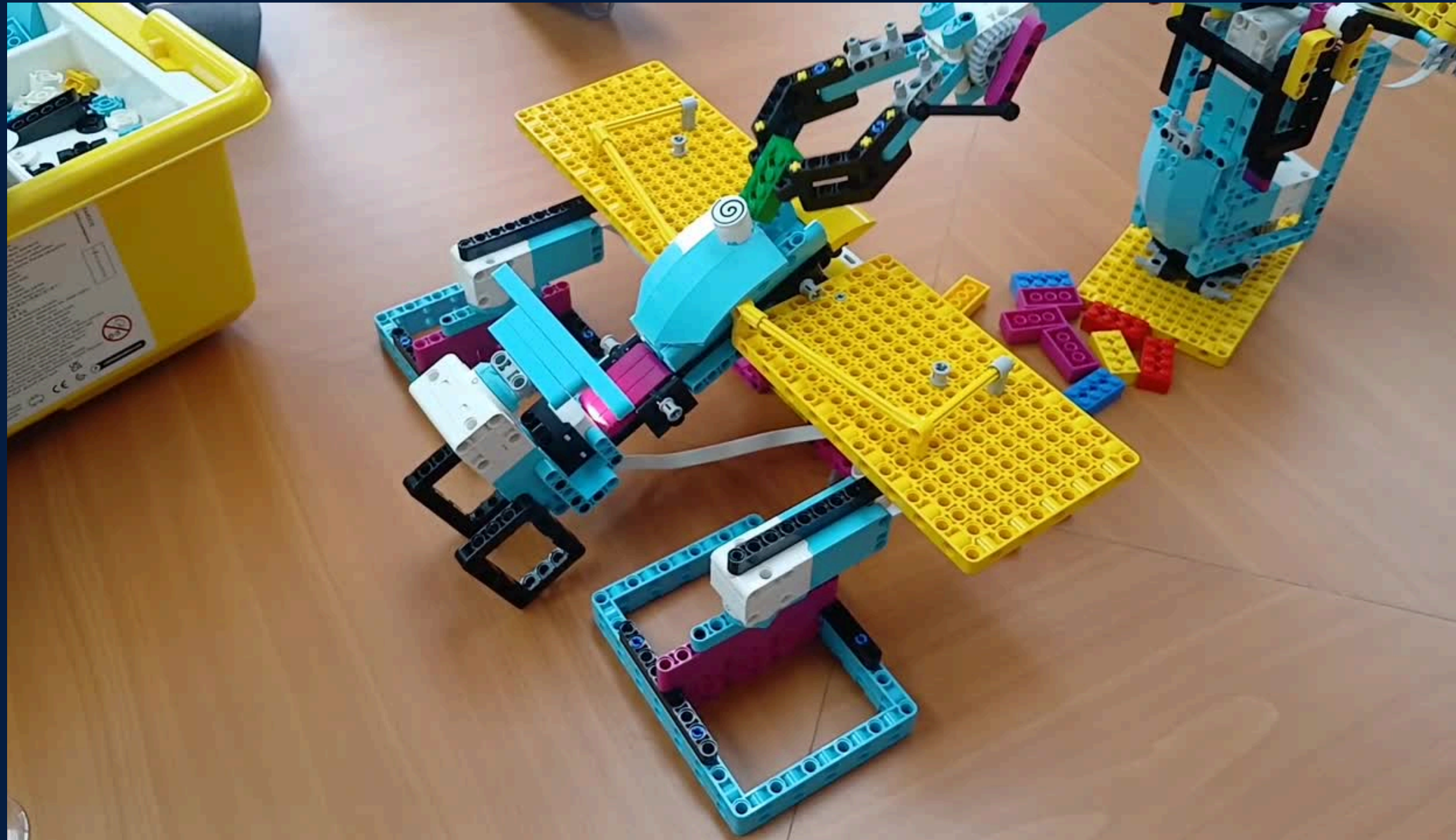
Botones que clasifican las piezas en los contenedores correspondientes.

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.

03 RESULTADOS

ESTADO ACTUAL



03 RESULTADOS

PROBLEMAS

01

Durante el ensamblaje se detectó que los bloques ingresaban al sistema en posiciones irregulares, lo que dificultaba su correcta toma por la garra y provocaba fallos en la clasificación, reduciendo la confiabilidad del sistema.

02

Se detectó que el sistema realizaba movimientos antes de finalizar la lectura del sensor de color, lo que provocaba errores en la clasificación de algunos bloques.

03

Se presentó una dificultad clave al integrar el HUB con la interfaz en Tkinter vía Bluetooth, lo que detuvo el avance del proyecto durante dos semanas por falta de una conexión funcional.

SOLUCIONES

Se rediseñó la entrada de los bloques, guiándolos de tal manera que solo pudieran ingresar en posición vertical, estandarizando su orientación antes de ser clasificados.

Se reestructuró la lógica de control, incorporando puntos de verificación intermedios y reorganizando el orden de los comandos, con el fin de asegurar que cada etapa del proceso de clasificación

La solución se logró mediante el apoyo puntual de un compañero externo al equipo de trabajo quien nos colaboró en la implementación inicial de la conexión entre el HUB y la librería Tkinter

CONCLUSION

En conclusión, las correcciones realizadas para esta segunda presentación se centraron en complementar de mejor manera los puntos establecidos, como los requerimientos funcionales y la arquitectura de software.

Se avanzó considerablemente en la interfaz gráfica de usuario, pasando del código en bloques a código en Python, empleando Tkinter para la parte visual y Pybricks para la lógica y la comunicación con el hub.

Finalmente, se solucionaron los problemas que fueron apareciendo, tales como la codificación de la interfaz gráfica y la conexión vía Bluetooth entre el hub y la pc.

GRACIAS

