

Universidad de Tarapacá

**Universidad de Tarapacá**



**UNIVERSIDAD DE TARAPACÁ**  
*Universidad del Estado*

**Facultad de Ingeniería**

Departamento de Ingeniería en Computación e Informática



**Proyecto 1: Robot Garra con LEGO Spike Prime**

Integrantes:

Guillermo Contreras

Ariel Colque

Daniel Flores

Jhilmar Solares

Fernanda Tobar

CC091: Proyecto 1

Profesor: Baris Klobertanz

**Arica, Chile**

26 de septiembre, 2025

**Tabla 1**

*Historial de versiones del documento*

Fecha	Versión	Descripción	Autores
22/09/2025	1.0	Concepción inicial del documento	Jhilmar Solares
29/09/2025	1.1	Elaboración del primer prototipo	Jhilmar Solares
01/10/2025	1.2	Ampliación y mejora de la tabla de contenidos	Jhilmar Solares
03/10/2025	1.3	Desarrollo de la sección de restricciones y gestión de riesgos	Jhilmar Solares
13/10/2025	1.4	Mejora de la sección de organización proporcional	Jhilmar Solares
15/10/2025	1.5	Extensión y revisión de la bibliografía	Jhilmar Solares
15/10/2025	1.6	Completación de la sección de gestión de riesgos	Jhilmar Solares
17/10/2025	1.7	Finalización de la tabla estimación de costos	Jhilmar Solares
17/10/2025	1.8	Corrección del formato APA y validación final del documento	Jhilmar Solares
17/10/2025	1.9	Redacción y finalización de la conclusión del informe, incorporando el cierre y proyección del proyecto.	Jhilmar Solares
12/11/2025	1.10	Incorporación de los cambios de roles que se hicieron en el equipo.	Ariel Colque
14/11/2025	1.11	Actualización en el cronograma de actividades para la fase 2.	Ariel Colque

## Tabla de contenidos

<b>Panorama general.....</b>	<b>5</b>
1.1 Introducción.....	5
1.2 Objetivo general.....	6
1.3 Objetivos específicos.....	6
1.4 Restricciones.....	7
1.5 Entregables.....	8
<b>Organización del Personal.....</b>	<b>9</b>
2.2 Personal Designado Fase 1.....	9
2.3 Personal Designado Fase 2.....	10
<b>Planificación del proyecto.....</b>	<b>11</b>
3.1 Cronograma de actividades Fase 1.....	11
3.2 Cronograma de actividades Fase 2.....	12
3.3 Carta Gantt.....	13
3.4 Gestión de Riesgos.....	14
<b>Estimación de Costos.....</b>	<b>16</b>
4.1 Hardware.....	16
4.2 Software.....	17
4.3 Costo por rol.....	18
<b>Análisis–Diseño.....</b>	<b>18</b>
5.1 Especificación de Requerimientos.....	18
5.2 Arquitectura.....	20
5.3 Interfaz gráfica del sistema.....	21
<b>Conclusión.....</b>	<b>25</b>
<b>Referencias.....</b>	<b>25</b>
6.1 Fuentes de compra.....	26
6.2 Fuentes de información.....	26



## Tabla de tablas

Tabla 1.....	1
Historial de versiones del documento.....	1
Tabla 2.....	8
Personal designado fase 1.....	8
Tabla 3.....	9
Personal designado fase 2.....	9
Tabla 4.....	10
Cronograma de actividades fase 1.....	10
Tabla 5.....	11
Cronograma de actividades fase 2.....	11
Tabla 6.....	14
Gestión de riesgos del proyecto.....	14
Tabla 7.....	16
Costo de hardware.....	16
Tabla 8.....	16
Costo de software.....	16
Tabla 9.....	17
Costo por rol.....	17

## Panorama general

### 1.1 Introducción

En este proyecto se presenta un acercamiento a un escenario real: la minería moderna exige soluciones vanguardistas, sustentables y además seguras. Innovaciones como la teleoperación, la automatización y la robótica han reducido el riesgo para los trabajadores. Accidentes en Grasberg (Indonesia) y El Teniente (Chile) evidencian la urgencia de proteger la vida humana. Este proyecto aborda la seguridad mediante la implementación de la robótica.

Se dispondrá un kit LEGO Spike Prime. El robot será operado de forma remota por un dispositivo cliente con interfaz gráfica de usuario (GUI), que permitirá enviar órdenes al hub del robot, el cual actuará como un servidor. El enlace con el hub se realizará vía Bluetooth, y para el desarrollo se usará el lenguaje Python.

Nuestro objetivo será crear un brazo robótico con garra denominado: Spike, capaz de tomar, levantar y trasladar objetos de diferentes formas y pesos. Para la comunicación con el microcontrolador del hub utilizaremos la librería Pybricksdev, que permite conectarse por USB o Bluetooth, enviar programas Python, y controlar motores, sin depender de la app de LEGO. Elegimos Thonny como nuestro entorno de desarrollo y Tkinter para crear la interfaz gráfica.

La planificación se apoyará en una carta Gantt que mostrará actividades y plazos a lo largo del semestre, además de reuniones y bitácoras semanales para registrar los avances del equipo.

Esperamos que el proyecto contribuya a la seguridad minera y cumpla con estándares del sector. Además, nos permitirá adquirir conocimientos sobre construcción y programación de (Spike), y desarrollar habilidades de trabajo colaborativo, administración del tiempo y documentación mediante bitácoras y seguimiento constante.

## 1.2 Objetivo general

Diseñar, construir y programar un robot manipulador móvil, basado en LEGO SPIKE Prime y una arquitectura cliente–servidor con GUI, capaz de sujetar, elevar y trasladar objetos, para apoyar actividades formativas vinculadas a procesos de la minería moderna.

## 1.3 Objetivos específicos

- Experimentar con los componentes del kit LEGO Education SPIKE Prime para comprender el funcionamiento de sus motores, sensores y piezas estructurales.
- Diseñar la estructura del robot optimizando su equilibrio, movilidad y resistencia al peso de los objetos transportados.
- Construir una garra mecánica capaz de abrir, cerrar y mantener un agarre firme sobre objetos de distintas formas y tamaños.
- Probar y ajustar los movimientos del robot mediante iteraciones que mejoren la precisión del agarre y la eficiencia del desplazamiento.
- Evaluar el desempeño del robot en diferentes condiciones (peso del objeto, superficie de trabajo, velocidad de movimiento).
- Documentar el proceso de diseño, ensamblaje y programación para dejar registro técnico del desarrollo del proyecto.
- Comprobar el cumplimiento de los objetivos mediante pruebas que verifiquen la capacidad del robot para levantar y trasladar correctamente los objetos asignados.



## 1.4 Restricciones

Corresponden a los límites, condiciones o factores que restringen la forma en que será llevada a cabo la solución para este proyecto.

- Se debe programar exclusivamente en Python.
- Uso exclusivo del set de Lego spike prime y el set de expansión para la construcción del robot.
- Entrega del proyecto en la fecha establecida.
- El robot no se puede trasladar fuera del establecimiento.
- La documentación y seguimiento del proyecto debía realizarse exclusivamente a través de la plataforma Redmine.
- La garra debe ser capaz de agarrar, levantar y mover un objeto.



## 1.5 Entregables

- Bitácoras: Son informes semanales que describen el avance del equipo en el proyecto, abarcando actividades realizadas, dificultades encontradas, recomendaciones para mejorar y acciones tomadas. Preparadas por un individuo designado, ofrecen un panorama exhaustivo para apoyar decisiones estratégicas, asignan responsabilidades y resaltan asuntos a tratar en grupo.
- Carta Gantt: Representación visual de la programación del proyecto, mostrando en una línea de tiempo las tareas, su duración y secuencia, facilitando la gestión del tiempo y los recursos al visualizar la evolución de las actividades a lo largo del proyecto.
- Informe de Formulación: Este documento detalla nuestra organización y estrategia para alcanzar los objetivos de la asignatura. Abordaremos la asignación de roles, las metas del equipo y las medidas que implementaremos para lograr el propósito académico. Además, compartiremos nuestras primeras impresiones durante el proceso de desarrollo y presentaremos la documentación relevante recopilada a lo largo del semestre.
- Presentaciones: Se detallan los objetivos del proyecto, los retos superados y las soluciones aplicadas. También se resaltan los éxitos obtenidos, la distribución del equipo y se ofrece una visión general del robot.

## Organización del Personal

En esta sección se define el personal asignado al proyecto, sus roles, sus responsabilidades y su dedicación semanal. También se establecen las competencias mínimas por puesto, la línea de reporte y los hitos bajo su cargo. Precisa sustituciones ante ausencias y los criterios de disponibilidad. Sirve como base para planificación, control de la carga y evaluación del desempeño.

### 2.2 Personal Designado Fase 1

**Tabla 2**

#### *Personal designado fase 1*

Rol	Descripción del Rol	Responsable
Jefe de proyecto	Coordina el equipo, organiza tareas y supervisa el cumplimiento de los objetivos del proyecto.	Guillermo Contreras
Ensamblador	Construye el robot utilizando el kit LEGO Spike Prime, asegurando su estabilidad y funcionamiento.	Ariel Colque
Programador	Desarrolla y ajusta el código del robot, configurando sensores y motores.	Daniel Flores
Documentador	Registra los avances del proyecto y redacta los informes técnicos.	Jhilmar Solares
Diseñadora	Diseña el material visual y la presentación final del proyecto.	Fernanda Tobar

## 2.3 Personal Designado Fase 2

**Tabla 3**

### *Personal designado fase 2*

Rol	Descripción del Rol	Responsable
Jefe de proyecto	Coordina el equipo, organiza tareas y supervisa el cumplimiento de los objetivos del proyecto.	Jhilmar Solares
Ensamblador	Construye el robot utilizando el kit LEGO Spike Prime, asegurando su estabilidad y funcionamiento.	Daniel Flores
Programador	Desarrolla y ajusta el código del robot, configurando sensores y motores.	Guillermo Contreras y Fernanda Tobar
Documentador	Registra los avances del proyecto y redacta los informes técnicos.	Ariel Colque

### Planificación del proyecto

En este apartado se explica cómo organizaremos el trabajo y los tiempos del proyecto. Muestra las tareas, quién las hará, en qué orden, cuánto durarán y con qué recursos. Incluye riesgos, presupuesto y puntos de control para revisar avances y corregir desvíos.

#### 3.1 Cronograma de actividades Fase 1

**Tabla 4**

##### *Cronograma de actividades fase 1*

Actividad	Responsable (Rol)	Inicio	Término	Duración
Prototipo inicial del robot (boceto y pruebas rápidas)	Jefe de proyecto	22/09/2025	26/09/2025	1 semana
Construcción del robot: estructura y base.	Ensamblador	29/09/2025	03/10/2025	1 semana
Redacción del informe (resultados y conclusiones)	Documentador	22/09/2025	31/10/2025	6 semanas
Bitácoras y seguimiento semanal del equipo	Jefe de proyecto	22/09/2025	31/10/2025	6 semanas
Programación v1: motores y sensores en Pybricks	Programador	06/10/2025	31/10/2025	4 semanas
Investigación y pruebas de librerías (Pybricks/pybricksdev)	Programador	05/10/2025	15/10/2025	2.5 semanas
Pruebas funcionales con prototipo 1 (agarre–elevación–traslado)	Programador	22/09/2025	31/10/2025	3 semanas

### 3.2 Cronograma de actividades Fase 2

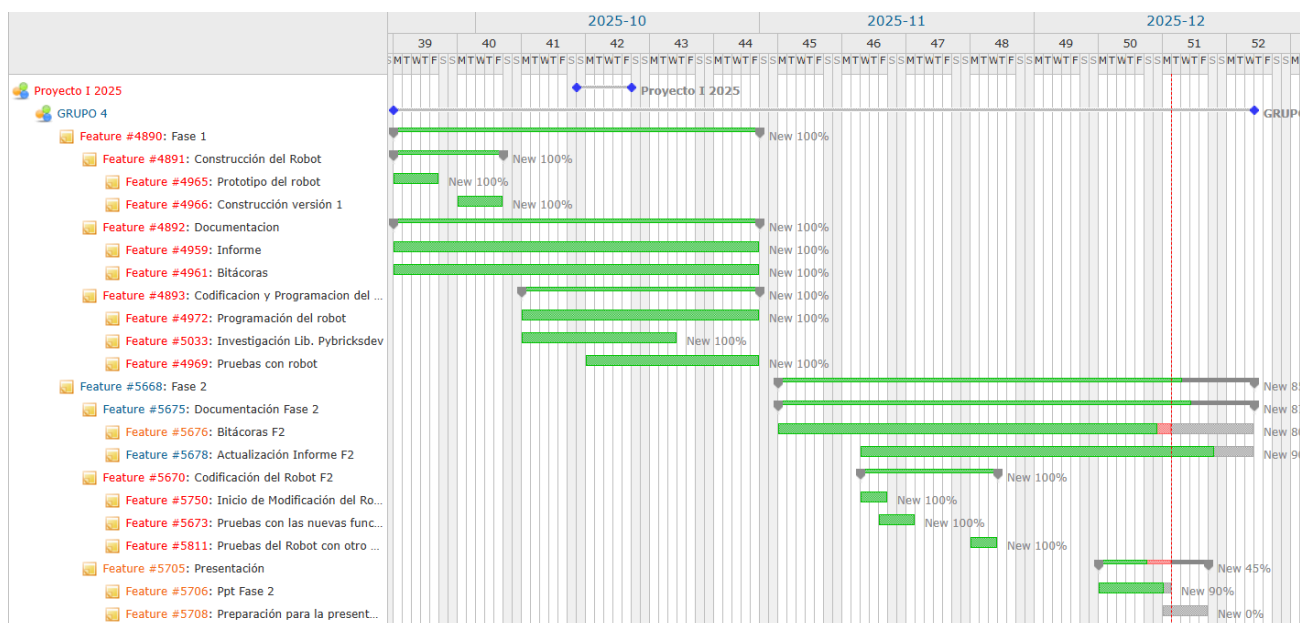
**Tabla 5**

*Cronograma de actividades fase 2*

Actividad	Responsable (Rol)	Inicio	Término	Duración
Bitácoras y seguimiento de la Fase 2	Documentador	03/11/2025	19/12/2025	7 semanas
Actualización del Informe 2 (nuevos resultados y conclusiones)	Documentador	03/11/2025	19/11/2025	5 semanas
Modificación del robot: mejoras de código para desplazamiento homogéneo	Programador	12/11/2025	14/11/2025	3 días
Pruebas de nuevas funciones del robot	Programador	14/11/2025	17/11/2025	3 días
Prueba cruzada del robot con otro equipo	Equipo Completo	24/11/2025	26/11/2025	3 días
Creación del PPT para la presentación final	Equipo Completo	21/11/2025	05/12/2025	2 semanas
Preparación de la presentación (roles y ensayo)	Equipo Completo	08/12/2025	19/12/2025	2 semanas

### 3.3 Carta Gantt

En este apartado se presenta la carta Gantt del proyecto: un cronograma por fases, actividades y entregables con fechas de inicio y fin, con dependencias y responsables. Define la ruta crítica, holguras e hitos de control que sirven como la línea base para el correcto seguimiento de los avances, con la reasignación de recursos y gestión de riesgos.



**Figura 1. Carta gantt del proyecto.**

### 3.4 Gestión de Riesgos

A continuación se presenta una tabla que exhibe un desglose de los problemas que se han presentado a lo largo de la primera fase del proyecto. Esta tabla resume el impacto de cada desafío al clasificar el daño en cinco niveles distintos. Cada nivel está asociado con diferentes tipos de daño:

1. Daño catastrófico: Las medidas a tomar en el caso son de forma inmediata, puede provocar que el proyecto se detenga o retrase significativamente, teniendo que volver a empezar desde cero.
2. Daño crítico: Se deben tomar medidas necesarias para resolver el riesgo, debido a que puede provocar que el proyecto se retrase en varias etapas.
3. Daño circunstancial: El riesgo se debe resolver en el momento, debido a que puede retrasar el desarrollo de una etapa base del proyecto.
4. Daño irrelevante: El riesgo no es de mayor importancia, es un detalle imprevisto que no necesita mucha atención y se puede resolver en cualquier momento.
5. Daño recurrente: El riesgo no es significativo, pero es reiterativo, retrasa en las sesiones de trabajo, pero no en etapas.

**Tabla 6**

*Gestión de riesgos del proyecto*

Riesgo	Nivel de Impacto	Acción Remedial
Pieza rota	Daño circunstancial	Solicitar nuevamente la pieza faltante a los ayudantes, siguiendo el protocolo establecido
Falta de piezas	Daño circunstancial	Solicitar la pieza faltante a los ayudantes.
El robot sufre daños	Daño crítico	Asegurar el robot y realizar pruebas en entornos controlados.
Pérdida de piezas	Daño crítico	Buscar en el área donde se realizó la última manipulación de componentes.
Mala programación	Daño crítico	Revisar tutoriales e implementar soluciones en el código.
Ausencia del personal en el horario de trabajo	Daño crítico	Redistribución temporal de tareas entre el personal disponible para evitar retrasos
Dificultades con la conexión wifi	Daño catastrófico	Verificar la conectividad probando con un dispositivo alternativo



## Estimación de Costos

En esta sección se presenta el cálculo estimado de los recursos necesarios para el desarrollo del proyecto, considerando tanto los materiales utilizados como las herramientas digitales y la participación del equipo de trabajo. Los costos se dividen en tres categorías principales:

- Hardware, que incluye los componentes físicos indispensables para la construcción y funcionamiento del robot.
- Software, donde se consideran las licencias o suscripciones empleadas para la programación y documentación.
- Costo por rol, que estima el valor del tiempo dedicado por cada integrante del equipo según sus funciones y horas de trabajo asignadas.
- El desarrollo del proyecto se llevó a cabo entre el 22 de septiembre de 2025 y el 30 de diciembre de 2025, período durante el cual se realizaron las etapas de diseño, ensamblaje, programación y documentación.

### 4.1 Hardware

Planificación:

- Kit LEGO Spike Prime: Componentes principales para construir la estructura, motores y sensores del robot. Es el núcleo físico del proyecto.
- Notebook Gaming: Herramienta esencial para programar el robot, realizar simulaciones y gestionar toda la documentación del proyecto.
- Mouse Ergonómico: Periférico para facilitar el trabajo prolongado en el diseño y la programación, mejorando la comodidad y eficiencia.

**Tabla 7**

*Costo de hardware*

Producto	Cantidad	Precio(CLP)
Set Lego Spike Prime	1	622.872
Notebook Gaming Victus 16-r1015la	1	1.299.990
Mouse Vertical Ergonómico Inalámbrico Negro	1	12.990
Huawei Intel Core i5	1	479.990
Lenovo N-9 15" AMD Ryzen 7 - 8GB RAM	1	649.990
Dell Inspiron 135301 15"	1	560.000
Acer Aspire 5 A515 15"	1	549.990
Total:	9	4.175.822

## 4.2 Software

Planificación:

- Microsoft 365 Personal: Suite de productividad utilizada para la documentación integral del proyecto (informes en Word, presentaciones en PowerPoint), la organización de datos en Excel y la comunicación y coordinación del equipo.

**Tabla 8**

*Costo de software*

Producto	Cantidad	Precio(CLP)
Microsoft 365 Personal (anual)	1	84.990
Total:	1	84.990

### 4.3 Costo por rol

Los costos por rol se estimaron a partir de las horas planificadas en el cronograma. Para cada actividad se asignó el rol responsable y se convirtieron las duraciones a horas. Las horas acumuladas por rol se multiplicaron por su tarifa horaria de referencia; la suma de estos valores entrega el costo total del proyecto.

Las tarifas horarias de referencia se obtuvieron a partir del portal de salarios de Indeed (Indeed, s. f.).

**Tabla 9**

*Costo por rol*

Rol	Horas	Precio por hora (CLP)
Jefe de proyecto	27	7.309
Ensamblador	19	3711
Programador	27	7.860
Documentador	25	4.560
Analista tester	22	9.674
Total:	103	467.300

## Análisis–Diseño

### 5.1 Especificación de Requerimientos

#### *Requerimientos Funcionales*

##### *Requerimientos Funcionales*

##### Identificación de Usuario y Cliente:

Usuario: Operador minero. Es la persona que, desde una estación de control, manipula la garra robótica para realizar tareas de recolección o manipulación de materiales en un entorno de simulación minera o en un prototipo a escala.

Cliente: Supervisión de la mina. Es el departamento o equipo de ingeniería que define las necesidades operativas y valida que el prototipo cumpla con los criterios de funcionalidad, seguridad y eficiencia establecidos para su aplicación.

##### Calidad de los Requerimientos Funcionales:

Completo: La lista abarca todos los aspectos críticos del sistema: entrada (comandos), proceso (movimiento, detección, regulación) y salida (agarre/liberación).

Claros: Cada requerimiento está redactado en un formato conciso que describe una capacidad observable y medible, sin ambigüedad.

Accionables y Concretos: Cada uno define una acción o comportamiento específico que los desarrolladores pueden implementar y probar.

*Los requerimientos se alinean directamente con el objetivo de crear un efector final robótico autónomo y controlable para un entorno educativo/competitivo. Responden a necesidades como precisión, seguridad, integración con control externo (RF-04) y retroalimentación (RF-05), que son esenciales para un prototipo funcional y didáctico.*

### Lista de Requerimientos Funcionales (RF)

#### RF-01: Accionamiento Motorizado

La garra debe abrir y cerrar sus pinzas mediante un motor o servomotor controlable.

#### RF-02: Rango de Prensión

Debe poder agarrar objetos cuya dimensión se encuentre dentro del rango de apertura máximo y mínimo físicamente definido de sus pinzas.

#### RF-03: Liberación por Comando

Debe liberar el objeto sujeto al recibir un comando específico de apertura desde el usuario, en este caso utilizando el teclado del computador.

#### RF-04: Interfaz de Control Externa

Debe recibir comandos de operación (abrir/cerrar) desde un controlador externo mediante un protocolo de comunicación establecido.

#### RF-05: Confirmación de Agarre

Debe incorporar un mecanismo (sensor o límite mecánico) para detectar y confirmar si un objeto ha sido sujetado con éxito.

#### RF-06: Protección por Límites

Debe detener automáticamente el motor al alcanzar los límites físicos de apertura o cierre completo, previniendo daños por bloqueo o sobretorque.

#### RF-07: Regulación de Fuerza

Debe permitir la regulación de la fuerza de agarre aplicada por las pinzas para manipular objetos de diferente fragilidad.

#### RF-08: Tiempo de Respuesta

La secuencia completa de agarre (desde abierto hasta sujetado firme) debe completarse en un tiempo definido.

### ***Requerimientos No Funcionales***

#### **Requerimientos No Funcionales (RNF) — Garra**

##### **RNF-01 — Seguridad física**

Descripción: La garra no debe presentar bordes filosos ni aristas cortantes que puedan lesionar al usuario ni al entorno.

Métrica / Criterio: No hay superficies con radio  $< 0.5$  mm ni aristas afiladas; cantos lijados o redondeados  $\geq 0.5$  mm.

Verificación: Inspección visual y táctil; medición con calibre y comprobación mediante checklist de seguridad.

##### **RNF-02 — Movimiento predecible / sin sorpresas**

Descripción: No debe haber movimientos inesperados (aperturas/cierres involuntarios) bajo condiciones de operación normales.

Métrica / Criterio: Probabilidad de movimiento no solicitado  $< 0.1\%$  durante 10.000 ciclos de prueba; no detectar comandos no intencionados.

Verificación: Prueba de estrés (10.000 ciclos abrir/cerrar con entradas válidas y sin entradas — observar comandos espurios), logs de control y revisión de la lógica de entradas (debounce, filtro).

##### **RNF-03 — Resistencia / vida útil**

Descripción: El sistema debe soportar uso repetido sin fallas.

Métrica / Criterio: Soportar al menos 50.000 operaciones de apertura/cierre sin fallo funcional degradante (lubricación y mantenimiento según especificación).

Verificación: Prueba de durabilidad acelerada en bancada, inspección post-prueba (holguras, fatiga del material, fallos del motor) y registro de fallos.

##### **RNF-04 — Estabilidad del mecanismo**

Descripción: El mecanismo no debe presentar aperturas o cierres involuntarios por vibración o golpes leves.

Métrica / Criterio: Resistencia a vibración: sin activación con vibraciones de hasta 2 g RMS en frecuencias 10–200 Hz durante 60 s; retención de posición con carga prevista  $\pm 10\%$  sin deslizamiento.

Verificación: Ensayo de vibración en bancada; test de retención de posición con carga (peso).

#### RNF-05 — Peso / Ligereza

Descripción: La garra debe ser ligera para no sobrecargar motores.

Métrica / Criterio: Masa total < 150 g (incluye estructura y fijaciones, excluye motor si motor no es parte de la garra).

Verificación: Pesaje en balanza con resolución 0.1 g. Documentar componentes que suman masa.

#### RNF-06 — Materiales y durabilidad

Descripción: Materiales duraderos como PLA, PETG o aluminio.

Métrica / Criterio: Material documentado en especificación; propiedades mínimas: resistencia a tracción y al desgaste compatibles con la carga prevista.

Verificación: Lista de materiales (BOM) y certificaciones/hojas de datos; pruebas de desgaste (abrasión) y ensayo mecánico si es crítico.

#### RNF-07 — Consumo energético bajo

Descripción: El sistema debe consumir un nivel de energía adecuado al microcontrolador/plataforma usada.

Métrica / Criterio: Consumo promedio en operación normal < X mA (definir X según tu plataforma; ejemplo: < 500 mA pico, < 200 mA promedio).

Verificación: Medición con multímetro/registrador de consumo durante uso típico y en máximo esfuerzo; comparar con límites definidos.

#### RNF-08 — Ruido aceptable

Descripción: El ruido de funcionamiento del motor debe ser mínimo para evitar molestias.

Métrica / Criterio: Nivel de presión sonora (SPL) < 60 dB(A) a 1 m durante operación normal (valor orientativo — ajustar según entorno).

Verificación: Medición con sonómetro calibrado o app con referencia; comparar a 1 m de distancia en condiciones de carga.

#### RNF-09 — Latencia / respuesta

Descripción: La garra debe tener respuesta rápida, sin retrasos evidentes para el usuario.



Métrica / Criterio: Latencia entre comando (UI/mando/ps) y comienzo de movimiento < 300 ms; tiempo hasta posición objetivo depende del movimiento (documentar).

Verificación: Medición con cronómetro/registro de eventos: generar comando y medir tiempo hasta que el motor inicia movimiento (sensor o cámara de alta velocidad) — repetir N=30 y reportar media/percentil 95.

#### RNF-10 — Usabilidad de la Interfaz

##### Descripción:

La interfaz gráfica debe ser intuitiva y fácil de utilizar para cualquier tipo de usuario, incluso sin experiencia técnica previa.

##### Métrica / Criterio:

Tiempo promedio para identificar la función principal < 10 segundos para un usuario nuevo; número de errores de navegación < 2 durante una prueba básica de uso.

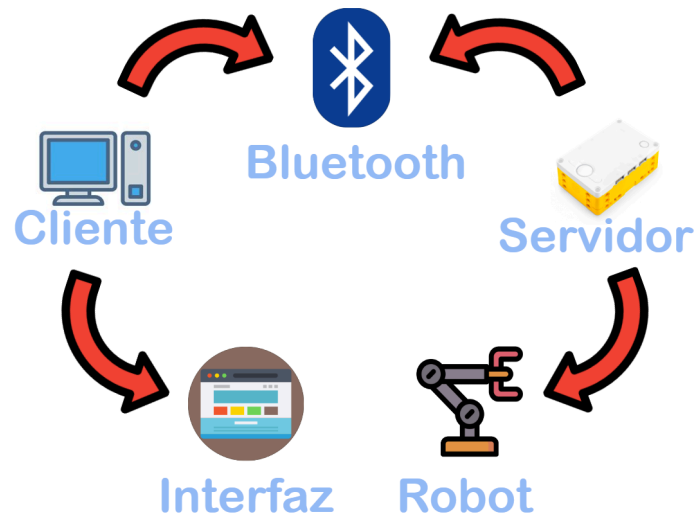
##### Verificación:

Prueba de usabilidad con usuarios sin experiencia; observación directa y registro de tiempos de interacción; checklist de facilidad de navegación.



## 5.2 Arquitectura

La arquitectura tiene como base la de Cliente-Servidor, este se basa entre la interacción entre un cliente(PC) y un servidor(Hub de lego), mediante un ciclo de petición-respuesta.



**Figura 2.** Ciclo de petición-respuesta.

1. Ambos dispositivos deben estar conectados a la misma red(cliente y servidor) para que se pueda realizar la comunicación.
2. Se encarga de la conexión remota entre cliente y robot , este se aloja en el Large Hub del spike army.
3. Por medio de una pc entramos a la interfaz, este se conectará al servidor del robot y se podrá controlarlo.
4. El robot garra(Spike) ejecuta acciones recibidas por el usuario y procede a realizarlas.
5. La interfaz gráfica de "Spike" es donde se encuentran las acciones que puede realizar el robot, está se encuentra programada usando la librería Tkinter.

### 5.3 Interfaz gráfica del sistema

La interfaz gráfica (GUI) se desarrolló en Python con Tkinter para controlar el movimiento del robot LEGO y la garra mediante Bluetooth, además de mostrar el estado de la conexión y un registro de acciones.

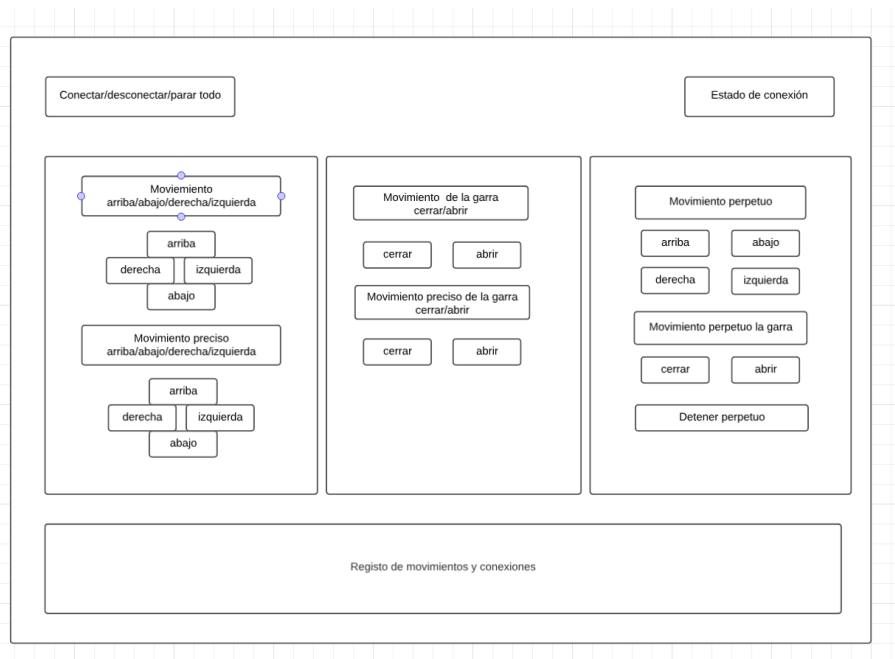


Figura 3. Wireframe de la interfaz gráfica.

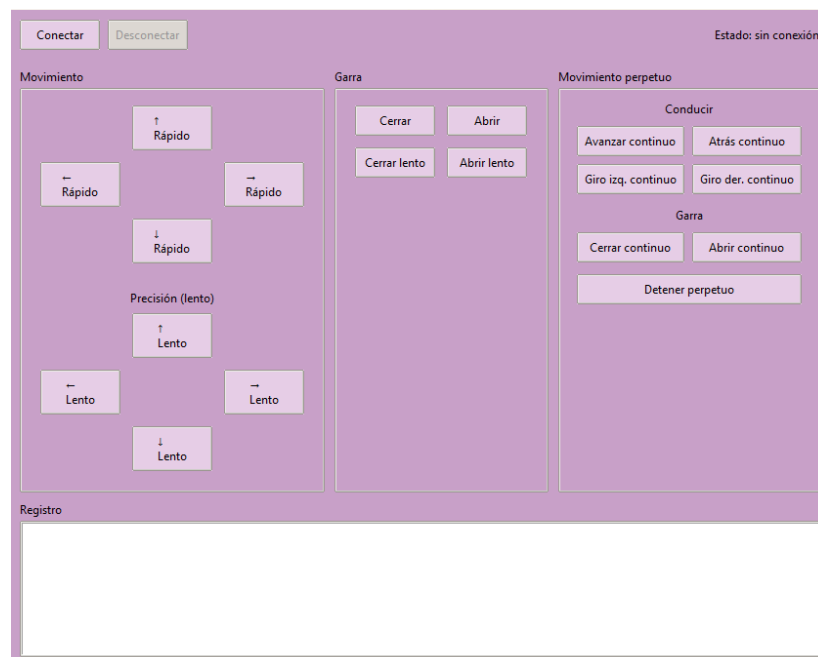


Figura 4. Interfaz gráfica final para el control del robot LEGO.

La **Figura 3** muestra el *wireframe* de baja fidelidad utilizado como boceto inicial, donde se planificó la disposición de los paneles de Movimiento, Garra, Movimiento perpetuo y Registro.

La **Figura 4** muestra la interfaz gráfica final implementada, con el mismo orden de paneles y un esquema de color lila para mejorar la visibilidad de los elementos

A continuación se describe brevemente la función de cada botón y zona de la GUI final.

### Barra superior

- **Conectar:** inicia la búsqueda del hub LEGO y establece la conexión Bluetooth.
- **Desconectar:** corta la conexión con el hub LEGO.
- **Estado:** indica el estado actual del sistema, por ejemplo “sin conexión” o “conectado”.

### Panel *Movimiento*

Permite desplazar el robot en dos modos: rápido y de precisión (lento).

- ↑ **Rápido:** el robot avanza a mayor velocidad.
- ↓ **Rápido:** el robot retrocede a mayor velocidad.
- ← **Rápido:** giro rápido hacia la izquierda.
- → **Rápido:** giro rápido hacia la derecha.
- ↑ **Lento (Precisión):** avance lento para aproximaciones finas.
- ↓ **Lento (Precisión):** retroceso lento.
- ← **Lento (Precisión):** giro lento a la izquierda.
- → **Lento (Precisión):** giro lento a la derecha.

### Panel *Garra*

Controla el motor de la garra del robot.

- **Cerrar:** cierra la garra a velocidad normal.
- **Abrir:** abre la garra a velocidad normal.
- **Cerrar lento:** cierra la garra más lentamente, útil para objetos delicados.
- **Abrir lento:** abre la garra de forma lenta y controlada.



### **Panel *Movimiento perpetuo***

Agrupar los botones para mantener acciones continuas sin tener que mantenerlos presionados.

#### **Conducir:**

- **Avanzar continuo:** el robot avanza de forma continua.
- **Atrás continuo:** el robot retrocede de forma continua.
- **Giro izq. continuo:** giro continuo hacia la izquierda.
- **Giro der. continuo:** giro continuo hacia la derecha.

#### **Garra:**

- **Cerrar continuo:** mantiene la garra cerrando de forma continua.
- **Abrir continuo:** mantiene la garra abriendo de forma continua.
- **Detener perpetuo:** desactiva todos los movimientos continuos, tanto del robot como de la garra.

### **Área *Registro***

- **Registro:** área de texto donde se muestran los movimientos realizados, los cambios de conexión y posibles errores.

Permite al usuario ver qué comandos se han enviado y comprobar el comportamiento del sistema.

## 5.4 Fundamentos de los movimientos

En esta sección se justifica físicamente la acción más importante: sujetar y levantar un objeto.

El robot está construido con LEGO Education SPIKE Prime y realiza este ciclo:

1. Baja la garra (motor grande de elevación).
2. Cierra para agarrar el objeto (motor grande de la garra).
3. Sube el objeto (motor grande de elevación).
4. Gira a derecha/izquierda para posicionarlo (motor pequeño).

**Restricción importante del diseño:** la distancia desde el eje del brazo hasta las pinzas no se puede reducir. Es decir, “r” es fijo porque la garra no cambia su largo: solo baja, agarra, sube y gira.

**Acción específica:** levantar un objeto de 160 g, se analiza el movimiento de elevar un objeto sujetado por la garra, subiéndolo 10 cm en 2 s, con el objeto ubicado a 22 cm del eje del brazo (distancia fija por el diseño).

Datos (en SI):

- Masa:  $m=160\text{ g}=0,160\text{ kg}$
- Gravedad:  $g=9,81\text{ m/s}^2$
- Altura:  $h=10\text{ cm}=0,10\text{ m}$
- Tiempo:  $t=2\text{ s}$
- Distancia al eje:  $r=22\text{ cm}=0,22\text{ m}$  (fijo)

1. **Fuerza principal:** el peso del objeto

El objeto ejerce su peso hacia abajo:  $F=m\cdot g=0,160\cdot 9,81=1,57\text{ N}$

El brazo debe “hacer fuerza” equivalente a 1,57 N para sostener el objeto sin que caiga.

2. **Velocidad y aceleración del levantamiento:**

Velocidad promedio al subir:  $v=h/t=0,10/2=0,05\text{ m/s}$ .

Aceleración aproximada (modelo simple con aceleración constante):

$$a=2h/t^2=2(0,10)/2^2=0,05\text{ m/s}^2$$

Como “a” es muy pequeña comparada con “g”, levantar en 2 s hace el movimiento suave y reduce “tirones”. Eso en la práctica ayuda a que:

- el motor no se esfuerce de golpe,
- el objeto no se sacuda,
- la garra no pierda el agarre.

### 3. Trabajo y potencia del levantamiento

$$W=mgh=0,160*9,81*0,10=0,157 \text{ J}$$

$$P=W/t=0,157/2=0,0785 \text{ W}$$

Esto muestra que el levantamiento no es “caro” en energía. El punto crítico real es el torque por el brazo tipo palanca.

### 4. Torque requerido en el motor que eleva/baja

Como la distancia r es fija, el torque depende directamente de r:  $\tau=F*r$

$$\tau=1,57*0,22=0,345 \text{ N*m}$$

Se agrega un margen del 30% por fricción y pérdidas:

$$\tau_{\text{req}}=1,3*0,345=0,449 \text{ N*m} \approx 0,45 \text{ N*m}$$

Como “r” no se puede reducir, la forma de que el robot funcione bien es controlar cómo se levanta y cómo se gira, para no aumentar el esfuerzo.

### Relación explícita con las decisiones de diseño del robot

Con los resultados anteriores se justifican estas decisiones reales:

#### A. Motor grande para elevar/bajar:

El levantamiento exige torque alto (hasta  $\approx 0,45 \text{ N*m}$ ). Por eso se usa un motor grande en elevación: es la parte que más “fuerza” necesita.

#### B. Motor grande para abrir/cerrar la garra:

Para que el objeto no se caiga durante el levantamiento y el giro, la garra debe apretar lo suficiente para que la fricción lo sostenga:

$$F_{\text{fric}}=\mu N \text{ y debe cumplirse } F_{\text{fric}} \geq mg$$

Cómo  $mg \approx 1,57 \text{ N}$ , se usa un motor grande en el cierre para asegurar un agarre firme y reducir deslizamientos.



C. Motor pequeño para giro izquierda–derecha:

El giro izquierda–derecha es un movimiento de posicionamiento, pero si se gira bruscamente aparecen fuerzas laterales y vibraciones que pueden hacer que el objeto se deslice. Por eso:

el giro se realiza después de levantar,  
se evita girar rápido con el objeto en el aire,  
se prefiere un giro controlado.

D. Regla de control: levantar lento (2 s):

Con  $v=0,05$  m/s y  $a=0,05$  m/s<sup>2</sup>, el levantamiento es suave, disminuye tirones y mejora la estabilidad del agarre.

## Conclusión

A lo largo del desarrollo del proyecto, nuestra experiencia con el robot *LEGO Education SPIKE Prime* ha sido sumamente didáctica. Aunque la etapa de construcción resultó ser la más relajada y sencilla, el verdadero desafío comenzó al momento de programar las acciones de la garra. Desde un inicio, nuestra idea principal era simple pero prometedora: sincronizar un mando de consola que nos permitiera ejecutar las funciones básicas del robot, como abrir y cerrar la garra o desplazarse sobre su propio eje.

Los problemas aparecieron cuando comenzamos a utilizar la plataforma oficial de LEGO Spike, la cual no soportaba funciones externas ni librerías ajenas a su sistema. Esto nos llevó a probar *LEGO Mindstorms*, pero su aplicación había sido descontinuada y no era compatible con nuestro hub. Luego intentamos trabajar con *Pybricks*, donde surgieron dificultades de conexión y líneas de código que no realizaban las acciones esperadas. Después de varios intentos, decidimos avanzar parte del código en *Visual Studio Code*, y fue allí donde logramos un progreso significativo: el programa se sincroniza correctamente con el hub, pudimos controlar la garra mediante el teclado, y se abrió la posibilidad de incorporar un mando más adelante.

Por ahora, no hemos probado aún el control inalámbrico, ya que nuestro enfoque actual se centra en conseguir que la garra funcione correctamente al presionar los botones. Esperamos que, en los próximos días, podamos optimizar el tiempo de respuesta y ajustar los movimientos para que sean más precisos y fluidos.

En general, esta primera fase del proyecto nos ha permitido aprender mucho sobre diseño, ensamblaje y programación. La parte más entretenida fue construir el robot y ver cómo cobraba forma, mientras que la más complicada fue lograr que las librerías funcionaran como queríamos. A pesar de las dificultades, el proyecto avanza conforme a lo planeado, y confiamos en que con tiempo y perseverancia podremos perfeccionar aún más el funcionamiento de nuestra garra y del robot en general.





## Referencias

### 6.1 Fuentes de compra

Canva. (s. f.). *Canva Pro – Herramientas de diseño profesional (suscripción anual)*. Recuperado de <https://n9.cl/s6nf3i>

Microsoft. (s. f.). *Microsoft 365 Personal (suscripción anual)*. Recuperado de <https://n9.cl/5od5j>

Paris Chile. (s. f.). *Mouse vertical ergonómico inalámbrico (negro)*. Recuperado de <https://n9.cl/3ylfte>

Paris Chile. (s. f.). *Notebook Gaming Victus 16-r1015la, Intel Core i7, 16 GB RAM, 1 TB SSD, RTX 4060, 16.1" FHD 144 Hz, Windows 11 Home*. Recuperado de <https://n9.cl/l3vl7>

Ubuy Chile. (s. f.). *LEGO Education SPIKE Prime Set*. Recuperado de <https://n9.cl/tac8z>

### 6.2 Fuentes de información

LEGO. (s. f.). *Acerca de LEGO Mindstorms*. Recuperado de <https://n9.cl/udokh>

YouTube. (2022, marzo 15). *LEGO Mindstorms – Introducción y funciones básicas* [Video]. YouTube. Recuperado de <https://n9.cl/w42fs>

YouTube. (2021, noviembre 10). *Programación del robot LEGO Mindstorms paso a paso* [Video]. YouTube. Recuperado de <https://n9.cl/x6ptp>

Indeed. (s. f.). *Salarios: búsqueda por cargo y ubicación*. Recuperado el 28 de noviembre de 2025, de <https://cl.indeed.com/career/salaries?from=gnav-homepage>

Visure Solutions. *Requerimientos Funcionales*  
<https://visuresolutions.com/es/gu%C3%ADa-de-limosna/requerimientos-funcionales>

Visure Solutions. *Requerimientos No Funcionales*  
<https://visuresolutions.com/es/gu%C3%ADa-de-limosna/requerimientos-no-funcionales>



Gannouji, R. (s. f.). *Física 2 (mecánica clásica)*. Instituto de Física, Pontificia Universidad Católica de Valparaíso. Recuperado de <https://surl.li/bafofc>

Lucid Software. (s. f.). *Lucidchart* [Herramienta de diagramación en línea]. Recuperado de <https://surl.lu/sifqjz>

# REQUISITOS

Secciones nuevas

Análisis - Diseño

- Requerimientos
  - Funcionales no funcionales
- Diseño inicial Gui

Implementación

- Código: (Arquitectura Cliente-Servidor)
  - Cliente
  - Servidor
- Interfaz (GUI)

Iniciar el manual de usuario

Debe mostrar un caso en particular y se debe explicar su uso en base a este.w

Crear un ppt

el primer informe final se entrega el 12 de diciembre

El último informe debe contener un manual de usuario, daniel se encarga

Requerimientos funcionales y no funcional, (cuales son las funcionalidades que debe tener)

En la parte de la interfaz gráfica

debemos poner como creamos la interfaz gráfica y luego poner la interfaz final  
análisis inicial de la interfaz gráfica (como será la interfaz)

segundo informe

Capacidad física del robot.

## Pestaña 3

### 5.3 Interfaz gráfica del sistema

La interfaz gráfica (GUI) se desarrolló en Python con Tkinter para controlar el movimiento del robot LEGO y la garra mediante Bluetooth, además de mostrar el estado de la conexión y un registro de acciones.

La **Figura 1** muestra el *wireframe* de baja fidelidad utilizado como boceto inicial, donde se planificó la disposición de los paneles de Movimiento, Garra, Movimiento perpetuo y Registro.

La **Figura 2** muestra la interfaz gráfica final implementada, con el mismo orden de paneles y un esquema de color lila para mejorar la visibilidad de los elementos.

**Figura 1. Wireframe de la interfaz gráfica.**

**Figura 2. Interfaz gráfica final para el control del robot LEGO.**

A continuación se describe brevemente la función de cada botón y zona de la GUI final.

#### Barra superior

- **Conectar:** inicia la búsqueda del hub LEGO y establece la conexión Bluetooth.
- **Desconectar:** corta la conexión con el hub LEGO.
- **Estado:** ... (etiqueta de texto): indica el estado actual del sistema, por ejemplo “sin conexión” o “conectado”.

#### Panel *Movimiento*

Permite desplazar el robot en dos modos: rápido y de precisión (lento).

- ↑ **Rápido:** el robot avanza a mayor velocidad.
- ↓ **Rápido:** el robot retrocede a mayor velocidad.
- ← **Rápido:** giro rápido hacia la izquierda.
- → **Rápido:** giro rápido hacia la derecha.
- ↑ **Lento (Precisión):** avance lento para aproximaciones finas.
- ↓ **Lento (Precisión):** retroceso lento.
- ← **Lento (Precisión):** giro lento a la izquierda.
- → **Lento (Precisión):** giro lento a la derecha.

### Panel *Garra*

Controla el motor de la garra del robot.

- **Cerrar:** cierra la garra a velocidad normal.
- **Abrir:** abre la garra a velocidad normal.
- **Cerrar lento:** cierra la garra más lentamente, útil para objetos delicados.
- **Abrir lento:** abre la garra de forma lenta y controlada.

### Panel *Movimiento perpetuo*

Agrupar los botones para mantener acciones continuas sin tener que mantenerlos presionados.

#### **Conducir:**

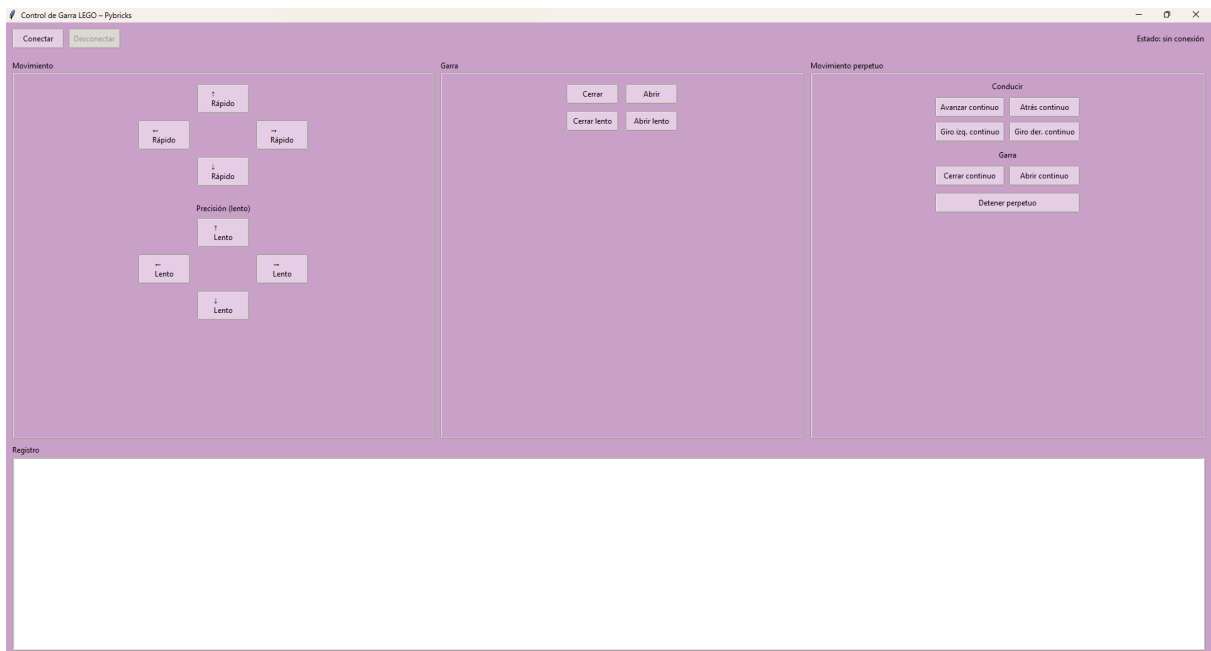
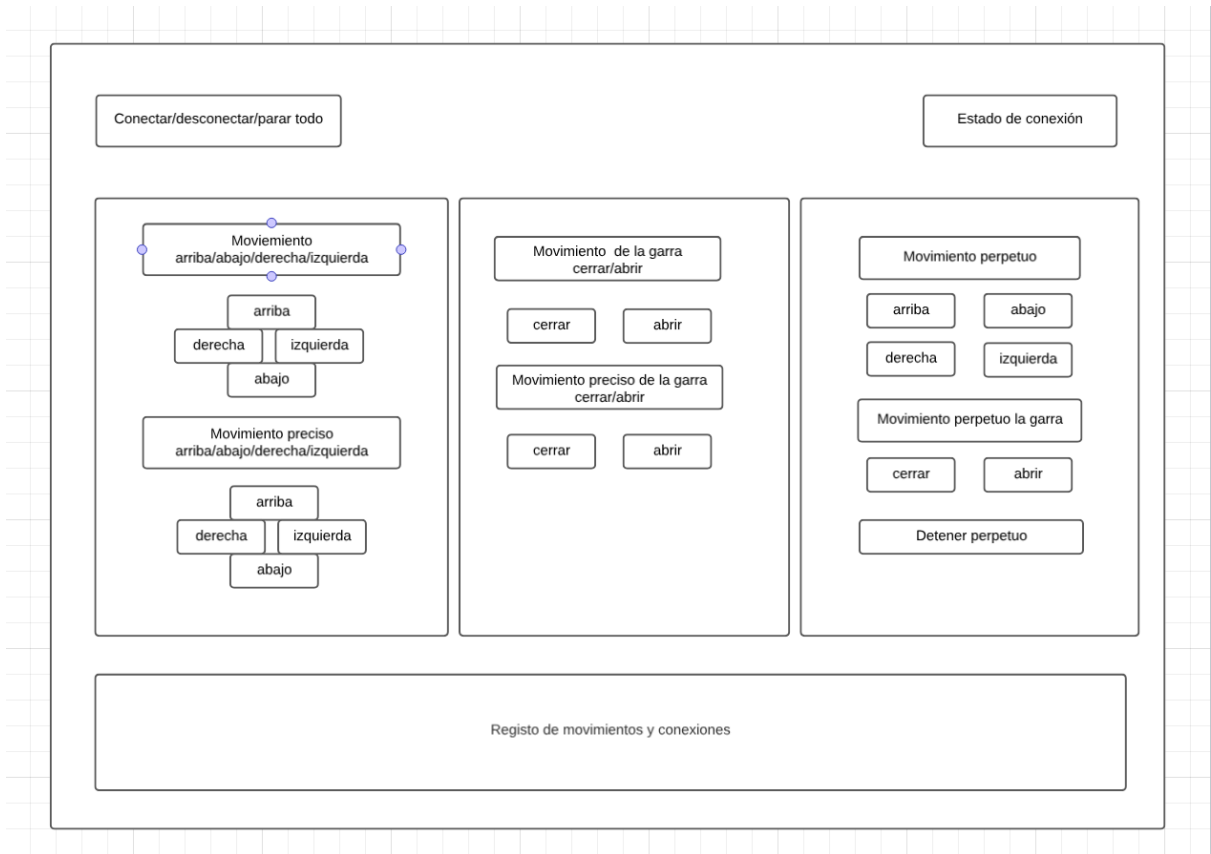
- **Avanzar continuo:** el robot avanza de forma continua.
- **Atrás continuo:** el robot retrocede de forma continua.
- **Giro izq. continuo:** giro continuo hacia la izquierda.
- **Giro der. continuo:** giro continuo hacia la derecha.

#### **Garra:**

- **Cerrar continuo:** mantiene la garra cerrando de forma continua.
- **Abrir continuo:** mantiene la garra abriendo de forma continua.
- **Detener perpetuo:** desactiva todos los movimientos continuos, tanto del robot como de la garra.

### *Área Registro*

- **Registro:** área de texto donde se muestran los movimientos realizados, los cambios de conexión y posibles errores.  
Permite al usuario ver qué comandos se han enviado y comprobar el comportamiento del sistema.





Conectar

Desconectar

Estado: sin conexión

Movimiento

↑  
Rápido

←  
Rápido

→  
Rápido

↓  
Rápido

Precisión (lento)

↑  
Lento

←  
Lento

→  
Lento

↓  
Lento

Garra

Cerrar

Abrir

Cerrar lento

Abrir lento

Movimiento perpetuo

Conducir

Avanzar continuo

Atrás continuo

Giro izq. continuo

Giro der. continuo

Garra

Cerrar continuo

Abrir continuo

Detener perpetuo

Registro

## 5.4 Fundamentos de los movimientos

En esta sección se justifica físicamente la acción más importante: sujetar y levantar un objeto.

El robot está construido con LEGO Education SPIKE Prime (45678) y realiza este ciclo:

1. Baja la garra (motor grande de elevación).
2. Cierra para agarrar el objeto (motor grande de la garra).
3. Sube el objeto (motor grande de elevación).
4. Gira a derecha/izquierda para posicionarlo (motor pequeño).

Restricción importante del diseño: la distancia desde el eje del brazo hasta las pinzas no se puede reducir. Es decir,  $r$  es fijo porque la garra no cambia su largo: solo baja, agarra, sube y gira.

Acción específica: levantar un objeto de 160 g

Se analiza el movimiento de elevar un objeto sujetado por la garra, subiéndolo 10 cm en 2 s, con el objeto ubicado a 22 cm del eje del brazo (distancia fija por el diseño).

Datos (en SI)

- Masa:  $m=160\text{ g}=0,160\text{ kg}$
- Gravedad:  $g=9,81\text{ m/s}^2$
- Altura:  $h=10\text{ cm}=0,10\text{ m}$
- Tiempo:  $t=2\text{ s}$
- Distancia al eje:  $r=22\text{ cm}=0,22\text{ m}$  (fijo)

1) Fuerza principal: el peso del objeto

El objeto ejerce su peso hacia abajo:  $F=m \cdot g=0,160 \cdot 9,81=1,57\text{ N}$

Significado: el brazo debe “hacer fuerza” equivalente a 1,57 N para sostener el objeto sin que caiga.

2) Velocidad y aceleración del levantamiento (por qué levantar lento ayuda)

Velocidad promedio al subir:  $v=h/t=0,10/2=0,05\text{ m/s}$

Aceleración aproximada (modelo simple con aceleración constante):

$$a=2h/t^2=2(0,10)/2^2=0,05\text{ m/s}^2$$

Como  $a$  es muy pequeña comparada con  $g$ , levantar en 2 s hace el movimiento suave y reduce “tirones”. Eso en la práctica ayuda a que:

- el motor no se esfuerce de golpe,
- el objeto no se sacuda,
- la garra no pierda el agarre.

### 3) Trabajo y potencia del levantamiento

Trabajo mínimo para subir el objeto:

$$W=mgh=0,160 \cdot 9,81 \cdot 0,10=0,157 \text{ J}$$

Potencia promedio en 2 s:

$$P=W/t=0,157/2=0,0785 \text{ W/}$$

Esto muestra que el levantamiento no es “caro” en energía. El punto crítico real es el torque por el brazo tipo palanca.

### 4) Torque requerido en el motor que eleva/baja (cálculo clave)

Como la distancia  $r$  es fija, el torque depende directamente de  $r$ :  $\tau=F \cdot r$

$$\text{Con nuestros datos: } \tau=1,57 \cdot 0,22=0,345 \text{ N} \cdot \text{m}$$

Se agrega un margen del 30% por fricción y pérdidas:  $\tau_{\text{req}}=1,3 \cdot 0,345=0,449 \text{ N} \cdot \text{m} \approx 0,45 \text{ N} \cdot \text{m}$

Interpretación clara: como  $r$  no se puede reducir, la forma de que el robot funcione bien es controlar cómo se levanta y cómo se gira, para no aumentar el esfuerzo.

Relación explícita con las decisiones de diseño del robot

Con los resultados anteriores se justifican estas decisiones reales:

#### 1. Motor grande para elevar/bajar:

El levantamiento exige torque alto (hasta  $\approx 0,45 \text{ N} \cdot \text{m}$ ). Por eso se usa un motor grande en elevación: es la parte que más “fuerza” necesita.

#### 2. Motor grande para abrir/cerrar la garra:

Para que el objeto no se caiga durante el levantamiento y el giro, la garra debe apretar lo suficiente para que la fricción lo sostenga:

$$F_{\text{fric}}=\mu N \text{ y debe cumplirse } F_{\text{fric}} \geq mg$$

Como  $mg \approx 1,57 \text{ N}$ , se usa un motor grande en el cierre para asegurar un agarre firme y reducir deslizamientos.

3. Motor pequeño para giro izquierda–derecha:

El giro izquierda–derecha es un movimiento de posicionamiento, pero si se gira bruscamente aparecen fuerzas laterales y vibraciones que pueden hacer que el objeto se deslice. Por eso:

el giro se realiza después de levantar,

se evita girar rápido con el objeto en el aire,

se prefiere un giro controlado.

4. Regla de control: levantar lento (2 s):

Con  $v=0,05 \text{ m/s}$  y  $a=0,05 \text{ m/s}^2$ , el levantamiento es suave, disminuye tirones y mejora la estabilidad del agarre.

#### 5.4 Fundamentos de los movimientos

En esta sección se justifica físicamente la acción más importante: sujetar y levantar un objeto.

El robot está construido con LEGO Education SPIKE Prime (45678) y realiza este ciclo:

Baja la garra (motor grande de elevación).

Cierra para agarrar el objeto (motor grande de la garra).

Sube el objeto (motor grande de elevación).

Gira a derecha/izquierda para posicionarlo (motor pequeño).

Restricción importante del diseño: la distancia desde el eje del brazo hasta las pinzas no se puede reducir. Es decir,  $r$  es fijo porque la garra no cambia su largo: solo baja, agarra, sube y gira.

Acción específica: levantar un objeto de 160 g

Se analiza el movimiento de elevar un objeto sujetado por la garra, subiéndolo 10 cm en 2 s, con el objeto ubicado a 22 cm del eje del brazo (distancia fija por el diseño).

Datos (en SI)

Masa:  $m=160\text{ g}=0,160\text{ kg}$

Gravedad:  $g=9,81\text{ m/s}^2$

Altura:  $h=10\text{ cm}=0,10\text{ m}$

Tiempo:  $t=2\text{ s}$

Distancia al eje:  $r=22\text{ cm}=0,22\text{ m}$  (fijo)

1) Fuerza principal: el peso del objeto

El objeto ejerce su peso hacia abajo:  $F=m \cdot g=0,160 \cdot 9,81=1,57\text{ N}$

Significado: el brazo debe “hacer fuerza” equivalente a 1,57 N para sostener el objeto sin que caiga.

2) Velocidad y aceleración del levantamiento (por qué levantar lento ayuda)

Velocidad promedio al subir:  $v=h/t=0,10/2=0,05\text{ m/s}$

Aceleración aproximada (modelo simple con aceleración constante):

$a=2h/t^2=2(0,10)/2^2=0,05\text{ m/s}^2$

Como  $a$  es muy pequeña comparada con  $g$ , levantar en 2 s hace el movimiento suave y reduce “tirones”. Eso en la práctica ayuda a que:

el motor no se esfuerce de golpe,

el objeto no se sacuda,

la garra no pierda el agarre.

3) Trabajo y potencia del levantamiento

Trabajo mínimo para subir el objeto:

$W=mgh=0,160 \cdot 9,81 \cdot 0,10=0,157\text{ J}$

Potencia promedio en 2 s:

$P=W/t=0,157/2=0,0785\text{ W}$

Esto muestra que el levantamiento no es “caro” en energía. El punto crítico real es el torque por el brazo tipo palanca.

4) Torque requerido en el motor que eleva/baja (cálculo clave)

Como la distancia  $r$  es fija, el torque depende directamente de  $r$ :  $\tau=F \cdot r$

Con nuestros datos:  $\tau=1,57 \cdot 0,22=0,345\text{ N} \cdot \text{m}$

Se agrega un margen del 30% por fricción y pérdidas:

$\tau_{\text{req}}=1,3 \cdot 0,345=0,449\text{ N} \cdot \text{m} \approx 0,45\text{ N} \cdot \text{m}$

Interpretación clara: como  $r$  no se puede reducir, la forma de que el robot funcione bien es controlar cómo se levanta y cómo se gira, para no aumentar el esfuerzo.

Relación explícita con las decisiones de diseño del robot

Con los resultados anteriores se justifican estas decisiones reales:

Motor grande para elevar/bajar:

El levantamiento exige torque alto (hasta  $\approx 0,45 \text{ N} \cdot \text{m}$ ). Por eso se usa un motor grande en elevación: es la parte que más “fuerza” necesita.

Motor grande para abrir/cerrar la garra:

Para que el objeto no se caiga durante el levantamiento y el giro, la garra debe apretar lo suficiente para que la fricción lo sostenga:

$F_{\text{fric}} = \mu N$  y debe cumplirse  $F_{\text{fric}} \geq mg$

Como  $mg \approx 1,57 \text{ N}$ , se usa un motor grande en el cierre para asegurar un agarre firme y reducir deslizamientos.

Motor pequeño para giro izquierda–derecha:

El giro izquierda–derecha es un movimiento de posicionamiento, pero si se gira bruscamente aparecen fuerzas laterales y vibraciones que pueden hacer que el objeto se deslice. Por eso:

el giro se realiza después de levantar,

se evita girar rápido con el objeto en el aire,

se prefiere un giro controlado.

Regla de control: levantar lento (2 s):

Con  $v = 0,05 \text{ m/s}$  y  $a = 0,05 \text{ m/s}^2$ , el levantamiento es suave, disminuye tirones y mejora la estabilidad del agarre.

## Pestaña 4

Al momento de crear la interfaz gráfica lo que hicimos primero crear un wireframe o prototipo de baja fidelidad, en el cual definimos la arquitectura de la información antes de programar, primeramente determinamos que era crítico separar visualmente 3 zonas:

1. La gestión de conexión que se ubica en la parte superior
2. La manipulación y movimiento de la garra en la parte central
3. y el recuadro final que se encargara de almacenar los registros de movimientos y conexiones del programa.

A la derecha se puede apreciar la implementación final desarrollada en python utilizando la librería tkinter, esta elección nos permitió tener el control total sobre los eventos del teclado y la comunicación bluetooth mediante la librería pybricksdev

Ahora me gustaría señalar la dualidad en el control de movimiento, que diseñamos pensando en un escenario de operación real:

1. **Modo Rápido (Teclas WASD):** Esto envía potencias altas a los motores de tracción y su objetivo es reducir los tiempos muertos al desplazarse desde el punto de inicio hasta la zona de trabajo.
2. **Modo de Precisión (Teclas IJKL):** El cual es fundamental. Al estar cerca del objetivo, reducimos drásticamente la potencia. Esto permite movimientos milimétricos para alinear la garra sin golpear el objeto, esto mismo es algo crítico cuando se manipulan materiales peligrosos o frágiles.

Además, implementamos funciones de '**Movimiento Perpetuo**', que permiten al robot realizar tareas de largas distancias sin que el operador tenga que presionar repetidamente una tecla, reduciendo la fatiga de este mismo.

## Diapositiva 11: Fundamentos de los Movimientos

**Presentador:** "Ahora, ¿cómo garantizamos que el diseño mecánico sea viable? lo que hicimos fue realizar un análisis físico detallado para validar la selección de los motores.

El desafío principal de nuestro diseño es que **el brazo de la garra tiene un radio fijo de 22 centímetros**. A diferencia de un brazo humano que se encoge, nuestra garra es una palanca rígida. Esto obliga al motor a trabajar siempre en la condición de mayor esfuerzo.

Ahora analicemos un caso de levantar una carga de **160 gramos**

**Fuerza:** Primero, el peso del objeto genera una fuerza hacia abajo de 1.57 N que se calcula multiplicando la masa del objeto por la gravedad.

1. **Torque (La clave del diseño):** Al aplicar la fórmula de Torque  $F \cdot r$ , obtenemos un torque base de 0.345 N\*m. Sin embargo, en ingeniería no trabajamos al límite. Agregamos un **margen de seguridad del 30%** por fricción y pérdidas mecánicas, resultando en un **Torque Requerido de 0.45 N\*m**. Este valor justificó tres decisiones críticas de diseño:



- **Decisión 1 (Motores):** Usamos los **Motores Grandes** del LEGO para la elevación y el cierre de la garra. Un motor mediano operaría demasiado cerca de su límite de bloqueo, arriesgando un sobrecalentamiento. Además, el motor de la garra debe ejercer una fuerza de fricción mayor al peso del objeto  $F_{\text{fric}} > mg$  para que no se resbale.
- **Decisión 2 (Tiempo de Elevación):** Definimos por software que el levantamiento dure **2 segundos**. Esto resulta en una potencia promedio muy baja y una aceleración casi nula. Al levantar lento, evitamos la inercia que podría multiplicar el peso aparente del objeto y romper el agarre.
- **Decisión 3 (Giro):** Usamos un motor pequeño para el giro lateral, pero solo se activa una vez que la carga está estabilizada, minimizando las fuerzas laterales."
- Se usa un motor pequeño para el giro, realizándose suavemente para evitar fuerzas laterales.
- 

### Preparación para Preguntas (Slide 11)

- **P: ¿Qué pasa si intentan levantar algo más pesado de 160g?**

*R:* El margen de seguridad del 30% nos da holgura. Sin embargo, si el peso excede la capacidad de torque del motor (aprox  $> 0.6 \text{ N} \cdot \text{m}$ ), el motor entraría en "stall" (bloqueo) y el sistema de protección térmica del Hub lo apagaría para evitar daños.

- **P: ¿Por qué no diseñaron una garra que se pudiera encoger (retraer)?**
  - *R:* Eso requeriría articulaciones adicionales (como un codo) y más motores, lo que aumentaría el peso total del brazo. Un brazo más pesado requeriría aún más torque para levantarse a sí mismo. El diseño de brazo rígido es más eficiente para la potencia limitada del Hub Spike Prime.
- **P: ¿A qué se refiere con que el radio es fijo?**
  - *R:* A que la distancia desde el eje de rotación (el motor) hasta el punto donde se aplica la fuerza (la garra) es constante ( $22 \text{ cm}$ ). No podemos reducir esa distancia para disminuir el efecto palanca.

### Preparación para Preguntas (Slide 10)

- **P: ¿Por qué usaron Tkinter y no una web app o la app de LEGO?**
  - *R:* La app de LEGO es cerrada y limita la integración de lógica compleja. Tkinter en Python nos permite procesar datos en el PC, usar librerías avanzadas de comunicación y tener una latencia mínima al procesar los eventos del teclado directamente en el sistema operativo.
- **P: ¿Cómo se comunica la interfaz con el robot?**
  - *R:* Usamos una arquitectura Cliente-Servidor. La interfaz (Cliente) envía comandos serializados por Bluetooth. El Hub del robot (Servidor) corre un script en MicroPython que escucha estos comandos y acciona los motores.