UNIVERSIDAD DE TARAPACÁ

FACULTAD DE INGENIERÍA

DEPARTAMENTO DE INGENIERÍA CIVIL EN COMPUTACIÓN E INFORMÁTICA

Ingenierí@ Computación e Informática

Plan de proyecto: Fase 2

“Claw-tylda”

Alumnos: Juan-Daniel Castillo
Javier Echeverría
Alexander Pinto
José Terrazas

Profesor: Baris Klobertanz

12 – 2025

Tabla 1: Historial de cambios

Fecha	Versión	Descripción	Autor(es)
26/09/2025	1.0	Formulación del Proyecto	Juan-Daniel Castillo Javier Echeverria Alexander Pinto José Terrazas
02/10/25	1.1	Elaboración Panorama General y Especificación del Problema	Juan-Daniel Castillo
03/10/25	1.2	Se añadieron Objetivos, Restricciones y se definió la Organización del Personal	Javier Echeverria Alexander Pinto
09/10/25	1.3	Definición de Actividades y actualización Carta Gantt	Juan-Daniel Castillo
10/10/25	1.4	Finalización del encabezado 2	Juan-Daniel Castillo
12/10/25	1.5	Elaboración completa del encabezado 4 y finalización de Entregables	Alexander Pinto Juan-Daniel Castillo
15/10/25	1.6	Elaboración Gestión de Riesgos	José Terrazas
17/10/25	1.7	Elaboración de conclusión y revisión del documento	Juan-Daniel Castillo Alexander Pinto
28/11/25	1.8	Revisión del informe.	Todo el grupo
18/12/25	2.0	Informe fase 2	Todo el grupo
14/12/25	2.1	Fase 2 completa	Todo el grupo

Tabla de Contenidos

Índice de Tablas

Índice de figuras

1. Panorama General

En la actualidad la industria minera es una de las principales fuentes económicas de Chile. Según datos del Banco Central de Chile (2024), la minería representó un 11.7% del PIB nacional, por lo que resulta vital para el país mantenerla competitiva a nivel global, para esto es necesario cumplir con los estándares contemporáneos de productividad, costos y seguridad.

El medio para afrontar este desafío es transicionar a la Minería 4.0, un modelo que se caracteriza por la digitalización y automatización de procesos.

1.1. Especificación del Problema

Durante este proyecto, estudiantes de Ingeniería Civil en Computación e Informática, experimentaran una aproximación al escenario real planteado en el panorama general.

Específicamente se implementará una garra robótica capaz de cargar el material fragmentado hacia los vehículos de transporte y controlada a través de una interfaz gráfica, con el fin de mejorar la seguridad y optimizar la eficiencia del proceso.

Para la construcción del prototipo se utilizará el kit Lego Spike Prime, este ofrece las herramientas necesarias para el ensamblaje, ya que cuenta con piezas para construir la estructura de la garra y motores que se encargan de generar los movimientos de apertura, cierre, elevación y desplazamiento lateral. Además cuenta con una API que permite programar los componentes del prototipo para controlar su funcionamiento para conectarlo a una interfaz gráfica.

1.2. Objetivos

1.2.1. Objetivo General

Implementar un prototipo de garra robótica con el kit Lego Education: Spike Prime, capaz de cargar el material minero hasta el vehículo transportador. Se debe controlar con una interfaz gráfica y contar con capacidad para realizar apertura, cierre, ajuste de altura y desplazamientos laterales.

1.2.2. Objetivos Específicos

- Investigar al menos tres diseños para el robot en las primeras dos semanas de proyecto y compararlos en términos de resistencia estructural, movilidad y tamaño.

- Investigar y seleccionar al menos una biblioteca que sea compatible con el kit Lego Spike Prime, para codificar el movimiento de los motores del robot en las primeras dos semanas.

- Seleccionar y ensamblar un diseño funcional durante el primer mes, debe contar con una estructura que le permita abrir y cerrar la garra, elevarse y desplazarse horizontalmente.

- Codificar los movimientos de apertura y cierre de garra, elevación y desplazamiento lateral de la estructura, hasta la mitad del segundo mes del proyecto.

- Realizar pruebas de compatibilidad de la estructura del robot con la codificación de los movimientos, se deben poder realizar los movimientos principales de la garra. Este objetivo se desarrollará las dos semanas siguientes después de tener listo un prototipo funcional.

- Definir y ensamblar el diseño final para el robot, ajustando los problemas encontrados en las pruebas de compatibilidad, asegurándose de que funcione correctamente, durante el segundo mes del proyecto.

- Realizar ajustes en la codificación de los movimientos del robot, después de definir el diseño final, debe ser capaz de realizar la apertura y cierre de la garra, movimiento de la base,brazo y codo sin romperse, este objetivo debe estar completo al menos 2 semanas antes de la fecha final del proyecto.

- Investigar y seleccionar bibliotecas para codificar la interfaz de usuario de control del robot, durante el primer mes y medio. Este objetivo se debe iniciar después de contar con un prototipo funcional tanto de la estructura como de los movimientos.

- Desarrollar una interfaz gráfica que permita a un usuario controlar los movimientos del brazo,codo,garra y base del robot, se debe tener un prototipo funcional al menos 3 semanas antes de fecha final del proyecto.

1.3. Restricciones

Tabla 2: Restricciones del proyecto

Restricción	Detalle
Limite de tiempo	3 meses.
Material	kit Lego Spike Prime.
Personal	5 personas.
Software	Se debe programar con librerías compatibles con el kit de Lego.
Ubicación	El kit de lego solo se puede usar en la universidad.
Registro	El registro del proyecto se debe realizar en Redmine.

1.4. Entregables

1. **Informe inicial:** Documento que proyecta la fase de planificación como el alcance y los objetivos que se tiene como expectativa, los roles asignados y los primeros registros de desarrollo que se tuvieron como propuesta inicial.

2. **Informe final:** Documento que mostrará los resultados y describirá los objetivos completados, servirá para analizar el desempeño y los riesgos que se enfrentaron y así mismo cómo lo afrontaron.

3. **Presentación oral:** Presentación integral del proyecto, basándose en el informe final del proyecto propiamente tal, junto a puntos claves determinados en su desarrollo. Se concluirá con una demostración del robot, demostrando así todas sus capacidades y un rendimiento satisfactorio.

4. **Código base del cliente y servidor:** Archivos del código utilizado en el proyecto.

5. **Página en redmine del proyecto**: Página donde se registra y visualiza la organización de actividades del proyecto mediante la Carta Gantt.

6. **Bitácora**: Informe semanal que describe el avance del proyecto y los desafíos encontrados, asimismo la distribución de las tareas asignadas y desafíos futuros.

2. Organización del Personal

La organización dentro de un grupo es fundamental para llevar a cabo un trabajo de manera efectiva. Es importante distribuir las tareas de forma adecuada, de modo que cada integrante contribuya al cumplimiento de los objetivos. Los roles fueron elegidos en una reunión, centrándose principalmente en que cada integrante esté en un rol en el que tenga habilidades y se sienta cómodo realizando.

2.1. Descripción de los Roles

- **Jefe de Proyecto:** Es el encargado de la representación del equipo de trabajo, y tiene la responsabilidad de planificar, organizar y supervisar todas aquellas etapas en las que fue desarrollado el proyecto. Asimismo, es quien coordina las tareas que tratará cada integrante y vela por el cumplimiento de los objetivos y los plazos previamente establecidos.

- **Ensamblador:** Es quien se encarga del montaje y armado de las piezas del robot, asegurando la eficiencia de cada componente. Por otro lado, contribuye con el programador para verificar el cumplimiento de las funcionalidades y el rendimiento del sistema.

- **Programador:** Tiene la responsabilidad del desarrollo de codificación y programación del robot, velando por su óptimo rendimiento. Coopera con el ensamblador para todos aquellos ajustes y optimizaciones del robot.

- **Documentador:** Tiene a su cargo el registro de avances del proyecto, por ejemplo: elaborar las bitácoras semanales y realizar los informes finales.

2.2. Personal que cumplirá los Roles

Tabla 3: Organización del personal

Rol	Responsable
Jefe de proyecto	Alexander Pinto
Programador	Javier Echeverria
Ensamblador	José Terrazas
Documentador	Juan-Daniel Castillo

2.3. Mecanismos de comunicación

Los principales medios de comunicación que utilizaremos son los siguientes:

- **WhatsApp:** Mensajería y coordinación rápida entre los integrantes del grupo
- **Discord:** Para realizar reuniones personales y grupales, aprovechando sus canales de voz y texto para una mejor organización y comunicación en tiempo real.
- **Reuniones presenciales:** Útiles para una comunicación más directa que aprovecharemos para tomar decisiones, registrar avances y ajustar el progreso del proyecto.

3. Planificación del proyecto

La planificación del proyecto es una parte fundamental para asegurar la correcta ejecución y cumplimiento de los objetivos. En esta sección se presenta como se planificó el proyecto por medio de objetivos específicos y los integrantes que son responsables de este, junto con que se busca realizar por medio de este mismo, además, se mostrará el cumplimiento de las actividades propuestas por medio de la carta Gantt.

3.1. Actividades

Tabla 4: Planificación de actividades

Nombre	Descripción	Responsables	Producto
Experimentación con el kit de LEGO	Se realiza la inducción al kit de LEGO	Todo el grupo.	Familiarización con la construcción y codificación del kit.
Identificación del problema	Análisis del panorama general.	Todo el grupo.	Comprender qué problema debe resolver el proyecto.
Objetivos del proyecto	Se definen objetivos generales y específicos.	Juan-Daniel Castillo.	Claridad sobre qué tareas realizar.
Investigación de modelos	Se buscan modelos útiles para	José Terrazas. Alexander Pinto.	Obtener referencias que sirvan de base para el desarrollo del prototipo.
Construcción del prototipo	Se construye la base y brazo del prototipo	José Terrazas.	Ensamblar las piezas del kit y asegurar la estabilidad del modelo y diseño del prototipo de manera eficiente.
Pruebas con el código	Se experimenta con el uso de motores	Javier Echeverria. Juan-Daniel Castillo	Probar y optimizar el código para lograr un funcionamiento correcto del prototipo.

Tabla 3: Planificación de actividades (Continuación)

Nombre	Descripción	Responsables	Producto
Construcción del prototipo	Se construye la garra	Alexander Pinto.	Establecer la estructura base sobre la cual se desarrollarán las siguientes etapas del prototipo .
Ensamblado del prototipo	Se unen las partes construidas para tener el prototipo funcional	José Terrazas.	Asegurar que todas las piezas estén correctamente conectadas y que el sistema funcione de manera estable.
Codificación de movimientos	Se codifican los movimientos del brazo y garra	Javier Echeverria.	Desarrollar la lógica programando las acciones planificadas.
Pruebas iniciales	Pruebas para corregir y ajustar el funcionamiento del robot	Todo el grupo.	Detectar posibles fallas y realizar ajustes en la programación o en el prototipo.
Bitácoras semanales	Registro de avance,problemas,solucionar y tareas a realizar.	Juan-Daniel Castillo	Tener un seguimiento claro del progreso del proyecto.

3.2. Carta Gantt

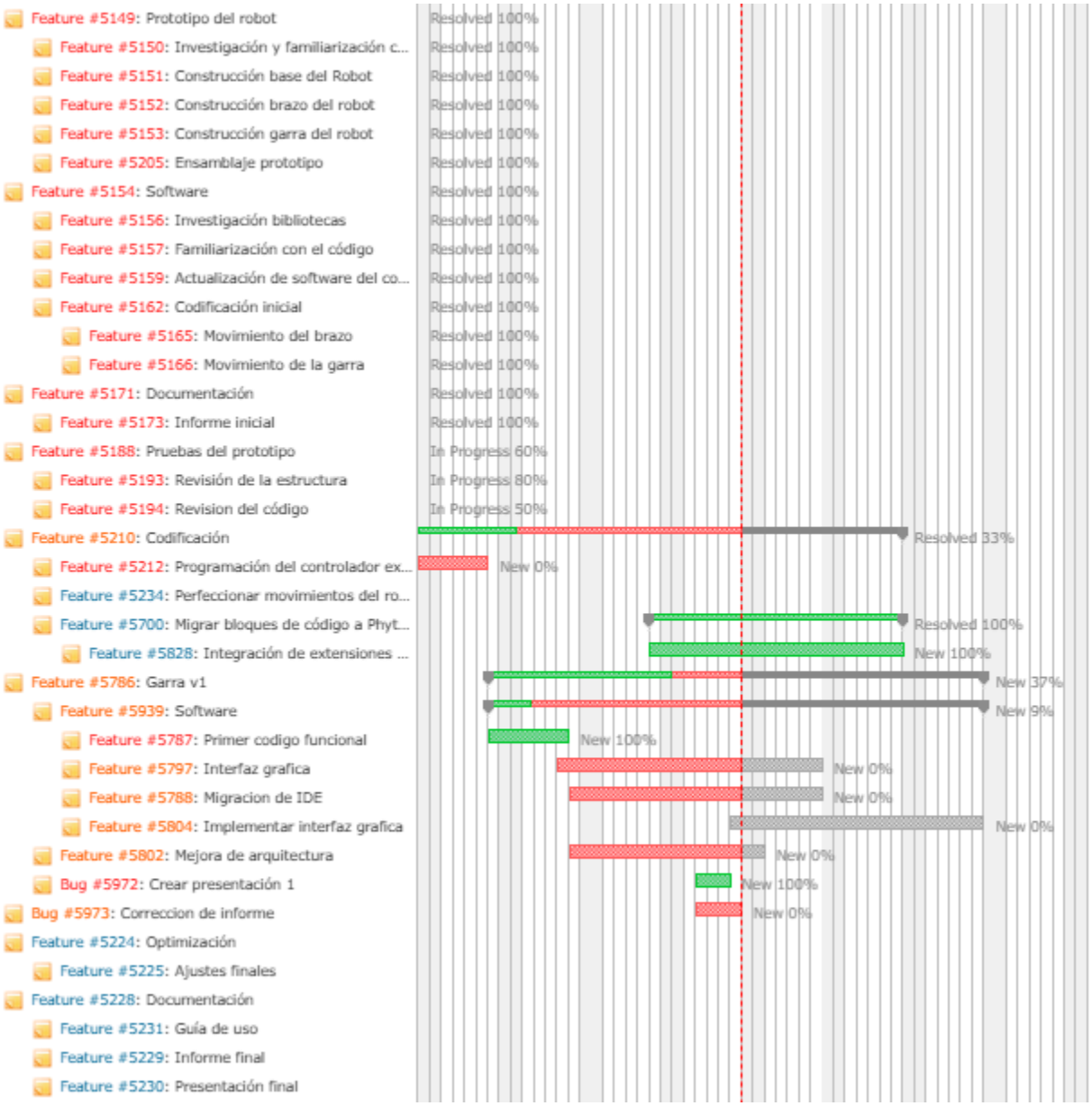

Figura 1: Carta Gantt

3.3. Gestión de Riesgos

A continuación, se expondrá una tabla que desglosa los problemas que se han presentado en esta primera fase del proyecto. La tabla será un resumen del impacto de los desafíos que estarán clasificados en cinco niveles distintos. Cada nivel será asociado al tipo de daño.

1. Daño catastrófico: Las medidas que se tomarán en este caso serán de manera inmediata, puede provocar que el proyecto se detenga y retrase de forma considerable, teniendo que empezar desde cero.

2. Daño crítico: Se necesita tomar medidas que resuelvan el riesgo, puesto que puede producir retrasos en varias fases del proyecto.

3. Daño circunstancial: El riesgo, amerita que se solucione de inmediato, debido a que puede perjudicar en el desarrollo en una etapa base del proyecto.

4. Daño irrelevante: El problema no es esencial, puede ser una situación imprevista que puede solucionarse en cualquier momento.

5. Daño recurrente: El riesgo no es significativo, simplemente es reiterativo, puede retrasar en algunas tareas, pero no en etapas.

Tabla 5: Riesgos del proyecto

Riesgos	Nivel de impacto	Acción remedial
Abandono de personal	1	Reestructurar la gestión de tareas y recortar labores con menor importancia, para no sacrificar la eficacia del proyecto.
Pérdida de robot	1	Comprar un nuevo kit de lego spike prime con el dinero en conjunto del grupo y poner una nueva custodia al robot.
Atraso en el cumplimiento de tareas	2	Instaurar límites de tiempo y organizar las tareas parecidas para mejorar la eficiencia y reajustar los horarios para encontrar mayor tiempo en los encargos críticos en caso de atraso.
Ausencia repentina por fuerza mayor	2	Llenar ese rol con un integrante que tenga las capacidades que se requieren o retrasar tareas que se puedan postergar.
Rotura de piezas	3	Solicitar un reemplazo de las piezas rotas o comprar nuevas.
Inestabilidad del diseño del robot	3	Investigar un prototipo de mayor estabilidad o hacerle ajustes al mismo diseñado y repartir el peso a los puntos que provocan las inestabilidad.
Fallo en la programación del robot	4	Depurar el código para enmendar el fallo.
Problemas con la señal de internet	5	Probar con otras señales de la universidad, cambiar a una conexión privada o por vía cable ethernet.

4. Planificación de los Recursos

Para el desarrollo adecuado del proyecto es fundamental identificar.
Estos recursos se dividen en tres categorías principales: hardware, software y costos estimados.
En primer lugar, el hardware corresponde a todos los componentes físicos utilizados para llevar a cabo las actividades, como los Set Lego SPIKE Prime y equipos de computación utilizados para la programación y el control del sistema.

En segundo lugar, el software incluye las herramientas digitales y plataformas requeridas para programar, gestionar y operar, abarcando desde entornos de desarrollo hasta aplicaciones específicas de LEGO y VS Code.

Finalmente, la estimación de costos permite proyectar el presupuesto necesario para adquirir estos recursos, ofreciendo una visión clara del gasto asociado al desarrollo del proyecto y facilitando una correcta planificación económica.

4.1 Hardware

- Set Lego SPIKE Prime.
- Expansión de lego SPIKE Prime.
- Computador con el sistema operativo necesario para poder programar las instrucciones para el robot.
- Mandos de ps4.

4.2 Software

- Licencia Microsoft Office .
- VS Code.
- Lego Mindstorms.
- Lego Spike.

4.3 Estimación de Costos

Tabla 6: Costo de hardware

Producto	Precio (CLP)
Set “LEGO EDUCATION: Juego Spike Prime	$500.000
LEGO Education SPIKE Prime Expansion	$460.000
Notebook Samsung Essential Windows 11	$500.000
Notebook Lenovo ThinkPad T14	$320.000
Notebook HP 15-fc0004la AMD Ryzen 3 8GB 512GB SSD 15,6"	$349.999
Mandos de PS4	$60.000
Precio total	$2.189.999

Tabla 7: Costo de software

Producto	Precio (CLP)
Licencia Microsoft Office	$10.000
Precio total	$10.000

Tabla 8: Costo de personal de trabajo

Rol	Horas	Horas extra	Precio/Hora (CLP)
Jefe de proyecto	63 horas	19 horas	$50.000
Programador	63 horas	25 horas	$40.000
Ensamblador	63 horas	15 horas	$20.000
Documentador	63 horas	22 horas	$30.000
Total :	-	-	$11.730.000

La contabilización de horas comienza cuando se forma el grupo de trabajo, considerando como fecha de inicio el 09-09-2025. Las horas se registran según el tiempo dedicado en clases, mientras que las horas extras corresponden al trabajo realizado fuera del horario de clases dentro del mismo departamento de informática.

Tabla 9: Costos totales

Total de costos (CLP)	
Costo Hardware	$2.189.999
Costo Software	$10.000
Costo Empleados	$11.730.000
Total	$13.929.999

5. Análisis y diseño

5.1. Especificación de requerimientos

Para definir los requerimientos de la garra se tuvo en cuenta como problema a solucionar: “La falta de automatización en el proceso de carga de materiales”. El cliente interesado en financiar la implementación, son las empresas mineras y el usuario, encargado de utilizar el prototipo en faena, los trabajadores.

5.1.1. Requerimientos funcionales

- El robot debe poder cerrar y abrir su garra.
- El robot debe poder ajustar la altura de su brazo.
- El robot debe poder mover su garra horizontalmente.
- El robot debe poder recoger un bloque de lego determinado usando su garra.
- El robot debe dejar este mismo bloque de lego en una base o plataforma de carga.
- El usuario debe poder controlar todos los movimientos del robot y finalizar su funcionamiento, a través de la interfaz gráfica.
- El sistema debe validar las entradas del usuario y notificar cualquier error, mediante la interfaz gráfica o el robot.

5.1.2. Requerimientos no funcionales

Con el fin de asegurar que el sistema cumpla con criterios de calidad y rendimiento se establecieron los siguientes requerimientos no funcionales:

- El robot debe poder funcionar de manera continua durante al menos una hora.
- La latencia máxima entre la acción del usuario y el movimiento debe ser como máximo de 1000 ms.
- La estructura del robot debe mantenerse estable durante los movimientos.
- La interfaz gráfica debe ser intuitiva y fácil de usar, permitiendo al usuario identificar de forma autónoma como conectarse al robot y seleccionar el modo de control en al menos 5 minutos.
- La interfaz gráfica debe soportar conexión remota con Bluetooth.

- Los movimientos deben poder ser controlados, mediante teclado o mando.

5.2. Arquitectura de software

El sistema de software encargado del control de los movimientos del prototipo, se compone por la interfaz gráfica, el cliente, Bluetooth, el hub de Lego y la garra robótica.

El flujo que sigue el sistema es:

1.Interfaz gráfica: Recibe y válida las entradas del usuario.

2. Cliente: El computador cumple este rol, se encarga de procesar las entradas de la interfaz gráfica y convertirlas en comandos que serán enviados al servidor.

3. Bluetooth: Es el canal de comunicación, por el que se mandan los comandos desde el cliente hasta el servidor.

4. Servidor: El hub de lego cumple este rol. Procesa los comandos del cliente e indica a la garra robótica qué movimientos debe realizar.

5. Garra robótica: Realiza los movimientos físicos indicados por el servidor.

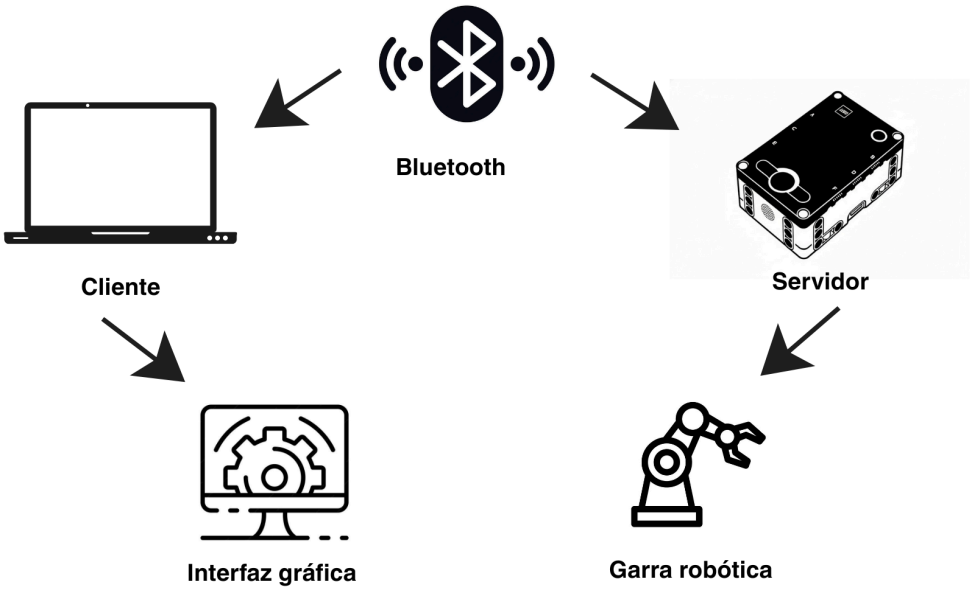

Figura 2: Diagrama Cliente-Servidor

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

Para esta etapa se desarrollaron bocetos de baja fidelidad con el objetivo de definir la estructura básica de la pantalla principal de la aplicación *CLAW-TY*. Estos bocetos iniciales permiten visualizar la organización general de los elementos sin centrarse aún en colores, tipografías ni detalles estéticos.

Un diseño de baja fidelidad utiliza únicamente formas simples como rectángulos, círculos y texto básico para representar botones, títulos o áreas de contenido. Este enfoque facilita evaluar la disposición de los componentes y comparar posibles alternativas antes del diseño final.

Para su elaboración se empleó un editor digital llamado canva, y se generaron cuatro propuestas distintas, cada una explorando un estilo diferente pero manteniendo la misma estructura funcional. La pantalla diseñada corresponde al menú de bienvenida de la aplicación, donde el usuario selecciona el modo de control: Teclado o Mando.

5.3.1 Bocetos de pantalla de inicio

5.3.1.1. Boceto 1 Diseño minimalista y claro

1. Fondo limpio y sin detalles para resaltar los botones.
2. Título “CLAW-TY” centrado en la parte superior.
3. Subtítulo “BIENVENIDO” debajo del título.
4. Dos botones principales (“TECLADO” y “MANDO”) ubicados de forma simétrica.
5. Enfoque en simplicidad y legibilidad.

Figura 3: Pantalla de inicio boceto 1

5.3.1.2. Boceto 2 — Variación con textura ligera

1. Mantiene la misma estructura funcional del primer boceto.
2. Se incluye un fondo con textura suave para dar profundidad.
3. Los botones y el texto continúan siendo simples, priorizando la claridad.
4. Permite evaluar una interfaz más visual sin perder funcionalidad.

Figura 4: Pantalla de inicio boceto 2

5.3.1.3. Boceto 3 — Estilo oscuro con alto contraste

1. Fondo negro o gris oscuro para aumentar el contraste visual.
2. Colores llamativos (por ejemplo, verde) para el título y los botones.
3. Proporciona una opción más moderna y orientada a videojuegos.
4. Sigue siendo un wireframe, ya que solo se representan las posiciones básicas de los elementos.

Figura 5: Pantalla de inicio boceto 3

5.3.1.4 Boceto 4 — Variante oscura con distribución más definida

1. Misma línea visual del boceto 3, pero ajustando proporciones, espaciado y forma de los botones.

2. Revisión del estilo para evaluar cuál de las dos versiones oscuras funciona mejor.

3. Explora equilibrio entre estética y legibilidad.

Figura 6: Pantalla de inicio boceto 4

5.3.1.5 Conclusión del diseño final interfaz de inicio

Luego de analizar las cuatro propuestas de wireframes de baja fidelidad, se decidió adoptar como base el segundo diseño. Este boceto mantiene una estructura simple y clara, pero incorpora una textura de fondo ligera que aporta mayor profundidad visual sin afectar la legibilidad ni distraer al usuario.

El segundo diseño logró un equilibrio adecuado entre simplicidad y estética, ofreciendo una interfaz limpia pero con suficiente personalidad para diferenciar la aplicación. Además, los botones y el título se mantienen bien organizados y fácilmente identificables, lo que facilita la navegación desde la primera interacción del usuario.

Por estos motivos, el equipo concluyó que el segundo boceto es el más adecuado para continuar con el desarrollo del diseño de alta fidelidad y la implementación final de la GUI.

5.3.2 Diseño del controlador por teclado

Para la sección correspondiente al modo de control mediante teclado, se desarrolló un boceto inicial de baja fidelidad con el objetivo de definir la distribución de las teclas asociadas al movimiento del brazo robótico. A diferencia de la pantalla de inicio que requirió cuatro versiones exploratorias, por lo tanto en este caso se generó un único boceto, ya que desde el primer planteamiento se logró una organización clara, funcional y fácil de interpretar para el usuario.
El diseño básico representó únicamente la posición de las teclas y los grupos de controles, sin emplear colores, efectos ni detalles estéticos. El diseño se organizó en torno a cuatro secciones principales: Base, Brazo, Codo, Garra, además del botón de Stop, distribuidos de forma que el usuario pudiera identificar rápidamente qué tecla corresponde a cada acción del robot.

Figura 7: Boceto pantalla de control por teclado

La simplicidad del boceto permitió visualizar de inmediato la lógica del control, lo que facilitó su selección como propuesta principal. No obstante, al avanzar en el desarrollo de la interfaz general de CLAW-TY, se consideró necesario actualizar este diseño para mantener una línea estética coherente con el estilo visual escogido para la pantalla de inicio (el segundo boceto).
En esta actualización se incorporaron elementos como:

1. Un fondo con textura similar al diseño final del menú de bienvenida.

2. Botones estilizados con bordes más suaves y colores consistentes con la identidad visual del proyecto.

3. Tipografía mejorada para reforzar la legibilidad.

4. Distribución más equilibrada para lograr una interfaz visualmente unificada.

Estas mejoras permitieron que la interfaz del controlador conservará su estructura funcional original, pero adoptará un aspecto más profesional y coherente dentro del conjunto de pantallas de la aplicación.

5.3.3 Conclusión del diseño del controlador por teclado

El diseño del controlador por teclado evolucionó de un boceto inicial sencillo pero funcional hacia una versión visualmente más pulida y coherente con la identidad gráfica de la aplicación *CLAW-TY*. Aunque la primera propuesta definió de manera efectiva la distribución y lógica de los controles, su actualización fue fundamental para garantizar una experiencia uniforme en todas las pantallas del proyecto.

La integración de elementos visuales consistentes como el estilo de botones, la tipografía y la textura del fondo permitió mantener la claridad del diseño original, pero aportando una presentación más profesional y armonizada con la interfaz de inicio. De esta manera, el controlador conserva su facilidad de uso y comprensión inmediata, a la vez que se adapta al estilo final de la aplicación.

En conjunto, este proceso aseguró que el controlador por teclado no solo cumpliera con los requisitos funcionales, sino que también contribuye a una experiencia de usuario coherente e intuitiva dentro de *CLAW-TY*.

6. Implementación

6.1. Fundamentos de los movimientos

Para que el robot lleve a cabo correctamente sus funciones, los principios físicos que debemos tener en cuenta principalmente son la fuerza y torque que debe realizar el brazo para poder acelerar y mover un objeto, en este caso bloques de lego de forma eficaz y segura, además del peso máximo de carga de la garra en base a la fuerza y fricción que esta genera sobre el bloque. Para estos cálculos se ignora el peso de las partes que conforman el robot, en otras palabras, la fuerza extra a la que debe realizar para moverse.

Velocidad y aceleración:
Para calcular la fuerza primero se necesita determinar el largo del brazo desde el punto de pivote, se aproxima un largo de r = 20(cm) = 0,2(m), además una velocidad deseada de movimiento y el tiempo en el que quiere acelerar a este. Tomando que se quiere dar un cuarto

de vuelta cada segundo se obtiene una velocidad de v = 0,3125(m/s). Tomando que se quiere llegar a esta velocidad en 0,2 segundos, se concluye una aceleración a = 1,5625 (m/s2).

Fuerza y torque:
con los datos r = 0,2(m), v = 0,3125(m/s) y a = 1,5625 (m/s2), podemos calcular la fuerza como F = m*a, y el torque como T = F*r = m*a*r = 1,5625 * 0,2 * m = 0,3125 * m(N). Obteniendo así que para que el brazo acelere desde el reposo a ¼ de vuelta por segundo, debe ejercer 0,3125*m(N).

Fricción y fuerza de garra:
En este caso se usará el coeficiente de fricción entre goma y UHMW, el cual es 0,56, ya que el UHMW tiene una fricción menor que un plástico normal, se tomará un coeficiente de fricción de 0,6. Junto a esto se usará el siguiente diagrama como referencia:

Con m siendo el bloque de masa m y su peso mg, los rectángulos laterales representando las garras, cada uno con fricción u y ejerciendo una fuerza Fg y causando las fuerzas de fricción estáticas fe, finalmente con una aceleración del sistema a. Cabe aclarar que como el bloque no se desliza de las garras, la fricción es estática.

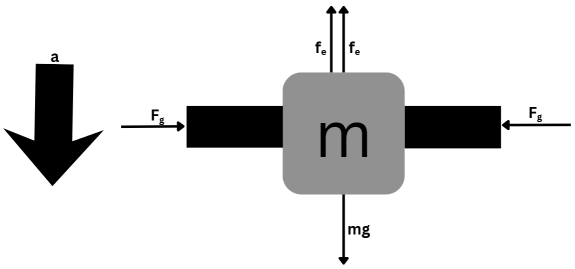

Figura 8: Diagrama de cuerpo libre

Sabemos que la suma de las fuerzas es $\Sigma F = ma$, si sumamos las fuerzas verticales quedamos con que $mg - 2fe = ma$, despejando fe tenemos: $fe = m(g-a)/2$. También se sabe que la fuerza de fricción estática $fe \leq Nu$, ya que la única fuerza que afecta a las garras son estas mismas, y asumiendo que las dos son simétricas, N = Fg, o sea que $fe \leq Fgu$, reemplazando fe y despejando para la masa: $m \leq 2Fgu/ga$. obteniendo que $m \leq 0,78Fg$. Con esto concluimos que la masa máxima que debe tener un bloque para que este no se caiga es de 0,78 veces la fuerza que ejerce una pata de la garra, o sea 0,39 veces la fuerza de la garra completa.

6.2. Descripción del sistema

En esta sección se presentarán los componentes del software utilizado para implementar la garra robótica.
El enlace para acceder al repositorio es : [github/Garra-Sp-11]

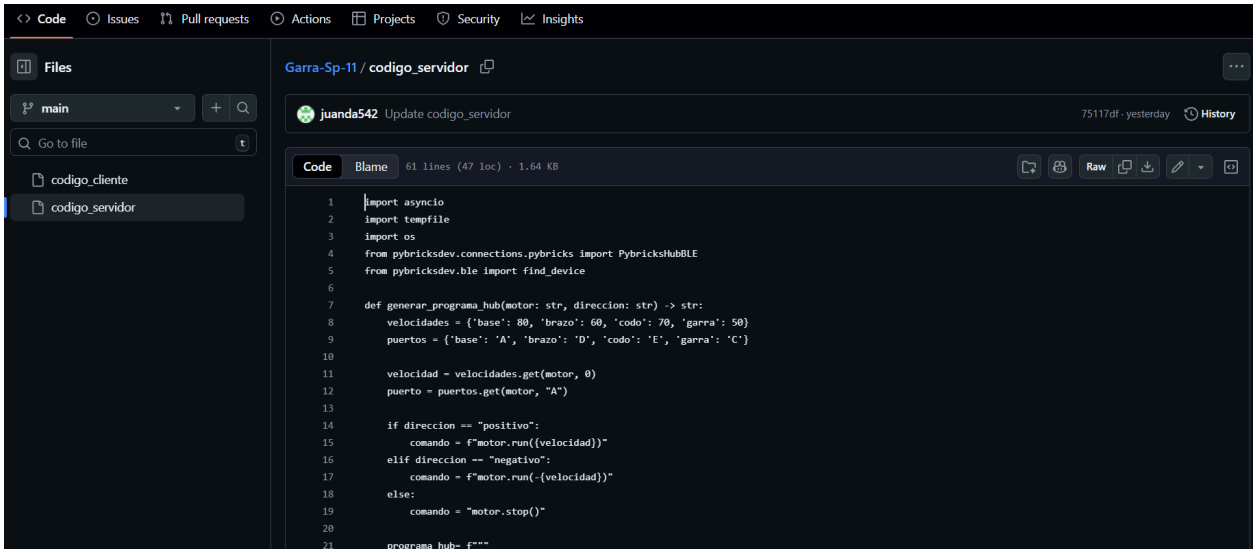

Figura 9: Imagen de referencia de github N1

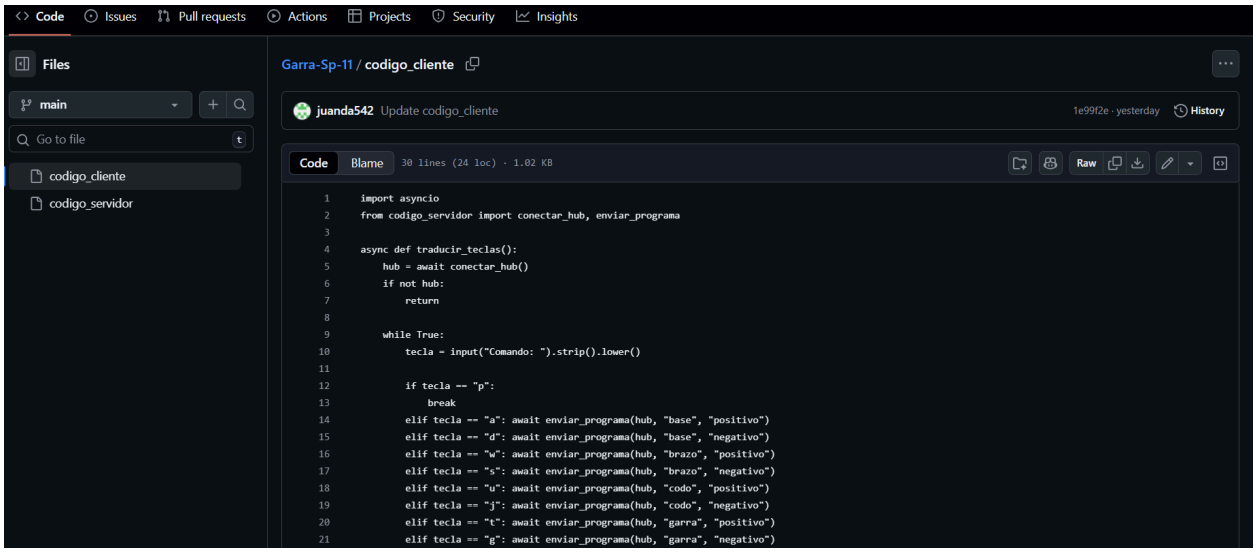

Figura 10: Imagen de referencia de github N2

6.2.1. Cliente

En la figura 11, se muestra la función que recibe las teclas ingresadas por el usuario y las traduce a un programa utilizable por el Hub Lego, para esto primero se conecta al servidor utilizando la función conectar_hub, luego comienza un bucle que recibe las entradas del usuario, dependiendo de la tecla que ingrese el usuario entrega los parámetros correspondientes al movimiento deseado a la función enviar_programa que crea y envía al servidor el programa con el código para ejecutar el movimiento, al pulsar p se finaliza el bucle y se cierra la conexión entre el cliente y el servidor.

```python
async def traducir_teclas():
    hub = await conectar_hub()
    if not hub:
        return

    while True:
        tecla = input("Comando: ").strip().lower()

        if tecla == "p":
            break
        elif tecla == "a": await enviar_programa(hub, "base", "positivo")
        elif tecla == "d": await enviar_programa(hub, "base", "negativo")
        elif tecla == "w": await enviar_programa(hub, "brazo", "positivo")
        elif tecla == "s": await enviar_programa(hub, "brazo", "negativo")
        elif tecla == "u": await enviar_programa(hub, "codo", "positivo")
        elif tecla == "j": await enviar_programa(hub, "codo", "negativo")
        elif tecla == "t": await enviar_programa(hub, "garra", "positivo")
        elif tecla == "g": await enviar_programa(hub, "garra", "negativo")
        else:
            print("Comando inválido.")

    await hub.disconnect()


if __name__ == "__main__":
    asyncio.run(traducir_teclas())
```

Figura 11: Código cliente

En la figura 12, se exhibe el código que se encarga de establecer la conexión entre el cliente y el servidor. Para esto se utiliza find_device, una función de pybricksdev que se encarga de devolver la información necesaria para conectarse por bluetooth al hub lego. Con esta información crea un objeto PybricksHubBle que representa al hub en el código y sobre este se ejecuta la función connect para establecer la conexión.

```python
async def conectar_hub():
    dispositivo_ble = await find_device(name="Sp-11")
    if not dispositivo_ble:
        print("No se pudo encontrar al Hub Sp-11")
        return None

    hub = PybricksHubBLE(dispositivo_ble)
    await hub.connect()
    print("Conexión al hub completa")
    return hub
```

Figura 12: Código de conexión bluetooth

En la figura 13, se observa la función enviar_progama, la cual utiliza el texto generado por la función generar_programa_hub, para crear un archivo python temporal denominado archivo_codigo_hub que será el que contenga el código que va a ejecutar el hub. La ruta de este archivo se guarda en la variable ruta y se le entrega como parámetro a la función run del hub para que lo ejecute. Finalmente se elimina el archivo temporal del sistema.

```python
async def enviar_programa(hub, motor, direccion):
    programa_hub = generar_programa_hub(motor, direccion)

    try:
        with tempfile.NamedTemporaryFile(delete=False, suffix=".py") as archivo_codigo_hub:
            archivo_codigo_hub.write(programa_hub.encode("utf-8"))
            ruta =  archivo_codigo_hub.name

        await hub.run(ruta)

    finally:
        if 'ruta' in locals() and os.path.exists(ruta):
            os.remove(ruta)
```

Figura 13: Código enviar programa al Hub

6.2.2 Servidor

En la figura 14, se observa la función que se encarga de generar un String con el código que será ejecutado en el hub. Para ello recibe como parámetros el motor y dirección a la que se va a mover, en función de estos define una velocidad negativa o positiva que indicará la dirección en la que se moverá el motor seleccionado. Las partes dinámicas del código corresponden a las variables puerto y comando, todo lo demás es constante para cualquier movimiento.

```
def generar_programa_hub(motor: str, direccion: str) -> str:
    velocidades = {'base': 80, 'brazo': 60, 'codo': 70, 'garra': 50}
    puertos = {'base': 'A', 'brazo': 'D', 'codo': 'E', 'garra': 'C'}

    velocidad = velocidades.get(motor, 0)
    puerto = puertos.get(motor, "A")

    if direccion == "positivo":
        comando = f"motor.run({velocidad})"
    elif direccion == "negativo":
        comando = f"motor.run(-{velocidad})"
    else:
        comando = "motor.stop()"

    programa_hub= f"""
from pybricks.hubs import PrimeHub
from pybricks.pupdevices import Motor
from pybricks.parameters import Port
from pybricks.tools import wait

hub = PrimeHub()
motor = Motor(Port.{puerto})

{comando}
wait(300)
motor.stop()
"""
    return programa_hub
```

Figura 14: Código generar programa para el hub

6.2.3. Interfaz gráfica de usuario (GUI)

6.2.3.1 Pantalla de Inicio – Menú Principal

Figura 15: Pantalla de inicio

La pantalla de inicio constituye el punto de entrada principal a la aplicación **CLAW-TY**. Su diseño adopta la estética final seleccionada, basada en el segundo boceto de baja fidelidad: una interfaz clara, con fondo texturizado suave y componentes centrados para favorecer la legibilidad.

Componentes y funciones

1. Título principal “CLAW-TY” y Subtítulo “BIENVENIDO”

Se ubica en la parte superior y cumple la función de identificar la aplicación. Su tamaño prominente atrae la atención del usuario y establece un punto de referencia visual mientras que el Bienvenido es presentado justo debajo del título, actúa como mensaje introductorio y contribuye a una bienvenida clara y directa al usuario.

2. Botón “TECLADO”

Es uno de los dos botones principales del menú su función es acceder al modo de control mediante teclado, redirigiendo a la pantalla donde se visualizan las teclas asignadas para manipular la garra robótica.Este es el acceso directo al controlador descrito en la siguiente sección.

3. Botón “MANDO”

Permite seleccionar el modo de control mediante un mando físico en la GUI su rol es ofrecer al usuario una alternativa de control distinta al teclado.

4. Botón “SALIR”

Ubicado en la esquina inferior derecha, destaca por su color rojo para resaltar su función crítica al presionarlo, la aplicación se cierra por completo Su ubicación evita activaciones accidentales y mantiene coherencia con prácticas estándar de diseño.

6.2.3.2 Pantalla del Controlador por Teclado

Figura 16: Pantalla controlar por teclado

Esta pantalla aparece cuando el usuario selecciona el modo “TECLADO” en el menú principal. Su objetivo es mostrar de forma clara la distribución de teclas asignadas a cada parte del brazo robótico, manteniendo el estilo visual definido para la interfaz.

Componentes y funciones

1. Título “CONTROLADOR DE LA GARRA”

Indica de manera explícita la función de esta pantalla.Se divide en dos líneas para facilitar la lectura y mantener el equilibrio en la visual.

2. Secciones de control

La pantalla organiza las teclas según la parte del brazo robótico que controlan.
Cada sección incluye un rótulo descriptivo y dos botones que representan las teclas físicas del teclado.

1-BASE teclas A y D

Controlan el giro de la base del brazo robótico.

- **A:** movimiento hacia la izquierda.
- **D:** movimiento hacia la derecha.

Esta agrupación facilita la interpretación inmediata por parte del usuario.

▪ BRAZO – teclas W y S

2-Controlan el desplazamiento vertical del brazo:

- **W:** levantar el brazo.
- **S:** descender el brazo.

La disposición vertical coincide con la lógica del movimiento.

3-CODO – teclas U y J

Controlan la articulación intermedia del brazo:

- **U:** elevar el codo.
- **J:** bajar el codo.

Se colocan más abajo para mantener jerarquía visual coherente con la estructura del brazo robótico.

4- GARRA – teclas G y T

Permiten abrir o cerrar la garra:

- **G:** cerrar.
- **T:** abrir.

La separación respecto a las demás secciones ayuda a distinguir este módulo crítico.

5- STOP – tecla P

Botón central destacado que frena cualquier movimiento en ejecución.
Su función es esencial para la seguridad operativa y control inmediato del robot.

6- Botón “VOLVER”

Ubicado en la parte inferior derecha, permite retornar al menú principal sin cerrar la aplicación.
Su color rojo coincide con el estilo del botón “SALIR”, pero su etiqueta evita confusión.
Facilita la navegación y evita que el usuario quede bloqueado en esta pantalla.

7. Resultados

7.1. Estado actual del proyecto

De acuerdo al seguimiento de las bitácoras semanales, se logró el desarrollo de la máquina de garra usando el kit de LEGO SPIKE PRIME, cumpliendo los objetivos que se tenía en la primera etapa.

En la primera fase, se definieron los objetivos del proyecto, las bibliotecas necesarias para la codificación del sistema y se realizó el prototipo inicial de la garra, logrando que el robot cumpla con todos los requerimientos como una estructura sólida y funcional capaz de realizar todos los movimientos necesarios.

Durante la siguiente etapa, se realizaron pruebas del funcionamiento de la estructura, usando el código para corroborar si se encontraba alguna deficiencia en el robot. Se comprobó el estado de los movimientos de la garra, el agarre y la estabilidad del sistema, las cuales sirvieron para realizar ajustes sucesivos en el diseño físico y en la programación, además de las actualizaciones de la carta Gantt y del informe del proyecto.

Luego, en una etapa más avanzada, se migró el código de bloques a Python, para mejorar la flexibilidad y control sobre el comportamiento del robot, siguiendo con la investigación del diseño gráfico, que actualmente se encuentra terminada.

Por último, se logró establecer la conexión del robot por vía bluetooth, y actualmente se están desarrollando pruebas para mejorar la fluidez y precisión de los movimientos del robot con diferentes versiones de código, además del controlador externo, que se está desarrollando con un control mediante un mando de Xbox.

En síntesis, el proyecto cuenta con una estructura mecánica operativa, movimientos funcionales, conexión inalámbrica estable y el diseño de la interfaz gráfica finalizado. Queda pendiente la integración completa de la interfaz, el control por mando y el sistema de movimiento del robot.

7.1.2 Relación con los requerimientos funcionales

En relación con los requerimientos funcionales en la etapa de especificación, se puede apreciar que el robot cumple con la apertura y cierre de la garra, el ajuste de la altura del brazo y el desplazamiento horizontal de la garra. Asimismo, el sistema del robot es capaz de recoger un bloque de lego y desplazarlo hasta una base o plataforma de carga; por lo cual, cumple con los requerimientos relacionados a la manipulación de materiales.

Respecto al control del sistema, el usuario cuenta con la opción de interactuar con el robot por medio de la interfaz gráfica, la cual tiene el diseño completado; sin embargo, la integración completa de los movimientos a través de la interfaz aún se encuentra en desarrollo. Igualmente, la validación de entradas del usuario y la notificación de errores se encuentran implementadas de manera parcial, ya que aún se encuentra trabajando con la comunicación efectiva de la interfaz y el robot.

De modo que los requerimientos mecánicos y de movimientos del sistema se encuentran mayormente cumplidos, a diferencia de los requerimientos asociados al control del sistema avanzado y a la validación mediante la interfaz gráfica, que están en un estado de desarrollo parcial.

7.2. Problemas encontrados y solucionados

Durante el desarrollo del proyecto se identificaron varios problemas técnicos y organizativos, los cuales se registraron en las bitácoras semanales. A continuación, se describen los problemas más importantes junto con las soluciones aplicadas o en el proceso de implementación.

El primer problema enfrentado fue la dificultad para programar correctamente el movimiento y el agarre del robot, especialmente en la coordinación de los motores. Para solucionar este problema se tuvo que realizar investigaciones sobre el funcionamiento de los motores del kit LEGO SPIKE PRIME, donde se tuvo que llevar a cabo varias versiones de código para entender su comportamiento y así mejorar el control del sistema.

En las etapas iniciales, también se encontraban fallos en la estructura del robot, las cuales estaban afectando la estabilidad general. El problema fue solucionado revisando las instrucciones de su construcción, así mismo, también se identificó que el software

de LEGO SPIKE se encontraba desactualizado y se tuvo que actualizar para garantizar la compatibilidad con el hardware.

Durante las pruebas de los movimientos que se le realizaba al robot, también se evidenció que la potencia ejercita a los motores, desprendía algunas piezas del robot, por lo cual se tuvo que regular la potencia desde el código para mantener un movimiento más controlado seguro.

Relacionado al software, se presentaron limitaciones con el entorno de LEGO, ya que el entorno de LEGO Mindstorm, utiliza una versión limitada de MicroPython, lo que restringe algunas funcionalidades para el control de dispositivos externos. Además, surgieron dificultades
al intentar programar los movimientos del robot mediante teclado y al ejecutar el código de Visual Studio Code, debido a que no se reconocía los motores y fallaba la conexión de las terminales del robot.

Otro de los problemas encontrados fue que los movimientos del robot solo respondiera cuando el botón se encontrara presionado, para limitar los movimientos y evitar los errores estructurales. Este problema aún se encuentra vigente y se está realizando ajustes iterativos con versiones de código.

Finalmente, se encontraron problemas para integrar la interfaz gráfica, tanto en el control de los movimientos como el diseño visual utilizando la biblioteca de Tkinter. Estos problemas se encuentran en desarrollo, pero lo del diseño visual usando Tkinter se encuentra terminado, solo se encuentra pendiente poder controlar los movimientos con la interfaz gráfica.

7.2.1 Relación de los problemas con riegos del proyecto

Los problemas que representaron riesgos relevantes en el desarrollo del proyecto son: la inestabilidad del robot, la dificultad de poder controlar adecuadamente los motores y los retrasos que se produjeron en el trabajo de tareas. Particularmente, la potencia excesiva de los motores y las limitaciones del entorno de programación que comprometen en la continuidad del desarrollo del proyecto.

Las soluciones que se aplicaron permitieron mitigar algunos riesgos mediante ajustes estructurales como: regulación de potencia desde el código, actualización del software y reorganización de las tareas en el equipo, pero teniendo aún los problemas relacionados a la integración de la interfaz gráfica y el control del robot mediante un dispositivo externo, las cuales se siguen trabajando en su desarrollo.

8. Conclusión

En síntesis con todo lo mencionado anteriormente, se logró planificar, diseñar y construir una máquina de garra robótica orientada a la simulación de movimientos básicos a tareas del ámbito minero. Para ello se utilizó el kit de LEGO SPIKE PRIME como plataforma principal.

El sistema desarrollado cuenta con una estructura capaz de realizar movimientos verticales y horizontales, junto con un mecanismo de agarre que cumple con los requerimientos iniciales del proyecto. Asimismo, se avanzó con el desarrollo del software, migrando la programación de bloques a Python, establecer una conexión inalámbrica por vía Bluetooth y completar el diseño de la interfaz gráfica, la cual facilitara la interacción del usuario con el robot.

Para llegar a estos resultados, se necesitó un proceso de análisis, pruebas y ajustes, donde se identificaron problemas técnicos, relacionados con la programación de los motores, la estabilidad de la estructura del robot, la compatibilidad del software y las limitaciones que entrega el entorno de programación LEGO. Estas dificultades, fueron abordadas por medio de una investigación, pruebas iterativas, revisión de la estructura física y organización del trabajo en equipo.

A pesar de los avances alcanzados, aún se encuentran pendientes tareas relevantes para la siguiente etapa como la integración completa de la interfaz gráfica, el control físico del robot y la optimización de los movimientos por medio de un mando de Xbox. Estas actividades permitirán consolidar el sistema para tener esta simulación de manera más robusta, precisa, segura y confiable para fortalecer la utilidad de herramienta de simulación para el ámbito minero a la cual se está formando.

8.1 Reflexión sobre aprendizaje

El desarrollo de este proyecto nos ayudó a fortalecer conocimientos en áreas como robótica, programación en Python, control de motores, diseño de estructuras mecánicas y desarrollo de interfaces gráficas. Asimismo, se logró comprender la importancia de la planificación, la documentación de las bitácoras y la gestión de riesgos en proyectos.

En el ámbito grupal, el trabajo colaborativo y la distribución de los roles, al igual que las modificaciones, fueron fundamentales para enfrentar dificultades técnicas y mantener el avance en el proyecto. Además, la experiencia obtenida fue relevante para comprender problemáticas reales relacionadas a la automatización de procesos industriales, como lo es la minería.

9. Referencias

Banco Central de Chile. (2024). *Cuentas Nacionales: Participación sectorial del PIB*. Base de Datos Estadísticos (BDE)

Ebay. (s.f.). *Página de compra de LEGO Education SPIKE Prime*.

Líder. (s.f.). *Página de compra de LEGO Education SPIKE Prime Expansión*.

Abc. (s.f.). *Página de compra de Notebook Samsung Essential Windows 11*. https://www.abc.cl/notebook

Tecnoboss. (s.f.). *Página de compra de Notebook Lenovo ThinkPad T14*. https://tecnoboss.cl/notebook

Abc. (s.f.). *Página de compra de Notebook HP 15-fc0004la AMD Ryzen 3 8GB 512GB SSD 15,6*. https://tecnoboss.cl/notebook-lenovo-thinkpad

Paris. (s.f.). *Página de compra de mandos PS4*. https://www.paris.cl/control-ps4

Softpro. (s.f.). *Página de compra de licencia Office 2024 Professional*. https://softpro.cl/producto/office-2024-professional