**INFORME ll**



**Plan de Proyecto:**

**Robot EV3 “Selector de colores”**



**Integrantes:** Sebastián Cáceres, Rubén Salas, Aaron Saravia, Mario Villalobos

**Profesor:** Leonel Alarcón Bravo.

**Asignatura:** Proyecto 1.

**Fecha:** 3 de noviembre.

2022

Tabla de Contenidos.

1. Panorama General \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 1
	1. Introducción\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_1
	2. Objetivos\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_1
		1. Objetivo general\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_1
		2. Objetivo especifico\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_1
	3. Restricciones\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2
	4. Entregables\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2
2. Organización del Personal\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_3
	1. Descripción de los roles\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_3
	2. Personal que cumplirá los roles\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_3
	3. Mecanismos de comunicación\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_3
3. Planificación del Proyecto \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_4
	1. Actividades \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_4
	2. Asignación de tiempo\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_4
	3. Gestión de riesgos\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_5
4. Planificación de los Recursos\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_6
	1. Hardware\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_6
	2. Software\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_6
	3. Estimación de costos \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_6
5. Análisis y diseño\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
	1. Especificación de requerimiento\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
	2. Arquitectura\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
	3. Interfaz\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
6. Implementación\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
	1. Fundamentos de Proyectiles\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
	2. Descripción de los programas\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
	3. Diagramas\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
7. Resultados\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
	1. Estado Actual del Proyecto\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
	2. Problemas encontrados y soluciones propuestas\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
8. Conclusión\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
9. Referencias\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Panorama General

**1.1. Introducción**

EV3 es un módulo del plan de estudios diseñado para enseñar habilidades básicas de razonamiento y lógica de programación informática utilizando un contexto de ingeniería robótica. Contiene una secuencia de 10 proyectos (más un desafío final) organizados en torno a conceptos clave de robótica y programación.

La historia se remonta hasta 1986, que fue cuando la compañí­a lanzó un producto de LEGO controlado por un ordenador.

Desde entonces no han parado de presentar productos, aunque la esencia sigue siendo la misma: ayudar a los niños a aprender a programar y a conocer los sensores y actuadores más básicos.

**1.2. Objetivos**

**1.2.1. Objetivo General**

Construir y programar un robot que selecciona colores con la ayuda de EV3 mindstorms y el lenguaje de programación Python mediante la interfaz de Tkinter.

**1.2.2. Objetivo Especifico**

* Construir un robot con la capacidad de poder separar cubos por colores.
* Poder programar el robot usando el lenguaje de programación Python.
* Poder almacenar instrucciones con código en una memoria SD.
* Poder manejarlo a través de un dispositivo móvil para tener más control sobre este.
* Conseguir implementar el código Python en la interfaz de Tkinter para el funcionamiento del robot.



**1.3. Restricciones**

* La limitación al escoger algún lenguaje de programación ya que solo se puede usar estrictamente el lenguaje de programación Python.
* Codificar el robot de manera que pueda hacer tareas según la construcción para la cual fue hecho, no se pueden generar instrucciones que el robot no puede hacer físicamente.
* El robot tiene que ser controlado desde una distancia limitada ya que de estar muy lejos o podrá seguir bien las instrucciones que le demos desde el dispositivo de control.
* El límite de tiempo que tendremos para realizar la construcción y programación del robot

**1.4. Entregables**

* Bitácoras semanales (del avance del proyecto del robot).
* Idea física del robot de cómo se verá a futuro.
* Funciones principales que el robot ejecutaría al ser controlado mediante código Python en este caso (Escanear piezas, Ordenarlas dependiendo de su color).



2. Organización del Personal

Nos distribuiremos el trabajo entregando tareas distintas para cada integrante del grupo, teniendo en cuenta que al ser un equipo debemos velar por el apoyo mutuo en estas tareas para un mejor trabajo colaborativo.

**2.1. Descripción de los Roles**

Para poder llevar a cabo este proyecto asignamos roles a cada integrante.

* Jefe de grupo: Es el representante del equipo, el encargado de mantener una buena organización en el grupo y tomar decisiones.
* Ensamblador: Es el encargado de unir, armar y darle forma al robot para un buen funcionamiento.
* Programador: Es el encargado de guiar la mayor parte del código, para que a su vez el robot tenga un funcionamiento optimo.
* Documentador: Es el encargado de documentar toda la información escrita (Avances) en bitácoras, e informes.

**2.2. Personal que cumplirá los Roles**

* Jefe de Grupo: Ruben Salas
* Ensamblador: Mario Villalobos
* Programador: Sebastián Cáceres
* Documentador: Aaron Saravia

**2.3. Mecanismo de Comunicación**

Para poder llevar a cabo el proyecto de manera eficaz se estableció como medio de comunicación principal la aplicación Telegram, además del Gmail institucional, y a su vez nosotros tenemos como mecanismo y herramientas de comunicación un grupo de la Aplicación WhatsApp (Mensajería), Discord (Llamadas), siendo estas dos las más rápidas y fáciles de manejar.



3. Planificación del Proyecto

**3.1. Actividades**

Entre las actividades que estarán presentes tenemos durante el desarrollo del proyecto están:

* La designación de rol de cada integrante
* El ensamblado del Ev3
* La documentación de las bitácoras
* El desarrollo del informe 2
* La escritura del código con el lenguaje Python
* Implementación en la interfaz Tkinter
* Solución de errores en el ensamblaje y en la codificación.

Estas actividades y algunas más serán las que tendremos, y cada una de ellas podrá ser asistida por uno o más integrantes del grupo, en caso de que alguien necesite ayuda, siendo así más flexibles a la hora de progresar en cada actividad.

**3.2. Asignación de Tiempo**

Dentro de las horas asignadas para las clases, elegiremos las actividades, los días y el tiempo en que se llevaran a cabo las actividades.

Por lo general los días Lunes, jueves y viernes después de horario de clases.



**Carta Gantt**

****

**3.3. Gestión de Riesgos**

|  |  |  |
| --- | --- | --- |
| **Riesgos** | **Descripción** | **Soluciones** |
| Errores al armar EV3. | Presencia de piezas chuecas o mal puestas en el equipo. | Unión del grupo para arreglar el error lo antes posible. |
| Ausencia de integrantes. | Caso en el que un integrante no esté presente y no cumpla su actividad. | Reasignar las actividades para no ralentizar su desarrollo. |
| Fallos de código. | Errores en el código de Python. | Investigación del error por parte del encargado del código, asistencia del grupo en caso de ser necesario. |

4. Planificación de los Recursos

**4.1. Hardware**

Entre el Hardware que utilizaremos en el proyecto tenemos un Lego Ev3, un USB Wireless, sensor de movimiento y un computador para poder escribir el código.

**4.2. Software**

El Software que estará implementado en este proyecto será Python, escribiremos el código con este lenguaje de programación, para darle instrucciones al Ev3.

**4.3. Estimación de Costos.**

* Mindstorm Ev3 Core Set 45544 new **$500.000-$1.300.000**
* Memoria SD $**5.000- $12.000**
* Piezas de repuesto **$25.000-$50.000**
* Adaptador de memoria SD para micro SD **$1.500**
* Computador **$1.000.000**





5. Análisis y Diseño

**5.1. Especificación de requerimiento**

El sistema requiere del equipo Ev3 y el software Visual Studio Code con la extensión de Micro Python.

Este tiene la capacidad de seguir instrucciones solo en su lenguaje de programación de fábrica y con el uso de una computadora para la creación del código.

**5.2. Arquitectura**



**5.3. Interfaz**

Cuando inicia el programa la primera ventana que se nos muestra es esta con un solo boton que calibra el robot.



Una vez que se calibra salta una notificación que nos indica que se terminó el proceso de calibración.



Luego nos muestra dos botones más que son Sortear colores y Apagar.



El botón de Sortear colores abre una nueva ventana y nos muestra todos los colores escaneados.



Y el botón Apagar cancela todo y cierra el programa.



6. Implementación

**6.2. Descripción de los programas**

Los programas que utilizamos son Visual Studio Code y Micro Python. Y a continuacion tenemos el codigo:

#!/usr/bin/env pybricks-micropython

from pybricks.hubs import EV3Brick

from pybricks.ev3devices import Motor, TouchSensor, ColorSensor

from pybricks.parameters import Port, Button, Color, ImageFile, SoundFile

from pybricks.tools import wait

# Los objetos coloreados son rojos, verdes, azules o amarillos.

POSSIBLE\_COLORS = [Color.RED, Color.GREEN, Color.BLUE, Color.YELLOW]

# Inicialice el ladrillo EV3.

ev3 = EV3Brick()

# Inicializa los motores que accionan la cinta transportadora y expulsan los objetos.

belt\_motor = Motor(Port.D)

feed\_motor = Motor(Port.A)

# Inicialice el sensor táctil. Se utiliza para detectar cuando el motor de la correa tiene

# movió el módulo clasificador completamente hacia la izquierda.

touch\_sensor = TouchSensor(Port.S1)

# Inicialice el sensor de color. Se utiliza para detectar el color de los objetos.

color\_sensor = ColorSensor(Port.S3)

# Este es el bucle principal. Te espera para escanear e insertar 8 objetos de colores.

# Luego los ordena por color. Luego, el proceso comienza de nuevo y puede escanear

# e inserte el siguiente conjunto de objetos coloreados.

while True:

 # Coloque el motor de avance en la posición de inicio correcta.

 # Esto se hace haciendo funcionar el motor hacia adelante hasta que se pare. Este

 # significa que no puede moverse más. A partir de este punto final, el motor

 # gira hacia atrás 180 grados. Entonces está en la posición inicial.

 feed\_motor.run\_until\_stalled(120, duty\_limit=50)

 feed\_motor.run\_angle(450, -200)

 # Coloque el motor de la cinta transportadora en la posición de inicio correcta.

 # Esto se hace haciendo funcionar primero el motor de la correa hacia atrás hasta que el sensor táctil

 # se presiona. Entonces el motor se detiene y el ángulo es

 # poner a cero. Esto significa que cuando gira hacia atrás a cero más tarde

 # encendido, vuelve a esta posición inicial.

 belt\_motor.run(-500)

 while not touch\_sensor.pressed():

 pass

 belt\_motor.stop()

 wait(1000)

 belt\_motor.reset\_angle(0)

 # Cuando escaneamos los objetos, almacenamos todos los números de color en una lista.

 # Comenzamos con una lista vacía. Crecerá a medida que le agreguemos colores.

 color\_list = []

 # Este bucle escanea los colores de los objetos. Se repite hasta 8 objetos

 # se escanean y se colocan en el conducto. Esto se hace repitiendo el bucle.

 # mientras que la longitud de la lista sigue siendo inferior a 8.

 while len(color\_list) < 8:

 # Muestra una flecha que apunta al sensor de color.

 ev3.screen.load\_image(ImageFile.RIGHT)

 # Mostrar cuántos objetos de colores ya hemos escaneado.

 ev3.screen.print(len(color\_list))

 # Espere a que se presione el botón central o se escanee un color.

 while True:

 # Almacene True si se presiona el botón central o False si no.

 pressed = Button.CENTER in ev3.buttons.pressed()

 # Guarde el color medido por el sensor de color.

 color = color\_sensor.color()

 # Si se presiona el botón central o se detecta un color, y

 # salir del bucle.

 if pressed or color in POSSIBLE\_COLORS:

 break

 if pressed:

 # Si se presionó el botón, termine el ciclo antes. ya no lo haremos

 # esperar a que los objetos restantes se escaneen y se agreguen al

 # rampa.

 break

 # De lo contrario, se escaneó un color. Así que lo agregamos (añadimos) a la lista.

 ev3.speaker.beep(1000, 100)

 color\_list.append(color)

 # No queremos registrar el mismo color una vez más si todavía estamos

 # mirando el mismo objeto. Así que antes de continuar, esperamos hasta que el sensor

 # ya no ve el objeto.

 while color\_sensor.color() in POSSIBLE\_COLORS:

 pass

 ev3.speaker.beep(2000, 100)

 # Mostrar una flecha apuntando al botón central, para preguntar si hemos terminado.

 ev3.screen.load\_image(ImageFile.BACKWARD)

 wait(2000)

 # Reproduce un sonido y muestra una imagen para indicar que hemos terminado de escanear.

 ev3.speaker.play\_file(SoundFile.READY)

 ev3.screen.load\_image(ImageFile.EV3)

 # Ahora ordena los ladrillos de acuerdo a la lista de colores que almacenamos.

 # Hacemos esto repasando cada color de la lista en un bucle.

 for color in color\_list:

 # Espere un segundo entre cada acción de clasificación.

 wait(1000)

 # Haga funcionar el motor de la cinta transportadora en la posición correcta según el color.

 if color == Color.BLUE:

 ev3.speaker.say('blue')

 belt\_motor.run\_target(500, 10)

 elif color == Color.GREEN:

 ev3.speaker.say('green')

 belt\_motor.run\_target(500, 132)

 elif color == Color.YELLOW:

 ev3.speaker.say('yellow')

 belt\_motor.run\_target(500, 360)

 elif color == Color.RED:

 ev3.speaker.say('red')

 belt\_motor.run\_target(500, 530)

 # Ahora que la cinta transportadora está en la posición correcta, expulse el

 # objeto coloreado.

 feed\_motor.run\_angle(1500, 180)

 feed\_motor.run\_angle(1500, -180)

**6.3. Diagramas**



7. Resultados

**7.1. Estado actual del Proyecto**

El proyecto está en la etapa de culminación teniendo listo el robot y la interfaz hecha por la extensión Tkinter en Python lo que nos quedaría por hacer seria ir mejorando el código y así mismo mejorar la interfaz gráfica para que sea más didáctico y cómodo a la vista del espectador.

**7.2. Problemas encontrados y soluciones propuestas**

|  |  |
| --- | --- |
| **Problema Encontrado** | **Solución Propuesta** |
| No poder ingresar la tarjeta microSD al computador. | Adquirir un adaptador (tarjeta) de micro SD. |
| Ausencia de piezas en la caja del robot. | Solicitar las piezas faltantes al docente. |
| Choque de horario. | Reorganizar el tiempo de cada integrante para encontrar un tiempo libre fuera del horario de clase. |
| El robot Ev3 no expulsa las piezas cuando debe hacerlo, posiblemente por problemas en el código de Micropython. | Revisar el código hasta encontrar el error probando con distintos métodos hasta que funcione. |

8. Conclusión

Para poder concluir podemos afirmar que al empezar con el proyecto nos vimos en la problemática de no conocer este material (LEGO) , la manera en que se usa las piezas para llegar a construir un robot optimo, nos inspiramos en una grúa pero al cabo del tiempo nos dimos cuenta que otro grupo realizando el mismo modelo así que optamos y tomamos el riesgo de cambiar el proyecto y hacer uno que pudiera leer los colores para encajarlos en su lugar correspondiente (Selector de colores), uno de los problemas mas importantes que tuvimos fue el tema del choque de horarios que teníamos 3 de los 4 integrantes del grupo, por lo que tuvimos que reorganizar nuestro tiempo para encontrar un tiempo libre en el que coincidamos todos para poder avanzar en el proyecto fuera del horario de clase, ya sean las bitácoras, el avance del informe hasta el funcionamiento del robot mediante la interfaz.

9. Referencias

* Manual de robot selector de colores
https://es.slideshare.net/maestro21edu/proyecto-selector-de-colores-lego-mindstorm-ev3
* Python Programming Lenguage
<https://www.python.org/>