Informe Taller 2 - Robótica

Grupo 8

April 2, 2022

Listado de materiales

Elemento	Dimensiones [LxWxH] (cm)
Arduino Mega	(8 x 4 x 1.5)
Raspberry Pi 4	(9 x 5.5 x 3)
Motor DC (2)	(7 x 2 x 2)
Encoder (2)	(1.5 x 2 x 2)
Jumpers (16)	25
Llantas de precisión (2)	(6.6 x 2 x 6.6)
Base de acrílico	(20 x 20 x 0.4)
Puente H (L293D)	(4 x 4 x 3)
Baterías de Litio (2)	(6 x 1.5 x 1.5)

Table 1: Listado de materiales

Plano mecánico

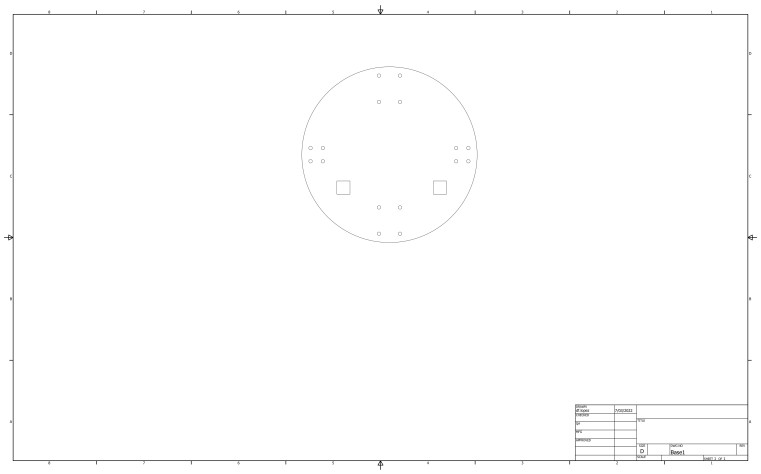

Figure 1: Plano mecánico de QWERTY

En el plano mecánico, se utilizó una base de acrílico circular de un radio de 10cm. Además, cada llanta tiene un ancho de 2.5cm (estableciendo un robot de tamaño 20x25cm) y 5.5cm de altura. También se agregaron 2 ruedas castor,

uno en la parte posterior y otro en la parte anterior del robot para que esté equilibrado. La finalidad de realizarlo de manera circular era tener el mismo centro de masa para el ancho y el largo del robot.

La parte mecánica fue basada en un robot diferencial con dos movimientos: Avanzar/retroceder y rotar a la izquierda/derecha. Más adelante se explicará con detalle cómo se realizaba el movimiento de los motores para la parte mecánica.

Plano electrónico

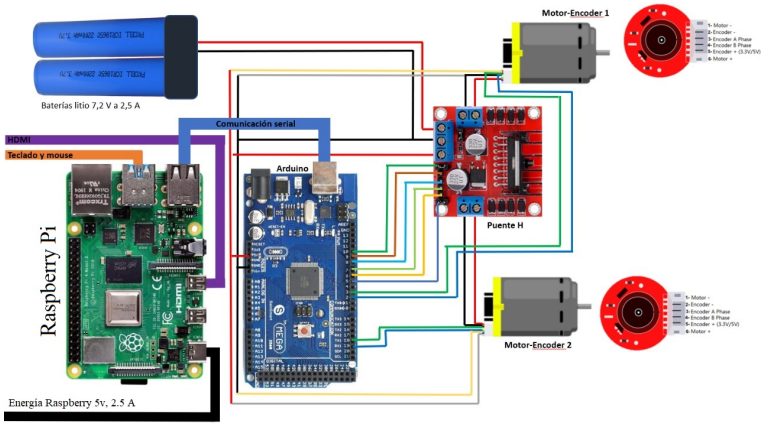

Figure 2: Plano electrónico de QWERTY

Como se observa en la figura 5, se utilizó un Arduino Mega para encargarse del movimiento de los motores, empleando un puente H para controlar con mayor facilidad estos motores por medio de los encoders. Mientras que el Arduino está siendo alimentado por la Raspberry, los motores están alimentados por dos baterías de Litio conectadas en serie; cabe resaltar que esta alimentación es distribuida por medio del puente H.

Relación parte mecánica y electrónica

Para realizar un movimiento congruente, fue necesario conectar la parte mecánica con la electrónica. Para esto, se utilizaron dos motores DC con *encoders* conectados a dos llantas por medio de un acople metálico. Para mantener el equilibrio, se tienen las dos ruedas castor. Para el control mecánico, los *encoders* envían pulsos por dos canales (A) y (B), correspondientes a la llanta derecha e izquierda, respectivamente. Los *exencoders* usan sensores de efecto Hall para determinar y controlar posición y/o velocidad. Esta información es recibida por pines digitales del Arduino. Y por medio del puente H se alimenta y se controla dirección de cada uno de los motores.

Al realizar el ensamblaje electrónica con la mecánica, se obtuvo el ensamble final:

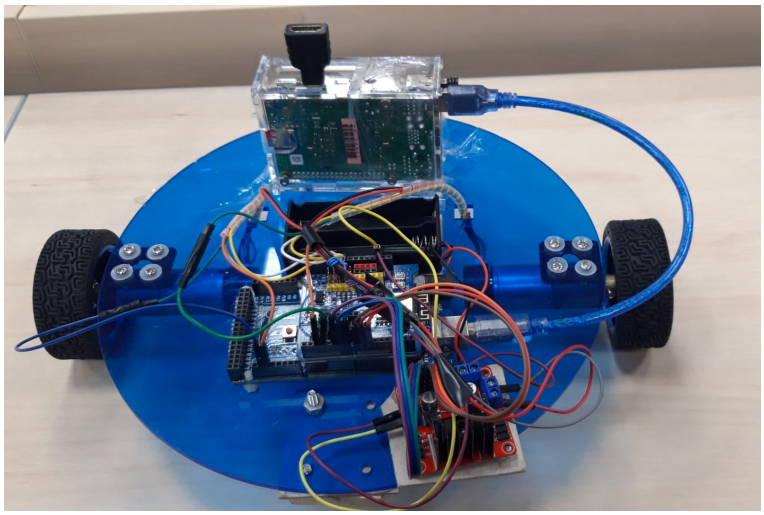

Figure 3: Ensamble final de QWERTY

Cabe resaltar que se va a realizar un cambio de base a una más pequeña, por lo que la localización de varios elementos sobre la base pueden cambiar.

Funcionamiento general del robot:

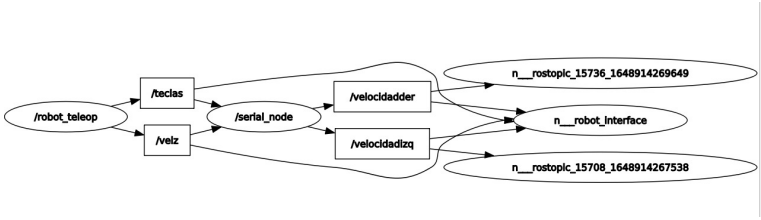

Figure 4: Grafo del funcionamiento del robot

0.1 Movimiento del robot:

Con respecto al funcionamiento, se tiene un computador máster donde se realiza todo el manejo de ROS y la raspberry se comporta como *slave*, recibiendo la información de movimiento. Desde el máster se controla el robot al presionar las teclas 'w' (avanzar), 'a' (rotar en su eje a la derecha), 's' (retroceder), 'd' (rotar en su eje a la izquierda).

La Raspberry está conectada al Arduino Mega, en donde se realiza el control de bajo nivel para el movimiento de los motores. Dependiendo de la tecla, el Arduino decide qué llanta mover:
Al presionar la tecla 'w', ambos motores se mueven hacia adelante a la misma potencia, avanzando linealmente en el terreno.
Al presionar la tecla 's', ambos motores se moverán hacia atrás a la misma potencia, retrocediendo linealmente en el terreno.
Al presionar la tecla 'a', el motor derecho se moverá hacia adelante, mientras que el motor izquierdo se moverá hacia atrás, rotando en su propio eje en sentido anti-horario.
Al presionar la tecla 'd', el motor izquierdo se moverá hacia adelante, mientras que el motor derecho se moverá hacia

atrás, rotando en su propio eje en sentido horario.

Para que la Raspberry sepa la velocidad de cada motor, se está recibiendo las señales de pulso de los encoders. Con estas señales, se puede obtener la rapidez rpm. Después, dependiendo la tecla se establece la velocidad rpm:
Al presionar la tecla 'w', ambos motores mantienen el mismo signo en la rapidez leída.
Al presionar la tecla 's', la rapidez leída de ambos motores tendrán un signo negativo.
Al presionar la tecla 'a', la rapidez leída en el motor derecho se mantendrá positiva, mientras que la leída en el motor izquierdo, tendrá un valor negativo.
Al presionar la tecla 'd', la rapidez leída en el motor izquierdo será negativo, mientras que la leída en el derecho será positivo.

0.2 Gráfico del desplazamiento:

En este punto, se tiene un nodo interfaz donde se realiza la odometría. En este nodo se recibe la velocidad de los motores y se obtiene la posición aplicando la ecuación de odometría. Con esta posición, se va graficando en tiempo real por medio de la biblioteca *Matplotlib*. La interfaz gráfica se realizó con Tkinter para agregar el botón de guardar recorrido y cerrar:

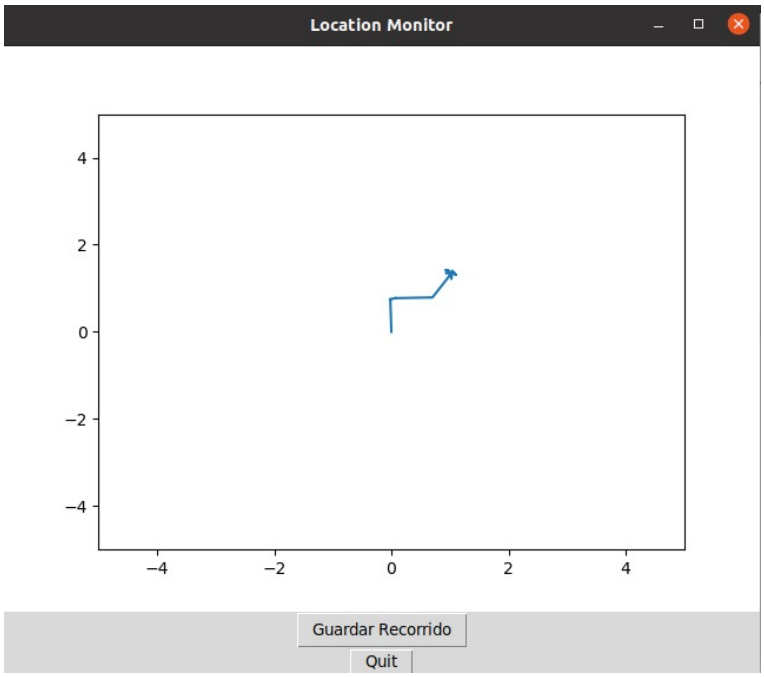

Figure 5: Plano electrónico de QWERTY

Cuando se termina el recorrido deseado, se presiona 'guardar recorrido' y se escribe el nombre del archivo en la terminal para guardarlo con extensión *.txt*.

0.3 Recreación del recorrido:

para este punto se utilizo un servicio ros de nombre "robot_player" que permite recrear un recorrido realizado con anterioridad guardado en un archivo .txt. el archivo se guarda en el nodo robot teleop guardando todas las teclas

que se utilizaron en el recorrido original y se guarda mediante un boton en la interfaz- un vez guardado el archivo, se utilizan dos codigos pyton necesarios para usar el servcio. al correr el servidor se crea un ambiente que puede leer el nombre del archivo desde la consola yen una nueva terminal corremos el cliente python con el nombre del archivo que se desea recrear. al hacer esto, se corre toda la lista de teclas comando y el robot recrea el recorrido guardado. para esto se utilizaron las misma bibliotecas y tópicos del nodo teleop. también se uso la biblioteca para leer archivos txt.