Taller 2 de Robótica

Ivan David Perez Moreno
Universidad de los Andes
id.perez@uniandes.edu.co

Juan José Arévalo
Universidad de los Andes
jj.arevaloh@uniandes.edu.co

Diego Felipe Rodriguez
Universidad de los Andes
df.rodriguezg@uniandes.edu.co

April 2, 2022

Introducción

Descripcion del robot

Planos

Planos mecánicos

En estos planos se tienen 3 vistas del robot desde diferentes perspectivas, desde las cuales se observan las medidas mas importantes a tomar en cuenta, para facilitar la comprensión del plano se enseñarán aquí las diferentes partes del robot representadas.
En la siguiente imagen se puede observar el robot de forma completa, los 4 cilindros azules visibles son baterias de 3.7V, conectadas en serie por pares, las 2 de arriba alimentan a la raspberry a través de usb c ylas 2 de abajo al puente H de forma independiente.

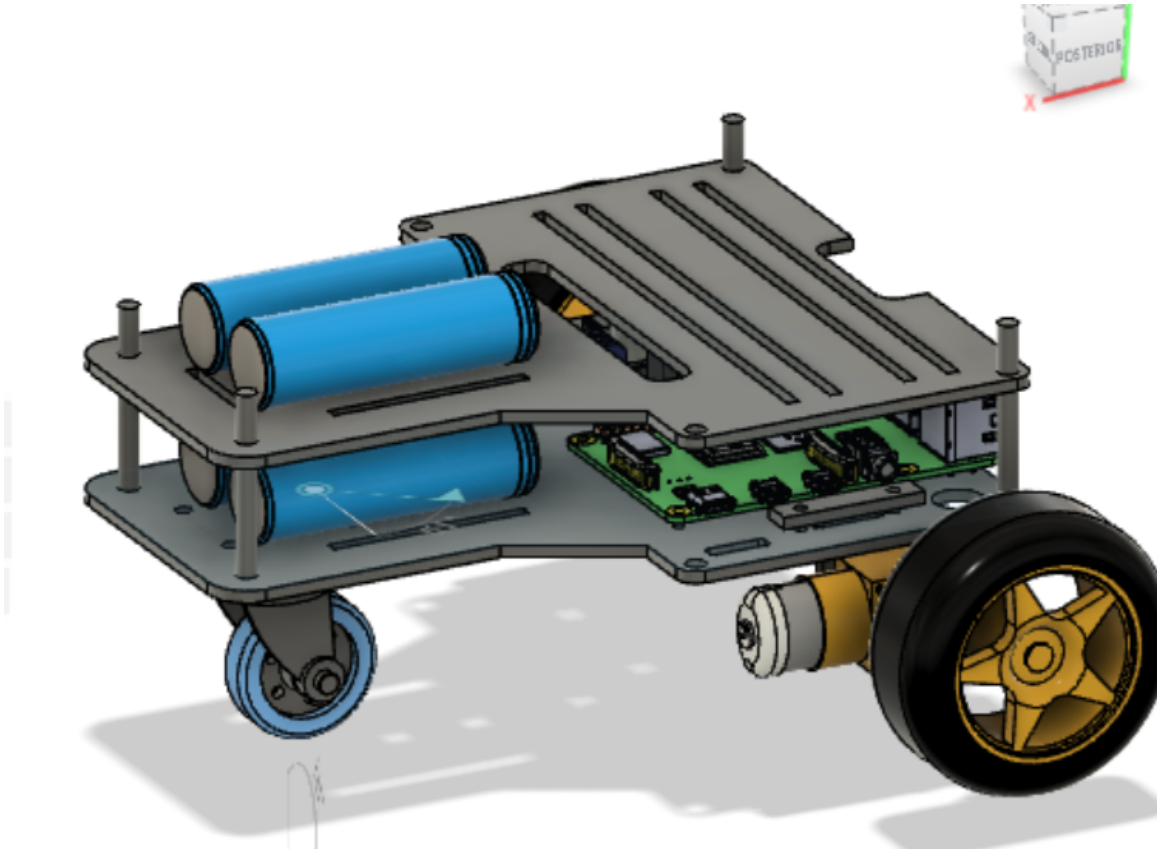

Figure 1: Vista oblicua del robot

También se hizo no visible el segundo piso del robot para enseñar los 2 componentes internos mas importantes, el puente H L298n y la RaspBerry Pi 4B, el primero se observa en la parte superior derecha de la siguiente imagen, en color rojo, la RaspBerry por su parte es el elemento verde de tamaño mas grande que se encuentra en la parte inferior izquierda.

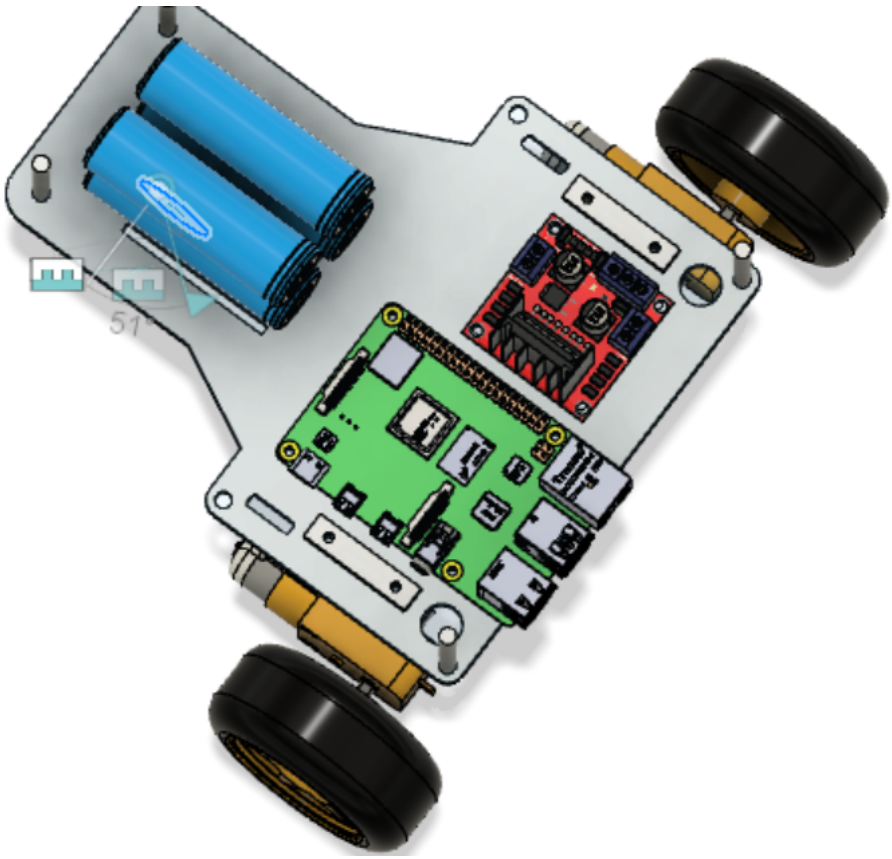

Figure 2: Vista superior del robot eliminando el segundo piso

Los planos mecánicos se pueden encontrar adjuntos a este documento.

Plano electrónico

Para el plano electrónico se tuvieron en cuenta 3 componentes, la RaspberryPi, el puente H y el módulo conversor DC-DC LM2596, en el plano se pueden observar las conexiones entre la raspberry y el puente H, la alimentación de dicho puente como una entrada de 7.5V (la cual varía alrededor de este valor conforme se descarga) y las salidas que van hacia los motores, también se encuentra el circuito del integrado LM2596, este circuito se encarga de alimentar la raspberry pi a través del puerto USB-C que tiene integrada esta, puerto que no se logra apreciar en el diagrama de Eagle. El plano además se encuentra adjunto por si se requiere una mejor apreciación.

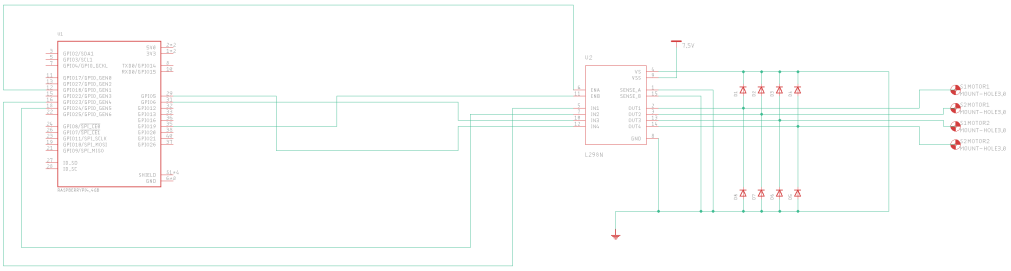

Figure 3: Plano electrónico, conexión Raspberry-PuenteH-Motores

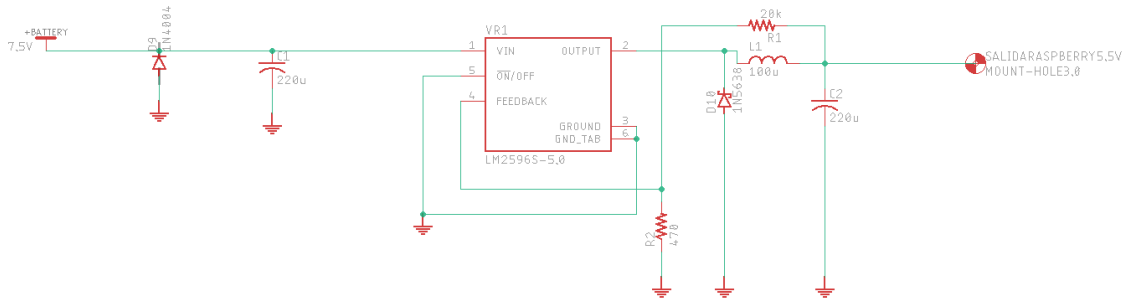

Figure 4: Plano electrónico, modulo alimentador

1 Solucion

Para el desarrollo de la solucion y la creacion de todos los nodos necesarios para el robot se creo el paquete de ROS mi_robot_2 donde se encuentran todos los codigos del proyecto. La idea era conseguir que un robot diferencial de 2 llantas motorizadas que pudiese moverse con 2 grados de libertad, para su movimiento se utilizaron dos motorreductores conectados a un puente H l298N, el cuál se alimentó con 2 baterias conectadas en serie que suministran alrededor de 7.4 voltios, el control del robot se realizó a través de una Rasberry Pi 4B, la cuál tiene su propio par de baterias exactamente igual al anterior, pero esta vez la bateria se conecta a un conversor DC-DC LM2596, el cuál pone en la salida 3A y posee un potenciómetro con el que se estableció la salida de voltaje en 5.5 V.

A continuacion se muestran los grafos de ROS donde se relacionan los distintos nodos utilizados.

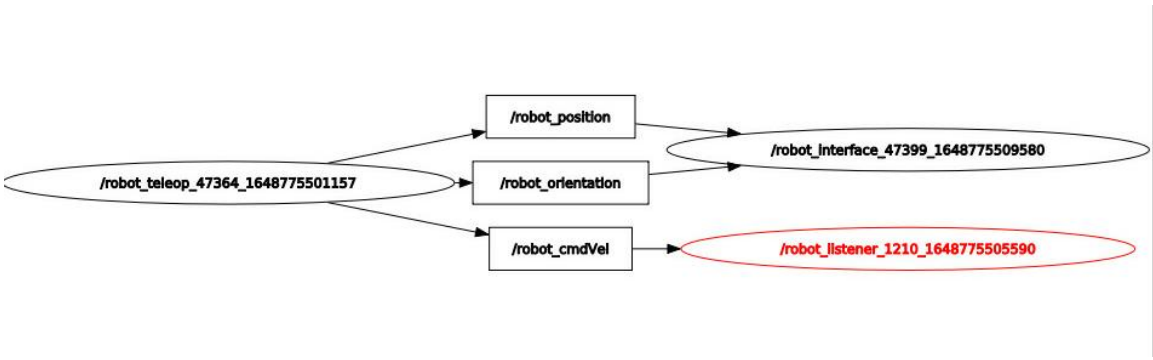

Figure 5: Diagrama $/robot\_interface$, $/robot\_teleop$ y $/robot\_listener$

Como se puede observar el nodo teleop se encarga de enviar la informacion de orientacion, poscicion y velocidad a los topicos correspondientes a partir de la informacion recibida por teclado. el nodo interface optiene informacion de los topicos $/robot\_position$ y $/robot\_orientation$ para graficar la infromacion. Finalmente el nodo listener toma la informacion del topico $/robot\_cmdVel$ para enviar la informacion a traves de la *Raspberry*.

1.1 Nodo Teleop

Se creo un nodo llamado $/robot\_teleop$ (ver codigo adjunto) el cual permite al usuario controlar el robot mediante el uso del teclado. Este nodo se inicializa mediante la libreria *rospy* e incluye un listener que se encarga de detectar las pulsaciones del teclado , dicho listener proviene de la libreria *keyboard*.

Este nodo cuenta con dos funciones , la primer funcion es definicda como $talker()$ funcion que se inicia con el main y se encarga de recibir la informacion por consola de la velocidad lineal y la velocidad angular del robot, de igual manera inicia el listener $keyboard.hook(key\_press)$ el cual llama a la segunda funcion cada vez que se presiona o suelta una tecla y finalmente publica el mensaje de tipo *twist* el cual contiene la informacion de las velocidades a ejecutar en el robot .Esta segunda funcion es definida como $key_press(key)$ cumple el papel de leer la informacion de la telca presionada y a partir de ella construir el mensaje *twist* con las velocidades que utiliza la funcion $talker()$.

Tambien se cuenta con la funcion $getPos(x, y, \theta, vl, vr)$ la cual tiene como parametros la posicion actual del robot en el marco de referencia global y las velocidades lineales de las ruedas , a partir de ello y del modelo cinematico del robot diferencial calcula la posicion del robot despues de un intervalo de tiempo dt, esta infromacion se utiliza para graficar la posicion actual del vehiculo.

Posteriormente se modifico este nodo para que al iniciarce se le pregunte al usuario si desea guardar el recorrido en un archivo de texto. Si la respuesta es afirmativa se preguntar por el nombre que se desea asignar al archivo , creal el *.txt* y cada vez que se modifica el mensaje del nodo se almacena en este archivo con la siguiente estructura **"msg.linear.x , msg.linear.y , msg.linear.z , msg.angular.x , msg.angular.y , msg.angular.z"**.

A continuacion se muestra el diagrama de flujo del funcionamiento del nodo.

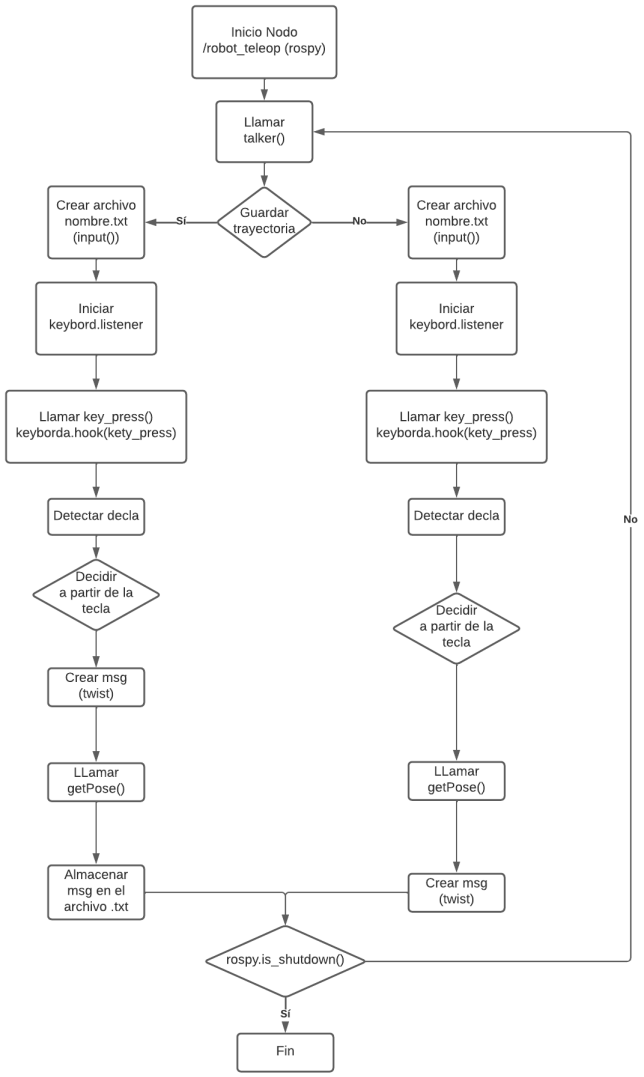

Figure 6: Diagrama */robot_teleop*

Este nodo también publica en el topic Twist la posición del robot, utilizando las velocidades de ambas llantas, lo anterior a partir de funciones que dependen del tiempo.

1.2 Nodo Interface

Se creó el nodo */robot_interface* de tipo suscriber, el cuál recibe la posición del nodo teleop y se encarga de graficarla en tiempo real la posición del robot en un marco global de referencia. Este nodo se suscribe a un topico de tipo *Twist* para las coordenadas rectangulares y a otro del tipo *Float32* para los angulos de la orientación, datos que son escritos por el nodo teleop ,las coordenadas rectangulares entran a la función callback_pose la cual redondea el valor de la posición x y y, la función orientation realiza lo mismo con los valores de angulos. Posteriormente la función animate se encarga de graficar la posición como tal usando plot, la cual grafica la posición actual del robot y

utiliza un marcador para poder identificar la dirección de este, se utiliza Tkinter para poder guardar la imagen, y las graficas de las posiciones individuales se agregan al canvas de Tkinter para crear la animación. A continuación un diagrama de flujo que explica el funcionamiento del nodo.

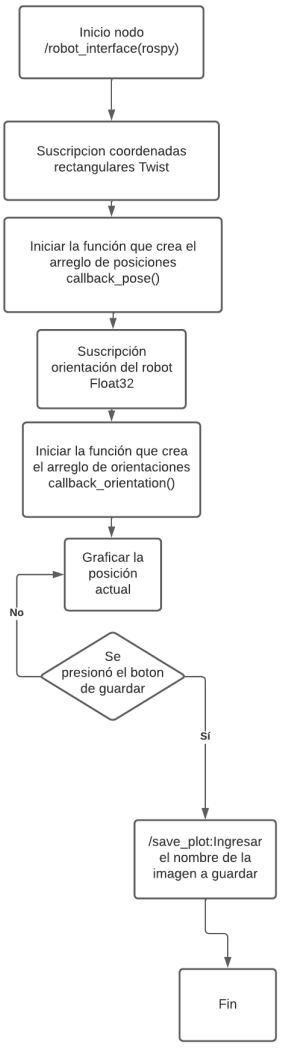

Figure 7: Diagrama $/robot\_interface$

Tambien se muestra un ejemplo de las gráficas que permite realizar este nodo:

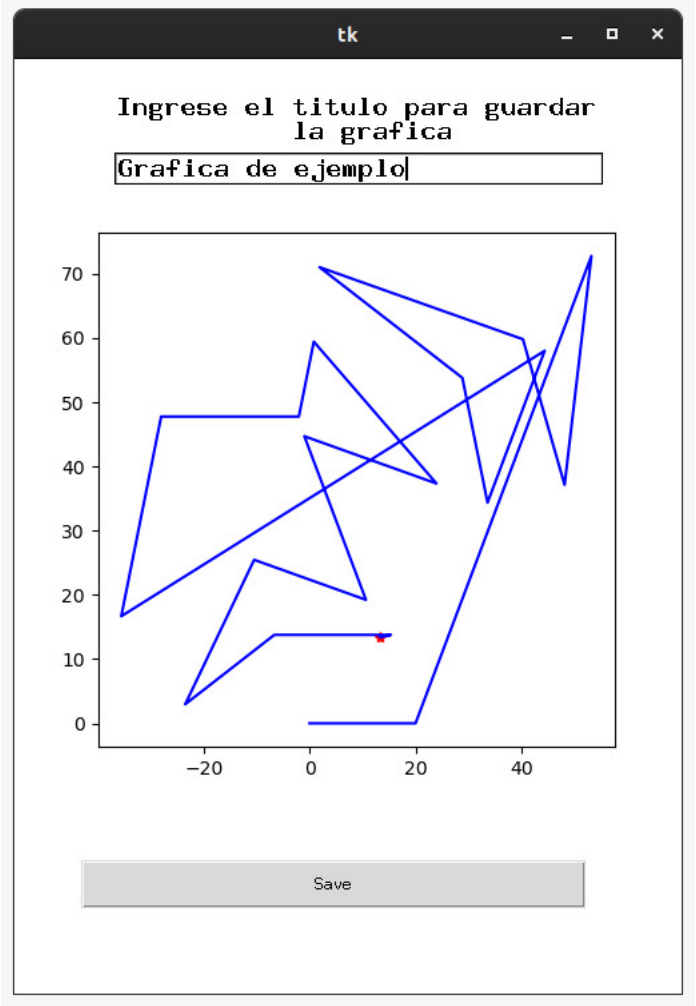

Figure 8: Ejemplo gráfica

En este caso se observa que los giros estan conformados por picos, esto dado que el robot no gira mientras avanza, gira sobre su propio eje para cambiar de dirección.

1.3 Nodo Player

Se creo el nodo */robot_player* el cual toma una trayectoria anteriormente generada por el nodo */robot_teleop* y la replica. Para cumplir con esta tarea como es habitual se inicializa el nodo mediante *rospy* y utiliza la libreria *os* para buscar y cargar los archivos.

Este nodo cuenta con la funcion main donde se inicializa todo y se llaman a las demas funciones , en esta funcion se muestran los arcihvos con las trajectorias disponibles para su seleccion, una vez se escoje el archivo a utilizar llama a la funcion $move(str)$ en esta funcion se leen las trayectorias y se escriben los mensajes linea a linea con las velocidades almacenadas.

Tambien se cuenta con la funcion $getPos(x, y, \theta, vl, vr)$ la cual tiene como parametros la posicion actual del robot en el marco de referencia global y las velocidades lineales de las ruedas , a partir

de ello y del modelo cinematico del robot diferencial calcula la posicion del robot despues de un intervalo de tiempo dt, esta infromacion se utiliza para graficar la posicion actual del vehiculo.

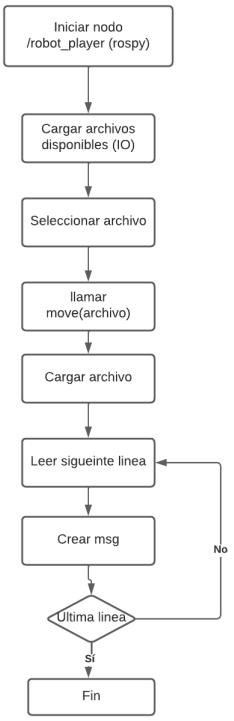

Figure 9: Diagrama $/robot\_player$

1.4 Nodo Listener

Adicionalmente se creo un nodo llamado $robot\_listener$ el cual se encarga tomar la informacion del nodo $robot\_teleop$ y traducirla al robo fisico mediante la $Raspberry$ utilizando la libreria $RPi.GPIO$ de igual manera calcula la posicion actual del robot a partir de su geometria y velocidad lineal en cada rueda.

El nodo cuenta con el main , donde se inicializa la funcion $listener()$ en esta funcion se inicializa el nodo mediante $rospy$, adicionalmente actualiza la informacion de posicion y orientacion del robot. La funcion principal de codigo es la funcion $callback_move(data)$ en esta funcion se recibe la informacion de las velocidades lineales y angulares en las ruedas del robot, una vez se tiene esta infromacion se calcula la señal PWM que debe enviarse al puente H mediante GPIO. Finalmente se verifica la direccion del moviento y se llaman las funciones $Giro\_x\_x\_A()$ y $Giro_x_x_B()$las cuales definen la direcion de giro de cada motor (motor A, motor B) y envian la señal fisica a cada uno de ellos.

$$x(t) = x_o + \frac{1}{2}\int_0^t [V_r(t) + V_l(t)]\cos\big(\theta(t)\big)\,dt$$

$$y(t) = y_o + \frac{1}{2}\int_0^t [V_r(t) + V_l(t)]\sin\big(\theta(t)\big)\,dt$$

$$\theta(t) = \theta_o + \frac{1}{l}\int_0^t [V_r(t) - V_l(t)]dt$$

Figure 10: Modelo cinematica directa

A continuacion se presenta el diagrama del flujo del nodo listener.

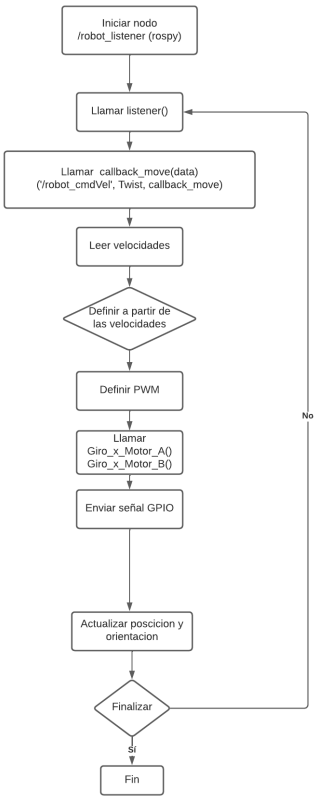

Figure 11: Diagrama */robot_listener*