Taller 3 - Robótica

Rubiano Enciso, Brayan Felipe - *bf.rubiano@uniandes.edu.co*
Arias Castrillon, Juan Camilo - *jc.ariasc@uniandes.edu.co*
Vargas Guerrero, Angela Maria - *am.vargasg@uniandes.edu.co*
Escobar Jaramillo, Edgard Mateo - *em.escobar@uniandes.edu.co*
Universidad de los Andes
Departamento de Ingeniería Eléctrica y Electrónica

10 de Mayo 2022

1. Introducción

En este taller numero 3 se busca crear un robot manipulador capaz de agarrar objetos e identificarlos. Para esto se utilizará una tarjeta raspberry pi 4 la cual será el medio de comunicación con el robot por medio del enguaje *python*, se realizará la comunicación via ROS. Este manipulador debe encontrarse en la parte posterior del robot, sin embargo no se tendrá en cuenta un rango de dimensiones fisicas exáctas. Se pretende explicar el funcionamiento del manipulador y el funcionamiento del reconocimiento de objetos por medio de diagramas. A continuación se muestran los objetivos:

2. Objetivos

1. Crear un nodo en ROS, llamado /robot_manipulator_teleop, que permita a un usuario controlar por teclado su robot. Es decir, que las velocidades de cada una de las junturas que tiene su robot se puedan controlar de forma directa con las teclas del computador.

2. Crear un nodo de ROS, llamado /robot_manipulator_interface que permita visualizar la posición en tiempo real del end-effector de su robot, a través de una interfaz gráfica. Es decir, el programa deberá ir mostrando en la pantalla una gráfica donde se representa la posición del end-effector del robot en el marco global de referencia en tiempo real y muestre el camino recorrido por el mismo desde donde inició. Para esto puede usar técnicas de visión por computador. La interfaz debe contar con el espacio para asignarle un nombre a la gráfica y poderlo guardar en el directorio deseado, al finalizar el recorrido.

3. Crear un nodo en ROS, llamado /robot_manipulator_planner, que permita a un usuario llevar el end-effector del robot a una posición destino deseada. Es decir, dada una posición del end-effector deseada dentro de su volumen de trabajo, el robot debe realizar todo el cálculo de cinemática inversa o de planeación de trayectorias para alcanzar esta posición. Estas posiciones serán definidas por los instructores el día de la sustentación basado en el volumen de trabajo que cada grupo le asigne a su robot manipulador.

4. En una estructura de madera (Similar a la que se encontrará en el proyecto final) de dimensiones 10 cm de alto, 25 cm de ancho y 5 cm de profundidad se encontrarán varios ping pong de diversos colores. El robot debe estar en capacidad de tomar el ping pong del color especificado a través del tópico /robot_manipulator_ping_pong. Este valor de color será publicado de la forma como mejor le convenga al grupo. Es decir, el grupo decidirá si se pública a través de terminal desde un computador externo, o conectado a la raspberry de forma remota

3. Implementación

Para el taller se creará un paquete de ROS llamado llamado mi_robot_manipulador_14 el cual incluye los archivos utilizados en el taller. Para que el robot funcione de forma idonea se sugiere crear 3 tópicos los cuales se explican a continuación:

/robot_manipulator_position = Posición actual del end-effector del robot en el marco inercial o global de referencia

1. • msg.x = posición en x del end-effector del robot en marco inercial

2. • msg.y = posición en y del end-effector del robot en marco inercial

3. • msg.z = posición en z del end-effector del robot en marco inercial

/robot_manipulator_target = posición deseada del end-effector del robot en el marco de referencia de la camara

1. • msg.x = posición en x deseada del end-effector del robot en marco de referencia de la camara

2. • msg.y = posición en y deseada del end-effector del robot en marco de referencia de la camara

/robot_manipulator_ping_pong = : color del ping pong que se pedirá recoger.

1. • msg.data = b - Ping pong de color azul

2. • msg.data = r - Ping pong de color rojo
3. • msg.data = y - Ping pong de color amarillo

3.1. Listado de materiales

Se utilizaron los siguiented elementos para diseñar el manipulador y su control:

- Raspberry pi 4
- 2 baterías BRC18650 9800mAh 4.2V Li-on
- Shield de Arduino
- Power Bank para alimentación de la raspberry
- 4 servomotores (Negro)
- Arduino UNO
- 32 tuercas y 32 tornillos
- 37 piezas en acrilico correspondientes al manipulador
- Adaptador de la garra

3.2. Planos mecánicos del robot

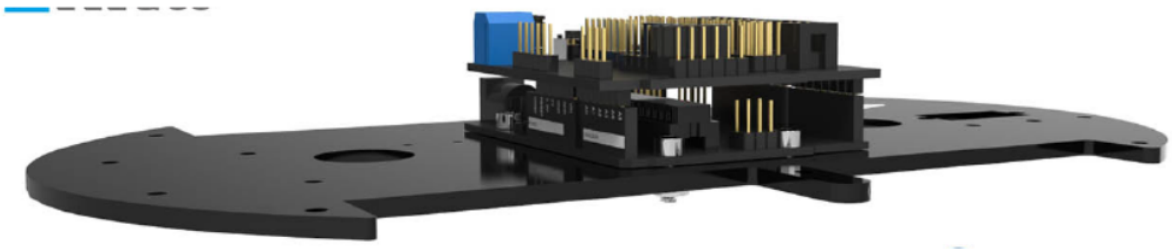

Figura 1: Plano

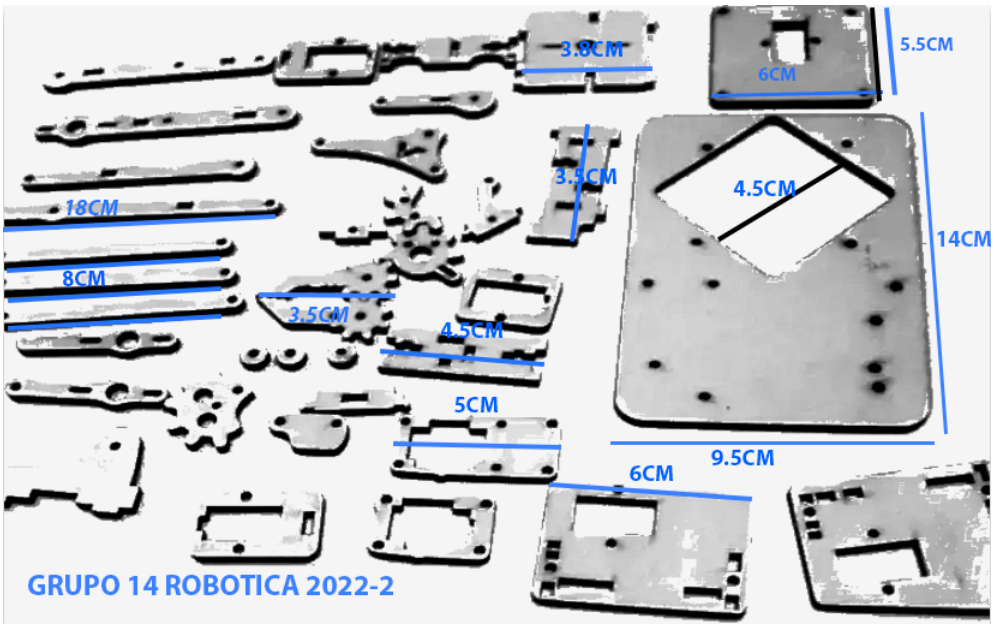

Figura 2: Listado de piezas manipulador

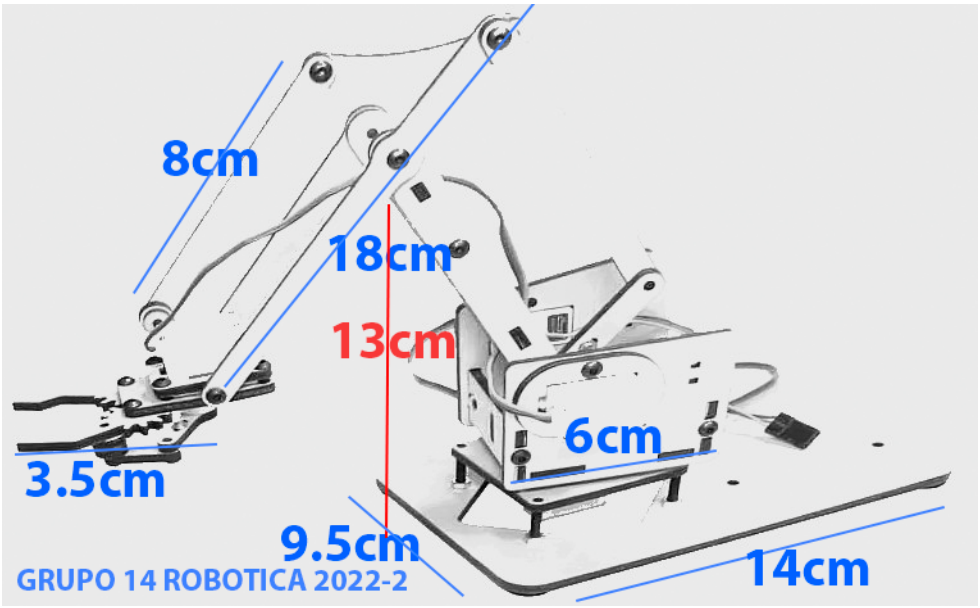

Figura 3: Manipulador con sus dimensiones

El robot cuenta con un total de 4 servomotores, esto permite una interacción total con el mismo, debido a que se puede rotar en diferentes ejes de forma continua y esto permite mejorar el movimiento del mismo. Inicialmente es capaz de moverse 90° en el eje XY. Del mismo modo, los 2 motores que se ubican en los planos YZ permiten contraer el brazo lo mayor posible, y también elongarlo hasta el punto más lejano. Finalmente

el motor 4 correspondiente a la garra permite abrir y cerrar el mecanísmo. De este modo es muy sencillo mover el brazo mecánico a una posición determinada sin algún problema.

3.3. Plano eléctrico

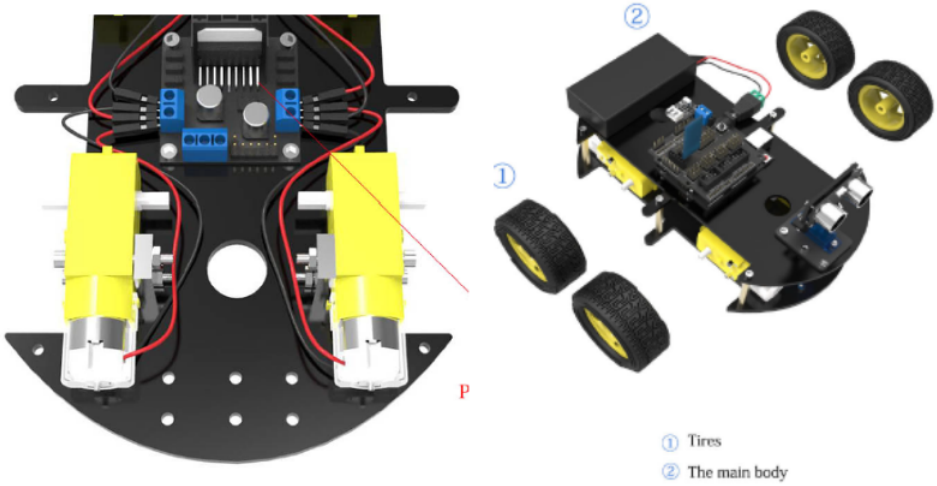

Figura 4: plano

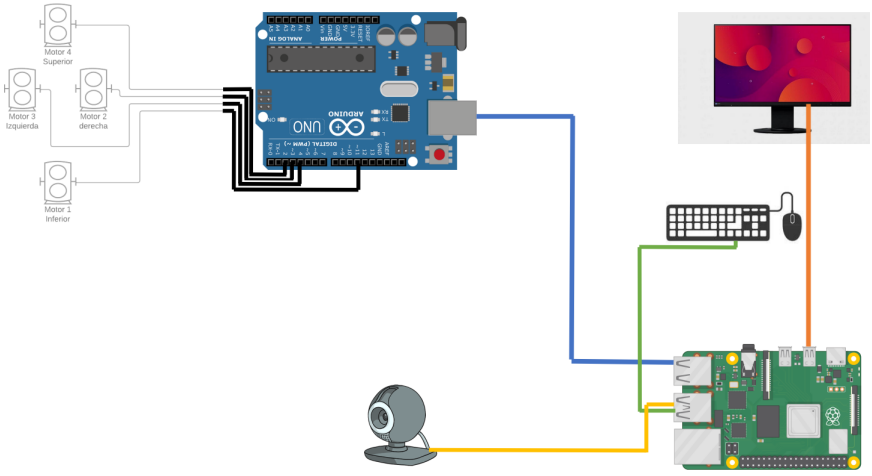

Figura 5: Plano eléctrico

En el plano eléctrico se muestra la unión de cada uno de los 4 servomotores conectados al arduino UNO, y como este mismo se conecta a la raspberry para generar la

conexión entre los servomotores y la camara. También se muestra la conexión entre los perifericos y la raspberry.

3.4. Unión parte mecánica y eléctrica

En la siguiente figura se logra observar la union entre los planos mecanicos y elèctricos. Donde se observa el uso de la camara, y los servomotes junto con el brazo mecanico, se logrò obtener una muy buena sinergia entre estos componentes, pues, el funcionamiento se diò en òptimas condiciones.

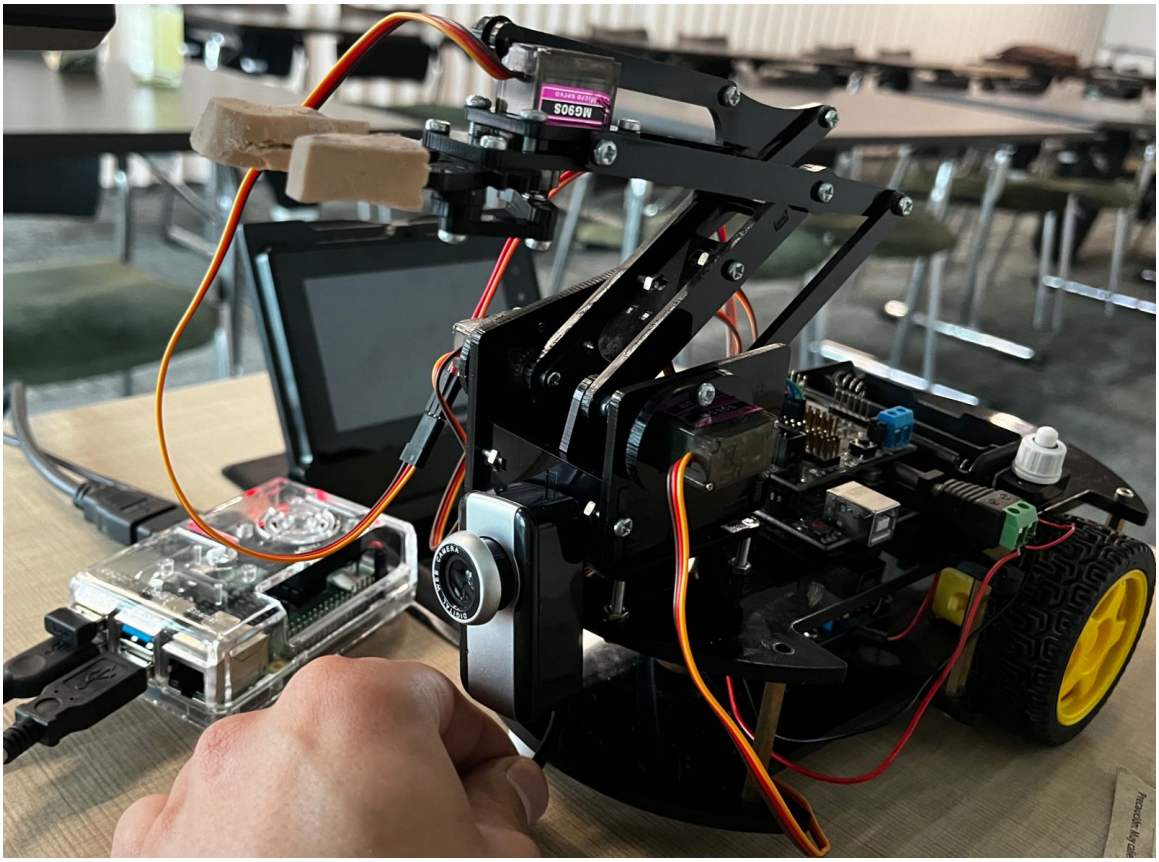

Figura 6: Robot ensamblado

Punto 1

Para el primer punto se crea el nodo llamado /robot_manipulator_teleop se crea un codigo el cual escucha los comandos que son mandados al arduino, para esto inicialmente se inicializan los motores, dandoles valores de high y low, de modo que se pueda manejar de acuerdo a nuestras solicitudes. Posteriormente se configura para que al oprimir una tecla se mueva uno de estos valores inicialmente creados. A continuacion se muesrta el diagrama:

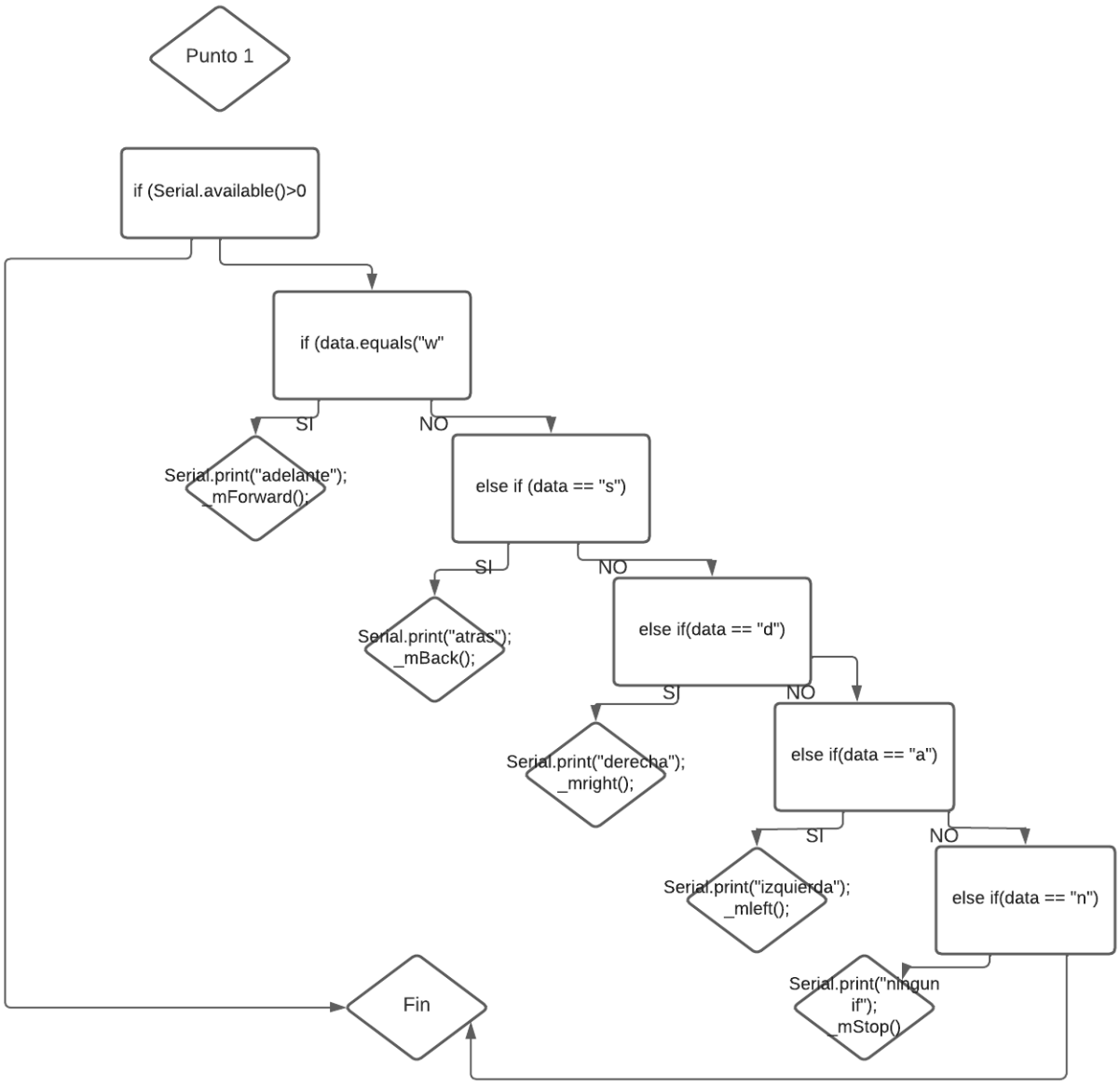

Figura 7: Punto 2

Punto 2

Para la realización del punto 2 se pide crar un nodo llamado /robot_manipulator_interface para graficar junto con una interfaz el movimiento del manipulador. Se muestra a continuacion la comunicaciòn entre nodos.

La gráfica se realiza en tres ejes: x. y, z. Para dar cada punto se toman los angulos de los servos de rotación y de profundidad (base y hombro). A partir de estos es posible determinar la posición de la garra. De emplearse el cuarto servomotor, tambien se

debería tomar su respectivo ángulo, el cual varia la altura.

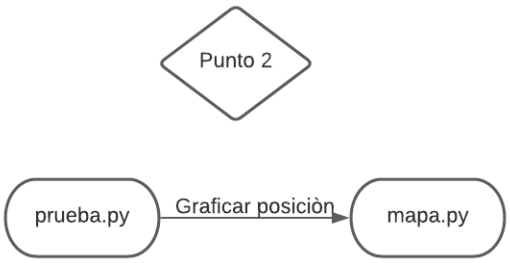

Figura 8: Punto 1

Punto 3

En este literal se crea un nodo que toma el iput del usuario desde la consola y lo publica en el topico 'robot_manipulator/goal'. El mensaje publicado por este es leido por el nodo planner en donde se debe hacer la planeacion de la ruta.

Figura 9: RQT Graph Punto 3

Punto 4

Finalmente en el punto 4 se crea un nodo que define el color de la bolita a recoger, este toma las teclas r (rojo), b (azul), y (amarillo), lo publica en el topico /robot_manipulator/color, este lo lee el nodo visual que es el encargado de hacer el analisis de la imagen de la camara. Dado el color se le pasa la mascara correspondiente para dibujar los contornos de los elementos en la imagen que tienen el color correspondiente, es decir, de la bolita de ping pong, y a partir de los contornos y de los momentos se obtienen las coordenadas en x,y del centro del objeto identificado. Estas coordenadas se publican en el topico /robot_manipulator/goal, a este topico se suscribe el nodo planner que determina la trayectoria que debe seguir el manipulador para poder agarrar la pelota de ping pong.

Figura 10: RQT Graph Punto 4

A continuacion se puede observar como se ven los contornos obtenidos en una imagen con elementos de formas distintas de los colores determinados.

Figura 11: Identificar objeto amarillo

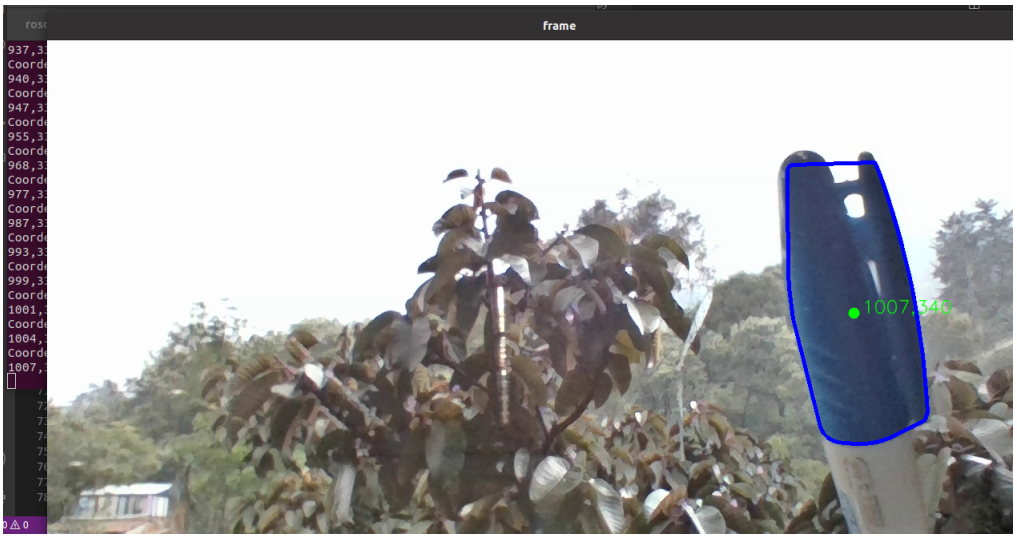

Figura 12: Identificar objeto azul

Figura 13: Identificar objeto rojo

Nota: En el archivo visualP se muestra claramente la identificacion de colores y coordenada del centro del objeto, este archivo es de solo python. En el archivo visualDetection.py se encuentra la misma informacion pero ya suscribiendose y publicando a los topicos correspondientes, es decir, detecta solo el color indicado.