**INSTITUTO TECNÓLOGICO DE LA PAZ**

**DIVISIÓN DE ESTUDIOS PROFESIONALES**

**Reporte semanal de residencias Semana 04**

**“Reconocimiento y seguimiento de objetos en entornos controlados”**

**CARRERA**

**Ingeniería en Sistemas Computacionales**

**PRESENTA**

**17310793 – Eloy Antonio Clemente Rosas**

La Paz, Baja California Sur, México, 26 de Abril del 2021

# Introducción

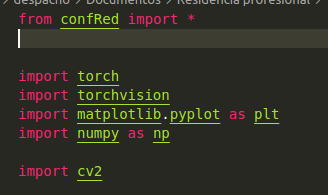
El presente documento contiene las actividades realizadas la semana 11 y 12 de la residencia profesional. Estas dos semanas fueron dedicadas a un único objetivo. Mapeo de los objetos identificados. Es decir, calcular su posición relativa y posición en el mundo real.

Para esto fue necesario tener listos los mejores modelos, el modelo que mostró un mejor reconocimiento fue el modelo para imágenes pequeñas que se encuentra en el siguiente link: <https://drive.google.com/drive/folders/1tI2NadA33NbBEO_JdqL_sQEeVVVnqWRa?usp=sharing>

# Desarrollo de actividades

Para poder usar el modelo para clasificar primeramente es necesario, tener una instancia de la clase U-Net que almacena las configuraciones de los pesos de la red. El código se encuentra en el siguiente repositorio: <https://github.com/EloyFtw/Residencia-profesional/blob/main/Programas/Mapeo/confRed.py>

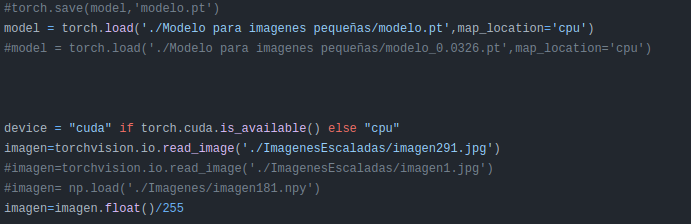
Ahora que tenemos una instancia de la configuración de la red, se puede cargar con el siguiente código donde también vienen las librerías usadas:

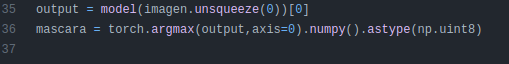


El código usado se encuentra en el siguiente repositorio:

https://github.com/EloyFtw/Residencia-profesional/blob/main/Programas/Mapeo/mape.py

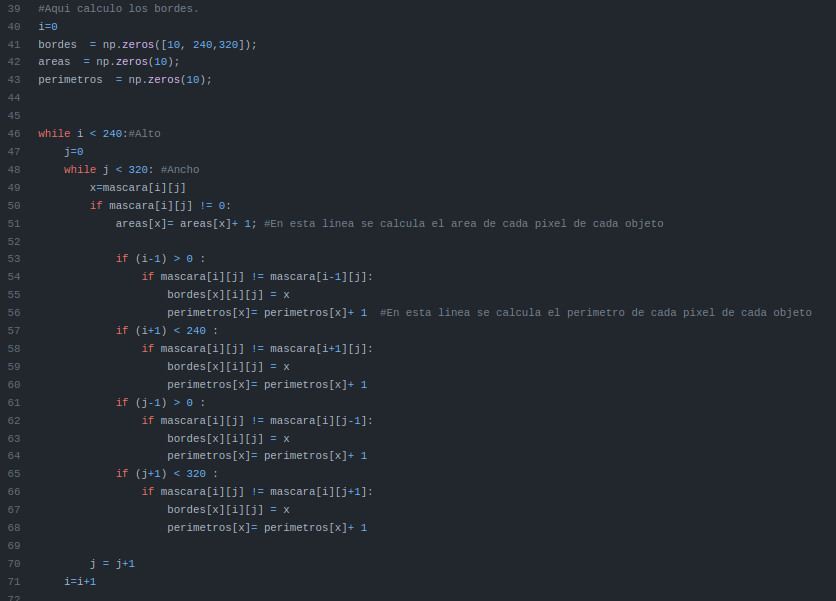
A continuación se carga el modelo y se indica la imagen a la cual se le va a realizar la clasificación.

Con la siguiente sección de código se realiza la clasificación y se guarda como arreglo Numpy.

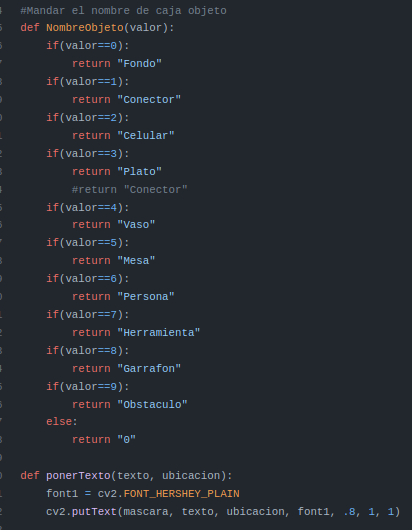


Ahora que esta guardada la mascara donde se hizo la prediccion pixel a pixel es momento de calcular el area, el perimetro, el centro de cada objeto y finalmente la distancia al objeto.

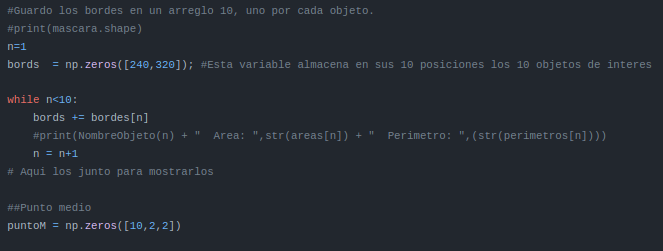
Con el siguiente codigo se recorre la mascara pixel a pixel. Cuando encuentra un pixel diferente al siguiente lo guarda como un borde, cuando encuentra un pixel con uno de los indices del 1-10 lo guarda dependiendo el indice al que pertenece, y suma la cantidad de pixeles de cada tipo para obtener el area y perimetro en pixeles.



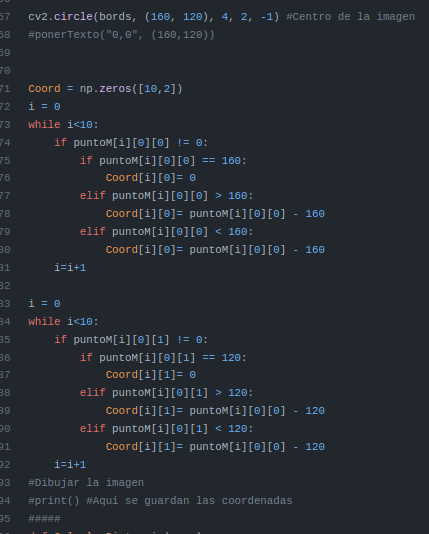
Los siguientes metodos se usaron para proporcionar de un nombre a el valor del pixel que se mande como parametro.



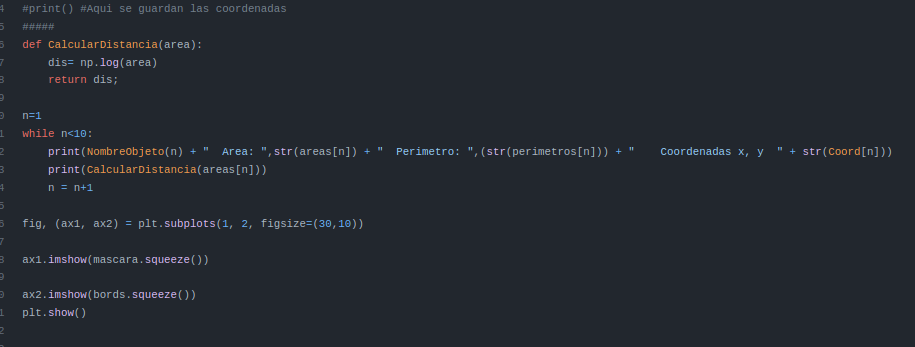
Aqui se guardan los bordes en una varaible para posteriormente mostrarlos y trabajar con ellos



Finalmente se calculan los puntos medios y las coordenadas en el eje X eje Y



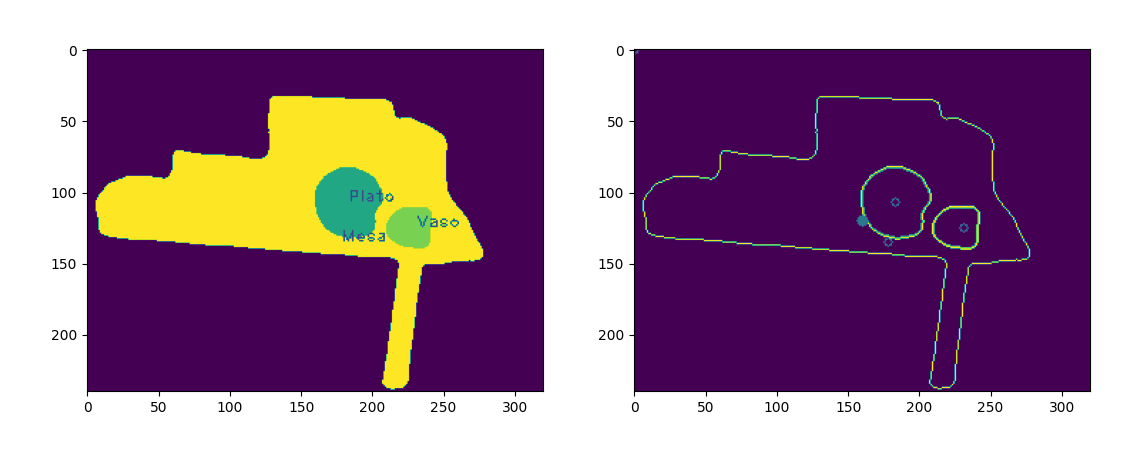
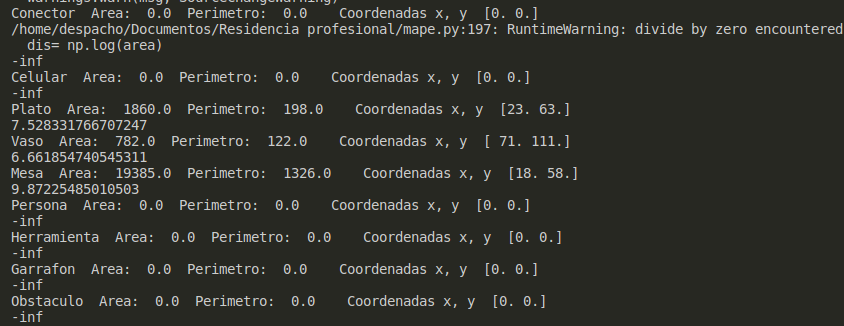
En este caso calcule la distancia en pixeles a partir de una funcion logaritmitca que se adecuaba un poco a los resultados que esperaba obtener y finalmente se muestran los resultados.

****

**Resultados**

La primera imagen es la mascara obtenida de la clasificacion.

La segunda es el procesado donde se obtuvo los bordes de cada objeto, su centro, ademas se le asigno el nombre al que pertenece cada objeto.

Como puede observarse los resultados son muy buenos, solamente la distancia no esta de manera correcta.

**Conclusiones**

Los resultados funcionan el problema es precisamente que toma demasiado tiempo de ejecucion realizar todos estos calculos de esta forma, esto se soluciono hoy con unas optimizaciones aplicadas y el metodo para calcular la distancia cambio aplicando otro metodo que se mostrara en el reporte de esta semana.

Pero es lo que se espera del mapeo obtener las coordenadas “x”, para centrar, “y” para la altura y “z” para la distancia de la camara al objeto.

Los archivos usados se encuentran en el repositorio siguiente:

**https://github.com/EloyFtw/Residencia-profesional/tree/main/Programas**