Trabajo Final
Automación Industrial

Gianfranco Muscariello
Alejo Figueroa
Facundo Molina
Xi Lin

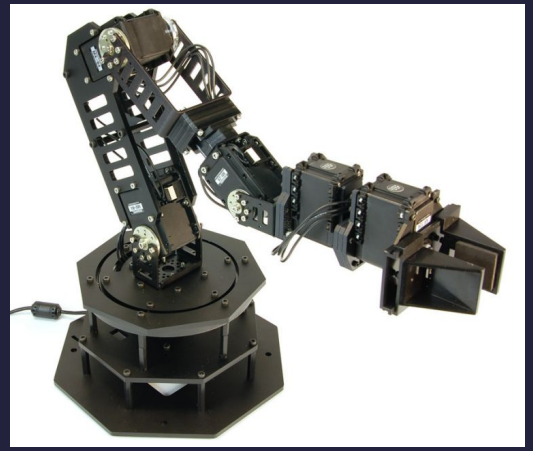
01

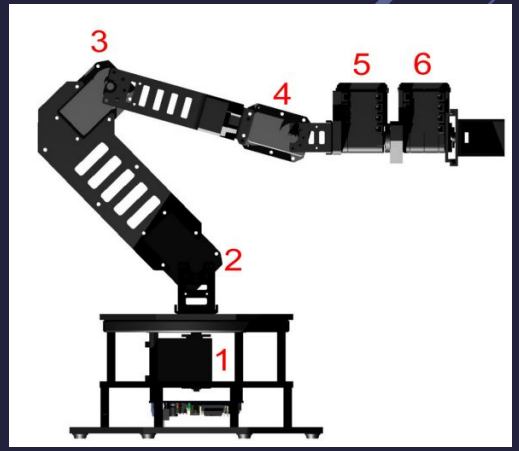
Caracterización

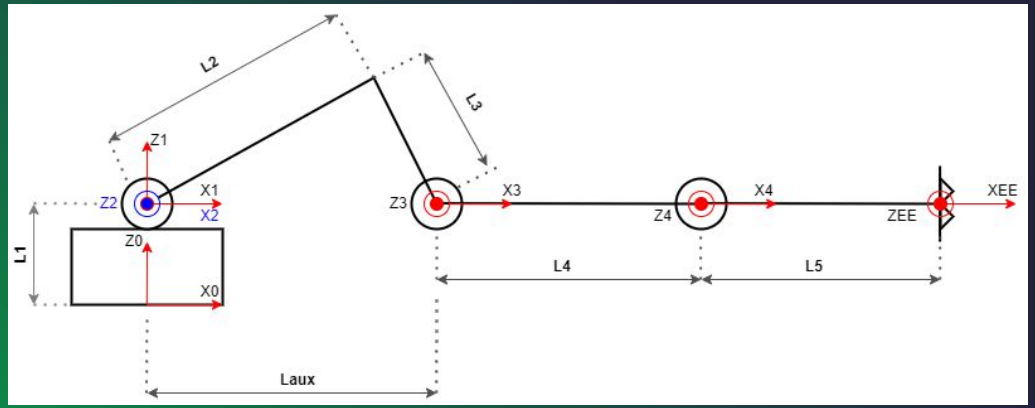
Análisis del modelo

Parte 1: Caracterización

- Brazo robótico WidowX MK-II
- 4 articulaciones + 1 gripper/EE.
- Medidas:
  L1 = 130mm,
  L2 = 144mm,
  L3 = 53mm,
  L4 = 144mm,
  L5 = 144mm



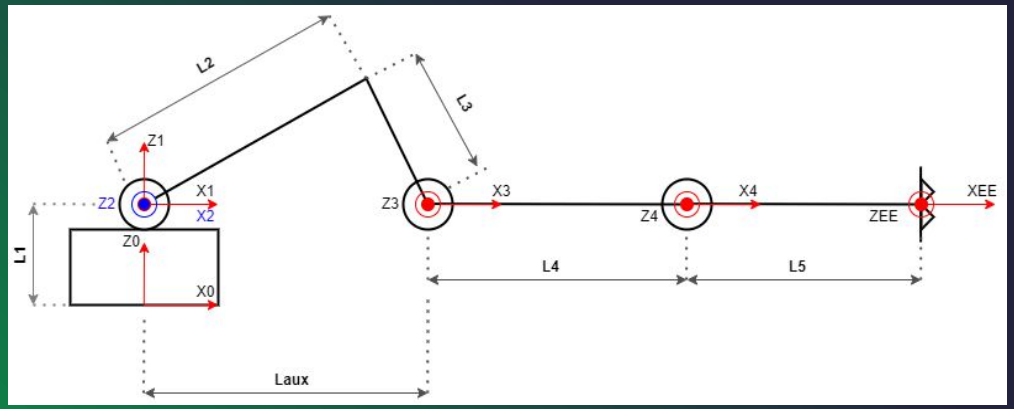




i	α_{i-1}	a_{i-1}	d_i	θ_i
1	0	0	L_1	θ_1
2	$\frac{\pi}{2}$	0	0	θ_2
3	0	L_{aux}	0	θ_3
4	0	L_4	0	θ_4
EE	0	L_5	0	0

Parte 1: Caracterización

A partir de sus parámetros DH, se obtienen las transformadas homogéneas y la cinemática inversa del EE.



$$T_{EE}^{0} = T_1^0 T_2^1 T_3^2 T_4^3 T_{EE}^4$$

$$T_1^0 = \begin{bmatrix} c1 & -s1 & 0 & 0 \\ s1 & c1 & 0 & 0 \\ 0 & 0 & 1 & L1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_2^1 = \begin{bmatrix} c2 & -s2 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ s2 & c2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_3^2 = \begin{bmatrix} c3 & -s3 & 0 & L_{aux} \\ s3 & c3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_4^3 = \begin{bmatrix} c4 & -s4 & 0 & L4 \\ s4 & c4 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_{EE}^4 = \begin{bmatrix} 1 & 0 & 0 & L_{EE} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

T =

$$\begin{pmatrix} \sigma_3\cos(t_1) & -\sigma_1\cos(t_1) & \sin(t_1) & \cos(t_1)\,\sigma_2 \\ \sigma_3\sin(t_1) & -\sigma_1\sin(t_1) & -\cos(t_1) & \sin(t_1)\,\sigma_2 \\ \sigma_1 & \sigma_3 & 0 & L_1 + L_4\sin(t_2+t_3) + \text{Laux}\sin(t_2) + \text{Lee}\,\sigma_1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

where

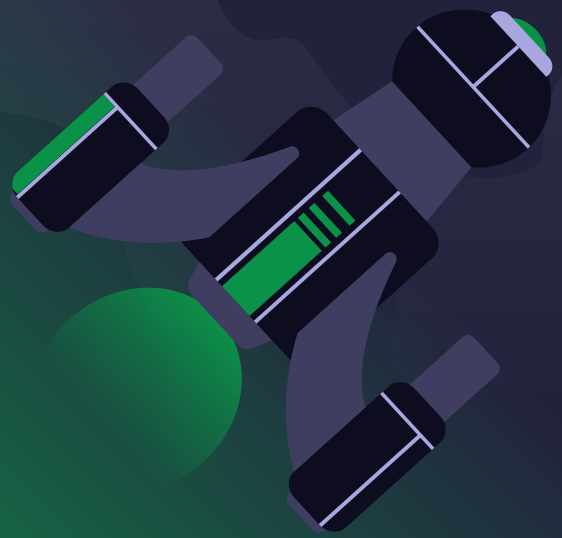
$$\sigma_1 = \sin(t_2+t_3+t_4)$$

$$\sigma_2 = L_4\cos(t_2+t_3) + \text{Laux}\cos(t_2) + \text{Lee}\,\sigma_3$$

$$\sigma_3 = \cos(t_2+t_3+t_4)$$

TTee =

$$\begin{pmatrix} \cos(t_1)\ (L_4\cos(t_2+t_3) + \text{Laux}\cos(t_2) + \text{Lee}\cos(t_2+t_3+t_4)) \\ \sin(t_1)\ (L_4\cos(t_2+t_3) + \text{Laux}\cos(t_2) + \text{Lee}\cos(t_2+t_3+t_4)) \\ L_1 + L_4\sin(t_2+t_3) + \text{Laux}\sin(t_2) + \text{Lee}\sin(t_2+t_3+t_4) \end{pmatrix}$$



02

Control

Movimiento del robot y
espacio de trabajo

Parte 2: Robot

Creación de objetos que se van a utilizar para la simulación del movimiento del brazo.

Table
X0, Y0, Z0 width, length color
drawTable()

Widow
L1, L2, L3, L4, L5, Lee initPos
drawRobot() drawWorkspace(thlim) drawLine(initPos,endPos,step)

Parte 2: *class* Widow

Creación del objeto

```
Widow(L1,L2,L3,L4,L5,Lee,initPos);
```

Muestra en la figura el robot con el brazo en la posición inicial

```
drawRobot(obj);
```

Muestra una nube de puntos del espacio alcanzable por el robot según las limitaciones de los ángulos provistos

```
drawWorkspace(thlim);
```

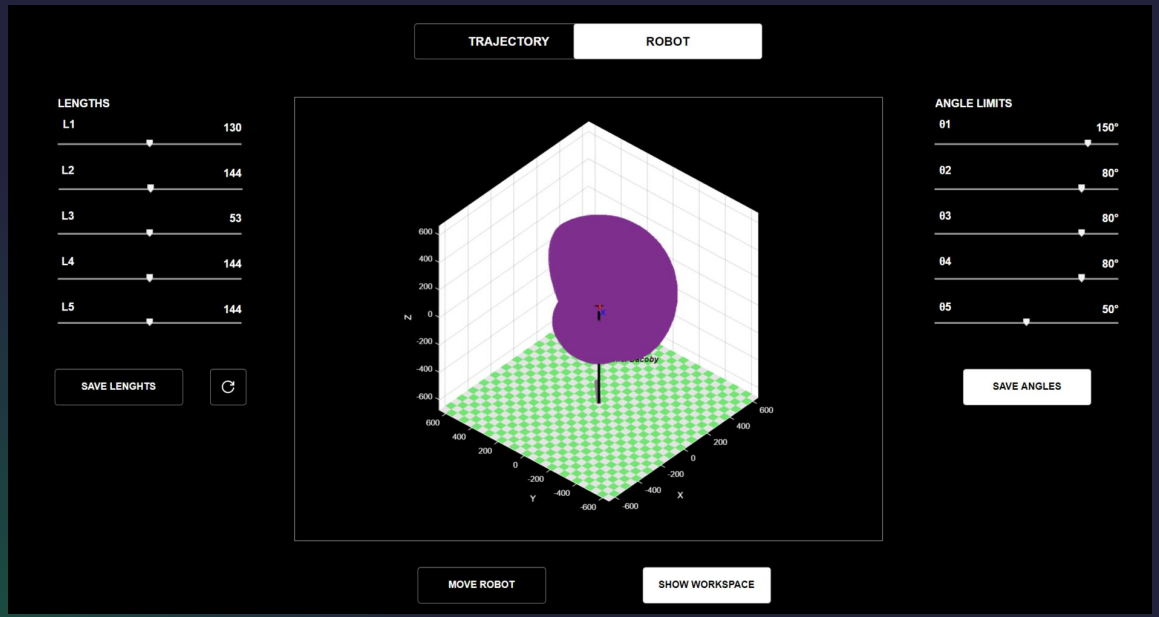
Crea una secuencia de puntos equiespaciados para indicar el movimiento del brazo. Hace el movimiento del brazo y muestra en la figura la recta final dibujada.

```
drawLine(obj,initPos,endPos,step);
```

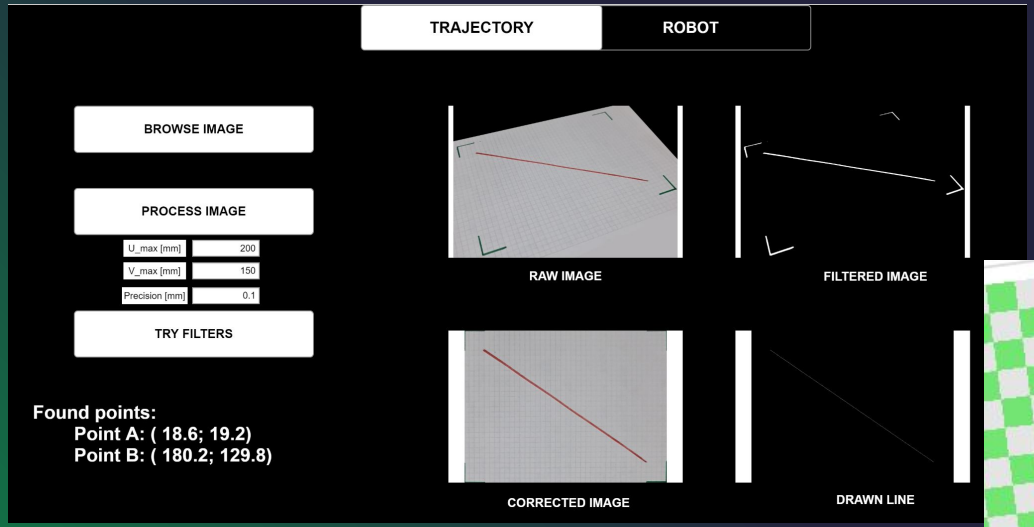
Parte 2: *class* Widow

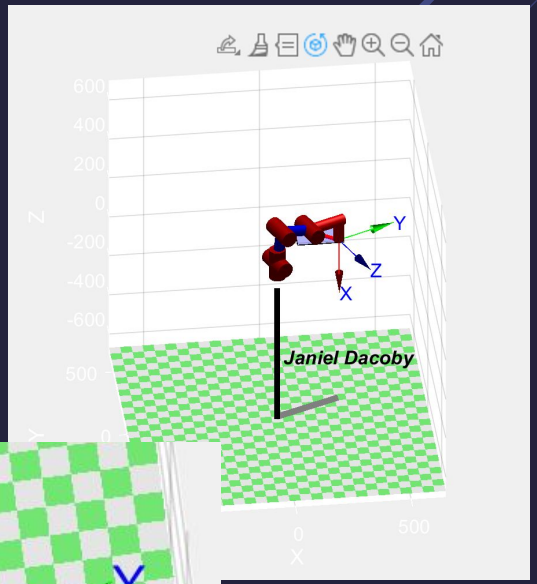
Muestra una nube de puntos del espacio alcanzable por el robot según las limitaciones de los ángulos provistos

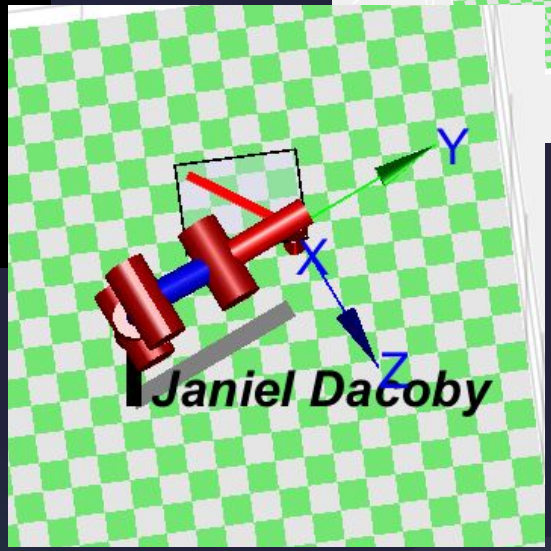
drawWorkspace(thlim);

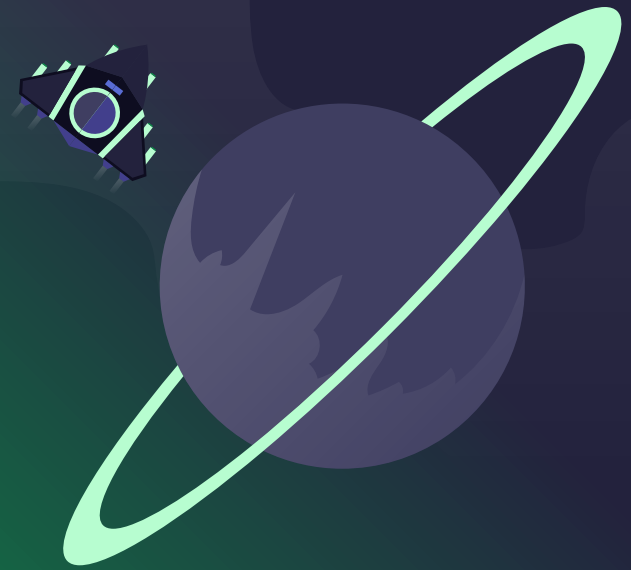


Parte 2: drawLine()









03

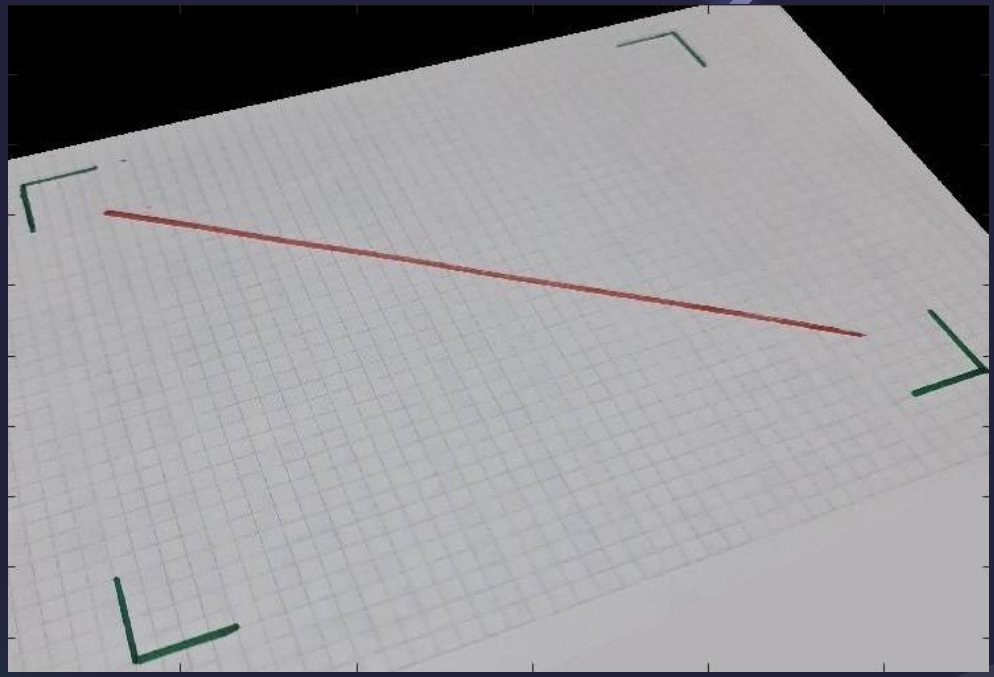
Vision

Procesamiento de imagen
y trayectoria

Parte 3: Visión

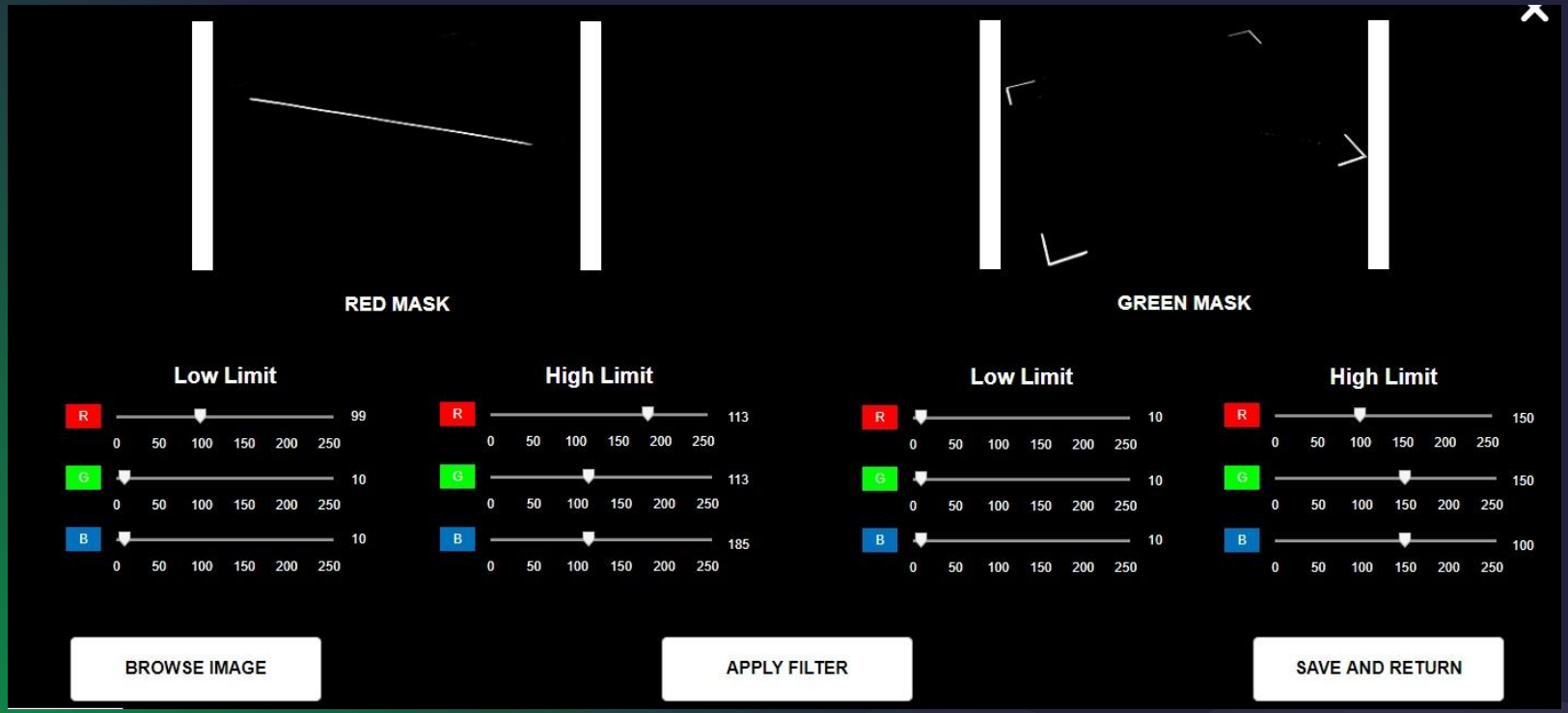
La imagen original se saca con un ángulo muy agresivo, para probar los límites del programa.

Se usaron colores similares a la imagen de prueba, y las dimensiones son iguales.



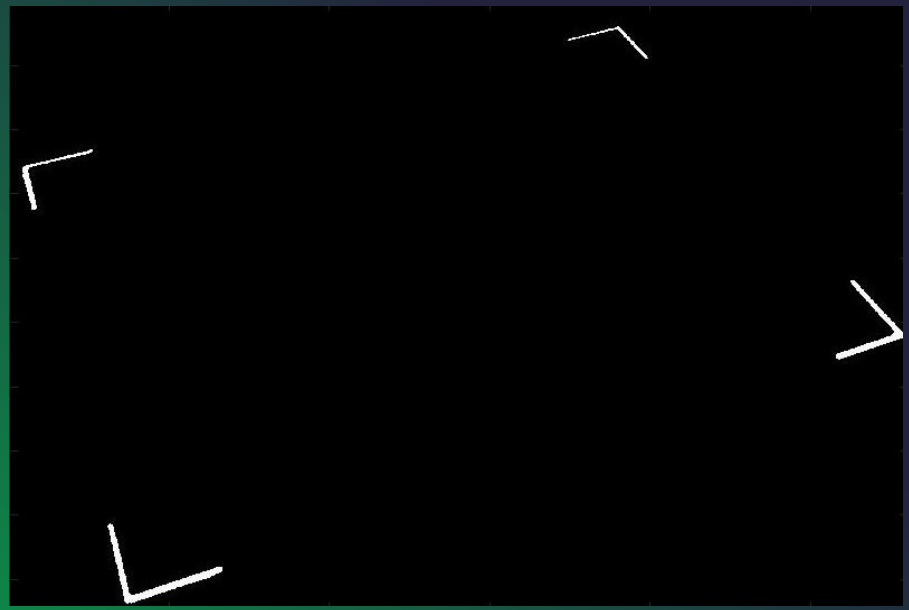
Parte 3: Visión

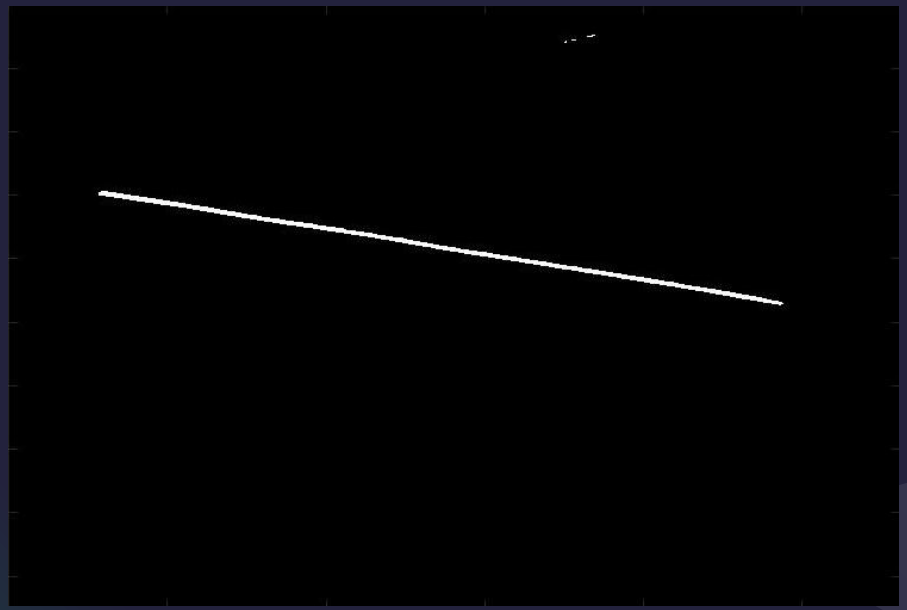
La imagen original se pasa por un filtrado RGB para aislar los colores rojo y verde. Mediante la GUI se puede modificar esta máscara para casos particulares, pero ya viene con valores por defecto que funcionan en la mayoría de casos.



Parte 3: Visión

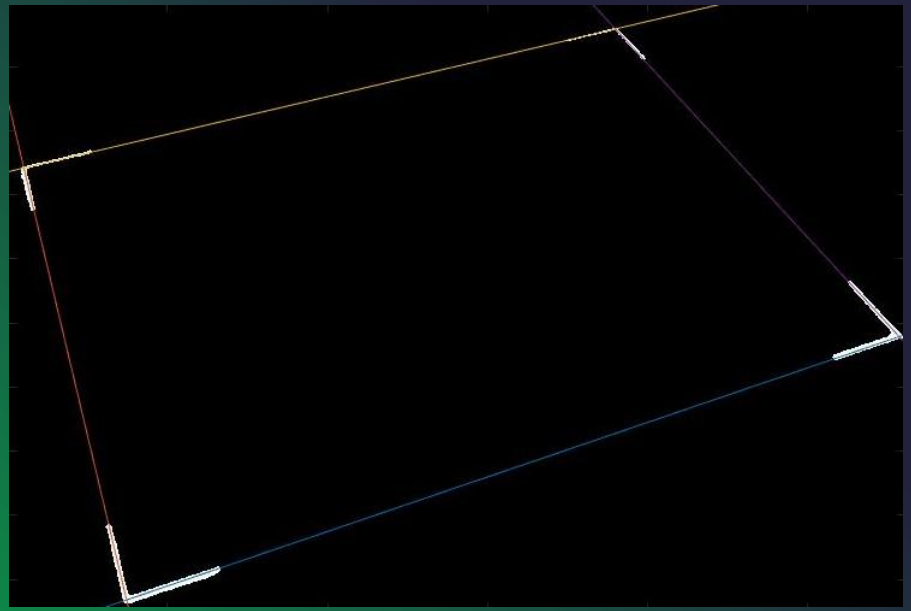
Adicionalmente, se pasa a las imágenes con colores aislados por un proceso de apertura y clausura para eliminar pequeños blobs que hayan sobrevivido al filtrado inicial. Las imágenes filtradas se encuentran a continuación:





Parte 3: Visión

Luego se pasa por el método de Hough para identificar líneas en una imagen. En el programa se usa un proceso de reducción sucesiva de los thresholds hasta detectar la cantidad de líneas deseadas.



Parte 3: Visión

En la imagen con las esquinas, se obtienen las intersecciones entre las líneas encontradas, lo que delimita los valores de las esquinas.

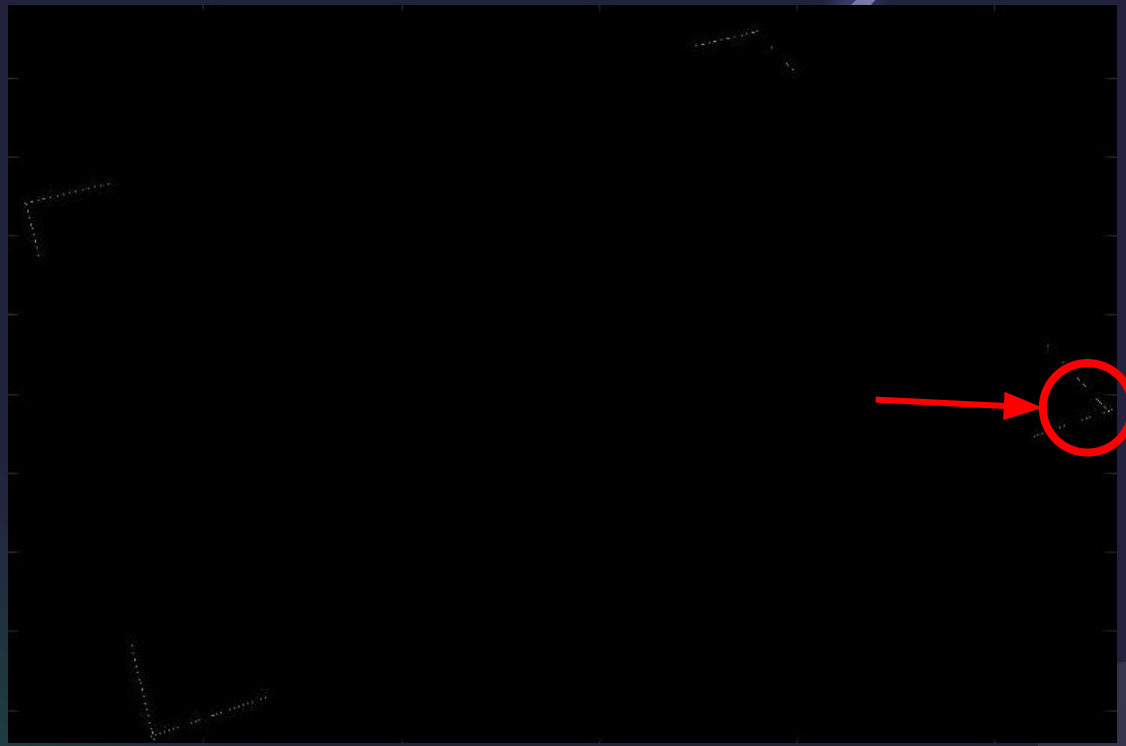
Los puntos en los que hay intersección deben marcar un 2, mientras que el resto marca un 1.

Parte 3: Visión

En la imagen con las esquinas, se obtienen las intersecciones entre las líneas encontradas, lo que delimita los valores de las esquinas.

Los puntos en los que hay intersección deben marcar un 2, mientras que el resto marca un 1.

Sin embargo...

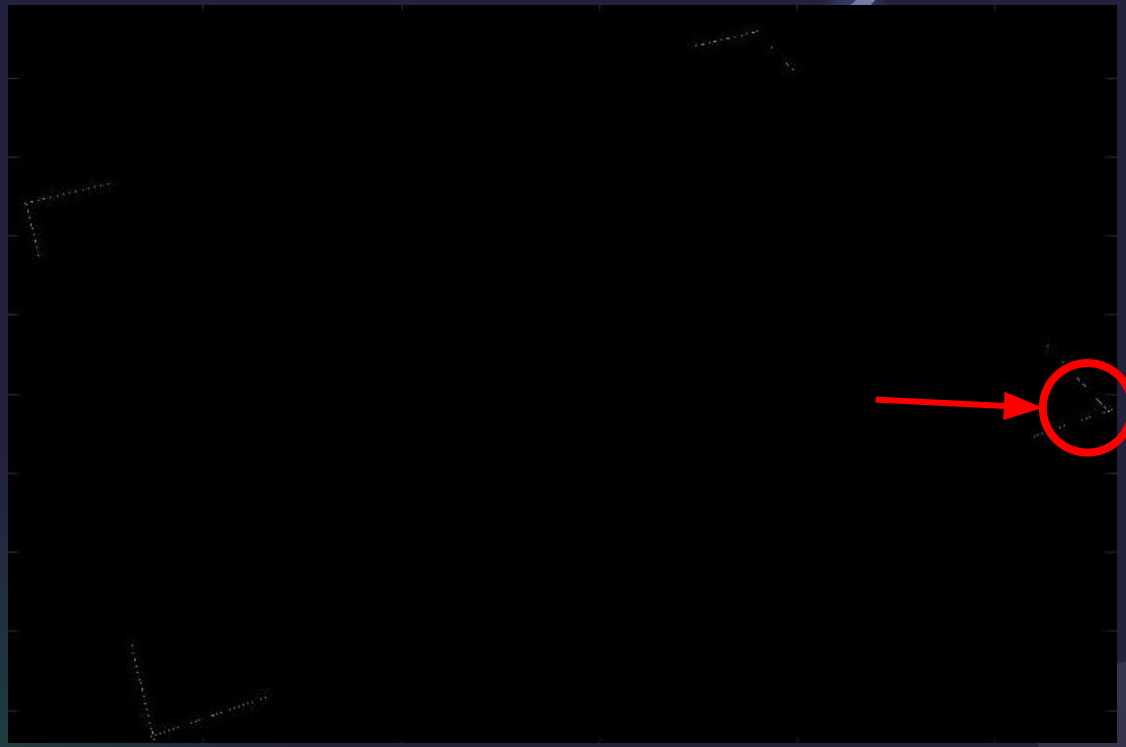


Parte 3: Visión

En la imagen con las esquinas, se obtienen las intersecciones entre las líneas encontradas, lo que delimita los valores de las esquinas.

Los puntos en los que hay intersección deben marcar un 2, mientras que el resto marca un 1.

Sin embargo, a veces la intersección se pierde porque las líneas cambian en u y v simultáneamente.

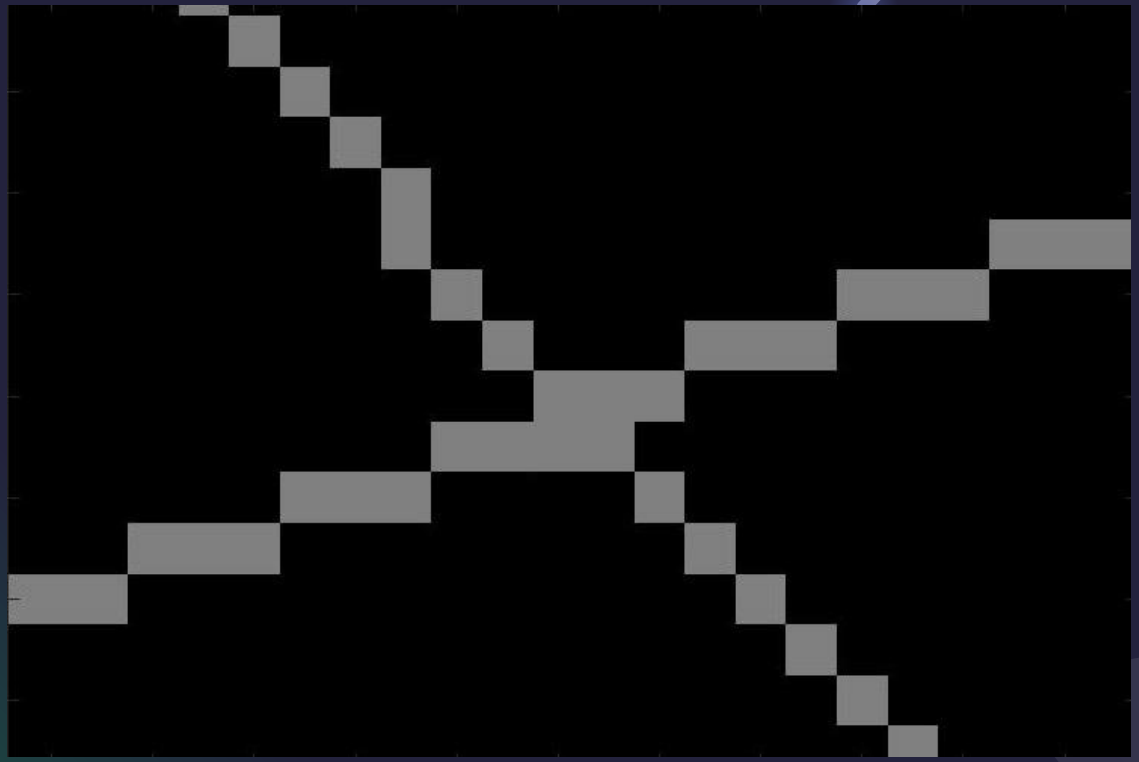


Parte 3: Visión

En la imagen con las esquinas, se obtienen las intersecciones entre las líneas encontradas, lo que delimita los valores de las esquinas.

Los puntos en los que hay intersección deben marcar un 2, mientras que el resto marca un 1.

Sin embargo, a veces la intersección se pierde porque las líneas cambian en u y v simultáneamente.

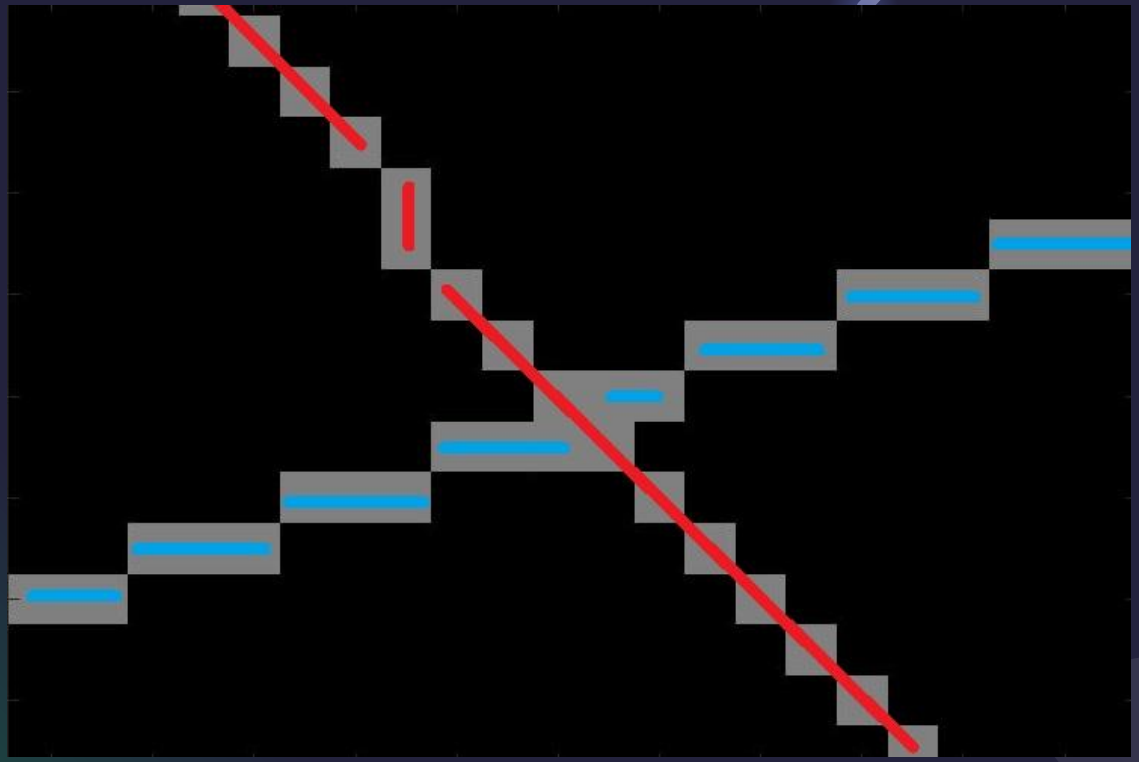


Parte 3: Visión

En la imagen con las esquinas, se obtienen las intersecciones entre las líneas encontradas, lo que delimita los valores de las esquinas.

Los puntos en los que hay intersección deben marcar un 2, mientras que el resto marca un 1.

Sin embargo, a veces la intersección se pierde porque las líneas cambian en u y v simultáneamente.



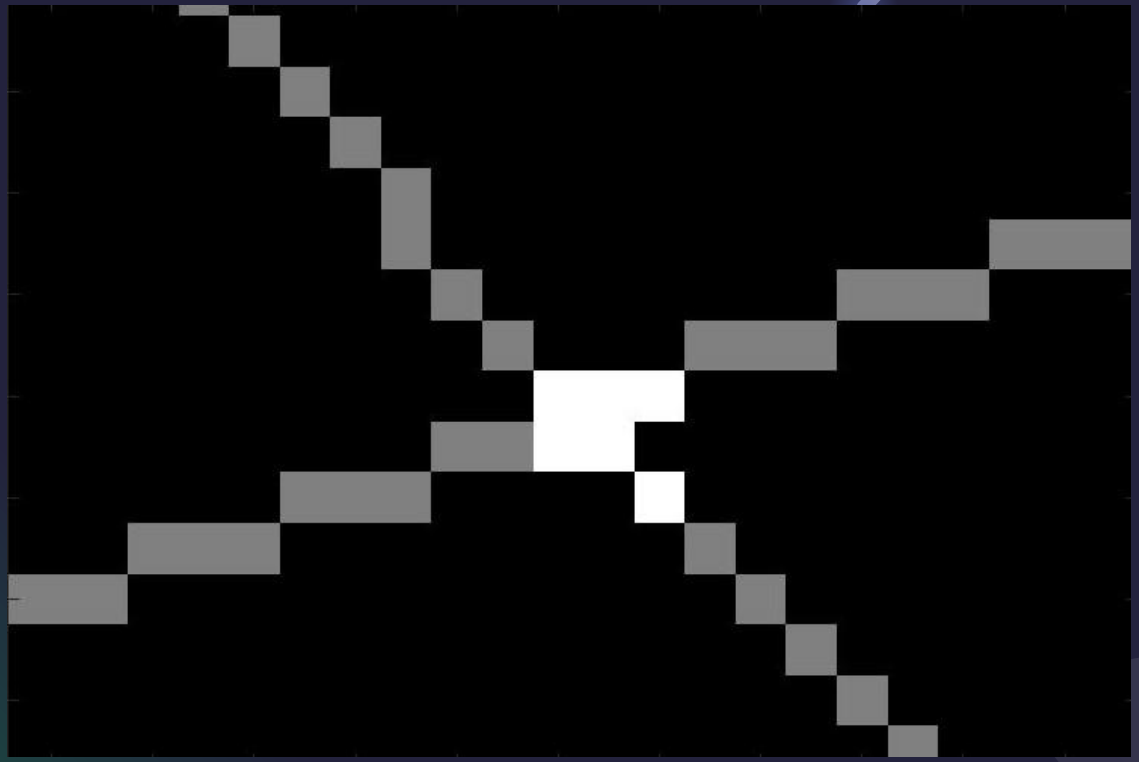
Parte 3: Visión

En la imagen con las esquinas, se obtienen las intersecciones entre las líneas encontradas, lo que delimita los valores de las esquinas.

Los puntos en los que hay intersección deben marcar un 2, mientras que el resto marca un 1.

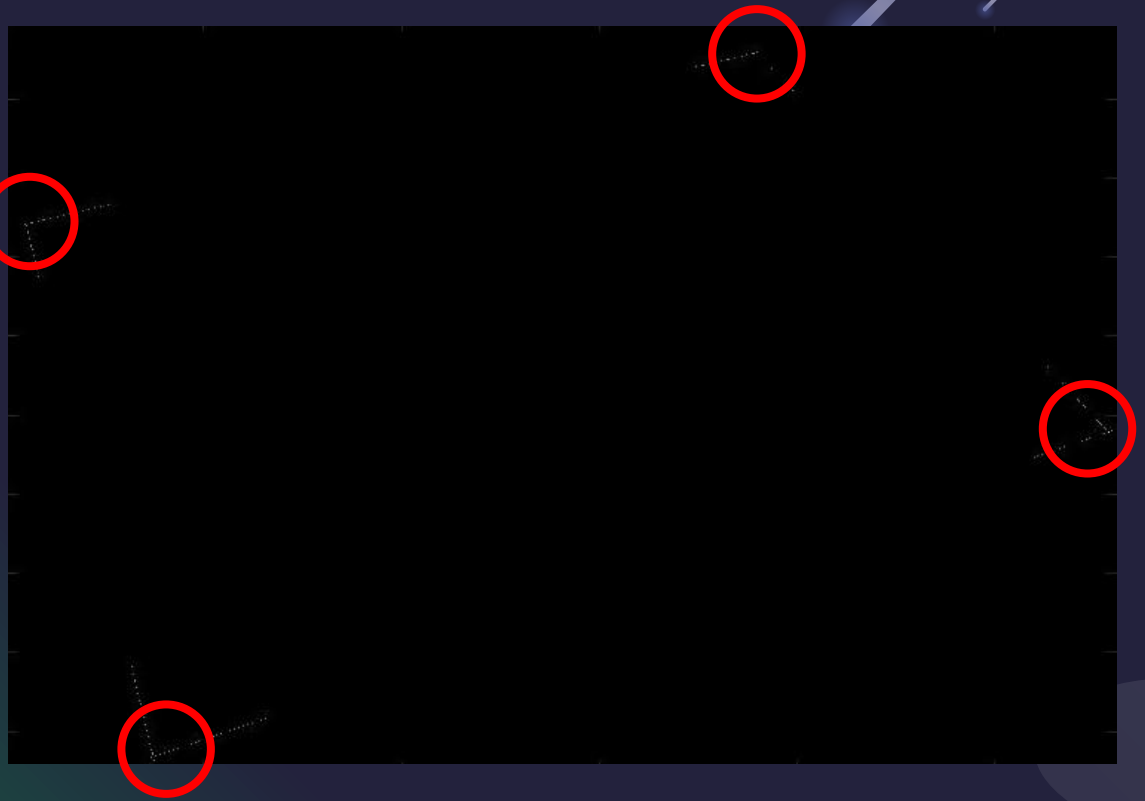
Sin embargo, a veces la intersección se pierde porque las líneas cambian en u y v simultáneamente.

Para solucionarlo, se eligen todos los puntos pertenecientes a un entorno de la intersección, y se los promedia



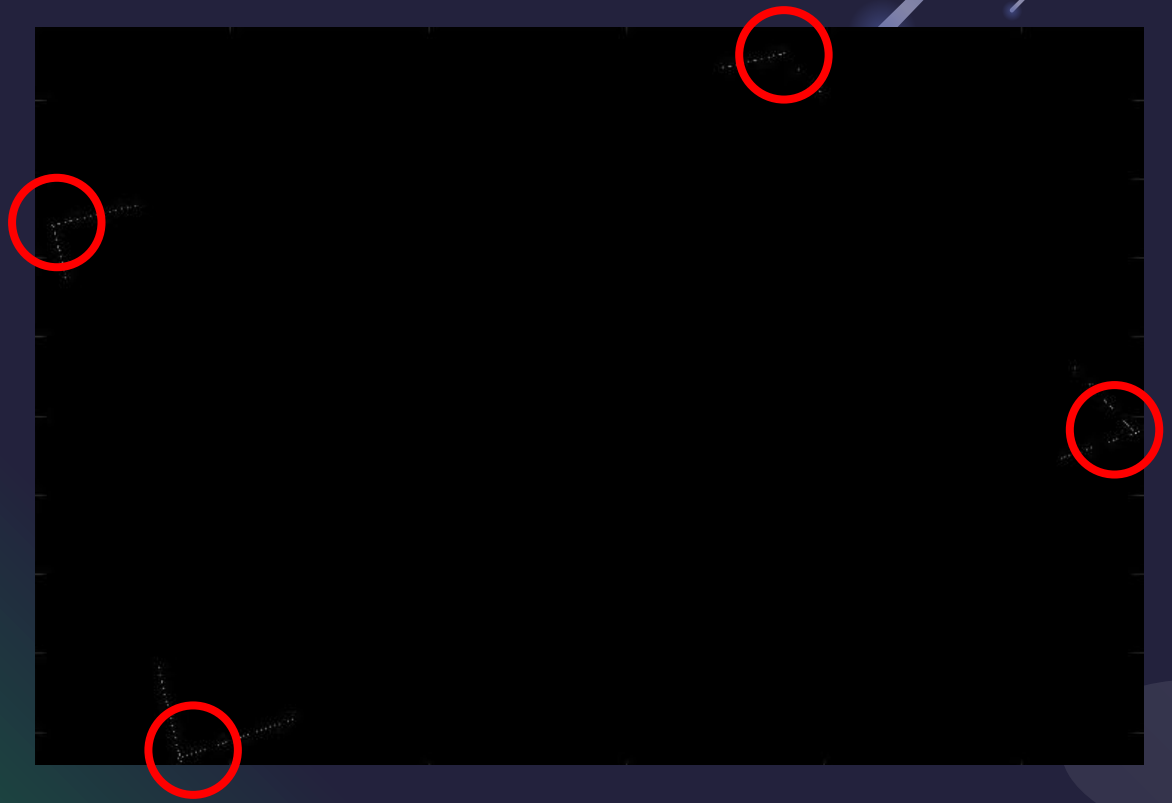
Parte 3: Visión

Ahora que se tienen las 4 esquinas...

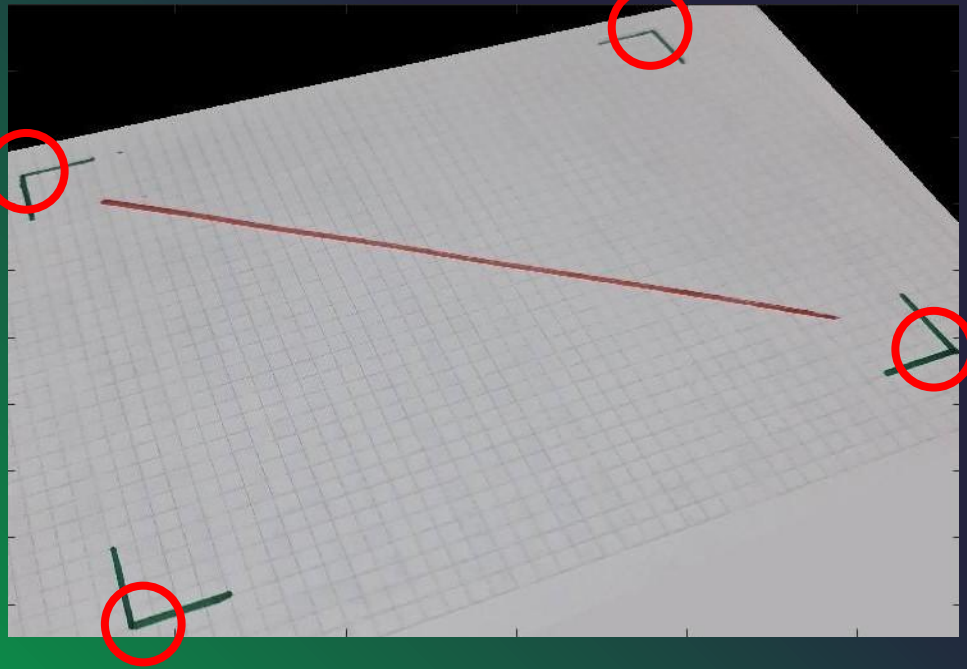


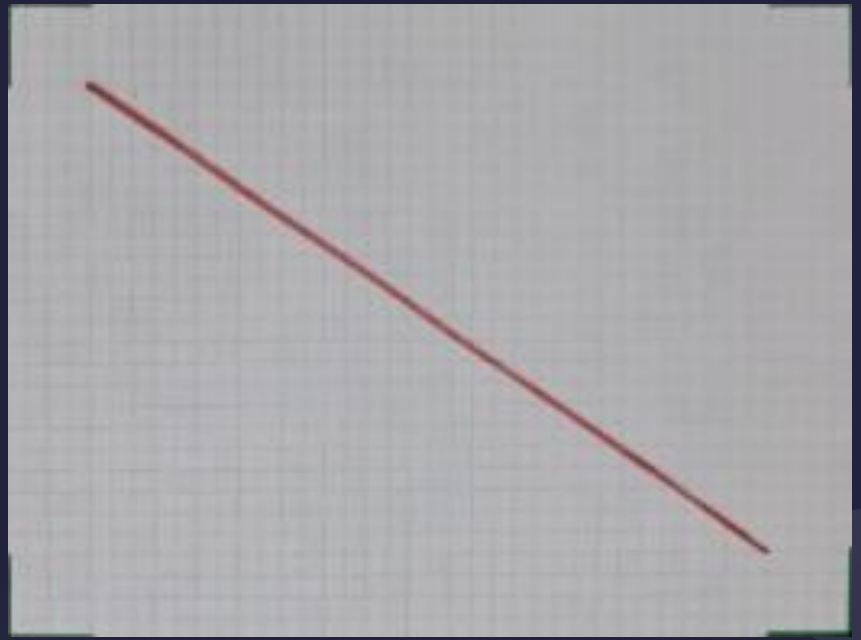
Parte 3: Visión

Ahora que se tienen las 4 esquinas, se usa una transformación morfológica para obtener la imagen corregida.



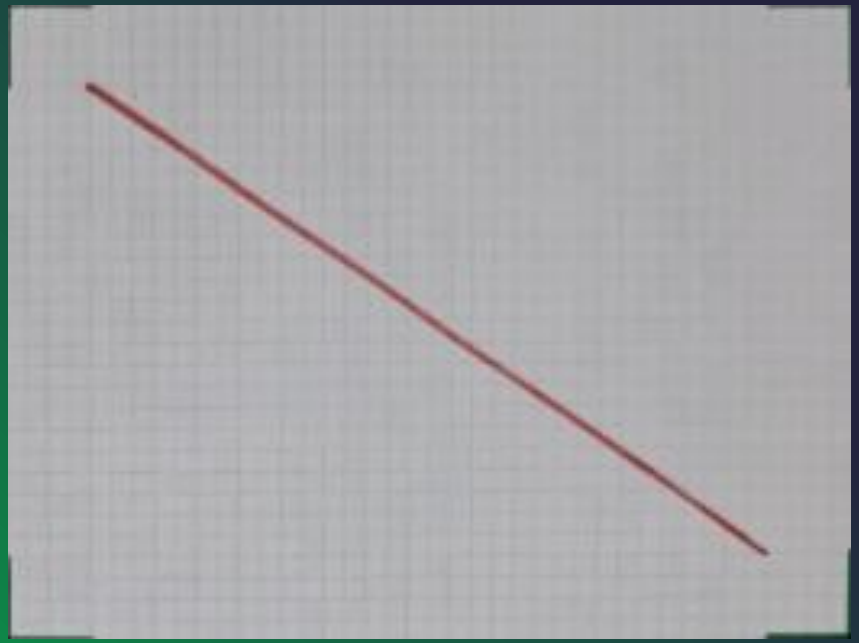
Parte 3: Visión

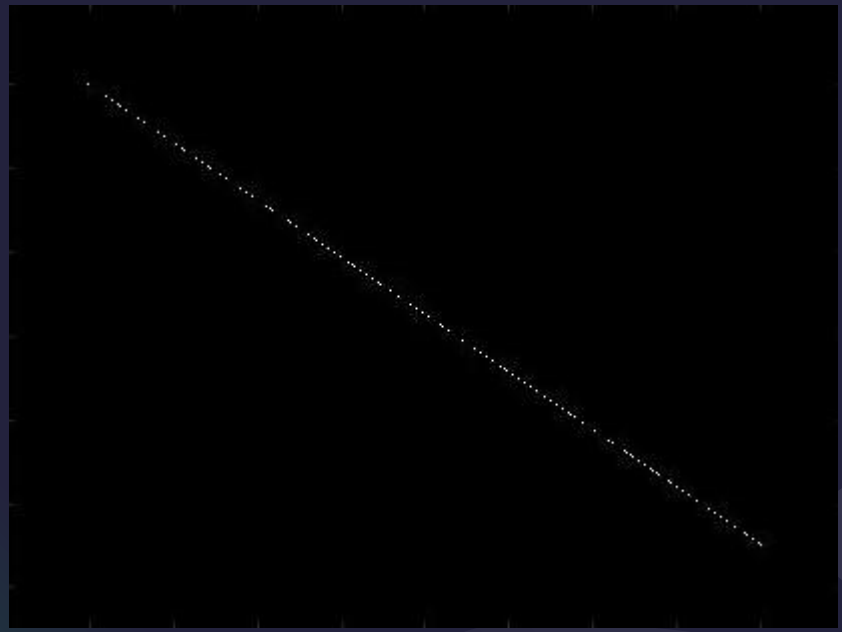




Parte 3: Visión

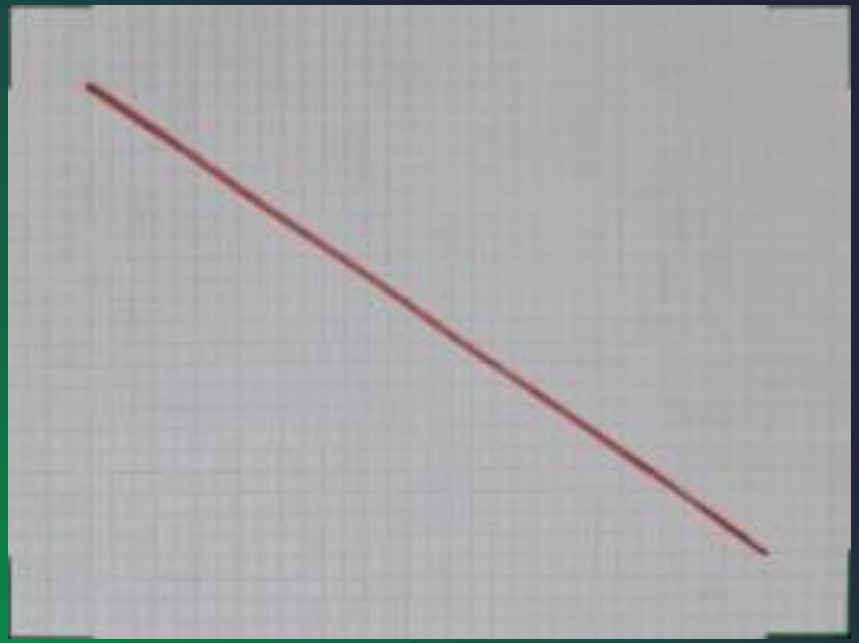
Esa misma transformación se aplica a la línea filtrada, y se obtienen las coordenadas del primer punto y el último. Estos puntos se escalan desde las coordenadas de la imagen a las coordenadas en la vida real.

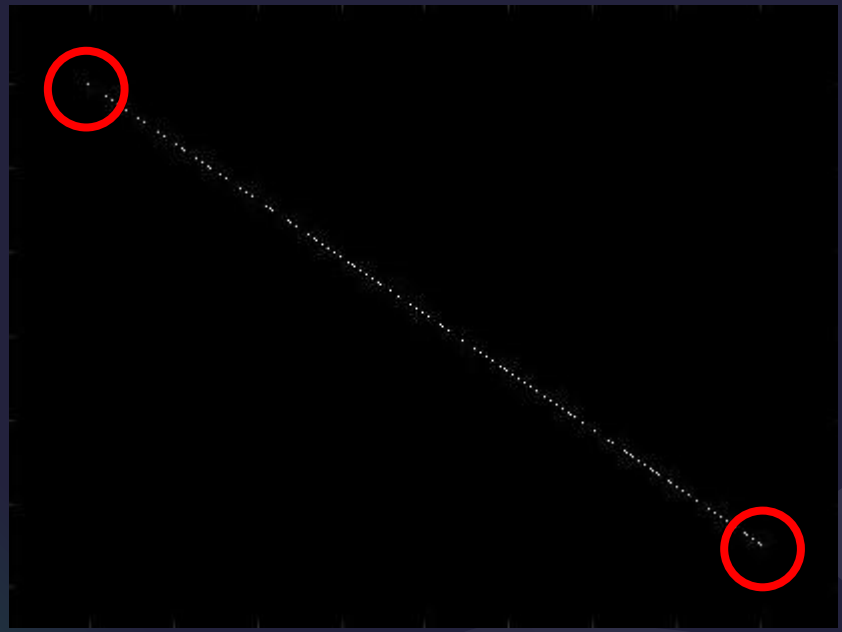




Parte 3: Visión

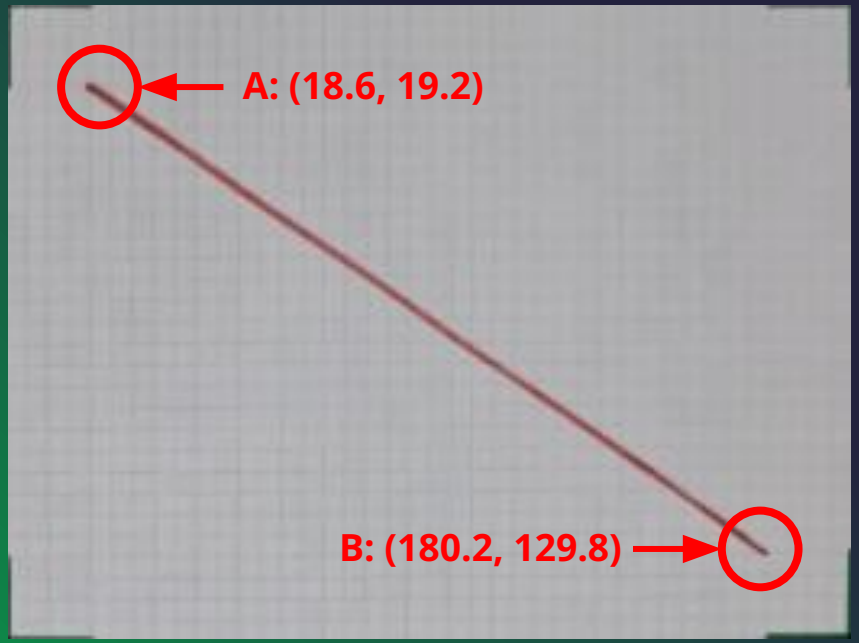
Esa misma transformación se aplica a la línea filtrada, y se obtienen las coordenadas del primer punto y el último. Estos puntos se escalan desde las coordenadas de la imagen a las coordenadas en la vida real.

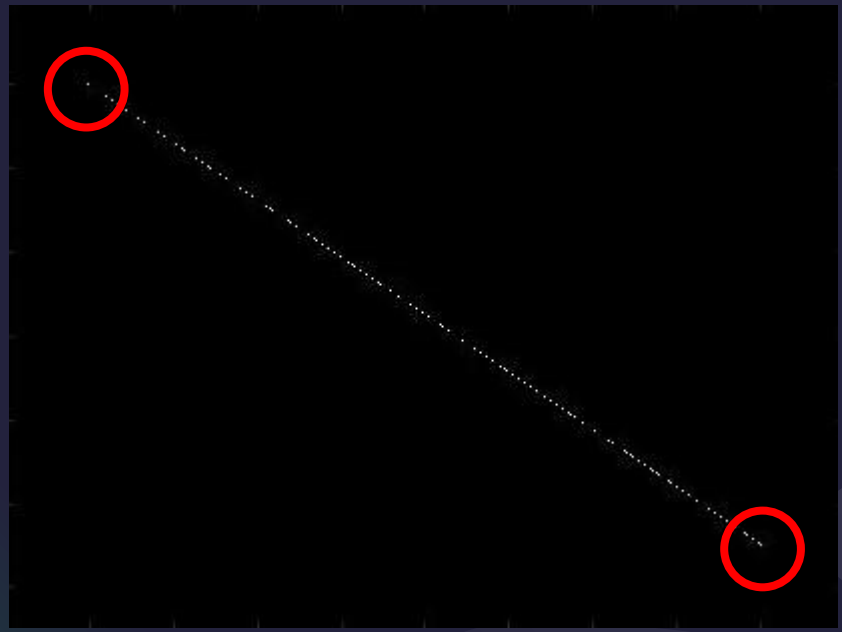




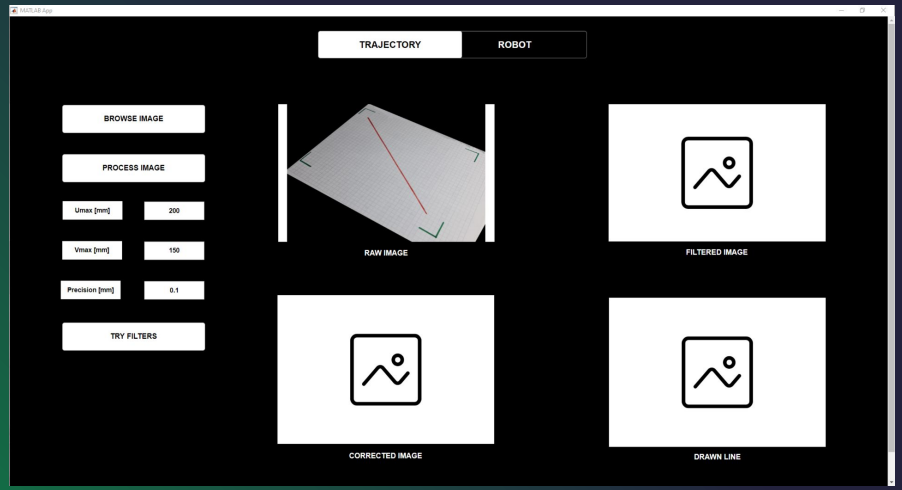
Parte 3: Visión

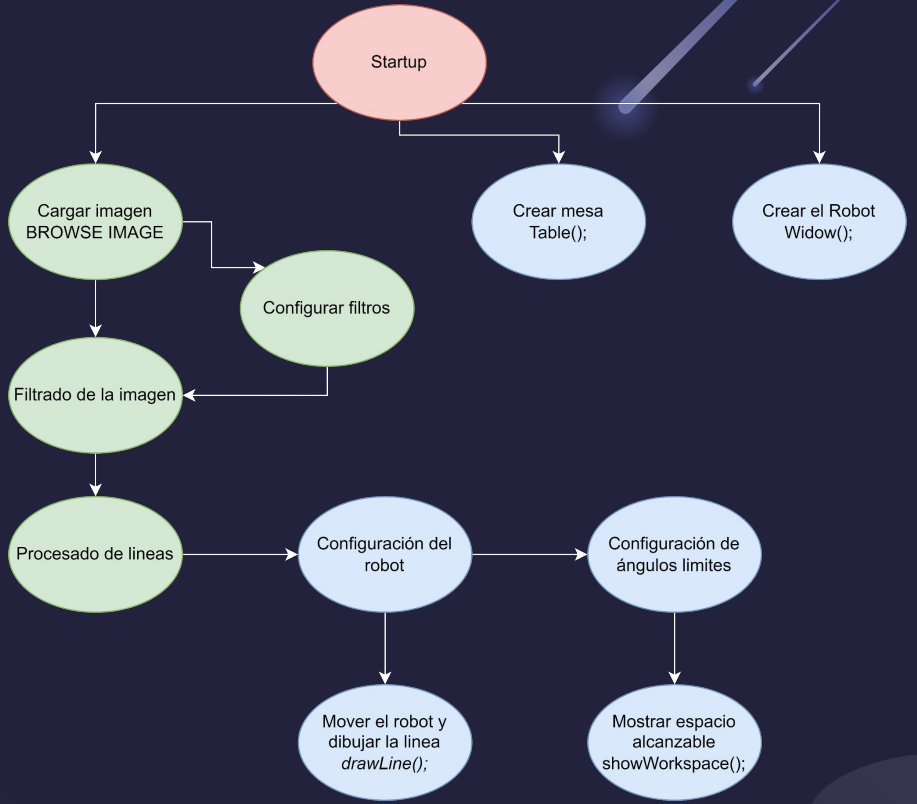
Esa misma transformación se aplica a la línea filtrada, y se obtienen las coordenadas del primer punto y el último. Estos puntos se escalan desde las coordenadas de la imagen a las coordenadas en la vida real.





Funcionamiento del programa





Gracias