



FACULTAD DE
CIENCIAS EXACTAS,
INGENIERIA Y AGRIMENSURA



UNR - FCEIA

ROBÓTICA MÓVIL

Trabajo Práctico N° 3: Visión por Computadora

Có Santiago, Spoletini Bruno

TRABAJO PRÁCTICO N° 3

VISIÓN POR COMPUTADORA

Có Santiago, Spoletini Bruno

1 de noviembre de 2025

Ejercicio 1 - Calibración

Para la calibración de la cámara, publicamos las imágenes de ambas cámaras en loop desde la bag del dataset de calibración de EuRoC con el comando:

```
ros2 bag play archivos/rosbags/roslaunch/cam_calibrator/cam_checkerboard.db3 --loop
```

Luego ejecutamos camera_calibration con Ros2, al cual le pasamos como parámetros las propiedades del tablero que se usarán en la calibración:

```
ros2 run camera_calibration cameracalibrator --size 7x6 --square 0.06  
--no-service-check --ros-args --remap left:=/cam0/image_raw --ros-args --remap  
right:=/cam1/image_raw -p camera:=/my_camera
```

El parámetro –no-service-check hace que el calibrador no busque el servicio set_camera_info del driver de la cámara, ya que estamos publicando las imágenes desde un bag que no contiene esa información.

Este comando inicia el programa de calibración, el cual captura las imágenes del tablero de la bag, y una vez calibrado, nos devuelve los siguientes parámetros extrínsecos e intrínsecos de las cámaras:

- La Cámara derecha
 - La matriz de calibración, que describe la distancia focal y el punto principal.

$$K = \begin{bmatrix} 460,53953 & 0 & 373,61327 \\ 0 & 459,50022 & 254,30122 \\ 0 & 0 & 1 \end{bmatrix}$$

- Los coeficientes de distorsión:

$$D = \begin{bmatrix} -0,288338 & 0,078489 & -0,000242 & 0,000124 & 0,000000 \end{bmatrix}$$

- La matriz de rectificación

$$R = \begin{bmatrix} 0,99990874 & -0,00382163 & -0,01295751 \\ 0,00370313 & 0,99995122 & -0,00915709 \\ 0,01299187 & 0,00910827 & 0,99987412 \end{bmatrix}$$

- La matriz de proyección, que contiene la posición y la orientación de la cámara

$$P = \begin{bmatrix} 437,86901 & 0 & 375,54622 & -47,97551 \\ 0 & 437,86901 & 256,51072 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

- La Cámara izquierda

- La matriz de calibración

$$K = \begin{bmatrix} 461,45666 & 0 & 364,35977 \\ 0 & 460,19502 & 249,31317 \\ 0 & 0 & 1 \end{bmatrix}$$

- Los coeficientes de distorsión:

$$D = \begin{bmatrix} -0,292081 & 0,082722 & 0,000190 & -0,000076 & 0,000000 \end{bmatrix}$$

- La matriz de rectificación

$$R = \begin{bmatrix} 0,99998634 & -0,00148348 & -0,00501188 \\ 0,00152919 & 0,99995716 & 0,00912909 \\ 0,00499813 & -0,00913663 & 0,99994577 \end{bmatrix}$$

- La matriz de proyección

$$P = \begin{bmatrix} 437,86901 & 0 & 375,54622 & 0 \\ 0 & 437,86901 & 256,51072 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Ejercicio 2

Para el ejercicio 2 usamos el dataset "Vicon Room 1 01" de EuRoC, el cual consta de una bag de ROS2 con imágenes de una cámara estéreo, junto con la trayectoria ground-truth que siguió la cámara. Tomando un par de imágenes cualesquiera del dataset, realizamos los apartados siguientes.

Apartado a)

Para rectificar las imágenes usamos la librería de OpenCV.

Llamamos a `stereoRectify` para obtener las matrices de rectificación en base a los parámetros que obtuvimos en la calibración.

Usando los parámetros obtenidos en la calibración, llamamos a la función `stereoRectify` para obtener las matrices de rectificación, que luego usamos para obtener los mapas de remapeo con `initUndistortRectifyMap`. Una vez que tenemos los mapas de remapeo para cada cámara, usamos la función `remap` para obtener las imágenes rectificadas.

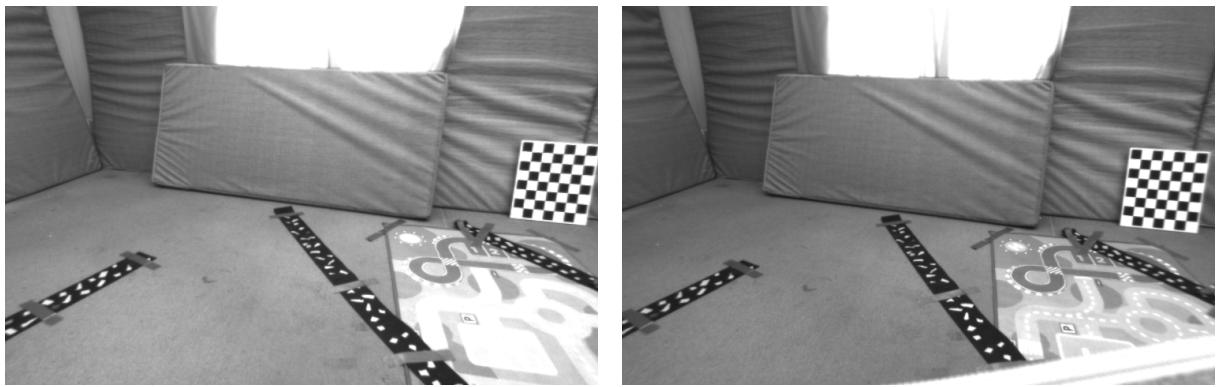


Figura 1: Imágenes izquierda y derecha rectificadas

Apartado b)

Para la extracción de features usamos el detector de keypoints FAST y el descriptor BRIEF:

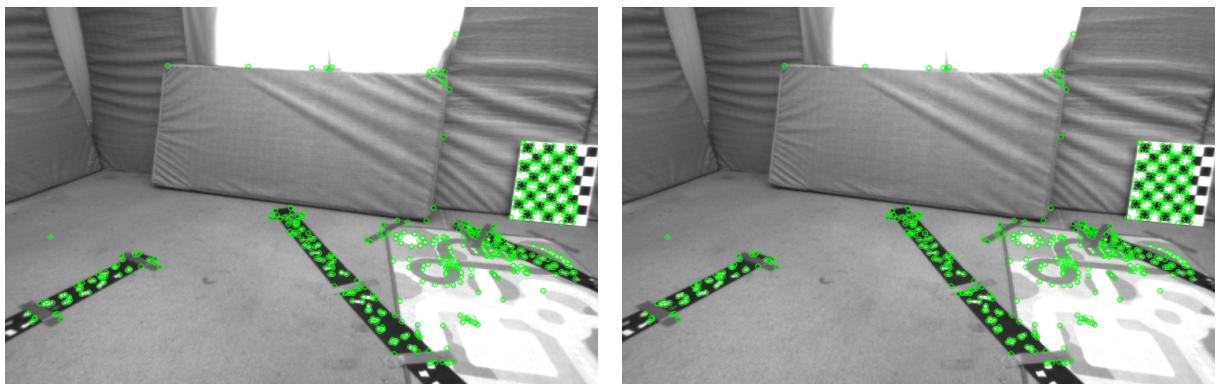


Figura 2: Features de imágenes izquierda y derecha

Apartado c)

Con las features obtenidas en el apartado b buscamos los matches entre ambas imágenes:

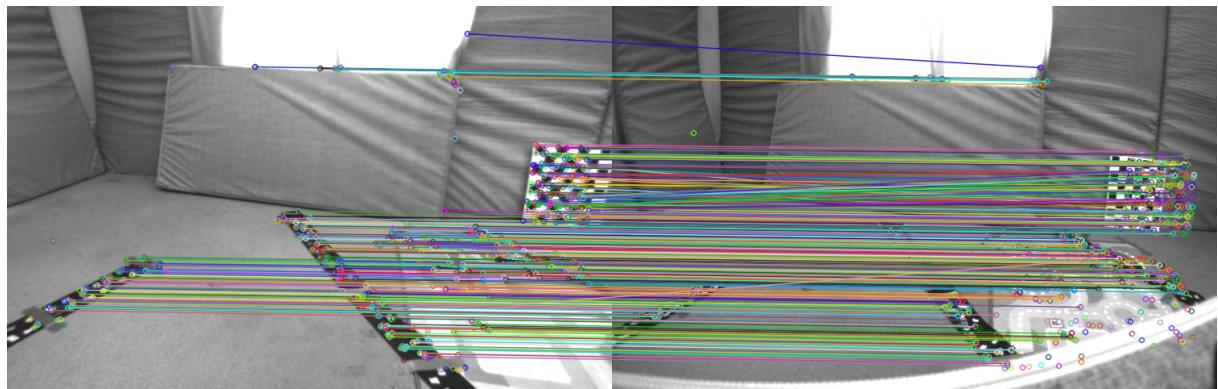


Figura 3: Todos los matches

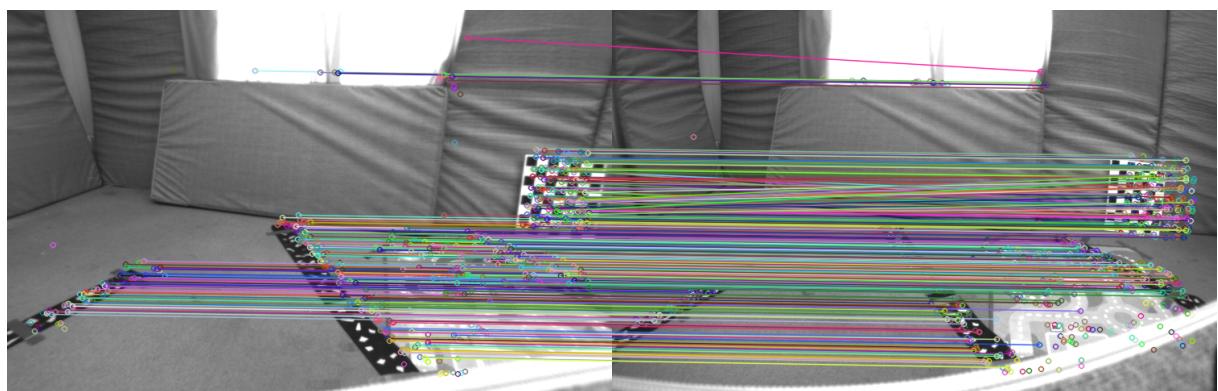


Figura 4: Matches con distancia menor a 30

Apartado d)

Usando triangulatePoints obtenemos los puntos 3d de los matches obtenidos en el apartado anterior. Una vez obtenidos los puntos 3d, usamos un nodo de ROS2 para publicarlos en forma de nube de puntos. A continuación, vemos el resultado en RViz:

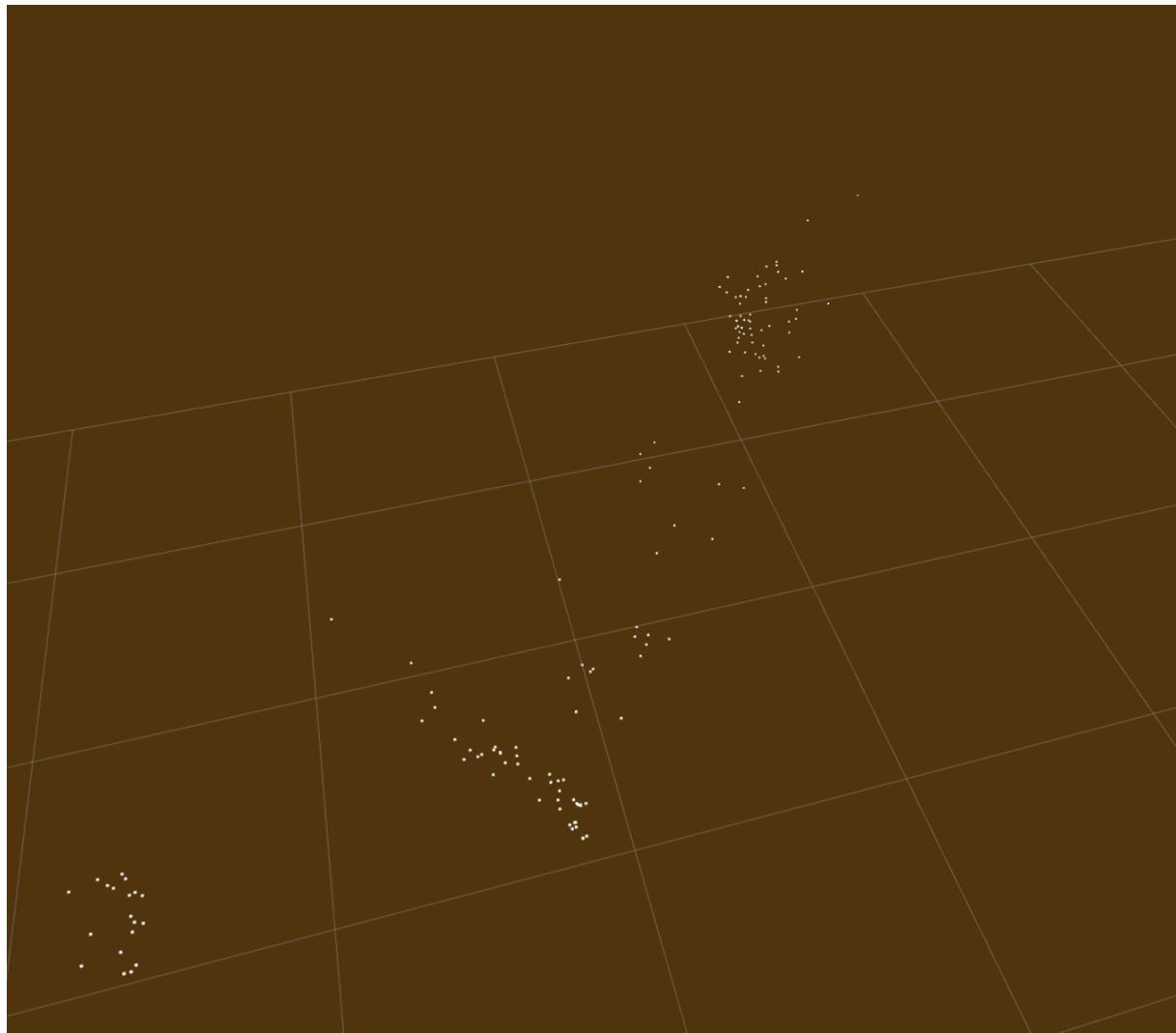


Figura 5: Nube de puntos de los matches

Apartado e)

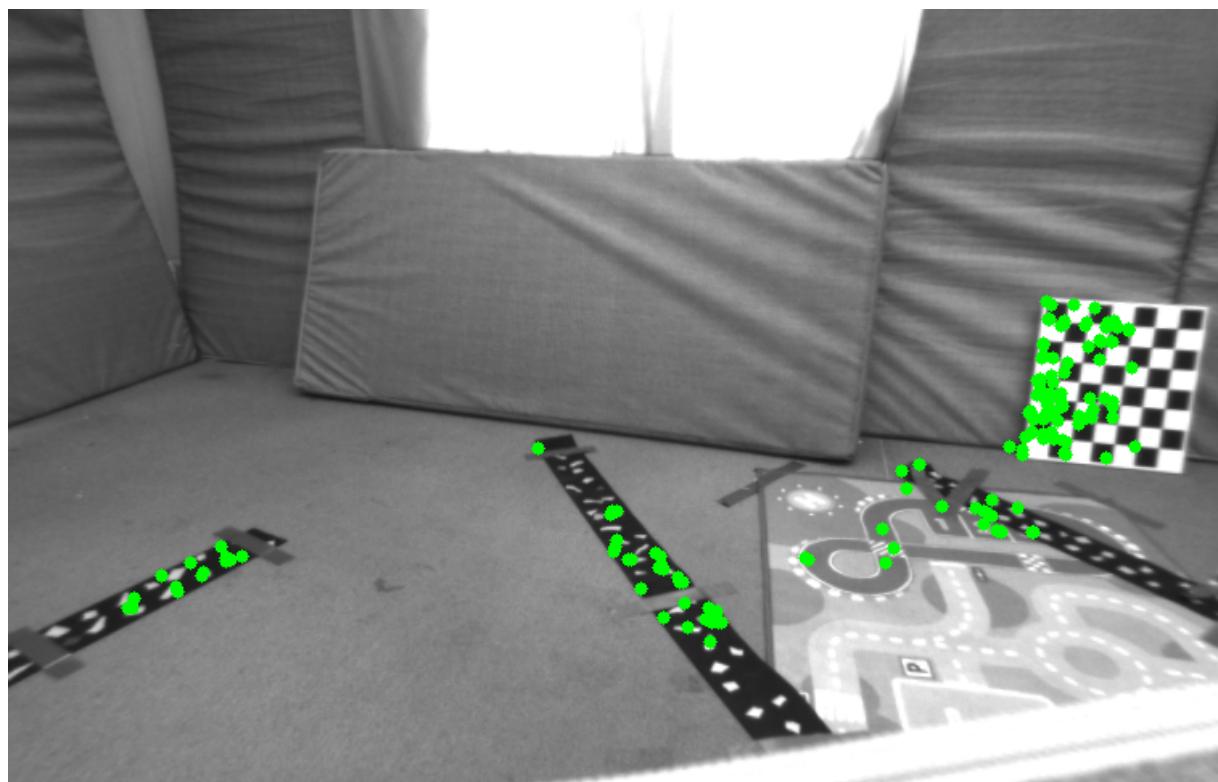


Figura 6: Puntos de la primera imagen transformados por la matriz homográfica

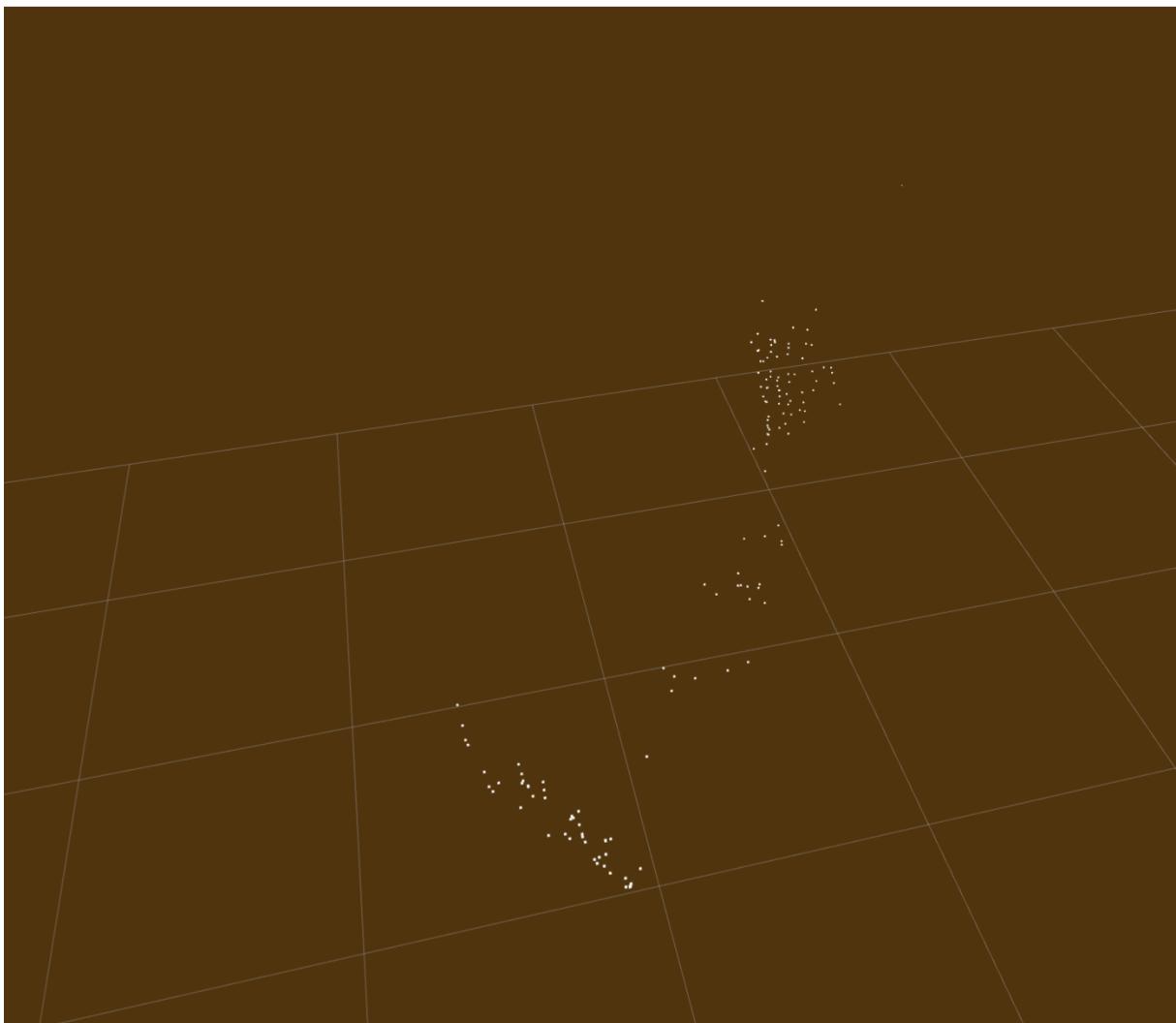


Figura 7: Nube de puntos filtrados

Apartado f)

Usando todas las imágenes del dataset, mapeamos el entorno usando las poses dadas por el ground-truth del dataset.

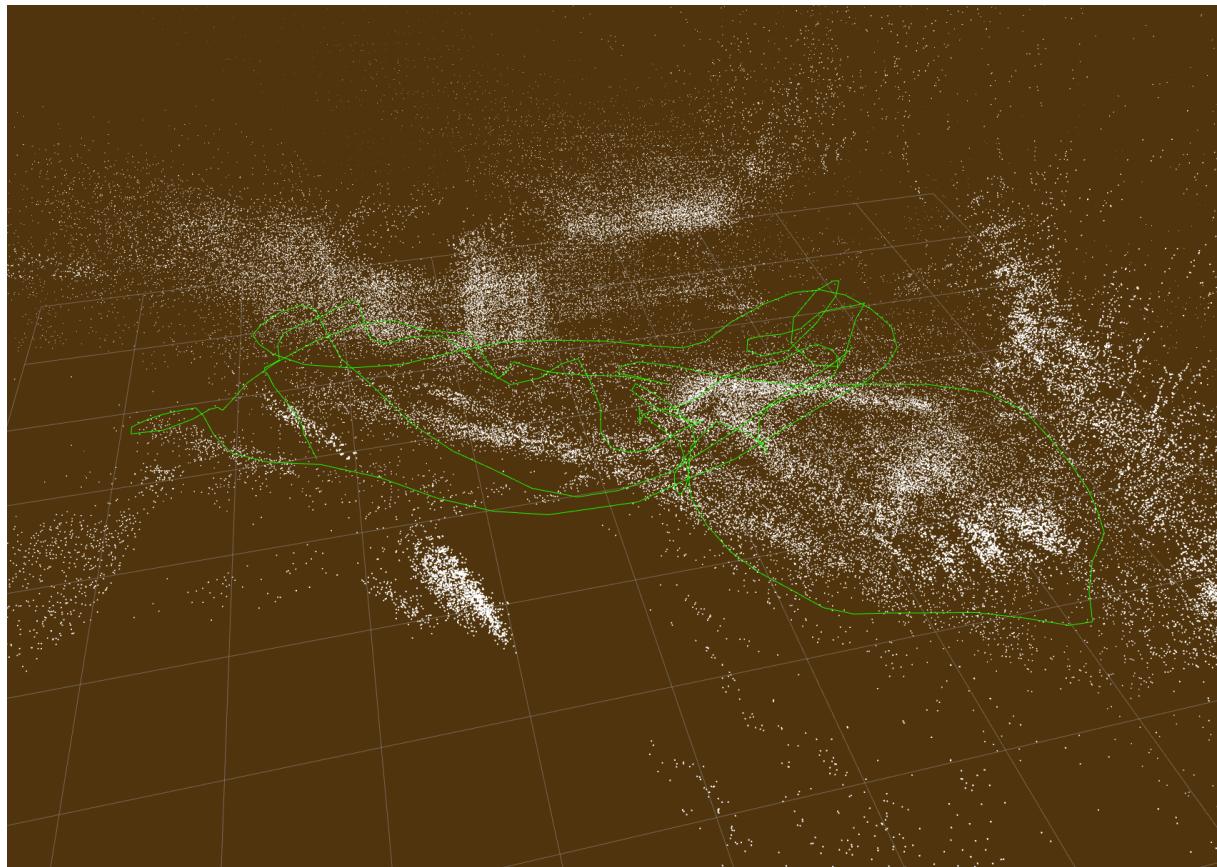


Figura 8: Nube de puntos del dataset con matches espúreos filtrados

Apartado g)

Obtenemos el mapa de disparidad usando la función StereoMatcher::compute:



Figura 9: Mapa de disparidad

Apartado h)

Obtenemos el mapa de disparidad del paso anterior para cada par de imágenes del dataset para reconstruir de manera densa la escena observada.

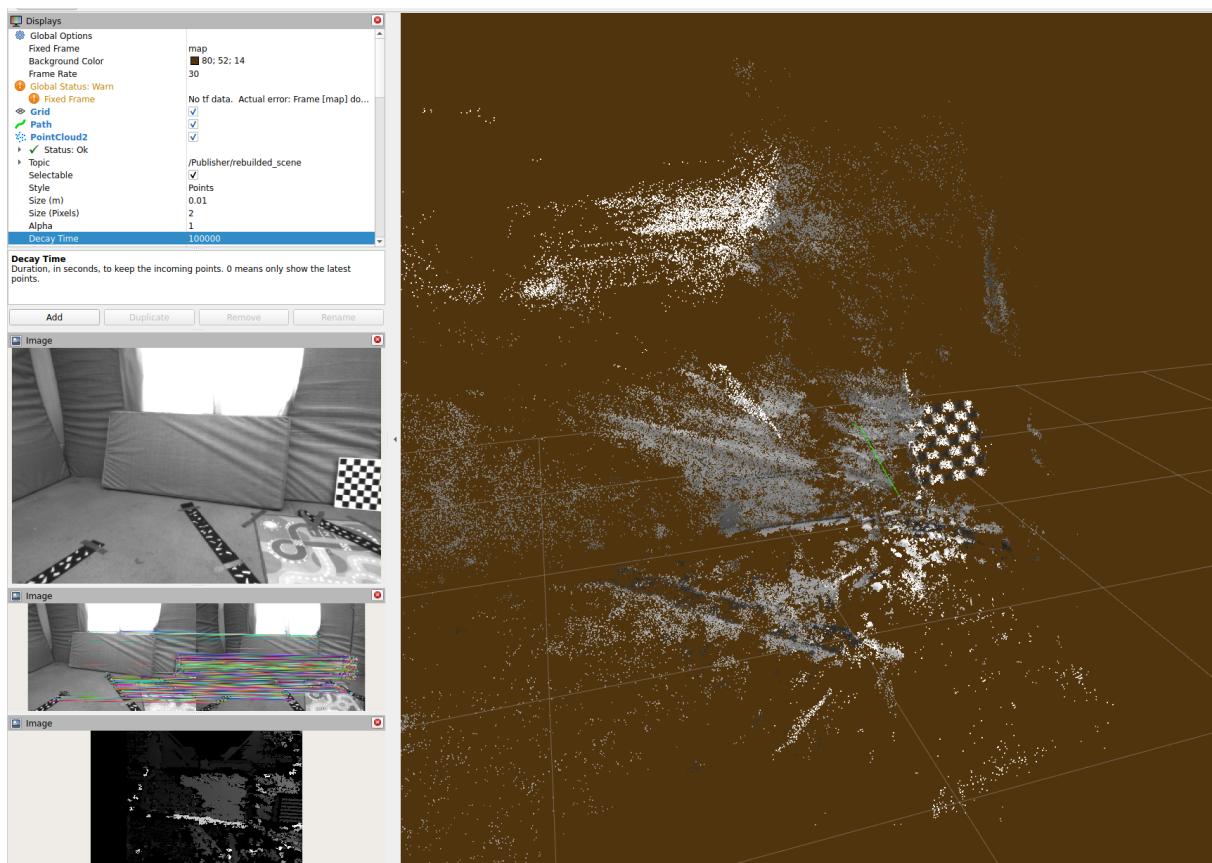


Figura 10: Reconstrucción densa

Apartado j)

Subapartado i

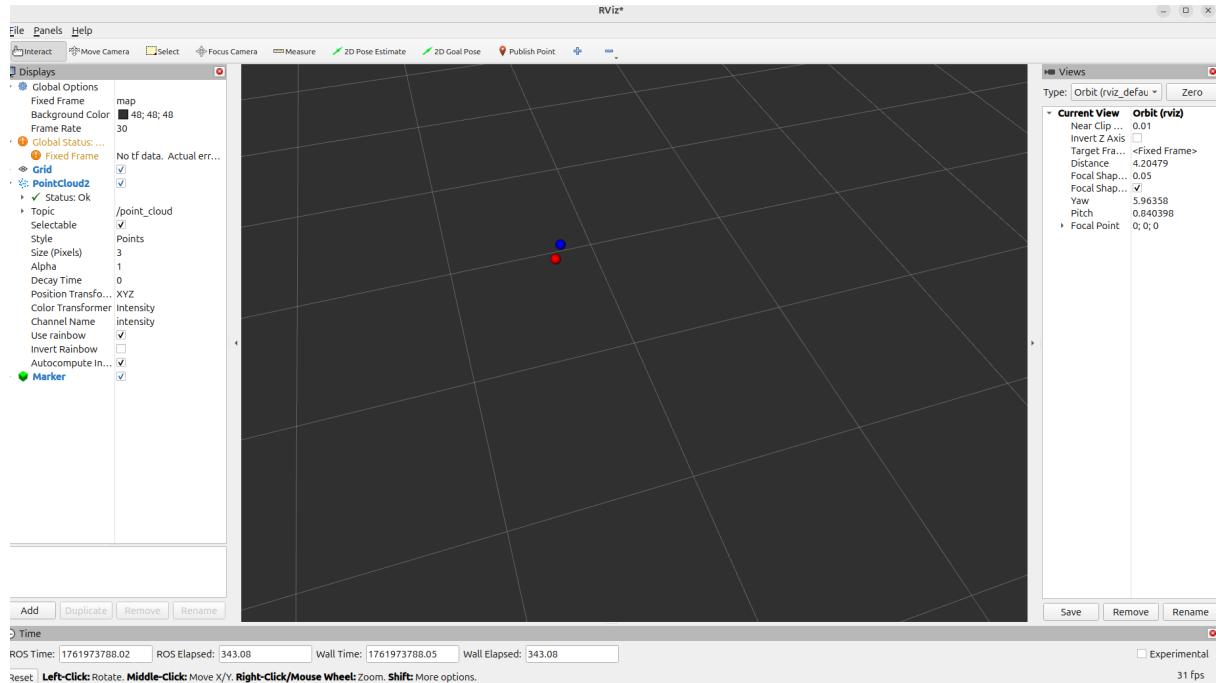


Figura 11: Vista 1 de las poses de las cámaras

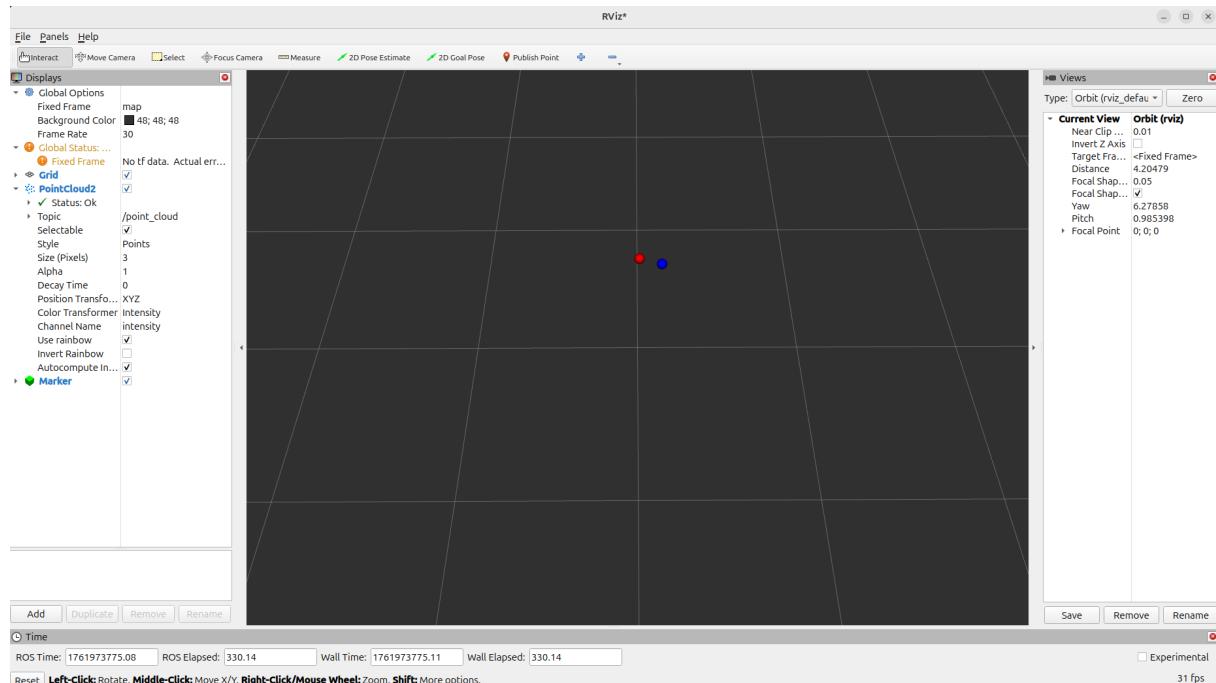


Figura 12: Vista 2 de las poses de las cámaras

Subapartado ii

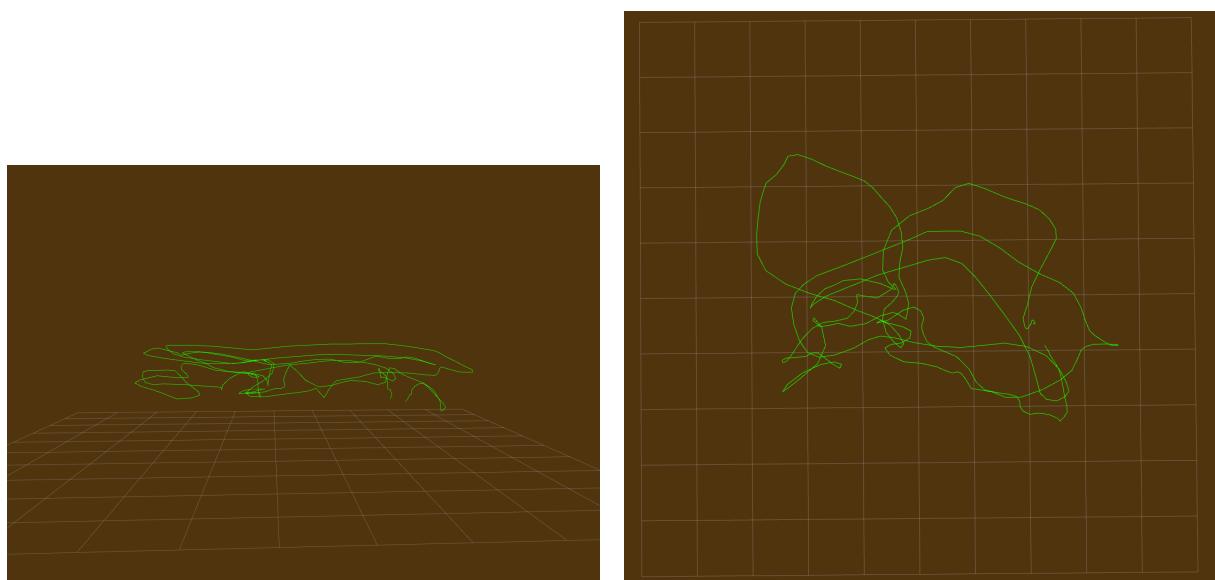


Figura 13: Trayectoria de la cámara izquierda - Vista de lado y superior



Figura 14: Vista completa de la trayectoria de la cámara izquierda