

Robótica Móvil - Trabajo Práctico 3 - Visión por computadora

Francisco Raverta, Tania Ferreyra

2do cuatrimestre - 2023

Los scripts utilizados durante el desarrollo de este trabajo pueden encontrarse en <https://github.com/FranciscoRaverta/RoboticaMovilTP3>

1. Datos

Se utilizarán los datos provistos por la cátedra, tanto del rosbag con la secuencia de imágenes capturadas con una cámara estéreo como los parámetros de calibración provistos.

2. Calibración

Se toma como calibración de la cámara estéreo utilizada en la generación del rosbag otorgado por la cátedra a los siguientes parámetros, también otorgados por la cátedra:

$$K_{left} = \begin{pmatrix} 721.247679 & 0.000000 & 326.193683 \\ 0.000000 & 721.052113 & 233.670426 \\ 0.000000 & 0.000000 & 1.000000 \end{pmatrix} \quad (1)$$

$$Distortion_{left} = (0.076741 \quad -0.102294 \quad 0.000895 \quad 0.001320 \quad 0.000000) \quad (2)$$

$$P_{left} = \begin{pmatrix} 759.361724 & 0.000000 & 327.398655 & 0.000000 \\ 0.000000 & 759.361724 & 242.447603 & 0.000000 \\ 0.000000 & 0.000000 & 1.000000 & 0.000000 \end{pmatrix} \quad (3)$$

$$K_{right} = \begin{pmatrix} 720.272965 & 0.000000 & 332.897424 \\ 0.000000 & 719.755990 & 251.597012 \\ 0.000000 & 0.000000 & 1.000000 \end{pmatrix} \quad (4)$$

$$Distortion_{right} = (0.073915 \quad -0.098917 \quad -0.001737 \quad 0.002291 \quad 0.000000) \quad (5)$$

$$P_{right} = \begin{pmatrix} 759.361724 & 0.000000 & 327.398655 & -45.300238 \\ 0.000000 & 759.361724 & 242.447603 & 0.000000 \\ 0.000000 & 0.000000 & 1.000000 & 0.000000 \end{pmatrix} \quad (6)$$

donde K es la matriz de parámetros intrínsecos, $Distortion$ es un vector con los parámetros de distorsión de la cámara, y P es la matriz extrínseca de la cámara. Los subíndices *left* y *right* indican la cámara izquierda y derecha respectivamente de la cámara estéreo.

3. Rectificación y Sincronización de Imágenes

Se utilizó el paquete de *stereo_image_proc* de ROS2 para rectificar las imágenes provistas en el rosbag. Éste paquete publica en los tópicos */left/image_rect* y */right/image_rect* las imágenes rectificadas. Para leerlas se crearon subscribers a dichos tópicos, y luego se utilizó un objeto TimeSynchronizer para sincronizar los mensajes recibidos en tiempo, de forma que se obtengan simultáneamente las imágenes capturadas por la cámara izquierda y derecha que se corresponden temporalmente.

A este TimeSynchronizer se le asignó un callback que contiene el desarrollo de los siguientes puntos.

4. Extracción de Features Visuales y Correspondencia entre Frames

Se utilizó ORB (*Oriented FAST and Rotated BRIEF*) como detector y descriptor de features sobre las imágenes utilizando librerías de opencv. Se eligió este par detector-descriptor por sus propiedades de invariancia a rotaciones, su velocidad de cómputo y su sensibilidad al ruido. En la Figura 1 se puede ver un ejemplo de puntos detectados en un frame del dataset. De aquí hasta el punto 10 se utilizará el mismo par de imágenes estéreo para ejemplificar y graficar lo realizado.

Se realizó además una correspondencia de features entre los pares de imágenes de las cámaras izquierda y derecha sincronizadas temporalmente, mediante funciones de opencv que comparan la descripción de cada feature entre ambas imágenes, e identifica como *matches* aquellas cuya descripción es semejante. En la Figura 3 pueden observarse visualmente estas correspondencias para un par de imágenes del dataset.

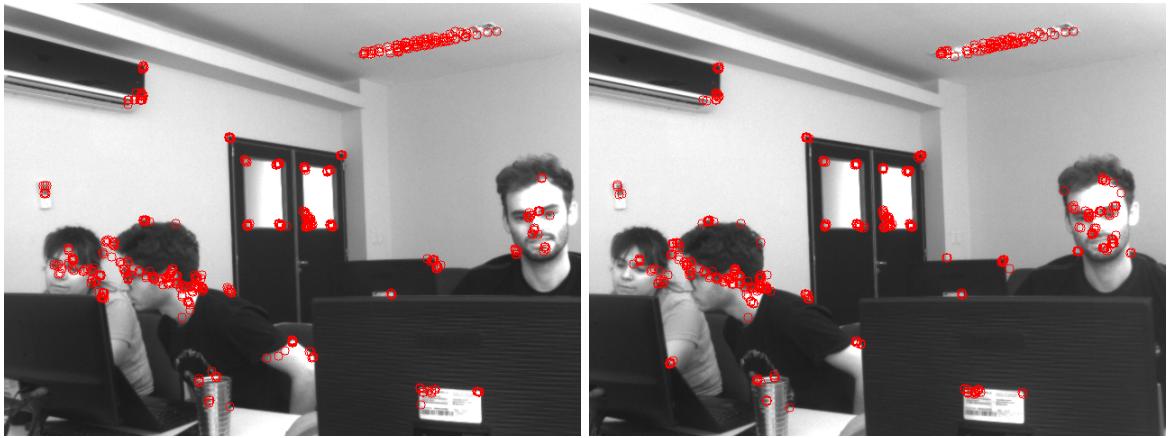


Figure 1: Puntos detectados con ORB en una imagen estéreo tomada del dataset.



Figure 2: Correspondencia entre puntos detectados entre pares de imágenes izquierda y derecha.

5. Triangulación de puntos

A partir de las correspondencias visuales obtenidas antes, se realizó una triangulación de los puntos correspondidos para encontrar su proyección en el espacio 3D. Para ésto se los pasa a coordenadas homogéneas, y se aplican técnicas de geometría epipolar para encontrar el punto de intersección entre los rayos que proyectan los puntos matcheados desde el centro de cada cámara. En la Figura 4 se puede observar la nube de puntos obtenida para el par de imágenes correspondidas mostradas en la Figura 3.

6. Matriz Homográfica y clasificación *inliers-outliers*

Se calculó la matriz homográfica entre cada par de imágenes izquierda y derecha. La matriz homográfica es una matriz de transformación que mapea puntos de una imagen a su posición correspondiente en la otra imagen. El cálculo de esta matriz usando opencv se realizó utilizando un algoritmo de RANSAC que selecciona pares de puntos correspondidos entre ambas imágenes de forma aleatoria, y con ellos busca la matriz de homografía H que explique la mayor cantidad de pares de puntos correspondidos. Ésta será la homografía encontrada para cada par de imágenes. Los puntos que resulten no explicados por esta homografía se denominan *outliers*, y se consideran como pares de puntos mal correspondidos. De aquí en adelante se trabajarán con los pares de puntos *inliers*, considerados bien correspondidos.

En la Figura 5 pueden verse sobre cada imagen (izquierda y derecha) sus keypoints con matching con la otra cámara, más la proyección de los puntos matcheados correspondientes de la otra imagen transformados por la matriz de homografía.

En la Figura 6 se pueden ver los puntos triangulados correspondientes a los pares de puntos considerados *inliers* en el mismo par de imágenes utilizadas en la Figura 4.

7. Mapa de Disparidad

Computamos el mapa de disparidad para cada par de imágenes, que muestra la distancia entre dos pares de pixeles en cada imagen que se correspondan con el mismo punto triangulado 3D. La disparidad puede

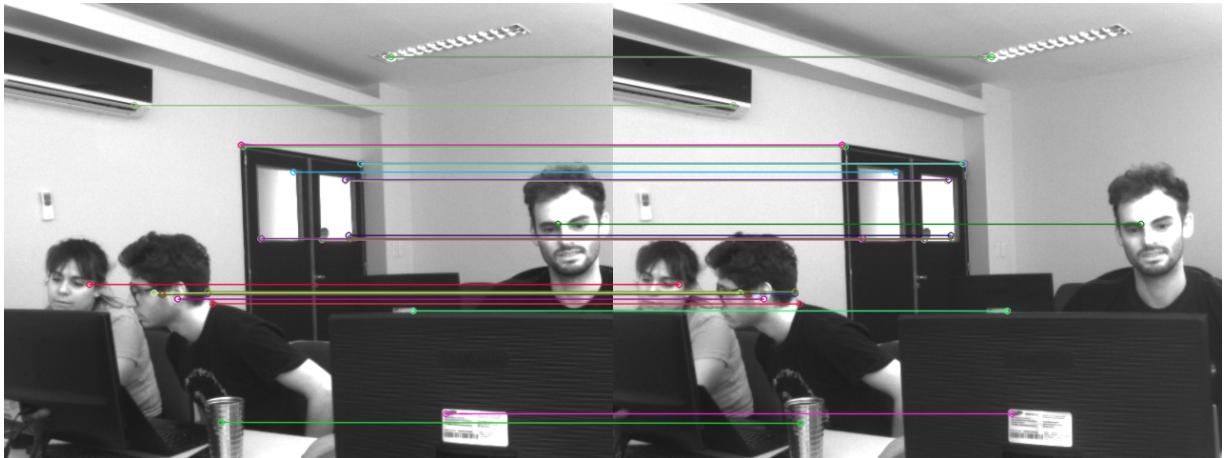


Figure 3: Las 30 correspondencias entre puntos detectados entre pares de imágenes izquierda y derecha con la menor distancia de matching.



Figure 4: Puntos triangulados a partir del matching de características entre imágenes izquierda y derecha

relacionarse con la distancia desde la cámara estéreo hasta el punto 3D según la relación:

$$disparity = x - x' = \frac{Bf}{Z} \quad (7)$$

donde x y x' son las coordenadas del pixel en cada imagen, B es la distancia entre la cámara izquierda y derecha (*baseline*), f es a distancia focal de las cámaras y Z es la distancia desde la cámara estéreo hasta el punto 3D. Esta relación puede verse en la Figura 7.

En la Figura 8 se puede ver el mapa de disparidad correspondiente a las imágenes estudiadas en los incisos anteriores.

8. **Reconstrucción 3D Densa** Para realizar la reconstrucción densa en 3D de la escena vista en el par de imágenes estéreo se utiliza un algoritmo que proyecta los puntos del mapa disparidad calculado antes, y la matriz Q que permite la transformación entre disparidad y distancia:

$$Q = \begin{pmatrix} 1 & 0 & 0 & c_x \\ 0 & 1 & 0 & c_y \\ 0 & 0 & 0 & f \\ 0 & 0 & \frac{1}{B} & \frac{c_x - c'_x}{B} \end{pmatrix} \quad (8)$$

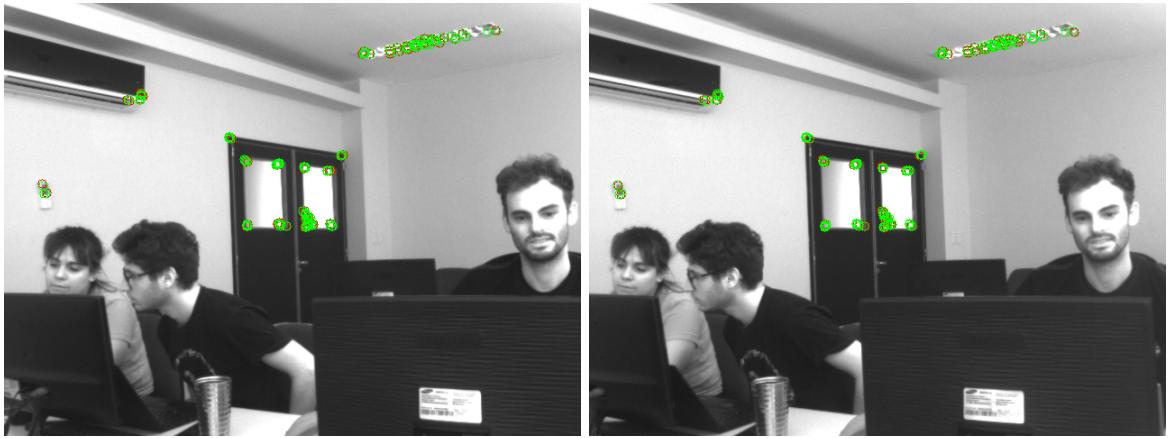


Figure 5: Puntos detectados con ORB en la imagen estéreo respectivamente (verde) y puntos de la otra imagen proyectados mediante la matriz de homografía (rojo).



Figure 6: Puntos *inliers* triangulados a partir del matching de características entre imágenes izquierda y derecha

donde c_x y c_y son las coordenadas del punto principal en la cámara izquierda, c'_x es la coordenada x del punto principal en la cámara derecha, f es la distancia focal y B es la distancia entre ambas cámaras (*baseline*).

En la Figura 9 se puede ver el resultado de generar la reconstrucción 3D densa para el par de imágenes que analizamos previamente. En ésta se pueden distinguir algunos detalles de la imagen estéreo estudiada, como las personas, la puerta y la luminaria. En la Figura ?? puede verse la la misma nube de puntos desde uno de los laterales.

9. Estimación de pose estéreo

Buscamos la matriz de rotación R y el vector de traslación t relativos de la cámara izquierda a la cámara derecha. Para eso buscamos primero la matriz esencial, calculada mediante un algoritmo de RANSAC que selecciona aleatoriamente pares de puntos correspondidos entre ambas imágenes, de forma similar al cálculo de la matriz homogénea explicada anteriormente. Luego, a partir de los puntos *inliers* calculados anteriormente de cada imagen, la matriz epipolar y los coeficientes de distorsión de cada cámara se realiza una estimación de R y t . En la Figura 11 se puede ver la posición relativa entre un par de cámaras estéreo, correspondiente al par de imágenes que se analizó previamente en el trabajo.

10. Seguimiento de trayectoria

Buscamos la trayectoria que recorre la cámara izquierda siguiendo la secuencia de imágenes provistas desde

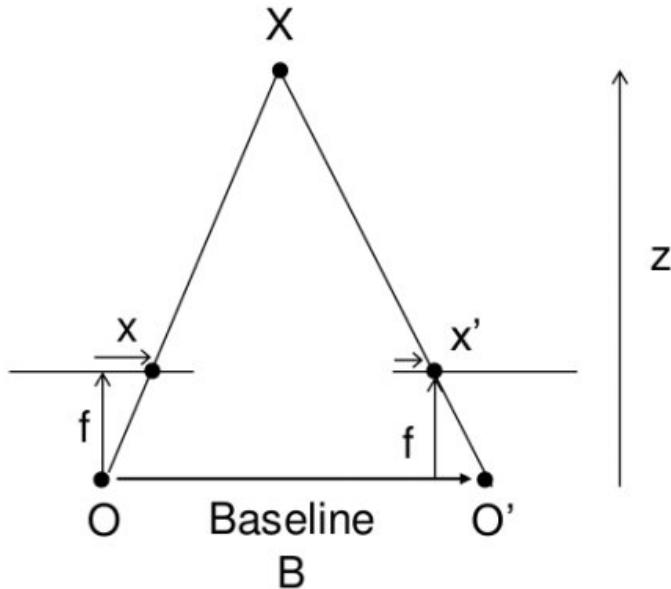


Figure 7: Esquema de disparidad y distancias. Extraída de https://docs.opencv.org/4.5.4/dd/d53/tutorial_py_depthmap.html

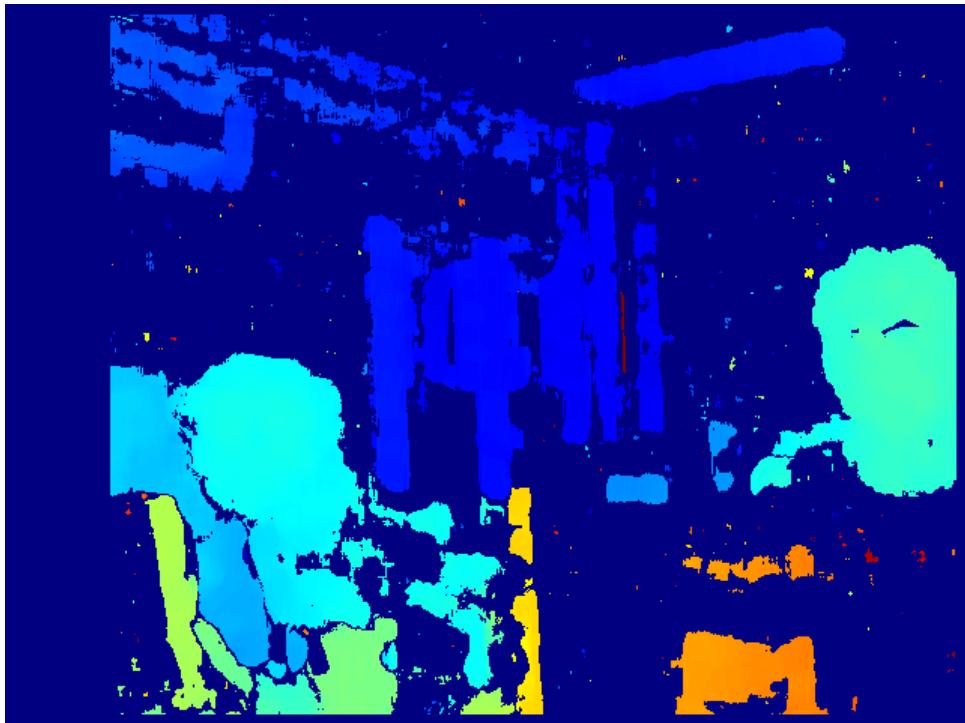


Figure 8: Mapa de disparidad entre las imágenes izquierda y derecha estudiadas.

el rosbag. Para lograr esto se implementó la siguiente idea: en primer lugar se posicionó la primer imagen estéreo en una posición fija predeterminada, quedando la pose de la cámara izquierda P_0 posicionada en el origen de coordenadas, de forma que la posición de la derecha queda determinada a partir de la matriz P_{right} obtenida en la etapa de calibración. Luego se proyectan los puntos correspondidos al espacio.

Para la siguiente imagen estéreo se procede a buscar puntos característicos compartidos entre la cámara izquierda y las dos cámaras posicionadas anteriormente, y se identifican cuáles de los puntos 3D del espacio son vistos desde la nueva imagen izquierda. De esta forma, dado que tenemos la posición de los puntos 3D con una escala ya definida por la separación conocida entre las cámaras izquierda y derecha, entonces, ahora podemos utilizar un algoritmo para reproyectar estos puntos sobre la nueva imagen izquierda y encontrar su pose $P_{0,1}$ relativa a la cámara izquierda anterior, respetando la escala definida por las imágenes anteriores. Como la pose de la cámara izquierda anterior está en el origen, entonces la pose de la nueva cámara izquierda queda como $P_1 = P_{0,1}$.

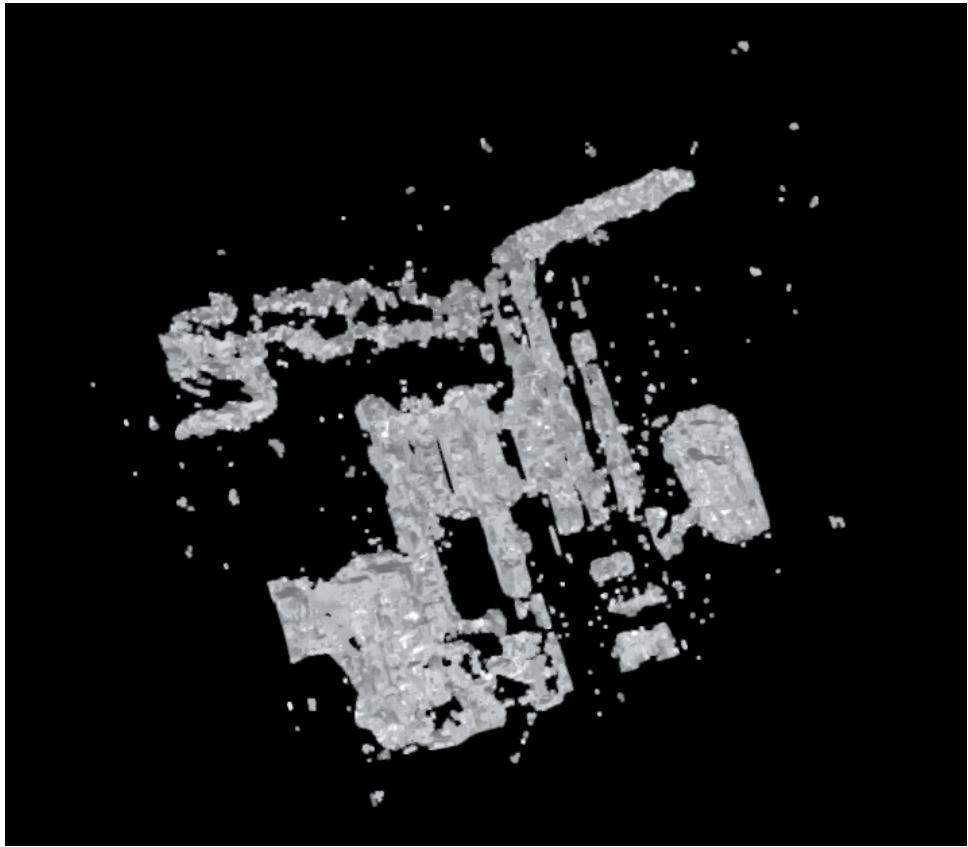


Figure 9: Reconstrucción 3D densa del par de imágenes estudiadas.



Figure 10: Reconstrucción 3D densa del par de imágenes estudiadas.

Luego, se realiza el mismo procedimiento para el siguiente par de imágenes estéreo: Se supone que la cámara izquierda (cuya pose estimamos como P_1) está en el origen, y por lo tanto la cámara derecha tendrá su pose dada por P_{right} . Se realiza la misma técnica para encontrar la posición de la nueva cámara izquierda respecto de la pose de la anterior, obteniendo $P_{1,2}$. Para encontrar P_2 , usamos la pose de P_1 y le concatenamos el movimiento relativo de calculado:

$$R_2 = R_1 R_{1,2} \quad (9)$$

$$t_2 = t_1 + R_1, t_{1,2} \quad (10)$$

$$P_2 = \begin{pmatrix} R_2 & t_2 \\ 0 & 1 \end{pmatrix} = P_1 P_{1,2} \quad (11)$$

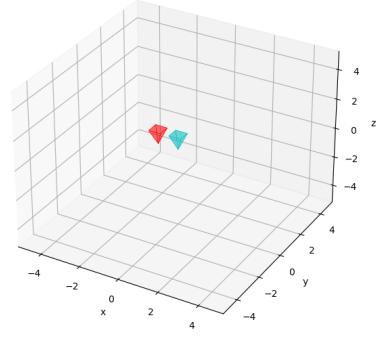


Figure 11: Pose relativa entre la cámara izquierda (azul) y la cámara derecha (roja).

Este procedimiento se realiza para todo nuevo par de imágenes, usando puntos 3D proyectados desde la imagen estéreo inmediatamente anterior. De forma general:

$$R_i = R_{i-1} R_{i-1,i} \quad (12)$$

$$t_i = t_{i-1} + R_{i-1} t_{i-1,i} \quad (13)$$

$$P_i = \begin{pmatrix} R_i & t_i \\ 0 & 1 \end{pmatrix} = P_{i-1} P_{i-1,i} \quad (14)$$

En la Figura 12 se puede observar un ejemplo de correspondencias entre las tres imágenes que se utilizan para determinar la posición de una de ellas, y en la Figura 13 puede observarse la trayectoria estimada con el método empleado. Puede observarse allí que hay entre algunos frames ocurren saltos no deseados en la pose, por lo que podemos concluir que la estimación así realizada no resulta buena para todo par de imágenes estéreo consecutivas.

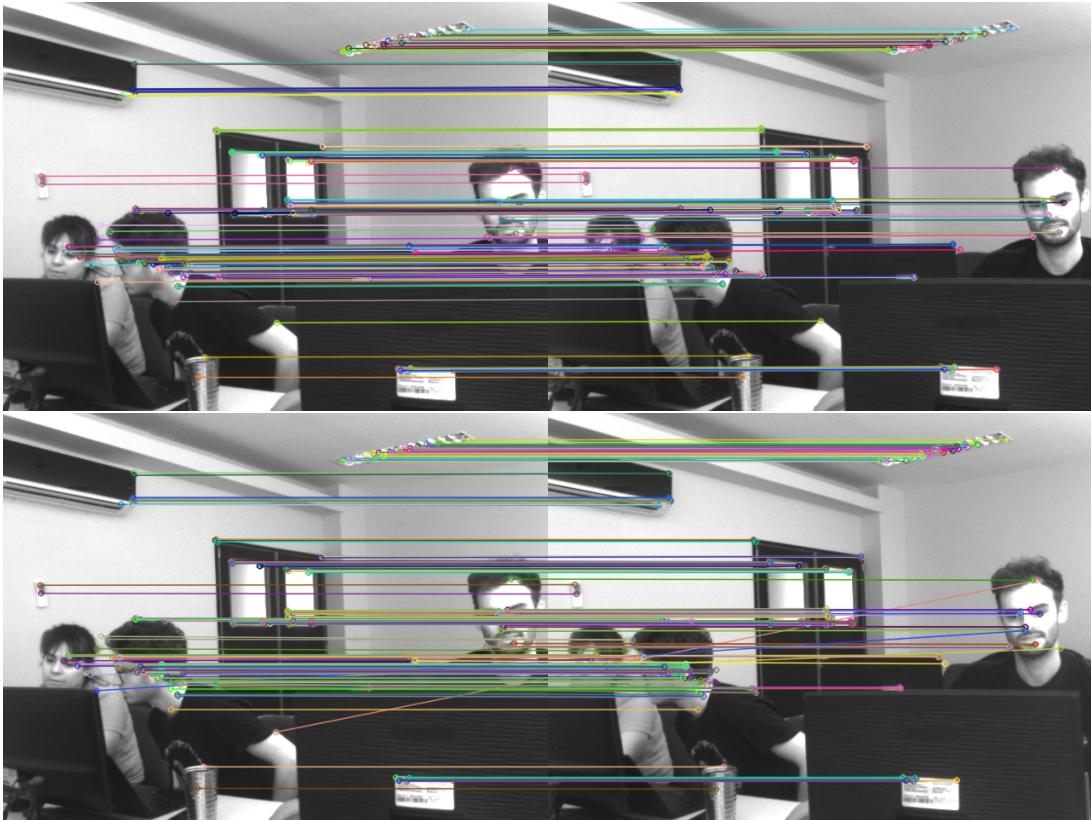


Figure 12: Arriba: correspondencias entre imágenes izquierda y derecha de un mismo instante $i - 1$; abajo: correspondencias entre imágenes izquierdas en instantes sucesivos $i - 1$ e i .



Figure 13: Estimación de la trayectoria de la cámara izquierda.

Entre las posibles mejoras a realizar para lograr una mejor estimación de la trayectoria están: generar un mapa de puntos global en vez de local, y trackear qué puntos se pueden ver desde cada par de imágenes estereo sucesivas, de forma de otorgarle mayor robustez a la nube de puntos utilizada para reproyectar sobre la imagen a la que se le desea estimar la pose, y utilizar herramientas de minimización tal como *bundle adjustment* para minimizar los errores de estimación de poses y de puntos 3D en el espacio.