Exercise 3

Deadline: 16. December, 2020

Please upload your solutions to to your personal **Submissions** folder in OLAT.
Use the **Latex template** provided in Exercise_0 for the report.
Hand-written results will not be accepted!

Theory

Assume two fully calibrated cameras with intrinsic calibration matrices $\mathbf{K}_i$ and extrinsics $\mathbf{R}_i$ and $\mathbf{t}_i$, $i = 0, 1$. The projections can be assumed to be free of any distortions.

(a) Given an image point $\mathbf{x}_0$ in the first view, how does this constrain the position of the corresponding point $\mathbf{x}_1$ in the second image?

(b) Assume corresponding image points $\mathbf{x}_0 \leftrightarrow \mathbf{x}_1$ are given. Describe in words how the 3D world point $\mathbf{X}$ can be computed from its projected image points $\mathbf{x}_i = \mathbf{K}_i \; [\mathbf{R}_i|\mathbf{t}_i] \begin{pmatrix} \mathbf{X} \\ 1 \end{pmatrix}$ (only the idea, no mathematical derivation).

(c) How can the epipoles be computed for the cameras? How are epipolar lines and the epipoles related?

(d) How can the fundamental matrix be computed if no calibration is given (only the idea, no mathematical derivation)?

(e) How can the fundamental matrix be computed if the calibration (intrinsics and pose) is given?

(f) Name a fundamental problem of the matching technique used in the pracical part. When does it make sense to match features this way?

Practical Part

The goal of this exercise is to match detected features and reconstruct the original 3D structure. For this task, two calibrated cameras are given (intrinsics $\mathbf{K}_0$, $\mathbf{K}_1$ and extrinsics $\mathbf{R}_1$, $\mathbf{t}_1$ are provided in `data.mat`). The first camera coincides with the world coordinate system, thus no rotation ($\mathbf{R}_0 = \mathbf{I}$) and no translation ($\mathbf{t}_0 = \mathbf{0}$) can be assumed. The features originated from a chessboard seen by both cameras (`Camera00.jpg` and `Camera01.jpg`) and are also provided in `data.mat` (*cornersCam0*, *cornersCam1*). To load the information in `data.mat` you may run `dict = io.loadmat('./data/data.mat')` (io is scipy.io). Then, *dict* is a dictionary and the necessary information can be accessed via the following keys: '*K_0*', '*K_1*', '*R_1*', '*t_1*', '*cornersCam0*', '*cornersCam1*'.

1. Feature Matching Using the Epipolar Constraint

A corresponding pair of checkerboard corners $\mathbf{x}_0 \leftrightarrow \mathbf{x}_1$ satisfies $\mathbf{x}_0\mathbf{F}\mathbf{x}_1 = \mathbf{x}_0\mathbf{l}_0 = 0$. Use this property to implement a simple matching algorithm by following the steps in this exercise.

(a) Compute the fundamental matrix $\mathbf{F}$ and print it to the console.

(b) For each checkerboard feature in camera image 0 (`Camera00.jpg`), compute the corresponding epipolar line $\mathbf{l}_1 = (a, b, c)^\mathsf{T}$ in the image of camera 1 (`Camera01.jpg`).

(c) Draw each epipolar line computed above. It should intersect the corresponding chessboard corner in the image of camera 1. Note that the image resolution is 4752×3168 pixels. This might be useful to define valid image borders for drawing a line. Save the image with the epipolar lines to `epilines.jpg`.

(d) Compute the matching feature as the one in *cornersCam1* with minimal algebraic distance to $\mathbf{l}_0$.

(e) Create a new image with camera 0's image on top and camera 1's image on the bottom (is should be a *stack* with image 0 on top). Connect corresponding points between the images with lines. Use random line colors and do not forget the y-displacement for the correspondences in the lower image. Save this image to `matches.jpg` and show it in your report.

Remark: The cross-product matrix of a vector $a = (x, y, z)^\mathsf{T}$ is given by

$$[a]_\times = \begin{bmatrix} 0 & -z & y \\ z & 0 & -x \\ -y & x & 0 \end{bmatrix}$$

2. Structure Reconstruction

Use the correspondences to triangulate the 3D positions of the corners of the chessboards in the world coordinate system. Use the method that was introduced in the lecture. Each camera image yields 2 linearly independent equations. Thus, an over-determined system with 4 equations and 3 unknowns for each correspondence has to be solved. Plot your reconstructed chessboard points in 3D space. Therefore, it is recommended to use `matplotlib` library. Also visualize the optical center and the optical axis (the z-axis) of each camera in the world coordinate system.

Remark:
Make sure your code executes the tasks above sequentially by simply calling `python main.py` (include `data.mat` alongside `main.py` in your `.zip` file).

Good Luck!