CS280 - Computer Vision - Spring 2023
Homework 2

Due: Thursday, April 6, 2023, 11:59pm

1 Multi-view Reconstruction

In this problem, we will see how we can recover the 3D shape of an object given its image from two different viewpoints. In particular, given 2D point correspondences in the 2 images, we will estimate their 3D position in the scene and also estimate the camera positions and orientations.

1.1 Camera Models

Assume $OXYZ$ is the global coordinate system in the 3D scene, while $O_cX_cY_cZ_c$ is the camera coordinate system. Note that, in general these two systems may not be identical. Let $\widetilde{\mathbf{X}}$ be a (non-homogeneous) point in the scene (in global coordinates), and $\widetilde{\mathbf{X}}_{cam}$ be the coordinates for the same point in the camera coordinate system. It holds that

$$\widetilde{\mathbf{X}}_{cam} = R(\widetilde{\mathbf{X}} - \widetilde{\mathbf{C}}) \tag{1}$$

where $\widetilde{\mathbf{C}}$ is the center of the camera in the global coordinate system and R is a 3×3 rotation matrix which describes the orientation of the camera coordinate system with respect to the global coordinate system.

If $\mathbf{X}$ is the 3D point in homogeneous coordinates (4th coordinate is 1) and $\mathbf{x}$ is the corresponding 2D point in the image, then

$$\mathbf{x} = KR[I| - \widetilde{\mathbf{C}}]\mathbf{X} \tag{2}$$

where K is the camera's calibration matrix which is defined by the intrinsic parameters of the camera. Refer Lecture-17. We can use the above equation to describe the camera matrix, P: $P = K[R|\mathbf{t}]$, where $\mathbf{t} = -R\widetilde{\mathbf{C}}$.

In case of two cameras, it is often easier to assume the global coordinate system to be same as the first camera's coordinate system. In that case, $P_1 = K_1[I|\mathbf{0}]$ and $P_2 = K_2[R|\mathbf{t}]$.

1.2 Fundamental Matrix

The fundamental matrix F is a 3×3 matrix which relates corresponding points in stereo images. Assume $\mathbf{x}_1 \leftrightarrow \mathbf{x}_2$ are corresponding points in an image pair, then the following relation holds:

$$\mathbf{x}_2^t F \mathbf{x}_1 = 0 \tag{3}$$

Note that the fundamental matrix is of rank 2. (Why? – because Essential Matrix E is rank 2 and F is uncalibrated version of E)

1.3 The Essential Matrix

The essential matrix E relates calibrated corresponding image points, and it can be defined from the fundamental matrix as follows:

$$E = K_2^t F K_1 \tag{4}$$

If $P_1 = K_1[I|\mathbf{0}]$ and $P_2 = K_2[R|\mathbf{t}]$, the essential matrix can also be written as

$$E = [\mathbf{t}]_x R \tag{5}$$

where $[\mathbf{t}]_x$ is the representation of the cross product with $\mathbf{t}$. We can also show that the essential matrix is of rank 2 and has two identical real singular values while the third one is zero (as an optional exercise, you can try to prove it!). For more details refer to Chapter 7 in [1].

Eqn. 5 also tells us that knowing the essential matrix can allow us to estimate the camera orientation and center location as is described below.

1.4 Algorithms

In this section, we describe algorithms that are used to solve various problems related to the task of 3D reconstruction.

1.4.1 Eight-Point Algorithm

This algorithm fits the fundamental matrix defined by corresponding points $\mathbf{x}_1 \leftrightarrow \mathbf{x}_2$ in an image pair. In particular, assume points $\mathbf{x}_1^{(i)} = (x_1^{(i)}, y_1^{(i)})$ in the first image corresponds to points $\mathbf{x}_2^{(i)} = (x_2^{(i)}, y_2^{(i)})$ in the second image for $i = 1, ..., N$. Note that, $N \geq 8$, we need at least 8 corresponding points.

Normalization. Usually, the point correspondences are not completely accurate. To avoid numerical errors due to noise, we can normalize the points in the image, by translating them by μ (so that the mean of the points is at the origin), and scaling them by σ (so that the mean distance from the origin is a constant (e.g. $\sqrt{2}$)). Assume that the linear transformation described above is represented by the matrices T_1 and T_2 for the two images respectively.

Optimization. Eq. 3 can be rewritten equivalently as $Af = 0$, where f is formed from the entries of F stacked to a 9-vector row-wise and A is a $N \times 9$ dimensional matrix. In particular, the i-th row of A is equal to

$$A_i = [x_1^{(i)}x_2^{(i)} \quad y_1^{(i)}x_2^{(i)} \quad x_2^{(i)} \quad x_1^{(i)}y_2^{(i)} \quad y_i^{(i)}y_2^{(i)} \quad y_2^{(i)} \quad x_1^{(i)} \quad y_1^{(i)} \quad 1] \tag{6}$$

where the point coordinates have already been normalized, i.e. $\mathbf{x} \leftarrow T\mathbf{x}$.

The linear system$Af = 0$ has an exact non-zero solution if $rank(A) = 8$, which would happens if the point correspondences are exact. But usually due to noise, $rank(A) > 8$. In that case, there is no exact solution to the linear system and an approximate solution has to be found.

An approximate solution can be found by solving the following optimization problem

$$\min_f \ ||Af||_2 \quad s.t. \ ||f||_2 = 1 \tag{7}$$

which can be solved by using the singular value decomposition of matrix A.

The solution F^* found by the approximately solving the problem $Af = 0$, does not guarantee that the fundamental matrix will be of rank 2. We can enforce this constraint, by solving another optimization problem:

$$\min_F \ ||F - F^*||_F \quad s.t. \ rank(F) = 2 \tag{8}$$

Again, this problem can be solved using the singular value decomposition of F^*. In particular, if $F^* = USV^t$, where $S = diag(s_1, s_2, s_3)$ and $s_1 \geq s_2 \geq s_3$ then $F = U\hat{S}V^t$, where $\hat{S} = diag(s_1, s_2, 0)$, is the matrix that solves the above optimization problem.

Denormalization. After enforcing the rank-2 constraint, we need to remove the normalization such that the fundamental matrix corresponds to points in the actual 2D image space, i.e. $F \leftarrow T_2^t F T_1$.

1.4.2 Estimating Extrinsic Camera Parameters from the Essential Matrix

Another problem is to estimate the extrinsic camera parameters, i.e. $R, \mathbf{t}$ for the second camera, when only the essential matrix E is known.

The operation $[\mathbf{t}]_x$ can also be written as

$$[\mathbf{t}]_x = SZR_{90^o}S^t \tag{9}$$

where $S = [\mathbf{s}_0 \ \ \mathbf{s}_1 \ \ \mathbf{t}]$ is an orthogonal matrix, $Z = diag(1, 1, 0)$ and R_{90^o} is the rotation matrix for rotation angle $\theta = 90^o$.

From Eq. 5

$$E = [\mathbf{t}]_x R = SZR_{90^o}S^tR = U\Sigma V^t \tag{10}$$

Since we know that $\Sigma = diag(1, 1, 0)$ (remember in homogeneous coordinates we define everything up to a multiplication factor), it holds that $U = S$ and $V = R^tUR^t_{90^o}$. Thus, $\mathbf{t}$ is the third left singular vector of E.

Thus, the direction of $\mathbf{t}$ as well as R can be determined by the SVD analysis of E. Notice that there is no way we can determine the actual norm of the vector $\mathbf{t}$ from just image point correspondences. Absolute values can only be determined if we have a known reference distance in the scene. In addition, the signs of both $\mathbf{t}$ and R also can not be determined from SVD decomposition. Therefore, this algorithm generates numerous candidate positions and orientations for the camera. You can also see Ch-7 of [1] for detailed explanation.

Different combinations of $\mathbf{t}$ and R result in different camera matrix $P_2 = K_2[R|\mathbf{t}]$. After triangulation (3D reconstruction of the point cloud), the combination that gives the largest number of points in front of the image planes is selected.

1.4.3 Triangulation

Triangulation refers to the estimation of the 3D position of a point when the corresponding points in the two image planes are given as well as the parameters of the camera. In particular, assume $\mathbf{x}_1 \leftrightarrow \mathbf{x}_2$ are two corresponding points in the image plane and the camera matrices are P_1 and P_2 respectively. We want to find the 3D point $\mathbf{X} = (X, Y, Z, W)$ such that

$$\mathbf{x}_1 = P_1\mathbf{X} \quad and \quad \mathbf{x}_2 = P_2\mathbf{X} \tag{11}$$

Eq. 11 can also be written as:

$$x_j = \frac{P^{(j)}_{11}X + P^{(j)}_{12}Y + P^{(j)}_{13}Z + P^{(j)}_{14}W}{P^{(j)}_{31}X + P^{(j)}_{32}Y + P^{(j)}_{33}Z + P^{(j)}_{34}W} \tag{12}$$

$$y_j = \frac{P^{(j)}_{21}X + P^{(j)}_{22}Y + P^{(j)}_{23}Z + P^{(j)}_{24}W}{P^{(j)}_{31}X + P^{(j)}_{32}Y + P^{(j)}_{33}Z + P^{(j)}_{34}W} \tag{13}$$

where $j = 1, 2$ are the two camera indices and $\mathbf{x}_j = (x_j, y_j)$.

The above system of non linear equations can be turned into a linear system, if both equations are multiplied by the denominators. The system will be homogeneous if the point in 3D is represented in homogeneous coordinates and thus can be solved via SVD. Alternatively, if we set $W = 1$ then the system is no longer homogenous and can be solved using least squares, i.e. by taking pseudo inverse by reducing it to $A\mathbf{X} = b$ where A and b are appropriate matrices.

1.5 3D reconstruction

In this assignment, we give you two sets: `house` and `library`. Since the intrinsic parameters of a camera are nowadays known and stored when a picture is taken, we provide the calibration matrices K_1 and K_2 for the pair of images in each example. In general, these parameters also need to be estimated via calibration when they are unknown. We also give you corresponding points for both pairs of images. In general, the corresponding points are found by detecting interest points in images and then comparing their descriptors across images to find matches.

Given a pair of images and their corresponding points as well as the intrinsic parameters for the two cameras, you will have to estimate the 3D positions of the points as well as the camera matrices. In particular, initially you will have to compute the fundamental matrix. You can implement the 8-point algorithm described above or any other algorithm you wish. Subsequently, you will estimate the extrinisic camera parameters of the second camera from the essential matrix. From all the possible combination of parameters, you will keep the one that results in most points in front of the two image planes. Last, you will plot the 3D points as well as the two camera centers. It is up to you how you want to show the point cloud.

1.5.1 Starter Code and Data

`starter_code/code/reconstruct_3d.m` contains the starter code. This function is the main function and should output the points in the 3D space as well as the camera matrices. The input argument to this function is either `'house'` or `'library'` to select which dataset you want to use. `starter_code/data` contains the image pairs and camera matrices for the `house` and the `library`.

You will write the following four functions:

1. `fundamental_matrix.m`: This function computes the fundamental matrix F given the data provided. It should return F as well as the residual *res_err*. The residual is defined as the mean squared distance between the points in the two images and the corresponding epipolar lines. More precisely, if the fundamental matrix is F and the 2 corresponding points are x_1 and x_2, then the epipolar line corresponding to the point x_2 in the first image is given by $F\mathbf{x}_2$. So the distance between $\mathbf{x}_1$ and $F\mathbf{x}_2$ is

$$d_{1->2} = \frac{|\mathbf{x}_1^t F \mathbf{x}_2|}{||F\mathbf{x}_2||}$$

The distance between $\mathbf{x}_2$ and $F\mathbf{x}_1$ is similar. So the residual is

$$\text{residual} = \frac{1}{2n} \sum_{\text{match}} d_{1->2}^2 + d_{2->1}^2$$

Is this what you are directly optimizing using SVD when solving the homogeneous system? If yes, explain. If no, how does the objective relate to the residual? **Deliverables:** Your answer in your report should include - description of how you calculate F, the value of fundamental matrix, residuals that you get for the 2 sample inputs and answer of above questions. Again, if you like, you may include a code snippet if you think it makes your explanation any easier to understand.

2. `find_rotation_translation.m`: This function estimates the extrinsic parameters of the second camera. The function should return a cell array R of all the possible rotation matrices and a cell array t with all the possible translation vectors. **Deliverables:** Your answer in your report should include - all possible choices for t and R for the 2 sample inputs.

3. `find_3d_points.m`: This function reconstructs the 3D point cloud. In particular, it returns a $N \times 3$ array called *points*, where N is the number of the corresponding matches. It also returns the reconstruction error *rec_err*, which is defined as the mean distance between the 2D points and the projected 3D points in the two images. **Deliverables:** Your answer in your report should include - exactly how you calculate the reconstruction error, and reconstruction error for the 2 sample inputs

4. `plot_3d.m`: This function plots the 3D points in a 3D plot and displays the camera centers for both cameras. Plotting in 3D can be done using the `plot3` command. A plot of the point cloud and the camera centers is enough for the purpose of the assignment. Note in this part, use that combination of R and $\mathbf{t}$ obtained in part-2 which gives the largest number of points in front of the image planes is selected. **Deliverables:** Your answer in your report should include - a plot for each of the 2 sample inputs from a reasonable viewpoint after selecting appropriate R and $\mathbf{t}$

2 Instructions

1. You're welcome to do the assignment in Python/NumPy instead of Matlab after re-implementing the (relatively simple) started code in Python/NumPy. If you do this, please maintain the same directory structure and implement all the functions in `<filename>.py` instead of `<filename>.m`.

2. Please submit the assignment using gradescope. Upload the following files:

   (a) `A PDF file`. The top of the first page should contain your name, student ID, and date of submission. The file should contain answers to all questions, and supporting images. Questions should be answered in order. Each problem should be on a new page.

   (b) `A tar/zip file`. The file should be a folder with subdirectories for each coding problem, containing any code you wrote (in `Matlab` or `Python`) for the assignment.

3. The HW is due on Due: Thursday, April 6, 2023, 11:59pm.

4. This assignment should be done individually.

References

[1] Richard Szeliski. *Computer vision: algorithms and applications*. Springer, 2011.