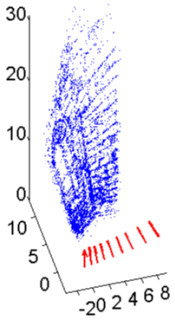
MVG Assignment 2
Calibration and DLT

Instructions

- Due date: May 4 2021.
- Submission link: `https://www.dropbox.com/request/qM0g5gbVlinoh70Sn7Y3`
- You should submit a zip file containing: A pdf file with all complete solutions and the code with a main script which calls the required scripts/functions for each computer exercise.
- The report should be written individually, however you are encouraged to work together.
- We highly recommend typing your solutions in Latex.
- Notice that in the end of this exercise you are given a list of useful matlab commands for all the computer exercises.

(a) An image from the scene



(b) The reconstruction of the scene

Figure 1

Calibrated vs. Uncalibrated Reconstruction

Exercise 1

Show that when estimating structure and motion (3D points and cameras) simultaneously, under the assumption of uncalibrated cameras, there is always an unknown projective transformation of 3D space that cannot be determined using only image projections. That is, if $\mathbf{X}$ is the estimated 3D-points, then show that a new solution can always be obtained from $T\mathbf{X}$ for any projective transformation T of 3D space. (Hint: Look at the camera equations.)

Computer Exercise 1

Figure 1 shows an image of a scene and a reconstruction using uncalibrated cameras. The file `compEx1data.mat` contains the 3D points of the reconstruction $\mathbf{X}$, the camera matrices P, the image points x and the filenames `imfiles` of the images. Here $\mathbf{X}$ is a 4×9471 matrix containing the homogeneous coordinates for all 3D points, $\mathbf{x}\{i\}$ is a 3×9471 matrix containing the homogeneous coordinates of the image points seen in image i (NaN means that the point has not been detected in this image). $P\{i\}$ contains the camera matrix of image i and `imfiles{i}` contains the name of that image.

a.

Plot the 3D points of the reconstruction, using the file `plotcams.m` to plot the cameras in the same figure. Does this look like a reasonable reconstruction? (Don't forget to use axis equal otherwise you may get additional distortion.)

b.

Using the first image and the first camera:
Project the 3D points onto the camera.
Plot the image, the projected points, and the image points in the same figure. Do the projections appear to be close to the corresponding image points? (make sure to divide by the third coordinate)

c.

You are given in `compEx1data.mat` two projective transformations:

$$T_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1/10 & 1/10 & 0 & 1 \end{bmatrix}, T_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1/16 & 1/16 & 0 & 1 \end{bmatrix} \tag{1}$$

Modify all the 3D points and cameras using T_1, T_2 (as in Exercise 1) so that two *new projective solutions* are obtained.
Plot all the 3D points and cameras for each of the solutions. Do any of them seem reasonable? (Don't forget to divide the points by the fourth coordinate before plotting using the `pflat.m` function.)

d.

Project the new 3D points you got from using T_1 into the first camera. **Plot** the image, the projected points, and the image points in the same figure, similarly to **b**. Do the projections appear to have changed?

For the report: The figures and short answers.

Exercise 2

When using calibrated cameras we do not get the distortions as in computer exercise 1. In exercise 1 you showed that if $\mathbf{X}$ is the estimated 3D-points then $T\mathbf{X}$ is also a solution for any projective transformation T of 3D space. **What is** the corresponding statement for calibrated cameras? (What are the constraints on this T?)

Camera Calibration

Exercise 3

Suppose that a camera has got the inner parameters:

$$K = \begin{bmatrix} f & 0 & x_0 \\ 0 & f & y_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2}$$

The inverse of K is:

$$K^{-1} = \begin{bmatrix} 1/f & 0 & -x_0/f \\ 0 & 1/f & -y_0/f \\ 0 & 0 & 1 \end{bmatrix} \tag{3}$$

And K^{-1} can be factorized into

$$K^{-1} = \underbrace{\begin{bmatrix} 1/f & 0 & 0 \\ 0 & 1/f & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} 1 & 0 & -x_0/f \\ 0 & 1 & -y_0/f \\ 0 & 0 & 1 \end{bmatrix}}_{B} \tag{4}$$

What is the geometric interpretation of the the transformations A and B?
When normalizing the image points of a camera with known inner parameters we apply the transformation K^{-1}. **What** is the interpretation of this operation? **Where** does the principal point (x_0, y_0) end up? And **where** does a point with distance f to the principal point end up?
Suppose that for a camera with resolution 640×480 pixels we have the inner parameters:

$$K = \begin{bmatrix} 320 & 0 & 320 \\ 0 & 320 & 240 \\ 0 & 0 & 1 \end{bmatrix} \tag{5}$$

Normalize the points $(0, 240)^T, (640, 240)^T$.
Show that the camera $K \begin{bmatrix} R & \mathbf{t} \end{bmatrix}$ and the corresponding normalized version $\begin{bmatrix} R & \mathbf{t} \end{bmatrix}$ have the same camera center and principal axis.

Exercise 4

Consider the camera

$$P = \begin{bmatrix} 1000 & -250 & 250\sqrt{3} & 500 \\ 0 & 500(\sqrt{3} - \frac{1}{2}) & 500(1 + \frac{\sqrt{3}}{2}) & 500 \\ 0 & -\frac{1}{2} & \frac{\sqrt{3}}{2} & 1 \end{bmatrix} \tag{6}$$

Find K^{-1} given that the focal length is 1000, the principal point is $(500, 500)$, the skew is zero and the aspect ratio is one.

Normalize the camera P.

If the images are of size 1000×1000 pixels **normalize** the corners of the image: $(0, 0)$, $(0, 1000)$, $(1000, 0)$, $(1000, 1000)$ and the center $(500, 500)$.

RQ Factorization and Computation of K

Exercise 5

a.

Consider the upper triangular matrix

$$K = \begin{bmatrix} a & b & c \\ 0 & d & e \\ 0 & 0 & f \end{bmatrix} \tag{7}$$

where the diagonal elements of K are positive. Verify by matrix multiplication that

$$KR = \begin{bmatrix} aR_1^T + bR_2^T + cR_3^T \\ dR_2^T + eR_3^T \\ fR_3^T \end{bmatrix} \text{ Where } R = \begin{bmatrix} R_1^T \\ R_2^T \\ R_3^T \end{bmatrix} \tag{8}$$

b.

Given the uncalibrated camera matrix:

$$P = \begin{bmatrix} \frac{800}{\sqrt{2}} & 0 & \frac{2400}{\sqrt{2}} & 4000 \\ -\frac{700}{\sqrt{2}} & 1400 & \frac{700}{\sqrt{2}} & 4900 \\ -\frac{1}{\sqrt{2}} & 0 & \frac{1}{\sqrt{2}} & 3 \end{bmatrix} \tag{9}$$

Find R_3 and f. (Hint: R is a rotation matrix)

Computer Exercise 2

The file `rq.m` computes the RQ factorization of a matrix. **Compute** K for the first camera, and then for that camera with the projective transformations T_1 and T_2. **Do they** represent the same transformation? Two matrices can differ by a scale factor and still give the same transformation so make sure you normalize the matrices by: `K=K./K(3,3)`.

Direct Linear Transformation (DLT)

Exercise 6

The linear least squares system:

$$\min_{\mathbf{v}} \|M\mathbf{v}\|^2 \tag{10}$$

always has the minimum value 0. In order to remove this solution, the DLT algorithm uses the least squares system:

$$\min_{\|\mathbf{v}\|^2=1} \|M\mathbf{v}\|^2 \tag{11}$$

a.

Assume M has the singular value decomposition $M = U\Sigma V^T$. **Show** that:

$$\|M\mathbf{v}\|^2 = \|\Sigma V^T\mathbf{v}\|^2 \tag{12}$$

and **show** that:

$$\|V^T\mathbf{v}\|^2 = 1 \text{ if } \|\mathbf{v}\|^2 = 1 \tag{13}$$

b.

If we let $\tilde{\mathbf{v}} = V^T\mathbf{v}$ then we get the new problem

$$\min_{\|\tilde{\mathbf{v}}\|^2=1} \|\Sigma\tilde{\mathbf{v}}\|^2 \tag{14}$$

Explain why this problem gives the same minimal value as (11). **How** can you obtain a solution to the first problem from the second? **Why** are there always at least two solutions to this problem?
If $rank(M) < n$ that is, the last singular value n is zero, **what** is the optimal $\tilde{\mathbf{v}}$? (Hint: rewrite (14) as a sum).

(a) `cube1.JPG`

(b) `cube2.JPG`

Figure 2

Exercise 7

When using DLT it is often advisable to normalize the points before doing computations. Suppose the image points $\mathbf{x}$ are normalized by the mapping N by:

$$\tilde{\mathbf{x}} \sim N\mathbf{x} \tag{15}$$

and that we compute a camera matrix in the new coordinate system, that is, we obtain a camera P that solves:

$$\tilde{\mathbf{x}} \sim \tilde{P}\mathbf{X} \tag{16}$$

Write a formula for how to use $\tilde{P}$ to compute the camera P that solves the original problem:

$$\mathbf{x} \sim P\mathbf{X} \tag{17}$$

Computer Exercise 3

Figure 2 shows two images `cube1.jpg` and `cube2.jpg` of a scene with a Rubics cube.
The file `compEx3data.mat` contains a point model `Xmodel` of the visible cube sides, the measured projections `x` of the model points in the two images and two variables `startind`, `endind` that can be used for plotting lines on the model surface.
For each of the two images, **normalize** the measured points by applying a transformation N that subtracts the mean of the points and then re-scales the coordinates by the standard deviation in each coordinate.
In order to debug your normalization matrices, **plot** the normalized points $\tilde{x}\{1\}, \tilde{x}\{2\}$ in a new figure. Does it look like the points are centered around $(0, 0)$ with mean distance 1 to $(0, 0)$?

a.

Set up the DLT equations for resectioning, and **solve** the resulting homogeneous least squares system using SVD. Is the smallest eigenvalue close to zero? Is $\|Mv\|$ close to zero?
Extract the entries of the camera from the solution and set up the camera matrix. Make sure that you select the solution where the points are in front of the camera. (If $\mathbf{X}$ has 4th coordinate 1 then the 3rd coordinate of $P\mathbf{X}$ should be positive for $\mathbf{X}$ to be in front of the camera.)

b.

Project the model points into the images. (Before projecting don't forget to transform the camera matrix to the original (unnormalized) coordinate system, as in Exercise 7)
Plot the measured image points in the same figure. Are they close to each other?
Plot the camera centers and viewing directions in the same plot as the 3D model points. Does the result look reasonable?
Compute the inner parameters of the first camera using `rq.m`. How can we know that these are the "true" parameters? **Why** is there no ambiguity as in Exercise 1?

c.

Compute the RMS error:

$$E_{RMS} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\mathbf{x}_i - \tilde{\mathbf{x}}_i)^2} \tag{18}$$

Where $\mathbf{x_i}$ are the coordinates for the measured points and $\tilde{\mathbf{x}}_i$ are the projected model points from the cameras you have found.
Next, **repeat** the resectioning but this time don't normalize the points. (you can simply use $N = I$ as normalization matrix and run the same code again.) **What** is the RMS error now?

For the report: The figures and short answers.

Triangulation using DLT

Computer exercise 4

In the file `cube_matches.mat` you are given an array of matching points `x1` and `x2` that were found using SIFT feature extraction. Using the estimated cameras you've found in computer exercise 3 we will triangulate these points.

a.

Set up the DLT equations for triangulation, and **solve** the homogeneous least squares system. (You will have to do this in a loop, once for each point.)
Project the computed points back into the two images and compare with the corresponding SIFT-points `x1` and `x2`.
Compare with the results you get when you normalize with inner parameters of the cameras, that is, using the inverse of K for normalization. Is there any improvement?

b.

A portion of the SIFT matches are incorrect. Most of the time (but not always) this will result in triangulations with large error. For both images, **compute** the distance between the projected 3D points and the corresponding SIFT points. **Remove** the points for which the error in at least one of the images is larger than 3 pixels.
Plot the remaining 3D points, the cameras and the cube model in the same 3D plot. Can you distinguish the dominant objects (the cups and the paper)?

For the report: The figures and short answers.

Usefull Matlab commands

1

- im = imread(imfiles{i}); % Reads the imagefile with name in imfiles{i}
- visible = isfinite(x{i}(1 ,:)); % Determines which of the points are visible in image i
- plot(x{i}(1, visible), x{i}(2 , visible),'*'); % Plots a '*' at each point coordinate
- plot(xproj(1, visible), xproj (2, visible),'ro'); % Plots a red 'o' at each visible point in xproj
- plot3(X(1 ,:) ,X(2 ,:) ,X(3 ,:) , '.','Markersize ' ,2); % Plots a small '.' at all the 3D points.

3

- mean(x{i}(1:2 ,:) ,2) % Computes the mean value of the 1st and 2nd rows of $x\{i\}$.
- std(x{i}(1:2 ,:) ,0 ,2) % Computes the STD value of the 1st and 2nd rows of $x\{i\}$.
- [U,S,V] = svd(M); % Computes the singular value decomposition of M.
- P = reshape(sol(1:12) ,[4 3])'; % Takes the first 12 entries of sol and row - stacks them in a 3x4 matrix
- plot3([Xmodel(1, startind); Xmodel (1, endind)],...
[Xmodel(2, startind); Xmodel(2, endind)],...
[Xmodel(3, startind); Xmodel (3, endind)], 'b-'); % Plots the lines of the cube model
- RMSE = sqrt(mean((y - yhat).^2));

4

- xproj1 = pflat(P1*X);
- xproj2 = pflat(P2*X); % Project the points X to cameras 1 and 2.
- goodpoints = (sqrt(sum((x1 - xproj1(1:2 ,:)).^2)) < 3 & (sqrt(sum((x2 - xproj2(1:2 ,:)).^2)) < 3
% Finds the points with reprojection error less than 3 pixels in both images
- X = X(:, goodpoints); % Removes points that are not good enough