Computer Vision, Assignment 2
Calibration and DLT

1 Instructions

In this assignment you will study camera calibration and the DLT method. You will solve the resection and triangulation problems using DLT and compute inner parameters using RQ factorization. In addition try out SIFT for feature detection/matching. The data for the assignments is available from the course page `https://canvas.education.lu.se/courses/21938`

The assignment is due at the end (Sunday 12/2 at midnight) of study week 4. The deadline is firm and reports are to be handed in on time through the canvas page. Not everything has to be correct the first time you hand in however you have to present solutions/attempts for each mandatory exercise. It is not ok to hand in blank solutions. (If you have problems solving something you should come to the Q & A-sessions or contact the lecturer by email in good time before the deadline.) In exceptional cases extensions can be offered (due to unforeseen circumstances). In such cases contact the lecturer by email before the deadline (or as soon as possible) for instructions on what to do.

Make sure you answer all questions and provide complete solutions to the exercises. Your solutions should be be submitted as a single pdf-file. Note that it is fine to submit handwritten solutions (by including scans in the pdf-file) as long as they are well structured and readable. After each exercise there is a gray box with instructions on what should be included in the report. In addition, all the code should be submitted as m- and mat-files in a zip-archive. Make sure that your matlab scripts are well commented and can be executed directly, that is, without loading any data, setting parameters etc. Such things should be done in the script.

If you run into problems with any of the exercises you can go to the Q & A-sessions to get help (see course schedule) or send an email to the lecturer (`viktor.larsson@math.lth.se`).

The report should be written individually, however you are encouraged to work together. Keep in mind that everyone is responsible for their own report and should be able to explain all the solutions.

Some exercises are marked as OPTIONAL. You do not have to do these to pass the assignment. However if you submit good solutions to these you will be awarded at most 0.2 bonus points for the home-exam.

2 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.)

For the report: Submit a complete (but short) solution.

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 `X`, the camera matrices `P`, the image points `x` and the filenames `imfiles` of the images. Here `X` is a 4×9471 matrix containing the homogeneous coordinates for all 3D points, `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.

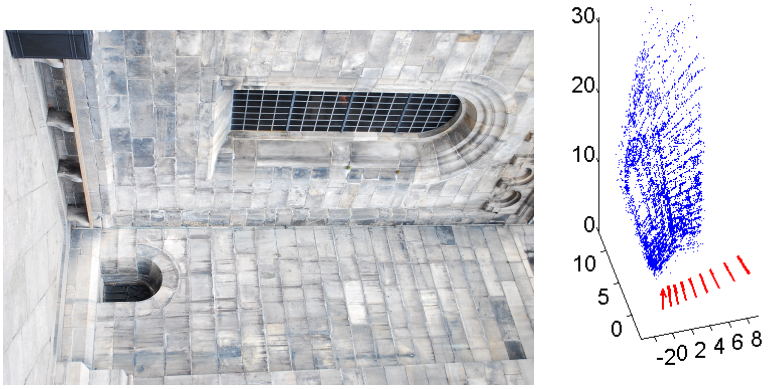

Figure 1: Left: An image of the scene. Right: A projective (uncalibrated) reconstruction.

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.)

Project the 3D points into one of the cameras (only those that have been detected in that 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? (If not: Did you forget to divide by the third coordinate?)

Using the two projective transformations

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

modify the 3D points and cameras (as in Exercise 1) so that two new projective solutions are obtained. Plot the 3D points and cameras in the same figure for each of the solutions. Do any of them seem reasonable? (Don't forget to divide the points by the fourth coordinate before plotting. Feel free to use your `pflat.m` function.)

Project the new 3D points into one of the cameras. Plot the image, the projected points, and the image points in the same figure. Do the projections appear to have changed?

```
Useful matlab commands:

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.
```

For the report: Submit the m-file, the 3D-plots and answers to the questions.

Exercise 2. Explain why we can not get the same projective distortions as in Computer Exercise 1 when we use calibrated cameras. What is the corresponding statement for calibrated cameras to that of Exercise 1?

For the report: Submit a short explanation.

3 Camera Calibration

Exercise 3. Suppose that a camera has got the inner parameters

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

Verify that the inverse of K is

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

and that this matrix can be factorized into

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

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

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

Normalize the points $(0, 240)$, $(640, 240)$.

What is the angle between the viewing rays projecting to these points?

Show that the camera $K[R\ t]$ and the corresponding normalized version $[R\ t]$ have the same camera center and principal axis.

For the report: Complete solution.

Exercise 4. Consider the camera

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

If the focal length is 1000 and the principal point is $(500, 500)$ normalize the camera (the skew is zero and the aspect ratio is one).

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 $(500, 500)$.

For the report: Answers are enough.

4 RQ Factorization and Computation of K

Exercise 5. Consider an upper triangular matrix

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

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

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

If

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

what is R_3 (R is an orthogonal matrix) and f?

Using the second row of the camera matrix, and the fact that $R_3 \perp R_2$, $||R_3|| = ||R_2|| = 1$ compute R_2,d,e. (Hint: If $v = dR_2 + eR_3$ how do you compute the coefficient e?)

Similarly, using the first row compute, R_1,a,b and c. What is the focal length, skew, aspect ratio, and principal point of the camera?

For the report: Complete solution.

Computer Exercise 2. The file `rq.m` computes the RQ factorization of a matrix. Compute K for one camera in each of the solutions you obtained in computer exercise 1. Do they represent the same transformation? (Don't forget that two matrices can differ by a scale factor and still give the same transformation.)

For the report: Submit the K matrices. (Make sure that element $K(3, 3) = 1$ by division; $K = K./K(3, 3)$.)

5 Direct Linear Transformation DLT

Exercise 6. (OPTIONAL.) Show that the linear least squares system

$$\min_v ||Mv||^2 \tag{10}$$

always has the minimum value 0. In the DLT algorithm we use the least squares system

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

to remove the zero solution. Show that if M has the singular value decomposition $M = U\Sigma V^T$ then

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

and

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

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

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

Explain why this problem gives the same minimal value as (11). How can you obtain a solution to the first problem from the second? Also, explain why there are always at least two solutions to this problem.

If $\text{rank}(M) < n$, that is, the last singular value σ_n is zero write down the solution of (11). (The assumption $\text{rank}(M) < n$ is in fact not necessary but the general case is more difficult to prove, see lecture notes.)

For the report: The exercise is optional, but if you want bonus points for the exam submit a complete solution.

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 $\tilde{P}$ that solves

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

How do you compute the camera P that solves the original problem

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

from $\tilde{P}$?

For the report: It is a very simple exercise, just give the formula.

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.

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. Plot the normalized points in a new figure. Does it look like the points are centered around $(0, 0)$ with mean distance 1 to $(0, 0)$?

Set up the DLT equations for resectioning, and solve the resulting homogeneous least squares system (11) using SVD. Is the smallest eigenvalue close to zero? How about $||Mv||$?

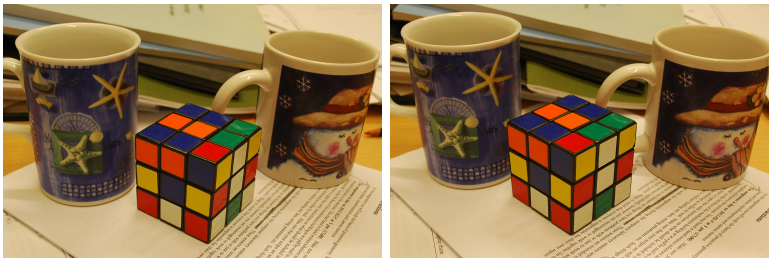

Figure 2: Two images of a scene containing a rubics cube.

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 X has 4th coordinate 1 then the 3rd coordinate of PX should be positive for X to be in front of the camera.)

Project the model points into the images. (Don't forget to transform the camera matrix to the original (un-normalized) coordinate system, as in Exercise 7, before projecting.) Plot the measured image points in the same figure. Are they close to each other? Also plot the camera centers and viewing directions in the same plot as the 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?

The rest of this exercise is OPTIONAL:
Compute the RMS error

$$e_{RMS} = \sqrt{\frac{1}{n}||x_{meas} - x_{proj}||^2}, \tag{18}$$

where x_{meas}, x_{proj} are the Cartesian coordinates for the measured points and projected model points respectively, and n is the number of points.

Next, do everything one more time, but this time don't normalize the points. (The easiest way of doing this is probably to use $N = I$ as normalization matrix and run the same code again.) Is the difference large? Remove all the points except points number $1, 4, 13, 16, 25, 28, 31$ and try again. Is there any difference between when you use normalization and not?

```
Useful matlab commands:

mean(x{i}(1:2,:),2) %Computes the mean value of the 1st and 2nd rows ow x{i}

std(x{i}(1:2,:),0,2) %Standard deviation of the 1st and 2nd rows ow 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 (works only if all points are included)
```

For the report: Submit the m-file, the 3D-plot and the figures. Also specify the inner parameters of the cameras for this case. If you want bonus points for the exam also submit the RMS errors for the normalized and unnormalized cases, with all the points and with only the listed subset of points.

6 Feature Extraction and Matching using SIFT

Computer Exercise 4. In this exercise you will get to try feature extraction using SIFT.

First you will have to download and start VLFeat. Go to `http://www.vlfeat.org/download.html` and extract the binary package to a directory of your choice, e.g. `H:\vlfeat`. Then start Matlab, go the `H:\vlfeat\toolbox` subdirectory and run `vl_setup`. Now you should see the following message:

```
** Welcome to the VLFeat Toolbox **
```

You will now be able to use VLFeat throughout this Matlab session.

First load the images `cube1.jpg` and `cube2.jpg` from Exercise 3.

Compute sift features using vlfeat.

```
[f1 d1] = vl_sift( single(rgb2gray(im1)), 'PeakThresh', 1);
```

The SIFT detector searches for peaks in scale space (similar to peaks in the autocorrelation function, see lecture notes). The second argument filters out peaks that are too small.

The vector `f1` contains 4 rows. The first two rows are the coordinates of the detected features. The second two contains an orientation and scale for which the the feature was detected. Plot the features together with the images using the command:

```
vl_plotframe(f1);
```

Compute the features for the second image and match the descriptors using the command:

```
[matches,scores] = vl_ubcmatch(d1,d2);
```

Thus for each descriptor in image 1 the function finds the two best matches. If the quality of the best and the second best match is similar then the match is rejected, otherwise the best match is selected.

We can now extract matching points using:

```
x1 = [f1(1,matches(1,:));f1(2,matches(1,:))];
x2 = [f2(1,matches(2,:));f2(2,matches(2,:))];
```

The following code randomly selects 10 matches, plots the two images next to each other and plots lines between the matching points.

```
perm = randperm(size(matches,2));
figure;
imagesc([im1 im2]);
hold on;
plot([x1(1,perm(1:10)); x2(1,perm(1:10))+size(im1,2)], ...
        [x1(2,perm(1:10)); x2(2,perm(1:10))],'-');
hold off;
```

How many of the matches appear to be correct?

For the report: Nothing, but you will need the point sets $x1$ and $x2$ for the next exercise.

6.1 (ALTERNATIVE) SIFT Implementation in MATLAB

If you are unable to run the VLFeat implementation of SIFT above, there also exist one in the more recent versions of MATLAB. However the interface is slightly different. Below is an example of how to extract the matches using the MATLAB implementation.

```
% Detect keypoints and compute the descriptors
f1 = detectSIFTFeatures(rgb2gray(im1));
[d1, valid_points1] = extractFeatures(rgb2gray(im1), f1);

f2 = detectSIFTFeatures(rgb2gray(im2));
[d2, valid_points2] = extractFeatures(rgb2gray(im2), f2);

% Perform the matching
matches = matchFeatures(d1,d2,'MatchThreshold',30, 'MaxRatio',0.7);

% Extract the matched keypoints
matchedPoints1 = valid_points1(matches(:,1),:);
matchedPoints2 = valid_points2(matches(:,2),:);
x1 = matchedPoints1.Location';
x2 = matchedPoints2.Location';
```

7 Triangulation using DLT

Computer Exercise 5. Using the estimated cameras from Computer Exercise 3 you will now triangulate the points detected in Computer Exercise 4.

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 into the two images and compare with the corresponding SIFT-points.

As in the resection the accuracy can be improved by normalizing the DLT equations and then solving for the unknown 3D points. Use the inner parameters for normalization (that is, multiply the equations with K^{-1}) and compare with the results you got previously without normalization. Is there any improvement?

As we saw in Computer Exercise 4 a portion of the SIFT matches will be incorrect. Most of the time (but not always) this will result in triangulations with large error. Compute the errors (both images) between the projected 3D points and the corresponding SIFT points. Remove those 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)?

```
Useful matlab commands:

xproj1 = pflat(Pa*X);
xproj2 = pflat(Pb*X);
%Computes the projections

good_points = (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(:,good_points);
%Removes points that are not good enough.
```

For the report: Submit the m-file, and the plots.