Computer Vision, Assignment 4
Model Fitting and Optimization

1 Instructions

In this assignment you will study model fitting. In particular you will use random sampling consensus RANSAC to robustly fit various models such as homographies and essential matrices. The data for the assignments is available from the course page `https://canvas.education.lu.se/courses/27507`

The assignment is due at the end (10/3 at midnight) of study week 8. 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. **I strongly suggest using LaTeX to typeset your report.** It is also allowed 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 bonus points for the home-exam.

2 Robust Homography Estimation and Stitching

Exercise 1. Show that if the two cameras $P_1 = [A_1\ t_1]$ and $P_2 = [A_2\ t_2]$ have the same camera center then there is a homography H that transforms the the first image into the second one. (You can assume that A_1 and A_2 are invertible.)

For the report: Complete solution.

Exercise 2. Suppose that we want to find a homography that transforms one 2D point set into another. How many degrees of freedom does a homography have?

What is the minimal number of point correspondences that you need to determine the homgraphy?

If the number of incorrect correspondences is 10% how many iterations of RANSAC do you need to find an outlier free sample set with 98% probability?

For the report: Answers are enough.

Computer Exercise 1. In this exercise you will use RANSAC to estimate homographies for creating panoramas.

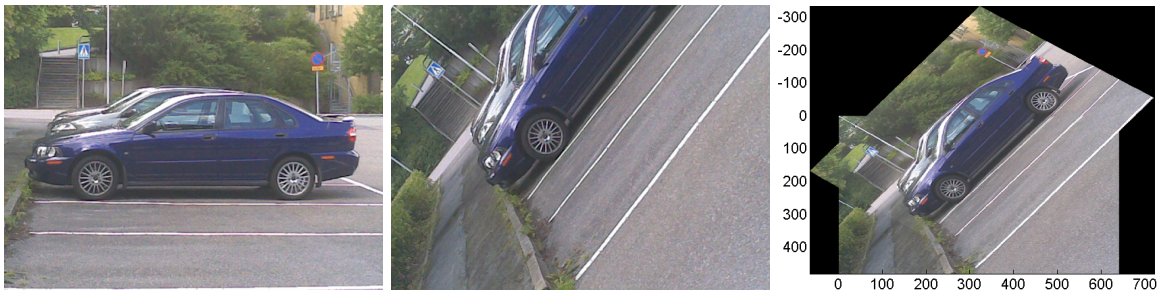


Figure 1: Image a.jpg, b.jpg and a panorama.

You can use the two images `a.jpg` and `b.jpg` (see Figure 1), but feel free to use other images if you want to. You will need to use VLfeat as in assignment 2 to generate potential matches, and then determine inliers using RANSAC.

Begin by loading the two images in Matlab and displaying them. The images are partly overlapping. The goal is to place them on top of each other as in Figure 1. Use VLFeat to compute SIFT features for both images and match them.

```
Useful matlab commands:

[fA dA] = vl_sift( single(rgb2gray(A)) );
[fB dB] = vl_sift( single(rgb2gray(B)) );

matches = vl_ubcmatch(dA,dB);

xA = fA(1:2,matches(1,:));
xB = fB(1:2,matches(2,:));
```

How many SIFT features did you find for the two images, respectively? How many matches did you find?

Now you should find a homography describing the transformation between the two images. Because not all matches are correct, you need to use RANSAC to find a set of good correspondences (inliers). To estimate the homography use DLT with a minimal number of points needed to estimate the homography. (Note that in this case the least squares system will have an exact solution, so normalization does not make any difference.) A reasonable threshold for inliers is 5 pixels.

How many inliers did you find?

Next transform the images to a common coordinate system using the estimated homography.

```
Useful matlab commands:

Htform = projective2d(bestH');
%Creates a transfomation that matlab can use for images.
%Note: MATLAB uses transposed transformations.

Rout = imref2d(size(A),[-200 800],[-400 600]);
%Sets the size and output bounds of the new image.

[Atransf] = imwarp(A,Htform,'OutputView',Rout);
%Transforms the image

Idtform = projective2d(eye(3));
[Btransf] = imwarp(B,Idtform,'OutputView',Rout);
%Creates a larger version of the second image

AB = Btransf;
AB(Btransf < Atransf) = Atransf(Btransf < Atransf);
%Writes both images in the new image. %(A somewhat hacky solution is needed
%since pixels outside the valid image area are not always zero...)

imagesc(Rout.XWorldLimits,Rout.YWorldLimits,AB);
%Plots the new image with the correct axes
```

For the report: The m-file, a plot of the panorama (and the two images if you don't use a.jpg and b.jpg), and answers to all the questions.

3 Robust Essential Matrix Estimation

Exercise 3. Suppose that we want to estimate the epipolar geometry between two calibrated cameras. How many degrees of freedom does an Essential matrix have?

What is the minimal number of point correspondences that you need to determine the essential matrix?

If the number of incorrect correspondences is 10% how many iterations of RANSAC do you need to find an outlier free sample set with 98% probability?

For the report: Answers are enough.

Computer Exercise 2. The file `compEx2data.mat` contains image points matched using SIFT descriptors from the images in Figure 2. The variable `x{i}` contains point coordinates from image i. The file

Figure 2: Two images of the Basilica di Santa Maria del Fiore, Florence.

`fivepoint_solver.m` contains a function that solves for the essential matrix from 5 point correspon-

dences. Use RANSAC with the five point solver to robustly estimate an essential matrix and its inlier matches. Note that the solver gives multiple solutions and you have to test them all in the RANSAC loop. **Hint: Run at least 100 iterations to make sure you find a good solution.**

For a match to be considered an inlier its distance to the corresponding epipolar line should be less than 5 pixels (in both images). To properly compute this distance you will need the camera calibration, which is the same for both images and can be found in the variable `K`.

How many inliers did you find?

Extract cameras from the best essential matrix and triangulate the 3D points. **Make sure that the points are in front of both cameras.** (Note that you can use your code from Computer Exercise 4, Assignment 3 for this part.) Plot a histogram of the reprojection errors in both images (only using the inliers, with unnormalized camera matrices and errors measured in pixels) and compute the RMS error.

For the report: Submit the m-file, a plot of the reconstruction, the histogram and state the number of inliers and the the RMS value.

4 Calibrated Structure from Motion and Local Optimization

Computer Exercise 3. Unfortunately there is no formula for computing the ML estimation when we use general pinhole cameras. The only way to find the ML estimate is to try to improve a starting solution using local optimization. In general we want to minimize a function of the form

$$\sum_i r_i(v)^2 = ||r(v)||^2. \tag{1}$$

Here v is the variables we can change, that is, camera positions, orientations and 3D-point positions, and and $r_i(v)$ is a reprojection error. In addition $r(v)$ is a vector containing all the $r_i(v)$.

The first order Taylor expansion at a point v_k is

$$r(v) \approx r(v_k) + J(v_k)\delta v, \tag{2}$$

where $\delta v = (v - v_k)$ and the Jacobian $J(v_k)$ is a matrix whose rows are the gradients of $r_i(v)$ at v_k. Given a current solution consisting of cameras and 3D-points the function `LinearizeReprojErr.m` computes the values of J and r. The file `update_solution.m` contains a function that computes a new set of cameras and 3D points from an update δv computed by any method. The file `ComputeReprojectionError.m` computes the reprojection error for a given set of cameras, 3D points and image points. It also returns the values of all the individual residuals as a second output.

The steepest descent method updates the solution by searching along the direction opposite to the gradient $\nabla\|r(v)\|^2 = 2J(v_k)^T r(v_k)$ giving the update

$$\delta v = -\gamma_k J(v_k)^T r(v_k). \tag{3}$$

Here γ_k is some unknown positive number that is determined by testing different values and evaluating the objective function. Note that unless v_k is a local minimizer it is always possible to find γ_k so that the objective function decreases by making γ_k small enough.

Implement the steepest descent method using the functions mentioned above, with your own strategy for selecting γ_k. Using the solution (unnormalized cameras and 3D points for the inlier matches) that you obtained in Computer Exercise 2 test how much you can decrease the objective value with your steepest descent implementation. **Note that you might need a very small γ_k to converge!** Plot the objective value vs. iteration number for 10 iterations and compute the final RMS error.

```
Useful matlab commands:
%Takes two camera matrices and puts them in a cell.
P = {P1,P2}
```

```
%Computes the reprejection error and the values of all the residuals
%for the current solution P,U,u.
[err,res] = ComputeReprojectionError(P,U,u);

%Computes the r and J matrices for the appoximate linear least squares problem.
[r,J] = LinearizeReprojErr(P,U,u)

%Computes the LM update.
deltav = -gammak*J'*r;

%Updates the variabels
[Pnew,Unew] = update_solution(deltav,P,U);
```

For the report: Submit the m-file, the plot and the RMS value.

Computer Exercise 4. In the Levenberg-Marquardt method we chose the update step

$$\delta v = -(J(v_k)^T J(v_k) + \lambda I)^{-1} J(v_k)^T r(v_k) \tag{4}$$

Implement the Levenberg-Marquardt method with some appropriate value for λ. Starting from the solution to Computer Exercise 2 test how much you can decrease the objective value and compare to the steepest descent implementation. Plot the objective value vs. iteration number for 10 iterations and compute the final RMS error.

```
Useful matlab commands:
%Computes the LM update.
C = J'*J+lambda*speye(size(J,2));
c = J'*r;
deltav = -C\c;
```

For the report: Submit the m-file, the plot and the RMS value.

5 Dynamic Objects and Factorization

Exercise 4. (OPTIONAL) Let the vector $x_{ij} \in \mathbb{R}^{2\times 1}$ $(i = 1, ..., m,\ j = 1, ..., n)$ represent noise free coordinates of point j in image i. Suppose that the coordinates fulfill

$$\begin{bmatrix} x_{i1} & x_{i2} & \dots & x_{in} \end{bmatrix} = \sum_{k=1}^{r} c_{ik} B_k + t_i \mathbb{1}, \tag{5}$$

where $c_{ik} \in \mathbb{R}^{2\times 1}$, $B_k \in \mathbb{R}^{1\times n}$ and $\mathbb{1}$ is a $1 \times n$ vector of all ones, in each image i. Then what is the rank of the matrix

$$X = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{1n} \\ x_{21} & x_{12} & \dots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m1} & x_{m2} & \dots & x_{mn} \end{bmatrix} ?$$

(You may assume that the row vectors $B_1, ..., B_n$ and $\mathbb{1}$ are linearly independent.)

Now suppose that A is a matrix containing the vectors $a_{ij} \in \mathbb{R}^{2\times 1}$ which are a noisy measurement of x_{ij}. If we want to minimize $\|A - X\|_F^2$ so that the rows of X are of the form (5) show that the optimal t_i is of the form

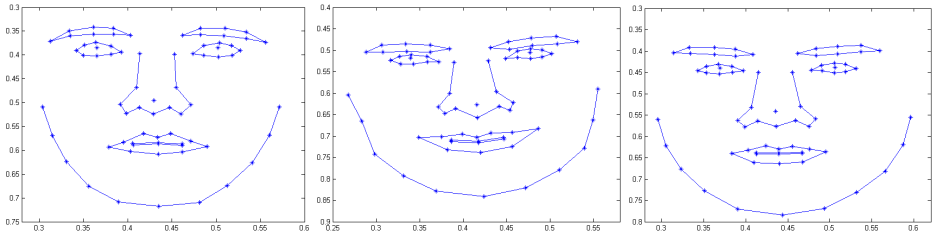
$$t_i = \bar{A}_i - \sum_{k=1} c_{ik} \bar{B}_k, \tag{6}$$

where $\bar{A}_i \in \mathbb{R}^{2\times 1}$ contains the row means over $\begin{bmatrix} a_{i1} & a_{i2} & \dots & a_{in} \end{bmatrix}$ and $\bar{B}_k$ is the mean over the row B_k.

Explain how can you find B_k and c_{ik} when t_i has been eliminated?

For the report: This exercise is optional, but if you want bonus for it submit a complete solution.

Computer Exercise 5. (OPTIONAL) The file `compEx4data.mat` contains a matrix `A` with point landmarks from 114 faces (The format is: First row x-coordinates image 1, second row y-coordinates image 1, third row x-coordinates image 2, fourth row y-coordinates of image 2 ...) You can plot a face using `drawface(x,y)`, where `x` and `y` are the x- and y-coordinates of the face (in the same order as in the matrix A).



Find the best approximation X of A, where X is of the form (5) with $k = 7$. What is the resulting relative error $\frac{||A-X||_F}{||A||_F}$? Plot the original and the estimated faces from image 1 in one figure. Do they look similar?

Determine the shape basis B_k, $k = 1, ..., 7$ such that (5) is fulfilled. Is the solution unique?

If the shape basis is known and we are given points from a previously unseen image we can find the coefficients that give the most similar shape in our model by solving

$$\min_{c_{ik},t_i} \left\| \begin{bmatrix} a_{i1} & a_{i2} & \dots & a_{in} \end{bmatrix} - \sum_{k=1}^{r} c_{ik} B_k - t_i \mathbb{1} \right\|_F^2 . \tag{7}$$

The matrix `Anoise` contains point coordinates (heavily corrupted by noise) from 3 previously unseen images. Solve (7) for the unseen images and plot the resulting faces together with the noisy points. Does the result look reasonable?

Suppose that we are given a new image where only a few of the points have been detected. To determine the coefficients c_{ik} we can remove unseen entries of B_k (and $\mathbb{1}$) from (7) and solve using the remaining residuals. How many points are needed to determine the coefficients c_{ik} if $r = 7$?

The matrix `Amissing` contains a subset of the point coordinates from 3 previously unseen images. (The missing entries are marked with NaN). Determine the coefficients c_{ik} by solving (7). Use the coefficients and the full B_k vectors to compute coordinates for all of the points. Plot the result together with the data from `Amissing`. Does the it seem ok?

```
Useful matlab commands:
%Draw the first face in A
drawface(A(1,:),A(2,:));

%Compute the mean over the rows of A.
m = mean(A,2);

%Subtract row means from A and ompute the SVD.
[u,s,v] = svd(A-m*ones(1,size(A,2)),'econ');

%Minimize ||A-UV||^2 over U
U = A/V;
```

For the report: This exercise is optional, but if you want bonus for it submit answers to the questions and plots of your results.