Name:

ID number: LiU-ID:

Passed: Date:

TSBB06 Computer Exercise D
Stereo Geometry

Developed by Klas Nordberg. Revised HT 2018 by Per-Erik Forssén.

Computer Vision Laboratory, Linköping University, Sweden

November 5, 2018

Introduction

This computer exercise has four parts:

1. A chapter with preparatory exercises that should be completed **before** you arrive at the lab session. This is a crucial for understanding of the lab and **not optional**.
2. A second part where you will investigate how to rectify stereo images, using a fundamental matrix computed from known camera matrices. Rectification is done via rectifying homographies that transform image coordinates. After applying the rectification to the two images, corresponding points in the images are found at the same row coordinate.
3. In practice the camera matrices are not known, instead the fundamental matrix is estimated directly from the image data. This is done in the third part of the exercise.
4. In the fourth part, you will investigate the problem of triangulation, where 3D points are computed from their corresponding image coordinates.

Extra

Tasks marked with an "Extra" box may be temporarily skipped and completed when the other tasks are finished.

1 Preparatory exercises

Before coming to the computer session it is necessary to read through the course material related to lecture D on stereo geometry; the lecture notes, the specified chapters in the compendium [3] (or the book by Hartley and Zisserman [1]), and the paper by Loop & Zhang [2][1]. It is not necessary that you understand all details of the paper, but at least have an idea of which problem is addressed and in principle how it is solved. In the following you will find a number of home exercises to be answered before the session.

1. Let H_1 and H_2 be a pair of rectifying homographies and let F_{21} be the fundamental matrix between image 1 and image 2, before the homographies are applied. Which relation must H_1 and H_2 satisfy in order to be rectifying homographies?

ANSWER:

2. If the camera matrices are known, the fundamental matrix that relates corresponding points in image 1 and 2 can be computed from the cameras. How? Which information do you need and how do you compute it from the camera matrices?

ANSWER:

3. In Section 3.1, the fundamental matrix is used to map points in one image to epipolar lines in the other image, inside a for-loop. Which computations take place inside the for-loop? Why?

ANSWER:

[1] Equation (17) in this paper is incorrect, the upper left 2×2 sub-matrix should have the opposite sign.

4. If you do an SVD of **F**: $\mathbf{F} = \mathbf{U}\,\mathbf{S}\,\mathbf{V}^T$, what characteristic property do you expect to find for **S** in this case that **F** is a fundamental matrix? As a consequence, what information do you expect to find in the third column of **U** and **V**, respectively?

ANSWER:

5. How can you compute the epipolar points from the matrix `F21` representing the fundamental matrix?

ANSWER:

6. Given the two epipoles, the MATLAB function `LoopZhangDistortion` computes the error within a certain interval for λ and plots the error and its approximate derivative. Look at the code of this function (use MATLAB command `type`) and verify that it performs the computations described in section 5.1 of the paper [2]. Does it?

ANSWER:

7. In Section 3.2, the homographies `Hp1`, `Hp2`, `Hr1` and `Hr2` are computed in accordance with the rectification method of Loop and Zhang. Verify that the MATLAB code works in accordance with [2]. In particular, verify that these calculations correspond to equations (16) and (17) in the paper[2]. Do they?

ANSWER:

8. In Section 3.3, the homographies H1 and H2 are used to rectify the two images, by applying them onto the pixel coordinates. Describe how the resulting homography T depends on H1 and H2.

ANSWER:

9. A possible solution to the problem of estimating rectifying homographies is to produce rectified images that are both upside down compared to what you expect. What transformation matrix on the homogeneous coordinates will flip vertical coordinates?

ANSWER:

10. In Section 4, the fundamental matrix is estimated from image data. The computations are found in the MATLAB function fmatrix_n8pa. Examine the file fmatrix_n8pa and identify where in the function each step is implemented. Can you do that?

ANSWER:

11. What does it mean for the solution of the problem of estimating the fundamental matrix if there is more than one small singular value?

ANSWER:

12. In Section 5 you will need to compute the coordinates of a 3D point given its image coordinates in two images, `y1` and `y2`, and the corresponding camera matrices `C1` and `C2`. Write a Matlab function that computes the 3D coordinates `xb` from the image points and camera matrices, using one of the linear methods described in the lecture.

ANSWER:

2 Starting Matlab and setting things up

Start MATLAB and add the path to the Matlab functions that are used in these exercises:

~~`addpath /site/edu/bb/MultidimensionalSignalAnalysis/ExerciseD/`~~

addpath /courses/tsbb06/ExerciseD/

(The files are also available in Lisam, under "Course documents".)

2.1 Useful Matlab functions

These functions are useful in the exercise:

- `null()` computes the (right) null space of a matrix.
- `svd()` computes the singular value decomposition of a matrix.
- `drawline()` draws a line in the current figure, the line is represented in dual homogeneous coordinates.
- `liu_crossop()` computes the 3 × 3 cross product matrix.
- `image_resample()` resamples an image using a homography.
- `map_points()` transforms 2D coordinates using a homography.
- `fmatrix_nn8pa()` an implementation of the 8-point algorithm. See also `fmatrix_n8pa()`.
- `vgg_get_nonhomg()` computes normalised coordinates from homogeneous coordinates of points in 2D or 3D.

Look at the Matlab code (with `type`) to see which arguments the functions use and how they are implemented.

3 Rectifying homographies

In this exercise, a pair of rectifying homographies will be determined according to the method described in the paper by Loop & Zhang [2]. This method determines three homographies for each image, a projective homography H_p, a similarity homography H_r, and a shearing homography H_s. The resulting homography H is given by the combination $H = H_s H_r H_p$. Both H_p and H_s are determined such that they minimize the resulting geometric distortion produced by H, while H_r is determined to make the epipolar lines parallel.

H_s can initially be set to the identity mapping, and the resulting homographies will rectify the images, but not with minimal geometric distortion. The final step of determining H_s is left as an extra exercise.

3.1 Introduction

In this exercise, we will try to rectify two images of a dinosaur (included in the standard image sequence from Oxford University). Load the images and the corresponding camera matrices, and display the images:

```
im1=imread('dinosaur0.png');
im2=imread('dinosaur5.png');
load('dino_Ps','P');
figure(1);image([im1 im2]);
C1=P{1};       % The camera matrix corresponding to image 1
C2=P{6};       % The camera matrix corresponding to image 2
```

Use the expression for the fundamental matrix that you derived in preparatory exercise 2:

```
F21 = ...
```

This matrix satisfies $\mathbf{y}_2^T \mathbf{F}_{21} \mathbf{y}_1 = 0$ for corresponding points (in the ideal case with no noise).

Use the `correspondences_select()` function to get coordinates of 4–5 corresponding image points. Make sure to maximize the window to get accurate correspondences. **Do not select points that all lie in a common plane**:

```
[y1,y2]=correspondences_select(im1,im2);
```

Then append a row of ones to your coordinate arrays, e.g.: `y1(end+1,:)=1;`

QUESTION: With what accuracy can you determine the coordinates of corresponding points in these images?
ANSWER:

The fundamental matrix maps a point in one image to an epipolar line in the other image:

```
l2=F21*y1;
l1=F21'*y2;
for ix=1:size(y1,2),
  l1(:,ix)=-l1(:,ix)/norm(l1(1:2,ix))*sign(l1(3,ix));
  l2(:,ix)=-l2(:,ix)/norm(l2(1:2,ix))*sign(l2(3,ix));
end
```

Plot the points and the lines:

```
figure(2);clf;
subplot(1,2,1);image(im1);hold on;
subplot(1,2,2);image(im2);hold on;
for k=1:size(y1,2),
  subplot(1,2,1);plot(y1(1,k),y1(2,k);drawline(l1(:,k));
  subplot(1,2,2);plot(y2(1,k),y2(2,k);drawline(l2(:,k));
end
```

QUESTION: What are the Cartesian coordinates of the two epipoles? Where are they in the images?

ANSWER:

QUESTION: Do the epipolar lines intersect with the corresponding points? Are the epipolar lines in each of the two images parallel? Is this result consistent with the positions of the epipoles?

ANSWER:

Compute the distances between the epipolar lines in an image and the corresponding points:

```
abs(sum(y1.*l1))
abs(sum(y2.*l2))
```

QUESTION: Are the distances of the same order as the estimated accuracy?

ANSWER:

Use the Matlab code that you produced in preparatory exercise 5 to determine the two epipoles:

```
...
e12 = ...
e21 = ...
```

QUESTION: Did you get epipolar points that are consistent with your observed epipolar lines?

ANSWER:

3.2 The Loop & Zhang method

Given that we now have a fundamental matrix and the corresponding epipoles, we can compute the rectifying homographies $\mathbf{H}_1$ and $\mathbf{H}_2$. This is done by minimizing a geometric distortion, eq. (11), over $\mathbf{z} = (\lambda \quad 1)^T$. In [2], it is described how to minimize this distortion based on iterative methods. Here, we will instead minimize the distortion by plotting the distortion as a function of λ and estimate which λ gives the minimal distortion.

At the computer exercise, use this function to plot the error for different intervals for λ and VERIFY that you can determine the minimum of eq. (11), at least to some degree of accuracy. Hint: The optimal λ lies in the range $-5 < \lambda < 5$. Use the zoom-tool of Matlab's figure window and Matlab function `diff` to get approximation of derivatives. Functions `min` and `find` may also be useful. Examples:

```
w=720;h=576; % Image width and height
lambda = start:step:stop; %Replace with the interval you choose
dist = LoopZhangDistortion(e12,e21,w,h,lambda);
figure(...);plot(lambda,dist);
set(gca,'YScale','log');grid on;
lmin = lambda(dist==min(dist)) %Print minimum lambda
```

QUESTION: How did you verify that you have found the minimum?

ANSWER:

QUESTION: What value of λ gives the minimal error?

ANSWER:

Compute the projective homographies for the two images given by this λ:

```
lambda = ...    % Insert your value here
w1=liu_crossop(e12)*[lambda 1 0]';
w1=w1/w1(3);
w2=F21*[lambda 1 0]';
w2=w2/w2(3);
Hp1=[1 0 0;0 1 0;w1'];
Hp2=[1 0 0;0 1 0;w2'];
```

With the projective homographies determined, the similarity homographies can be computed directly. This computation includes a free parameter `vcp`, that can be set to zero:

```
vcp=0;
Hr1=[F21(3,2)-w1(2)*F21(3,3) w1(1)*F21(3,3)-F21(3,1) 0;
     F21(3,1)-w1(1)*F21(3,3) F21(3,2)-w1(2)*F21(3,3) F21(3,3)+vcp;
     0 0 1];
Hr2=[w2(2)*F21(3,3)-F21(2,3) F21(1,3)-w2(1)*F21(3,3) 0;
     w2(1)*F21(3,3)-F21(1,3) w2(2)*F21(3,3)-F21(2,3) vcp;
     0 0 1];
```

According to the paper, $\mathbf{H}_1 = \mathbf{H}_{r1}\mathbf{H}_{p1}$ and $\mathbf{H}_2 = \mathbf{H}_{r2}\mathbf{H}_{p2}$ are rectifying homographies.

QUESTION: Given the homographies that you now have computed, how can you verify this relation? Does it work?

ANSWER:

3.3 Rectifying the images

Now that we have a pair of rectifying homographies, we can transform the two images accordingly and verify that the result is correct. This is done by computing where each image point (pixel) in the RESULTING image is found in the original image. In general this position is not at an integer coordinate, so we need to interpolate (linear interpolation) a pixel value at that point which then is used in the resulting image.

In order to produce a useful image we must also take care about which coordinate system the resulting image uses. Each image is transformed by its own general homography which means that the resulting images may end up in arbitrary positions and orientations relative the origin. Thus, we transform both images with a common translation and scaling to assure that both images fit inside the same width and height as the original images:

```
oldcorners=[1 w w 1;1 1 h h];
newcorners1=map_points(H1,oldcorners);
newcorners2=map_points(H2,oldcorners);
mincol=min([newcorners1(1,:) newcorners2(1,:)]);
minrow=min([newcorners1(2,:) newcorners2(2,:)]);
T=[1 0 -mincol+1;0 1 -minrow+1;0 0 1]
newcorners1=map_points(T*H1,oldcorners);
newcorners2=map_points(T*H2,oldcorners);
maxcol=max([newcorners1(1,:) newcorners2(1,:)]);
maxrow=max([newcorners1(2,:) newcorners2(2,:)]);
T=inv(diag([maxcol/w maxrow/h 1]))*T
```

QUESTION: How can you see that `T` is a scaling and translation transformation?

ANSWER:

Finally, transform the images and display the result

```
imr1=image_resample(double(im1),T*H1,h,w);
imr2=image_resample(double(im2),T*H2,h,w);
figure(3);image(uint8([imr1 imr2]));
```

Transform the points that were initially chosen. These should now lie (approximately) on the same row in both images. Verify this:

```
map_points(T*H2,y2(1:2,:))
map_points(T*H1,y1(1:2,:))
```

QUESTION: Do the transformed images and points appear rectified?

ANSWER:

If you get images that are upside down, use the flip transformation you derived in preparatory exercise 9 to turn them the right way. In this case multiply the flip transformation matrix onto $\mathbf{H}_1$ and $\mathbf{H}_2$ and redo the calculations in this section. **In particular, you should apply this transformation before computing the transformation** `T`.

Extra

Determine also the shearing homographies $\mathbf{H}_{s1}$ and $\mathbf{H}_{s2}$ which reduce the total geometric error even more while keeping the resulting homographies rectified. See section 7 of the paper. Verify your results.

4 The fundamental matrix and the 8-point algorithm

In this task, we will go through the steps of the 8-point algorithm for estimation of the fundamental matrix $\mathbf{F}$. The algorithm has its name from the fact that at least 8 pairs of corresponding points are required for determining the fundamental matrix with this method.

Given a set of $N \geq 8$ corresponding point pairs $\{\mathbf{y}_{1k}, \mathbf{y}_{2k}\}$, the 8-point algorithm consists of the following steps:

1. Form a data matrix $\mathbf{A}$ from the image coordinates.
2. Let $\mathbf{F}_{\text{vec}}$ be the right singular vector corresponding to the smallest singular value, alternatively the eigenvector of $\mathbf{A}^T\mathbf{A}$ corresponding to the smallest eigenvalue.
3. Reshape $\mathbf{F}_{\text{vec}}$ to a matrix $\mathbf{F}$.
4. Enforce the constraint $\det \mathbf{F} = 0$ which gives $\mathbf{F}_0$.

Use the same images as in the previous task. Identify 8 corresponding points in the two images and insert the homogeneous coordinates in the columns of matrices `y1` and `y2`:

```
y1 = [ ... ];
y2 = [ ... ];
```

Now use the function `F = fmatrix_nn8pa(y1,y2)` to estimate the fundamental matrix. The function prints the singular values of the data matrix **A** and, ideally, there should be a only one of them that is significantly smaller than the others.

QUESTION: Is this the case?

ANSWER:

If you do not get a well-defined solution, it may be that the points you have chosen are positioned in 3D space approximately in a plane. Such a configuration of the 3D points results in degeneracies in the estimation. **In this case, you may need to select some of the points differently such that they are not in a plane, alternatively add more corresponding image points**.

Compare this new estimate of the fundamental matrix to the matrix `F21` determined in the previous section.

QUESTION: How can you do this comparison?

ANSWER:

A good way to determine the quality of a fundamental matrix is to see how points in one image are mapped to epipolar lines in the other image and how close these lines are to the corresponding points. Use some MATLAB code from the previous task to compute the epipolar lines, display the points and lines, and compute the distances from the lines to the points:

```
figure(1);clf;
subplot(1,2,1);image(im1);hold on;
subplot(1,2,2);image(im2);hold on;

l2=F*y1;
l1=F'*y2;
for k=1:size(y1,2),
  l1(:,k)=-l1(:,k)/norm(l1(1:2,k))*sign(l1(3,k)); %Normalise dual
  l2(:,k)=-l2(:,k)/norm(l2(1:2,k))*sign(l2(3,k)); %homog. coord.
end
for k=1:size(y1,2),
  subplot(1,2,1);plot(y1(1,k),y1(2,k),'or');drawline(l1(:,k));
  subplot(1,2,2);plot(y2(1,k),y2(2,k),'or');drawline(l2(:,k));
end

abs(sum(y1.*l1))
abs(sum(y2.*l2))
```

QUESTION: Is `F` a better or worse estimate of the fundamental matrix compared to `F21`? Why?

ANSWER:

Extra

In the following, some of the details of the 8-point algorithm are examined more closely. This can be done by

- Examine how the quality of the estimated **F** changes when the number of corresponding points pairs, N, increases. How does the smallest singular value of the data matrix change when N increases?
- Examine how the quality of the estimated fundamental matrix changes if no enforcement of the constraint $\det \mathbf{F} = 0$ is made.

This can be done be modifying the function `fmatrix_nn8pa()`, but remember to keep the original file as a copy.

Extra

Compute **F** using the 8-point algorithm together with the Hartley-normalisation, using `fmatrix_n8pa()`. Investigate how the quality of the estimated **F** changes compared to the non-normalised case.

5 Triangulation

Triangulation or reconstruction includes methods for determining the 3D point $\mathbf{x}$ given its projection onto two image points $\mathbf{y}_1 \sim \mathbf{C}_1\mathbf{x}$ and $\mathbf{y}_2 \sim \mathbf{C}_2\mathbf{x}$. In this case, we assume that the image points $\mathbf{y}_1$ and $\mathbf{y}_2$ are known together with the camera matrices $\mathbf{C}_1$ and $\mathbf{C}_2$.

This exercise can be applied to a data set that consists of 450 3D points and their projections onto two images and the corresponding camera matrices. The 3D coordinates in `x` have units in millimeters, and the 2D coordinates in `y1` and `y2` are in pixels. The two images and the camera matrices are also included:

```
load('tridata','x','y1','y2','im1','im2','C1','C2');
```

The image points `y1` and `y2` have been extracted from the images automatically using an interest point detector. They are adjusted for lens distortion but the images are not resampled to the adjusted coordinates. Thus, the extracted image coordinates do not fall exactly onto point features in the images.

```
figure(1);image([im1 im2]);
colormap(gray(256));hold on;
for k=1:size(y1,2),
  plot(y1(2,k),y1(1,k),'ro');
  plot(y2(2,k)+size(im1,2),y2(1,k),'ro');
end
hold off;
```

Use the function that you have written in preparatory exercise 12 to triangulate the 3D points.

```
xrec=[];
for ix=1:length(y1),
  recpoint = ...;
  xrec=[xrec recpoint];
end
```

Plot the 3D coordinates and verify that the reconstructed points appear as they should:

```
figure(3);plot3(xrec(1,:),xrec(2,:),xrec(3,:),'o');
```

QUESTION: Do they?

ANSWER:

QUESTION: Compute the error in the reconstructed 3D coordinates relative to the true 3D coordinates in `x` (units in mm). What is the mean of the error, and what is its standard deviation?

ANSWER:

QUESTION: Compute the *reprojection error* by projecting the estimated 3D points into each of the two images and computing the error relative to the given image coordinates in `y1` and `y2` (units in pixels). What is the mean and standard deviation?

ANSWER:

Extra In the following, the triangulation method is investigated more closely. This can be done by implementing two different triangulation methods and investigate how their 3D errors and reprojection errors differ.

Extra Test estimation of the 3D points using Hartley normalization. How much do the positions of the 3D points differ with and without normalization?

References

[1] R. Hartley and A. Zisserman. *Multiple-view Geometry in Computer Vision.* Cambridge University Press, 2003.

[2] C. Loop and Z. Zhang. Computing rectifying homographies for stereo vision. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 125–131, Dublin, Ireland, 1999.

[3] K. Nordberg. *Introduction to Representation and Estimation in Geometry.* Linkping University, 2013.