LUND UNIVERSITY
FACULTY OF ENGINEERING
MATHEMATICS

TAKE-HOME EXAM
COMPUTER VISION 2023

The solutions should be handed in no later than 48 hours after downloading the exam.
Data for the exam can be downloaded from Canvas.
It is not permitted to get help from other persons. Credits can be given for partially solved problems. Similar to the assignments, you should hand in a pdf-file (handwritten solutions should be scanned and inserted in the pdf) together with Matlab code. Write your solutions neatly, explain your calculations and specify what Matlab-scripts you have used. The pdf and your m-files should also be submitted through the canvas page.

1. *Consider the camera matrix*

$$P = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 2 & 4 & 5 \\ 0 & 0 & -1 & 2 \end{pmatrix}.$$

a) *Compute the projections (in regular 2D Cartesian coordinates) of the 3D scene points $(1,2,3)$ and $(4,5,6)$ in the camera* *(0.3)*

b) *What is the camera center (in 3D Cartesian coordinates) and principal axis?* *(0.3)*

c) *Which set of 3D points project onto vanishing points in $\mathbb{P}^2$?* *(0.2)*

d) *Give a geometric interpretation of this set and why it projects onto vanishing points.* *(0.2)*

2. a) *Compute the intersection point between the following lines in $\mathbb{P}^2$* *(0.3)*

$$\ell_1 = (0,1,1) \quad \textit{and} \quad \ell_2 = (1,0,1).$$

b) *Transform the lines ℓ_1 and ℓ_2 using the homography*

$$H = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 1 & -1 \\ 1 & 1 & 2 \end{bmatrix},$$

and compute the new intersection point. *(0.4)*

c) *Give a geometric interpretation of the new intersection point.* *(0.3)*

3. *Consider the three cameras*

$$P_1 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}, \quad P_2 = \begin{pmatrix} 1 & 0 & 4 & 1 \\ 0 & 1 & 2 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix}, \quad P_3 = \begin{pmatrix} -1 & 1 & -2 & 1 \\ -1 & 0 & -1 & 0 \\ 0 & -2 & 0 & 0 \end{pmatrix}.$$

a) *Compute the fundamental matrix F_{12} between P_1 and P_2, and the fundamental matrix F_{13} between P_1 and P_3.* *(0.6)*

b) *The point $\boldsymbol{x}_2 = (1,1)$ is seen in P_2 and the point $\boldsymbol{x}_3 = (2,1)$ is seen in P_3.*
Use the two fundamental matrices to compute the coordinates of the corresponding point $\boldsymbol{x}_1$ seen in P_1, without triangulating the 3D point. (Hint: Recall that $F_{21} = F_{12}^T$) *(0.4)*

4. *Below we show an image (`ex4.jpg`) of a building together with a 3D model. In this exercise we will estimate the camera matrix P from 2D-3D point correspondences. Unfortunately the matches contain some outliers, so RANSAC will be necessary.*


a) *How many degrees of freedom does an uncalibrated pinhole camera have?
How many point matches do you need to be able to compute the camera matrix? (0.2)*

b) *Suppose that the number of mismatched points are roughly 35%. If you use a minimal set of correspondences, how many RANSAC iterations do you need to find an outlier-free set of point correspondences with 99.99% probability? (0.3)*

c) *Write a function that computes a camera matrix from a minimal number of correspondences using DLT. Use RANSAC with this function to determine the camera matrix from the matches in `ex4.mat`. A point is considered to be an inlier if its projection is less than 5 pixels from the corresponding image point. Plot the 3D model and the camera center and principal axis in a 3D plot. (Don't forget to make sure that the principal axis point towards the visible 3D points) (0.5)*

5. *In this exercise we consider the epipolar geometry of a camera which is moving forward while zooming (changing focal length). The two cameras can then be written as*

$$P_1 = \begin{bmatrix} f & 0 & x_0 \\ 0 & f & y_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \qquad P_2 = \begin{bmatrix} \alpha f & 0 & x_0 \\ 0 & \alpha f & y_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -1 \end{bmatrix}$$

where $\alpha \in \mathbb{R}$ is the zoom-factor.

a) *Compute the fundamental matrix for P_1 and P_2, and show that it does not depend on the focal length f or zoom-factor α. (0.6)*

b) *Show that all corresponding points (x, y) and $(\bar{x}, \bar{y})$ satisfy (0.2)*

$$\det\left(\begin{bmatrix} x - x_0 & \bar{x} - x_0 \\ y - y_0 & \bar{y} - y_0 \end{bmatrix}\right) = 0$$

c) *Give a geometric interpretation of why this is the case. (0.2)*

6. *Points on a plane seen in two images are related by a homography. If the cameras are*

$$P_1 = K[I\ 0] \quad \textit{and} \quad P_2 = K[R\ t]$$

and the 3D plane is given by $\{X \mid n^T X + 1 = 0\}$*, then the homography can be written as*

$$H = K(R - tn^T)K^{-1}$$

where $n \in \mathbb{R}^3$ *is the (scaled) plane normal. The figure below shows two images with two dominant planes. The file* `ex6.mat` *contains* R, t, K *and matched 2D points. The goal of this assignment is to estimate the plane normals* $n_a \in \mathbb{R}^3$ *and* $n_b \in \mathbb{R}^3$ *corresponding to the left and right side of the building.*


`ex6_im1.jpg`


`ex6_im2.jpg`

a) *Show that for given cameras, we can estimate the normal vector* $n \in \mathbb{R}^3$ *from corresponding points by solving a linear equation system.*
(Hint: The linear system will be inhomogeneous, $Ax = b$*, so DLT does not apply but we can simply use backslash ,* $x = A\backslash b$*, in Matlab).* *(0.2)*

b) *Implement code for estimating the plane normal* n_a *to the first plane using the correspondences* `x1a` *and* `x2a`*. The matches have no outliers, so RANSAC is not necessary. Visualize the homography by transforming the first image onto the other.* *(0.3)*

c) *Let* n_b *denote the normal vector to the right wall, which we know is orthogonal to* n_a*. Show that given the first normal vector* n_a *(and the camera matrices) you can linearly estimate* n_b *from point correspondences.* *(0.2)*
(Hint: can you parameterize all vectors n_b *such that* $n_a^T n_b = 0$*.)*

d) *Implement code for estimating the plane normal* n_b *(such that it is orthogonal to* n_a*) to the second plane using the correspondences in* `x1b` *and* `x2b`*.* *(0.2)*

Good Luck!