Module: **M4. 3D Vision** **Final exam**
Date: February 21, 2019
Teachers: Antonio Agudo, Coloma Ballester, Josep Ramon Casas, Gloria Haro, Javier Ruiz
Time: 2h

- Books, lecture notes, calculators, phones, etc. are not allowed.
- All sheets of paper should have your name.
- Answer each problem in a separate sheet of paper.
- All results should be demonstrated or justified.

Problem 1 *0.75 Points*

(a) *(0.25p)* Let us denote by H a homography in the 2D projective space. What is the size of the matrix H and which kind of matrix is it? How many degrees of freedom does it have (justify it)?

H is a 3×3 non-singular matrix; thus, it is an invertible mapping. It has 8 degrees of freedom: 9 elements - 1 because of the scaling invariance.

(b) *(0.25p)* Enumerate the different situations where two images may be related by a homography.

A homography relates two images:

- of the same plane (flat object) in the 3D scene;
- taken with a camera that doesn't vary its position, i.e. either rotating about its centre or varying its focal length;
- the whole scene is far away from the camera.

(c) *(0.25p)* Mention two different problems in computer vision whose solution involves the estimation of a homography. Justify why the use of a homography is reasonable.

See last slides of lecture 2a.

Problem 2 *0.75 Points*

Consider the problem of computing a 2D homography H between two image views of a plane object. Let $\mathbf{x}_i \in \mathbb{P}^2$ and $\mathbf{x}'_i \in \mathbb{P}^2$, $i = 1, \ldots, n$, be a set of points on the first image and the second image, respectively, such as, in pairs, they correspond: $\mathbf{x}_i \longleftrightarrow \mathbf{x}'_i$, $\forall i = 1, \ldots, n$.

(a) *(0.25 points)* What is the minimum value of n? More precisely, how many corresponding points in general position do you need to compute H such that $\mathbf{x}'_i = H\mathbf{x}_i$, $\forall i = 1, \ldots, n$?
(Recall that general position means that no three points are collinear).

The minimum number n of corresponding points in general position is four because the 2D homography H has nine entries to compute, minus one scale factor. That is, eight unknowns. On the other hand, each pair of corresponding points provides two equations.

(b) *(0.5 points)* Describe the Normalized Direct Linear Transformation (Normalized-DLT) algorithm to compute H.

The Normalized-DLT applies a normalization of the data consisting of translation and scaling of image coordinates before applying the DLT algorithm. Finally, an appropriate correction to the result expresses the computed H with respect to the original coordinate system. More precisely, slides 38-39 of lecture2_CB.pdf, where the usual DLT algorithm is summarized on slide 37.

Problem 3 *0.75 Points*

What is the general form of a finite projective camera matrix P? Describe in detail its internal and external parameters.

A general projective camera P maps world points $\mathbf{X} \in \mathbb{P}^3$ to image points $\mathbf{x} \in \mathbb{P}^2$ according to $\mathbf{x} = P\mathbf{X}$. A general finite projective camera P can be decomposed as $P = K[R|\mathbf{t}]$, where K and R are 3×3 matrices and $\mathbf{t}$ is a 3×1 vector. K is the calibration matrix containing the internal parameters, and R, $\mathbf{t}$ represent the external parameters of the camera. In particular,

$$K = \begin{pmatrix} \alpha_x & s & x_0 \\ 0 & \alpha_y & y_0 \\ 0 & 0 & 1 \end{pmatrix},$$

where $x_0 = m_x p_x, y_0 = m_y p_y$, with f the focal length of the camera, (p_x, p_y) the coordinates of the principal point in a reference system in the image where the origin of coordinates in the image plane is situated at a corner of the image plane, $\alpha_x = m_x f, \alpha_y = m_y f$ are parameters allowing the possibility of having non-square pixels for the image coordinates (of units m_x, m_y) and s is the skew parameter. Finally R and $\mathbf{t}$ represent the external parameters of the camera and give the position and orientation of the camera in the world coordinate system (in particular, R and $\mathbf{t}$ relate the inhomogeneous 3-vector $\widetilde{\mathbf{X}}$ representing the coordinates of a point in the world reference system with the same point in the camera coordinate frame, $\widetilde{\mathbf{X}}_{\text{cam}}$, that is, $\widetilde{\mathbf{X}}_{\text{cam}} = R(\widetilde{\mathbf{X}}_{\text{w}} - \widetilde{\mathbf{C}})$, and $\mathbf{t} = -R\widetilde{\mathbf{C}}$, where $\widetilde{\mathbf{C}}$ are the inhomogeneous coordinates of the camera centre $\mathbf{C}$ in the world coordinate frame.

Problem 4 *1.25 Points*

Camera calibration

(a) *(0.75p)* Explain the fundamental idea and the main steps of Zhang's calibration method studied in class (you can explain it in words, there is no need to include formulas but you can add them if you prefer).

The Zhang's calibration method uses three or more images of a planar pattern in different positions. It uses the fact that two images of a planar pattern are related by a homography. The different homographies depend on the calibration matrix K of the camera and on the relative orientation, R, and translation $\mathbf{t}$, of the camera with respect to the 3D planar pattern. Using that R is an orthogonal matrix we arrive to a homogeneous least-squares problem on the unknown coefficients of the image of the absolute conic, $\omega = K^{-T}K^{-1}$, which is related to the internal camera parameters. The main steps of the algorithm are the following:

- Compute the keypoints and the keypoint correspondences between the image of the planar pattern and the three (or more) images taken by the camera to be calibrated.
- Compute the homographies between the planar pattern and the three (or more) images.
- Estimate ω by solving a homogeneous least-squares problem (using the SVD).
- Extract K from ω by a Cholesky factorization.
- (Optional) compute R and $\mathbf{t}$.

(b) *(0.25p)* Define the image of the absolute conic. How many degrees of freedom does it have?

The image of the absolute conic is the projection of the absolute conic to the image plane. It has the following expression $\omega = (KK^T)^{-1} = K^{-T}K^{-1}$ where K is the matrix of the internal parameters of the camera. It is a symmetric matrix and thus has 6 different elements that reduce to 5 degrees of freedom because of the scaling invariance.

(c) *(0.25p)* Define the Perspective-n-Point (PnP) problem and specify which are unknown variables and the available/known data.

The PnP problem seeks to estimate the absolute pose of a calibrated camera, that is, the orientation, R, and position, $\mathbf{t}$, of the camera, where 6 dof need to be determined.
The available data is a set of n 3D-2D point correspondences and the camera internal parameters, i.e. matrix K.

Problem 5 *2 Points*

Consider two images I and I' taken by the same camera (with intrinsic matrix $K = \begin{bmatrix} 5 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 1 \end{bmatrix}$) and the estimated Fundamental matrix between them $F = \begin{bmatrix} 0 & 1 & 1 \\ -1 & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix}$. Answer the following questions:

a) Find the two epipolar lines in image I' corresponding to points $p_1 = (1, 0)$ and $p_2 = (10, 10)$ in image I.

b) Find the epipole e' in image I'.

c) Are the two images I and I' rectified?

d) If the correspondence point to p_1 is $p'_1 = (2, 1)$ and to p_2 is $p'_2 = (11, 10)$. Would you say any of them is an outlier?

e) Find the essential matrix E of the system.

f) What is the main difference between the fundamental matrix F and the Essential matrix E?

g) Would you be able to reconstruct the structure of the camera configuration (rotation, translation and scale) in this system?

f) What would you need to reconstruct everything (rotation, translation and scale)?

a) From $l' = Fp$ we can find $l'_1 \equiv x - y - 1 = 0$ and $l'_2 \equiv 11x - 10y - 10 = 0$

b) The epipole e' will be in the intersection between both epipolar lines l'_1 and l'_2, $e' = (0, -1)$

c) No, as epipoles should be at infinity in rectified images.

d) p'_1 lies in the epipolar line l'_1 therefore it is an inlier. However, p'_2 does not, so it is an outlier.

e) $E = K'^T F K = \begin{bmatrix} 0 & 5 & 1 \\ -5 & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix}$

f) F expresses the relation in uncalibrated cameras (in pixel coordinates) while E expresses the relation in the calibrate cased (camera coordinates).

g) We can obtain R and T (normalized).

h) The scale is not possible to obtain unless there is an object in the scene with known length.

Problem 6 *1 Point*

Triangulation methods.

(a) *(0.75p)* Derive the expression of the matrix A that describes the linear system of equations in the linear triangulation methods.

We have a correspondence $\mathbf{x} \longleftrightarrow \mathbf{x}'$ and we want to compute $\mathbf{X} \in \mathbf{R}^3$ such that $\mathbf{x} = P\mathbf{X}$ and $\mathbf{x}' = P'\mathbf{X}$.

Since these equations are written in projective coordinates, the sense is that the points $\mathbf{x} = (x, y, 1)^T$ and $P\mathbf{X}$ are colinear. Similarly, $\mathbf{x}' = (x', y', 1)^T$ and $P'\mathbf{X}$ are colinear. We can express the colinearity in two ways:

(i) Using two unknown scalar factors λ, and λ':

$$\lambda \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = P\mathbf{X}, \text{ and } \lambda' \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = P'\mathbf{X}.$$

(ii) Using the cross product:

$$\mathbf{x} \times P\mathbf{X} = 0, \tag{1}$$

$$\mathbf{x}' \times P'\mathbf{X} = 0. \tag{2}$$

We develop the equations in (ii) but the same result is obtained from the equations in (i). Let $\mathbf{p}^{iT}$ be the rows of P and $\mathbf{p}'^{iT}$ be the rows of P'. From (1), we obtain the equations

$$\begin{aligned} x\mathbf{p}^{3T}\mathbf{X} - \mathbf{p}^{1T}\mathbf{X} &= 0, \\ y\mathbf{p}^{3T}\mathbf{X} - \mathbf{p}^{2T}\mathbf{X} &= 0, \\ x\mathbf{p}^{2T}\mathbf{X} - y\mathbf{p}^{1T}\mathbf{X} &= 0. \end{aligned} \tag{3}$$

These equations are linear in the components of $\mathbf{X}$. Two of them are linearly independent.

Thus, collecting the equations coming from (1) and (2) we can write them as

$$\mathbf{A}\mathbf{X} = 0, \tag{4}$$

where

$$\mathbf{A} = \begin{pmatrix} x\mathbf{p}^{3T} - \mathbf{p}^{1T} \\ y\mathbf{p}^{3T} - \mathbf{p}^{2T} \\ x'\mathbf{p}'^{3T} - \mathbf{p}'^{1T} \\ y'\mathbf{p}'^{3T} - \mathbf{p}'^{2T} \end{pmatrix}.$$

(b) *(0.25p)* State the advantages and disadvantages of using a pair of views with a small angle between the visual rays.

The uncertainty region in the localization of the 3D point gets larger as the angle between rays becomes smaller but on the other hand it's easier to find 2D-2D correspondences since the point of views are more similar.

Problem 7 *0.5 Points*

Assume that we have N different calibrated views of a 3D object and their corresponding depth maps d_i, $i = 1, ..., N$. Describe the main steps of the depth map fusion algorithm that reconstructs the object surface.

1. Create a Signed Distance Function (sdf) for every depth map d_i, $i = 1, ..., N$ at every voxel z:

$$sdf_i(\mathbf{z} = [\mathbf{X}]) = (P_i\mathbf{X})_3 - d_i([P_i\mathbf{X}])$$

where $\mathbf{X}$ is a 3D point in homogeneous coordinates, [.] is the projection operator that transforms homogeneous coordinates to Cartesian ones, and P_i is the projection matrix of the i-th camera.

2. Average the different signed distance functions at every voxel.
3. The reconstructed surface can be extracted as the zero iso-surface of the average function.

Problem 8 *0.5 Points*

Formulate the projection equation (indicating the corresponding size of every matrix) in terms of a measurement matrix, assuming a perspective camera for: 1) a rigid shape, 2) a non-rigid one. To compute shape and motion by rigid factorization, which rank do we have to enforce in every case and how can this be done?

Considering m the number of images and n the number of points, the projection equation can be written as $\mathbf{M} = \mathbf{PX}$:

In both cases: $\mathbf{M}$ is a $3m \times n$ measurement matrix, $\mathbf{P}$ and $\mathbf{X}$ are a $3m \times 4$ motion and $4 \times n$ shape matrix components, respectively. We will use a SVD factorization, imposing a rank-4 and rank-(K+1) decomposition, respectively, where K is the rank of a linear subspace.

Problem 9 *1.0 Points*

Let us assume a collection of I image frames with extrinsic parameters $\mathbf{P}_i$ with $i = \{1, \ldots, I\}$, where a 3D rigid object composed of P points is observed. Due to lack of visibility and outliers, a few points are not viewed in some frames. Particularly, the corresponding visibility vectors contain 12, 10, 14, and 14 components for every image, respectively. Assuming $P = 14$, for this particular case we always observe the points with smaller indexes $p = \{1, \ldots, P\}$. We want to simultaneously estimate 3D shape $\mathbf{X} \equiv [\mathbf{x}_1, \ldots, \mathbf{x}_P]$ (every $\mathbf{x}_p$ contains the 3D coordinates of the p-th point) and motion solely from 2D annotations by sparse bundle adjustment. For this toy example, represent the corresponding structure of the Jacobian matrix to code the problem and indicate the final matrix size. The intrinsic parameters of the camera can be assumed to be known. (0.7 points)

If the four image frames are a part of a monocular video, could we impose more constraints to sort out the problem? If so, describe them and represent this type of priors in the previous pattern. (0.3 points)

The rows correspond to the number of equations, and the columns with the number of parameters to be estimated. Thus, the number of rows is $(12 + 10 + 14 + 14) * 2$, and the number of columns is $(I * e + P * 3)$, where e represents the number of extrinsic parameters, i.e., 3 translations and 3 rotations (or 4 using quaternions). The corresponding pattern is displayed in Fig.1.

For a monocular video, we may add temporal smoothness priors to estimate the camera motion. To this end, first- (pure sequential) or high-order approximations (such as a sliding-window with several frames) can be used. For simplicity, we incorporate and represent a pure sequential case (adding 3 constraints), i.e., using a first-order approximation.

Problem 10 *0.75 Points*

3D data captured by a 3D sensor has a double nature (photometry + geometry) which is, let's say, more balanced than for 2D data captured by regular camera.

(a) Describe the two different natures of 3D SENSOR data

Geometric information is represented by purely geometric data, i.e. geometric measures, which compose a numerical representation of objects (like in CAD applications). Using computer graphics techniques, one can generate a (rendered) view of the objects in a scene by calculation (e.g. raytracing techniques) if illumination and material properties (reflectivity) are provided.
Photometric information basically refers to some form of light captured by imaging sensors, which can be rendered, displayed and presented visually more directly. Geometry is not explicit in photometric information, but by exploiting human perception capabilities (exploration, analysis and understanding) we can derive geometric values (up to a projectivity) in the same way than looking at the physical world.

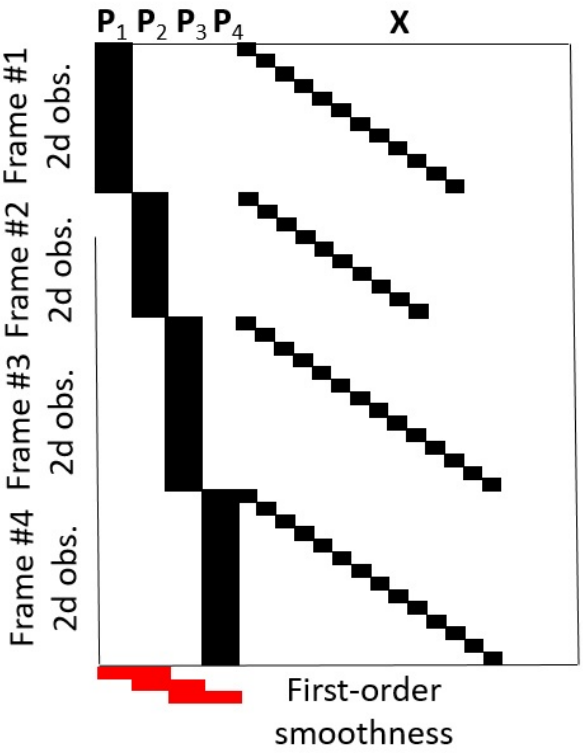

Figure 1: Jacobian pattern.

(b) The data we capture with a 3D sensor (3D vision), is it equivalent to the 3D geometry in a blueprint/CAD?

Absolutely not! 3D vision usually assumes one (or several) points of view from which 3D geometry is computed, leaving part of the scene geometry unavailable due to occlusions. On the contrary 3D geometry in Graphics or CAD design is a complete representation of the scene (even rendering occlusions and transparencies if needed)

Problem 11 *0.75 Points*

(a) In J. Papon, et al (2018), "Voxel Cloud Connectivity Segmentation - Supervoxels for Point Clouds," CVPR 2013, the following explanation introduces what is a superpixel approach:

> *Segmentation algorithms aim to group pixels in images into perceptually meaningful regions that conform to object boundaries. Graph-based approaches, such as Markov Random Field (MRF) and Conditional Random Field (CRF), have become popular, as they merge relational low-level context within the image with object-level class knowledge. The cost of solving pixel-level graphs led to the development of mid-level inference schemes which do not use pixels directly, but rather use groupings of pixels, known as superpixels, as the base level for nodes. Superpixels are formed by over-segmenting the image into small regions based on local low-level features, reducing the number of nodes that must be considered for inference. Due to their strong impact on the quality of the eventual segmentation, it is important that superpixels have certain characteristics. Of these, avoiding violating object boundaries is the most vital, as failing to do so will decrease the accuracy of classifiers used later - since they will be forced to consider pixels that belong to more than one class. Additionally, even if the classifier does manage a correct output, the final pixel level segmentation will necessarily contain*

errors. Another useful quality is regular distribution over the area being segmented, as this will produce a simpler graph for later steps.

Note: Old image segmentation techniques using quadtrees may be seen as a predecessor of modern superpixel approaches.

Papon used supervoxels for the segmentation of point cloud data. Could you extend the explanation above to the concept of supervoxels? Derive your explanation from what you know about 2D segmentation and the nature of point cloud data, and use the following terms in your answer (over-segmentation, object boundaries, octree, even distribution, 26-adjacency).

Voxel Cloud Connectivity Segmentation (VCCS) is a "superpixel" method which generates volumetric **over-segmentations** of 3D point cloud data, known as supervoxels. Supervoxels represent the temporal coherence of segmented objects along time and adhere to **object boundaries** better than state-of-the-art 2D methods, while remaining efficient enough to use in online applications. VCCS uses a region growing variant of k-means clustering for generating its labeling of points directly within a voxel octree structure. Supervoxels have two important properties; they are **evenly distributed** across the 3D space, and they cannot cross boundaries unless the underlying voxels are spatial connected. The former is accomplished by seeding supervoxels directly in the cloud, rather than the projected plane, while the latter uses an octree structure, which maintains adjacency information of leaves. Supervoxels maintain adjacency relations in voxelized 3D space; specifically, **26-adjacency** that is neighboring voxels are those that share a face, edge, or vertex

(b) Supervoxel connectivity graphs can be used at the basis of a hierarchical tree segmentation structure in several levels, varying from coarse to fine. Such structures represent object segmentation at different scales of object-connectivity, from the super-voxel graph up to the scene level. Could you explain the advantage of using graphs based on over-segmentation in fine elements (super-voxels) for video object segmentation and tracking in stream data (RGBD + time).

Temporal video segmentation (or RGBD stream data segmentation) should keep spatiotemporal homogeneity. Propagating and temporally connecting enriched frame graph representations, such as higher level tree structures, may help out in keeping temporal coherence (temporal stability) of segmented objects along time. Nodes may represent meaningful (and possibly heterogeneous) primitives, and edges represent relationships and hierarchies, where data pooling enriches edges and nodes with features and classes easing segmentation, classification and tracking of stream data.