AVG Practical Session 2 : Two-views Geometry

Fall Semester

1 The Fundamental Matrix

1.1 Basic Properties

This practical session has the following goal : you will practically verify some basic properties of the fundamental matrix F that allow us to take advantage of multiple views when performing structure computation. The presented treatment is essentially due to [Hartley et Zisserman, 2004].

First of all, let's recall the basic properties of the fundamental matrix :

- $\mathtt{F}$ is a homogenous matrix of rank 2 with 7 DOF.
- The epipolar line (see Fig. 1)
  $l' = \mathtt{F}\mathbf{x}$ is the line corresponding to $\mathbf{x}$
  $l = \mathtt{F}^{\mathsf{T}}\mathbf{x}'$ is the epipolar line corresponding to $\mathbf{x}'$
- Epipoles :
  $\mathtt{F}e = \mathbf{0}$
  $\mathtt{F}^{\mathsf{T}}e' = \mathbf{0}$
- Calculation of the camera matrices $\mathtt{P}$ and $\mathtt{P}'$:
  $\mathtt{P} = \mathtt{K}\left[\ \mathtt{I} \mid \mathbf{0}\ \right]$, $\mathtt{P}' = \mathtt{K}'\left[\ \mathtt{R} \mid \mathbf{T}\ \right]$, $\mathtt{F} = \mathtt{K}'^{-\mathsf{T}}[\mathbf{T}]_{\times}\mathtt{R}\mathtt{K}^{-1}$
- If $\mathbf{x}$ et $\mathbf{x}'$ are corresponding points between the two images, then $\mathbf{x}'^{\mathsf{T}}\mathtt{F}\mathbf{x} = 0$.

This last property allows us to estimate F from pairs of correspondent points in two images.

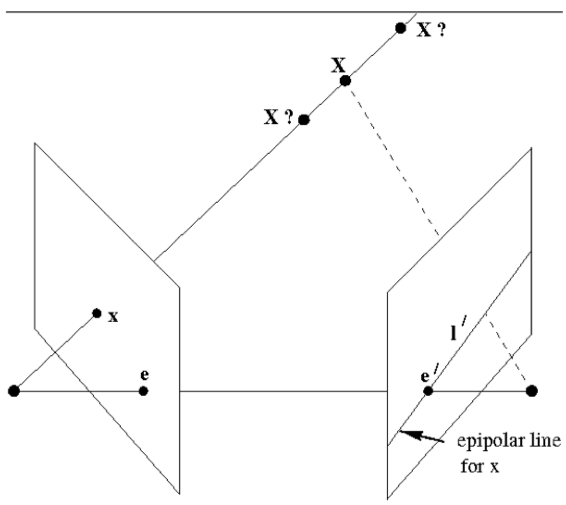


FIGURE 1 – A point $\mathbf{x}$ in the first image back-projects to a ray in the 3D space, defined by the camera centre $\mathbf{C}$ and $\mathbf{x}$. This ray is imaged as an epipolar line in the second image.

1.1.1 Practical

Data Acquisition : The two images to be loaded are *chapel00.png* and *chapel01.png*. They are geometrically linked by the fundamental matrix F saved in the file *chapel.00.01.F* (to load with *F=numpy.loadtxt('chapel.00.01.F')*).
Compute $\mathbf{l}'$: Extract a point $\mathbf{x}$ from the first image (function *ginput()* from *pyplot* in *matplotlib*) and compute the epipolar line in the second image (see Fig. 2). You can draw the line using *plot()* from *pyplot* in *matplotlib*, or you can use drawing functions from OpenCV (*cv2.line()*).
Compute $\mathbf{e}'$: compute the epipole $\mathbf{e}'$ as the right null vector of F^T (using SVD) and verify that the epipolar lines pass through the epipoles.

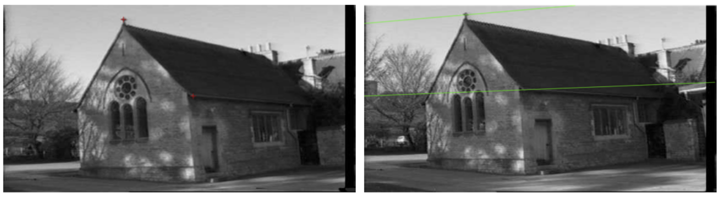


Figure 2 – The marked points in the first image correspond to two epipolar lines in the second.

1.2 Estimation of the Fundamental Matrix

The Fundamental Matrix is defined by the equation :

$$\mathbf{x}'^T \mathrm{F}\mathbf{x} = 0 \tag{1}$$

Given a pair of correspondent points $\mathbf{x} \leftrightarrow \mathbf{x}'$, the constraint 1 gives :

$$x'xf_{11} + x'yf_{12} + x'f_{13} + y'xf_{21} + y'yf_{22} + y'f_{23} + xf_{31} + yf_{32} + f_{33} = 0 \tag{2}$$

Which gives in matrix form for $n(n > 8)$ pairs of correspondences :

$$\mathrm{A}\mathbf{f} = \begin{pmatrix} x'_1x_1 & x'_1y_1 & x'_1 & y'_1x_1 & y'_1y_1 & y'_1 & x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x'_nx_n & x'_ny_n & x'_n & y'_nx_n & y'_ny_n & y'_n & x_n & y_n & 1 \end{pmatrix} \begin{pmatrix} f_{11} \\ f_{12} \\ f_{13} \\ f_{21} \\ f_{22} \\ f_{23} \\ f_{31} \\ f_{32} \\ f_{33} \end{pmatrix} = \mathbf{0} \tag{3}$$

We can compute a least square solution by computing the vector of elements $\mathbf{f}$ as the null vector of A using SVD factorization.

1.2.1 The 8-Point Algorithm

Given $n \geq 8$ pairs of corresponding points $\mathbf{x_i} \leftrightarrow \mathbf{x'_i}$, the 8-Point Algorithm can be implemented in order to determine the Fundamental Matrix F such that $\mathbf{x'_i}^T \mathrm{F}\mathbf{x_i} = 0$.
The 8-Point Algorithm (Algo. 1) is very simple but provides good results if a preconditioning step is carried out before formulating Eq. 3. The proposed normalization consists

in translating and scaling the points in order to have zero mean and RMS distance equal to $\sqrt{2}$. See [Hartley and Zisserman, 2004].

1.2.2 Practical

FEATURE POINTS DETECTION : In the script *lab2.py*, the instruction to extract features (here ORB features) and to match them between the two views, are provided (results like in Fig. 3). Think about a matrix structure to store the matched feature coordinates between the two views, and do not forget to convert them in homogenous coordinates.
NORMALIZATION : From the matched features, compute the T and T′ normalizing transformation matrices and apply them to the corresponding points saved in your matrix structure.
ESTIMATE F : Estimate the fundamental matrix F using the 8-Point Algorithm.
RANSAC ESTIMATION OF F : Estimate the fundamental matrix F using function in OpenCV with RANSAC (*cv2.findFundamentalMat()*).
TEST : Verify the result with the groundtruth Fundamental Matrix(e.g. as in Section 1.1.1).

Algorithme 1 8-Point Algorithm [Hartley, 1997]

Preconditioning : normalize point coordinates in the two images according to $\hat{\mathbf{x}}_i = \mathrm{T}\mathbf{x}_i$ and $\hat{\mathbf{x}}'_i = \mathrm{T'}\mathbf{x}'_i$. **Estimation** : Compute the fundamental matrix $\hat{\mathrm{F}}'$ for the normalized points :

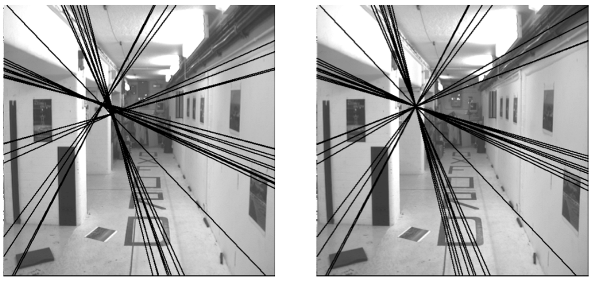
1. **Linear solution** : determine $\hat{\mathrm{F}}$ from the last column of V associated to the smallest singular value obtained from the SVD of $\mathrm{A} = \mathrm{USV}^T$ (See Eq. 3).
2. **Singularity Constraint Enforcement** : substitute $\hat{\mathrm{F}}$ with $\hat{\mathrm{F}}'$ such that the Frobenius norm $\left\|\hat{\mathrm{F}} - \hat{\mathrm{F}}'\right\|_F$ is minimum by imposing $\det\left(\hat{\mathrm{F}}'\right) = 0$ (See reminder and Fig. 4).

De-normalisation : The matrix $\mathrm{F} = \mathrm{T}'^T\hat{\mathrm{F}}'\mathrm{T}$ is the fundamental matrix for original points pairs $\mathbf{x_i} \leftrightarrow \mathbf{x'_i}$.

(a) chapel00

(b) chapel01

FIGURE 3 – The points are extracted from *chapel00.png* and *chapel01.png* using the Harris Detector [Harris and Stephens, 1988] and matched using a similarity measure ; Same color means corresponding point ; Many false matching are removed by the robust estimation of F using RANSAC estimation.



(a) (b)

FIGURE 4 – In (a), F does not respect its singularity ; (b) shows the result after imposing the rank-2 constraint.

Disparity Map and 3D Reconstruction

Here you will use two rectified images from the KITTI dataset : *KITTI_Left.png* and *KITTI_Right.png*.

FIGURE 5 – Images from KITTI.

The corresponding projection matrices and the calibration matrix are :

$$\mathtt{P}_{left} = \begin{bmatrix} 7.215377e+02 & 0 & 6.095593e+02 & 0 \\ 0 & 7.215377e+02 & 1.728540e+02 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathtt{P}_{right} = \begin{bmatrix} 7.215377e+02 & 0 & 6.095593e+02 & -3.875744e+02 \\ 0 & 7.215377e+02 & 1.728540e+02 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathtt{K}_{left} = \mathtt{K}_{right} = \begin{bmatrix} 7.215377e+02 & 0 & 6.095593e+02 \\ 0 & 7.215377e+02 & 1.728540e+02 \\ 0 & 0 & 1 \end{bmatrix}$$

— Load the two images.
— Perform dense stereo matching using the OpenCV function *cv2.StereoBM_ create()* and *compute()*.
— From the disparity map and using the projection and calibration matrices, compute the 3D triangulation to obtain the 3D point cloud.
— Visualize the 3D point cloud using *plot()* from *matplotlib*. Be careful to filter 3D points that are too far, keep points in a 3D volume as $x \in [-10m : +10m]$, $y \in [-5m : +5m]$ and $z \in [+5m : +30m]$.

Reminder

Approximating a matrix M with the closer matrix $\tilde{\mathrm{M}}$ under Frobenius norm, such that $rank\left(\tilde{\mathrm{M}}\right) = r$, can be achieved using the SVD : If $\mathrm{M} = \mathrm{U}\Sigma\mathrm{V}^T$, then $\tilde{\mathrm{M}} = \mathrm{U}\tilde{\Sigma}\mathrm{V}^T$ where $\tilde{\Sigma}$ is the same matrix as Σ except that it contains only the r largest singular values (all other singular values are zero).

References

[Hartley and Zisserman, 2004] Hartley, R. I. and Zisserman, A. (2004). Multiple View Geometry in Computer Vision. Cambridge University Press, ISBN : 0521540518, second edition.
[Harris and Stephens, 1988] Harris, C. and Stephens, M. (1988). A combined corner and edge detector. In Alvey vision conference, volume 15, page 50. Manchester, UK.
[Hartley, 1997] Hartley, R. I. (1997). In defense of the eight-point algo- rithm. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 19(6) :580–593.