AVG Practical Session 1: Projective Geometry

Fall Semester

1 Projective 2D geometry

1.1 Affine Rectification

The goal is to find the transformation that performs affine rectification of a plane in the image. Subsequently, apply the found transformation to warp the image. We follow the procedure proposed in [1]. Affine properties can be recovered simply specifying a line (2 dof). The line at infinity $\mathbf{l}_\infty$ is a fixed line under projective transformation $\mathtt{H}$ if and only if $\mathtt{H}$ is an affinity. Referring to Fig. 1 the projective transformation $\mathtt{H}_p$ moves the line at infinity from its canonical position on the Euclidean plane π_1 to a finite location on the plane π_2. If rectification is carried out by $\mathtt{H}'_p$ in order to move the line at infinity to its canonical position, then, since $\mathbf{l}_\infty$ is preserved by the projective transformation $\mathtt{H}_A = \mathtt{H}_p\mathtt{H}'_p$ from π_1 to π_3, $\mathtt{H}_A$ is an affine transformation, i.e. affine properties of the first plane can be measured on the third.

The projective matrix that transforms $\mathbf{l}'_\infty$ to its canonical position $\mathbf{l}_\infty = (0, 0, 1)^T$ can be applied to every point location in the image in order to affinely rectify the image.

Affine Rectification

1. Find the imaged line at infinity $\mathbf{l}'_\infty = (l_1, l_2, l_3)^T$ in the image.
2. Find a transformation matrix $\mathtt{H}$ which maps $\mathbf{l}'_\infty$ back to its nominal position.
3. Use $\mathtt{H}$ to projectively warp the image.

Once computed, the projective transformation can be applied to every pixel location to obtain the warped image. A better result can be obtained by the so called inverse warping (see Fig. 2). To do so, you will have to use the Opencv Python function *cv2.warpPerspective()*.

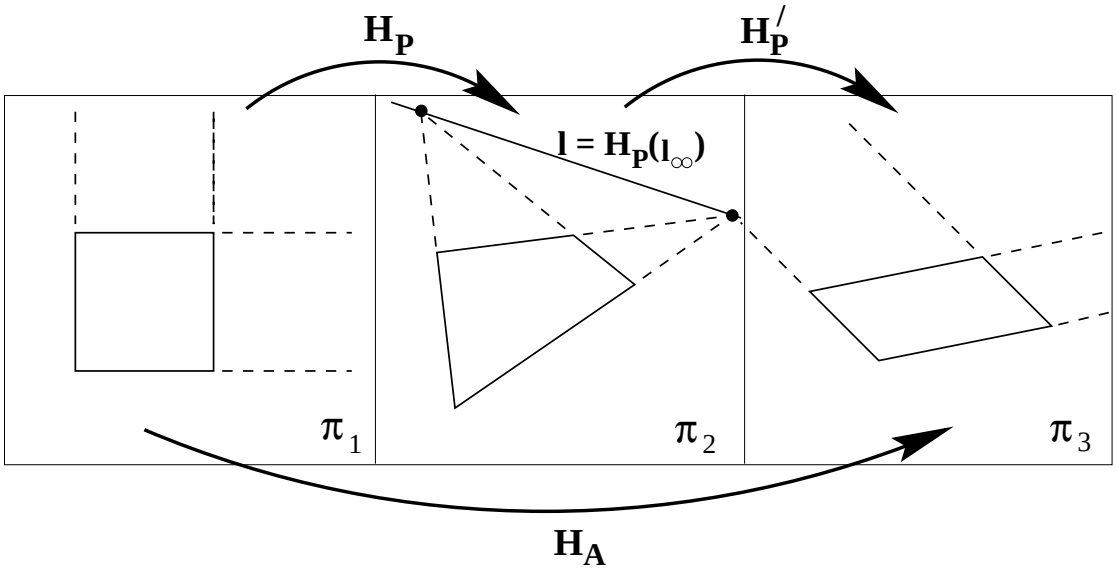

FIGURE 1 – Transformations

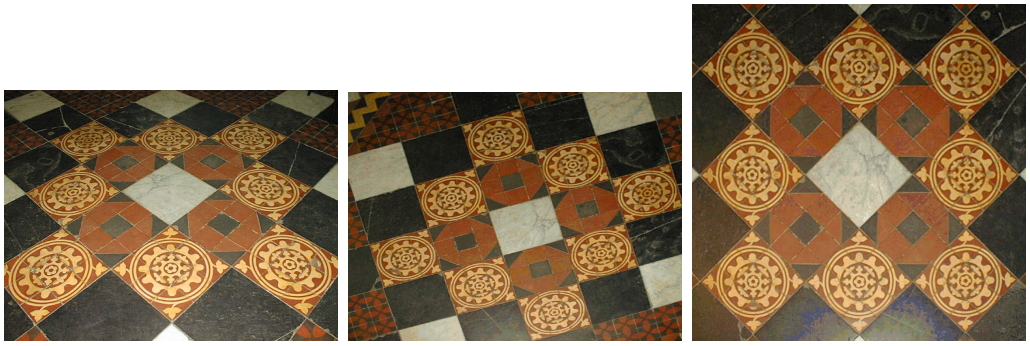

FIGURE 2 – The first image is affected by perspective distortion. The second has been affinely rectified by image warping. Note that affine properties can be measured in the warped image (e.g. parallelism has been recovered). The last image has been obtained by metric rectification.

Practical

You are asked to go through the following steps using Python programming :

— **Acquire the image** : Load the images *tilef.jpg* or *fig1_ 6c.jpg* and code the work in Python using the file *lab1.py*.

— **Extract the lines** : By clicking in a circular way on the image using *ginput()* from *pyplot* in *matplotlib*.

— **Affine Rectification** : As seen during the lecture, rectify the image in order to recover affine properties (e.g. line parallelism).

Reminder

— The line at infinity $\mathbf{l}_\infty$ can be obtained from a pair of vanishing points (e.g. intersections of "parallel" lines).
— The line $\mathbf{l}$ joining two (homogeneous) points $\mathbf{x}_1$ and $\mathbf{x}_2$ is

$$\mathbf{l} = \mathbf{x}_1 \times \mathbf{x}_2$$

— To get the point $\mathbf{x}$ as the intersection of the lines $\mathbf{l}_1$ and $\mathbf{l}_2$:

$$\mathbf{x} = \mathbf{l}_1 \times \mathbf{l}_2$$

— You can find lines in an image by using the Hough transform (*cv2.HoughLines()* function).

1.2 Metric Rectification

The goal is to find the transformation that performs metric rectification using a pair of orthogonal lines in the image. Subsequently, apply the found transformation to warp the image. In the previous section, we recovered affine properties simply specifying a line (2 dof). The line at infinity $\mathbf{l}_\infty$ is a fixed line under projective transformation $\mathtt{H}$ if and only if $\mathtt{H}$ is an affinity. In a similar way, circular points and their dual, the conic $\mathtt{C}^*_\infty$, are fixed under similarity transformations. In particular, we use the following result : the dual conic $\mathtt{C}^*_\infty$ is fixed under the projective transformation $\mathtt{H}$ if and only if $\mathtt{H}$ is a similarity. The homography that transforms $\mathtt{C}^*_\infty$ to its canonical position

$$\mathtt{C}^*_\infty = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix},$$

can be applied to every point in the image in order to perform a metric rectification. As explained during the lectures, the objective here is about computing such a transformation. The dual conic $\mathtt{C}^*_\infty$ can be identified using a pair of orthogonal relations. Suppose the lines $\mathbf{l}'$ and $\mathbf{m}'$ in the *affinely rectified* image correspond to a orthogonal line pair $\mathbf{l}$ and $\mathbf{m}$ on the world plane. Then, $\mathtt{C}^*_\infty$ can be computed as follows :

Dual Conic Identification

1. Stack a pair of constrains on orthogonal lines in the form

$$(l'_1 m'_1, l'_1 m'_2 + l'_2 m'_1, l'_2 m'_2)\mathbf{s} = 0$$

and determine $\mathbf{s}$ as the null vector.

2. Build the 2×2 matrix

$$\mathtt{S} = \begin{pmatrix} s_1 & s_2 \\ s_2 & s_3 \end{pmatrix},$$

using the $s_i, i = 1 \ldots 3$ elements of $\mathbf{s}$.

3. Build the matrix of the imaged Dual Conic :

$$\mathtt{C}'^{*}_{\infty} = \begin{pmatrix} \mathtt{S} & \mathbf{0} \\ \mathbf{0}^T & 0 \end{pmatrix},$$

Metric Rectification

1. Find the imaged dual conic $\mathtt{C}'^{*}_{\infty}$ in the image as explained above.
2. Compute the transformation $\mathtt{U}$ that moves $\mathtt{C}'^{*}_{\infty}$ to its canonical position

$$\mathtt{C}'^{*}_{\infty} = \mathtt{U} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} \mathtt{U}^T$$

using the SVD of $\mathtt{C}'^{*}_{\infty}$.

3. Use $\mathtt{U}$ to projectively warp the image. Once computed, the projective transformation can be applied to every pixel to obtain the warped image. Again, a better result can be obtained by the so called inverse warping.

Practical

You are asked to go through the following steps :

— **Metric Rectification** : Metrically rectify the affinely rectified image (previous section) in order to recover metric properties (e.g. line perpendicularity) according to the explained steps above using Python programming.

Références

[1] R. I. Hartley and A. Zisserman. Multiple View Geometry in Computer Vision. Cambridge University Press, ISBN : 0521540518, second edition, 2004.