POLITECNICO MILANO 1863

IACV Homework
2021-2022

Alessandro Polidori - 10573078 / 963429

Table of Contents

1 Introduction

An image of the back side of Villa Melzi d'Eril is taken by a digital camera. The camera skew factor is assumed to be null; the aspect ratio is unknown (thus natural camera can not be assumed), as well as the principal point and the focal distance. The camera height (z-coordinate) over the ground plane is 1.5 m.

The back side of Villa Melzi d'Eril in Bellagio. In the bottom view below, all lines lie on a same horizontal plane. In addition the facades 1 and 5 are coplanar, and both are parallel to facade 3. Furthermore, facades 2 and 4 are perpendicular to facades 1, 3 and 5. The sun is placed at the point at the infinity S=[3.9 -1 z 0] , where z is irrelevant wrt to the reference frame reported in the picture. The layout of the architectonic elements on facade 3 is symmetric with respect to a central vertical axis. Windows of facades 1, 3 and 5 are are equally wide. Windows of facades 2 and 4 can not be expoited in reconstruction since they are poorly visible.

Figure 1: starting image

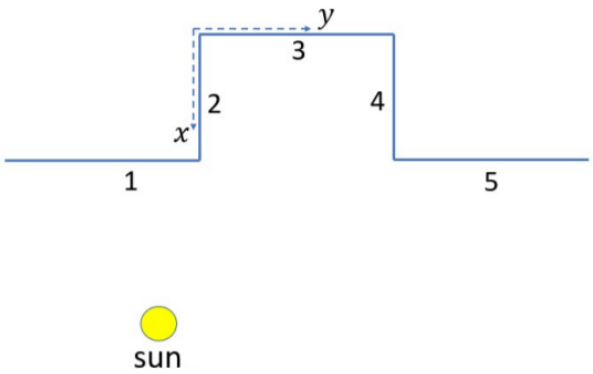


Figure 2: horizontal pov and sun direction

2 Image processing

To extract lines from the image i followed the steps reported below:

- rgb to gray conversion
- histogram eq to increase contrast
- Canny algorithm to find edges
- Hough transform on Canny's output

2.1 Canny edge detector

Canny algorithm manages to detect edges following five steps:

- smoothing using a Gaussian Filter
- finding intensity gradient of the image
- non maximum suppression (edge thinning)
- double thresholding (weak edges have gradient value between low and high thresholds)
- edge tracking (keeping only the weak edges that are connected to strong ones)

Standard deviation of the Gaussian Filter and thresholding values have been found empirically (SIGMA = 2.5, t-values = [0 , 0.09])

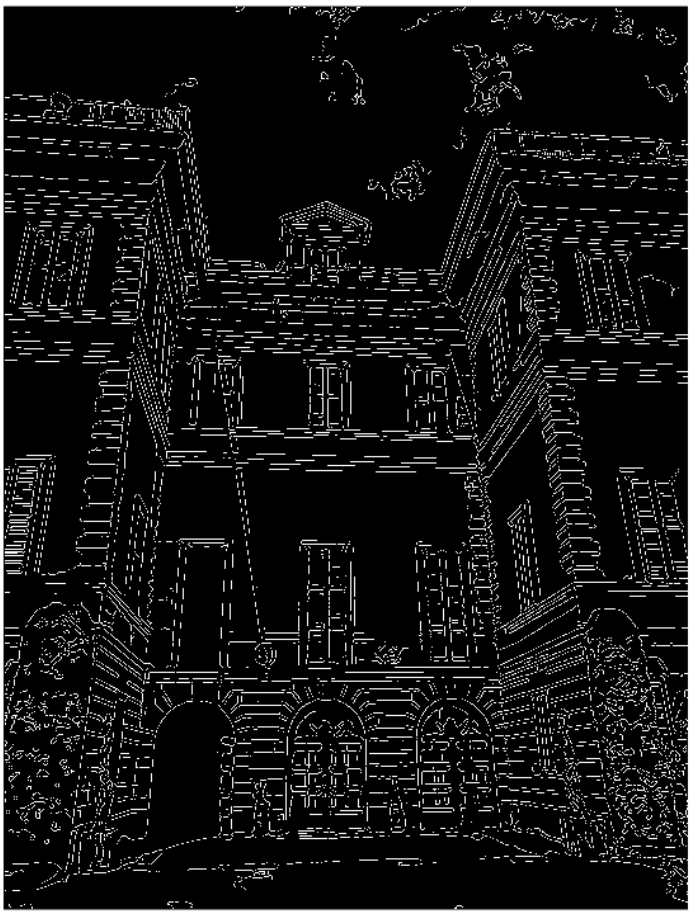


Figure: Canny output

2.2 Hough Transform

To identify lines in the image i used the Hough transform, a famous voting based model fitting technique. To tackle computational problems tied to unbounded angular coefficient "m" of vertical lines, classical y = mx +b is replaced by Hesse normal form:

$$r = x\cos\theta + y\sin\theta \tag{1}$$

where r is the distance from the origin to the closest point of the line and θ is the angle between the line and x-axis. Given a single point in the starting image space, the set of all lines passing through it is represented by a sinusoid in (r,θ)

parameter space. The problem of fitting a line, then, is converted to finding where the various sinusoidal curves associated to points intersect in parameter space. As the Matlab implementation outputs a subset of all the lines, i applied Hough Transform considering different ranges of θ to find all the lines i was interested in.

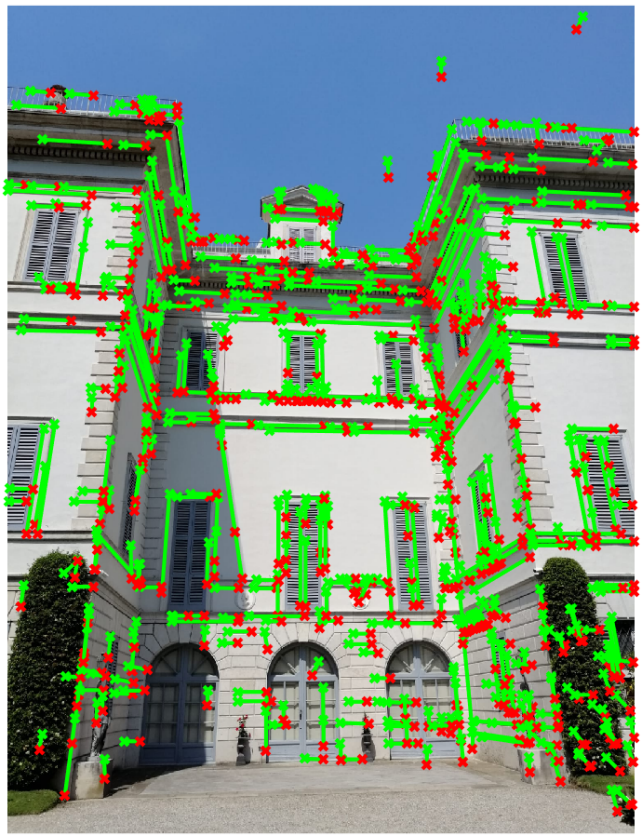


Figure: lines detected

2.3 Corners detection

The best result in corner detection was obtained using SURF algorithm.

Figure: SURF output

3 Shape Reconstruction

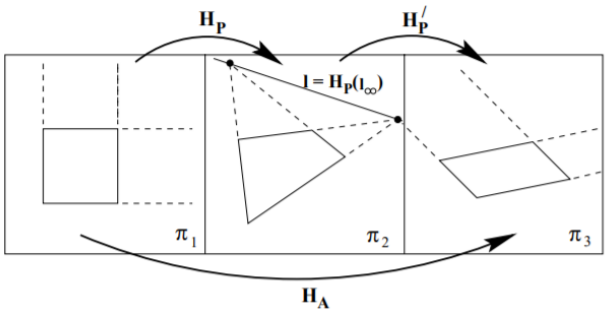
To obtain the shape reconstruction of an horizontal plane i decided to exploit a stratified approach based on two steps:

- Affine reconstruction (from projective to affine)
- Shape reconstruction (from affine to metric)

3.1 Affine Reconstruction

The first step is to recover the affine properties of the original image. Affine transformations preserve ... If the imaged line at the infinity is the line l = (l1 , l2 , l3), then provided l3 = 0 a suitable projective point transformation which will map l back to the real line at the infinity (0,0,1) is (Multiple View Geometry in Computer Vision, p. 49):

$$H = H_A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ l1 & l2 & l3 \end{bmatrix}$$



The image of the line at the infinity can be found using two vanishing points of the X and Z directions (parallel to ground).

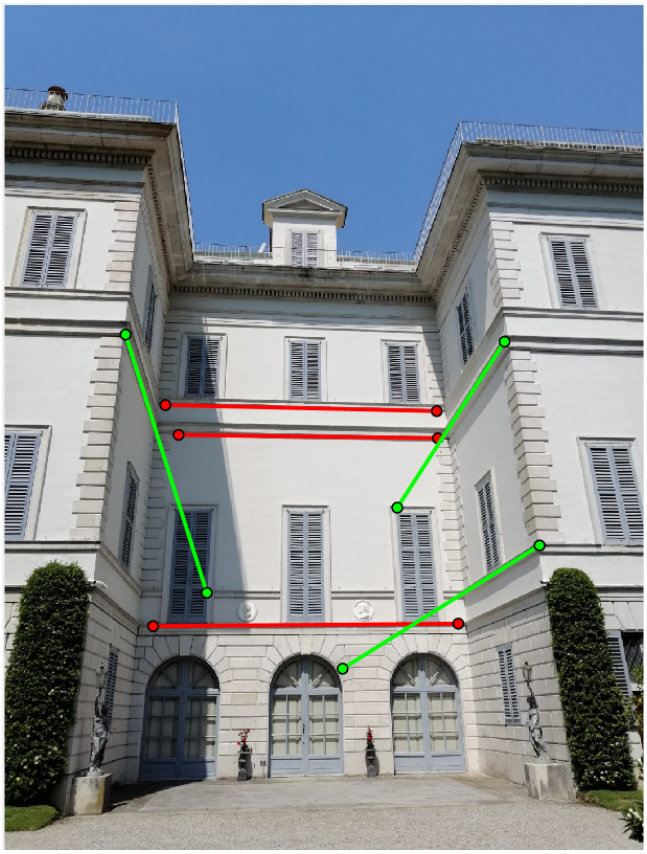


Figure: X and Z directions lines

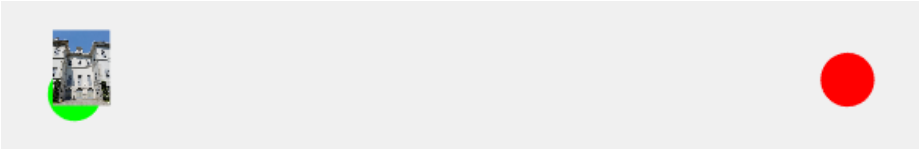


Figure: vanishing points

Figure: Affine rectification output

3.2 Euclidean Reconstruction

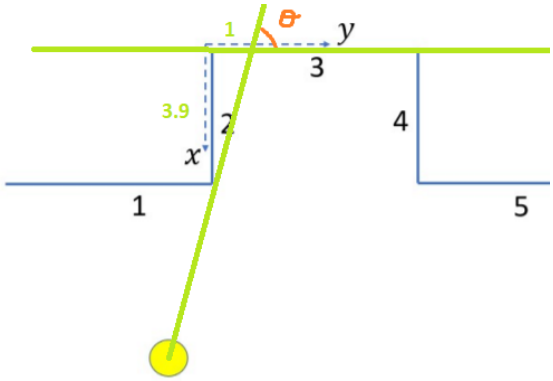
After affine rectification

$$C_{\infty}^{*'} = \begin{bmatrix} KK^T & 0 \\ 0^T & 0 \end{bmatrix}$$

thus there are only two unknowns to identify the image of the dual conic $C_{\infty}^{*'}$. In theory it was possible to use two pairs of orthogonal lines (you will find this solution in my code too). The best result, though, was obtained using a different constraint. I exploited the direction of the sun to set up the equation below:

$$\cos\theta = \frac{\mathbf{l}^{\mathsf{T}}\mathbf{C}_{\infty}^{*}\mathbf{m}}{\sqrt{(\mathbf{l}^{\mathsf{T}}\mathbf{C}_{\infty}^{*}\mathbf{l})(\mathbf{m}^{\mathsf{T}}\mathbf{C}_{\infty}^{*}\mathbf{m})}}$$

This is the same equation that in case of orthogonal lines becomes linear (as cos θ is 0). In our case, we can exploit this pair of lines, where θ should be equal to arctan(3.9):



The resulting ratio between the width of facades 3 and 2 (or 4) is around 1.3.

Figure: Euclidean rectification output

4 Camera Calibration

Camera calibration implies finding

$$K = \begin{bmatrix} f_x & 0 & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix}$$

K is related to the image of the absolute conic through the following equation:

$$w = (K * K^T)^{-1} \tag{2}$$

In our case we can assume that the camera skew factor is null, so we will need one less constraint to fit the IAC.

$$\omega = \begin{bmatrix} \alpha^2 & 0 & -u_0\alpha^2 \\ 0 & 1 & -v_0 \\ -u_0\alpha^2 & -v_0 & f_y^2 + \alpha^2 u_0^2 + v_0^2 \end{bmatrix}$$

Once we get w, we can obtain K exploiting the parametrization of the IAC. To fit w, we can exploit different types of constraints. Knowing the rectifying homography H found in G1, we could fit w to images of circular points:

$$I' = H'I = [h_1, h_2, h_3]I = [h_1, h_2, h_3]\begin{bmatrix} 1 \\ i \\ 0 \end{bmatrix} = h_1 + ih_2 \in \omega$$

$$(h_1 + ih_2)^T \omega (h_1 + ih_2) = 0$$

Real part = 0 and imaginary part = 0

$$h_1{}^T \omega h_2 = 0$$
$$h_1{}^T \omega h_1 - h_2{}^T \omega h_2 = 0$$

Another type of constraint can be derived from the orthogonality between a line at the infinity and a vanishing point. So, in our case, we could use the line at the infinity found during the horizontal rectification and the vertical vanishing point:

$$[l_{inf}]_\times \omega v_p = 0$$

I decided to fit w using only constraints on orthogonal vanishing points (we need at least 4 pairs):

$$v_{p1}^T \omega v_{p2} = 0$$

The additional couple of orthogonal vanishing points has been found using these directions on facade 3 doors:

Numerical result:

```
K =

   1.0e+03 *

    1.2716         0    0.5384
         0    1.1844    0.6399
         0         0    0.0010
```

5 Reconstruction of a vertical facade

In G3 it was requested to rectify a vertical facade (we will see facade 3) using K. To get an euclidean rectification we need to find the image of the conic dual to the circular points

$$C_\infty^{*\prime} = I'J'^T + J'I'^T$$

Thanks to G2, we now have the IAC. And it's easy to prove that

$$\{I', J'\} = l'_\infty \cap \omega$$

So once we have found the imDCCP we can use SVD to extract the rectification homography matrix.

$$\text{svd}(C_\infty^{*\prime}) = U \begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{vmatrix} U^T = H_{rect}^{-1} C_\infty^* H_{rect}^{-T}$$

Figure: vertical rectification

6 Camera Localization

The last request was to determine the relative pose between facade 3 and camera reference. Considering the plane π corresponding to the facade and having as origin the down left corner, a point on that can be written as:

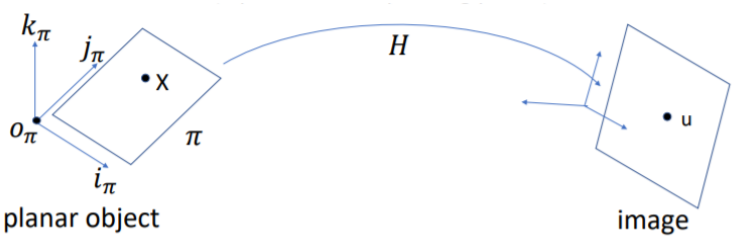
$$X_\pi = \begin{bmatrix} x \\ y \\ 0 \\ w \end{bmatrix}$$

Considering also camera reference frame corresponding to world reference frame:

$$P = [\mathbf{KR} \quad \mathbf{KRt}] = [\mathbf{K} \quad \mathbf{0}]$$

$$\mathbf{u} = PX_w = [\mathbf{K} \quad 0] \begin{bmatrix} i_\pi & j_\pi & o_\pi \\ 0 & 0 & 1 \end{bmatrix} \mathrm{x}_\pi = \mathbf{K}[i_\pi \quad j_\pi \quad o_\pi]\mathrm{x}_\pi$$

plane π to image homography $\quad H = \mathbf{K}[i_\pi \quad j_\pi \quad o_\pi]$



$$[i_\pi \quad j_\pi \quad o_\pi] = \mathbf{K}^{-1}H \qquad (k_\pi = i_\pi \times j_\pi)$$

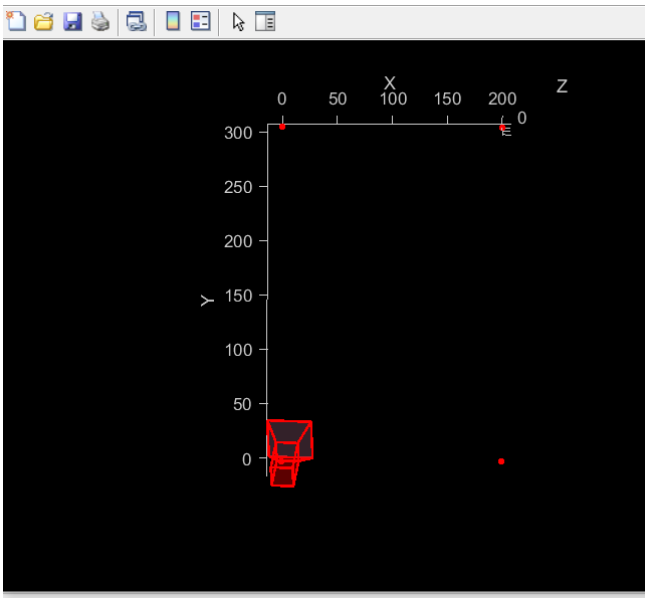
Where H is the homography mapping plane points to image points. We can find H fitting it through 4 points that map a rectangle with the real rateo (we can get it thanks to G3 result) to the corners of facade 3 in the original image.

$$\begin{aligned}
\lambda &= \frac{1}{|K^{-1}h_1|} \\
i_\pi &= K^{-1}h_1\lambda \\
j_\pi &= K^{-1}h_2\lambda \\
k_\pi &= i_\pi \times j_\pi \\
o_\pi &= K^{-1}h_3\lambda
\end{aligned}$$

where lambda is a normalization factor. To localize the camera wrt plane:

$$\begin{bmatrix} i_\pi & j_\pi & i_\pi \times j_\pi & 0_\pi \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1}$$

We still need to solve for scale. The assignment provides us with the height of the camera from the ground. So we can use it to have a final camera y value of 1.5 (z is the axis orthogonal to the facade).



Command Window

```
cameraPosition =

    2.6963
    1.5136
   26.3236
```