SAMUELE MOSCATELLI

IMAGE ANALYSIS AND COMPUTER VISION

- 2020/2021 -

POLITECNICO
MILANO 1863

HOMEWORK DOCUMENTATION

POLIMI COMPUTER SCIENCE AND ENGINEERING A.A. 2020 - 2021

Samuele Moscatelli,
Image Analysis and Computer Vision
Homework Documentation

E-MAIL:
Samuele Moscatelli - samuele1.moscatelli@mail.polimi.it

Documentation realized with LaTeX.

CONTENTS

1 INTRODUCTION

Figure 1: Plane pi

An image of the Castello di Miramare 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 aim is that of extract the main features, so edges, corner features and straight lines. Then starting from some of the information obtained in the first step, the second task is 2D reconstruction of the horizontal plane π (shown in the figure below). The third step is camera calibration, so find the intrinsic parameters, namely focal distance, aspect ratio and position of principal point. Subsequently, the task is camera localization, so identifying position and orientation of the camera. Finally, 2D reconstruction of a vertical facade using the knowledge of the calibration matrix previously found.

2 IMAGE FEATURES EXTRACTION

2.1 PROBLEM FORMULATION

Combining the learned techniques, find edges, corner features and straight lines in the image. Then manually select those features and those lines, that are useful for the subsequent steps.

2.2 EDGE DETECTION

2.2.1 Solution Approach

Edge detection has been performed through the Canny detector. The Canny edge detector is an edge detection operator that uses a multi-stage algorithm to detect a wide range of edges in images. The basic property of Canny edge detector can be summarized as finding edges by looking for local maxima of the gradient. The gradient is calculated using the derivative of a Gaussian filter.

In order to then being able of performing feature extraction, and in particular to find straight lines, edge detection has been repeated more than one time and each time in a slightly different way. In general, the following procedure has been used:

- The RGB image has been brought to gray scale since the edge function works only on intensity images.
- The gray scale image has been normalized in order to avoid numerical errors.
- In some cases a smoothing filter has been applied to some regions of the image in order to have more smoothed edges in those regions of interest.
- Finally the MATLAB edge function has been applied (the threshold values have been adjusted based on the desired level of detail.

2.2.2 Experimental Results

Given the great importance of edges for the subsequent steps of the homework, the edges function has been applied more than one time (8 times) always with a different initialization of the parameters. Here all the combinations are shown.

Case 1:

$$\text{Threshold} = \begin{bmatrix} 0.09 & 0.1 \end{bmatrix}$$

$$\sigma = 1.4142$$

Figure 2: Case 1 edges

Case 2: The imgaussfilt smoothing function has been applied over the bottom of the image, with standard deviation 5

$$\text{Threshold} = \begin{bmatrix} 0.09 & 0.1 \end{bmatrix}$$

$$\sigma = 1.4142$$

Figure 3: Case 2 edges

Case 3: The imgaussfilt smoothing function has been applied over the image, with standard deviation 7

$$\mathrm{Threshold} = \begin{bmatrix} 0.09 & 0.1 \end{bmatrix}$$

$$\sigma = 1.4142$$

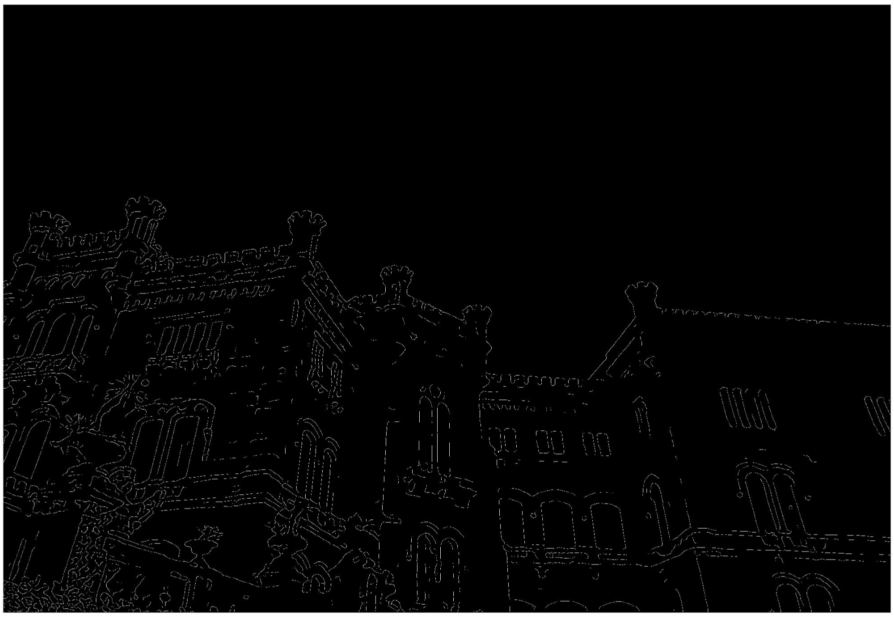

Figure 4: Case 3 edges

Case 4: The image has firstly been equalized with the histeq function and then the imgaussfilt smoothing function has been applied over the right half of the image, with standard deviation 8

$$\mathrm{Threshold} = \begin{bmatrix} 0.09 & 0.5 \end{bmatrix}$$

$$\sigma = 1.4142$$

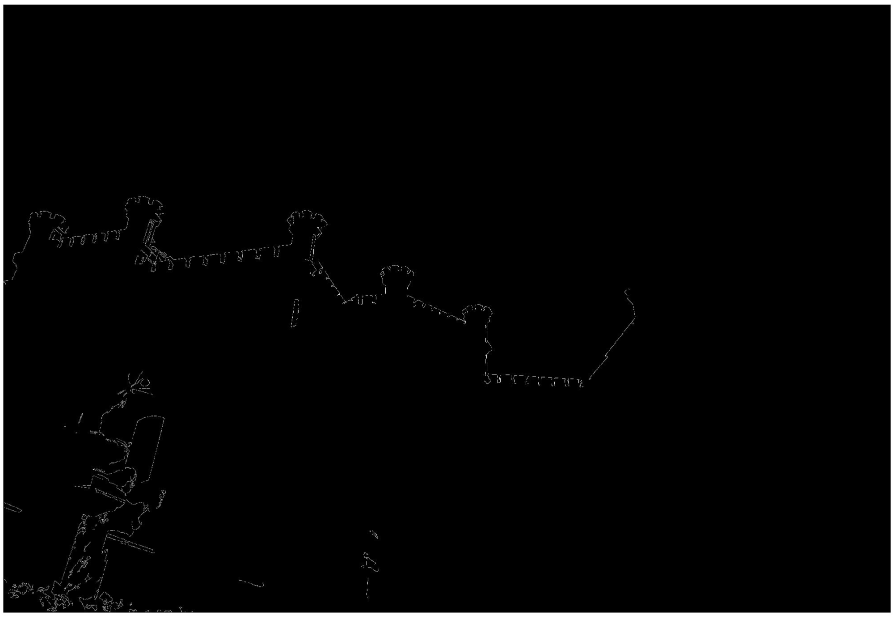

Figure 5: Case 4 edges

Case 5: The imgaussfilt smoothing function has been applied over the bottom part of the image, with standard deviation 4

$$\mathrm{Threshold} = \begin{bmatrix} 0.09 & 0.5 \end{bmatrix}$$

$$\sigma = 1.4142$$

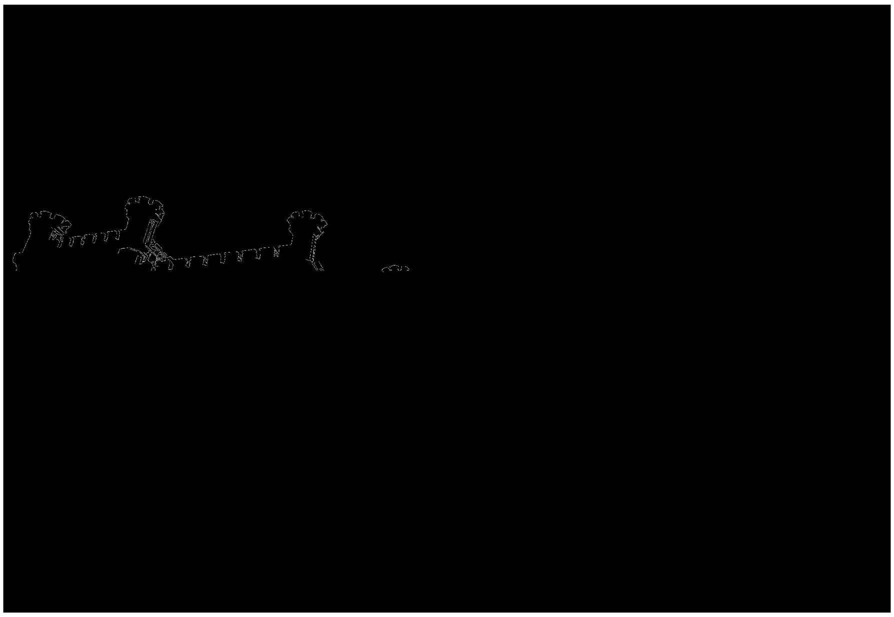

Figure 6: Case 5 edges

Case 6: The image has firstly been equalized with the histeq function and then the imgaussfilt smoothing function has been applied over the bottom part of the image, with standard deviation 8

$$\mathtt{Threshold} = 0.09$$

$$\sigma = 1.4142$$

Figure 7: Case 6 edges

Case 7: The image has firstly been equalized with the histeq function and then the imgaussfilt smoothing function has been applied over the image, with standard deviation 8

$$\mathtt{Threshold} = 0.09$$

$$\sigma = 0.1$$

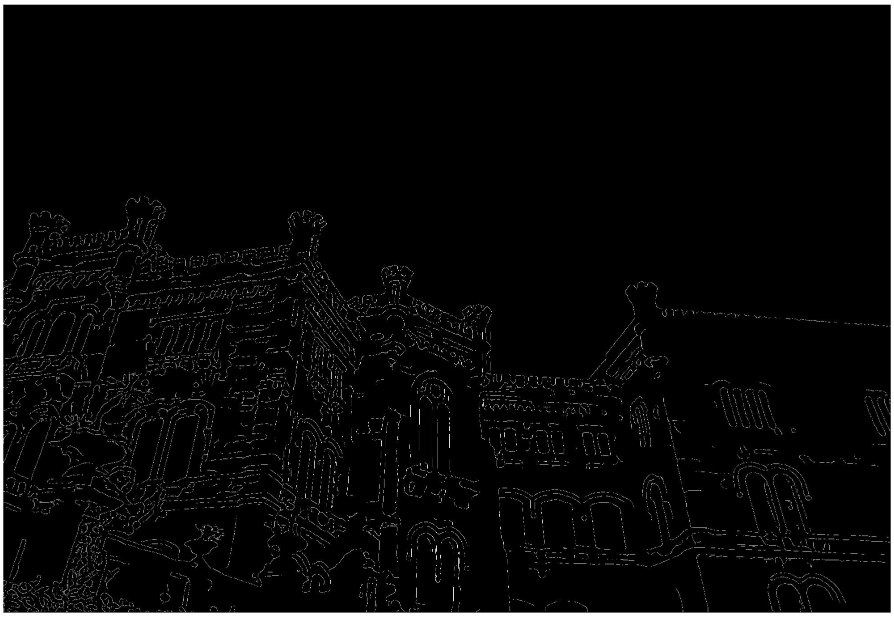

Figure 8: Case 7 edges

Case 8:

$$\text{Threshold} = \begin{bmatrix} 0.0188 & 0.0781 \end{bmatrix}$$
$$\sigma = 1.4142$$

Figure 9: Case 8 edges

2.3 LINE DETECTION

2.3.1 Solution Approach

The Hough transform is designed to detect lines, using the parametric representation of a line:

$$\rho = x * \cos(\theta) + y * \sin(\theta)$$

The variable ρ is the distance from the origin to the line along a vector perpendicular to the line. θ is the angle between the x-axis and this vector. The solution needs the results from the previous step, in particular the edges extracted before. The steps then needed are:

- Compute the Hough transform of the binary image returned by the edge function
- Find the peaks in the Hough transform matrix using the houghpeaks function
- Superimpose a plot on the image of the transform that identifies the peak
- Find lines in the image using the houghlines function
- Create a plot that displays the original image with the lines superimposed on it

2.3.2 Experimental Results

By exploiting all the different edges computed in the previous step the following lines have been carefully found and chosen

Figure 10: Straight lines

2.4 CORNER DETECTION

2.4.1 Solution Approach

A corner is a point whose local neighborhood stands in two dominant and different edge directions. In particular, a corner is the conjunction of two edges and it is invariant to translation, illumination and rotation. In order to detect corners, one of the most simple, efficient and reliable techniques is the Harris Corner Detection algorithm [HS88]. Commonly, it can be divided into five steps:

- Color to gray scale
- Spatial derivative calculation
- Structure tensor setup
- Harris response calculation
- Non-maximum suppression

In particular, what the Harris Corner Detection algorithm does is to continuously consider an image patch and calculate the sum of squared differences (SSD) between the considered patch all the other patches built by shifting the initial one in each possible direction. In this way, if the SSD value is high for each direction, it means that the patch is probably situated in a corner, while if it is low it is situated in a flat area.

So, in order to find the most interesting corners in the homework image it has been used the detectHarrisFeatures function, taking all the corners that reached at least the 10% of the maximum corner metric value in the image, using a filter of size 3 and as region of interest the portion of the image characterized by the Castello Di Miramare building.

2.4.2 Experimental Results

All the found corners are attached to the top of the building

Figure 11: Corners

3 2D RECONSTRUCTION

3.1 PROBLEM FORMULATION

Rectify (2D reconstruct) the horizontal plane π from the useful selected image lines and features, i.e., fix a suitable reference frame solid attached to the horizontal plane π, and determine the coordinates of points and lines relative to the chosen reference frame.

3.2 SOLUTION APPROACH

In order to 2D reconstruct the horizontal plane π, the approach has been that of following a stratified procedure:

- As a first step compute the matrix of the affine transformation which maps the original image to an affine reconstruction with respect to the real scene.
- As second step compute the matrix of the euclidean transformation which maps the affinity computed in the previous step to a euclidean reconstruction.
- As third and final step, compose the two previous transformations to get the matrix of the transformation which directly maps the original image to the euclidean reconstruction of the real scene.

Going deeper into details, an affine transformation (or more simply an affinity) is a non-singular linear transformation followed by a translation. Because an affine transformation includes non-isotropic scaling, the similarity invariants of length ratios and angles between lines are not preserved under an affinity. In particular, invariants of an affine transformation are parallel lines, Ratio of lengths of parallel segments and Ratio of areas.

What has been done to affinely reconstruct the image is to bring back the line at the infinity to its position in the real scene, so to the position $\begin{bmatrix} 0 & 0 & 1 \end{bmatrix}'$. So, the vanishing point of the horizontal plane has been computed by intersecting sets of parallel lines to the plane and then fitting each set with the least square approximation. In this way five vanishing points has been found, one for the long side of the left part of the plane, one for the short side of the left part of the plane (orthogonal to the previous), one for the long side of the right part of the plane, one for the short side of the right part of the plane and a last one which is given by the central part of the plane. Once they have been found it is used again the least squares approximation method in order to fit the image of the line at the infinity, which then allows to build the reconstruction matrix

$$H_{aff} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ l1 & l2 & l3 \end{bmatrix}$$

where the last row is composed by the homogeneous coordinates of the line at the infinity.

Then, to perform metric reconstruction it is used an approach analogous to the one used for the affine rectification, in particular the aim is that of finding the circular points from the plane and bring them back to their canonical position

$$I = \begin{bmatrix} 1 \\ i \\ 0 \end{bmatrix}, J = \begin{bmatrix} 1 \\ -i \\ 0 \end{bmatrix}$$

The transformation between the world plane and the rectified image is a similarity since it is projective and the circular points are fixed. So, in order to get the matrix of the transformation that brings the affine reconstruction to the euclidean reconstruction it is necessary to select two pairs of orthogonal lines such that enough constraints can be derived in order to compute the image of the dual conic to the circular points ($C_\infty^{*'}$). Once two pairs of orthogonal lines have been selected the constraints that allow to find the conic are given by

$$\cos(\theta) = \frac{l^T C_\infty^{*'} m}{\sqrt{(l^T C_\infty^{*'} l)(m^T C_\infty^{*'} m)}}$$

and since theta is known to be of 90° it becomes a linear constraint

$$0 = l^T C_\infty^{*'} m$$

where l and m are a pair of orthogonal lines. So, taking the perpendicular lines previously selected from the image and composing them in all the possible pairs (since more than one line for each direction has been selected) it is then possible to use the least squares approximation method in order to compute the image of the dual conic to the circular points ($C_\infty^{*'}$). Then, after having found the image of the dual conic to the circular points, by applying the SVD decomposition it is possible to find the transformation that maps it to the canonical form H_{can}

$$svd(l^T C_\infty^{*'} m) = USV^T = H_{can} C_\infty^* H_{can}^T$$

Since however the svd method does not return the matrix C_∞^* in its canonical form

$$C_\infty^* = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

it is then necessary to factorize the matrix S obtained from the svd method as

$$S = S_{\sqrt{s}} C_\infty^* S_{\sqrt{s}} = \begin{bmatrix} \sqrt{S[1,1]} & 0 & 0 \\ 0 & \sqrt{S[2,2]} & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \sqrt{S[1,1]} & 0 & 0 \\ 0 & \sqrt{S[2,2]} & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

such that then it is possible to obtain the true transformation that maps the image of the dual conic of the circular points to its canonical form as

$$H_{aff2euc} = H_{can} S_{\sqrt{s}}$$

Having found $H_{aff2euc}$ it is now possible to compute the overall transformation which directly maps the original image to the euclidean reconstruction of the real scene H_r

$$H_r = H_{aff2euc} H_{aff}$$

3.3 EXPERIMENTAL RESULTS

Original image: Starting from the original image

Figure 12: Original image

Reference frame: The following is the reference frame chosen for the rectification task

Figure 13: Reference frame

The coordinates of the shown lines are the following

$$1 : 10^{-4} \cdot \begin{bmatrix} -1.0330 \\ -7.3818 \\ 10000 \end{bmatrix}$$

$$2 : 10^{-4} \cdot \begin{bmatrix} -15.2702 \\ 9.1740 \\ 10000 \end{bmatrix}$$

$$3 : 10^{-4} \cdot \begin{bmatrix} 6.9736 \\ -14.6029 \\ 10000 \end{bmatrix}$$

$$4 : 10^{-4} \cdot \begin{bmatrix} 0.42119 \\ -6.0370 \\ 10000 \end{bmatrix}$$

$$5 : 10^{-4} \cdot \begin{bmatrix} -2.5407 \\ -1.8379 \\ 10000 \end{bmatrix}$$

$$6 : 10^{-4} \cdot \begin{bmatrix} 0.71436 \\ -8.1404 \\ 10000 \end{bmatrix}$$

The set of lines that have been used for the subsequent steps (comprising also lines parallel to the reference frame) is shown in the next figure

Figure 14: Lines parallel to the reference frame

Affine rectification: It has been found the matrix of the affine transformation which maps the original image to an affine reconstruction with respect to the real scene

$$H_{aff} = 10^{-5} \cdot \begin{bmatrix} 100000 & 0 & 0 \\ 0 & 100000 & 0 \\ -2.4628 & -9.0882 & 100000 \end{bmatrix}$$

which then, applied to the original image brings to

Figure 15: Affine rectification

Metric reconstruction: Subsequently the matrix of the euclidean transformation which maps the affinity computed in the previous step to a euclidean reconstruction has been found

$$H_{aff2euc} = \begin{bmatrix} -0.5245 & -0.0656 & 0 \\ -0.1267 & 1.0129 & 0 \\ 0 & -0 & 1 \end{bmatrix}$$

which then, rotated by 180° and multiplied by the matrix of the affine transformation, gives the overall transformation which directly maps the original image to the euclidean reconstruction of the real scene H_r

$$H_r = \begin{bmatrix} -0.5245 & -0.0656 & 0 \\ 0.1267 & -1.0129 & 0 \\ 0 & -0 & -1 \end{bmatrix}$$

and applying it to the original image, it brings to

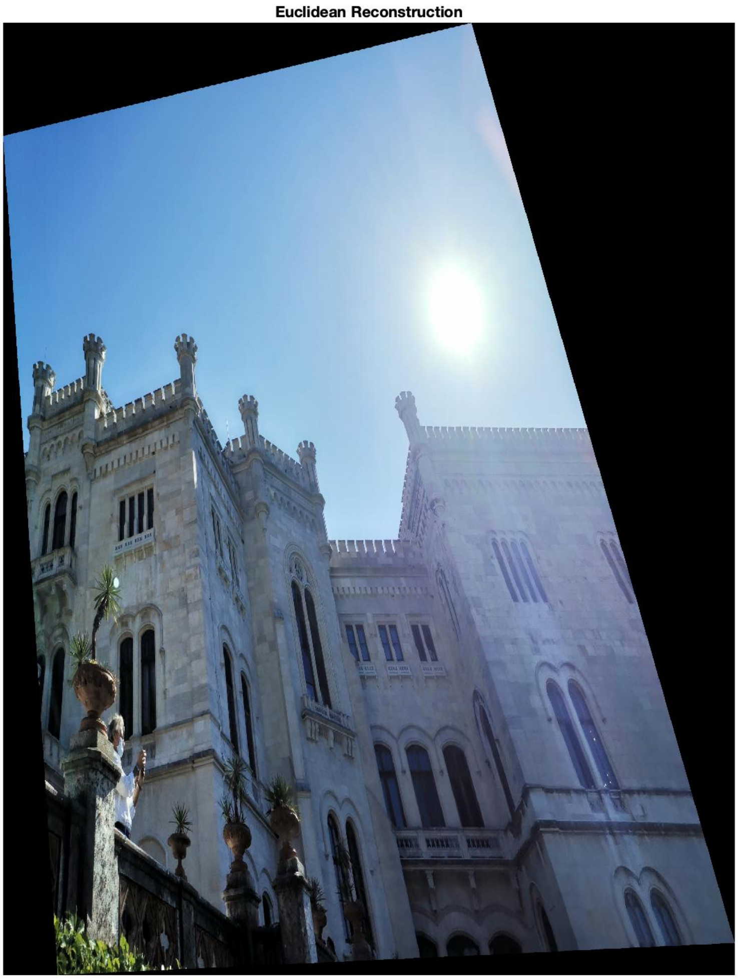

Figure 16: Metric reconstruction

4 CAMERA CALIBRATION

4.1 PROBLEM FORMULATION

First extract a vertical vanishing point and then use it together with useful information extracted during the rectification step, in order to estimate the calibration matrix K containing the intrinsic parameters of the camera, namely focal distance, aspect ratio and position of principal point.

4.2 SOLUTION APPROACH

Camera calibration means finding the matrix

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

where f_x, f_y, u_0 and v_0 are respectively the focal distance on the x axis, the focal distance on the y axis ad the two coordinates of the principal point.

In order to determine K, three information are needed:

- The reconstructing homography (from original to rectified) H_r
- The image of the line at the infinity l'_∞
- The vanishing point v along the direction orthogonal to the face

In fact, through this data it is then possible to compute ω (the image of the absolute conic), from which the camera intrinsic parameters can be extracted since the matrix ω has the form

$$\omega = \begin{bmatrix} \alpha^2 & 0 & -u0\alpha^2 \\ 0 & 1 & -v_0 \\ -u_0\alpha^2 & -v_0 & f_y^2 + \alpha^2 u_0^2 + v_0^2 \end{bmatrix} \tag{2}$$

(alternatively it would be also possible to apply the Cholensky factorization over ω in order to obtain the matrix K). Since it is a symmetric matrix with four degrees of freedom, at least four constraints are necessary in order to find it. The first two constraints are given by

$$l'_\infty = \omega v \tag{3}$$

This is because, given two vanishing points v_1 and v_2 from two independent directions it is true that

$$v_1^T \omega v = 0$$

$$v_2^T \omega v = 0$$

moreover

$$l'_\infty = v_1 \times v_2$$

so, combining the three equations, (3) can be derived.

The other two constraints can be derived through the homography method ([HZ03], [Vin20]): having the reconstructing homography H_r it is possible to compute the image of the circular points for the plane of the rectified face as

$$I' = H_r^{-1} I = H_r^{-1} \begin{bmatrix} 1 \\ i \\ 0 \end{bmatrix} = \begin{bmatrix} h_1 h_2 h_3 \end{bmatrix} \begin{bmatrix} 1 \\ i \\ 0 \end{bmatrix} = h_1 + i h_2$$

then, since the circular points lie on ω

$$(h_1 + i h_2)\omega(h_1 + i h_2) = 0$$

from which two linear equations can be derived

$$h_1^T \omega h_2 = 0$$

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

Once ω has been computed it is then possible to derive the calibration matrix K using the parametrization shown in equations (1) and (2).

4.3 EXPERIMENTAL RESULTS

After having normalized the line at the infinity, the vertical vanishing point and reconstructing homography H_r with respect to the size of the image through the following matrix

$$H_{scaling} = \begin{bmatrix} \frac{1}{ImgMaxSize} & 0 & 0 \\ 0 & \frac{1}{ImgMaxSize} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

the results obtained are

$$l_{inf} = \begin{bmatrix} -0.0977 \\ -0.3606 \\ 1 \end{bmatrix}$$

$$v = \begin{bmatrix} 0.4885 \\ -0.7724 \\ 1 \end{bmatrix}$$

$$H_{r-scaled} = 10^{-4} \cdot \begin{bmatrix} -4.7309 & 0.30652 & 0 \\ -0.59197 & -2.4497 & 0 \\ -0.67579 & -0.85344 & -1 \end{bmatrix}$$

From them, using the method explained in the previous section, ω (the image of the absolute conic) has been computed, obtaining

$$\omega = \begin{bmatrix} 0.3917 & 0 & -0.4964 \\ 0 & 1 & -0.3535 \\ -0.4964 & -0.3535 & 3.0916 \end{bmatrix}$$

from which it is possible to get the camera intrinsic parameters and so, subsequently the normalized calibration matrix K. After denormalizing K with $H_{scaling}$ matrix, the obtained results are

$$K = 10^3 \cdot \begin{bmatrix} 9.6936 & 0 & 5.0293 \\ 0 & 6.0665 & 1.4026 \\ 0 & 0 & 0.001 \end{bmatrix}$$

$$f_x = 10^3 \cdot 9.6936$$

$$f_y = 10^3 \cdot 6.0665$$

$$u_0 = 10^3 \cdot 5.0293$$

$$v_0 = 10^3 \cdot 1.4026$$

$$\alpha = 1.5979$$

5 CAMERA LOCALIZATION

5.1 PROBLEM FORMULATION

Determine the relative pose (i.e. position and orientation) between the reference attached to the horizontal plane π and the camera reference.

5.2 SOLUTION APPROACH

After the reconstruction, the shape of the upper face, so of the plane π, is known, and consequently it is possible to perform camera localization. In particular, it has been considered a rectangle posed over the left side of π as a reference, giving it fictitious measures for the sides, but keeping the right ratio between the long side and the short side. Once done this, it is possible to identify the position of a point in the world reference frame as $X_W = \begin{bmatrix} R_\pi | o_\pi \end{bmatrix} X_\pi$ where X_π is the position of the point in the reference plane and can be expressed as

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

Then using the projection matrix

$$P = \begin{bmatrix} KR | -KR_o \end{bmatrix}$$

it is possible to obtain

$$u = PX_W = \begin{bmatrix} KR | -KR_o \end{bmatrix} \begin{bmatrix} R_\pi | o_\pi \end{bmatrix} X_\pi$$

where u is the corresponding point in the original image. Then putting the world reference frame on the camera ($R = I$ and $o = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T$):

$$u = \begin{bmatrix} K | 0 \end{bmatrix} \begin{bmatrix} i_\pi & j_\pi & o_\pi \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix} = K \begin{bmatrix} i_\pi & j_\pi & o_\pi \end{bmatrix} X_\pi$$

consequently

$$\begin{bmatrix} i_\pi & j_\pi & o_\pi \end{bmatrix} = K^{-1} H$$

where $H = \begin{bmatrix} h_1 & h_2 & h3 \end{bmatrix}^T$ is the transformation that maps the real points to the image. Since the shape and the size of the chosen reference is known, also H is known because it can be found as the transformation that maps the shape of the reference object to the image. Once H has been obtained, the rotation matrix R_π can be also obtained as

$$R_\pi = \begin{bmatrix} i_\pi & j_\pi & k_\pi \end{bmatrix}$$

where

$$i_\pi = K^{-1} h_1 \lambda$$
$$j_\pi = K^{-1} h_2 \lambda$$
$$k_\pi = i_\pi \times j_\pi$$

with λ normalization coefficient defined as

$$\lambda = \frac{1}{|K^{-1} h_1|}$$

Then, applying SVD to R_π in order to avoid noise in the data and get the true rotation matrix

$$\begin{bmatrix} U & S & V \end{bmatrix} = svd(R_\pi)$$
$$R_{true} = UV^T$$

It is then possible to find the translation vector, which is the position of the plane with respect to the reference frame of the camera

$$T = K^{-1} \lambda h_3$$

Finally, the camera rotation matrix and the camera position matrix are defined as

$$cameraRotation = R_{true}^T$$
$$cameraPosition = -R^T T$$

5.3 EXPERIMENTAL RESULTS

Following the solution approach detailed in the previous section, the following experimental results have been achieved:

$$cameraRotation = \begin{bmatrix} 0.9565 & 0.0630 & -0.2848 \\ 0.2179 & 0.4950 & 0.8411 \\ 0.1940 & -0.8666 & 0.4597 \end{bmatrix}$$

$$cameraPosition = 10^3 \begin{bmatrix} 1.2968 \\ -1.2999 \\ -0.6517 \end{bmatrix}$$

Plotting both the camera and the reference rectangle

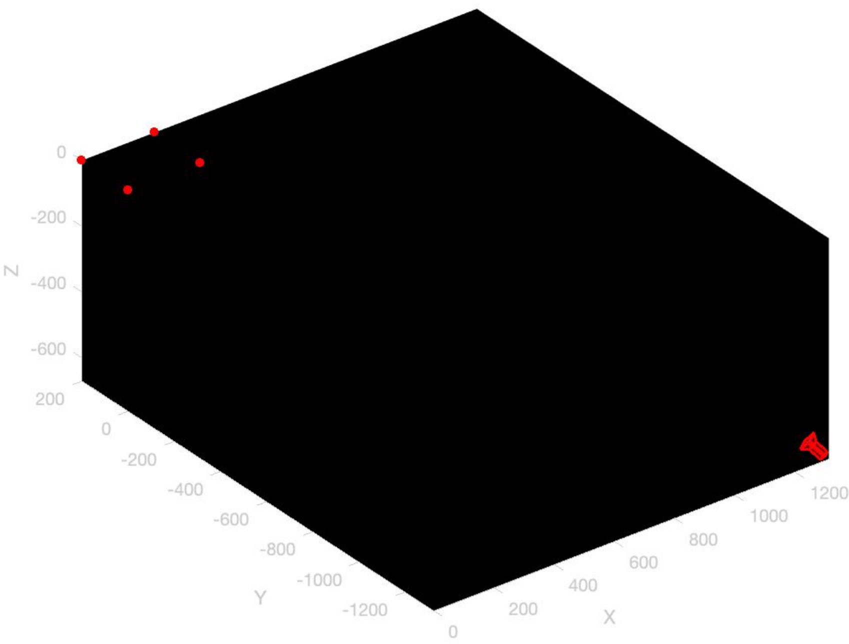

Figure 17: Camera localization - View 1

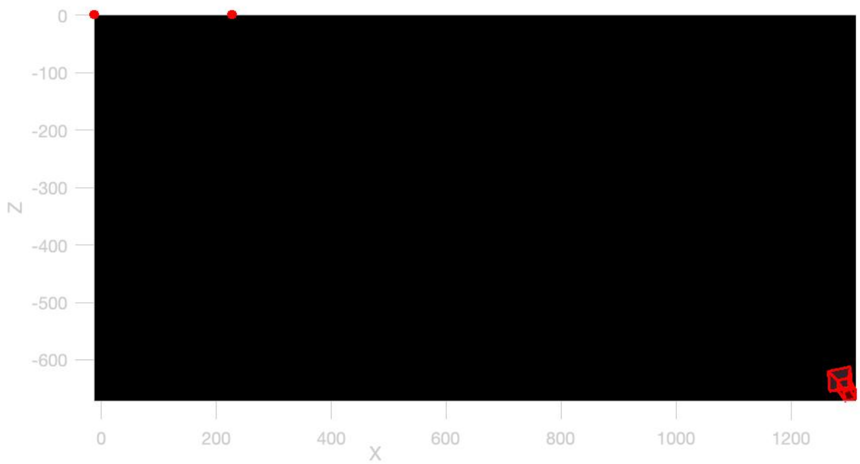

Figure 18: Camera localization - View 2

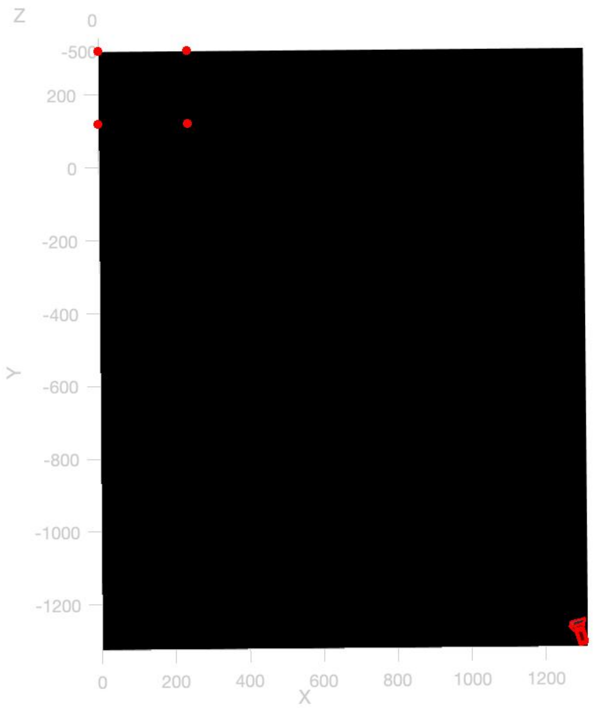

Figure 19: Camera localization - View 3

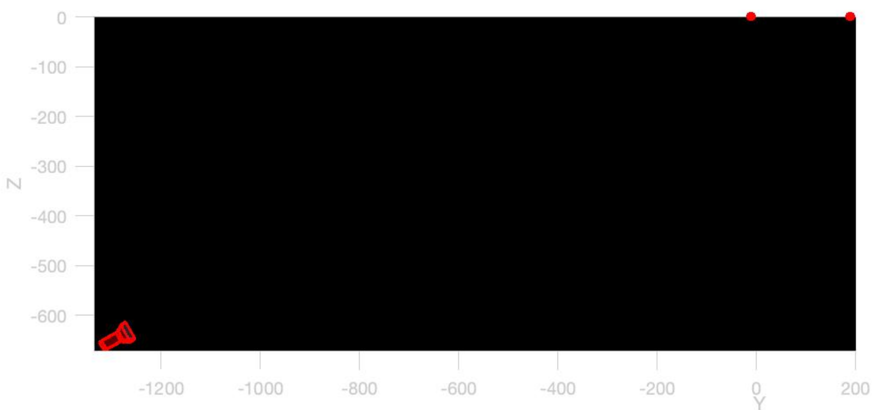

Figure 20: Camera localization - View 4

6 2D RECONSTRUCTION THROUGH THE CAMERA CALIBRATION MATRIX

6.1 PROBLEM FORMULATION

Use the knowledge of K to rectify also a vertical facade, as, e.g., facade containing line 1 or 4 or 6.

6.2 SOLUTION APPROACH

The camera localization step allowed to compute the matrices R_{true} and T, respectively the rotation of the plane with respect to the camera and the position of the plane with respect to the reference frame of the camera. Through these information, together with the camera calibration matrix, it is possible to compute the the projection matrix P

$$P = K \begin{bmatrix} R_{true} & T \end{bmatrix} = \begin{bmatrix} p_1 & p_2 & p_3 & p_4 \end{bmatrix}$$

Then, since a point on the left vertical face can be written as

$$X_v = \begin{bmatrix} x \\ 0 \\ z \\ w \end{bmatrix}$$

in the reference frame of the left horizontal face, it is possible to get the reconstruction matrix from the image to its shape as

$$H_v = \begin{bmatrix} p_1 & p_3 & p_4 \end{bmatrix}^{-1}$$

(it has been decided to rectify the vertical facade containing line 1, so the left vertical facade).

6.3 EXPERIMENTAL RESULTS

The obtained projection matrix is

$$P = 10^3 \begin{bmatrix} 7.8394 & 6.3425 & 4.1925 & 810.98 \\ -0.0174 & 4.1827 & -4.6125 & 2453.9 \\ -0.0002848 & 0.0008411 & 0.0004597 & 1.7624 \end{bmatrix}$$

so, the reconstruction matrix for the left vertical facade is

$$H_v = 10^{-2} \begin{bmatrix} -0.012110 & -0.0091780 & 18.3510 \\ 0.00087420 & -0.018376 & 25.1800 \\ -0.0000017291 & -0.0000062768 & -0.047205 \end{bmatrix}$$

which then, applied to the original image, gives

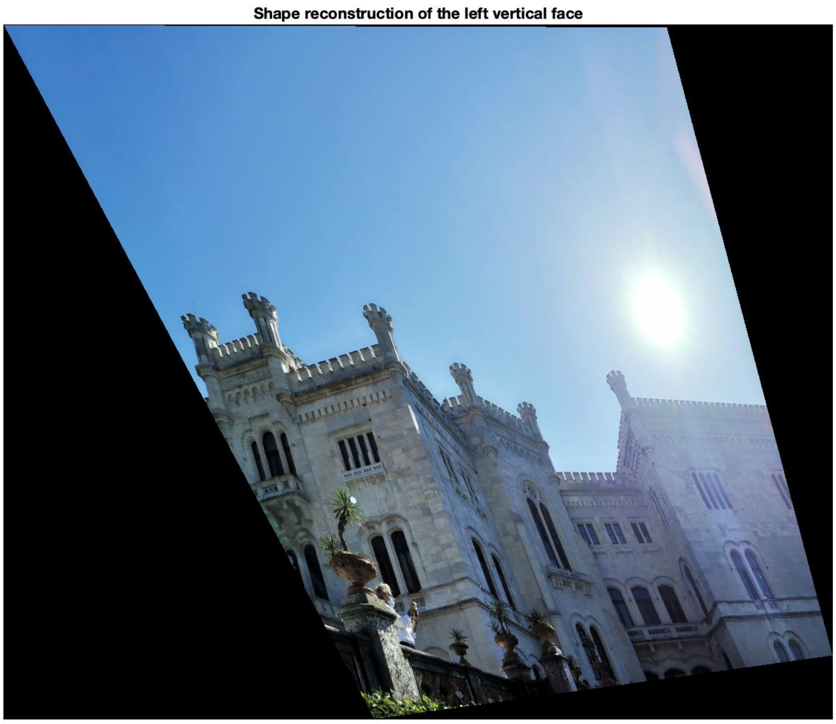

Figure 21: Shape reconstruction of the left vertical face

BIBLIOGRAPHY

[Gia20] Boracchi Giacomo. *Slides of the couse of Image Analysis and Computer Vision*. 2020.

[GW01] Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing*. 2nd. USA: Addison-Wesley Longman Publishing Co., Inc., 2001. ISBN: 0201180758.

[HS88] C. Harris and M. Stephens. "A Combined Corner and Edge Detector". In: *Alvey Vision Conference*. 1988.

[HZ03] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. 2nd ed. USA: Cambridge University Press, 2003. ISBN: 0521540518.

[Vin20] Caglioti Vincenzo. *Slides of the couse of Image Analysis and Computer Vision*. 2020.