Politecnico di Milano

Master's Degree in Computer Science

Image Analysis and Computer Vision

Homework: analysis of the cylindrical vault "Loggia delle Muse" in Palazzo Te

Student: Stefano Vighini

ID: 10622788

Academic year: 2023-2024

Contents

Chapter 1

Introduction and problem setting

This homework's objective is to analyse a scene containing a right circular cylinder, along with its corrisponding image taken with a zero-screw camera. The image depicts the vault *Loggia delle Muse*, situated in *Palazzo Te*, Mantova.

Following, the text of the homework is reported:

> Scene: *"A right circular cylinder together with two (or more) circular cross sections and two (or more) generatrix lines. A right circular cylinder is a surface, made of parallel lines, which is symmetric about a symmetry axis. A circular cross section of a right cylinder is a circumference centered on the symmetry axis and perpendicular to the symmetry axis. A generatrix line is a straight line, on the cylinder surface, which is parallel to its axis."*
> Image: *"A single image is taken of the above described cylinder by an uncalibrated, zero-skew, camera. (Its calibration matrix depends on four unknown parameters, namely f_x, f_y and the two coordinates U_0, V_0 of the principal point). Two circular cross sections are visible, and their images C_1, C_2 are extracted. Two (parallel) generatrix lines of the cylinder are also visible, and their images l_1, l_2 are extracted."*

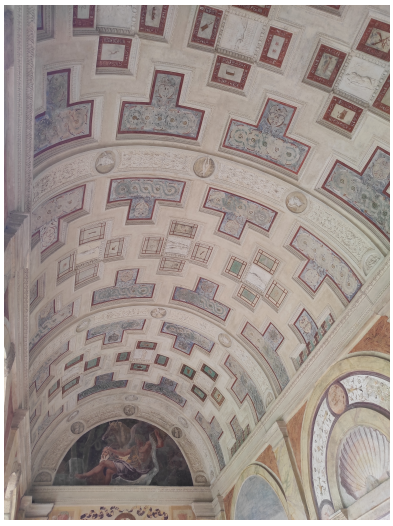

Figure 1.1: Loggia delle Muse in Palazzo Te

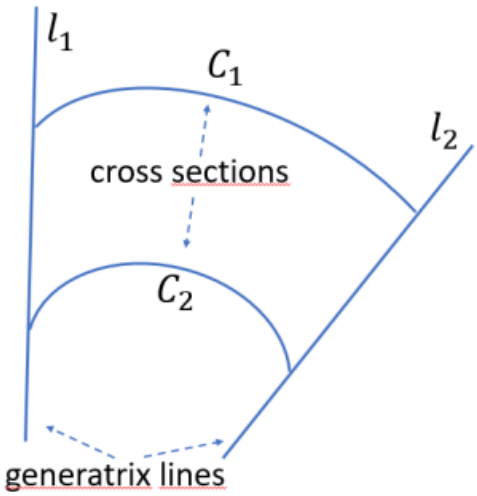

Figure 1.2: Scene

For clarification purposes, we define the following system of coordinates in the scene. The star is the

camera, and the yellow cone represents indicatively the field of view of the camera.

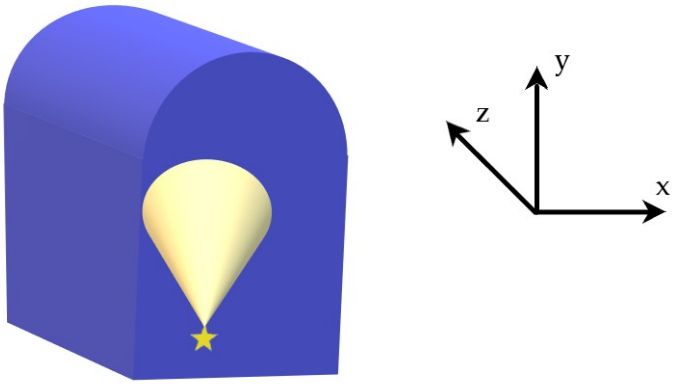

Figure 1.3: System of coordinates

Given two generatrix lines and two circular cross sections, we will extract information such as vanishing lines, vanishing points, cylinder axis, calibration matrix, camera orientation with respect to the cylinder axis, and finally the ratio between the radius of two circular cross sections and their distance. These matters will be discussed both theoretically and practically (using Matlab) in chapter 3.

Furthermore, an algorithm based on RANSAC for automatic model retrieval of the two generatrix lines and circular cross sections will be discussed in chapter 4 along with its modifications from the standard RANSAC, its strenghts and weaknesses. The initial manual detection of useful points to reconstruct lines and conics is instead given in chapter 2.

Finally, two methods and their implementation aimed at the rectification of the cylindrial vault to unfold the picture will be discussed in chapter 5.

The Matlab code has been developed starting from Luca Magri's code, available on the GitHub website of the course. Some pieces of code are also taken from *Image Analyst* on the forum *MATLAB Answers*.

Chapter 2

Choice of generatrix lines and conics from the image

A more accurate (but less scalable) way of generating the useful elements of the image such as the generatrix lines and circular cross sections is the manual approach. Pixels in the image get chosen manually in order to fit a line or a conic.

To retrieve the line, we need to choose 2 points in homogeneous coordinates and compute their cross product.

To retrieve a conic, we need to choose 5 points (x and y coordinates). Afterwards, we will create a system of equations stating that, for every points, the conic passes trough the point.

$$\begin{bmatrix} x_1^2 & x_1y_1 & y_1^2 & x_1 & y_1 & 1 \\ x_2^2 & x_2y_2 & y_2^2 & x_2 & y_2 & 1 \\ x_3^2 & x_3y_3 & y_3^2 & x_3 & y_3 & 1 \\ x_4^2 & x_4y_4 & y_4^2 & x_4 & y_4 & 1 \\ x_5^2 & x_5y_5 & y_5^2 & x_5 & y_5 & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{2.1}$$

Solving for the vector [a b c d e f] (calculating the null space of the first matrix) returns the coordinates of the conic.

To select the pixels in the image, the following has to be taken into account:

- points have to be the furthest possible from each other, to garantee the best precision possible to create the element;
- when taking the points of the 2 parallel lines, the angle they form on their intersection in the image has to be the closest possible to 90°. In this way, the estimation of the vanishing line will be more accurate.
- following the same principle, the conics have to be the most different possible (size and shape) in order for their intersection (the circular points) to be more precise.

The chosen lines l_1, l_2, C_1, C_2 are illustrated in the following picture. Conic $C3$ will be used to assess the result of Assignment 5.

Figure 2.1: l_1, l_2, C_1, C_2, C_3

Chapter 3

Assignments

3.1 Assignment 1

From C_1, C_2 find the horizon (vanishing) line h of the plane orthogonal to the cylinder axis.

Theoretical solution

"The plane orthogonal to the cylinder axis" is every plane whose direction is Z = [0, 0, 1] in the scene. Conics C_1, C_2 belong to two of these planes, indeed those planes are parallel. Using the following theorem:

Theorem 1 *Two parallels planes in the space share the same line at the infinity and the same circular points.*

we can intersect conics C_1 and C_2 in order to find the circular points (named from now on s_1, s_2). To intersect the conics, we must solve a system of equations in the variables x, y, derived from the matrix formulation of the point-conic intersection:

$$\begin{bmatrix} x & y & 1 \end{bmatrix} \begin{bmatrix} a & b/2 & d/2 \\ b/2 & c & e/2 \\ d/2 & e/2 & f \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \tag{3.1}$$

$$\begin{cases} a_1x^2 + b_1xy + c_1y^2 + d_1x + e_1y + f_1 = 0 \\ a_2x^2 + b_2xy + c_2y^2 + d_2x + e_2y + f_2 = 0 \end{cases} \tag{3.2}$$

where $[a_1, b_1, c_1, d_1, e_1, f_1], [a_2, b_2, c_2, d_2, e_2, f_2]$ are respectively the parameters of C_1, C_2.

This will produce two pairs of circular points, that create two lines at the infinity. In order to choose which line is the right one, we plot the results and trivially check which line is the correct one.

Matlab comments

The reference code for this assignment is "assignments_1.m".

No interesting additional comments are necessary for the code.

Results discussion

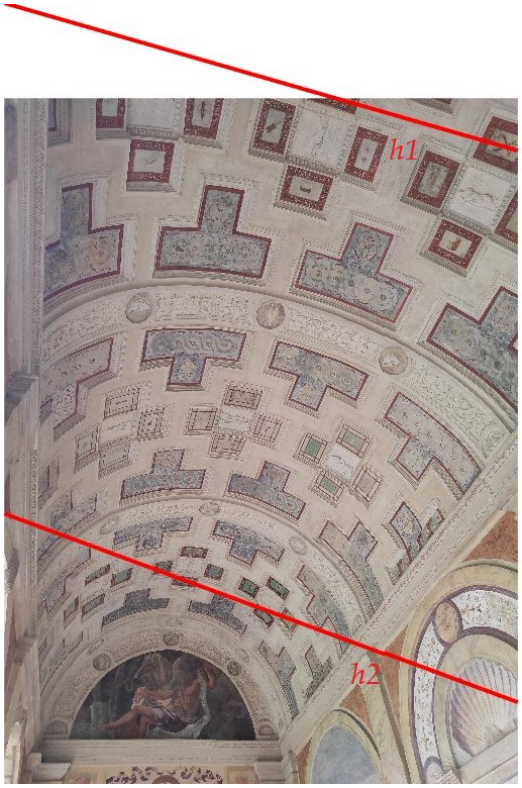

Figure 3.1: Lines h_1, h_2

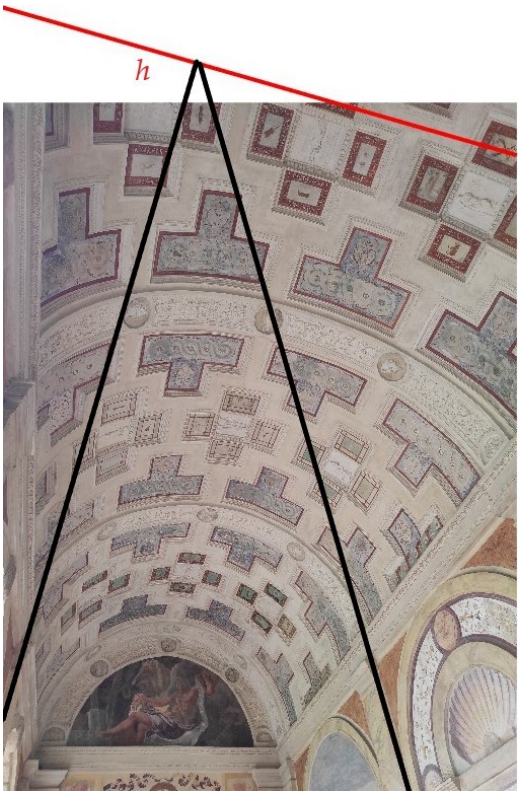

Figure 3.2: Line at the infinity h

Figure 3.1 shows the two candidate lines at the infinity. Figure 3.2 shows the chosen line (h) and plots two black lines, parallel in the scene, that belong to the same plane to which the conics do. We can notice that these black lines intersect at the line at the infinity h, as we would expect.

3.2 Assignment 2

From l_1, l_2, C_1, C_2 find the image projection a of the cylinder axis, and its vanishing point V.

Theoretical solution

Computing the vanishing point V is straighforward: it is the cross product between l_1, l_2. This because l_1, l_2, a are parallel.

An approach to find the cylinder axis a is to compute the cross product between V and the center of a conic, for example C_1. In order to find the center of a conic we need to compute an homography that maps the conic from a projectivity to an affinity of the original image (equation 3.3). Then, compute the affine trasformation of the conic (equation 3.4). After extracting the center of the ellipse CE, use theorem 2, and map it back to the original projectvity (equation 3.5).

$$H = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ & h & \end{bmatrix} \tag{3.3}$$

$$C_1' = H^{-T} C_1 H^{-1} \tag{3.4}$$

$$CE = H^{-1} CE' \tag{3.5}$$

Theorem 2 *The image of the center of an ellipse in an affine trasformation is the center of the ellipse in the image.*

In this way we can compute the cross product between CE, V in the projectivity, obtaining the cylinder axis a.

Matlab comments

The reference code for this assignment is "assignments_2.m".

The external function AtoG has been used to compute the center of the ellipse from its parameters, like seen in the lectures.

Results discussion

Figure 3.3 shows obtained results. In red, the cylinder axis. In blue, points CE (the center of the conic C_1) and V (the vanishing point of lines l_1, l_2). Black lines are different parallel lines used to

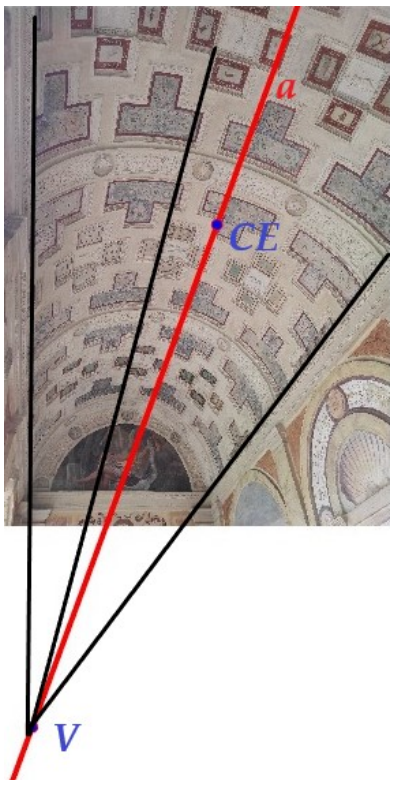

Figure 3.3: a, V

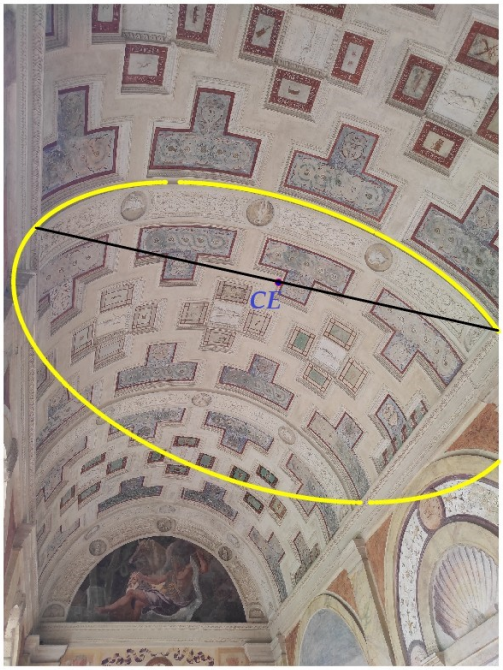

Figure 3.4: Center of conic $C1$: CE

assess wether the vanishing point is indeed correct, and, with a certain margin due to imprecision of the choice of points, we can assert it.

Furthermore, we can state by looking at figure 3.4 that the center of circular cross section C_1, named CE, is on the line that connects the points of intersection between the vault and the underlying structure. Therefore, we can deduce that CE is also correct and by inference the cylinder axis a is, too.

3.3 Assignment 3

From l_1, l_2, C_1, C_2 (and possibly h, a and V), find the calibration matrix K.

Theoretical solution

The calibration matrix K, under the assumption that the camera has zero-skew, is formulated as equation 3.6. As there is a one-to-one relation between K and the image of the absolute conic w, we can move the problem to finding w. This means creating a system of equation regarding w (expressed relative to the parameters of K, like in equation 3.7) using the following constraints:

- orthogonal relationship: find two vanishing points related to orthogonal lines and compute 3.8. The vanishing points used are V and the circular point s_1.
- "belonging" relationship: circular points belong to w, therefore it it possible to compute equation 3.9

As both these equations involve circular points, they are in reality 4 equations, because for each of those there is a real and imaginary part to be equal to 0. We now have a system of 4 equations in 4 unknowns, that can be solved using linear algebra.

$$K = \begin{bmatrix} f_x & 0 & U_0 \\ 0 & f_y & V_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.6}$$

$$w = (KK^T)^{-1} = \left(\begin{bmatrix} f_x^2 + U_0^2 & U_0V_0 & U_0 \\ U_0V_0 & f_y^2 + V_0^2 & V_0 \\ U_0 & V_0 & 1 \end{bmatrix} \right)^{-1} = \begin{bmatrix} a^2 & 0 & -U_0a^2 \\ 0 & b^2 & -V_0b^2 \\ -U_0a^2 & -V_0b^2 & 1 + U_0a^2 + V_0b^2 \end{bmatrix}$$
$$\text{with a} = \frac{1}{f_x}, \text{ b} = \frac{1}{f_y} \tag{3.7}$$

$$V^T w s_1 = 0 \tag{3.8}$$

$$s_1^T w s_1 = 0 \tag{3.9}$$

Matlab comments

The reference code for this assignment is "assignments_1.m".

No interesting additional comments are necessary for the code.

Results discussion

Matrix K found is the following:

$$K = \begin{bmatrix} -4257,8 & 0 & 1574 \\ 0 & 4137 & 2170,4 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.10}$$

This matrix can be an acceptable answer. We can check that the parameters U_0, V_0 are similar to half of the dimensions of the image (3472 x 4640), as we would expect to be the position of the principal point.

3.4 Assignment 4

From h, K and V determine the orientation of the cylinder axis with respect to the camera reference.

Theoretical solution

This is a localization problem. We want to find the vector [i j k o] related to the plane that contains the conics (for example C_1), that is also orthogonal to the cylinder axis. In this way, being i, j the vectors related to X, Y coordinates in the scene, k will point in the direction of the cylinder axis Z.

Assuming the world reference is on the camera reference, hence $R = 0, t = 0$, we define Z^w =[0 0 1] as the vector that points in the direction of the camera. If the image was taken "sitting" on the cylinder axis and looking in the direction of it, Z^w would be equal to Z. As this is not the case, projecting Z, Z^w on planes XZ, YZ and computing the angle between their directions we obtain the camera rotations with respect to the cylinder axis reference. Let's define θ as the angle on the plane XZ, and ϕ as the angle on the plane YZ.

To compute the vector $[ijko]$, in which we are truly interested only in k, we proceed as following:

- compute affine reconstruction homography H_1 as shown in equation 3.3;
- find all the parameters of the ellipse (center o, axes ν, angle ψ) and compute the rectifying homography H transforming the ellipse into a circle (equations 3.11, 3.12, 3.13);
- compute vector $[ijo]$ using equation 3.14 and compute $k = i \times j$ (cross product);
- make sure that k and Z^w have the same sign on the 3rd component. Otherwise, compute $k = -k$.
- finally, compute θ, ϕ using 3.15, 3.16

$$R = \begin{bmatrix} cos(\psi) & -sin(\psi) \\ sin(\psi) & cos(\psi) \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & \frac{\nu_1}{\nu_2} \end{bmatrix} \begin{bmatrix} cos(\psi) & sin(\psi) \\ -sin(\psi) & cos(\psi) \end{bmatrix} \tag{3.11}$$

$$A = \begin{bmatrix} R & 0 \\ 0 & 1 \end{bmatrix} \tag{3.12}$$

$$H = AH_1 \tag{3.13}$$

$$[ijo] = K^{-1}H^{-1} \tag{3.14}$$

$$\theta = arccos(\frac{k_1 Z_1^w + k_3 Z_3^w}{\sqrt{k_1^2 + k_3^2}}) \tag{3.15}$$

$$\phi = arccos(\frac{k_2 Z_2^w + k_3 Z_3^w}{\sqrt{k_2^2 + k_3^2}}) \tag{3.16}$$

Matlab comments

The reference code for this assignment is "assignments_4.m".

The external function AtoG has been used to compute the center of the ellipse from its parameters, like seen in the lectures.

Results discussion

Results obtained are: $\theta \approx 27°, \phi \approx 60°, k = [0{,}24\ 0{,}84,\ 0{,}48]$. From manual inspection of the image, these values are plausible.

3.5 Assignment 5

Compute the ratio between the radius of the circular cross sections and their distance.

Theoretical solution

For this assigment we make the assumption that **a line that passes through two generic points belonging respectively to $l1, l2$ doesn't intersect the cylinder axis**. Furthermore, I took the image of another conic C_3 NOT TO SOLVE the assignment, but to ASSESS it's correctness. This additional conic hasn't been used if not to compute the final ratio.

To compute a cross ratio between lengths that are not parallel, we need a shape rectification on a plane that contains both the distances we need to compute the ratio on.

The following steps need to be taken to address the task:

1. find the center of C_1 like in Assignment 2 and name it CE;
2. compute $l_{d1} = CE \times$ [0 0 1], being the latter a generic point in homogeneous coordinates. This line contains the diameter of C_1;
3. compute the intersection points $p_1, p_2 = intersect(C_1, l_{d1})$;
4. compute $l_{a1} = p_1 \times V$ and $l_{a2} = p_2 \times V$. These lines are parallel to cylinder axis a;
5. compute the intersection points $p_3 = intersect(C_2, l_{a1})$ and $p_4 = intersect(C_2, l_{a2})$.
6. compute $l_{d2} = p_3 \times p_4$. This line contains the diameter of C_2;
7. compute the vanishing point: $V_2 = l_{d1} \times l_{d2}$;
8. find the vanishing line: $l_{inf} = V \times V_2$;
9. compute the absolute conic: $w = (KK^T)^{-1}$;
10. find the circular points: $s_1, s_2 = intersect(w, l_{inf})$;
11. build the image of the conic dual to the circular points: $\chi = s_1 s_2^T + s_2 s_1^T$;
12. compute: $SVD(\chi) \rightarrow U, D, V$;
13. compute H, the shape rectifying homography: $H = (U\sqrt{D})^{-1}$;
14. apply the transformation to points p_1, p_2, p_3 and compute the ratio $\frac{dist(p_1,p_2)}{2dist(p1,p3)}$;
15. ADDITIONAL: repeat step 5 with C_3 to compute p_5, p_6 and step 14 using p_1, p_2, p_5.

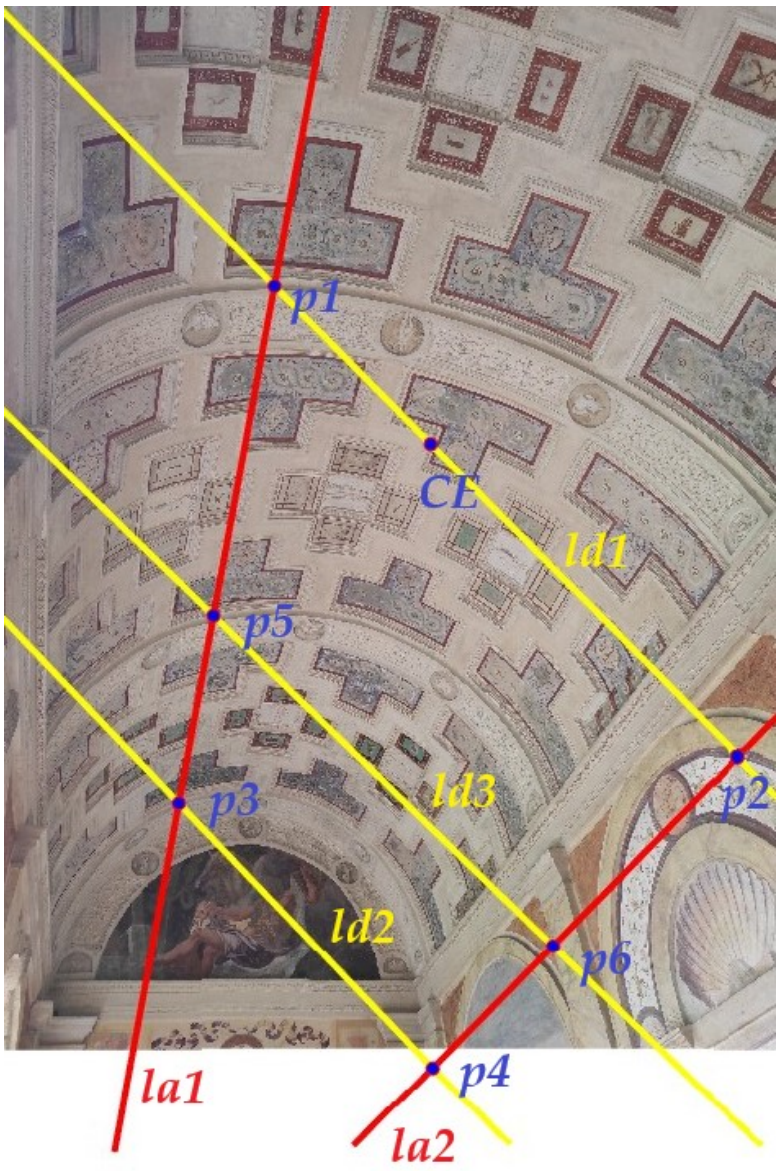

Figure 3.5: Points and lines

Matlab comments

The reference code for this assignment is "assignments_5.m".

The external function AtoG has been used to compute the center of the ellipse from its parameters, like seen in the lectures.

Function "getIntersectionConicLine" returns two points, as there should be two intersections between a line and a conic. In some cases, only one point is the actual intersection between the conic in the image and the line (because the conic is cut), therefore we need to select the interesting point by manual inspection (usually it's the first one).

Results discussion

The results are the following:

$$\frac{dist(p_1, p_2)}{2dist(p1, p3)} = 0,24$$

$$\frac{dist(p_1, p_2)}{2dist(p1, p5)} = 0,49$$

Looking at the values, we can claim they are correct. In fact, we can think about the shape formed by points p_1, p_2, p_6, p_5 as a square in which one edge is the diameter of the cylinder and the other edge is the distance between the two circular cross sections (we assume it's a square by manual inspection of the image). Therefore 0,49 is the correct ratio. Furthermore, as we can hypotize that the distance between C_1, C_2 is double of the distance between C_1, C_3, the ratio of the latter is half of the former.

Chapter 4

Automatic model detection

I anticipate that i wasn't successful in finding an algorithm that could automatically detect all the useful models of lines and conics needed to solve the homework. The useful images remain the ones computed by hand. Said so, i put an effort in using different approaches and building up algorithm modifications in order to try to achieve a good result.

The basis of the algorithm

1. The first issue is to recognize the corners that will then be used to fit the models. In matlab feature detection methods are already implemented (like FAST, SIFT, SURF...). Evaluating these algorithms empirically, the one that seemed to perform best is the KAZE algorithm. Anyways, the result doesn't really stand out from other methods.

2. The next step is to create a model starting from the detected corners. I used RANSAC (RANdom SAmple Consensus) algorithm, that works as follows: take a minimal subset of points, fit a model to these points and use the number of inliers (using distance from the model as a metric) as a score. Iterate this process a desired number of times and conclude selecting the model that provides the most inliers.

   In our case, the model of the line is computed as the cross product of 2 points extracted randomly among all points. As it was already difficult to select the line and the ellipse is even more challenging to retrieve, the ellipse model was not computed. However, the idea for selecting an ellipse is to use the same principle of a line, that is taking 5 points, fit a conic through them and then compute the distance from a point and the model.

Parameters used:

- n_corners (number of corners for KAZE algorithm): 1000;
- p (cardinality of minimum sample set): 2. This means that for every pair of points, I compute the exact line that passes through them. An alternative could have been to take more than 2 points and fit a linear model (that very likely will not pass through any point). I thought this approach was not effective, nor could be determinant for the success of the algorithm;

- t (inlier threshold): 0,03. Values like 0,05; 0,01; 0,005 have also been used;
- maxIterations: 10^6.

Modifications

1. **Problem:** the algorithm is random, and could therefore never take the best lines.

   Solution: considered the dimension of the problem, and the absence of time constraints, I chose to compute all the possible lines for every subset of 2 points. Of course this could not be done with ellipses, as the complexity doesn't increase like $O(N^2)$, but like $O(N^5)$ (considered that 5 points are needed to fit a conic).

2. **Problem:** the algorithm generates just one line, but two are needed.

   Solution: instead of taking the best line, preserve all lines, rank them based on the number of inliers, and take the best two lines.

3. **Problem:** the algorithm may return all lines with a similar slope. Useful lines that are parallel in the scene have very different slopes in the image.

   Solution: two possible solutions are available. The first one takes, instead of the best two lines, the best M lines and overwrites a line when $\theta < A$, where θ is the angle formed on the intersection of the two lines. In this way there will be just lines that have an angle difference of at least θ. There is an inaccuracy for which, when multiple overwrites occur, the angle difference is not garanteed. The second method divides the 180°angle in M sectors and assigns a line to a sector based on the angle formed with the x-axis. In this case, two similar lines are even more probable then in the first case, but the result still serves the purpose.

4. **Problem:** the lines found by the algorithm pass through a region dense in number of corners, creating useless lines.

   Solution: prune corners whose distance is less than a tolerance D.

Parameters used:

- A: $\frac{180°}{M}$;
- M: 10;
- D: 30.

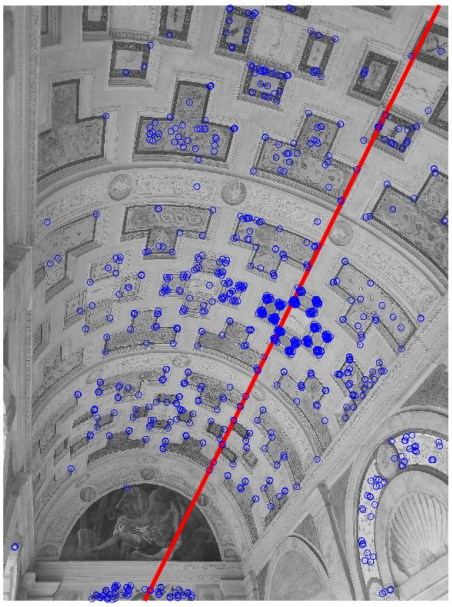

Figure 4.1: Best line

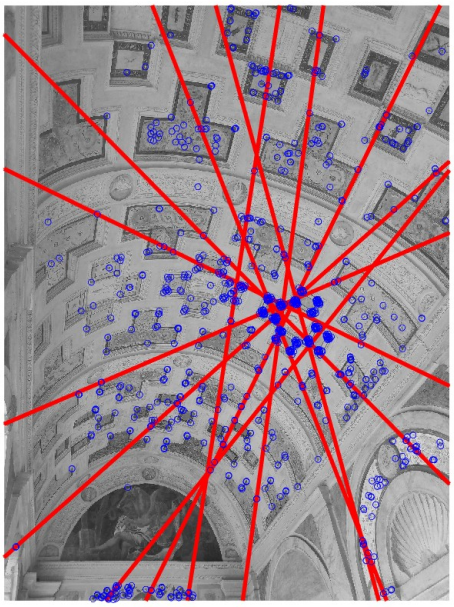

Figure 4.2: Method 1 of multiple lines

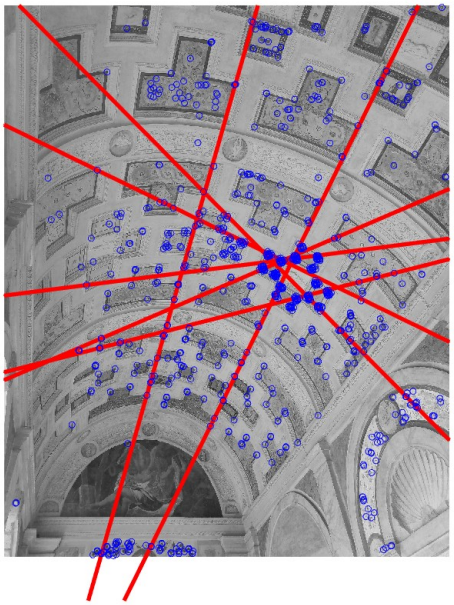

Figure 4.3: Method 2 of multiple lines

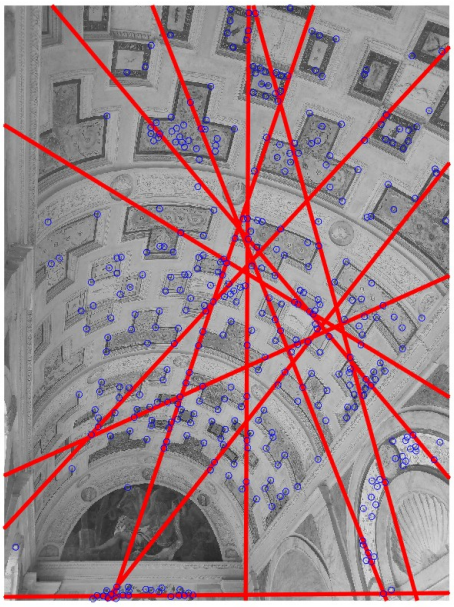

Figure 4.4: Pruning

Conclusions

As we can see in the attached pictures, it's difficult to extract more than one useful line from the image. Two correct lines could be extracted, but they are still too close and would lead to worse results than using the manual method. We can also note the inaccuracy at pixel level.

I think the reason why we can't successfully extract useful models is that this image is very colorful and full of details, and the feature detection algorithm struggles at distinguishing real edges from fancy items.

Chapter 5

Vault 2D rectification

Two methods will be proposed in this paper to perform 2D vault rectification, or "unfolding". The second method leads to a better result. Both algorithms start with the same initial steps.

First part

Performing 2D rectification of a cylinder surface means to create a new image that maps every point x,y on the original image to the corresponding θ, d, with θ being the angle rotating around the cylinder axis and d being the depth of the cylinder (in the direction of the cylinder axis).

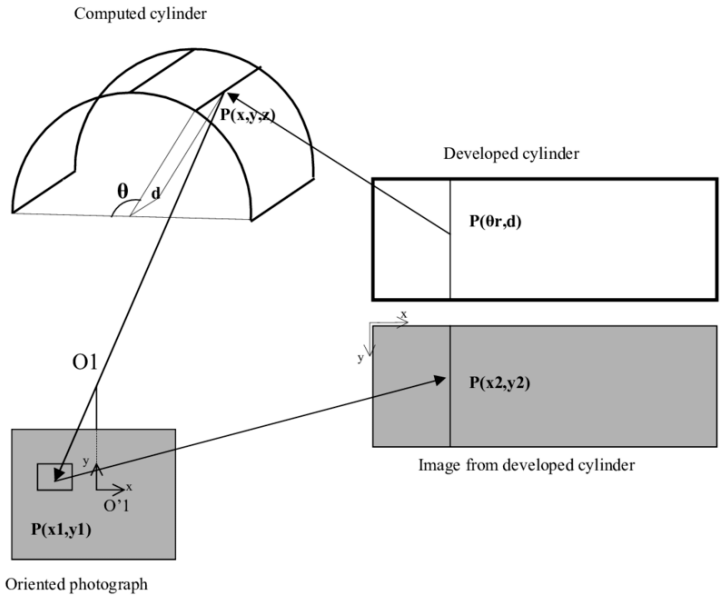

Figure 5.1: 2D Vault rectification

Below are the steps to address the first part of the problem:

1. compute the circle from the conic C_1 using the homography computed in Assignment 4;
2. extract its parameters (center and radius), with which we can create new points on the circumference in function of the angle θ;
3. apply the inverse homography to find the same points in the original image;

4. compute the cross product between the newly found points and the vanishing point V to create lines parallel to the cylinder axis;

5. finally, compute the intersection between these lines and the conic C_2.

The result is showed in the figure 5.2, that is a division of the cylinder surface into sectors of equal angle. The number of sectors is determined by $nPoints - 1$ and the angle is determined by $\frac{halfTurn}{nPoints-1}$. In practise, some correction parameters are necessary in MATLAB to address some accuracy problems (starting angle fix, $halfTurn$ equal to 170°instead of 180°...).

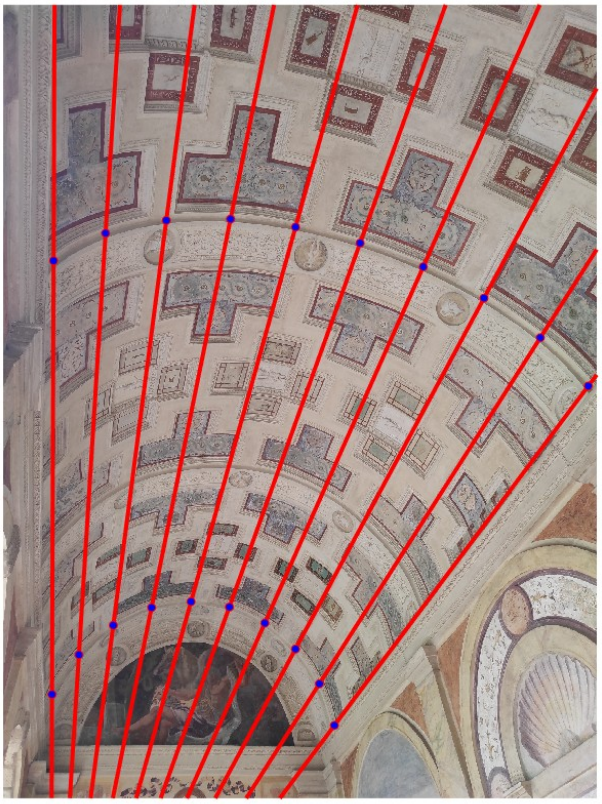

Figure 5.2: Sectors division

Second part

From now on, two different methods can be used to compute the rectified image. The results are not perfect because they suffer from inaccuracy of identification of the conics.

Method 1: "Infinite" homographies

This method maps four points related to a sector into a rectangle of known shape and, afterwards, concatenates all the rectangles. I made the assumption (based on the result of Assignment 5) that

the aspect ratio of the rectified cylinderical vault is 1:2 (1 being the width).

As the number of points increases, we would expect the curves to be approximated with lines, and therefore to obtain a faithful reconstruction. Unfortunately, obtained results are not fullfilling, as taking a high number of sectors introduces a problem of alignment of rows. In figure 5.3 is shown the result for $nPoints = 10$ (that is not high enough to remove the distortion of the approximation from curves to lines), and in figure 5.4 is shown the result for $nPoints = 20$ (where we can note the decreasing distortion but the introduction of the alignment problem).

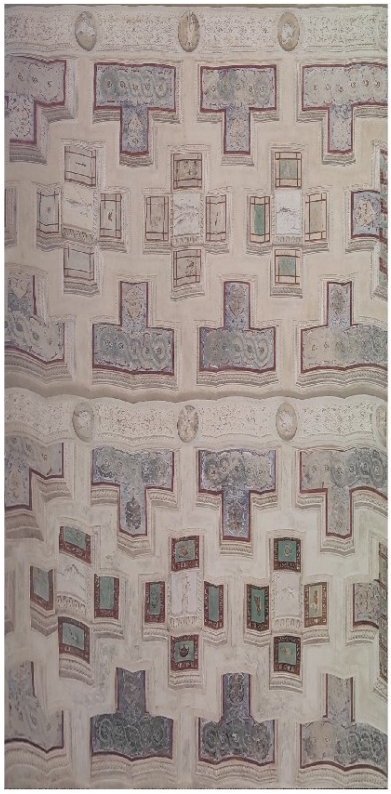

Figure 5.3: Method 1: nPoint = 10

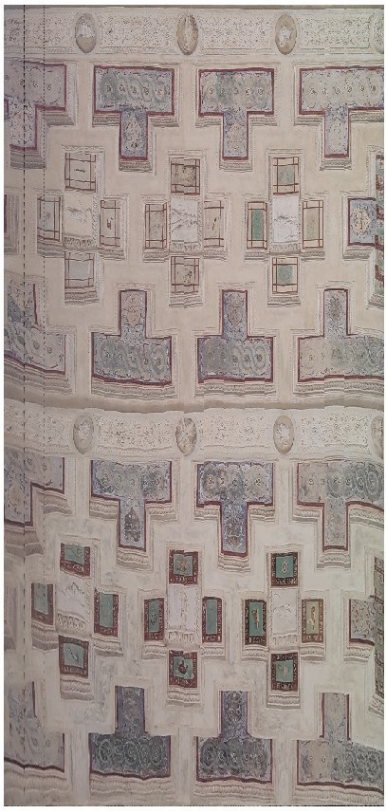

Figure 5.4: Method 1: nPoints = 20

Method 2: Pixel selection using cross-ratio

This method selects pixels one by one, filling in a matrix that will be the final vault image. Iterating on the lines is equal to moving on the θ axis, knowing that the distance (angle) between them is constant. We cannot use the same reasoning for the depth of the cylinder d as the scene is related to our working image with a projectivity, meaning that the same distance in the scene corresponds to different distances in the image.

What has been used to address this problem is the cross-ratio. It is possible to extrapolate the middle point between two points that lie on the same line knowing the vanishing point V (equations

from 5.1). We can apply this procedure recursively in order to find the position of all points on the line and map them to their correct depth d in the rectified image. Figure 5.5 shows the result for $nPoints = 1024, nRows = 2049$. $nRows$ is $2^x + 1$ (with generic x) in order to simplify the process of populating the vault matrix: in this way the index d of the middle point is always an integer number. $nPoints, nRows$ are dimensioned to preserve the aspect ratio of 1:2 and to provide a sufficient resolution. Fig 5.6 and 5.7 provide a toy example with low resolution to visualize the selected points on the original image.

Taking 4 collinear points, in order V (vanishing point), Y, M (middle point between X,Y), X:

$$M = \alpha X + (1 - \alpha)Y, \text{ collinearity relationship} \tag{5.1}$$

$$\frac{M - X}{M - Y} = -\frac{V - X}{V - Y}, \text{ cross ratio property} \tag{5.2}$$

$$\frac{M - X}{M - Y} = \frac{\alpha X + (1 - \alpha)Y - X}{\alpha X + (1 - \alpha)Y - Y} = \frac{(\alpha - 1)(X - Y)}{\alpha(X - Y)} = \frac{\alpha - 1}{\alpha} \tag{5.3}$$

$$\frac{\alpha - 1}{\alpha} = -\frac{V - X}{V - Y} \implies \alpha = \frac{1}{1 + \frac{V - X}{V - Y}} \tag{5.4}$$

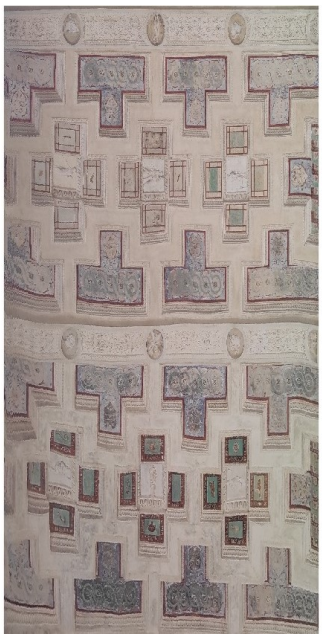

Figure 5.5: Method 2

Figure 5.6: Method 2: test

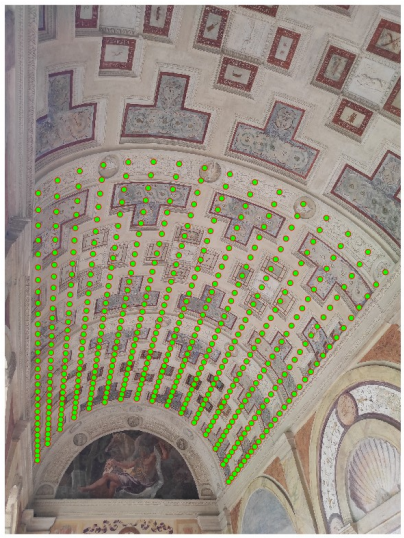

Figure 5.7: Method 2: test

Chapter 6

Conclusion

This paper discusses in detail all theoretical aspects related to the commissioned tasks. Useful information about potential issues have also been included. Hints about MATLAB solutions have been given, but to know the exact functioning of the MATLAB code, please see the attached ".m" files.

All images contained in the paper are also included in the folder. Auxiliary functions (of which the only one copied from another user, "Hui Ma", is "AtoG", the same function used in the lectures) are in "fun" directory, while the computed results in ".mat" format are in "vars" directory.

All tasks (in my opinion) have been completed and explained but the automatic model detection, for which a justification on the impossibility of extract useful models have been provided.

Bibliography

[1] Vincenzo Caglioti. “Slides for the course Image Analysis and Computer Vision”. In: (2023).

[2] Hui Ma. “AtoG: Conversion of conics parameters”. In: (2011).

[3] Luca Magri. “Matlab code shown in laboratory lectures”. In: (2023).