3D recovery of urban scenes: Planar transformations and image rectificaction

Josep Brugués i Pujolràs[1], Sergi García Sarroca[1], Òscar Lorente Corominas[1], and Ian Pau Riera Smolinska[1]

Universitat Autònoma de Barcelona, Bellaterra, 08193, Catalonia, Spain
{josep.brugues, sergi.garciasa, oscar.lorentec, ianpau.riera}@e-campus.uab.cat

1 Introduction

In this project, planar transformations and rectifications are applied to some of the images of the **EPFL-Stretcha** dataset. In this document, we present the methodology that we have followed for each section, as well as the results obtained and the problems that have arisen.

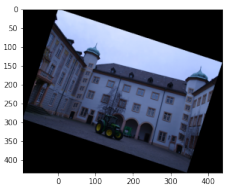
The core algorithm to complete the work in this week's project is to transform an image given a 2D homography, and will be used repeatedly in all the exercises. Given an image I and an homography H, the algorithm returns a transformed image I_{out}, following the steps provided in Algorithm 1. Furthermore, other utility functions have been implemented on this project, so that common actions across the different exercises can be performed easily without duplicating unnecessary code. Such functions are used to compute angles between lines, drawing lines on an image and get homographies given certain parameteres, among others.

Algorithm 1 apply_H function

Input: I, H, $corners$
Output: $I\_rectified$, $rectified\_image\_axis$, $rectified\_image\_corners$

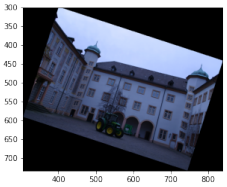
1. First, the homography H is applied to each of the corners of the image I, so that we can compute the minimum and maximum values of each axis, and thus the size of the output image. If corners are not provided as an argument, the corners are computed using the width and the height of the image.
2. Then, a mesh of coordinates is created using the values from the previous step.
3. We multiply the inverse of the homography H to each of the coordinates of the mesh, thus obtaining their correspondent positions on the image I.
4. The obtained coordinates are used in the scipy.map_coordinates function to find, for each pixel in the output image, its correspondent pixel in the input by interpolation.

(a) Original Image



(b) Scale and rotation

(c) Scale



(d) Scale, rotation and translation

Fig. 1: Similarity transformations - Image 0000

2 Image transformations

In this exercise, we use the apply_H function to apply different homographies to the same image: image 0000.

2.1 Similarity

Similarity transformations are movements of two-dimensional shapes within their plane. Specifically, we have implemented rotation, scale and translation. The homography H_s that encodes all these transformations is given by:

$$H_s = \begin{bmatrix} sR & \overrightarrow{t} \\ \overrightarrow{0}^T & 1 \end{bmatrix} = \begin{bmatrix} scos(\theta) & -ssin(\theta) & tx \\ ssin(\theta) & scos(\theta) & ty \\ 0 & 0 & 1 \end{bmatrix} \quad (1)$$

Fig. 1 shows different similarity transformations applied to image 0000 (Fig. 1a), combining the following parameters: $s = 0.5$, $\theta = 0.1 * \pi$ and $(tx, ty) = (400, 300)$. Making use of the returned axis from the apply_H function, we are able to show the effect of the translation, as it can be seen in Fig. 1b and Fig. 1d, where the difference on each of the axis between the two images corresponds to the values of tx and ty. Without explicitly changing the axis values, the images would be centered at $(0, 0)$ and it would be impossible to distinguish a translated image from a non-translated image.

2.2 Affinities

Affinity transformations are geometric transformations that preserve lines and parallelism, but not necessarily distances and angles. Given a non-singular 2x2

matrix A, the homography H_p that encodes all these transformations is given by:

$$H_a = \begin{bmatrix} A & \overrightarrow{t} \\ \overrightarrow{0}^T & 1 \end{bmatrix} \tag{2}$$

A helpful way to understand the geometric effects of the linear component A of an affine transformation is as the composition of two fundamental transformations, namely rotations and non-isotropic scalings. The affine matrix A can always be decomposed as

$$A = R(\theta)R(-\phi)DR(\phi) \tag{3}$$

where $R(\theta)$ and $R(\phi)$ are rotations by θ and ϕ respectively, and D is a scale. This decomposition follows directly from the SVD, and A can be written as well as:

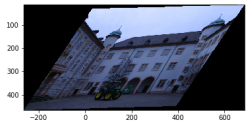
$$A = \mathrm{UDV^T} = \left(\mathrm{UV^T}\right)\left(\mathrm{VDV^T}\right) \tag{4}$$

In this project, both decompositions have been applied and compared, obtaining a negligible mean square error of $3.05e-32$ between them.

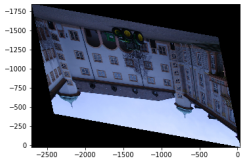
Fig. 2 shows different affinity transformations applied to image 0000. Combining the following parameters we obtain the image on Fig. 2b: $s = [1, 0.5]$, $\theta = 0.1 * \pi$, $\phi = 0.3 * \pi$ and $(tx, ty) = (30, 30)$. We can observe in Fig. 2d that the image obtained with the SVD decomposition returns the same output image.

Another example of affine transformation, with the parameters: $s = [3, 2]$, $\theta = \pi$, $\phi = -1.2 * \pi$ and $(tx, ty) = (30, 30)$, is presented in Fig. 2c. The result is an image sheared and flipped on both axis. We can observe how angles are not preserved, but the ratio of lengths is.

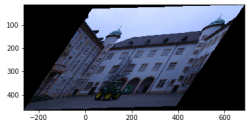
(a) Original Image



(b) Affine transformation 1



(c) Affine transformation 2



(d) Affine transformation 1 - SVD

Fig. 2: Affine transformations - Image 0000

2.3 Homographies

A projective transformation maps every figure into a projectively equivalent figure, leaving all its projective properties invariant. Given a non-singular 2x2 matrix A, the homography H_p that encodes all the transformations is given by:

$$H_p = \begin{bmatrix} A & \overrightarrow{t} \\ \overrightarrow{v}^T & v \end{bmatrix} \tag{5}$$

Fig. 3 shows different homographies applied to image 0000. We tried three different sets of parameters, which are presented in Tab. 1. We can observe how the angles are not preserved and neither the ratio of lengths.

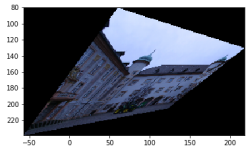
3 Affine Rectification

This method aims to remove *projective distorsions*. The affine transformation preserves points, straight lines and planes, i.e. parallel lines remain parallel after

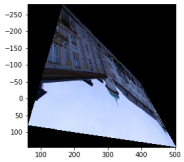
Table 1: Parameters of the different homographies and corresponding Figure.

Figure	A	$\overrightarrow{v}$	(tx,ty)
3b	$\begin{bmatrix} 0.5 & -0.25 \\ 0.25 & 0.5 \end{bmatrix}$	[0.0015, 0.001]	(60,80)
3c	$\begin{bmatrix} 3 & 1 \\ 0.8 & -2 \end{bmatrix}$	[0.005, 0.005]	(60,80)
3d	$\begin{bmatrix} 4 & 8 \\ -6 & 1 \end{bmatrix}$	[0.005, 0.002]	(60,80)

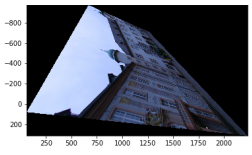
(a) Original Image



(b) Homography 1



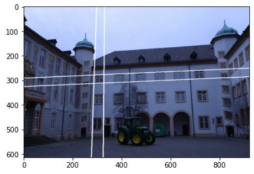
(c) Homography 2



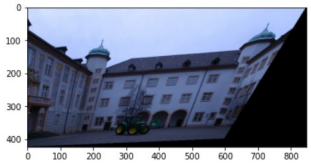
(d) Homography 3

Fig. 3: Homographies - Image 0000

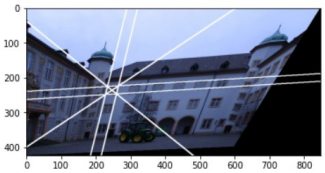
(a) Original Image



(b) Lines that are parallel in the world



(c) Affinely rectified image



(d) Transformed lines are parallel after affine rectification

Fig. 4: Affine rectification - image 0000

an affine transformation. In this case, we face the problem of transforming an image by a homography so that the vanishing line of the image plane becomes a line at infinity.

In order to compute the vanishing line of the image, we use two sets of lines that are parallel in the world (Fig. 4b). The two sets of lines may look parallel in the original image too, but they intersect at two vanishing points. Each set of parallel lines provides a vanishing point, which can be computed from the cross product (Eq. 6). With points v1, v2 we can compute the line at infinity by applying the cross product again. Once the vanishing line is obtained, we can obtain the homography $\mathrm{H}_{a \leftarrow p}$, as described in Eq. 8, to finally perform the affine rectification.

$$v_1 = l1 \times l2 \qquad v_2 = l3 \times l4 \tag{6}$$

$$L_\infty = v1 \times v2 \tag{7}$$

$$H_{a \leftarrow p} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ l1 & l2 & l3 \end{bmatrix} \tag{8}$$

Fig. 4d shows the transformed parallel lines after applying the obtained homography. The angles between the two sets of parallel lines before and after the affine rectification are presented in Tab. 2. As expected, the angles are 0^{o} after the affine rectification, so the parallel lines in the world are also parallel in the rectified image.

Table 2: Angles (^{o}) between parallel lines in the world before and after affine rectification - Image 0000

Image	L1/L2	L3/L4
Original	0.10	1.34
Affinely rectified	0.0	0.0

4 Metric rectification

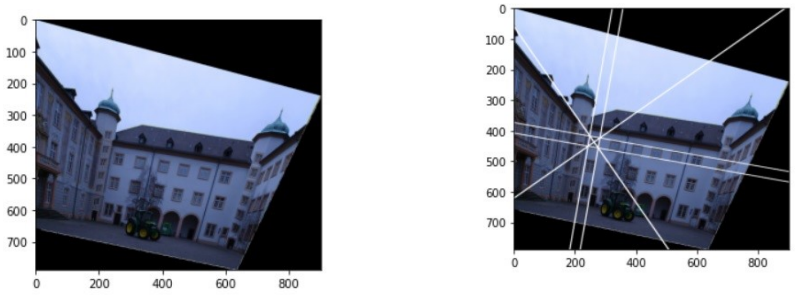
This method aims to recover the metric proportions of an image. For that purpose, we need two sets of orthogonal lines in order to have two linear equations from Eq. 9, which will be used to compute the parameters $\overrightarrow{s}$.

$$(l_1m_1, l_1m_2 + l_2m_1, l_2m_2)\overrightarrow{s} = 0 \tag{9}$$

The two sets of lines **cannot be parallel**, otherwise the system of equations won't be linearly independent. Consequently, we create another set of two orthogonal lines from the intersections of the other four lines. These lines can be seen in Fig. 4d as the *diagonal* lines (notice that we consider the windows to be square). Solving the system of equations we obtain $\overrightarrow{s}$ as its null vector. From $\overrightarrow{s}$, we get the matrix S, which is decomposed using the Cholesky algorithm to compute K. Finally, we obtain the homography $H_{s\leftarrow a}$ as described in Eq. 10. We use $H_{s\leftarrow a}$ to metrically rectify the image. This process is shown in Fig. 5.

$$H_{s\leftarrow a} = \begin{bmatrix} K^{-1} & \overrightarrow{0} \\ \overrightarrow{0^T} & 1 \end{bmatrix} \tag{10}$$

We can observe in Fig. 5b that the geometrical items (i.e. windows) have recovered its original shape. Furthermore, angles before and after the metric



(a) Metric rectified image

(b) Transformed lines are orthogonal after metric rectification

Fig. 5: Metric rectification - Image 0000

Table 3: Angles (^{o}) between orthogonal lines in the world before and after metric rectification - Image 0000

Image	L1/L3	L2/L4	L5/L6
Affinely rectified	72.64	72.64	72.01
Metric rectified	90	90	90

(a) Original

(b) Cropped

Fig. 6: Image 0001

rectification are presented in Tab. 3. These results confirm us that the metric rectification is properly performed, as orthogonal lines recover its 90 degrees.

5 Affine and Metric Rectifications on Left Facade

The objective of this exercise is to rectify the left facade of the image 0001 with the stratified method. To do so, we have to perform an affine rectification followed by a metric one, as explained in Section 3 and 4, respectively.

First of all, we crop the original image (Fig. 6a) at column 465 to obtain an image containing only the left facade (Fig. 6b). Otherwise, the resulting (rectified) image would be too distorted.

5.1 Affine Rectification

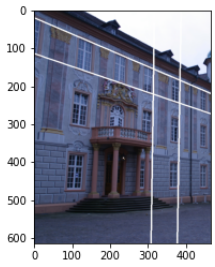
The affine rectification is shown in Fig. 7.

The angles between the two sets of parallel lines (L1/L2 and L3/L4) before and after the affine rectification are presented in Tab. 4. As expected, the angles are 0^{o} after the affine rectification, so the parallel lines in the world are also parallel in the rectified image.

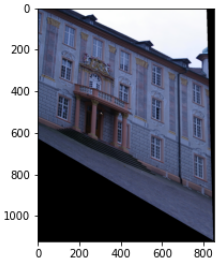
5.2 Metric Rectification

The last step is to perform metric rectification. In this exercise, we realized that we had to add a detail when computing the transformed lines. In some cases,

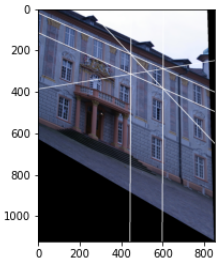
(a) Cropped image



(b) Lines that are parallel in the world



(c) Affinely rectified image



(d) Transformed lines are parallel in the affinely rectified image

Fig. 7: Affine rectification of the left facade (image 0001).

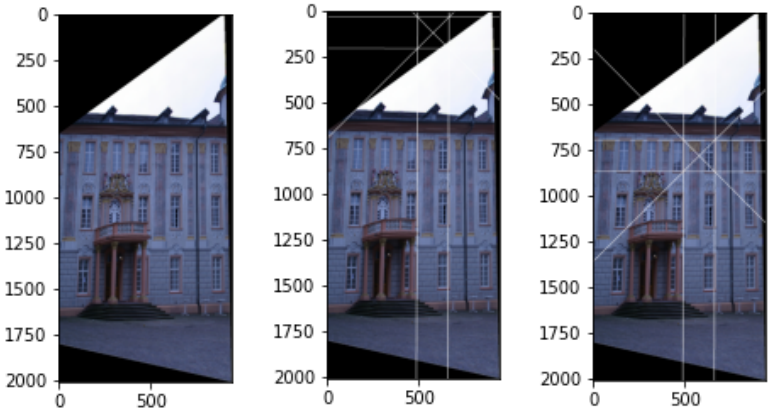
Table 4: Angles (^{o}) between parallel lines in the world before and after affine rectification - left facade of image 0001.

Image	L1/L2	L3/L4
Original	4.4	0.13
Affinely rectified	0.0	0.0

the corners of the rectified image may have negative coordinates, but the plot axis always starts at (0,0). In other words, the plot function is compensating the negative coordinates by moving the image to the (0,0). In these situations, we need to take this compensation into account when transforming the lines, so they appear where they should be.

After performing a metric rectification in the left facade of image 0001, the top right corner of the resulting image has a negative height (coordinate $y = -662$). Therefore, we need to compensate the transformed lines to obtain an accurate representation. This process is shown in Fig. 8.

The angles between the three sets of orthogonal lines (L1/L3, L2/L4 and L5/L6) before and after the metric rectification are presented in Tab. 5. As



(a) (b) (c)

Fig. 8: Metric rectification of the left facade (image 0001) with (a) no lines, (b) incorrect transformed lines and (c) corrected lines.

Table 5: Angles (^{o}) between orthogonal lines in the world before and after metric rectification - left facade of image 0001.

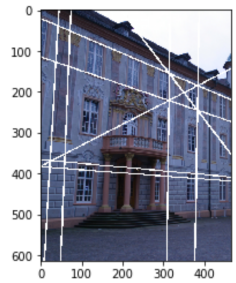
Image	L1/L3	L2/L4	L5/L6
Affinely rectified	113.8	113.8	124.14
Metric rectified	90.0	90.0	90.0

expected, the angles are 90^{o} after the metric rectification, so the orthogonal lines in the world are also orthogonal in the metric rectified image.

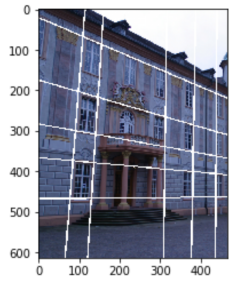
6 Optional. Metric rectification on a single step

The goal of this exercise is to rectify an image on a single step, as an alternative to the stratified method. A first approach following the algorithm proposed in pages 55-57 of Hartley-Zisserman book[1] is taken. We used two different sets of lines, as seen in Fig.9a and Fig.9b, obtaining the results in Fig. 9c and Fig. 9d, respectively. As it can be observed, the rectification failed for both sets.

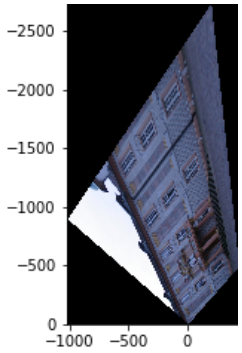
A second approach based on the slides was implemented as well, but results couldn't be obtained for the second set of lines as the generated matrix was not positive definite. Our belief is that the error relies on the set of lines chosen, as some of them might be parallel and therefore the algorithm fails to rectify correctly the image.



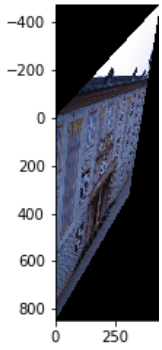
(a) Set of lines 1



(b) Set of lines 2



(c) Rectified set 1



(d) Rectified set 2

Fig. 9: Single step metric rectification of the left facade (image 0001).

7 Conclusions

In general terms, we have been able to correctly implement the different projective transformations and rectifications. The results have been certified both visually and numerically, computing the angles. We have seen that selecting the right set of point and lines is critical for the correct behaviour of the rectification methods.

Despite obtaining good results, we have also faced some problems. On the one hand, we had trouble identifying the correct values of the axis, and it was difficult to see if an image was translated correctly. Furthermore, this caused trouble when displaying transformed lines over an image, as we were not taking into account that the axis were not starting at $(0, 0)$ anymore, so the lines had to be adjusted. On the other hand, we had problems making metric rectification in one step. We think that this problem is caused by the fact that the selected pairs of orthogonal lines is not the optimal one.

References

1. R. Hartley and A. Zisserman, *Multiple View Geometry*, 2nd ed. Cambridge University Press, 2000.