Image analysis and computer vision homework

Emiliano Gagliardi

2018
February

1 Assignment

1.1 Given image and scene information

Figure 1: The input image

An image of a bandoneon is taken by a zero-skew camera (natural camera can NOT be assumed).
In the given image, the bandoneon is placed on a horizontal floor. Four rectangular faces are visible in the image: two coplanar horizontal faces, and two non-coplanar vertical faces. The long side of the horizontal faces is 243 mm, and it is slightly longer than the long side of the vertical face.
For each horizontal face, two groups of parallel lines can easily be identified. The two groups of lines are mutually perpendicular. One of the (short) lines on each horizontal face is also common to a vertical face. Part of the horizontal floor is also visible: we can assume to see groups of parallel lines (the groups are also mutually orthogonal): These lines can be used to help in robustly find, e.g., the image of the horizontal vanishing line (i.e. the image of the line at the infinity of the horizontal plane). We can NOT assume square patterns on the floor.
In addition, the long lines in the vertical faces are vertical (i.e. orthogonal both to the horizontal faces and to the floor).

1.2 Assignments

- Image feature extraction and selection: use the learned techniques to find edges and lines in the image. Then manually select those lines which are useful for the subsequent steps.
- Geometry:

- Using constraints on the horizontal lines, and their images, reconstruct the shape of the horizontal faces, and determine their relative position and orientation.
- Using also the images of vertical lines, calibrate the camera (i.e., determine the calibration matrix K) assuming it is zero-skew (but not assuming it is natural).
- Localize the camera with respect to (both) horizontal faces. From the image of the (short) horizontal segments common to a horizontal face and its neighboring vertical face, reconstruct the shape of the vertical faces.

2 Lines extraction

An expression for some lines is needed for solving the steps of the homework. First of all the canny algorithm is applied to the input image to obtain the image of the edges. The two thresholds have been manually fine tuned aiming at finding good lines on the instrument rather than the floor, since the floor presents a complex pattern. Indeed a severe enough threshold has been placed in the canny algorithm, to remove the big part of the noisy points on the right horizontal face of the instrument. This makes the interesting lines on the floor to vanish. The second step is to apply the hough space algorithm in order to find lines on the image containing the edges. Also here the parameters are manually tuned searching in particular for the lines that represents the edges of the horizontal faces and some vertical lines. These lines are indispensable for a minimal solution of the following points of the homework.
Since the checkerboard on the floor is very useful both for calibration and shape reconstruction, I run another time the lines extraction algorithm, in order to extract some lines on it. This time the canny threshold is very low, in order to preserve the floor lines, and the filtering is more severe on the length of the lines, so only long lines are accepted in the hough space algorithm.

Figure 2: The result of canny for extracting lines on the faces

3 Reconstruction of the horizontal plane

Using the lines obtained as described in the previous section, a reconstruction of the horizontal plane up to a similarity is obtained in two steps:

1. Compute the image of the line at the infinity l_∞ of the horizontal plane to reconstruct the scene up to an affine rectification

Figure 3: Lines extracted on the instrument

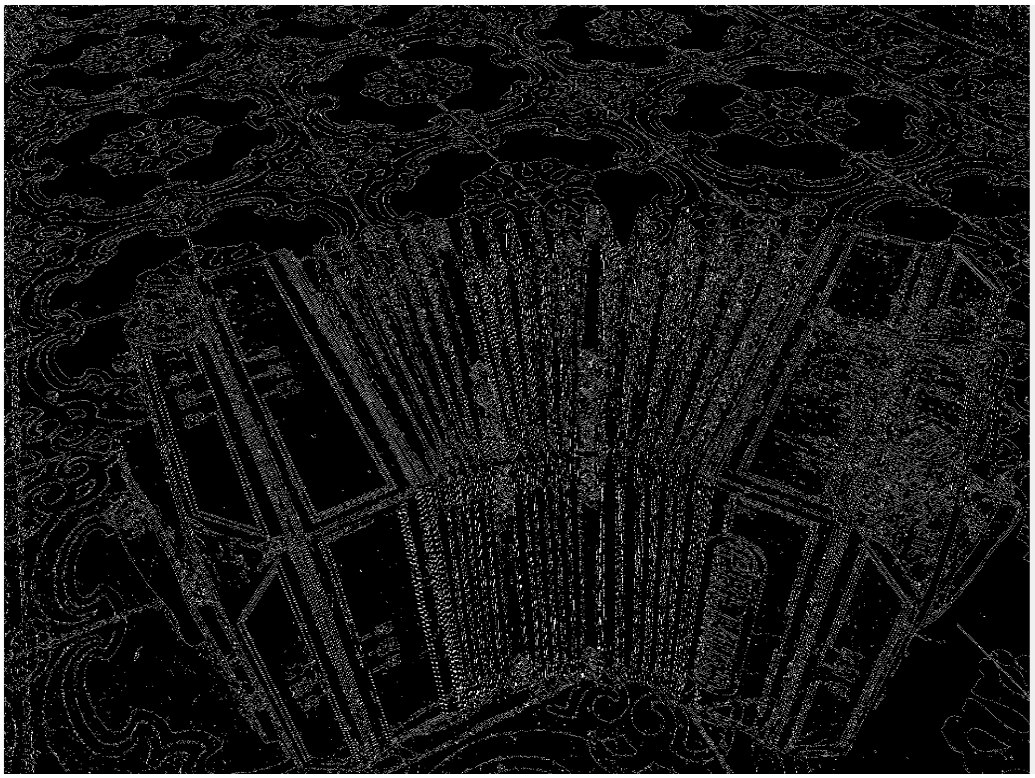

Figure 4: The result of canny for extracting lines on the floor

2. Compute the image of the degenerate dual conic $C_\infty^* = IJ^T + JI^T$ in the affine rectification, where I and J are the circular points of the horizontal plane in the scene.

Suppose we have a set of vanishing points images $V = \{v'_1, ..., v'_n\}$, where v_i belongs to an horizontal plane in the real scene. To obtain the affine reconstruction matrix of the horizontal plane, we need to fit the image of

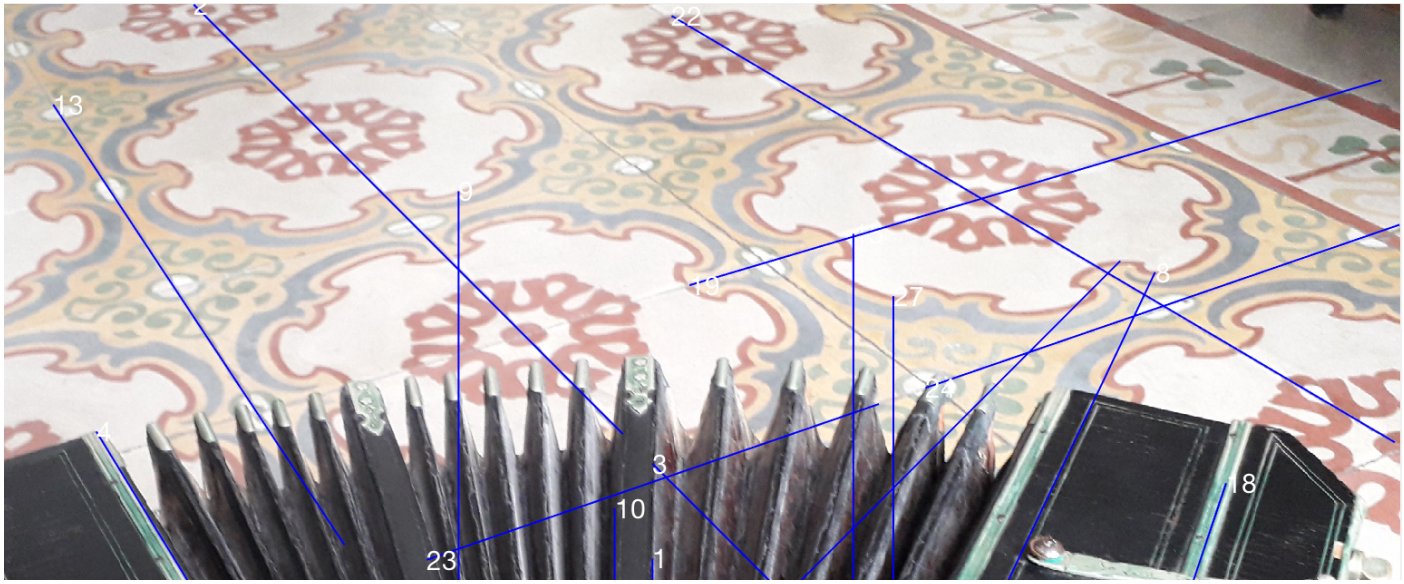

Figure 5: Lines extracted on the floor

the line at infinity against the elements of V. The used fitting algorithm is total least square:

$$\min \sum_{v_i' \in V} d_i^2$$

Where d_i is the orthogonal distance between the point v_i' and the line we want to fit.
Expressing the line l_∞' using as parameters (ρ, θ), where ρ is the orthogonal distance between the origin and l_∞', and θ is the angle that the line orthogonal to l_∞' form with the horizontal axis, we find the following expression for d_i:

$$d_i = x_i \cos\theta + y_i \sin\theta - \rho$$

where $v_i' = \begin{pmatrix} x_i & y_i \end{pmatrix}$. Rewriting $\cos\theta$ as a, $\sin\theta$ as b, and ρ as c, the problem becomes the minimization of a quadratic function under a quadratic constraint:

$$\min \sum_{v_i' \in V} (ax_i + by_i - c)^2$$
$$\text{subject to} \quad a^2 + b^2 = 1$$

Let $\bar{x}$ be the mean of all the x_i and $\bar{y}$ be the mean of all the y_i, this problem has closed form solution:

$$\begin{pmatrix} a & b \end{pmatrix} = eig_1$$
$$c = -a\bar{x} - b\bar{y}$$

Where eig_1 is the eigenvector associated to the smallest eigenvalue of the matrix:

$$A = \begin{pmatrix} \bar{x^2} - \bar{x}^2 & \bar{xy} - \bar{x}\bar{y} \\ \bar{xy} - \bar{x}\bar{y} & \bar{y^2} - \bar{y}^2 \end{pmatrix}$$

Given the solution of the above problem, we can write the image of the line at infinity in homogeneous coordinates as:

$$l_\infty' = \begin{pmatrix} a/b & 1 & c/b \end{pmatrix}^T$$

Now we only need to show how to obtain the set of vanishing points images V. First of all notice that, for the duality principle, the fitting algorithm just explained can be used to sove the overconstrained problem of finding the intersection point among a set of lines. So for the computation of v_i' we will call the total least square algorithm on the set of lines L_i, that should satisfy the following:

- The lines contained in L_i are all image of lines known to be parallel among each other in the real scene
- All the lines in L_i belongs to an horizontal plane

In the end, after the fitting of all the elements of V and the fitting of l'_∞, the affine rectification transformation is:

$$H_{\text{aff}} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ a/b & 1 & c/b \end{pmatrix}$$

In practice, the vanishing points used in the algorithm are obtained from:

- The parallel lines that can be seen on the floor
- Two groups of lines for the left horizontal face: the group of lines parallel to the long edges, and the group of lines parallel to the short edges
- The same for the right horizontal face

Figure 6: The obtained affine reconstruction

Now that we have obtained an affine rectification of the image, we have to transform the lines that are used in the following algorithm using H_{aff}:

$$l^T_{\text{aff}} = l^T * H^{-1}_{\text{aff}}$$

The choice of passing through an affine rectification before doing shape reconstruction is motivated by the following consideration. Since the line at infinity is invariant under affine transformation, the two images of the circular points in the affine rectified image are at the infinity, and so we can write:

$$C^{*'}_\infty = I'J'^T + J'I'^T = \begin{pmatrix} i^{(1)} \\ i^{(2)} \\ 0 \end{pmatrix} \begin{pmatrix} j^{(1)} & j^{(2)} & 0 \end{pmatrix} + \begin{pmatrix} j^{(1)} \\ j^{(2)} \\ 0 \end{pmatrix} \begin{pmatrix} i^{(1)} & i^{(2)} & 0 \end{pmatrix} = \begin{pmatrix} i^{(1)}j^{(1)} & i^{(1)}j^{(2)} & 0 \\ i^{(2)}j^{(1)} & i^{(2)}j^{(2)} & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

Since we are working in homogeneous coordinates we can write:

$$C^{*'}_\infty = \begin{pmatrix} a & b & 0 \\ b & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

That remains with only two degrees of freedom, where the other two have been solved with the affine reconstruction in the previous step. The previous use of an affine reconstruction is a robust approach, since the image presents a sufficient number of long parallel lines, and we can trust in the fitting of the vanishing points and the line at infinity. Also this method guarantees that $C^{*'}_\infty$ is singular, without imposing any quadratic constraint.
Now we are ready to apply the least square algorithm to obtain the conic, using as model the equation $l^T C^{*'}_\infty m = 0$ where l and m are the images of lines known to be perpendicular in the real scene.

$$l^T C^{*'}_\infty m = \begin{pmatrix} l^{(1)} & l^{(2)} & l^{(3)} \end{pmatrix} \begin{pmatrix} a & b & 0 \\ b & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} m^{(1)} \\ m^{(2)} \\ m^{(3)} \end{pmatrix} = 0$$

That results in the model:

$$\begin{pmatrix} l^{(1)}m^{(1)} & l^{(1)}m^{(2)} + l^{(2)}m^{(1)} \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = -l^{(2)}m^{(2)}$$

And we can compute a and b with the classical least square closed form solution.
Then SVD is applied to $C^{*'}_\infty$, to obtain the transformation matrix that maps the affine rectification of the image

previously found to a similarity with respect to the real scene. Here a simple trick is needed, since applying SVD we get the following result:

$$\mathrm{SVD}(C_\infty^{*'}) = U \begin{pmatrix} h & 0 & 0 \\ 0 & k & 0 \\ 0 & 0 & 0 \end{pmatrix} U^T$$

Where h and k are different from 1. Thus by rewriting the matrix in between in the above product as:

$$MC_\infty^* M = \begin{pmatrix} \sqrt{h} & 0 & 0 \\ 0 & \sqrt{k} & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \sqrt{h} & 0 & 0 \\ 0 & \sqrt{k} & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

We see that the desired reconstruction matrix is $H_{a\to s} = (UM)^{-1}$.
Now we can apply $H_{a\to s}$ to the affinely rectified image, or compute the composition of the two transformations and apply it directly to the input image: $H_{\mathrm{rec}} = H_{a\to s} H_{\mathrm{aff}}$.
Also here the used lines are the one belonging to the floor and to the horizontal faces.

Figure 7: The obtained reconstructed shape

4 Relative position of the two faces

Let's call O_l and O_r the two frames respectively placed on the left and on the right horizontal faces ot the instrument as follows: the origin is the bottom left point of the face, the y axis is directed toward the up left point of the face, and the x axis toward the bottom right point. The relative localization problem is solved by finding an expression for O_r in O_l.
Since we want to work in O_l, whit objects represented by their real shape and dimensions, we fit the homography from the image of the vertices of the left face to the points $(0,0),(0,L),(S,L),(S,0)$, where L and S are respectively the length of the long and the short edges in the real world. The images of the vertices can be obtained by intersecting the lines found in the image corresponding to the edges of the face (the lines are manually selected). The value L is given and it is 243 mm, while the length of the short side can be computed obtaining the ratio between L and S from the reconstructed image. After the fitting of the homography all the lines in the image are transformed to obtain their expression in O_l.
Now we can proceed for the computation of the rotation matrix $R_{O_r}^{O_l}$ that expresses the orientation of O_r in O_l, and the position vector $t_{O_r}^{O_l}$ that expresses the origin of O_r in O_l. For robustness, the angle between the two

faces is computed by averaging all the angles between two long lines (parallel to the long edge of their face) belonging to different faces. Since the two faces are on the same plane, $R^{O_l}_{O_r}$ si the rotation matrix around the axis z of the found angle. To determine $t^{O_l}_{O_r}$ is sufficient to compute the intersection points between the left long edge and the bottom short edge of the right face using the transformed lines.

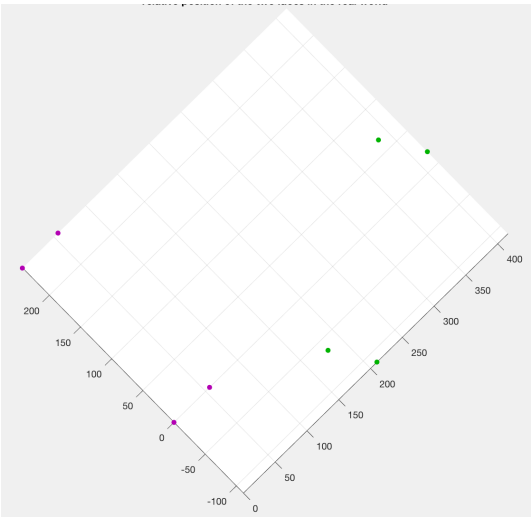

Figure 8: Vertices of the two horizontal faces represented in O_l, the right face ratio is computed as for the left one, and the long side length is given (equal to the one of the left face)

5 Calibration

The calibration problem is solved by imposing constraints on the image of the absolute conic. There are information enough in the image to make the problem overconstrained, that helps in robustness. The kind of constraints that have been used are the one deriving from vanishing points of orthogonal directions in the real scene:

$$v_1^T W v_2 = 0$$

Where W is the matrix representing the IAC. The used vanishing points are the ones obtained from:

- Vertical lines
- Two directions on the floor
- Two directions for each of the horizontal face

As in the reconstruction step a general model can be derived for fitting W using least square.
As described in one of the hints on the homework sheet the IAC can be written as function of only camera internal parameters, and so from W the calibration matrix K is easily computed.

6 Camera localization

6.1 Camera localization with respect to the left face

For simplicity, first the left face frame O_l is expressed in the camera frame, and then things are reversed to obtain an expression of the camera frame in O_l. Let's start with the localization of the left face frame in the camera frame. A generic point X on the plane of the horizontal faces can be written in O_l as $X^{O_l} = \begin{pmatrix} x & y & 0 & w \end{pmatrix}^T$. Its expression in the world reference frame is:

$$X^w = \begin{pmatrix} i^w_{O_l} & j^w_{O_l} & k^w_{O_l} & t^w_{O_l} \\ 0 & 0 & 0 & 1 \end{pmatrix} X^{O_l} = \begin{pmatrix} i^w_{O_l} & j^w_{O_l} & t^w_{O_l} \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ w \end{pmatrix}$$

Where $i^w_{O_l}$, $j^w_{O_l}$, $k^w_{O_l}$ are the expression of the versors of O_l in the word reference frame, and $t^w_{O_l}$ is the position of the origin of O_l in the word reference frame.

From the Pinhole model, fixing the world reference frame equal to the camera reference frame, we have:

$$X' = \begin{pmatrix} K & \begin{matrix}0\\0\\0\end{matrix} \end{pmatrix} \begin{pmatrix} i^c_{O_l} & j^c_{O_l} & t^c_{O_l} \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ w \end{pmatrix} = K \begin{pmatrix} i^c_{O_l} & j^c_{O_l} & t^c_{O_l} \end{pmatrix} \begin{pmatrix} x \\ y \\ w \end{pmatrix}$$

Now, fitting an homography H such that $X' = H \begin{pmatrix} x & y & w \end{pmatrix}^T$

$$\begin{pmatrix} i^c_{O_l} & j^c_{O_l} & t^c_{O_l} \end{pmatrix} = K^{-1}H$$

The homography H maps points on the horizontal plane expressed in O_l to points on the image, and so we already have computed its inverse in the solution of the faces relative localization problem.
In practice we need to normalize $i^c_{O_l}$ and $j^c_{O_l}$. Their norm should be very similar (not the same for numerical error) and we multiply $t^c_{O_l}$ by it to obtain it in real scale. The $k^c_{O_l}$ vector is obtained with the cross product between $i^c_{O_l}$ and $j^c_{O_l}$.
$R^c_{O_l} = \begin{pmatrix} i^c_{O_l} & j^c_{O_l} & k^c_{O_l} \end{pmatrix}$ is the rotation matrix that represent the orientation of O_l in the camera frame, and $t^c_{O_l}$ the vector that represents the origin of O_l in the camera frame. For numerical error the computed $R^c_{O_l}$ be not orthonormal, and so it is approximated using SVD. To localize the camera in O_l we simply compute:

$$R^{O_l}_c = R^{c^T}_{O_l} \quad t^{O_l}_c = -R^{O_l}_c t^c_{O_l}$$

6.2 Camera localization with respect to the right face

Since we have already localized the camera with respect to the right face, and we have localized the right face with respect to the camera, we only need to do some matrix inversion and multiplication to obtain $R^{O_r}_c$ and $t^{O_r}_c$. By setting:

$$A^{O_l}_{O_r} = \begin{pmatrix} R^{O_l}_{O_r} & & t^{O_l}_{O_r} \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad A^{O_l}_c = \begin{pmatrix} R^{O_l}_c & & t^{O_l}_c \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad A^{O_r}_c = \begin{pmatrix} R^{O_r}_c & & t^{O_r}_c \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

We have:

$$A^{O_r}_c = A^{O_r}_{O_l} A^{O_l}_c = A^{O_l^{-1}}_{O_r} A^{O_l}_c$$

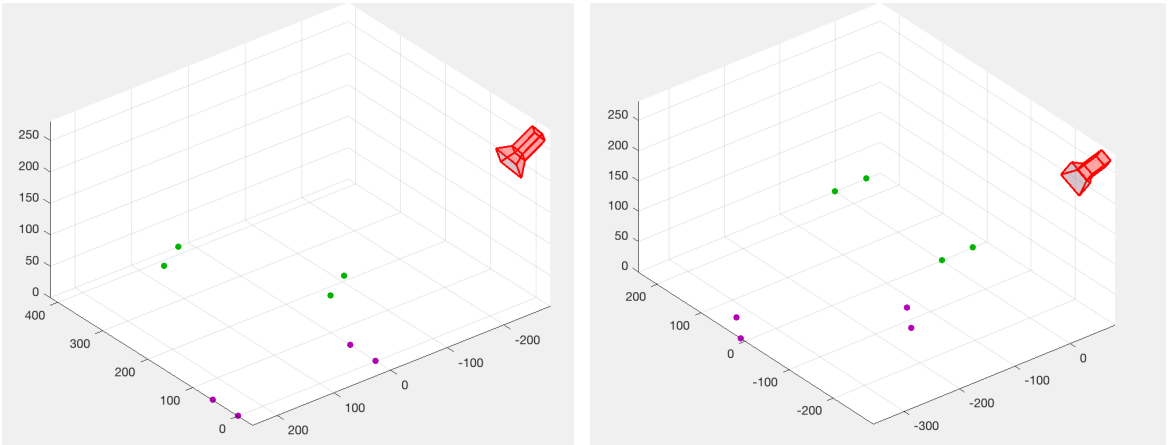

Figure 9: The faces and the camera represented in O_l (left) and O_r (right)

6.3 Vertical faces reconstruction

Now the problem to be solved is the shape reconstruction of the vertical faces of the instrument. Let's start with the left face. In the previous step we derived the position of the camera in O_l, so we have the camera Pinhole model where the world reference frame is equal to O_l.

$$P = K[R^c_{O_l} | t^c_{O_l}]$$

Now, given that points that are on the left vertical faces can be written in O_l as $X = \begin{pmatrix} x & 0 & z & w \end{pmatrix}^T$, we obtain their images as:

$$X' = PX = \begin{pmatrix} P^{(1)} & P^{(3)} & P^{(4)} \end{pmatrix} \begin{pmatrix} x \\ z \\ w \end{pmatrix}$$

Where $P^{(i)}$ is the i-th column of P. So we have an homography from the real world points that are on the left vertical face plane to the image. Its inverse is the homography from the image to the real world points, and so it is the reconstruction matrix for points that are images of points on the left vertical face plane in the real world.
The same reasoning can be done for the horizontal plane, obtaining a reconstruction of the two horizontal faces. By changing the extrinsic parameters, and using the ones that express the position of the right face in the camera frame, we obtain the reconstruction of the right vertical face.

Figure 10: Reconstruction of the two vertical faces

6.4 Localization of features on the vertical faces

In this step, since no significant point can be found on the vertical faces by using the lines previously obtained, the harris features extraction algorithm is applied to the input image. The obtained points will be expressed in O_l.
The reconstruction matrices obtained in the previous steps, do not only reconstruct the shape, but also bring image points that are on the respective vertical plane to the real world. It is only needed to make some change of reference system.
Let's start with the reconstruction matrix of the left vertical face H_l. The x and y axis of the output space of this transformation are respectively the top and the left edge of the left vertical face (x toward right, y toward the top). I will call this space O_{vl}. The only things to do are:

- Choose some points on the left vertical face
- Apply H_l to them
- Rotate of -90 degrees around the x axis, that is common to O_l and O_{vl}

The same three steps can be applied to points on the right vertical face (using this time H_r, the reconstruction matrix of the right vertical face plane) to obtain real world points expressed in O_r. Since the transformation between O_l and O_r have been computed in one of the previous steps, only remains to transform the real words points belonging the right vertical face and express them in O_l.

7 Numerical results

The assignment requires to provide the obtained numerical result for:

- Line at infinity
- Relative coordinates for planar shape reconstruction

- Calibration parameters
- Camera pose
- Coordinates of some interesting point on the vertical faces

Image of the line at infity in homogeneous coordinates, expressed in the image frame:

$$l'_\infty = \begin{pmatrix} 0.000039252049664 & 0.000701312058441 & 1.000000000000000 \end{pmatrix}$$

Horizontal left face vertices coordinates in O_l, expressed in millimeters (the face is considered to be only the black part here):

$$\text{bottom left} = \begin{pmatrix} 0.000000000000000 & 0.000000000000000 & 0.000000000000000 \end{pmatrix}$$

$$\text{top left} = \begin{pmatrix} 0.000000000000000 & 243.000000000000000 & 0.000000000000000 \end{pmatrix}$$

$$\text{bottom right} = \begin{pmatrix} 56.0488222233330 & 0.000000000000000 & 0.000000000000000 \end{pmatrix}$$

$$\text{top right} = \begin{pmatrix} 56.0488222233330 & 243.000000000000000 & 0.000000000000000 \end{pmatrix}$$

Horizontal right face vertices coordinates in O_l, expressed in millimeters (the face is considered to be only the black part here):

$$\text{bottom left} = \begin{pmatrix} 179.9689893778533 & -63.5863246136970 & 0.000000000000000 \end{pmatrix}$$

$$\text{top left} = \begin{pmatrix} 385.9100169134065 & 65.3993070574335 & 0.000000000000000 \end{pmatrix}$$

$$\text{bottom right} = \begin{pmatrix} 210.0342829497356 & -111.5891709085322 & 0.000000000000000 \end{pmatrix}$$

$$\text{top right} = \begin{pmatrix} 415.9753104852888 & 17.3964607625982 & 0.000000000000000 \end{pmatrix}$$

Calibration parameters:

$$K = \begin{pmatrix} 344.6194103369809 & 0.000000000000000 & 214.6836403592702 \\ 0.000000000000000 & 330.9038610937151 & 125.2141570880580 \\ 0.000000000000000 & 0.000000000000000 & 0.001000000000000 \end{pmatrix}$$

Camera pose in O_l:

$$R = \begin{pmatrix} 0.875176135402261 & -0.335196685169431 & 0.348869480284232 \\ -0.481486765159806 & -0.532949742608249 & 0.695798150924351 \\ -0.047299334015507 & -0.776921974271050 & -0.627817663734036 \end{pmatrix}$$

$$t = \begin{pmatrix} -7.5776862974045 & -264.9077723257021 & 320.8476444138181 \end{pmatrix}$$

Camera pose in O_r:

$$R = \begin{pmatrix} 0.872605044486171 & 0.273747577970796 & -0.404503028285836 \\ 0.486130856188927 & -0.566969584237018 & 0.664997955794696 \\ -0.047299334015507 & -0.776921974271050 & -0.627817663734036 \end{pmatrix}$$

$$t = \begin{pmatrix} 71.0679809751512 & -265.8071160556344 & 320.8476444138181 \end{pmatrix}$$

For the vertical faces the localized points are the ones shown in Fugure 11.
Left vertical face points in O_l:

$$\text{point1} = \begin{pmatrix} 7.64436 & 0.00000 & -8.06846 \end{pmatrix}$$

$$\text{point2} = \begin{pmatrix} 25.81392 & 0.00000 & -119.76267 \end{pmatrix}$$

$$\text{point3} = \begin{pmatrix} 6.94372 & 0.00000 & -218.20204 \end{pmatrix}$$

$$\text{point4} = \begin{pmatrix} 42.84373 & 0.00000 & -220.73520 \end{pmatrix}$$

Right vertical face points in O_l:

$$\text{point1} = \begin{pmatrix} 186.38826 & -73.83546 & -9.37461 \end{pmatrix}$$

$$\text{point2} = \begin{pmatrix} 206.75604 & -106.35508 & -9.81595 \end{pmatrix}$$

$$\text{point3} = \begin{pmatrix} 185.06279 & -71.71918 & -53.89124 \end{pmatrix}$$

$$\text{point4} = \begin{pmatrix} 180.25151 & -64.03740 & -233.93948 \end{pmatrix}$$

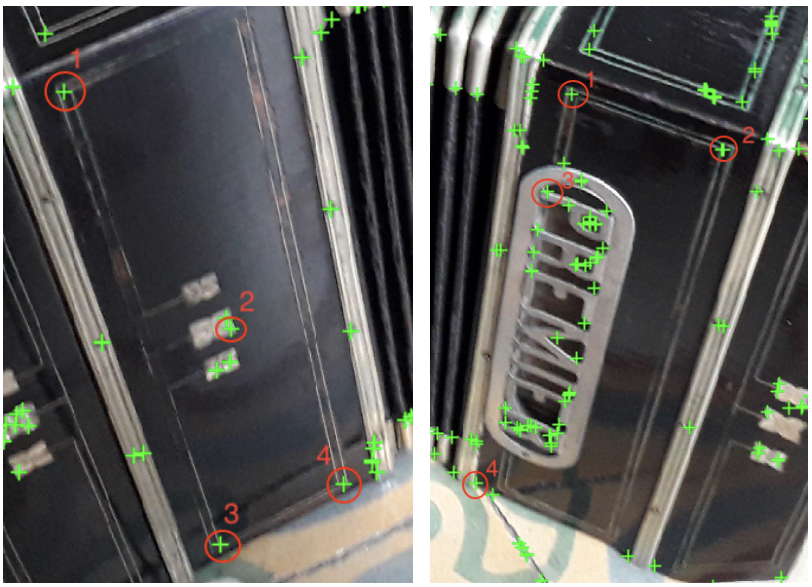

Figure 11: Points on the vertical faces that have been localized

8 Project structure and how to run

The project is organized in a matlab folder, containing:

- The input image
- The folder `reusable_code`
- Some function files
- The `workflow.m` script

The folder `reusable_code` contains functions general enough to be reused in other projects (all the assumptions they made are well commented), like `calibration.m`, `affine_rectification.m`, `reconstruct2D.m` and others, like fitting algorithms. Files outside this folder are custom for the project, and have not been written to be reusable.
The `workflow.m` script contains the execution of all the steps in order, and manage the following operations:

- Calls the functions that interact with the user for manual selection of lines and features
- Organizes the data for the functions and calls them
- Plot results
- Execute some simple geometry steps that have not been delegated to a function

In the script is present the `interactive` variable, a boolean that you can set to false if you want to skip the manual selection of the lines and you want the algorithm to use lines hardcoded in the `lines_selection.m` function.
To run the project you need to add the `reusable_code` folder to your path (right click on the folder, in the "Current folder" matlab window, then "Add to Path" → "Selected Folder and Subfolders"), then run the `workflow.m` script.