COVIS Lab 1: Camera Calibration

1 Theory

The target of this lab is to implement the Zhang's method for camera calibration which is the method of choice of OpenCV and MATLAB's computer vision toolbox. The detail of Zhang's method can be found in [1]. However, it is summarized in this section for the completeness of this document.

A pinhole camera model expresses the mapping from a 3D point $\mathbf{P} = [\mathrm{X}, \mathrm{Y}, \mathrm{Z}]^T$ to its $\mathbf{p} = [\mathrm{u}, \mathrm{v}]^T$ by the following equation

$$\lambda\, \widetilde{\mathbf{p}} = \mathbf{K} \begin{bmatrix} \mathbf{R} & \mathbf{t} \end{bmatrix} \widetilde{\mathbf{P}} \quad \text{with } \mathbf{K} = \begin{bmatrix} \alpha & c & \mathrm{u}_0 \\ 0 & \beta & \mathrm{v}_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{1}$$

here, $\widetilde{\mathbf{p}}$ and $\widetilde{\mathbf{P}}$ denote the homogeneous coordinate (with 1 padded to the end) of $\mathbf{p}$ and $\mathbf{P}$, respectively; $\mathbf{K}$ is camera intrinsic matrix; $[\mathbf{R}\ \mathbf{t}] \in \mathbb{R}^{3\times4}$ is the transformation from the frame relative to which $\mathbf{P}$ is identified to the camera frame; λ is a scaling factor. The problem is camera calibration is defined as identifying $\mathbf{K}, \mathbf{R}, \mathbf{t}$ given the 3D position of a set of $\mathbf{P}$ and the pixel coordinate of their images.

1.1 Homography from the plane and its image

Due to the need of 3D position of a set of points, calibration rig - an object which dimension is known is an essential instrument for camera calibration. Zhang's method requires arguably the simplest calibration rig - a chessboard as shown in Figure.1.

Assume the chessboard is placed at the plane $\mathrm{Z} = 0$, Equation.(1) becomes

$$\lambda\, \widetilde{\mathbf{p}} = \mathbf{K} \begin{bmatrix} \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{t} \end{bmatrix} \begin{bmatrix} \mathrm{X} \\ \mathrm{Y} \\ 1 \end{bmatrix} \tag{2}$$

Here, $\mathbf{r}_1, \mathbf{r}_2$ are the first two columns of $\mathbf{R}$. By abuse of notation, let $\mathbf{P} = [\mathrm{X},\ \mathrm{Y}]^T$. Equation.(2) shows that the 3D point $\mathbf{P}$ on the plane $\mathrm{Z} = 0$ and its image $\mathbf{p}$ are related by a homography $\mathbf{H}$

$$\lambda\, \widetilde{\mathbf{p}} = \mathbf{H}\, \widetilde{\mathbf{P}} \quad \text{with } \mathbf{H} = s\, \mathbf{K} \begin{bmatrix} \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{t} \end{bmatrix} \tag{3}$$

here, s is a scale factor

Given a set of 3D points $\{\mathbb{P}_i\}_{i=1}^n$ which positions are known and their images, the homography $\mathbf{H}$ is identified as following. Apply Equation.(3) to the point $\mathbf{P}_i$ and its image

$$\lambda \begin{bmatrix} \mathrm{u}_i \\ \mathrm{v}_i \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{h}_1^T \\ \mathbf{h}_2^T \\ \mathbf{h}_3^T \end{bmatrix} \mathbf{P}_i \tag{4}$$

here, $\mathbf{h}_i^T$ is the i-th row of $\mathbf{H}$. This equation results in

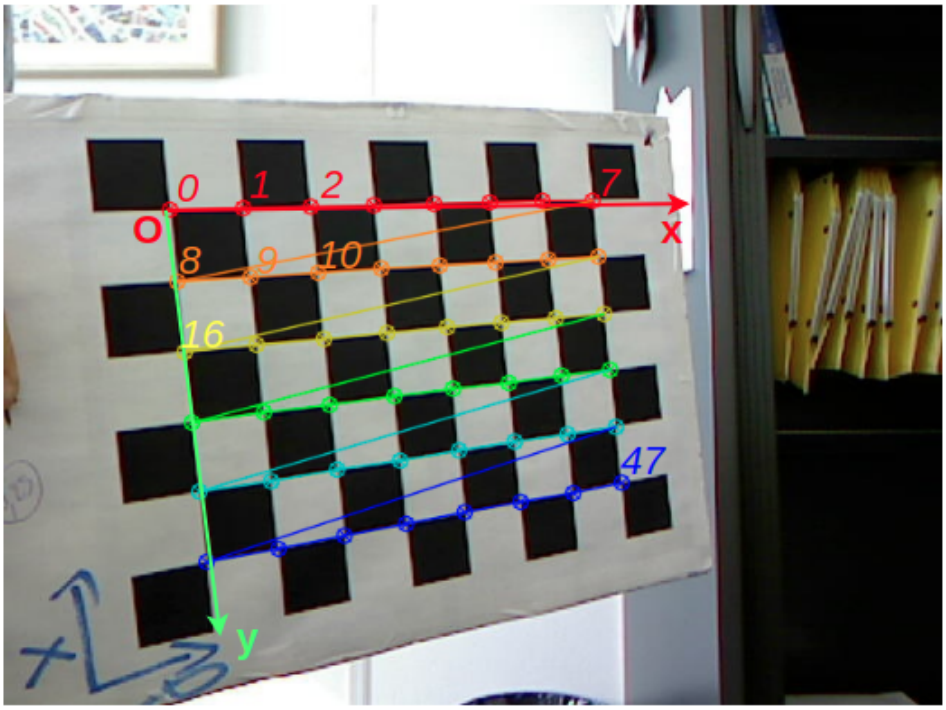

Figure 1: A chessboard used as calibration rig by Zhang's method. A grid of 6×8 is identified by corners marked by colored circles. Assume the chessboard is placed at the plane $Z = 0$ in the world frame and place the world frame as in the figure, the 3D position of every corners can be defined, given side of a square in the chessboard.

$$\begin{cases} u_i = \frac{\mathbf{h}_1^T \mathbf{P}}{\mathbf{h}_3^T \mathbf{P}} \\ v_i = \frac{\mathbf{h}_2^T \mathbf{P}}{\mathbf{h}_3^T \mathbf{P}} \end{cases} \tag{5}$$

which can be arranged into

$$\begin{cases} \left(\mathbf{h}_1^T - u_i \, \mathbf{h}_3^T\right) \mathbf{P}_i = 0 \\ \left(\mathbf{h}_2^T - v_i \, \mathbf{h}_3^T\right) \mathbf{P}_i = 0 \end{cases} \tag{6}$$

Equation.(6) can be rewritten into the matrix form as

$$\begin{bmatrix} \mathbf{P}_i^T & \mathbf{0}^T & -u_i \, \mathbf{P}_i^T \\ \mathbf{0}^T & \mathbf{P}_i^T & -v_i \, \mathbf{P}_i^T \end{bmatrix} \begin{bmatrix} \mathbf{h}_1 \\ \mathbf{h}_2 \\ \mathbf{h}_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{7}$$

Equation.(7) shows that a 3D point $\mathbf{P}_i$ results in two equations for 9 elements of $\mathbf{H}$. Therefore, for can stack $2n$ equations resulted from n 3D points to have a homogeneous linear equation for 9 elements of $\mathbf{H}$

$$\mathbf{Q} \begin{bmatrix} \mathbf{h}_1 \\ \mathbf{h}_2 \\ \mathbf{h}_3 \end{bmatrix} = \mathbf{0} \quad \text{with } \mathbf{Q} = \begin{bmatrix} \mathbf{P}_1^T & \mathbf{0}^T & -u_1 \, \mathbf{P}_1^T \\ \mathbf{0}^T & \mathbf{P}_1^T & -v_1 \, \mathbf{P}_1^T \\ \vdots & \vdots & \vdots \\ \mathbf{P}_n^T & \mathbf{0}^T & -u_n \, \mathbf{P}_n^T \\ \mathbf{0}^T & \mathbf{P}_n^T & -v_n \, \mathbf{P}_n^T \end{bmatrix} \in \mathbb{R}^{2n \times 9} \tag{8}$$

Equation.(8) can solved by the SVD trick. In short, the solution of Equation.(8) is the right singular vector $\mathbf{Q}$ associated with the smallest singular value which can be found by performing

the Singular Value Decomposition (SVD) on $\mathbf{Q}$.

1.2 Finding Camera Calibration

1.2.1 Constraints on camera's intrinsic parameters

As shown in Equation.(3), the homography $\mathbf{H}$ is up to a scale with the product between $\mathbf{K}$ and camera pose.

$$\begin{bmatrix}\mathbf{h}_1 & \mathbf{h}_2 & \mathbf{h}_3\end{bmatrix} = s\ \mathbf{K}\ \begin{bmatrix}\mathbf{r}_1 & \mathbf{r}_2 & \mathbf{t}\end{bmatrix} \tag{9}$$

Here, $\mathbf{h}_i$ is the i-th column of $\mathbf{H}$ .Exploiting the fact that $\mathbf{r}_1$ and $\mathbf{r}_2$ are orthonormal, the following equations can be deduced

$$\mathbf{h}_1^T\ \mathbf{K}^{-T}\ \mathbf{K}^{-1}\ \mathbf{h}_2 = 0 \tag{10}$$

$$\mathbf{h}_1^T\ \mathbf{K}^{-T}\ \mathbf{K}^{-1}\ \mathbf{h}_1 = \mathbf{h}_2^T\ \mathbf{K}^{-T}\ \mathbf{K}^{-1}\ \mathbf{h}_2 \tag{11}$$

1.2.2 Closed-form solution

Let

$$\mathbf{B} = \mathbf{K}^{-T}\ \mathbf{K}^{-1} = \begin{bmatrix} \frac{1}{\alpha^2} & -\frac{c}{\alpha^2\beta} & \frac{c\mathrm{v}_0-\beta\mathrm{u}_0}{\alpha^2\beta} \\ -\frac{c}{\alpha^2\beta} & \frac{c^2}{\alpha^2\beta^2}+\frac{1}{\beta^2} & -\frac{c(c\mathrm{v}_0-\beta\mathrm{u}_0)}{\alpha^2\beta^2}-\frac{\mathrm{v}_0}{\beta^2} \\ \frac{c\mathrm{v}_0-\beta\mathrm{u}_0}{\alpha^2\beta} & -\frac{c(c\mathrm{v}_0-\beta\mathrm{u}_0)}{\alpha^2\beta^2}-\frac{\mathrm{v}_0}{\beta^2} & \frac{(c\mathrm{v}_0-\beta\mathrm{u}_0)^2}{\alpha^2\beta^2}+\frac{\mathrm{v}_0^2}{\beta^2}+1 \end{bmatrix} \tag{12}$$

Because $\mathbf{B}$ is symmetric, it can be defined by a 6D vector

$$\mathbf{b} = \begin{bmatrix}B_{11} & B_{12} & B_{22} & B_{13} & B_{23} & B_{33}\end{bmatrix}^T \tag{13}$$

Let the i-th column of $\mathbf{H}$ be $\mathbf{h}_i = [h_{i1}, h_{i2}, h_{i3}]^T$, then

$$\mathbf{h}_i^T\ \mathbf{B}\ \mathbf{h}_j = \mathbf{v}_{ij}^T\mathbf{b} \tag{14}$$

with

$$\mathbf{v}_{ij} = \begin{bmatrix}h_{i1}h_{j1}, & h_{i1}h_{j2}+h_{i2}h_{j1}, & h_{i2}h_{j2}, & h_{i3}h_{j1}+h_{i1}h_{j3}, & h_{i3}h_{j2}+h_{i2}h_{j3}, & h_{i3}h_{j3}\end{bmatrix}^T \tag{15}$$

With this notation, the constraints in Equation.(10) and Equation.(11) can be rewritten as

$$\begin{bmatrix}\mathbf{v}_{12}^T \\ (\mathbf{v}_{11}-\mathbf{v}_{22})^T\end{bmatrix}\ \mathbf{b} = \begin{bmatrix}0\\0\end{bmatrix} \tag{16}$$

Equation.(16) shows that an image of the calibration rig (chessboard) provides two equations for $\mathbf{b}$ from which the intrinsic parameters can be retrieved. If n images of the chessboard are taken, the constraints (in the form of Equation.(16)) they induced can be stacked into a homogeneous linear equation for $\mathbf{b}$

$$\mathbf{V}\ \mathbf{b} = \mathbf{0} \tag{17}$$

Similar to Equation.(8), $\mathbf{b}$ can be solved from the equation above using the SVD trick. Once $\mathbf{b}$ is obtained, the intrinsic parameters can be found using Equation.(12), thus the intrinsic matrix $\mathbf{K}$ being found.

Given the value of $\mathbf{K}$ and the homography $\mathbf{H}$, camera pose can be calculated from Equation.(9)

$$\begin{aligned} \mathbf{r}_1 &= \upsilon \mathbf{K}^{-1}\mathbf{h}_1 \\ \mathbf{r}_2 &= \upsilon \mathbf{K}^{-1}\mathbf{h}_2 \\ \mathbf{r}_3 &= \mathbf{r}_1 \times \mathbf{r}_2 \\ \mathbf{t} &= \upsilon \mathbf{K}^{-1}\mathbf{h}_3 \end{aligned} \tag{18}$$

here, $\upsilon = 1/||\mathbf{K}^{-1}\mathbf{h}_1||$

2 Implementation

The theory presented in the previous section can be implemented as in the algorithm below

Algorithm 1: Camera Calibration with Zhang's method

Result: Camera matrix $\mathbf{K}$ and camera pose $\mathbf{R}, \mathbf{t}$ in every image
for *each image of the chessboard* **do**
  Find the grid pattern and store the pixel coordinate of every points in the grid ;
  Assign the 3D coordinate of these points using the grid size and the world frame defined as in Figure.1 ;
  Find the homography $\mathbf{H}$ from the chessboard to its image ;
  Construct Equation.(16) using the homography $\mathbf{H}$;
end
Stack Equation.(16) get from each image to form matrix $\mathbf{V}$ in Equation.(17) ;
Solve Equation.(17) for $\mathbf{b}$;
Using $\mathbf{b}$ and Equation.(9), find $\mathbf{K}$;
for *each image* **do**
  Find $\mathbf{R}, \mathbf{t}$ using Equation.(18) ;
end

As can be seen from the Algorithm.1, the majority of computation is in the first for-loop which involves each image individually. For this reason, the steps of this for-loop are implemented as 4 methods of the class `Image` in the file `image.py`.

2.1 class `Image`

This class read an image of the chessboard from a file and performs 4 steps of the first for-loop of Algorithm.1 by sequentially invoking the following methods

- `locate_landmark`( ): identify corners on the chessboard such that they form a grid defined by the field `rows` and `cols`
- `get_landmark_world_coordinate`( ): compute 3D coordinate for each chessboard's corner given the square size and the world coordinate defined as in Figure.1
- `find_homography`( ): find the homography $\mathbf{H}$ using Equation.(8)
- `construct_v`( ): compute the left-hand side of Equation.(16)

In addition, method `find_extrinsic` find camera pose $(\mathbf{R}, \mathbf{t})$ using Equation.(18). While `locate_landmark` is already implemented, the other methods need to be completed by you.

2.1.1 Method `locate_landmark`

This method returns a matrix of size 48×2. Each row of this matrix represent the pixel coordinate of a corner on the chessboard. The indexing of rows of this matrix is shown by the numbers on Figure.1. Attention need to be paid to this indexing when computing the 3D coordinate of these corners.

2.1.2 Method `get_landmark_world_coordinate`

This method returns a matrix of size 48×2. Each row of this matrix represent the X-Y coordinate of a corner on the chessboard in the world frame. It's worth to notice that the chessboard is assumed to be on the plane $Z = 0$, thus the omission of the third coordinate in the output of this method.

The definition of the world coordinate is as following and can be visualized in Figure.1

- The origin is placed at the first corner (indexed by 0)
- The x-axis is from corner 0 to corner 1
- The y-axis is from corner 0 to corner 8
- The z-axis points inward the chessboard

The size of a square on the chessboard is $0.03m$ and is assigned to the field `square_size`.

It is important to make sure the index of the output of this method is in accordance with the index of the output of method `locate_landmark` (i.e. row i-th of each of these two matrices refers to the same point).

2.1.3 Method `find_homography`

Because matrix $\mathbf{Q}$ in Equation.(8) contains both world coordinate and pixel coordinate, it is poorly conditioned which can reduce the accuracy of the solution to $\mathbf{H}$. To remedy this, the pixel coordinates $\mathbf{p}_i$ and the world coordinates $\mathbf{P}_i$ is normalized such that

- Their center is at the origin
- The average distance to the origin is $\sqrt{2}$

This normalization is carried out by pre-multiply $\widetilde{\mathbf{p}}_i$ and $\widetilde{\mathbf{P}}_i$ with $\mathbf{T}_p$ and $\mathbf{T}_w$ respectively.

$$\begin{aligned} \widehat{\mathbf{p}}_i &= \mathbf{T}_p \, \widetilde{\mathbf{p}}_i \\ \widehat{\mathbf{P}}_i &= \mathbf{T}_w \, \widetilde{\mathbf{P}}_i \end{aligned} \tag{19}$$

Then, the normalized points $\widehat{\mathbf{p}}_i$ and $\widehat{\mathbf{P}}_i$ are used to compute the normalized $\widehat{\mathbf{H}}$ using Equation.(8). The true homography $\mathbf{H}$ is recovered by de-normalize $\widehat{\mathbf{H}}$

$$\mathbf{H} = \mathbf{T}_p^{-1} \, \widehat{\mathbf{H}} \, \mathbf{T}_w \tag{20}$$

The normalize transformation $\mathbf{T}_p$ is calculated as

$$\mathbf{T}_p = \begin{bmatrix} s & 0 & -s\,\bar{u} \\ 0 & s & -s\,\bar{v} \\ 0 & 0 & 1 \end{bmatrix} \tag{21}$$

here, $[\bar{u}, \bar{v}]$ is the center of $\{\mathbf{p}_i\}_{i=1}^n$ and s is the ratio between $\sqrt{2}$ and the mean distance of $\mathbf{p}_i$ to the origin. $\mathbf{T}_w$ can be calculated from $\{\mathbf{P}_i\}_{i=1}^n$ similarly. These two matrices are computed by the function `normalize_trans` in the file `image.py` which need to be implemented.

2.2 `main.py`

This file instantiate each instance of class `Image` for each image of the chessboard to get the constraint on the camera intrinsic matrix (Equation.(16)) and then implements the rest of Algorithm.1.

2.3 Expected work

The skeleton code contains `image.py` and `main.py` is provided. The methods and functions need to be implemented is marked with **TODO**.

References

[1] Z. Zhang, "Flexible camera calibration by viewing a plane from unknown orientations," in *Proceedings of the seventh ieee international conference on computer vision*, vol. 1. Ieee, 1999, pp. 666–673.