Chessboard-based camera calibration

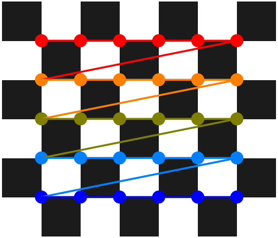
- Z. Zhang, Microsoft Research, 1998.
- Easy and accurate method
- Frequently-used
- Non-perspective distorsion can be handled
- Chessboard can be easily printed
- Efficient implementations available, e.g. in OpenCV
- See demos on Youtube
- Disadvantages
  - Multiple images required
  - Avoid checked patterns on shirts :(

Box-based camera calibration

- R.Y. Tsai, 1986.
- Non-perspective distortion can be handled
- Less user-friendly than chessboard-based one
  $\rightarrow$ not frequently used
- Hard to manufacture a precize calibration box
  - especially in large dimensions

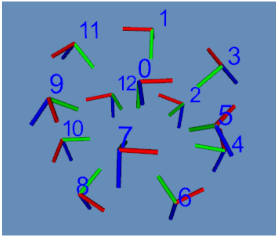
  $\rightarrow$ it is difficult to calibrate a camera using a small cube
- Benefits
  - One static image is satisfactory

Calibration of a camera-system using chessboards



Detected corners

Chessboard



Extrinsic camera params

- Chessboard patterns are assymetric $\longrightarrow$ corner detection unambiguous.
- Orientations of chessboard in images should differ.
- Extrinsic parameters can also be retrieved.

Chessboard-based calibration

- Main steps of calibration
  1. Homography exists between the calibration plane and an image
  2. Camera intrinsics in matrix K can be computed from homographies

- World coordinate is fixed to the board
  - Axis Z is perpendicular to the board $\longrightarrow Z = 0$ is the board plane

$$\alpha \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{K} \begin{bmatrix} \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{t} \end{bmatrix} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = \mathbf{H} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}, \quad \text{where} \tag{31}$$

  - $\mathbf{r}_1$ and $\mathbf{r}_2$ are the first two rows of rotation $\mathbf{R}$
  - $\mathbf{H} \doteq \mathbf{K} \begin{bmatrix} \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{t} \end{bmatrix}$

- Task is to (1) estimate **H**, then (2) computate intrinsic matrix **K**

Chessboard-based calibration

- Corners of chessboard fields can be easily detected.
  - chessboard -> image correspondences $\mathbf{x}_i \rightarrow \mathbf{u}_i$ used
  - at least 4 required
  - $\rightarrow$ More correspondences needed for contaminated data
  - subpixel corner detection $\rightarrow$ improved accuracy
  - $\rightarrow$ intersections of lines
- Estimation of homography $\mathbf{H}$
  - linear estimation minimizing algebraic error
  - non-linear estimation considering geometric error
- Homography $\mathbf{H}$ can be estimated up to an unknown scale
- Let $\mathbf{h}_1, \mathbf{h}_2, \mathbf{h}_3$ denote the columns of $\mathbf{H}$: $\mathbf{H} \doteq \begin{bmatrix} \mathbf{h}_1 & \mathbf{h}_2 & \mathbf{h}_3 \end{bmatrix}$

Computation of intrinsic parameters 1/3

- For homography matrix, the following equations are valid:

$$\begin{bmatrix}\mathbf{h}_1 & \mathbf{h}_2 & \mathbf{h}_3\end{bmatrix} \sim \mathbf{K}\begin{bmatrix}\mathbf{r}_1 & \mathbf{r}_2 & \mathbf{t}\end{bmatrix}$$
$$\begin{bmatrix}\mathbf{h}_1 & \mathbf{h}_2\end{bmatrix} \sim \mathbf{K}\begin{bmatrix}\mathbf{r}_1 & \mathbf{r}_2\end{bmatrix}$$
$$\mathbf{K}^{-1}\begin{bmatrix}\mathbf{h}_1 & \mathbf{h}_2\end{bmatrix} \sim \begin{bmatrix}\mathbf{r}_1 & \mathbf{r}_2\end{bmatrix}$$

- $\mathbf{r}_1$ and $\mathbf{r}_2$ are orthonormal, therefore

$$\mathbf{r}_1^T\mathbf{r}_2 = \mathbf{h}_1^T\mathbf{S}\mathbf{h}_2 = 0, \tag{32}$$
$$\|\mathbf{r}_1\|^2 - \|\mathbf{r}_2\|^2 = \mathbf{h}_1^T\mathbf{S}\mathbf{h}_1 - \mathbf{h}_2^T\mathbf{S}\mathbf{h}_2 = 0, \tag{33}$$

  - where $\mathbf{S} \doteq \mathbf{K}^{-T}\mathbf{K}^{-1}$, $\mathbf{K}^{-T} \doteq (\mathbf{K}^{-1})^T$
- This is a linear problem w.r.t. the elements of **S**. $\rightarrow$ They can be optimally estimated.

Computation of intrinsic parameters 2/3

- Elements of calibration matrix **K**:

$$\mathbf{K} = \begin{bmatrix} fk_u & s & u_0 \\ 0 & fk_v & v_0 \\ 0 & 0 & 1 \end{bmatrix}$$

- $\mathbf{S} = \lambda \mathbf{K}^{-T}\mathbf{K}^{-1}$

$$\frac{\mathbf{S}}{\lambda} = \begin{bmatrix} \frac{1}{(fk_u)^2} & -\frac{s}{(fk_u)^2 fk_v} & \frac{u_0 fk_v - v_0 s}{(fk_u)^2 fk_v} \\ -\frac{s}{(fk_u)^2 fk_v} & \frac{s^2}{(fk_u)^2 (fk_v)^2} + \frac{1}{(fk_v)^2} & \frac{-s(u_0 fk_v - v_0 s)}{(fk_u)^2 (fk_v)^2} + \frac{v_0}{(fk_v)^2} \\ \frac{u_0 fk_v - v_0 s}{(fk_u)^2 fk_v} & \frac{-s(u_0 fk_v - v_0 s)}{(fk_u)^2 (fk_v)^2} + \frac{v_0}{(fk_v)^2} & 1 + \frac{v_0^2}{fk_v^2} + \frac{(u_0 fk_v - v_0 s)^2}{(fk_u)^2 (fk_v)^2} \end{bmatrix}$$

- Matrix **S** has 5 parameters to be estimated: fk_u, fk_v, u_0, v_0, s
- Each chessboard image yields 2 equations $\longrightarrow$ at least 3 images required
- More images $\longrightarrow$ overdetermined system
- Robustification also requires many images

Computation of intrinsic parameters 3/3

- Matrix **S** $\longrightarrow$ closed-form solution for intrinsic parameters(**K**)

$$
\begin{aligned}
v_0 &= \frac{(S_{11}S_{23}-S_{21}S_{13})}{S_{11}S_{22}-S_{12}^2} \\
\lambda &= S_{33} - \frac{S_{13}^2+v_0(S_{12}S_{13}-S_{11}S_{23})}{S_{11}} \\
fk_u &= \sqrt{\frac{\lambda}{S_{11}}} \\
fk_v &= \sqrt{\lambda S_{11}/(S_{11}S_{22}-S_{12}^2)} \\
s &= -S_{12}fk_u^2 fk_v/\lambda \\
u_0 &= sv_0/fk_v - S_{13}{fk_v}^2/\lambda
\end{aligned}
$$

- Check: homework...

Computation of extrinsic parameters

$$\begin{bmatrix} \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{t} \end{bmatrix} = \mathbf{K}^{-1} H \tag{34}$$

$$\mathbf{r}_3 = \mathbf{r}_1 \times \mathbf{r}_2 \tag{35}$$

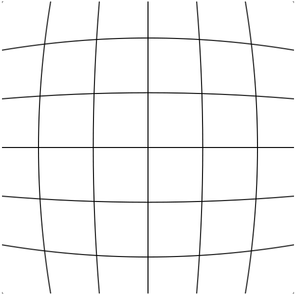
- For detailed description: Z. Zhang, Technical Report
- Implementation available in OpenCV, C++
- Matlab toolbox also exists
  - `http://sourceforge.net/projects/opencvlibrary/`

Outline

Radial distortion

- Non-perspective distorton is common for cheap or wide FoV (field of view) lenses
- Perspective model is only an approximation for real projection
  - e.g. projection of a line should be a straight line due to perspectivity
  - $\rightarrow$ This is not always true for real cameras
- Usually, type of distortion is **radial distortion**
  - Two subtypes: **Barrel/pillow** distortion
  - $\rightarrow$ barellel is more frequent
- Radial distortion has to be undistorted
  - Especially, when accurate 3D reconstruction should be achieved.
- Undistortion is usual part of camera calibration
  - It is included e.g. in OpenCV's calibration, in final numerical optimization

Radial distortion



barrell

pillow

- Straight lines become curves
- It is usual for wide FoV (small focal length)

Source of images: Wikipedia

Correction of radial distortion

$$\hat{\mathbf{u}} = \mathbf{u}_c + L(r)(\mathbf{u} - \mathbf{u}_c), \quad \text{ahol} \tag{36}$$

- $\mathbf{u}$: measured, $\hat{\mathbf{u}}$: corrected coordinates
- $\mathbf{u}_c$ center of distortion
  - it is usually assumed that $\mathbf{u}_c$ coincides with principal point $\mathbf{u}_0$.
- $L(r)$ is a cubic polynomial in r^2

$$L(r) = 1 + \kappa_1 r^2 + \kappa_2 r^4 + \kappa_3 r^6$$

  - $r = \|\mathbf{u} - \mathbf{u}_c\|$ is the distance from $\mathbf{u}_c$.
  - $L(r)$ is a Taylor approximation of the real distortion function

  $\rightarrow$ $\kappa_1, \kappa_2, \kappa_3$ are small real numbers
- Model is built in the non-linear homography estimation

  $\rightarrow$ Parameters $\kappa_1, \kappa_2, \kappa_3$ are stimated based on 2D geometric error

OpenCV: tangential distortion

$$\hat{x} = y_c + L_1(x, y)(x - x_c)$$
$$\hat{y} = y_c + L_2(x, y)(y - y_c)$$

- $L_{\{1,2\}}(x, y)$ are products of polynomials

$$L_1(x, y) = 1 + 2p_1 xy + p_2 \left(r^2 + 2x^2\right)$$
$$L_2(x, y) = 1 + 2p_2 xy + p_1 \left(r^2 + 2y^2\right)$$

  - $r = x^2 + y^2$ is the distance from the optical axis
  - p_1, p_2 are small real numbers
  - It is not mandatory to use all tangential parameters.
  - Tangential distortion does complete and not substitutes radial distortion.