Camera calibration

Morten R. Hannemose, mohan@dtu.dk

February 23, 2024

02504 Computer vision course lectures,
DTU Compute, Kgs. Lyngby 2800, Denmark

This lecture is being livestreamed and recorded

(hopefully)

Two feedback persons

Learning objectives

After this lecture you should be able to:

- implement and use the direct linear transformation to calibrate a pinhole camera
- implement and use Zhang's method (2000) to calibrate a pinhole camera using checkerboards

Presentation topics

- Direct linear transformation
- Zhang's method (2000)
    - Reprojection error
- Non-linear calibration
- Practical remarks

Culmination of previous weeks

- Pinhole camera model
- Homogeneous coordinates
- Homographies
- Linear algorithms
- Calibration

Direct linear transformation

Direct linear transformation

Start from the projection equation

$$\boldsymbol{q}_i = \boldsymbol{\mathcal{P}} \boldsymbol{Q}_i$$

$$\begin{bmatrix} sx_i \\ sy_i \\ s \end{bmatrix} = \boldsymbol{\mathcal{P}} \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix}$$

then rearrange into the form $\boldsymbol{B}^{(i)} \text{flatten}(\boldsymbol{\mathcal{P}}^{\top}) = \boldsymbol{0}$.

I use the form explained in LN: 2.7 Camera Resection.

Direct linear transformation i

$$\begin{aligned} \boldsymbol{q}_i &= \boldsymbol{\mathcal{P}} \boldsymbol{Q}_i, \\ \mathbf{0} &= \boldsymbol{q}_i \times \boldsymbol{\mathcal{P}} \boldsymbol{Q}_i, \\ &= [\boldsymbol{q}_i]_\times \boldsymbol{\mathcal{P}} \boldsymbol{Q}_i, \\ &= \boldsymbol{B}^{(i)} \mathsf{flatten}(\boldsymbol{\mathcal{P}}^\mathsf{T}), \end{aligned}$$

where (continue to next slide) ...

Direct linear transformation ii

$$
\begin{aligned}
\mathbf{0} &= \boldsymbol{B}^{(i)}\mathsf{flatten}(\boldsymbol{\mathcal{P}}^\mathsf{T}),\\
\boldsymbol{B}^{(i)} &= \begin{bmatrix} 0 & -X_i & X_i y_i & 0 & -Y_i & Y_i y_i & 0 & -Z_i & Z_i y_i & 0 & -1 & y_i \\ X_i & 0 & -X_i x_i & Y_i & 0 & -Y_i x_i & Z_i & 0 & -Z_i x_i & 1 & 0 & -x_i \\ -X_i y_i & X_i x_i & 0 & -Y_i y_i & Y_i x_i & 0 & -Z_i y_i & Z_i x_i & 0 & -y_i & x_i & 0 \end{bmatrix},\\
&= \boldsymbol{Q}_i \otimes [\boldsymbol{q}_i/s]_\times\\
\mathsf{flatten}(\boldsymbol{\mathcal{P}}^\mathsf{T}) &= \begin{bmatrix} \mathscr{P}_{11} & \mathscr{P}_{21} & \mathscr{P}_{31} & \mathscr{P}_{12} & \mathscr{P}_{22} & \mathscr{P}_{32} & \mathscr{P}_{13} & \mathscr{P}_{23} & \mathscr{P}_{33} & \mathscr{P}_{14} & \mathscr{P}_{24} & \mathscr{P}_{34} \end{bmatrix}^\mathsf{T}
\end{aligned}
$$

Direct linear transformation

Now let

$$\mathbf{0} = \boldsymbol{B}\ \mathsf{flatten}(\boldsymbol{\mathcal{P}}^\top),$$

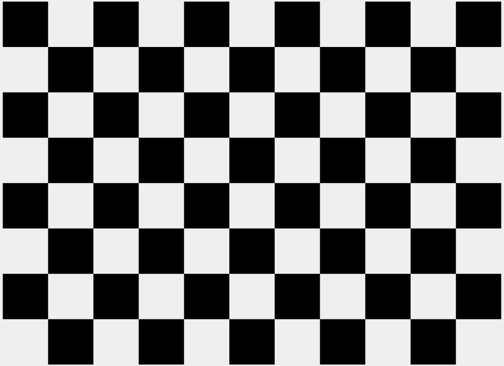
where

$$\boldsymbol{B} = \begin{bmatrix} \boldsymbol{B}^{(1)} \\ \boldsymbol{B}^{(2)} \\ \vdots \end{bmatrix},$$

and solve using SVD on $\boldsymbol{B}$.

Zhang's method (2000)

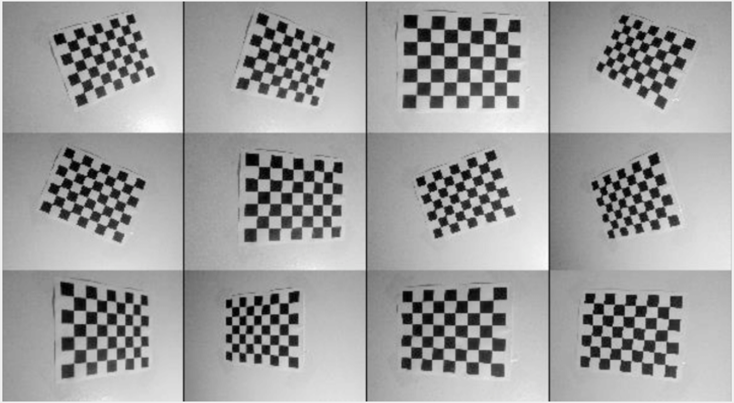
Using a checkerboard



www.calib.io | 8x11 | Checker Size: 15 mm.

Using a checkerboard

View the checkerboard in different poses:



Using a checkerboard

Problem:

Each view i has a different rotation $\boldsymbol{R}_i$ and translation $\boldsymbol{t}_i$

How do we find all $\boldsymbol{\mathcal{P}}_i$?

Zhang's method

First, assume all checkerboard corners are in the $Z = 0$ plane:

$$\boldsymbol{Q}_j = \begin{bmatrix} X_j \\ Y_j \\ 0 \\ 1 \end{bmatrix}$$

Simplifying the projection equation

Let $\boldsymbol{r}_i^{(c)}$ is the c^{th} column of $\boldsymbol{R}_i$. Now projection is

$$\boldsymbol{q}_{ij} = \mathcal{P}_i \boldsymbol{Q}_j = \boldsymbol{K} \begin{bmatrix} \boldsymbol{r}_i^{(1)} & \boldsymbol{r}_i^{(2)} & \boldsymbol{r}_i^{(3)} & \boldsymbol{t}_i \end{bmatrix} \underbrace{\begin{bmatrix} X_j \\ Y_j \\ 0 \\ 1 \end{bmatrix}}_{\boldsymbol{Q}}$$

Simplifying the projection equation

Let $\boldsymbol{r}_i^{(c)}$ is the c^{th} column of $\boldsymbol{R}_i$. Now projection is

$$\boldsymbol{q}_{ij} = \mathcal{P}_i \boldsymbol{Q}_j = \boldsymbol{K} \begin{bmatrix} \boldsymbol{r}_i^{(1)} & \boldsymbol{r}_i^{(2)} & \boldsymbol{r}_i^{(3)} & \boldsymbol{t}_i \end{bmatrix} \underbrace{\begin{bmatrix} X_j \\ Y_j \\ 0 \\ 1 \end{bmatrix}}_{\boldsymbol{Q}}$$

$$= \boldsymbol{K} \begin{bmatrix} \boldsymbol{r}_i^{(1)} & \boldsymbol{r}_i^{(2)} & \boldsymbol{t}_i \end{bmatrix} \underbrace{\begin{bmatrix} X_j \\ Y_j \\ 1 \end{bmatrix}}_{\tilde{\boldsymbol{Q}}_j}.$$

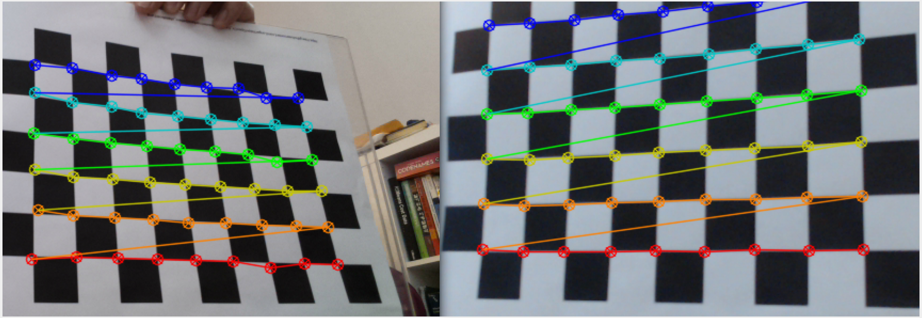
From projections to homographies

$$
\begin{aligned}
\boldsymbol{q}_{ij} &= \underbrace{\boldsymbol{K}\begin{bmatrix}\boldsymbol{r}_i^{(1)} & \boldsymbol{r}_i^{(2)} & \boldsymbol{t}_i\end{bmatrix}}_{\boldsymbol{H}_i}\underbrace{\begin{bmatrix}X_j \\ Y_j \\ 1\end{bmatrix}}_{\tilde{\boldsymbol{Q}}_j} \\
&= \boldsymbol{H}_i\tilde{\boldsymbol{Q}}_j
\end{aligned}
$$

The homographies $\boldsymbol{H}_i$ can be determined from the plane-plane correspondence $\tilde{\boldsymbol{Q}}_j$ to $\boldsymbol{q}_{ij}$ (week 2).

Corner correspondences

Need to find the unique positions of corners $\boldsymbol{q}_{ij}$.



Find all H_i from corners $\tilde{Q}_j$ and projections q_{ij} with $q_{ij} = H_i\tilde{Q}_j$

For example, using SVD.

From homographies to the camera matrix

$$\boldsymbol{H}_i = \begin{bmatrix} \boldsymbol{h}_i^{(1)} & \boldsymbol{h}_i^{(2)} & \boldsymbol{h}_i^{(3)} \end{bmatrix} = \lambda_i \boldsymbol{K} \begin{bmatrix} \boldsymbol{r}_i^{(1)} & \boldsymbol{r}_i^{(2)} & \boldsymbol{t}_i \end{bmatrix}.$$

$\boldsymbol{r}_i^{(1)}$ and $\boldsymbol{r}_i^{(2)}$ are orthonormal, i.e.

$$\begin{aligned} {\boldsymbol{r}_i^{(1)}}^{\mathsf{T}} \boldsymbol{r}_i^{(1)} &= {\boldsymbol{r}_i^{(2)}}^{\mathsf{T}} \boldsymbol{r}_i^{(2)} = 1, \\ {\boldsymbol{r}_i^{(1)}}^{\mathsf{T}} \boldsymbol{r}_i^{(2)} &= {\boldsymbol{r}_i^{(2)}}^{\mathsf{T}} \boldsymbol{r}_i^{(1)} = 0. \end{aligned}$$

From homographies to the camera matrix

Express $\boldsymbol{r}_i^{(\alpha)}$ using $\boldsymbol{h}_i^{(\alpha)}$:

$$\boldsymbol{h}_i^{(\alpha)} = \lambda_i \boldsymbol{K} \boldsymbol{r}_i^{(\alpha)}, \Leftrightarrow$$

$$\boldsymbol{K}^{-1} \boldsymbol{h}_i^{(\alpha)} = \lambda_i \boldsymbol{r}_i^{(\alpha)}.$$

From homographies to the camera matrix

Express $\boldsymbol{r}_i^{(\alpha)}$ using $\boldsymbol{h}_i^{(\alpha)}$:

$$\boldsymbol{h}_i^{(\alpha)} = \lambda_i \boldsymbol{K} \boldsymbol{r}_i^{(\alpha)}, \Leftrightarrow$$

$$\boldsymbol{K}^{-1} \boldsymbol{h}_i^{(\alpha)} = \lambda_i \boldsymbol{r}_i^{(\alpha)}.$$

Now the constraints from the previous slide are:

$$\boldsymbol{h}_i^{(1)\mathsf{T}} \boldsymbol{K}^{-\mathsf{T}} \boldsymbol{K}^{-1} \boldsymbol{h}_i^{(2)} = 0,$$

$$\boldsymbol{h}_i^{(1)\mathsf{T}} \boldsymbol{K}^{-\mathsf{T}} \boldsymbol{K}^{-1} \boldsymbol{h}_i^{(1)} = \boldsymbol{h}_i^{(2)\mathsf{T}} \boldsymbol{K}^{-\mathsf{T}} \boldsymbol{K}^{-1} \boldsymbol{h}_i^{(2)} = \lambda_i^2.$$

We have found constraints on the camera matrix! 🥳

Number of constraints

- Two constraints doesn't seem that impressive?

Number of constraints

- Two constraints doesn't seem that impressive?
- Homography has eight degrees of freedom
- Pose of checkerboard has six (3 rotation, 3 translation)
- A single homography can only fix two degrees of freedom of a camera matrix.

Define some new variables i

How to put into practice?

Define the matrix:

$$\boldsymbol{B} = \boldsymbol{K}^{-\mathsf{T}}\boldsymbol{K}^{-1} = \begin{bmatrix} B_{11} & B_{12} & B_{13} \\ B_{12} & B_{22} & B_{23} \\ B_{13} & B_{23} & B_{33} \end{bmatrix},$$

$$\boldsymbol{b} = \begin{bmatrix} B_{11} & B_{12} & B_{22} & B_{13} & B_{23} & B_{33} \end{bmatrix}^{\mathsf{T}},$$

Define some new variables ii

Now define $\boldsymbol{v}_i^{(\alpha\beta)}$ such that

$$\boldsymbol{v}_i^{(\alpha\beta)}\boldsymbol{b} = \boldsymbol{h}_i^{(\alpha)^T}\boldsymbol{B}\,\boldsymbol{h}_i^{(\beta)}.$$

Then $\boldsymbol{v}_i^{(\alpha\beta)}$ must be an 1×6 vector given by

$$\begin{aligned}\boldsymbol{v}_i^{(\alpha\beta)} = [&H_i^{(1\alpha)}H_i^{(1\beta)}, \quad H_i^{(1\alpha)}H_i^{(2\beta)} + H_i^{(2\alpha}H_i^{(1\beta)}, \quad H_i^{(2\alpha)}H_i^{(2\beta)}, \quad \ldots \\ &H_i^{(3\alpha)}H_i^{(1\beta)} + H_i^{(1\alpha)}H_i^{(3\beta)}, \quad H_i^{(3\alpha)}H_i^{(2\beta)} + H_i^{(2\alpha)}H_i^{(3\beta)}, \quad H_i^{(3\alpha)}H_i^{(3\beta)}],\end{aligned}$$

Define some new variables iii

where $H_i^{(rc)}$ is the element in row r and column c of $\boldsymbol{H}_i$.

From homographies to the camera matrix

Recall our constraints:

$$\boldsymbol{h}_i^{(1)^\mathsf{T}} \boldsymbol{K}^{-\mathsf{T}} \boldsymbol{K}^{-1} \boldsymbol{h}_i^{(1)} = \boldsymbol{h}_i^{(2)^\mathsf{T}} \boldsymbol{K}^{-T} \boldsymbol{K}^{-1} \boldsymbol{h}_i^{(2)} \Leftrightarrow$$

$$\boldsymbol{h}_i^{(1)^\mathsf{T}} \boldsymbol{B}\, \boldsymbol{h}_i^{(1)} - \boldsymbol{h}_i^{(2)^\mathsf{T}} \boldsymbol{B}\, \boldsymbol{h}_i^{(2)} = 0 =$$

$$\left(\boldsymbol{v}_i^{(11)} - \boldsymbol{v}_i^{(22)}\right) \boldsymbol{b} = 0.$$

Now we can express our constraints in matrix-form

$$\begin{bmatrix} \boldsymbol{v}_i^{(12)} \\ \boldsymbol{v}_i^{(11)} - \boldsymbol{v}_i^{(22)} \end{bmatrix} \boldsymbol{b} = \mathbf{0}$$

From homographies to the camera matrix

For all checkerboard poses

$$\boldsymbol{V}\boldsymbol{b} = \begin{bmatrix} \boldsymbol{v}_1^{(12)} \\ \boldsymbol{v}_1^{(11)} - \boldsymbol{v}_1^{(22)} \\ \boldsymbol{v}_2^{(12)} \\ \boldsymbol{v}_2^{(11)} - \boldsymbol{v}_2^{(22)} \\ \vdots \end{bmatrix} \boldsymbol{b} = \boldsymbol{0}$$

When $\boldsymbol{b}$ is found then $\boldsymbol{K}$ can be determined using the formulas in Zhang's paper.

Find the camera matrix K through b using $Vb = 0$ and SVD, where V is built from the homographies H_i.

Reprojection error

Is it a good camera calibration? How to find out?

Reprojection error

Is it a good camera calibration? How to find out?

Reproject the points from the checkerboards to the camera, and compare to where we've seen them

$$\tilde{\boldsymbol{q}}_{ij} = \boldsymbol{K} \begin{bmatrix} \boldsymbol{R}_i & \boldsymbol{t}_i \end{bmatrix} \boldsymbol{Q}_j$$

$$\Pi(\tilde{\boldsymbol{q}}_{ij}) - \Pi(\boldsymbol{q}_{ij})$$

Reprojection error

We can now compute the reprojection error as the root mean squared error (RMSE)

$$\sqrt{\frac{1}{n}\sum_{i}\sum_{j}\left\|\Pi(\tilde{\boldsymbol{q}}_{ij})-\Pi(\boldsymbol{q}_{ij})\right\|_2^2}$$

where n is the total number of points.

We have $\boldsymbol{K}$, but we are still missing something.

Reprojection error

We can now compute the reprojection error as the root mean squared error (RMSE)

$$\sqrt{\frac{1}{n}\sum_i \sum_j \left\|\Pi(\tilde{\boldsymbol{q}}_{ij}) - \Pi(\boldsymbol{q}_{ij})\right\|_2^2}$$

where n is the total number of points.

We have $\boldsymbol{K}$, but we are still missing something.

How do we recover $\boldsymbol{R}_i$ and $\boldsymbol{t}_i$?

From homographies to poses

Recall: $\boldsymbol{H}_i = \begin{bmatrix} \boldsymbol{h}_i^{(1)} & \boldsymbol{h}_i^{(2)} & \boldsymbol{h}_i^{(3)} \end{bmatrix} = \lambda_i \boldsymbol{K} \begin{bmatrix} \boldsymbol{r}_i^{(1)} & \boldsymbol{r}_i^{(2)} & \boldsymbol{t}_i \end{bmatrix}$

Now we can recover $\boldsymbol{R}_i$ and $\boldsymbol{t}_i$:

$$\begin{aligned} \boldsymbol{r}_i^{(1)} &= \tfrac{1}{\lambda_i} \boldsymbol{K}^{-1} \boldsymbol{h}_i^{(1)}, \\ \boldsymbol{r}_i^{(2)} &= \tfrac{1}{\lambda_i} \boldsymbol{K}^{-1} \boldsymbol{h}_i^{(2)}, \\ \boldsymbol{t}_i &= \tfrac{1}{\lambda_i} \boldsymbol{K}^{-1} \boldsymbol{h}_i^{(3)}, \\ \boldsymbol{r}_i^{(3)} &= \boldsymbol{r}_i^{(1)} \times \boldsymbol{r}_i^{(2)}, \end{aligned}$$

where $\lambda_i = \left\| \boldsymbol{K}^{-1} \boldsymbol{h}_i^{(1)} \right\|_2 = \left\| \boldsymbol{K}^{-1} \boldsymbol{h}_i^{(2)} \right\|_2$.

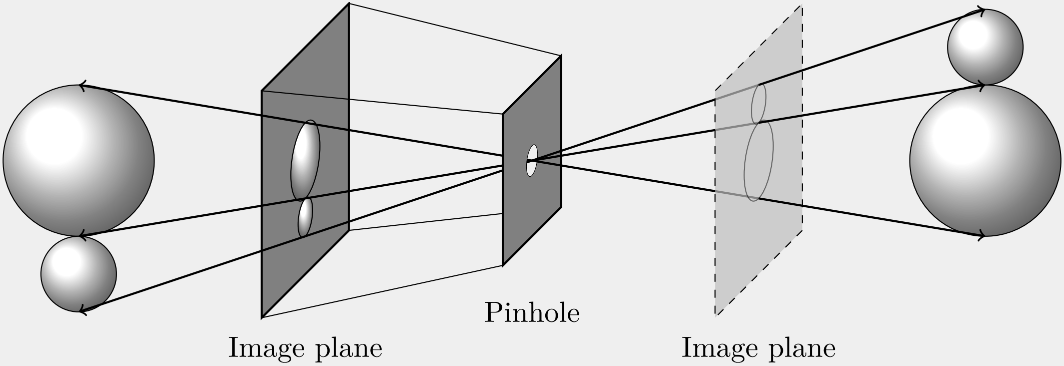
Homography and poses

$$\boldsymbol{t}_i = \frac{1}{\lambda_i}\boldsymbol{K}^{-1}\boldsymbol{h}_i^{(3)}$$

What happens if $t_{iz} < 0$? And what does that mean?

Can we ensure correctness?

Pinhole camera revisited



Points behind the camera also get projected to the image plane.
Multiply all 3D points by -1 and they still project to the same place.

Homography and poses

- If $t_{iz} < 0$ the checkerboard is behind the camera.
- How to get the correct solution?
- Estimate $\boldsymbol{R}_i$ and $\boldsymbol{t}_i$ again using $-\boldsymbol{H}_i$ (flipped sign).

Non-linear calibration

Least-squares method

With projection equation

$$\boldsymbol{q}_{ij} = \boldsymbol{K} \begin{bmatrix} \boldsymbol{R}_i & \boldsymbol{t}_i \end{bmatrix} \boldsymbol{Q}_j$$

then

$$E = \sum_{i,j} \left\| \mathrm{dist}\Big(\pi \left(\mathbf{K} \begin{bmatrix} \mathbf{R}_i & \mathrm{t}_i \end{bmatrix} \boldsymbol{Q}_j\right)\Big) - \pi(\boldsymbol{q}_{ij}) \right\|^2$$

Solution: minimize E w.r.t. $\mathbf{K}$, $\mathbf{R}_i$, t_i and lens distortion.

Practical remarks

What images should we take?

- Our two constraints per image are based on $\boldsymbol{r}_i^{(1)}$ and $\boldsymbol{r}_i^{(2)}$.
- Two parallel checkerboards express the same constraints.

What images should we take?

- Our two constraints per image are based on $\boldsymbol{r}_i^{(1)}$ and $\boldsymbol{r}_i^{(2)}$.
- Two parallel checkerboards express the same constraints.
  - at least when we don't consider lens distortion
- Make sure to rotate the checkerboards so they are not parallel.
- Make the checkerboards take up as as much of the frame as possible!

How many images?

- Without lens distortion: In theory at least three.
- In practice: It depends...
  - Some people use 2000+ images for a single calibration[1]
  - Look at the reprojection error
  - Both of the training set and the validation set

Learning objectives

After this lecture you should be able to:

- implement and use the direct linear transformation to calibrate a pinhole camera
- implement and use Zhang's method (2000) to calibrate a pinhole camera using checkerboards

Last week's exercise

The goal of exercise 3.6-3.10 last week was to:

- Understand how to handle when no camera has $\boldsymbol{R} = \boldsymbol{I}$ and $\boldsymbol{t} = \mathbf{0}$.
- Easily find epipolar lines in image 1 from points in image 2.

I have uploaded a new `TwoImageDataCar.npy`, where both have nontrivial $\boldsymbol{R}$ and $\boldsymbol{t}$.
For exercise 3.10, you can use the transpose.

$$
\begin{aligned}
\boldsymbol{q}_1^\top \boldsymbol{F} \boldsymbol{q}_2 &= 0 \\
\boldsymbol{q}_2^\top \boldsymbol{F}^\top \boldsymbol{q}_1 &= 0
\end{aligned}
$$

If you have already solved the exercise, loading the new file should not take long

Quiz & exercise time!