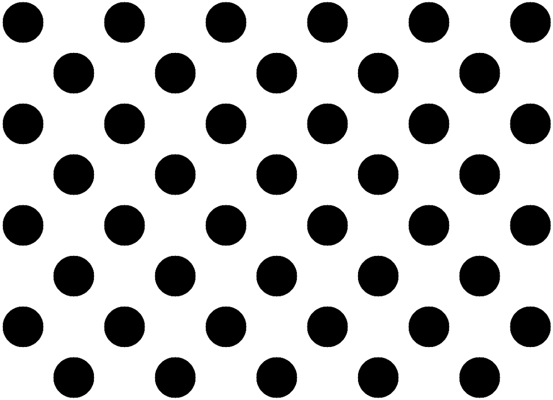
Basics for Enhanced Visualization: 3D/Data

Camera calibration



Rodrigo Cabral

Polytech Nice - Data Science

cabral@unice.fr

Source: circles pattern from OpenCV.

Outline

1. Introduction — This class
2. Model and parameters
3. Calibration with a rig
    - General problem
    - Estimation of the camera matrix
    - Camera parameters from camera matrix
4. Conclusions

---

5. Calibration with planes — Next class
    - Setting the calibration problem
    - Estimation of homographies
    - Camera parameters from homographies
6. Conclusions

Introduction

How to project a point $\begin{bmatrix} x_\mathrm{w} \\ y_\mathrm{w} \\ z_\mathrm{w} \end{bmatrix}$ in the image buffer?

Introduction

How to project a point $\begin{bmatrix} x_{\text{w}} \\ y_{\text{w}} \\ z_{\text{w}} \end{bmatrix}$ in the image buffer?

- If you know the **camera matrix M** you do

$$\begin{bmatrix} x'_{\text{im}} \\ y'_{\text{im}} \\ w'_{\text{im}} \end{bmatrix} = \mathbf{M} \begin{bmatrix} x_{\text{w}} \\ y_{\text{w}} \\ z_{\text{w}} \\ 1 \end{bmatrix} \Longrightarrow \begin{bmatrix} x_{\text{im}} \\ y_{\text{im}} \end{bmatrix} = \begin{bmatrix} \frac{x'_{\text{im}}}{w'_{\text{im}}} \\ \frac{y'_{\text{im}}}{w'_{\text{im}}} \end{bmatrix}$$

where $\mathbf{M} = \mathbf{K}_s \, \mathbf{K}_f \, \boldsymbol{\Pi}_0 \, \mathbf{K}_{\text{wc}}$.

- But in general we do not know neither **M**, nor any of its parameters.

Camera calibration

- Retrieving **M** and/or its internal parameters is called **camera calibration**.
- When **M** and its internal parameters are known we say that the camera is **calibrated**.
- This is in general an estimation problem. We have to estimate the camera parameters from data.

Model and parameters

Overall model: the camera matrix

$$\mathbf{M} = \mathbf{K}_s \, \mathbf{K}_f \, \mathbf{\Pi}_0 \, \mathbf{K}_{wc}$$

- The matrix **M** contains all the information from the camera.

Model and parameters

Overall model: the camera matrix

$$\mathbf{M} = \mathbf{K}_s \, \mathbf{K}_f \, \mathbf{\Pi}_0 \, \mathbf{K}_{wc}$$

- The matrix $\mathbf{M}$ contains all the information from the camera.
- This **linear transformation** relates a 3D point in world space to a 2D point in the image buffer. **But both are in homogeneous coordinates!**
- To get the true relation we need to transform in Cartesian coordinates. **This is a non linear transformation!**
- If we know $\mathbf{M}$ we can retrieve its internal parameters: parameters within $\mathbf{K}_s$, $\mathbf{K}_f$ $\mathbf{\Pi}_0$, $\mathbf{K}_{wc}$.

Model and parameters

Overall model: the camera matrix

$$\mathbf{M} = \mathbf{K}_s \, \mathbf{K}_f \, \mathbf{\Pi}_0 \, \mathbf{K}_{\mathrm{wc}}$$

- Depending on the application we can
  - Estimate directly its internal parameters. **M** is a result.
    Hard but robust to noise.
  - Estimate **M** only, generally without constraining its structure.
    Easy but not robust to noise.
  - Estimate **M** and then the internal parameters.
    Medium.

Model and parameters

Overall model: the camera matrix

$$\mathbf{M} = \mathbf{K}_s \, \mathbf{K}_f \, \mathbf{\Pi}_0 \, \mathbf{K}_{wc}$$

- Depending on the application we can
  - Estimate directly its internal parameters. **M** is a result.
    Hard but robust to noise.
  - Estimate **M** only, generally without constraining its structure.
    Easy but not robust to noise.
  - Estimate **M** and then the internal parameters.
    Medium.
- We can also know in advance some of its internal parameters
  $\Longrightarrow$ the three options above need to be adapted to this case.
- To initialize the **Hard** approach we use the **Medium** approach.
- To simplify the **Medium** approach we use the **Easy** approach first to estimate **M**.

Model and parameters

Intrinsic and extrinsic matrices

$$\mathbf{M} = \mathbf{M}_{\text{int}}\ \mathbf{M}_{\text{ext}}$$

- $\mathbf{M}_{\text{int}}$ is the **intrinsic matrix.** It contains the intrinsic parameters of the camera:

$$\mathbf{M}_{\text{int}} = \mathbf{K}_s\ \mathbf{K}_f = \begin{bmatrix} s_x f & s_\theta f & o_x \\ 0 & s_y f & o_y \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} f_x & f_\theta & o_x \\ 0 & f_y & o_y \\ 0 & 0 & 1 \end{bmatrix}$$

Model and parameters

Intrinsic and extrinsic matrices

$$\mathbf{M} = \mathbf{M}_{\text{int}}\ \mathbf{M}_{\text{ext}}$$

- $\mathbf{M}_{\text{int}}$ is the **intrinsic matrix**. It contains the intrinsic parameters of the camera:

$$\mathbf{M}_{\text{int}} = \mathbf{K}_s\ \mathbf{K}_f = \begin{bmatrix} s_x f & s_\theta f & o_x \\ 0 & s_y f & o_y \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} f_x & f_\theta & o_x \\ 0 & f_y & o_y \\ 0 & 0 & 1 \end{bmatrix}$$

- This matrix has 6 internal parameters. But we can retrieve uniquely only 5 (products *sf* have a scaling ambiguity).
- Sometimes these parameters are known if you have the camera specifications from the manufacturer.
- Note that this is an upper triangular matrix.
- In general we have $f_\theta = 0$ and for the assumption on the axis of the image buffer $f_y < 0$.

Model and parameters

Intrinsic and extrinsic matrices

$$\mathbf{M} = \mathbf{M}_{\text{int}}\,\mathbf{M}_{\text{ext}}$$

- If we assume $f_\theta = 0$ and that the center of the image buffer coincides with the center of the image plane, a fast and simple method to obtain the parameters in $\mathbf{M}_{\text{int}}$ is the following:

Intrinsic and extrinsic matrices

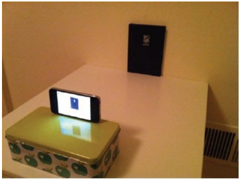
$$\mathbf{M} = \mathbf{M}_{\text{int}} \, \mathbf{M}_{\text{ext}}$$

- If we assume $f_\theta = 0$ and that the center of the image buffer coincides with the center of the image plane, a fast and simple method to obtain the parameters in $\mathbf{M}_{\text{int}}$ is the following:

  1. Retrieve the camera resolution in pixels p_x and p_y. This is normally available to you.
  2. Choose a planar rectangular object with known height H and width W, get an image from it by putting it at a distance d from the camera and in parallel with the image plane.
  3. Measure its height and width in pixels H_{im} and W_{im}.

Model and parameters

Simple intrinsic matrix calibration method



- Then we get

$$o_x = \frac{p_x}{2} \quad o_y = \frac{p_y}{2} \quad f_x = \frac{W_{\text{im}} d}{W} \quad f_y = -\frac{H_{\text{im}} d}{H}$$

- Note that if you have estimated $\mathbf{M}_{\text{int}}$ and $\mathbf{M}$, you can get

$$\mathbf{M}_{\text{ext}} = \mathbf{M}_{\text{int}}{}^{-1}\,\mathbf{M}$$

since $\mathbf{M}_{\text{int}}$ is always invertible (why?).

Source: Programming Computer Vision with Python, Jan Erik Solem, 2012.

Model and parameters

Intrinsic and extrinsic matrices

$$\mathbf{M} = \mathbf{M}_{\text{int}}\ \mathbf{M}_{\text{ext}}$$

- $\mathbf{M}_{\text{ext}}$ is the **extrinsic matrix.** It contains the extrinsic parameters of the camera:

$$\mathbf{M}_{\text{ext}} = \mathbf{\Pi}_0\ \mathbf{K}_{\text{wc}} = \left[\begin{array}{c|c} \mathbf{R}_{\text{wc}} & -\mathbf{R}_{\text{wc}}\mathbf{t}_{\text{cw}} \end{array}\right] = \left[\begin{array}{c|c} \mathbf{R} & \mathbf{t} \end{array}\right]$$

Model and parameters

Intrinsic and extrinsic matrices

$$\mathbf{M} = \mathbf{M}_{\text{int}}\ \mathbf{M}_{\text{ext}}$$

- $\mathbf{M}_{\text{ext}}$ is the **extrinsic matrix.** It contains the extrinsic parameters of the camera:

$$\mathbf{M}_{\text{ext}} = \mathbf{\Pi}_0\ \mathbf{K}_{wc} = \left[\begin{array}{c|c} \mathbf{R}_{wc} & -\mathbf{R}_{wc}\mathbf{t}_{cw} \end{array}\right] = \left[\begin{array}{c|c} \mathbf{R} & \mathbf{t} \end{array}\right]$$

- Matrix $\mathbf{R}$ is an orthogonal rotation matrix:

$$\mathbf{R} = \begin{bmatrix} \mathbf{r}_1^\top \\ \mathbf{r}_2^\top \\ \mathbf{r}_3^\top \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$$

- Note that we have only 3 free parameters (why? Hint: $\mathbf{r}_1^\top \mathbf{r}_2 = 0$ and $\mathbf{r}_3 = \pm(\mathbf{r}_1 \times \mathbf{r}_2)$).
- $\mathbf{t} = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}$ is a translation vector.

Model and parameters

Intrinsic and extrinsic matrices

$$\mathbf{M} = \mathbf{M}_{\text{int}} \, \mathbf{M}_{\text{ext}}$$

- Camera model is commonly given in the following form

$$\mathbf{M} = \mathbf{M}_{\text{int}} \left[\; \mathbf{R} \quad \middle| \quad \mathbf{t} \; \right]$$

Model and parameters

Intrinsic and extrinsic matrices

$$\mathbf{M} = \mathbf{M}_{\text{int}}\ \mathbf{M}_{\text{ext}}$$

- Camera model is commonly given in the following form

$$\mathbf{M} = \mathbf{M}_{\text{int}} \left[\ \mathbf{R}\ \middle|\ \mathbf{t}\ \right]$$

- Note that **M** has its left 3 columns which factorize in an upper triangular matrix and an orthogonal matrix.
- Also remember that in homogeneous coordinates all matrices of the form

$$\mathbf{M} = \alpha \begin{bmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \end{bmatrix}$$

for $\alpha \neq 0$ are equivalent.

General camera calibration problem

Given a set of **N point correspondences** $(\mathbf{u}^1_{\mathbf{im}}, \mathbf{u}^1_{\mathbf{w}}), \cdots, (\mathbf{u}^N_{\mathbf{im}}, \mathbf{u}^N_{\mathbf{w}})$

where $\mathbf{u}^i_{\text{im}}$ is the image buffer coordinates of the 3D point $\mathbf{u}^i_{\text{w}}$ in Cartesian coordinates,

estimate the camera model M, and its parameters:

M_{int}	f_x, f_y, f_θ, o_x, o_y	
M_{ext}	$\mathbf{R}$, $\mathbf{t}$	$\mathbf{R}_{wc}$, $\mathbf{t}_{cw}$

Calibration with a rig

General camera calibration problem

Given a set of **N point correspondences** $(\mathbf{u}^1_{\mathbf{im}}, \mathbf{u}^1_{\mathbf{w}}), \cdots, (\mathbf{u}^N_{\mathbf{im}}, \mathbf{u}^N_{\mathbf{w}})$

where $\mathbf{u}^i_{\mathrm{im}}$ is the image buffer coordinates of the 3D point $\mathbf{u}^i_{\mathrm{w}}$ in Cartesian coordinates,

estimate the camera model M, and its parameters:

M_{int}	f_x, f_y, f_θ, o_x, o_y	
M_{ext}	$\mathbf{R}$, $\mathbf{t}$	$\mathbf{R}_{wc}$, $\mathbf{t}_{cw}$

- 3D coordinate points are normally explicitly given.
- Image points for given 3D points are either obtained by hand (mouse-clicking on the corresponding image points) or automatically from known features: corners, edges, lines, circles, ellipses, *etc*.

Calibration with a rig

General camera calibration problem

Given a set of **N point correspondences** $(\mathbf{u}_{\mathbf{im}}^1, \mathbf{u}_{\mathbf{w}}^1), \cdots, (\mathbf{u}_{\mathbf{im}}^N, \mathbf{u}_{\mathbf{w}}^N)$

where $\mathbf{u}_{\mathrm{im}}^i$ is the image buffer coordinates of the 3D point $\mathbf{u}_{\mathrm{w}}^i$ in Cartesian coordinates,

estimate the camera model M, and its parameters:

M_{int}	$f_x, f_y, f_\theta, o_x, o_y$	
M_{ext}	$\mathbf{R}, \mathbf{t}$	$\mathbf{R}_{wc}, \mathbf{t}_{cw}$

- There are 11 internal parameters.

General camera calibration problem

- $\mathbf{u}_{\mathrm{im}}^{i}$ are noisy:
  - Circle of confusion from imperfect focusing.
  - Image is noisy.
  - Algorithm/user localizing image points is not ideal.

General camera calibration problem

- $\mathbf{u}^i_{\mathrm{im}}$ are noisy:
  - Circle of confusion from imperfect focusing.
  - Image is noisy.
  - Algorithm/user localizing image points is not ideal.
- Consequences:
  - We cannot fit perfectly a camera model (with zero error).
  - We should take into account the perturbations: search for parameters minimizing image projection residuals ε^i_x and ε^i_y (to be defined later).
  - Minimization can be done in least squares sense.

General camera calibration problem

Least squares minimization problem

minimize $\sum_{i=1}^{N} \left(\varepsilon_x^i\right)^2 + \left(\varepsilon_y^i\right)^2$

with respect to all camera parameters

subject to $\mathbf{R}^{\mathsf{T}}\mathbf{R} = \mathbf{I}$

where

$$\varepsilon_x^i = x_{\text{im}}^i - \left[\frac{f_x(\mathbf{r}_1^{\mathsf{T}}\mathbf{u}_{\text{w}}^i + t_x) + f_\theta(\mathbf{r}_2^{\mathsf{T}}\mathbf{u}_{\text{w}}^i + t_y)}{\mathbf{r}_3^{\mathsf{T}}\mathbf{u}_{\text{w}}^i + t_z} + o_x \right]$$

$$\varepsilon_y^i = y_{\text{im}}^i - \left[\frac{f_y(\mathbf{r}_2^{\mathsf{T}}\mathbf{u}_{\text{w}}^i + t_y)}{\mathbf{r}_3^{\mathsf{T}}\mathbf{u}_{\text{w}}^i + t_z} + o_y \right]$$

Calibration with a rig

General camera calibration problem

Least squares minimization problem

minimize	$\sum_{i=1}^{N} \left(\varepsilon_x^i\right)^2 + \left(\varepsilon_y^i\right)^2$
with respect to	all camera parameters
subject to	$\mathbf{R}^\mathrm{T}\mathbf{R} = \mathbf{I}$

- This is the **hard but robust** approach.
- The parameters are obtained by minimizing this nonlinear, non-convex minimization problem.
- It is a small-dimensional optimization problem.

Calibration with a rig

General camera calibration problem

Least squares minimization problem

minimize	$\sum_{i=1}^{N} \left(\varepsilon_x^i\right)^2 + \left(\varepsilon_y^i\right)^2$
with respect to	all camera parameters
subject to	$\mathbf{R}^\mathsf{T}\mathbf{R} = \mathbf{I}$

- It is commonly solved without the constraint with a nonlinear solver: gradient, conjugate gradient or Levenberg-Marquardt algorithm.
- The constraint is then imposed from the resulting $\hat{\mathbf{R}}$. Final rotation matrix $\mathbf{R} = \mathbf{U}_R\mathbf{V}_R^\mathsf{T}$ where these matrices come from the SVD of the unconstrained matrix $\hat{\mathbf{R}} = \mathbf{U}_R\mathbf{S}\mathbf{V}_R^\mathsf{T}$.

Calibration with a rig

General camera calibration problem

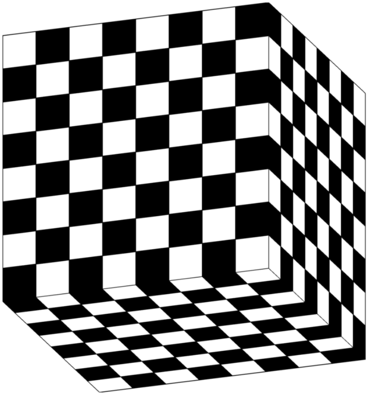
Least squares minimization problem

$$\begin{array}{ll} \text{minimize} & \sum_{i=1}^{N} \left(\varepsilon_x^i\right)^2 + \left(\varepsilon_y^i\right)^2 \\ \text{with respect to} & \text{all camera parameters} \\ \text{subject to} & \mathbf{R}^{\mathsf{T}}\mathbf{R} = \mathbf{I} \end{array}$$

- Since the problem is non-convex it requires a good initialization to get global minimum.
- It can be modified to include radial distortion parameters (how?).
- Parameters are identifiable (unique solution) if $N \geq 6$ and the points are not all co-planar $\Longrightarrow$ calibration with a rig.

Calibration with a rig

Calibration rig



Source: Brian moore81 - Own work, CC BY-SA 4.0, `https://commons.wikimedia.org/w/index.php?curid=37119802`.

Direct camera model estimation

- We can also first estimate **M** from data.
- We know that in homogeneous coordinates $\mathbf{u}^i_{\mathrm{im}} = \alpha\, \mathbf{M}\, \mathbf{u}^i_{\mathrm{w}}$.
  In homogeneous coordinates these vectors are parallel.

Direct camera model estimation

- We can also first estimate **M** from data.
- We know that in homogeneous coordinates $\mathbf{u}^i_{\mathrm{im}} = \alpha\,\mathbf{M}\,\mathbf{u}^i_{\mathrm{w}}$. In homogeneous coordinates these vectors are parallel.
- This can be rewritten with cross-product:

$$\mathbf{u}^i_{\mathrm{im}} \times \mathbf{M}\,\mathbf{u}^i_{\mathrm{w}} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} = \mathbf{0}$$

Direct camera model estimation

- We can also first estimate **M** from data.
- We know that in homogeneous coordinates $\mathbf{u}^i_{\mathrm{im}} = \alpha\,\mathbf{M}\,\mathbf{u}^i_{\mathrm{w}}$.
  In homogeneous coordinates these vectors are parallel.
- This can be rewritten with cross-product:

$$\mathbf{u}^i_{\mathrm{im}} \times \mathbf{M}\,\mathbf{u}^i_{\mathrm{w}} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} = \mathbf{0}$$

- If we define $\mathbf{M} = \begin{bmatrix} \mathbf{m}_1^\top \\ \mathbf{m}_2^\top \\ \mathbf{m}_3^\top \end{bmatrix}$, the cross-product can be rewritten as:

$$\begin{bmatrix} 0 & -1 & y^i_{\mathrm{im}} \\ 1 & 0 & -x^i_{\mathrm{im}} \\ -y^i_{\mathrm{im}} & x^i_{\mathrm{im}} & 0 \end{bmatrix} \begin{bmatrix} \mathbf{m}_1^\top \mathbf{u}^i_{\mathrm{w}} \\ \mathbf{m}_2^\top \mathbf{u}^i_{\mathrm{w}} \\ \mathbf{m}_3^\top \mathbf{u}^i_{\mathrm{w}} \end{bmatrix} = \begin{bmatrix} -\mathbf{m}_2^\top \mathbf{u}^i_{\mathrm{w}} + y^i_{\mathrm{im}}\,\mathbf{m}_3^\top \mathbf{u}^i_{\mathrm{w}} \\ \mathbf{m}_1^\top \mathbf{u}^i_{\mathrm{w}} - x^i_{\mathrm{im}}\,\mathbf{m}_3^\top \mathbf{u}^i_{\mathrm{w}} \\ -y^i_{\mathrm{im}}\,\mathbf{m}_1^\top \mathbf{u}^i_{\mathrm{w}} + x^i_{\mathrm{im}}\,\mathbf{m}_2^\top \mathbf{u}^i_{\mathrm{w}} \end{bmatrix} = \mathbf{0}$$

Calibration with a rig

Direct camera model estimation

- If we define $\mathbf{M} = \begin{bmatrix} \mathbf{m}_1^\mathsf{T} \\ \mathbf{m}_2^\mathsf{T} \\ \mathbf{m}_3^\mathsf{T} \end{bmatrix}$, the cross-product can be rewritten as:

$$\begin{bmatrix} 0 & -1 & y_{\text{im}}^i \\ 1 & 0 & -x_{\text{im}}^i \\ -y_{\text{im}}^i & x_{\text{im}}^i & 0 \end{bmatrix} \begin{bmatrix} \mathbf{m}_1^\mathsf{T}\mathbf{u}_\text{w}^i \\ \mathbf{m}_2^\mathsf{T}\mathbf{u}_\text{w}^i \\ \mathbf{m}_3^\mathsf{T}\mathbf{u}_\text{w}^i \end{bmatrix} = \begin{bmatrix} -\mathbf{m}_2^\mathsf{T}\mathbf{u}_\text{w}^i + y_{\text{im}}^i\,\mathbf{m}_3^\mathsf{T}\mathbf{u}_\text{w}^i \\ \mathbf{m}_1^\mathsf{T}\mathbf{u}_\text{w}^i - x_{\text{im}}^i\,\mathbf{m}_3^\mathsf{T}\mathbf{u}_\text{w}^i \\ -y_{\text{im}}^i\,\mathbf{m}_1^\mathsf{T}\mathbf{u}_\text{w}^i + x_{\text{im}}^i\,\mathbf{m}_2^\mathsf{T}\mathbf{u}_\text{w}^i \end{bmatrix} = \mathbf{0}$$

- We can factor the unknown camera model vectors $\mathbf{m}_1$, $\mathbf{m}_2$ and $\mathbf{m}_3$:

$$\begin{bmatrix} \mathbf{0}^\mathsf{T} & -\mathbf{u}_\text{w}^{i\,\mathsf{T}} & y_{\text{im}}^i\,\mathbf{u}_\text{w}^{i\,\mathsf{T}} \\ \mathbf{u}_\text{w}^{i\,\mathsf{T}} & \mathbf{0}^\mathsf{T} & -x_{\text{im}}^i\,\mathbf{u}_\text{w}^{i\,\mathsf{T}} \\ -y_{\text{im}}^i\,\mathbf{u}_\text{w}^{i\,\mathsf{T}} & x_{\text{im}}^i\,\mathbf{u}_\text{w}^{i\,\mathsf{T}} & \mathbf{0}^\mathsf{T} \end{bmatrix} \begin{bmatrix} \mathbf{m}_1 \\ \mathbf{m}_2 \\ \mathbf{m}_3 \end{bmatrix} = \mathbf{0}$$

- Note that the third row is a linear combination of the first two. So we can delete it from the linear equations.

Direct camera model estimation

- For N points we have $2N$ equations

$$\begin{bmatrix} \mathbf{0}^\mathsf{T} & -\mathbf{u}_\mathrm{w}^{1\,\mathsf{T}} & y_\mathrm{im}^1\,\mathbf{u}_\mathrm{w}^{1\,\mathsf{T}} \\ \mathbf{u}_\mathrm{w}^{1\,\mathsf{T}} & \mathbf{0}^\mathsf{T} & -x_\mathrm{im}^1\,\mathbf{u}_\mathrm{w}^{1\,\mathsf{T}} \\ & \vdots & \\ \mathbf{0}^\mathsf{T} & -\mathbf{u}_\mathrm{w}^{N\,\mathsf{T}} & y_\mathrm{im}^N\,\mathbf{u}_\mathrm{w}^{N\,\mathsf{T}} \\ \mathbf{u}_\mathrm{w}^{N\,\mathsf{T}} & \mathbf{0}^\mathsf{T} & -x_\mathrm{im}^N\,\mathbf{u}_\mathrm{w}^{N\,\mathsf{T}} \end{bmatrix} \begin{bmatrix} \mathbf{m}_1 \\ \mathbf{m}_2 \\ \mathbf{m}_3 \end{bmatrix} = \mathbf{A}\mathbf{m} = \mathbf{0}$$

Direct camera model estimation

- For N points we have $2N$ equations

$$\begin{bmatrix} \mathbf{0}^\top & -\mathbf{u}_\mathrm{w}^{1\,\top} & y_\mathrm{im}^1\,\mathbf{u}_\mathrm{w}^{1\,\top} \\ \mathbf{u}_\mathrm{w}^{1\,\top} & \mathbf{0}^\top & -x_\mathrm{im}^1\,\mathbf{u}_\mathrm{w}^{1\,\top} \\ & \vdots & \\ \mathbf{0}^\top & -\mathbf{u}_\mathrm{w}^{N\,\top} & y_\mathrm{im}^N\,\mathbf{u}_\mathrm{w}^{N\,\top} \\ \mathbf{u}_\mathrm{w}^{N\,\top} & \mathbf{0}^\top & -x_\mathrm{im}^N\,\mathbf{u}_\mathrm{w}^{N\,\top} \end{bmatrix} \begin{bmatrix} \mathbf{m}_1 \\ \mathbf{m}_2 \\ \mathbf{m}_3 \end{bmatrix} = \mathbf{A}\mathbf{m} = \mathbf{0}$$

- $\mathbf{m}$ has 11 free parameters, it can be retrieved from the null-space of $\mathbf{A}$ and we need to correct its scaling (how to do it with SVD?).
- For uniqueness, we need $\dim\{\mathrm{null}(\mathbf{A})\} = 1$, thus we need $\mathrm{rank}(\mathbf{A}) = 11$.
- We can show that $\mathrm{rank}(\mathbf{A}) = 11$ if and only if we have at least $N \geq 6$ non co-planar calibration points.

Direct camera model estimation

- Problem I (Noise): in practice $\mathbf{u}_{\text{im}}^{i}$ are noisy, so we will never get equality.
- Solution: find the $\mathbf{m}$ that is closest to what we want:

$$\begin{array}{ll} \text{minimize} & \|\mathbf{Am}\|_2^2 \\ \text{with respect to} & \mathbf{m} \\ \text{subject to} & \|\mathbf{m}\|_2^2 = 1 \end{array}$$

- The constraint is imposed to avoid trivial solution $\mathbf{m} = \mathbf{0}$.

Direct camera model estimation

- Problem I (Noise): in practice $\mathbf{u}^i_{\text{im}}$ are noisy, so we will never get equality.
- Solution: find the $\mathbf{m}$ that is closest to what we want:

$$\begin{aligned} &\text{minimize} && \|\mathbf{Am}\|_2^2 \\ &\text{with respect to} && \mathbf{m} \\ &\text{subject to} && \|\mathbf{m}\|_2^2 = 1 \end{aligned}$$

- The constraint is imposed to avoid trivial solution $\mathbf{m} = \mathbf{0}$.
- This problem has an analytic solution:

$$\mathbf{m} = \mathbf{v}_{\text{min}}$$

  where $\mathbf{v}_{\text{min}}$ is the singular vector of $\mathbf{A}$ corresponding to the smallest singular value (show it with KKT conditions!).

Direct camera model estimation

- Problem II (Ill-conditioning): in practice $\mathbf{u}_{\mathrm{w}}^{i}$ and $\mathbf{u}_{\mathrm{im}}^{i}$ are of different orders, ex.: $x_{\mathrm{im}} = 500$ pixels and $x_{\mathrm{w}} = 0.2$m, therefore $\mathbf{A}$ has entries in a wide range of values
$\Longrightarrow$ **it can be very ill-conditioned**.
- Noise will produce large errors in $\mathbf{v}_{\mathrm{min}}$.

Direct camera model estimation

- Problem II (Ill-conditioning): in practice $\mathbf{u}_{\mathrm{w}}^{i}$ and $\mathbf{u}_{\mathrm{im}}^{i}$ are of different orders, ex.: $x_{\mathrm{im}} = 500$ pixels and $x_{\mathrm{w}} = 0.2$m, therefore $\mathbf{A}$ has entries in a wide range of values
  $\Longrightarrow$ **it can be very ill-conditioned**.
- Noise will produce large errors in $\mathbf{v}_{\mathrm{min}}$.
- Solution: center and scale separately to unit standard deviation in each coordinate (like in PCA) $\mathbf{u}_{\mathrm{w}}^{i}$ and $\mathbf{u}_{\mathrm{im}}^{i}$.
- This can be done in homogeneous coordinates with linear transformations:

$$\tilde{\mathbf{u}}_{\mathrm{w}}^{i} = \mathbf{T}_{\mathrm{w}}\mathbf{u}_{\mathrm{w}}^{i} \quad \tilde{\mathbf{u}}_{\mathrm{im}}^{i} = \mathbf{T}_{\mathrm{im}}\mathbf{u}_{\mathrm{im}}^{i}$$

Direct camera model estimation

- Problem II (Ill-conditioning): in practice $\mathbf{u}_{\mathrm{w}}^i$ and $\mathbf{u}_{\mathrm{im}}^i$ are of different orders, ex.: $x_{\mathrm{im}} = 500$ pixels and $x_{\mathrm{w}} = 0.2$m, therefore $\mathbf{A}$ has entries in a wide range of values $\Longrightarrow$ **it can be very ill-conditioned**.
- Noise will produce large errors in $\mathbf{v}_{\min}$.
- Solution: center and scale separately to unit standard deviation in each coordinate (like in PCA) $\mathbf{u}_{\mathrm{w}}^i$ and $\mathbf{u}_{\mathrm{im}}^i$.
- This can be done in homogeneous coordinates with linear transformations:

$$\tilde{\mathbf{u}}_{\mathrm{w}}^i = \mathbf{T}_{\mathrm{w}}\mathbf{u}_{\mathrm{w}}^i \quad \tilde{\mathbf{u}}_{\mathrm{im}}^i = \mathbf{T}_{\mathrm{im}}\mathbf{u}_{\mathrm{im}}^i$$

- Previous minimization problem can be solved with $\tilde{\mathbf{A}}$ built with $\tilde{\mathbf{u}}_{\mathrm{w}}^i$ and $\tilde{\mathbf{u}}_{\mathrm{im}}^i$.
- Solution $\tilde{\mathbf{M}}$ is related to $\mathbf{M}$ as follows

$$\mathbf{M} = \mathbf{T}_{\mathrm{im}}^{-1}\tilde{\mathbf{M}}\mathbf{T}_{\mathrm{W}}$$

Direct camera model estimation

- Problem III (Scaling): we have estimated $\alpha\mathbf{M}$ with arbitrary α. How do we retrieve proper scaling?

Direct camera model estimation

- Problem III (Scaling): we have estimated $\alpha\mathbf{M}$ with arbitrary α. How do we retrieve proper scaling?
- Solution: to estimate the scaling α note that

  $$\mathbf{m}_3^{1:3} = [m_{31}\ m_{32}\ m_{33}]^\mathsf{T} = \mathbf{r}_3$$

  from the orthogonality constraint

  $$\|\mathbf{r}_3\|_2 = 1$$

  therefore the properly scaled matrix $\mathbf{M}_s$ is

  $$\mathbf{M}_s = \frac{\mathbf{M}}{\|\mathbf{m}_3^{1:3}\|_2}$$

Direct camera model estimation

- Following all these steps to estimate $\mathbf{M}$ is called the **direct linear transformation (DLT) method**.
- It is the **easy but not robust** approach.

DLT method

---

1. Retrieve $N > 6$ not all co-planar point correspondences:
$$(\mathbf{u}_{\mathrm{im}}^1, \mathbf{u}_{\mathrm{w}}^1), \cdots, (\mathbf{u}_{\mathrm{im}}^N, \mathbf{u}_{\mathrm{w}}^N)$$
2. Center and scale the points with $\tilde{\mathbf{u}}_{\mathrm{w}}^i = \mathbf{T}_{\mathrm{w}}\mathbf{u}_{\mathrm{w}}^i$ and $\tilde{\mathbf{u}}_{\mathrm{im}}^i = \mathbf{T}_{\mathrm{im}}\mathbf{u}_{\mathrm{im}}^i$.
3. Construct matrix $\tilde{\mathbf{A}}$ (slide 22).
4. Retrieve $\tilde{\mathbf{m}} = \mathbf{v}_{\min}$ with $\mathrm{SVD}(\tilde{\mathbf{A}})$.
5. Build $\tilde{\mathbf{M}}$ from $\tilde{\mathbf{m}}$.
6. Retrieve $\mathbf{M}$ with $\mathbf{M} = \mathbf{T}_{\mathrm{im}}^{-1}\tilde{\mathbf{M}}\mathbf{T}_{\mathrm{W}}$.
7. Rescale $\mathbf{M}$ with $\mathbf{M} := \dfrac{\mathbf{M}}{\|\mathbf{m}_3^{1:3}\|_2}$

Camera parameters from camera matrix

- Now that we have **M**, how do we get the camera parameters?

Camera parameters from camera matrix

- Now that we have **M**, how do we get the camera parameters?
- Remember that

$$\mathbf{M} = \mathbf{M}_{\text{int}} \left[\; \mathbf{R} \; \middle| \; \mathbf{t} \; \right]$$

  Its first three columns are $\mathbf{M}^{(1:3)} = \mathbf{M}_{\text{int}}\, \mathbf{R}$.
- This is a factorization into an upper triangular matrix and an orthogonal matrix.
- Such a factorization is known in algebra: **RQ factorization**.

Camera parameters from camera matrix

- Now that we have **M**, how do we get the camera parameters?
- Remember that

$$\mathbf{M} = \mathbf{M}_{\text{int}} \left[\begin{array}{c|c} \mathbf{R} & \mathbf{t} \end{array} \right]$$

  Its first three columns are $\mathbf{M}^{(1:3)} = \mathbf{M}_{\text{int}}\,\mathbf{R}$.
- This is a factorization into an upper triangular matrix and an orthogonal matrix.
- Such a factorization is known in algebra: **RQ factorization**.
- It is the "twin sister" of the QR factorization and many algorithms are available for this factorization (it can be retrieved with QR algorithms).

Camera parameters from camera matrix

- Retrieval of camera parameters - RQ factorization:

$$[\mathbf{M}_{\text{int}}, \mathbf{R}] = \text{RQ}(\mathbf{M}^{(1:3)})$$

- You can retrieve f_x, f_y, f_θ, o_x and o_y from $\mathbf{M}_{\text{int}}$.

Camera parameters from camera matrix

- Retrieval of camera parameters - RQ factorization:

$$[\mathbf{M}_{\text{int}}, \mathbf{R}] = \text{RQ}(\mathbf{M}^{(1:3)})$$

- You can retrieve f_x, f_y, f_θ, o_x and o_y from $\mathbf{M}_{\text{int}}$.
- RQ has sign ambiguities! You should correct it to have the right signs:
  1. f_x is normally positive
  2. f_y is normally negative
  3. You should have $\approx +1$ in the third diagonal element.

- Each time you multiply a column of $\mathbf{M}_{\text{int}}$ by -1 you should also multiply the corresponding row of $\mathbf{R}$.

Camera parameters from camera matrix

- Retrieval of camera parameters - $\mathbf{t}$, from the fourth column:

$$\mathbf{t} = \mathbf{M}_{\text{int}}{}^{-1}\mathbf{M}^{(4)}$$

- Trivially you have $\mathbf{R}_{\text{wc}} = \mathbf{R}$ and $\mathbf{t}_{\text{cw}} = -\mathbf{R}^{-1}\mathbf{t}$.

Calibration with a rig

Camera parameters from camera matrix

- This is the **medium** approach. You can call it **camera calibration using DLT**.
- Its estimates are often used as an initialization for the **hard** approach.

Camera parameters using DLT method

---

1. Retrieve $N > 6$ not all co-planar point correspondences:
$$(\mathbf{u}_{\text{im}}^1, \mathbf{u}_{\text{w}}^1), \cdots, (\mathbf{u}_{\text{im}}^N, \mathbf{u}_{\text{w}}^N)$$
2. Estimate $\mathbf{M}$ using DLT method.
3. Retrieve $\mathbf{M}_{\text{int}}$ and $\mathbf{R}$ with $[\mathbf{M}_{\text{int}}, \mathbf{R}] = \text{RQ}(\mathbf{M}^{(1:3)})$.
4. Correct RQ sign ambiguities by multiplying the columns of $\mathbf{M}_{\text{int}}$ and corresponding rows of $\mathbf{R}$ by -1.
5. Retrieve f_x, f_y, f_θ, o_x and o_y from $\mathbf{M}_{\text{int}}$.
6. Retrieve $\mathbf{t}$ with $\mathbf{t} = \mathbf{M}_{\text{int}}{}^{-1}\mathbf{M}^{(4)}$.

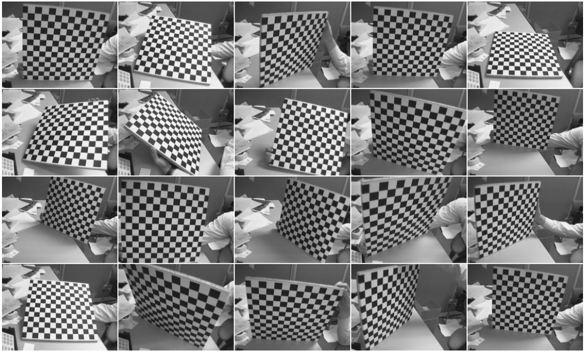
Conclusions

- Camera calibration: information to project 3D points on images.
- Still not enough information to retrieve the 3D structure. Stereo vision next class.
- Two approaches that can be combined: optimization (non linear least squares) and algebra (DLT).
- It is suggested to have 5 times more equations than unknowns in practice, since noise is present. This gives you an idea on how many point correspondences you should use (how many?).

Calibration with planes

Setting the calibration problem

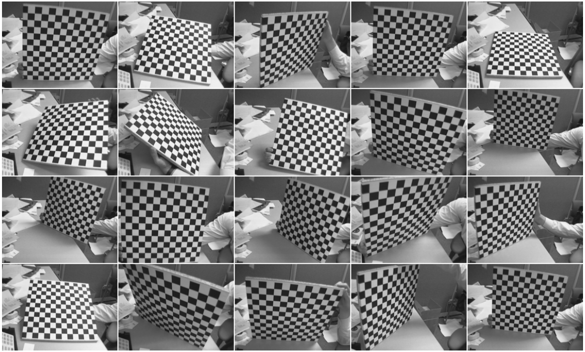
- Problem: it is often difficult to build a rig with good precision.



Source: https://commons.wikimedia.org/wiki/File:Multiple_chessboard_views.png.

Setting the calibration problem

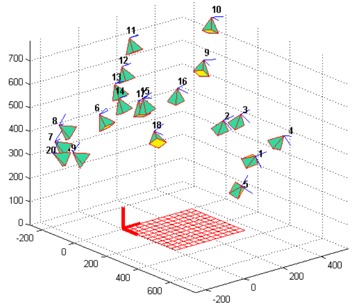
- Problem: it is often difficult to build a rig with good precision.
- Solution: use a plane for calibration, a checkerboard for example, but in different positions and orientations.

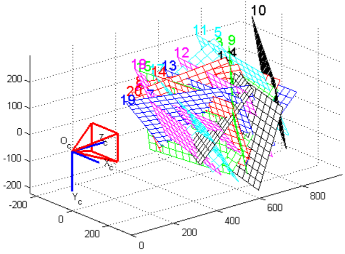


Source: https://commons.wikimedia.org/wiki/File:Multiple_chessboard_views.png.

Setting the calibration problem

- This is by far the most popular calibration method.
- The planes are all supposed to have $z_w = 0$ and the camera is supposed to change its pose (position and orientation). See left figure.
- Even if we actually change the plane pose. See right figure.





Sources: by Brian moore81 - Own work, CC BY-SA 4.0, `https://commons.wikimedia.org/w/index.php?curid=37007186` and `https://commons.wikimedia.org/wiki/File:Multiple_chessboard_views.png`.

Setting the calibration problem

- For K images (plane poses), we want to estimate the parameters of **1 intrinsic matrix** $\mathbf{M}_{\text{int}}$ and **K extrinsic matrices** $\mathbf{M}_{\text{ext}}^{1}, \cdots, \mathbf{M}_{\text{ext}}^{K}$.
- We can also use a non linear least squares approach, that is by appropriately modifying the previous **hard approach** (can you write the cost function?).

Setting the calibration problem

- For K images (plane poses), we want to estimate the parameters of **1 intrinsic matrix $\mathbf{M}_{\text{int}}$** and **K extrinsic matrices $\mathbf{M}_{\text{ext}}^{1}, \cdots, \mathbf{M}_{\text{ext}}^{K}$.**
- We can also use a non linear least squares approach, that is by appropriately modifying the previous **hard approach** (can you write the cost function?).
- In practice, we first estimate all camera parameters using a modification of the medium approach with DLT.
- Then we use the results to initialize the non linear least squares optimization algorithm.

Estimation of homographies

- Can we use the DLT method in this case?

Calibration with planes

Estimation of homographies

- Can we use the DLT method in this case?
- Each plane has N^i point correspondences. From the assumption of $z_w = 0$ for all planes, for the j-th point correspondence, we have

$$
\begin{aligned}
\begin{bmatrix} x_{\text{im}}^j \\ y_{\text{im}}^j \\ 1 \end{bmatrix} &= \alpha \mathbf{M}_{\text{int}} \left[\begin{array}{ccc|c} \mathbf{r}^1 & \mathbf{r}^2 & \mathbf{r}^3 & \mathbf{t} \end{array} \right] \begin{bmatrix} x_w^j \\ y_w^j \\ \mathbf{0} \\ 1 \end{bmatrix} \\
&= \alpha \mathbf{M}_{\text{int}} \left[\begin{array}{cc|c} \mathbf{r}^1 & \mathbf{r}^2 & \mathbf{t} \end{array} \right] \begin{bmatrix} x_w^j \\ y_w^j \\ 1 \end{bmatrix} \\
&= \alpha \mathbf{H}_i \begin{bmatrix} x_w^j \\ y_w^j \\ 1 \end{bmatrix}
\end{aligned}
$$

Estimation of homographies

- Each plane has N^i point correspondences. From the assumption of $z_w = 0$ for all planes, for the j-th point correspondence, we have

$$\begin{bmatrix} x^j_{\text{im}} \\ y^j_{\text{im}} \\ 1 \end{bmatrix} = \alpha \mathbf{H}_i \begin{bmatrix} x^j_{\text{w}} \\ y^j_{\text{w}} \\ 1 \end{bmatrix}$$

- $\mathbf{H}_i$ is a plane-to-plane homogeneous transformation, we call it an **homography**.

Calibration with planes

Estimation of homographies

- Each plane has N^i point correspondences. From the assumption of $z_w = 0$ for all planes, for the j-th point correspondence, we have

$$\begin{bmatrix} x^j_{\text{im}} \\ y^j_{\text{im}} \\ 1 \end{bmatrix} = \alpha \mathbf{H}_i \begin{bmatrix} x^j_{\text{w}} \\ y^j_{\text{w}} \\ 1 \end{bmatrix}$$

- $\mathbf{H}_i$ is a plane-to-plane homogeneous transformation, we call it an **homography**.
- It is a matrix with 9 elements:

$$\mathbf{H}_i = \begin{bmatrix} \mathbf{h}^1_i \ \mathbf{h}^2_i \ \mathbf{h}^3_i \end{bmatrix} = \begin{bmatrix} h^{11}_i & h^{12}_i & h^{13}_i \\ h^{21}_i & h^{22}_i & h^{23}_i \\ h^{31}_i & h^{32}_i & h^{33}_i \end{bmatrix}$$

but only 8 are free due to free scaling α.

Estimation of homographies

- For the point correspondences of each image, we can use the DLT method to estimate the $\mathbf{H}_i$ similarly to what we did for $\mathbf{M}$.
- We cannot rescale $\mathbf{H}_i$ at the end of the DLT procedure, since its third row is not unitary (we do not have the $\mathbf{r}^3$) column.
- We can show that the $\text{rank}(\mathbf{A}^i) = 8$ for $N^i \geq 4$ not all collinear points.

Camera parameters from homographies

- How to get $\mathbf{M}_{\text{int}}$ and $\mathbf{M}_{\text{ext}}^{1}, \cdots, \mathbf{M}_{\text{ext}}^{K}$ from $\mathbf{H}_1, \cdots, \mathbf{H}_k$?
- This is the tricky part, we cannot use the RQ decomposition.
- We are going to retrieve $\mathbf{M}_{\text{int}}$ first using all $\mathbf{H}_i$.

Camera parameters from homographies

- We will use 3 properties about our model:
  1. $\left[\mathbf{h}_i^1\ \mathbf{h}_i^2\ \mathbf{h}_i^3\right] = \alpha \mathbf{M}_{\text{int}} \left[\mathbf{r}_i^1\ \mathbf{r}_i^2\ \mathbf{t}_i\right] \Longrightarrow \mathbf{M}_{\text{int}}^{-1} \left[\mathbf{h}_i^1\ \mathbf{h}_i^2\ \mathbf{h}_i^3\right] = \alpha \left[\mathbf{r}_i^1\ \mathbf{r}_i^2\ \mathbf{t}_i\right]$
     $\Longrightarrow \mathbf{M}_{\text{int}}^{-1}\mathbf{h}_i^1 = \alpha \mathbf{r}_i^1$ and $\mathbf{M}_{\text{int}}^{-1}\mathbf{h}_i^2 = \alpha \mathbf{r}_i^2$ Definition
  2. $(\mathbf{r}^1)^T \mathbf{r}^2 = 0$ Orthogonality
  3. $\|\mathbf{r}^1\|_2 = \|\mathbf{r}^2\|_2$ Same norm

Calibration with planes

Camera parameters from homographies

- We will use 3 properties about our model:

1. $\left[\mathbf{h}_i^1\ \mathbf{h}_i^2\ \mathbf{h}_i^3\right] = \alpha \mathbf{M}_{\text{int}} \left[\mathbf{r}_i^1\ \mathbf{r}_i^2\ \mathbf{t}_i\right] \Longrightarrow \mathbf{M}_{\text{int}}^{-1} \left[\mathbf{h}_i^1\ \mathbf{h}_i^2\ \mathbf{h}_i^3\right] = \alpha \left[\mathbf{r}_i^1\ \mathbf{r}_i^2\ \mathbf{t}_i\right]$

   $\Longrightarrow \mathbf{M}_{\text{int}}^{-1}\mathbf{h}_i^1 = \alpha \mathbf{r}_i^1$ and $\mathbf{M}_{\text{int}}^{-1}\mathbf{h}_i^2 = \alpha \mathbf{r}_i^2$ Definition
2. $(\mathbf{r}^1)^\mathsf{T}\mathbf{r}^2 = 0$ Orthogonality
3. $\|\mathbf{r}^1\|_2 = \|\mathbf{r}^2\|_2$ Same norm

- From properties 1 and 2 we have:

$$(\mathbf{h}_i^1)^\mathsf{T}(\mathbf{M}_{\text{int}}^{-1})^\mathsf{T}\mathbf{M}_{\text{int}}^{-1}\mathbf{h}_i^2 = 0$$

Camera parameters from homographies

- We will use 3 properties about our model:

1. $\left[\mathbf{h}_i^1\ \mathbf{h}_i^2\ \mathbf{h}_i^3\right] = \alpha \mathbf{M}_{\text{int}} \left[\mathbf{r}_i^1\ \mathbf{r}_i^2\ \mathbf{t}_i\right] \Longrightarrow \mathbf{M}_{\text{int}}^{-1} \left[\mathbf{h}_i^1\ \mathbf{h}_i^2\ \mathbf{h}_i^3\right] = \alpha \left[\mathbf{r}_i^1\ \mathbf{r}_i^2\ \mathbf{t}_i\right]$

   $\Longrightarrow \mathbf{M}_{\text{int}}^{-1}\mathbf{h}_i^1 = \alpha \mathbf{r}_i^1$ and $\mathbf{M}_{\text{int}}^{-1}\mathbf{h}_i^2 = \alpha \mathbf{r}_i^2$ Definition
2. $(\mathbf{r}^1)^{\mathsf{T}}\mathbf{r}^2 = 0$ Orthogonality
3. $\|\mathbf{r}^1\|_2 = \|\mathbf{r}^2\|_2$ Same norm

- From properties 1 and 2 we have:

$$(\mathbf{h}_i^1)^{\mathsf{T}}(\mathbf{M}_{\text{int}}^{-1})^{\mathsf{T}}\mathbf{M}_{\text{int}}^{-1}\mathbf{h}_i^2 = 0$$

- From properties 1 and 3 we have:

$$(\mathbf{h}_i^1)^{\mathsf{T}}(\mathbf{M}_{\text{int}}^{-1})^{\mathsf{T}}\mathbf{M}_{\text{int}}^{-1}\mathbf{h}_i^1 = (\mathbf{h}_i^2)^{\mathsf{T}}(\mathbf{M}_{\text{int}}^{-1})^{\mathsf{T}}\mathbf{M}_{\text{int}}^{-1}\mathbf{h}_i^2$$

Camera parameters from homographies

- We can define a symmetric matrix $\mathbf{B}$ with 6 parameters as follows:

$$\mathbf{B} = \begin{bmatrix} b_{11} & b_{12} & b_{13} \\ b_{12} & b_{22} & b_{23} \\ b_{13} & b_{23} & b_{33} \end{bmatrix} = (\mathbf{M}_{\text{int}}^{-1})^{\mathsf{T}} \mathbf{M}_{\text{int}}^{-1}$$

- The 6 parameters can be stacked into a vector:

$$\mathbf{b} = [b_{11}\ b_{12}\ b_{22}\ b_{13}\ b_{23}\ b_{33}]^{\mathsf{T}}$$

Camera parameters from homographies

- We can define a symmetric matrix $\mathbf{B}$ with 6 parameters as follows:

$$\mathbf{B} = \begin{bmatrix} b_{11} & b_{12} & b_{13} \\ b_{12} & b_{22} & b_{23} \\ b_{13} & b_{23} & b_{33} \end{bmatrix} = (\mathbf{M}_{\text{int}}^{-1})^{\mathsf{T}} \mathbf{M}_{\text{int}}^{-1}$$

- The 6 parameters can be stacked into a vector:

$$\mathbf{b} = \left[b_{11} \; b_{12} \; b_{22} \; b_{13} \; b_{23} \; b_{33} \right]^{\mathsf{T}}$$

- The two last equations in the previous slide can be rewritten as:

$$\begin{bmatrix} (\mathbf{g}^i_{12})^{\mathsf{T}} \\ (\mathbf{g}^i_{11} - \mathbf{g}_{22})^{\mathsf{T}} \end{bmatrix} \mathbf{b} = \mathbf{G}^i \mathbf{b} = \mathbf{0}$$

where $(\mathbf{g}^i_{kl})^{\mathsf{T}} = \left[h^i_{k1} h^i_{l1},\; h^i_{k1} h^i_{l2} + h^i_{k2} h^i_{l1},\; h^i_{k2} h^i_{l2},\; h^i_{k3} h^i_{l1} + h^i_{k1} h^i_{l3},\; h^i_{k3} h^i_{l2} + h^i_{k2} h^i_{l3},\; h^i_{k3} h^i_{l3} \right]$

Camera parameters from homographies

- What is the idea behind all this?
  1. We are going to stack all the K linear systems in a big linear system

$$\begin{bmatrix} \mathbf{G}^1 \\ \vdots \\ \mathbf{G}^K \end{bmatrix} \mathbf{b} = \mathbf{G}\mathbf{b} = \mathbf{0}$$

  2. Since there is noise in the homography estimation we are going to find a non trivial **b** that minimizes $\|\mathbf{G}\mathbf{b}\|_2$.
  3. After finding the best **b** up to a scaling factor $\alpha = \dfrac{1}{\lambda}$ we can find the intrinsic parameters from the relation between **b** and $\mathbf{M}_{\text{int}}$:

$$b_{11} = \frac{\lambda}{f_x^2} \qquad b_{12} = -\frac{\lambda f_\theta}{f_x^2 f_y} \qquad b_{22} = \lambda\left(\frac{f_\theta^2}{f_x^2 f_y} + \frac{1}{f_y^2}\right)$$

$$b_{13} = \lambda\left(\frac{f_\theta o_y - f_y o_x}{f_x^2 f_y}\right) \qquad b_{23} = \lambda\left[-\frac{o_x}{f_y^2} - \frac{f_\theta(f_\theta o_y - f_y o_x)}{f_x^2 f_y^2}\right] \qquad b_{33} = \lambda\left[\frac{o_y^2}{f_y^2} - \frac{(f_\theta o_y - f_y o_x)^2}{f_x^2 f_y^2} + 1\right]$$

Camera parameters from homographies

- We can solve the minimization problem

$$\begin{aligned} \text{minimize} \quad & \|\mathbf{Gb}\|_2^2 \\ \text{with respect to} \quad & \mathbf{b} \\ \text{subject to} \quad & \|\mathbf{b}\|_2^2 = 1 \end{aligned}$$

exactly as we solved it for DLT, using the SVD.

Camera parameters from homographies

- We can solve the minimization problem

$$\begin{array}{cc} \text{minimize} & \|\mathbf{Gb}\|_2^2 \\ \text{with respect to} & \mathbf{b} \\ \text{subject to} & \|\mathbf{b}\|_2^2 = 1 \end{array}$$

  exactly as we solved it for DLT, using the SVD.
- It can be shown that this problem has a unique solution when $K > 3$ and the poses of the planes are different (2 equations per plane and 5 unknowns).

Calibration with planes

Camera parameters from homographies

- Intrinsic parameters and scaling factor can be retrieved from **b**:

$$o_y = \frac{b_{12}b_{13} - b_{11}b_{23}}{b_{11}b_{22} - b_{12}^2} \qquad \lambda = \frac{1}{\alpha} = b_{33} - \frac{b_{13}^2 + o_y(b_{12}b_{13} - b_{11}b_{23})}{b_{11}}$$

$$f_x = \sqrt{\frac{\lambda}{b_{11}}} \qquad f_y = -\sqrt{\frac{\lambda b_{11}}{b_{11}b_{22} - b_{12}^2}}$$

$$f_\theta = -b_{12}\frac{f_x^2 f_y}{\lambda} \qquad o_x = \frac{f_\theta o_y}{f_y} - b_{13}\frac{f_x^2}{\lambda}$$

Camera parameters from homographies

- After building $\mathbf{M}_{\text{int}}$ we can retrieve the extrinsic parameters from property 1 :

$$\left[\mathbf{r}_i^1 \ \mathbf{r}_i^2 \ \mathbf{t}_i\right] = \lambda \mathbf{M}_{\text{int}}^{-1} \left[\mathbf{h}_i^1 \ \mathbf{h}_i^2 \ \mathbf{h}_i^3\right]$$

- Vector $\mathbf{r}^3$ is given by $\mathbf{r}^3 = \mathbf{r}^1 \times \mathbf{r}^2$.

Camera parameters from homographies

- After building $\mathbf{M}_{\text{int}}$ we can retrieve the extrinsic parameters from property 1 :

$$\left[\mathbf{r}_i^1 \ \mathbf{r}_i^2 \ \mathbf{t}_i\right] = \lambda \mathbf{M}_{\text{int}}^{-1} \left[\mathbf{h}_i^1 \ \mathbf{h}_i^2 \ \mathbf{h}_i^3\right]$$

- Vector $\mathbf{r}^3$ is given by $\mathbf{r}^3 = \mathbf{r}^1 \times \mathbf{r}^2$.
- Due to noise estimated $\mathbf{R}$ may not be a rotation matrix. In this case we should search for the rotation matrix closest to $\mathbf{R}$ (solution given by SVD of $\mathbf{R}$ as previously presented).

Calibration with planes

Camera parameters from homographies

- You can call this method **camera calibration with homographies**.
- Its estimates are often used as an initialization for non linear least squares (**hard approach**).
- Non linear least squares can take into account radial distortion.

Camera calibration using homographies

---

1. Retrieve $N^i \geq 4$ point correspondences for $K \geq 3$ different planes.
2. Estimate all $\mathbf{H}^i$ using DLT method.
3. Build matrix $\mathbf{G}$ with the $\mathbf{H}^i$ (slides 39 and 40).
4. Find $\mathbf{b}$ using the SVD.
5. Retrieve intrinsic parameters and scaling from $\mathbf{b}$ (slide 42).
6. Build $\mathbf{M}_{\text{int}}$.
7. Retrieve the extrinsic parameters with $\left[\mathbf{r}_i^1\ \mathbf{r}_i^2\ \mathbf{t}_i\right] = \lambda \mathbf{M}_{\text{int}}^{-1} \left[\mathbf{h}_i^1\ \mathbf{h}_i^2\ \mathbf{h}_i^3\right]$ and $\mathbf{r}^3 = \mathbf{r}^1 \times \mathbf{r}^2$ (impose orthogonality on $\mathbf{R}$ if necessary).

Conclusions

- From all the presented methods calibration from planes is the most popular.
- If you already have intrinsic parameters, one plane is enough to estimate the extrinsic parameters (how do you do it?).
- It is suggested to have 5 times more equations than unknowns in practice, since noise is present. This gives you an idea on how many point correspondences and homographies you should use (how many?).