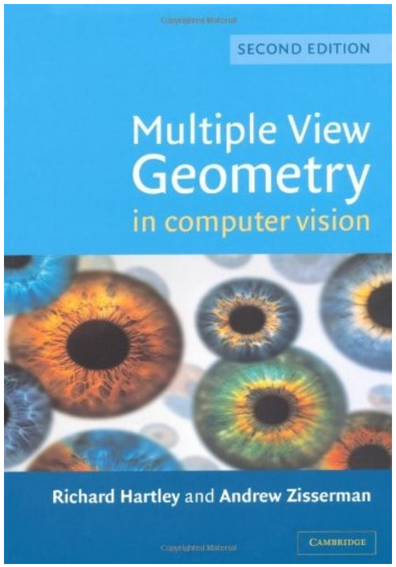
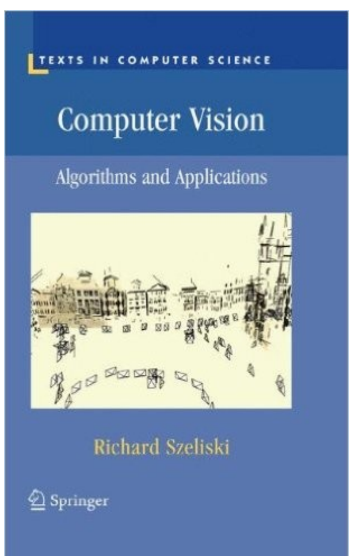
Outline

- Introduction
- Part I: Basics of Mathematical Optimization
  - Linear Least Squares
  - Nonlinear Optimization
- Part II: Basics of Computer Vision
  - Camera Model
  - Multi-Camera Model
  - Multi-Camera Calibration
- Part III: Depth Sensors
  - Passive Stereo
  - Structured Light Cameras
  - Time of Flight Cameras

Literature on Computer Vision



- Richard Hartley, Andrew Zisserman: *Multiple View Geometry in Computer Vision.* 2nd Edition. Cambridge University Press, 2004.
- URL: http://www.robots.ox.ac.uk/~vgg/hzbook/ (sample chapters, Matlab code)

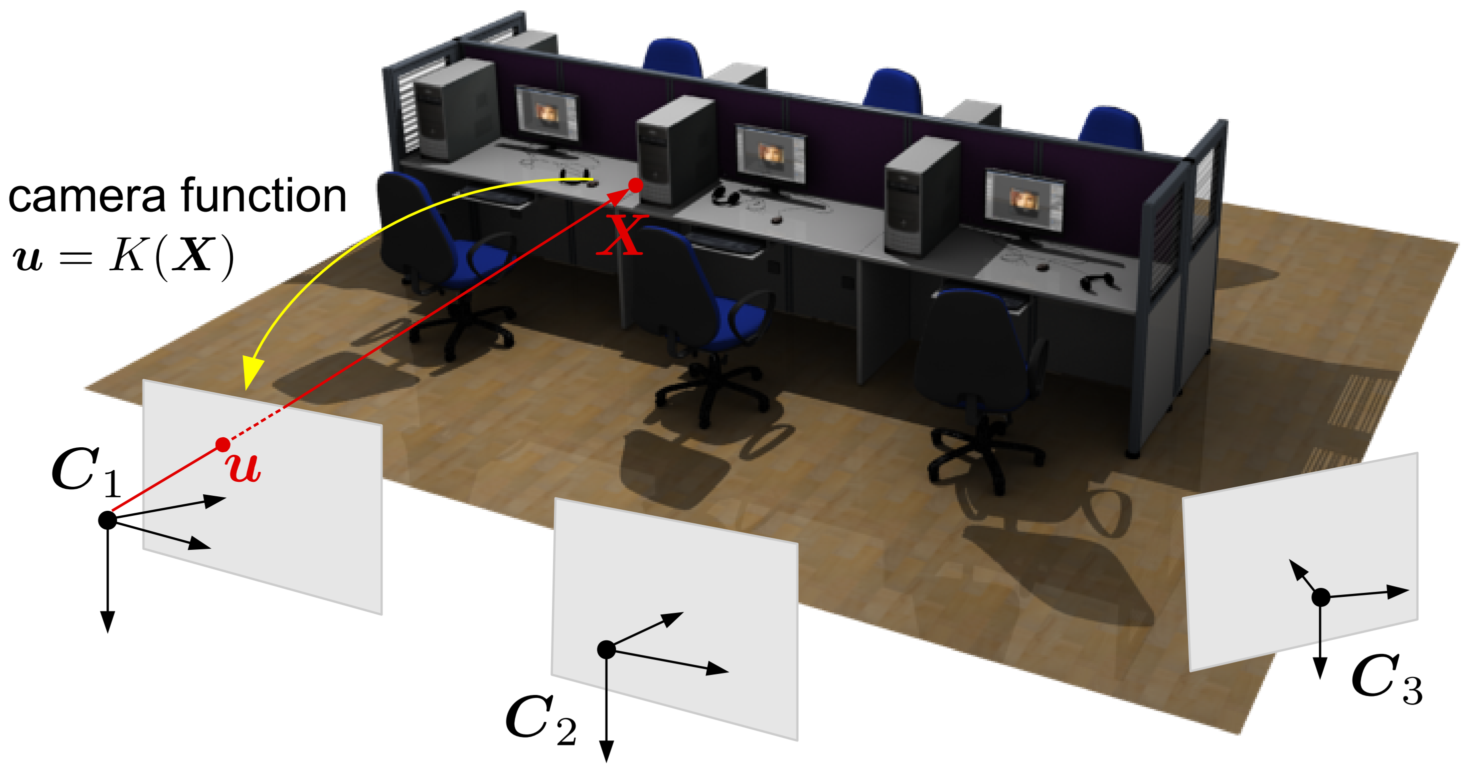


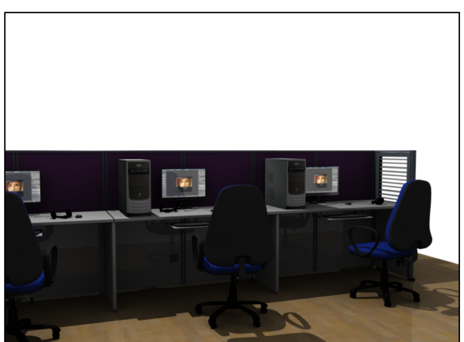
- Richard Szeliski: *Computer Vision. Algorithms and Applications.* Texts in Computer Science. Springer, 2010.
- URL: http://szeliski.org/Book (complete book)

Outline

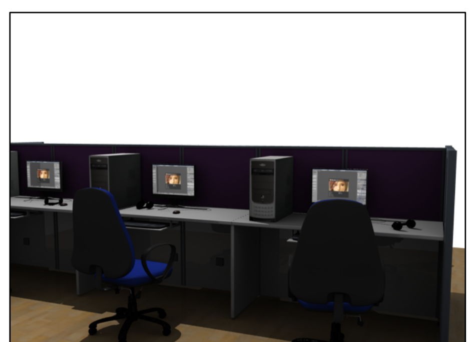
- Introduction
- Part I: Basics of Mathematical Optimization
  - Linear Least Squares
  - Nonlinear Optimization
- Part II: Basics of Computer Vision
  - Camera Model
  - Multi-Camera Model
  - Multi-Camera Calibration
- Part III: Depth Sensors
  - Passive Stereo
  - Structured Light Cameras
  - Time of Flight Cameras

Camera Model

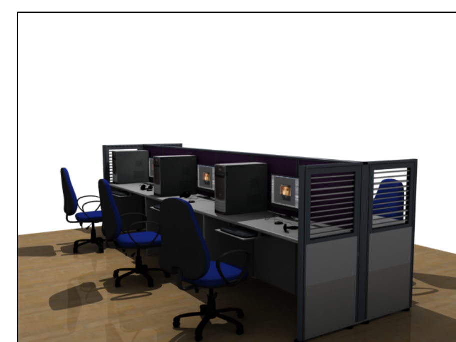




Camera image $\mathbf{I}_1$

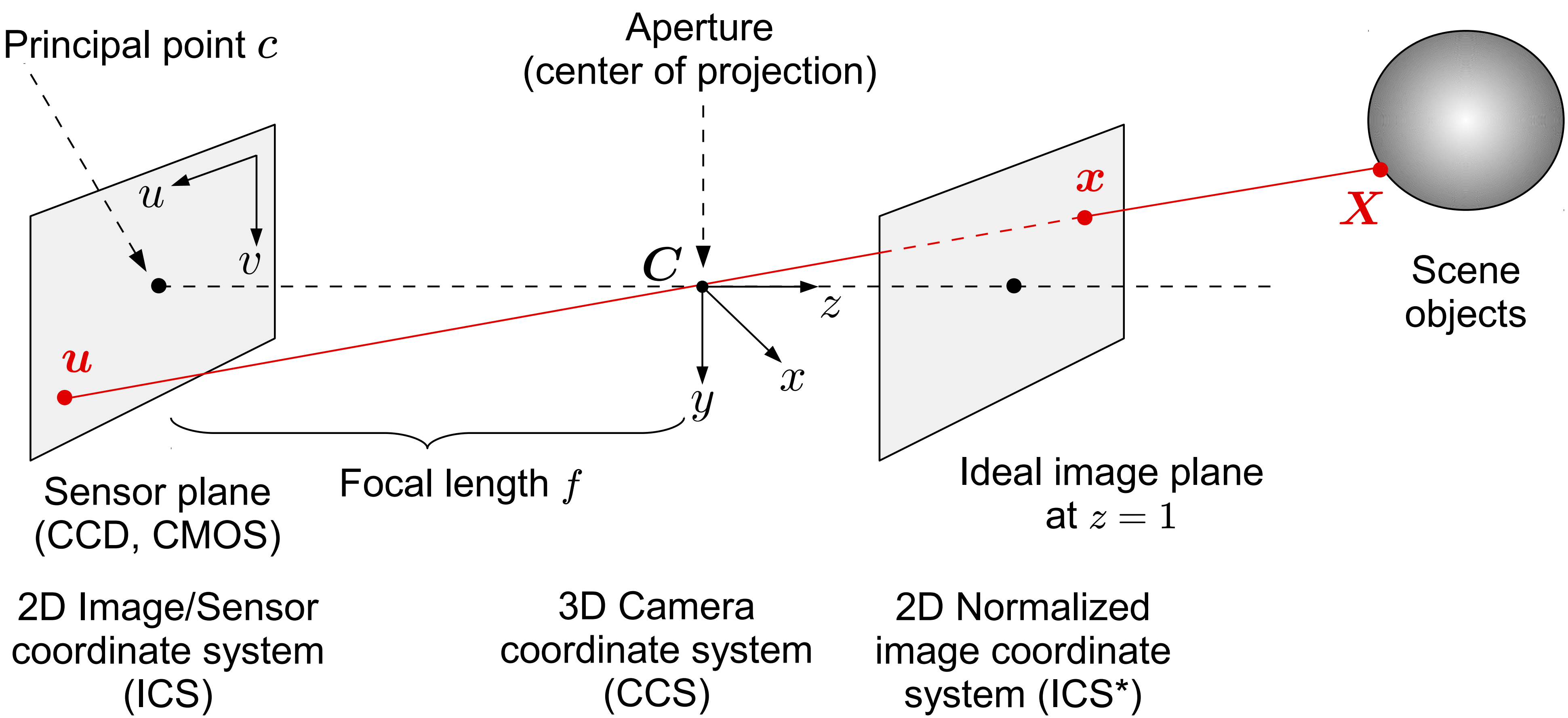


Camera image $\mathbf{I}_2$



Camera image $\mathbf{I}_3$

Camera Model: Pinhole Camera



Camera function K maps 3D points (CCS) to 2D pixel (ICS): $\boldsymbol{u} = K(\boldsymbol{X})$

Pinhole camera model: K is perspective projection (CCS→ICS*) followed by affine transformation to pixel coordinates (ICS→ICS)

Camera Model: Pinhole Camera

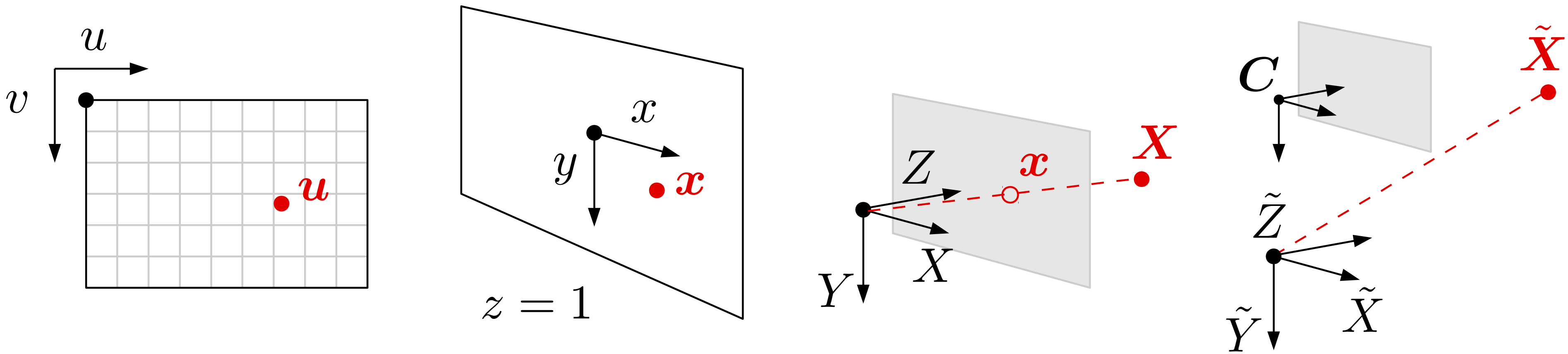


Image coordinate system (ICS) ← Normalized image coordinates ← Camera coordinate system (CCS) ← World coordinate system (WCS)

$$\underbrace{\begin{pmatrix} u \\ v \end{pmatrix}}_{\boldsymbol{u}} = \begin{pmatrix} f_x x + c_x \\ f_y y + c_y \end{pmatrix} \qquad \underbrace{\begin{pmatrix} x \\ y \end{pmatrix}}_{\boldsymbol{x}} = \begin{pmatrix} X/Z \\ Y/Z \end{pmatrix} \qquad \underbrace{\begin{pmatrix} X \\ Y \\ Z \end{pmatrix}}_{\boldsymbol{X}} = \mathbf{R}\tilde{\boldsymbol{X}} + \mathbf{t} \qquad \tilde{\boldsymbol{X}} = \begin{pmatrix} \tilde{X} \\ \tilde{Y} \\ \tilde{Z} \end{pmatrix}$$

Note: Camera position in world is $C := -\mathbf{R}^T \boldsymbol{t}$, columns of $\mathbf{R}$ are world coordinate axes in CCS

Rotation and translation from world to camera coordinates

Camera Model: Pinhole Camera

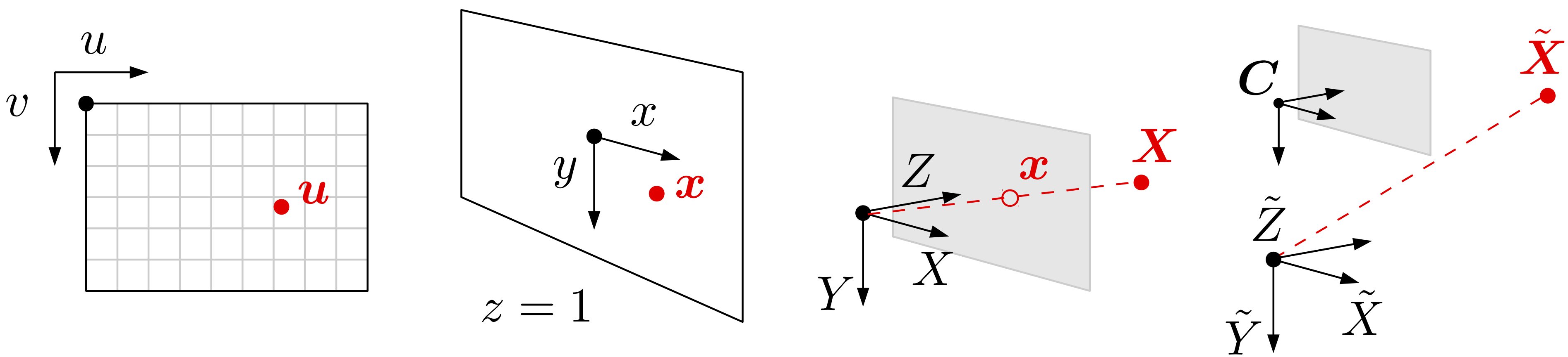
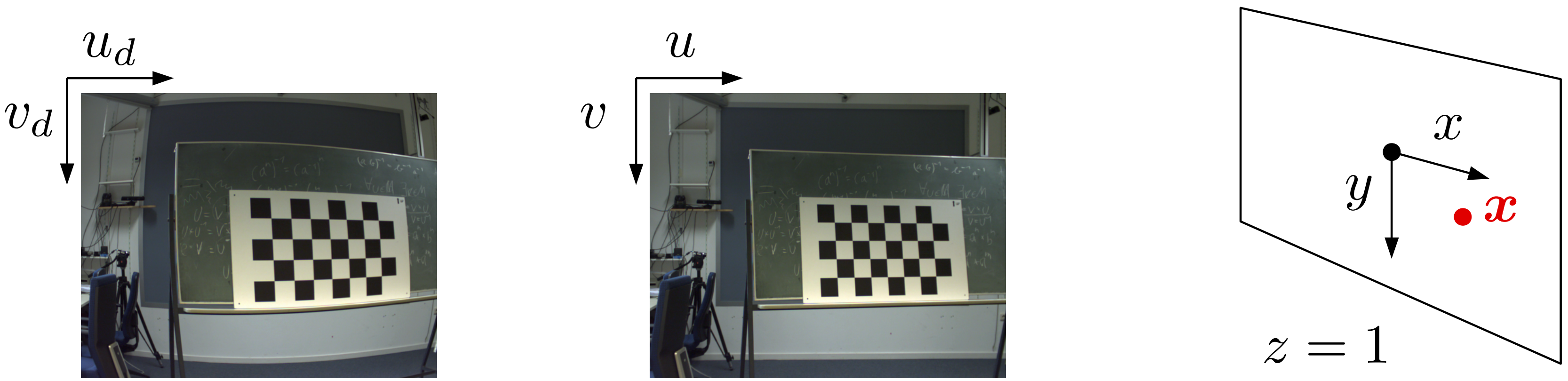


Image coordinate system (ICS) ← Normalized image coordinates ← Camera coordinate system (CCS) ← World coordinate system (WCS)

$$\underbrace{\begin{pmatrix} u \\ v \\ 1 \end{pmatrix}}_{\bar{\boldsymbol{u}}} \sim \underbrace{\begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix}}_{\text{Camera matrix } \mathbf{K} \text{ (intrinsic parameters)}} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \underbrace{\begin{pmatrix} R_{xx} & R_{xy} & R_{xz} & t_x \\ R_{yx} & R_{yy} & R_{yz} & t_y \\ R_{zx} & R_{zy} & R_{zz} & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix}}_{\text{Pose } \mathbf{R},\, \boldsymbol{t} \text{ of world in camera coordinate system}} \underbrace{\begin{pmatrix} \tilde{X} \\ \tilde{Y} \\ \tilde{Z} \\ 1 \end{pmatrix}}_{\bar{\tilde{\boldsymbol{X}}}}$$

Camera Model: Distortion



Distorted image coordinates ← Ideal image coordinates ← Normalized image coordinates

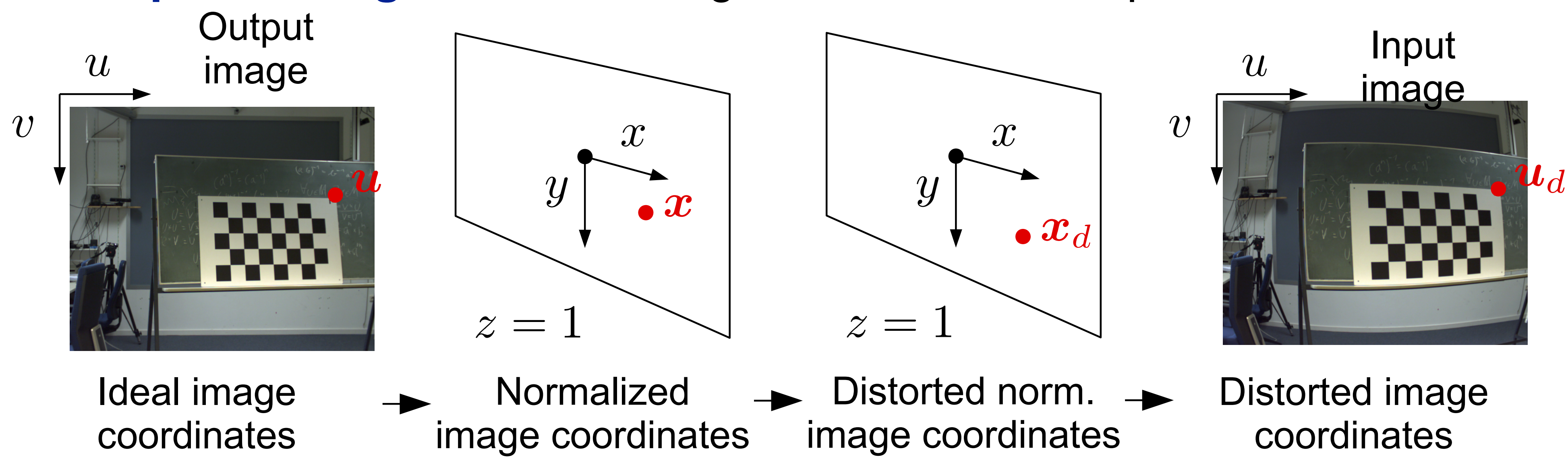
- Real cameras deviate from ideal pinhole model due to lens distortion
- Extend camera function by distortion function $\boldsymbol{x}_d = \mathrm{dist}(\boldsymbol{x})$
- "Plumb Bob" model (Brown, 1966) with parameters $(k_1, k_2, k_3, p_1, p_2)$:

$$\begin{aligned} x_d &= \underbrace{(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)}_{\text{radial distortion}} x + \underbrace{2p_1 xy + p_2(r^2 + 2x^2)}_{\text{tangential distortion}} \\ y_d &= (1 + k_1 r^2 + k_2 r^4 + k_3 r^6) y + p_1(r^2 + 2y^2) + 2p_2 xy \end{aligned}$$

with $r = \sqrt{x^2 + y^2}$

Camera Model: Distortion

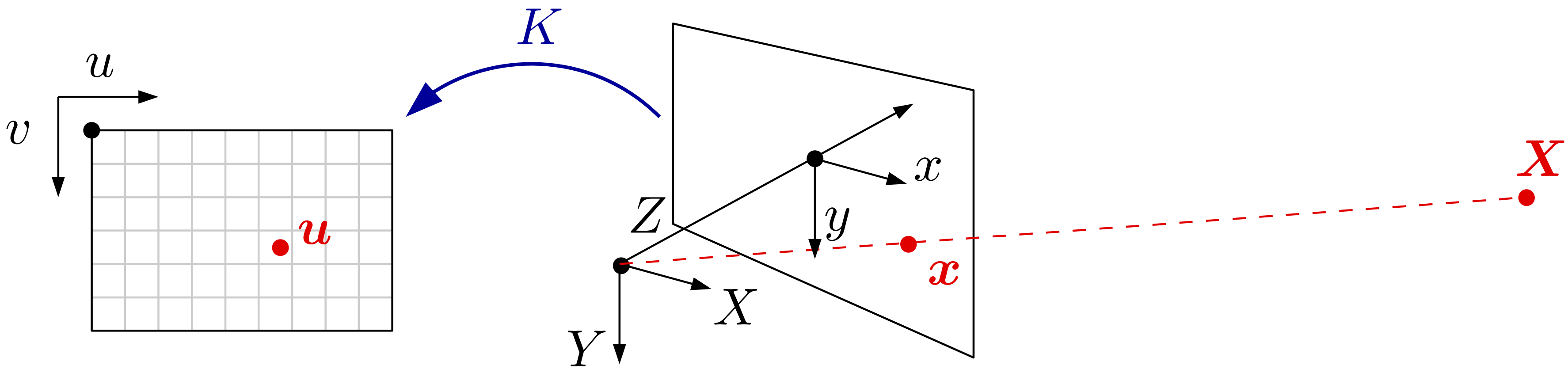
- **Pre-processing:** Undistort images to facilitate computer vision tasks



- Use Backward Mapping: For each output image pixel $(u_{\text{out}}, v_{\text{out}})$:
  - Compute normalized point $(x_{\text{out}}, y_{\text{out}}) = ((u_{\text{out}} - c_x)/f_x, (v_{\text{out}} - c_y)/f_y)$
  - Compute distorted normalized coordinates $(x, y) = \text{dist}(x_{\text{out}}, y_{\text{out}})$
  - Compute distorted input image pixel $(u, v) = (f_x x + c_x, f_y y + c_y)$
  - Set output image value from input image $\text{I}_{\text{out}}(u_{\text{out}}, v_{\text{out}}) = \text{I}(u, v)$

Camera Model: Notations

- Camera function (incl. distortion): $K(\boldsymbol{X}) = \boldsymbol{u}$
- 2D pixel coordinates: $\boldsymbol{u} = (u, v)$
- 3D point (local to camera): $\boldsymbol{X} = (X, Y, Z)$
- 3D point (global): $\tilde{\boldsymbol{X}} = (\tilde{X}, \tilde{Y}, \tilde{Z})$
- Normalized 2D image coordinates: $\boldsymbol{x} = (x, y)$
- Homogeneous coordinates: $\bar{\boldsymbol{x}} = (x, y, 1), \bar{\boldsymbol{u}} = (u, v, 1)$
- For brevity: Unprojection function: $K^{-1}(\boldsymbol{u}) = \bar{\boldsymbol{x}} \sim \boldsymbol{X}$

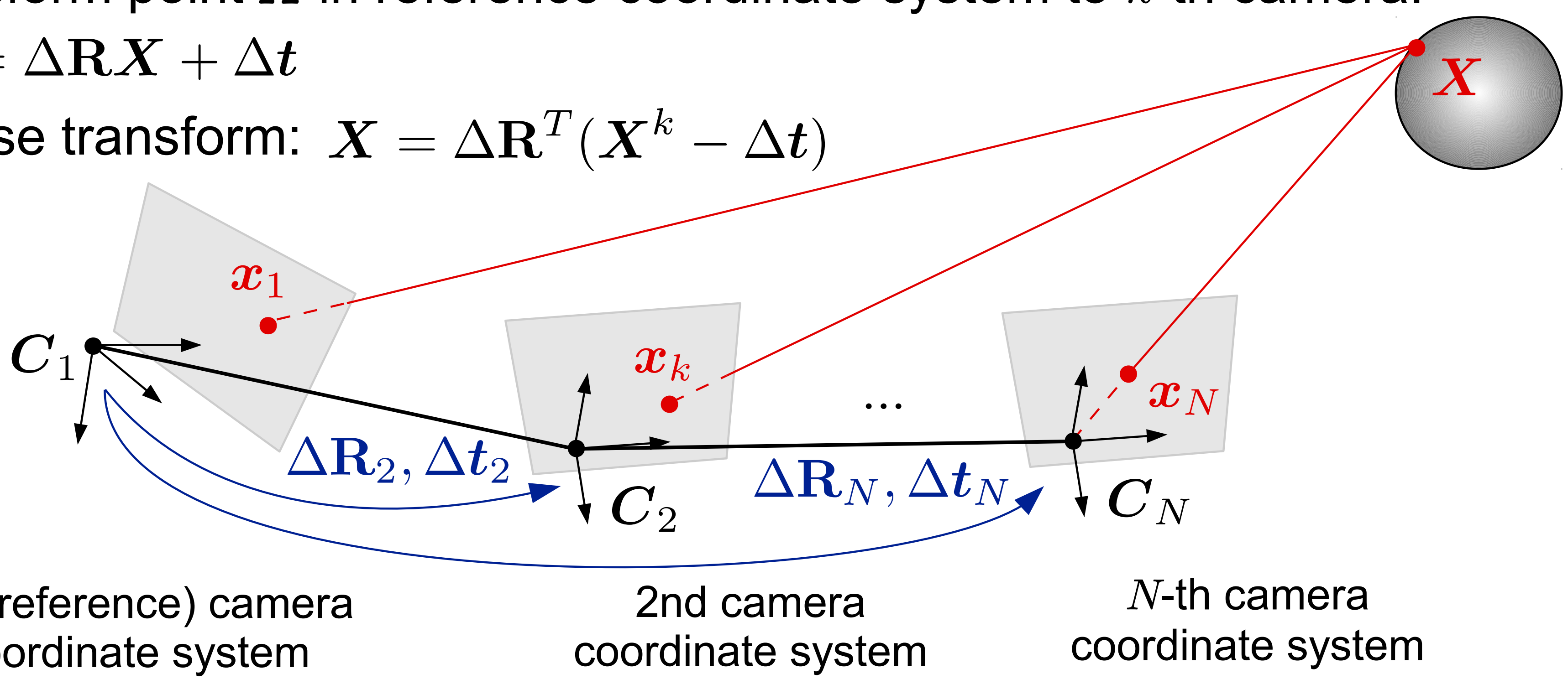


Outline

- Introduction
- Part I: Basics of Mathematical Optimization
  - Linear Least Squares
  - Nonlinear Optimization
- Part II: Basics of Computer Vision
  - Camera Model
  - Multi-Camera Model
  - Multi-Camera Calibration
- Part III: Depth Sensors
  - Passive Stereo
  - Structured Light Cameras
  - Time of Flight Cameras

Multi-Camera Model

- **Multi-camera rig:** Consider N rigidly coupled cameras
- Intrinsic parameters: Camera functions $K_1, \ldots, K_N$
- Extrinsic parameters: Rotation and translation from 1st (reference) to k-th camera coordinate system ($\Delta\mathbf{R}_k$, $\Delta\boldsymbol{t}_k$ for $k = 2, \ldots, N$)
- Transform point $\boldsymbol{X}$ in reference coordinate system to k-th camera:
  $\boldsymbol{X}^k = \Delta\mathbf{R}\boldsymbol{X} + \Delta\boldsymbol{t}$
- Inverse transform: $\boldsymbol{X} = \Delta\mathbf{R}^T(\boldsymbol{X}^k - \Delta\boldsymbol{t})$

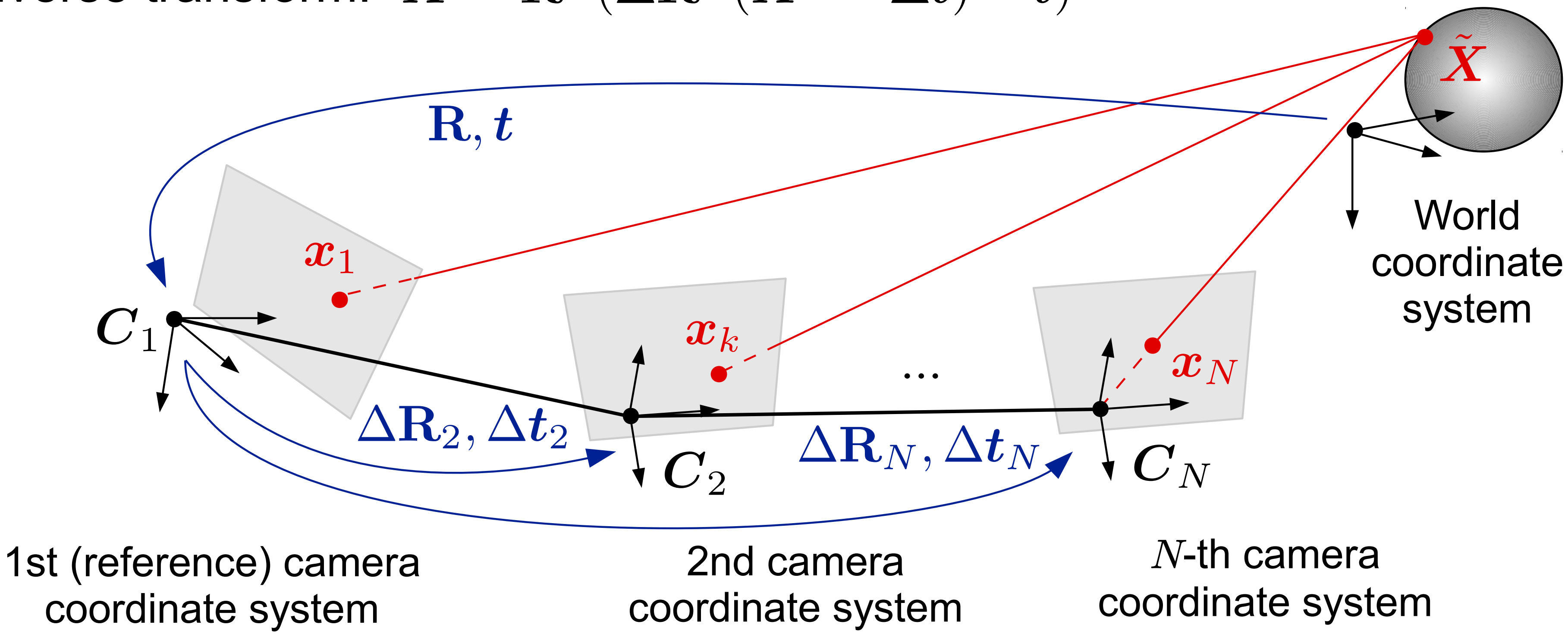


Multi-Camera Model

- Consider the reference camera is located in a world coordinate system
- Transformation from world to reference camera is $\mathbf{R}, \boldsymbol{t}$
- Transform point $\tilde{\boldsymbol{X}}$ in world coordinate system to k-th camera:

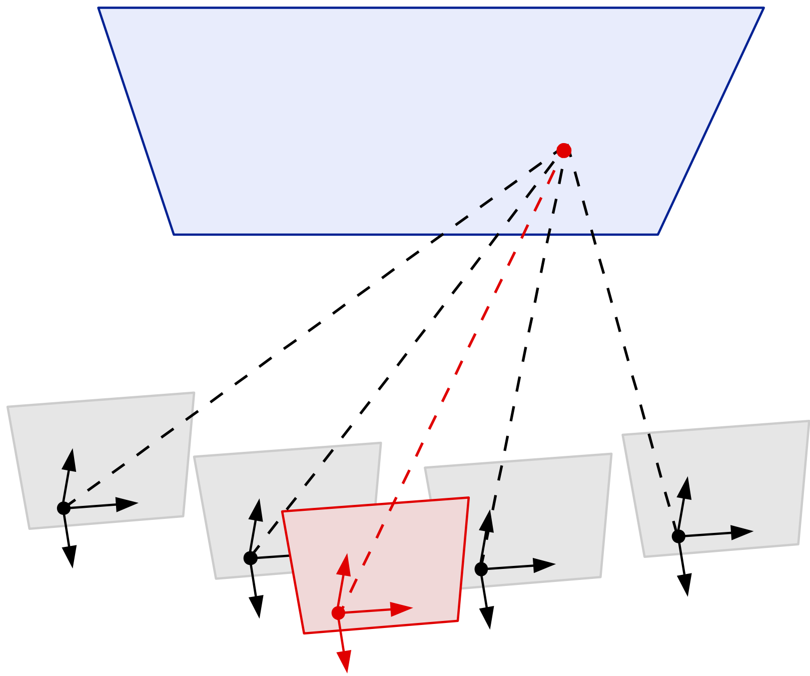
$$\boldsymbol{X}^k = \Delta\mathbf{R}(\mathbf{R}\tilde{\boldsymbol{X}} + \boldsymbol{t}) + \Delta\boldsymbol{t}$$

- Inverse transform: $\tilde{\boldsymbol{X}} = \mathbf{R}^T(\Delta\mathbf{R}^T(\boldsymbol{X}^k - \Delta\boldsymbol{t}) - \boldsymbol{t})$



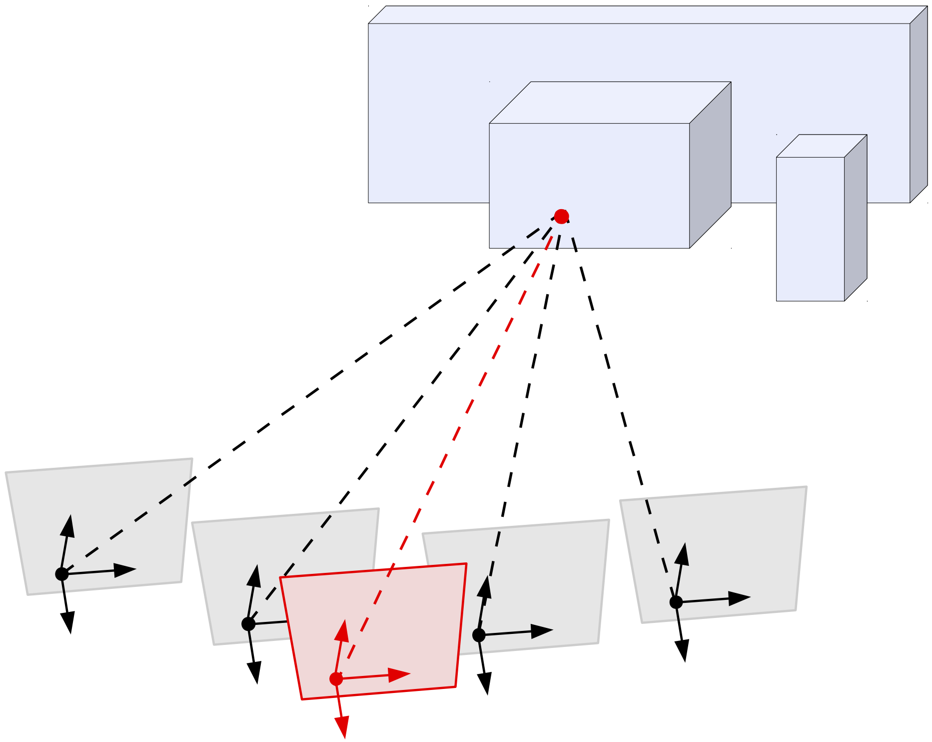
Multi-Camera Model: Applications

Synthetic aperture:



- Only mapping from focus plane to cameras needed (plane-and-parallax approach)

Free-viewpoint rendering:



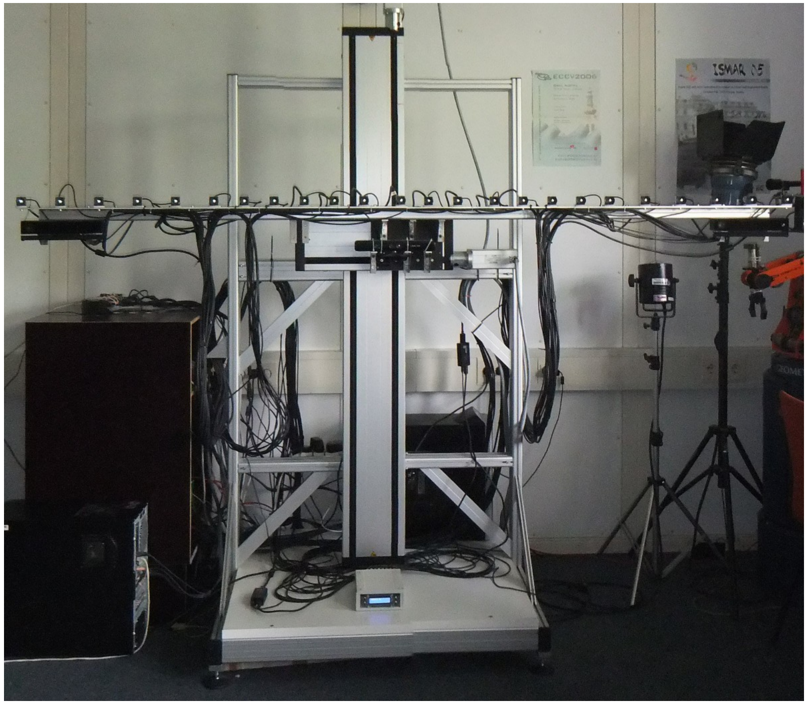
- 3D geometry should be known
- Full extrinsic calibration needed

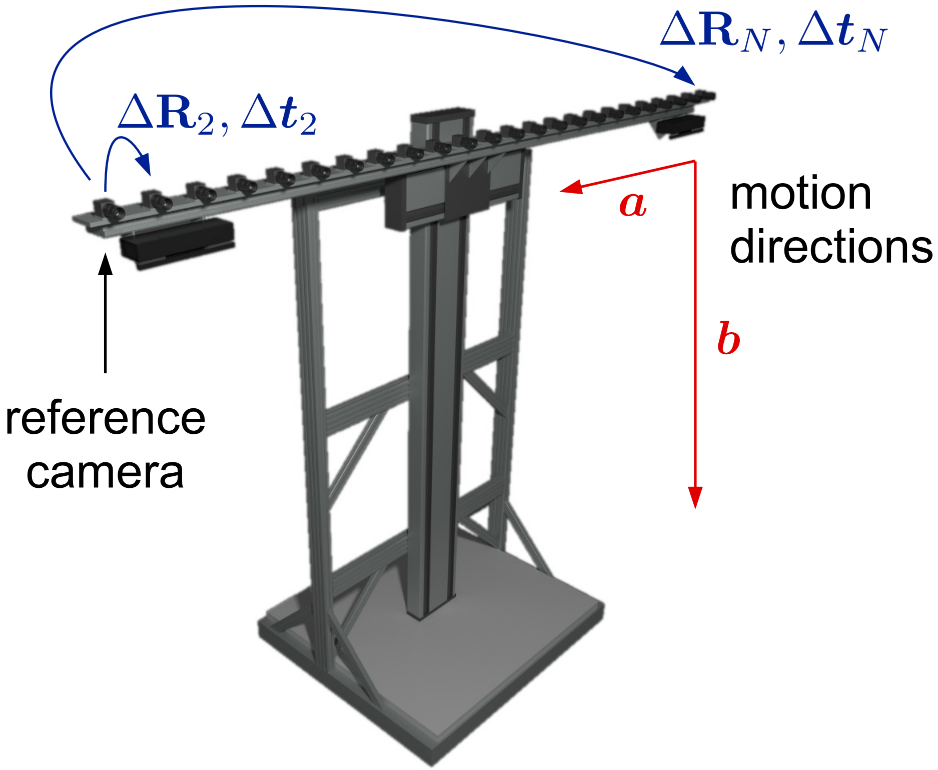
Outline

- Introduction
- Part I: Basics of Mathematical Optimization
  - Linear Least Squares
  - Nonlinear Optimization
- Part II: Basics of Computer Vision
  - Camera Model
  - Multi-Camera Model
  - Multi-Camera Calibration
- Part III: Depth Sensors
  - Passive Stereo
  - Structured Light Cameras
  - Time of Flight Cameras

Multi-Camera Calibration

- **Task:** Calibrate movable multi-camera system for light field capturing
- Intrinsic calibration: Find camera function parameters for each camera
- Extrinsic calibration: Find camera poses in reference coordinate system
- Hand-eye calibration: Find motion directions of motor within reference coordinate system

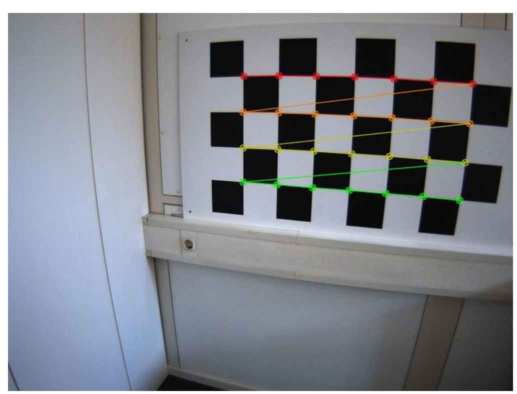


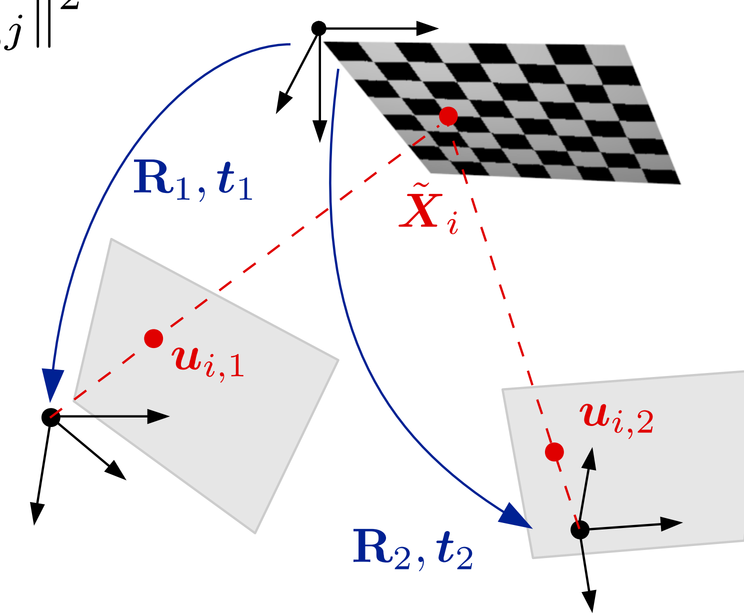


Intrinsic Calibration

- **Task:** Find model parameters of camera function K (intrinsic parameters)
- **General approach:**
  - Capture n images of known calibration object (*e. g.*, checkerboard)
  - Identify m projected 3d points in images (*e. g.*, checkerboard corner)
  - Estimate model parameters by minimizing the reprojection error

$$g(\boldsymbol{\theta}) = \sum_{j=1}^{n} \sum_{i=1}^{m} \| K(\mathbf{R}_j \tilde{\boldsymbol{X}}_i + \boldsymbol{t}_j) - \boldsymbol{u}_{i,j} \|^2$$





Intrinsic Calibration

- **Given:** 3D points $\tilde{\boldsymbol{X}}_1, \dots, \tilde{\boldsymbol{X}}_m \in \mathbb{R}^3$ in world coordinate system and corresponding 2D points $\boldsymbol{u}_{1,1} \dots, \boldsymbol{u}_{m,n} \in \mathbb{R}^2$ in n images captured at different positions ($\boldsymbol{u}_{i,j}$ is image of i-th point in j-th image)
- **Aim:** Find intrinsic parameters $\boldsymbol{\kappa} = (f_x, f_y, c_x, c_y)$ of camera function

  $$K(\boldsymbol{X}) = (f_x X/Z + c_x, f_y Y/Z + c_y)^T$$

  and pose parameters $\boldsymbol{\varrho}_j, \boldsymbol{t}_j$ (rotation + translation) for each image j
- **Solution:** Minimize reprojection errors (least squares problem):

  $$\min_{\substack{\boldsymbol{\kappa}, \boldsymbol{\varrho}_{1,\dots,n} \\ \boldsymbol{t}_{1,\dots,n}}} \sum_{j=1}^{n} \sum_{i=1}^{m} \| K_{\boldsymbol{\kappa}}(\mathbf{R}_{\boldsymbol{\varrho}_j} \boldsymbol{X}_i + \boldsymbol{t}_j) - \boldsymbol{u}_{i,j} \|^2$$

- Include image distortion in camera function:

  $$K(\boldsymbol{X}) = (f_x \mathrm{dist}_x(X/Z) + c_x, f_y \mathrm{dist}_y(Y/Z) + c_y)^T$$

  and append distortion coefficients to $\boldsymbol{\kappa} = (f_x, f_y, c_x, c_y, r_1, r_2, r_3, \dots)$

Intrinsic Calibration

- Nonlinear least squares problem can be solved with iterative methods (*e. g.*, Levenberg-Marquardt algorithm)
- However, initial solution $\boldsymbol{\theta}$ is needed as starting point!
- **Questions:**
  - How to obtain an initial solution for this problem?

    $\rightarrow$ lab session
  - How to parametrize rotation in nonlinear optimization?

Rotation Parametrization in Optimization Problems

- Rotation has 3 degrees of freedom
- Minimal (unconstrained) parametrization with vector of size 3
- **Task:** Select parameter vector $\boldsymbol{\varrho}$ for rotation matrix $\mathbf{R}$
  - 3×3 matrix entries $\boldsymbol{\varrho} = (R_{xx}, R_{xy}, R_{xz}, R_{yx}, R_{yy}, R_{yz}, R_{zx}, R_{zy}, R_{zz})$, with constraints $\mathbf{R}^T\mathbf{R} = \mathbf{I}$
  - Euler angles $\boldsymbol{\varrho} = (\alpha, \beta, \gamma)$
  - rotation axis and angle $\boldsymbol{\varrho} = (\boldsymbol{r}, \varphi)$ with constraint $\boldsymbol{r}^T\boldsymbol{r} = 1$
  - rotation vector $\boldsymbol{\varrho} = \varphi\boldsymbol{r}$
  - unit quaternion $\boldsymbol{\varrho} = \mathbf{q} = (\sin(\varphi/2)\boldsymbol{r}, \cos(\varphi/2))$ with constraint $\mathbf{q}^T\mathbf{q} = 1$
  - reduced unit quaternion $\boldsymbol{\varrho} \in \mathbb{R}^3$ with $\boldsymbol{\varrho} = f(\mathbf{q}), \mathbf{q} = f^{-1}(\boldsymbol{\varrho})$

Rotation Parametrization in Optimization Problems

- **Parametrization:** 3×3 matrix entries $\boldsymbol{\varrho} = (R_{xx}, R_{xy}, R_{xz}, \ldots, R_{zz}) \in \mathbb{R}^9$

$$\mathbf{R} = \begin{pmatrix} R_{xx} & R_{xy} & R_{xz} \\ R_{yx} & R_{yy} & R_{yz} \\ R_{zx} & R_{zy} & R_{zz} \end{pmatrix} = \begin{pmatrix} \boldsymbol{r}_x & \boldsymbol{r}_y & \boldsymbol{r}_z \end{pmatrix} = \begin{pmatrix} \boldsymbol{R}_x \\ \boldsymbol{R}_y \\ \boldsymbol{R}_z \end{pmatrix}$$

$$\mathbf{R}^T\mathbf{R} = \mathbf{I} \;\Rightarrow\; \boldsymbol{r}_x^T\boldsymbol{r}_x = 1,\; \boldsymbol{r}_y^T\boldsymbol{r}_y = 1,\; \boldsymbol{r}_z^T\boldsymbol{r}_z = 1,$$
$$\boldsymbol{r}_x^T\boldsymbol{r}_y = 0,\; \boldsymbol{r}_x^T\boldsymbol{r}_z = 0,\; \boldsymbol{r}_y^T\boldsymbol{r}_z = 0$$

  - Number of parameters: 9
  - Number of constraints: 6
- **Pro:** Rotation is linear transformation
- **Contra:** Multiple quadratic constraints per rotation

Rotation Parametrization in Optimization Problems

- **Parametrization:** Euler angles/Tait-Bryan angles $\boldsymbol{\varrho} = (\alpha, \beta, \gamma) \in \mathbb{R}^3$

$$\mathbf{R} = \mathbf{R}_x(\alpha)\mathbf{R}_y(\beta)\mathbf{R}_z(\gamma)$$

$$\mathbf{R}_x(\alpha) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\alpha) & -\sin(\alpha) \\ 0 & \sin(\alpha) & \cos(\alpha) \end{pmatrix}, \quad \mathbf{R}_y(\beta) = \begin{pmatrix} \cos(\beta) & 0 & \sin(\beta) \\ 0 & 1 & 0 \\ -\sin(\beta) & 0 & \cos(\beta) \end{pmatrix},$$

$$\mathbf{R}_z(\gamma) = \begin{pmatrix} \cos(\gamma) & -\sin(\gamma) & 0 \\ \sin(\gamma) & \cos(\gamma) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Other combinations also used, *e. g.*, rotate around z-, x-, y-axis, or rotate around x-, y-, x-axis

  - Number of parameters: 3
  - Number of constraints: 0
- **Pro:** Rotation parametrization is unconstrained
- **Contra:** 3rd order degree polynomial of trigonometric functions, singularities might occur (Gimbal lock)

Camera Calibration from 2D/3D Correspondences

- **Given:** 3D points $\boldsymbol{X}_1, \dots, \boldsymbol{X}_m \in \mathbb{R}^3$ in world coordinate system and corresponding 2D points $\boldsymbol{u}_{1,1} \dots, \boldsymbol{u}_{m,n} \in \mathbb{R}^2$ in n images captured at different positions ($\boldsymbol{u}_{i,j}$ is image of i-th point in j-th image)
- **Aim:** Find parameters $\boldsymbol{\theta} = (f_x, f_y, c_x, c_y)$ of camera function

  $K : \mathbb{R}^3 \to \mathbb{R}^2, \boldsymbol{X} \mapsto (f_x X/Z + c_x, f_y Y/Z + c_y)^T$

  and pose parameters $\boldsymbol{\varrho}_j, \boldsymbol{t}_j$ (rotation + translation) for each image
- **Solution:** Minimize reprojection errors (least squares problem):

$$\min_{\substack{\boldsymbol{\theta}, \boldsymbol{\varrho}_{1,\dots,n} \\ \boldsymbol{t}_{1,\dots,n}}} \sum_{j=1}^{n} \sum_{i=1}^{m} \| K_{\boldsymbol{\theta}}(\mathbf{R}_{\boldsymbol{\varrho}_j} \boldsymbol{X}_i + \boldsymbol{t}_j) - \boldsymbol{u}_{i,j} \|^2$$

- Include image distortion in camera function:

  $K : \mathbb{R}^3 \to \mathbb{R}^2, \boldsymbol{X} \mapsto (f_x \mathrm{dist}_x(X/Z) + c_x, f_y \mathrm{dist}_y(Y/Z) + c_y)^T$

  and append distortion coefficients to $\boldsymbol{\theta} = (f_x, f_y, c_x, c_y, r_1, r_2, r_3, \dots)$

Rotation Parametrization in Optimization Problems

- **Parametrization:** Rotation axis and angle $\boldsymbol{\varrho} = (\boldsymbol{r}, \varphi) \in \mathbb{R}^4$

$$\mathbf{R} = \begin{pmatrix} r_x^2(1-c)+c & r_x r_y(1-c) - r_z s & r_x r_z(1-c) + r_y s \\ r_x r_y(1-c) + r_z s & r_y^2(1-c)+c & r_y r_z(1-c) - r_x s \\ r_x r_z(1-c) - r_y s & r_y r_z(1-c) + r_x s & r_z^2(1-c)+c \end{pmatrix}$$

$$s := \sin(\varphi),\ c := \cos(\varphi),\ \boldsymbol{r}^T\boldsymbol{r} = 1$$

  - Number of parameters: 4
  - Number of constraints: 1

$$[\boldsymbol{r}]_\times := \begin{pmatrix} 0 & -r_z & r_y \\ r_z & 0 & -r_x \\ -r_y & r_x & 0 \end{pmatrix}$$

- Rodrigues formula: $\mathbf{R} = \mathbf{I} + \sin(\varphi)[\boldsymbol{r}]_\times + (1-\cos(\varphi))[\boldsymbol{r}]_\times^2$
- **Pro:** Less complex than Euler angles, geometrically intuitive
- **Contra:** Product of quadratic polynomial and trigonometric functions, still needs one quadratic constraint per rotation in optimization

Rotation Parametrization in Optimization Problems

- **Parametrization:** Rotation vector (Euler vector) $\boldsymbol{\varrho} = \varphi \boldsymbol{r} \in \mathbb{R}^3$

$$\mathbf{R} = \mathbf{I} + \sin(\varphi)[\boldsymbol{r}]_\times + (1 - \cos(\varphi))[\boldsymbol{r}]_\times^2$$

$$\varphi := \|\boldsymbol{\varrho}\|, \ \boldsymbol{r} := \frac{\boldsymbol{\varrho}}{\|\boldsymbol{\varrho}\|}$$

  - Number of parameters: 3
  - Number of constraints: 0
- **Pro:** Rotation parametrization is unconstrained
- **Contra:** More complex than separate axis/angle, zero rotation is special case ($\|\boldsymbol{\varrho}\| = 0$)

Rotation Parametrization in Optimization Problems

- **Parametrization:** Unit quaternion $\boldsymbol{\varrho} = \underbrace{(\boldsymbol{q}, q)}_{\mathbf{q}} \in \mathbb{R}^4,\ \underbrace{\boldsymbol{q}^T\boldsymbol{q} + q^2}_{\|\mathbf{q}\|} = 1$

$$\boldsymbol{q} = \sin(\tfrac{\varphi}{2})\boldsymbol{r},\ q = \cos(\tfrac{\varphi}{2}) \quad \Rightarrow \ \varphi := 2\,\mathrm{atan2}(\|\boldsymbol{q}\|, q),\ \boldsymbol{r} := \frac{\boldsymbol{q}}{\|\boldsymbol{q}\|}$$

$$\mathbf{R} = (q^2+1)\mathbf{I} + 2q[\boldsymbol{q}]_\times + 2[\boldsymbol{q}]_\times^2$$

  - Number of parameters: 4
  - Number of constraints: 1
- **Pro:** Rotation is only quadratic function
- **Contra:** Parametrization is geometrically unintuitive, still needs one quadratic constraint per rotation in optimization
- Reduced parametrization without constraints: $\boldsymbol{\varrho} = \dfrac{\boldsymbol{q}}{q+1} \in \mathbb{R}^3$

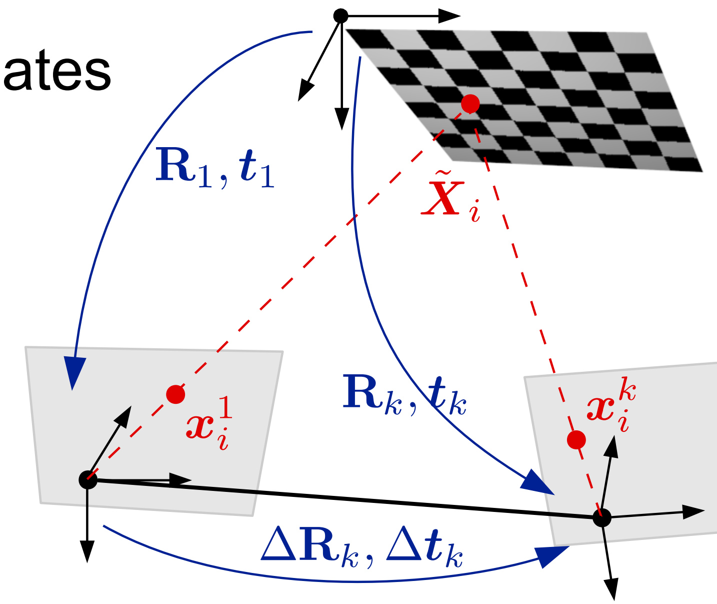
$$\boldsymbol{\varrho}^T\boldsymbol{\varrho} + 1 = \frac{\boldsymbol{q}^T\boldsymbol{q} + q^2 + 2q + 1}{(q+1)^2} = \frac{2}{q+1} \ \Rightarrow \ q := \frac{2}{\boldsymbol{\varrho}^T\boldsymbol{\varrho}+1} - 1,\ \boldsymbol{q} := (q+1)\boldsymbol{\varrho}$$

Extrinsic Calibration

- **Task:** Find rotation and translation from reference camera to k-th cameras (extrinsic parameters) with known camera functions $\mathbf{K}_1, \mathbf{K}_k$
- **General approach:**
  - Capture images of calibration object simultaneously for 1st camera and k-th camera
  - Compute world-to-camera transformations (absolute poses) $\mathbf{R}_1, \boldsymbol{t}_1$ and $\mathbf{R}_k, \boldsymbol{t}_k$ from 2D/3D correspondences for each camera
  - Use normalized 2D image coordinates for reprojection error computation
  - Transformation from 1st to k-th camera is computes as

$$\Delta\mathbf{R}_k = \mathbf{R}_k \mathbf{R}_1^T$$

$$\Delta\boldsymbol{t}_k = -\mathbf{R}_1^T \mathbf{t}_1 + \boldsymbol{t}_k$$

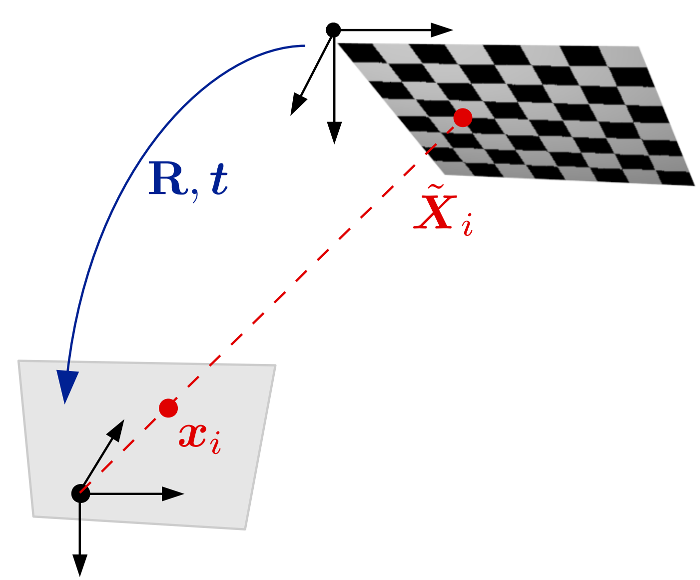


Absolute Pose Estimation

- **Given:** 3D points $\tilde{\boldsymbol{X}}_1, \ldots, \tilde{\boldsymbol{X}}_m \in \mathbb{R}^3$ in world coordinate system and corresponding 2D points $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_m \in \mathbb{R}^2$ in image coordinates
- Camera function $K : \mathbb{R}^3 \to \mathbb{R}^2$ assumed to be known from calibration
- **Aim:** Find Euclidean transformation (rotation $\mathbf{R} \in SO(3)$ + translation $\boldsymbol{t} \in \mathbb{R}^3$) so that $\boldsymbol{u}_i = K(\mathbf{R}\tilde{\boldsymbol{X}}_i + \boldsymbol{t})$ holds for all 2D/3D correspondences
- **Solution:** Minimize reprojection errors (least squares problem):

$$\min_{\boldsymbol{\varrho},\boldsymbol{t}} \sum_{i=1}^{m} \|P(\mathbf{R}_{\boldsymbol{\varrho}}\tilde{\boldsymbol{X}}_i + \boldsymbol{t}) - \boldsymbol{x}_i\|^2$$

where $P(\boldsymbol{X}) = \begin{pmatrix} X/Z \\ Y/Z \end{pmatrix}$



- **Questions:**
  - How to find initial solution? $\to$ lab session
  - How to parametrize rotation?

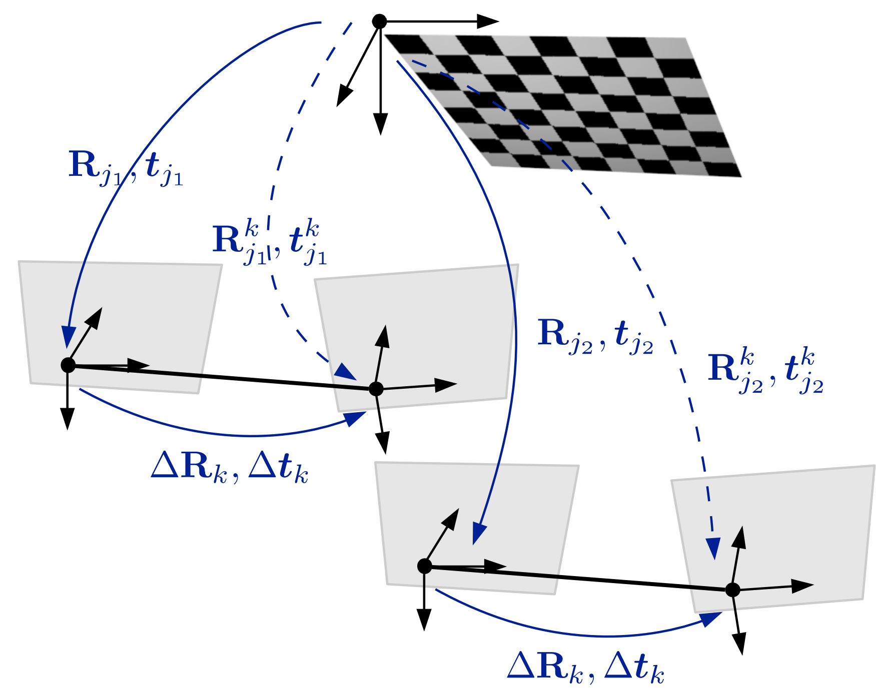
Extrinsic Calibration

- Refine extrinsic parameters using image from n positions of the multi-camera rig to improve accuracy
- **Parameters for optimization:** Reference camera poses $\mathbf{R}_j, \boldsymbol{t}_j$ for each image $j = 1, \ldots, n$ and extrinsic parameters $\Delta\mathbf{R}_k, \Delta\boldsymbol{t}_k$ for k-th camera (rotations are parametrized by $\boldsymbol{\varrho}_{1,\ldots,n}, \Delta\boldsymbol{\varrho}_k$)
- Absolute pose for k-th camera in j-th image is now predicted as

$$\mathbf{R}_j^k = \Delta\mathbf{R}_k\mathbf{R}_j, \; \boldsymbol{t}_j^k = \Delta\mathbf{R}_k\boldsymbol{t}_j + \Delta\boldsymbol{t}_k$$

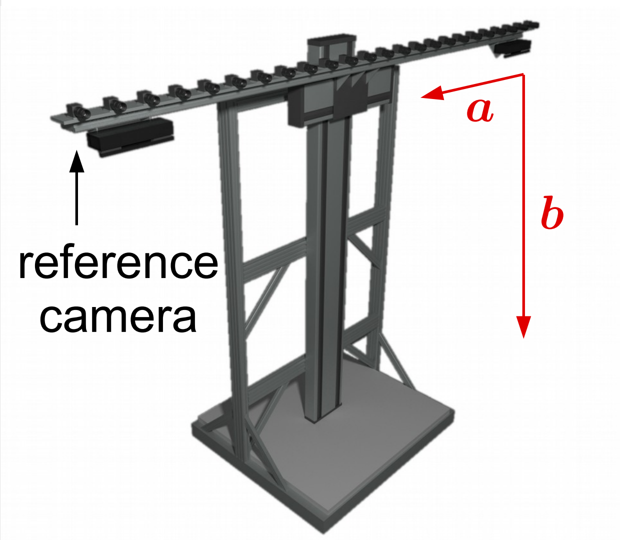
- Minimize joint reprojection error:

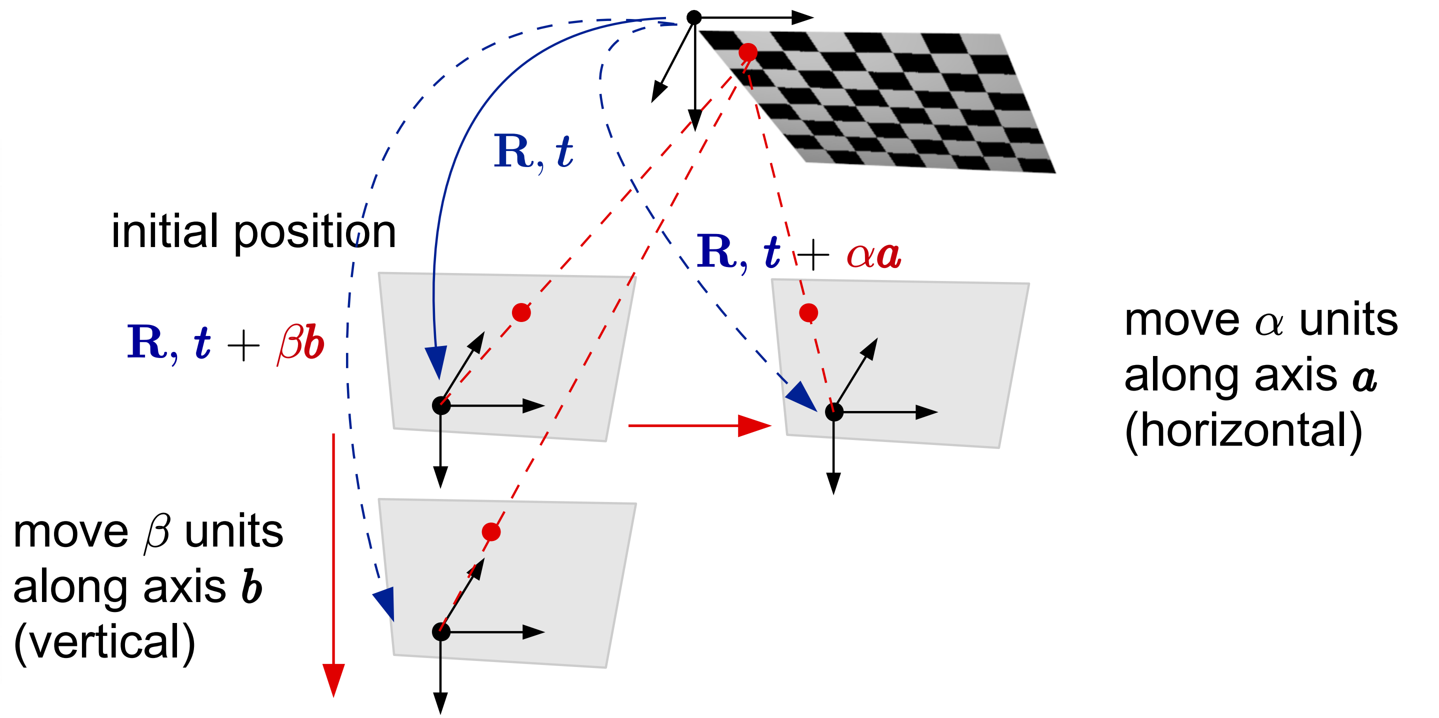
$$\min_{\substack{\boldsymbol{\varrho}_{1,\ldots,n} \\ \boldsymbol{t}_{1,\ldots,n} \\ \Delta\boldsymbol{\varrho}_k, \Delta\boldsymbol{t}_k}} \sum_{j=1}^{n}\sum_{i=1}^{m} \|P(\mathbf{R}_j\tilde{\boldsymbol{X}}_i + \boldsymbol{t}_j) - \boldsymbol{x}_{i,j}^1\|^2 + \sum_{j=1}^{n}\sum_{i=1}^{m} \|P(\mathbf{R}_j^k\tilde{\boldsymbol{X}}_i + \boldsymbol{t}_j^k) - \boldsymbol{x}_{i,j}^k\|^2$$



Hand-Eye Calibration

- Assume rig can be moved horizontally and vertically, offsets α, β (distances to initial position along motion axes) are measured precisely.
- **Question:** What are the motion directions $\boldsymbol{a}, \boldsymbol{b}$ within the reference camera coordinate system (RCS)?
- **Approach:** Deduce directions from relative poses to initial position





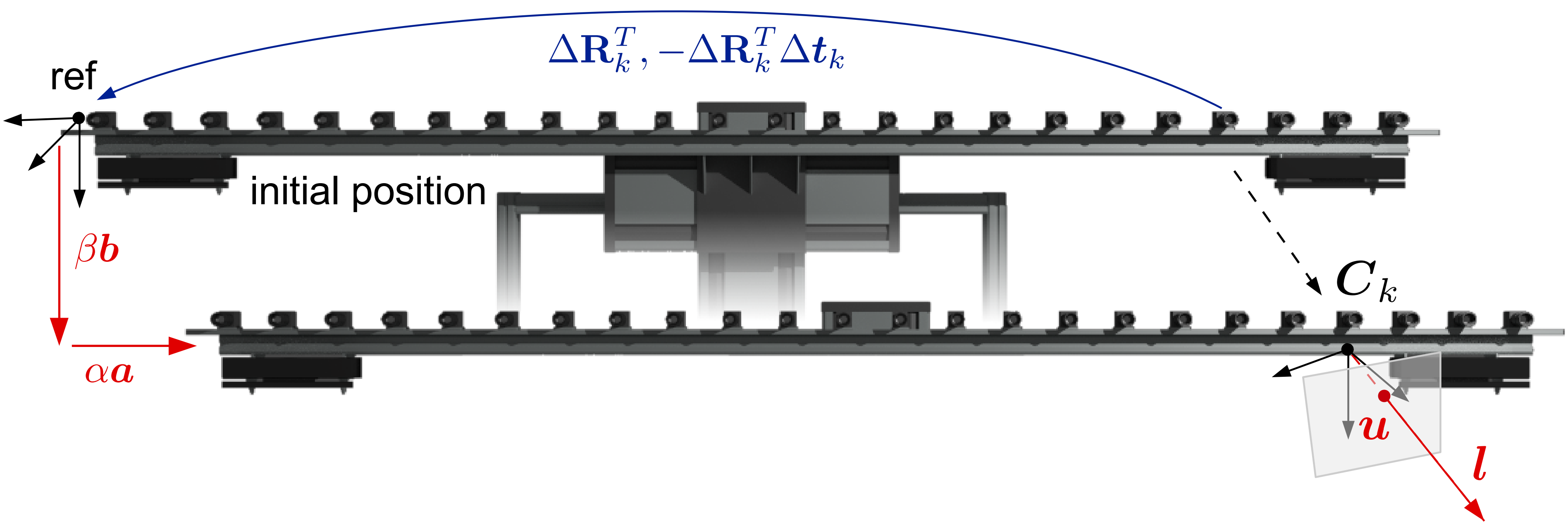
Hand-Eye Calibration

- Assume rig can be moved horizontally and vertically, offsets α, β (distances to initial position along motion axes) are measured precisely.
- **Question:** What are the motion directions $\boldsymbol{a}$, $\boldsymbol{b}$ within the reference camera coordinate system (RCS)?
- **Approach:** Deduce directions from relative poses to initial position
  - Compute absolute pose $\mathbf{R}$, $\boldsymbol{t}$ for reference camera from 2D/3D correspondences for initial position $\alpha = 0$, $\beta = 0$
  - For n positions α_j, β_j, $j = 1, \ldots, n$: Capture image, measure offset
  - Minimize reprojection error with initial pose $\mathbf{R}$, $\boldsymbol{t}$ and $\boldsymbol{a}$, $\boldsymbol{b}$ as optimization parameters (initial solution $\boldsymbol{a} = (1, 0, 0)$, $\boldsymbol{b} = (0, 1, 0)$):

$$\min_{\substack{\boldsymbol{\varrho},\boldsymbol{t}\\ \boldsymbol{a},\boldsymbol{b}}} \sum_{j=1}^{n} \sum_{i=1}^{m} \| P(\mathbf{R}\tilde{\mathbf{X}}_i + \boldsymbol{t}_j + \alpha_j \boldsymbol{a} + \beta_j \boldsymbol{b}) - \boldsymbol{x}_{i,j} \|^2$$

Multi-Camera Calibration

- **Result:** Parameters for mapping from k-th camera image pixels $I_k(u,v)$ to ray-parametrized light field $L(x,y,z,\varphi,\theta)$
- Origin of ray is k-th camera center in ref. system: $\mathbf{C}_k := -\Delta\mathbf{R}_k^T \Delta\boldsymbol{t}_k$
- If captured at motor position α, β, shift to: $\mathbf{C}_k = -\Delta\mathbf{R}_k^T \Delta\boldsymbol{t}_k + \alpha\boldsymbol{a} + \beta\boldsymbol{b}$
- Pixel $I_k(u,v)$ maps to $L(\mathbf{C}_k,\varphi,\theta)$ where (φ,θ) are spherical coordinates of ray direction $\boldsymbol{l} := \Delta\mathbf{R}_k^T K_k^{-1}(u,v)$



Summary

Topics covered in this lecture:

- Camera model used for (movable) multi-camera rigs
- Formulation of light field capturing using multi-camera rigs
- Basic approach to parameter estimation for multi-camera model
- Basic understanding of numerical methods used in nonlinear estimation

Open issues:

- Depth cameras → after the coffee break
- Initial solutions for intrinsic calibration and pose estimation
- Evaluation of intrinsic/extrinsic calibration, error analysis
- Radiometric camera calibration, color correction
  → will be covered as exercises in lab session