Intrinsic and Extrinsic Parameters
Rigid Transformations, Homogeneous Coordinates, and Perspective Projection

Amir

Lecture Outline

- Camera models and coordinate systems
- Rigid transformations in 3D (detailed)
- Homogeneous coordinates and projective geometry
- Intrinsic and extrinsic camera parameters
- Perspective projection model
- Lens distortion models
- Camera calibration overview
- Advanced examples and applications
- Discussion questions

Extrinsic Camera Parameters

- **Definition:** Parameters that define the location and orientation of the camera coordinate system with respect to the world coordinate system.
- **Components:**
  - **Rotation matrix (R):** Describes the camera's orientation (3 degrees of freedom).
  - **Translation vector (t):** Describes the position of the camera center (3 degrees of freedom).
- **Transformation:** $\mathbf{X}_c = \mathbf{R}\mathbf{X}_w + \mathbf{t}$

Intrinsic Camera Parameters

- **Definition:** Internal characteristics of the camera that map the 3D camera coordinates to 2D pixel coordinates.
- **Intrinsic Matrix (K):**

$$\mathbf{K} = \begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

- **Key Components:**
  - f_x, f_y: Focal lengths in pixels.
  - c_x, c_y: Principal point coordinates.
  - s: Skew coefficient (usually 0).

Why Camera Geometry?

- Relates 3D world points to 2D image measurements
- Foundation of computer vision tasks:
  - Camera calibration [Zhang, 2000]
  - 3D reconstruction [Hartley and Zisserman, 2004]
  - Visual odometry and SLAM [Davison et al., 2007]
  - Augmented reality
- Separates *geometry* from *appearance*
- Enables metric measurements from images

Key Insight: A camera is a function $f : \mathbb{R}^3 \to \mathbb{R}^2$ that loses depth information.

Red apple

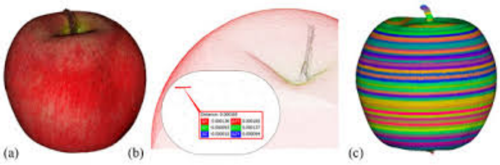


Figure: Photogrammetric dense reconstruction of an apple fruit (a), the highlighted point density of the 3D fruit (ca 0.16 mm) (b) and the apple sliced to find the maximum cross-section diameter (c).

Reference

Grilli, E., Battisti, R., & Remondino, F. (2021).
An advanced photogrammetric solution to measure apples.
Remote Sensing, 13(19), 3960.
DOI: 10.3390/rs13193960

Coordinate Systems in Detail

- **World coordinates** (X_w, Y_w, Z_w): Fixed reference frame
- **Camera coordinates** (X_c, Y_c, Z_c): Origin at optical center
- **Image coordinates** (x, y): Metric coordinates in image plane
- **Pixel coordinates** (u, v): Integer pixel locations

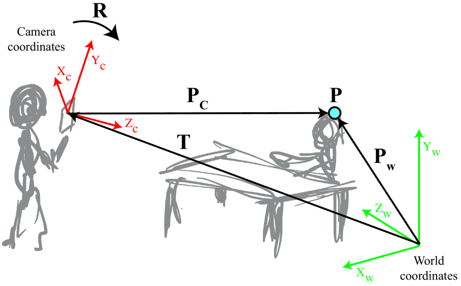


Figure: Camera coordinate systems [Szeliski, 2011]

Rigid Transformation

- **Definition:** A mapping $g : \mathbb{R}^3 \to \mathbb{R}^3$ that preserves the Euclidean distance between all points [Szeliski, 2011].
- **Key Characteristics:**
  - $\|g(\mathbf{X}) - g(\mathbf{Y})\| = \|\mathbf{X} - \mathbf{Y}\| \quad \forall \mathbf{X}, \mathbf{Y} \in \mathbb{R}^3$ [Szeliski, 2011].
  - Does not include scaling or reflection; only rotation and translation [Szeliski, 2011].
- **Matrix Representation:**

$$\mathbf{X}_c = \mathbf{R}\mathbf{X}_w + \mathbf{t}$$

where $\mathbf{R} \in SO(3)$ is a 3×3 rotation matrix and $\mathbf{t} \in \mathbb{R}^3$ is a translation vector [Szeliski, 2011].

Mathematical Representation of Rigid Transformations

Rigid Transformation: A mapping $g : \mathbb{R}^3 \to \mathbb{R}^3$ such that:

$$\|g(\mathbf{X}) - g(\mathbf{Y})\| = \|\mathbf{X} - \mathbf{Y}\| \quad \forall \mathbf{X}, \mathbf{Y} \in \mathbb{R}^3$$

Matrix Representation:

$$\mathbf{X}_c = \mathbf{R}\mathbf{X}_w + \mathbf{t}$$

where:

- $\mathbf{R} \in SO(3)$: Rotation matrix (3×3)
- $\mathbf{t} \in \mathbb{R}^3$: Translation vector
- $SO(3) = \{\mathbf{R} \in \mathbb{R}^{3\times 3} : \mathbf{R}^T\mathbf{R} = \mathbf{I}, \det(\mathbf{R}) = 1\}$

Rotation Representations

Three common representations of rotation:

1. **Rotation Matrices** (9 parameters with 6 constraints)

$$\mathbf{R} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$$

2. **Euler Angles** (3 parameters: yaw, pitch, roll)

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

3. **Axis-Angle and Quaternions** (4 parameters, no singularities)

$$\mathbf{q} = [\cos(\theta/2), \mathbf{v}\sin(\theta/2)]$$

Properties of Rotation Matrices

For any rotation matrix $\mathbf{R}$:

Theorem (Orthogonality)

$$\mathbf{R}^T\mathbf{R} = \mathbf{R}\mathbf{R}^T = \mathbf{I}$$

Theorem (Determinant)

$$\det(\mathbf{R}) = 1$$

Theorem (Inverse)

$$\mathbf{R}^{-1} = \mathbf{R}^T$$

Geometric Interpretation: Columns of $\mathbf{R}$ are the world axes expressed in camera coordinates.

Why Homogeneous Coordinates?

Definition

A system of coordinates used in projective geometry where a point in n-dimensional space is represented by a vector of $n+1$ components. For a 2D point (x, y), the homogeneous representation is (wx, wy, w) for any $w \neq 0$.

- **Linearization:** Perspective projection is non-linear in Euclidean space due to division by Z. Homogeneous coordinates allow us to represent this as a linear matrix multiplication.
- **Unified Framework:** Translation, rotation, and scaling can all be combined into a single 4×4 matrix operation.
- **Points at Infinity:** Allows for the mathematical representation of vanishing points where $w = 0$.

Why Homogeneous Coordinates?

- **Unify linear and non-linear transformations**
- **Handle points at infinity** (ideal points)
- **Simplify perspective projection** to matrix multiplication
- **Enable projective geometry** framework

Definition: For $\mathbb{R}^n$, homogeneous coordinates in $\mathbb{P}^n$:

$$(x_1, x_2, \ldots, x_n) \mapsto (x_1, x_2, \ldots, x_n, 1)$$

with equivalence relation $(x_1, \ldots, x_{n+1}) \sim \lambda(x_1, \ldots, x_{n+1})$ for $\lambda \neq 0$.

Homogeneous Coordinates: Examples

Example (3D Point)

Euclidean: (X, Y, Z)
Homogeneous: $(X, Y, Z, 1)$ or $(2X, 2Y, 2Z, 2)$

Example (2D Image Point)

Euclidean: (u, v)
Homogeneous: $(u, v, 1)$

Example (Point at Infinity)

Direction vector (a, b, c) in 3D:
Homogeneous: $(a, b, c, 0)$

Conversion back: $(x, y, z, w) \mapsto (x/w, y/w, z/w)$ for $w \neq 0$

Projective Transformations

In homogeneous coordinates, all transformations become matrix multiplications:

- **Rigid transformation:**

$$\begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix}$$

- **Perspective projection:**

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

- **Affine transformation:**

$$\begin{bmatrix} \mathbf{A} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix}$$

Intrinsic Matrix Components

The camera intrinsic matrix $\mathbf{K}$:

$$\mathbf{K} = \begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

Parameters:

- f_x, f_y: Focal length in pixels ($f_x = f \cdot m_x$, where m_x pixels/mm)
- c_x, c_y: Principal point (image center, ideally)
- s: Skew coefficient (usually 0 for modern cameras)

Physical Meaning: Maps metric image coordinates (x, y) to pixel coordinates (u, v):

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{K} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Focal Length Interpretation

- **Physical focal length** f: Distance from lens to image plane
- **Pixel focal lengths** f_x, f_y:

$$f_x = \frac{f \cdot \text{width in pixels}}{\text{sensor width in mm}}$$

$$f_y = \frac{f \cdot \text{height in pixels}}{\text{sensor height in mm}}$$

- Different f_x and f_y handle non-square pixels
- Field of View (FOV):

$$\text{FOV}_x = 2\arctan\left(\frac{\text{width}}{2f_x}\right)$$

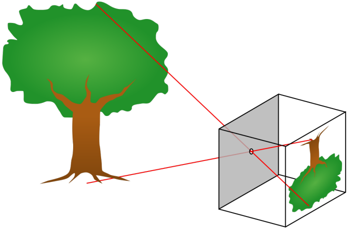


Figure: Pinhole camera model [Hartley and Zisserman, 2004]

Using similar triangles:

$$\frac{x}{f} = \frac{X_c}{Z_c} \Rightarrow x = f\frac{X_c}{Z_c}$$

$$\frac{y}{f} = \frac{Y_c}{Z_c} \Rightarrow y = f\frac{Y_c}{Z_c}$$

In homogeneous coordinates:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \sim \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix}$$

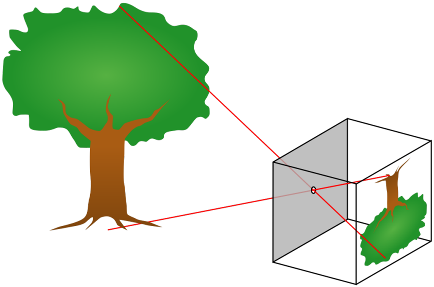


Figure: Perspective projection geometry [Szeliski, 2011]

Complete Projection Pipeline

The full mapping from 3D world to 2D pixels:

1. **World to Camera:** Rigid transformation

$$\mathbf{X}_c = \mathbf{R}\mathbf{X}_w + \mathbf{t}$$

2. **Camera to Image:** Perspective projection

$$\mathbf{x} = \frac{1}{Z_c}\begin{bmatrix} X_c \\ Y_c \end{bmatrix}$$

3. **Image to Pixels:** Intrinsic transformation

$$\mathbf{u} = \mathbf{K}\tilde{\mathbf{x}}$$

Combined:

$$\tilde{\mathbf{u}} \sim \underbrace{\mathbf{K}}_{3\times 3}\underbrace{[\mathbf{R}|\mathbf{t}]}_{3\times 4}\tilde{\mathbf{X}}_w$$

Real Cameras vs. Ideal Pinhole

The pinhole model assumes perfect lenses, but real cameras have distortion:

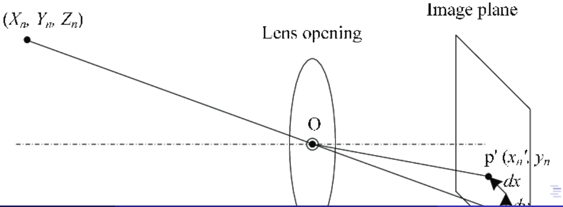
- **Radial distortion:** "Barrel" or "pincushion" effects

$$x_d = x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)$$
$$y_d = y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)$$

  where $r^2 = x^2 + y^2$

- **Tangential distortion:** Lens misalignment

$$x_d = x + [2p_1 xy + p_2(r^2 + 2x^2)]$$
$$y_d = y + [p_1(r^2 + 2y^2) + 2p_2 xy]$$



Camera Calibration Process

Goal: Estimate $\mathbf{K}$, distortion coefficients, and optionally $\mathbf{R}, \mathbf{t}$

Zhang's Method [Zhang, 2000]:

1. Capture images of a calibration pattern (checkerboard)
2. Detect corner points in images
3. Solve for homographies between pattern and images
4. Estimate intrinsic parameters from homographies
5. Refine all parameters using non-linear optimization

Output:

- Intrinsic matrix $\mathbf{K}$
- Distortion coefficients k_1, k_2, k_3, p_1, p_2
- For each image: rotation $\mathbf{R}_i$, translation $\mathbf{t}_i$

Example 1: Camera with Different Focal Lengths

Consider two cameras with same principal point $(320, 240)$ but different focal lengths:

- **Camera A:** $f_x = f_y = 500$ pixels
- **Camera B:** $f_x = f_y = 1000$ pixels

A point at $\mathbf{X}_c = (0.1, 0.2, 1.0)$ meters projects to:

- **Camera A:** $\mathbf{u} = (320 + 500 \cdot 0.1, 240 + 500 \cdot 0.2) = (370, 340)$
- **Camera B:** $\mathbf{u} = (320 + 1000 \cdot 0.1, 240 + 1000 \cdot 0.2) = (420, 440)$

Observation: Longer focal length = greater "zoom" = larger projection for same 3D point.

Example 2: Camera Rotation Effect

Camera at origin, rotated 30° around Y-axis:

$$\mathbf{R} = \begin{bmatrix} \cos 30^\circ & 0 & \sin 30^\circ \\ 0 & 1 & 0 \\ -\sin 30^\circ & 0 & \cos 30^\circ \end{bmatrix} = \begin{bmatrix} 0.866 & 0 & 0.5 \\ 0 & 1 & 0 \\ -0.5 & 0 & 0.866 \end{bmatrix}$$

World point $\mathbf{X}_w = (1, 0, 2)$:

- **Before rotation:** $\mathbf{X}_c = (1, 0, 2)$
- **After rotation:** $\mathbf{X}_c = \mathbf{R}\mathbf{X}_w = (0.866 \cdot 1 + 0.5 \cdot 2, 0, -0.5 \cdot 1 + 0.866 \cdot 2) = (1.866, 0, 1.232)$

Observation: Rotation changes which parts of scene are visible.

Example 3: Depth Ambiguity in Projection

The fundamental limitation: Multiple 3D points project to same 2D point. Consider pinhole projection:

$$(x, y) = \left(f\frac{X}{Z}, f\frac{Y}{Z}\right)$$

For any $\lambda > 0$:

$$(\lambda X, \lambda Y, \lambda Z) \mapsto \left(f\frac{\lambda X}{\lambda Z}, f\frac{\lambda Y}{\lambda Z}\right) = (x, y)$$

Consequence: Cannot recover absolute scale from single image. Need stereo, motion, or prior knowledge.

Example 4: Principal Point Offset Effect

Camera with $f_x = f_y = 800$, but different principal points:

- **Camera A:** $(c_x, c_y) = (320, 240)$
- **Camera B:** $(c_x, c_y) = (640, 480)$

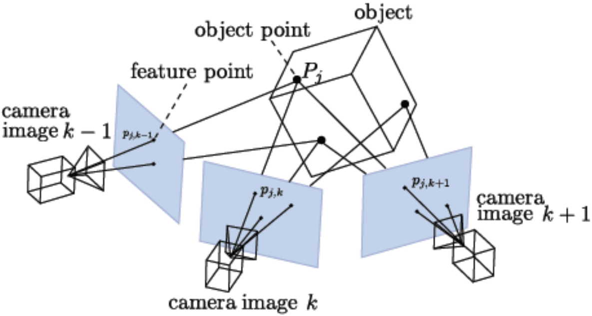
Same point $\mathbf{x} = (0.1, 0.2)$ in metric coordinates projects to:

- **Camera A:** $(320 + 800 \cdot 0.1, 240 + 800 \cdot 0.2) = (400, 400)$
- **Camera B:** $(640 + 800 \cdot 0.1, 480 + 800 \cdot 0.2) = (720, 640)$

Observation: Principal point shift translates the entire image.

Applications of Camera Models

- **Structure from Motion (SfM):** Reconstruct 3D scene from multiple images
- **Stereo Vision:** Compute depth from two cameras
- **Augmented Reality:** Overlay virtual objects in real scenes
- **Photogrammetry:** Create 3D models from photographs
- **Visual SLAM:** Simultaneous localization and mapping
- **Camera Tracking:** Match virtual and real camera motions



Conceptual Questions

1. Why do we need separate intrinsic and extrinsic parameters? Could we combine them into one matrix?
2. What happens when $Z_c \to 0$ in perspective projection? How does this relate to the "bas relief ambiguity" in 3D reconstruction?
3. How many degrees of freedom does a camera projection matrix have? (Hint: Consider scale ambiguity and constraints)
4. Why is camera calibration necessary for metric reconstruction but not for projective reconstruction?
5. How does the field of view relate to focal length and sensor size? Give a practical example.

Mathematical Exercises

1. Given $\mathbf{K} = \begin{bmatrix} 800 & 0 & 320 \\ 0 & 800 & 240 \\ 0 & 0 & 1 \end{bmatrix}$ and $\mathbf{R} = \mathbf{I}, \mathbf{t} = (0, 0, 0.5)^T$, where does $\mathbf{X}_w = (1, 2, 3)^T$ project?
2. Derive the formula for field of view in terms of focal length and sensor size.
3. Show that the cross product of two image points $\mathbf{x}_1 \times \mathbf{x}_2$ gives the line passing through them in homogeneous coordinates.
4. Prove that for a rotation matrix, $\|\mathbf{R}\mathbf{v}\| = \|\mathbf{v}\|$ for any vector $\mathbf{v}$.

References I

Duane C Brown. Close-range camera calibration. *Photogrammetric Engineering*, 37(8): 855–866, 1971.

Andrew J Davison, Ian D Reid, Nicholas D Molton, and Olivier Stasse. Monoslam: Real-time single camera slam. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(6):1052–1067, 2007.

Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2nd edition, 2004.

Noah Snavely, Steven M Seitz, and Richard Szeliski. Photo tourism: Exploring photo collections in 3d. In *ACM SIGGRAPH 2006 Papers*, pages 835–846, 2006.

Richard Szeliski. *Computer Vision: Algorithms and Applications*. Springer, 2011.

Zhengyou Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, 2000.

- OpenCV Camera Calibration Tutorial:
  `https://docs.opencv.org/master/dc/dbb/tutorial_py_calibration.html`
- MATLAB Camera Calibration Toolbox:
  `https://www.vision.caltech.edu/bouguetj/calib_doc/`
- Visual Geometry Group, University of Oxford:
  `http://www.robots.ox.ac.uk/~vgg/hzbook/`