Humanoid Robotics

Perception 1: Basics

Maren Bennewitz

UNIVERSITÄT BONN

Goal of This Chapter

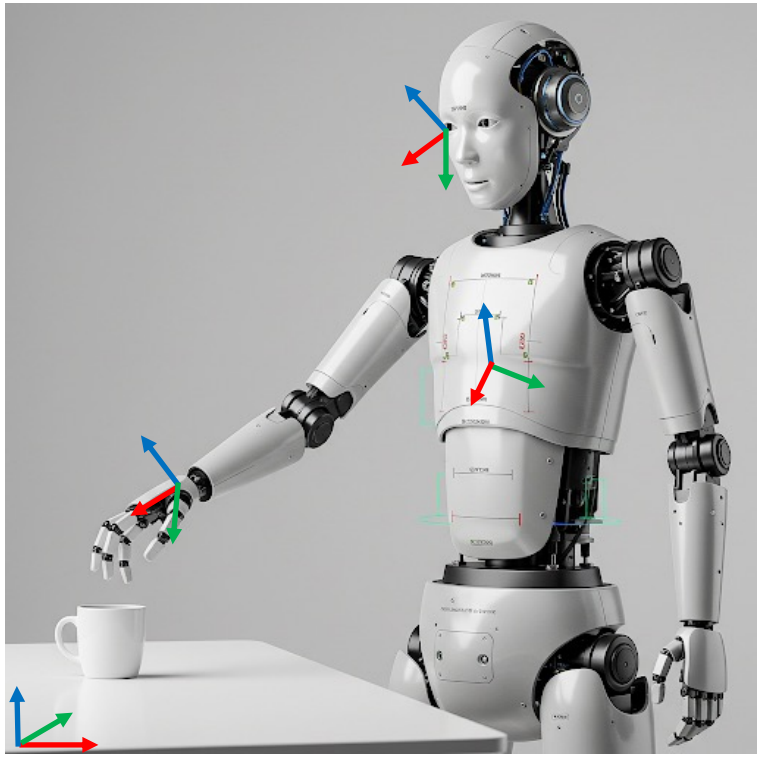
- Understand the concept of **transformations between coordinate systems** and why they are needed
- Learn a representation for transformations that allows for **easy chaining** and **inversion** of transformations
- Understand the mapping from the **world coordinate system** to the **robot's sensor coordinate system**

Why are Coordinate Transforms Needed

- **Accurate perception, motion planning, and control**
- Data from sensors (e.g., cameras, IMUs, joint encoders) are **captured in different reference frames**
- Motion planning (e.g., for object manipulation and navigation) requires **transforming between global, local, and joint-specific coordinates**

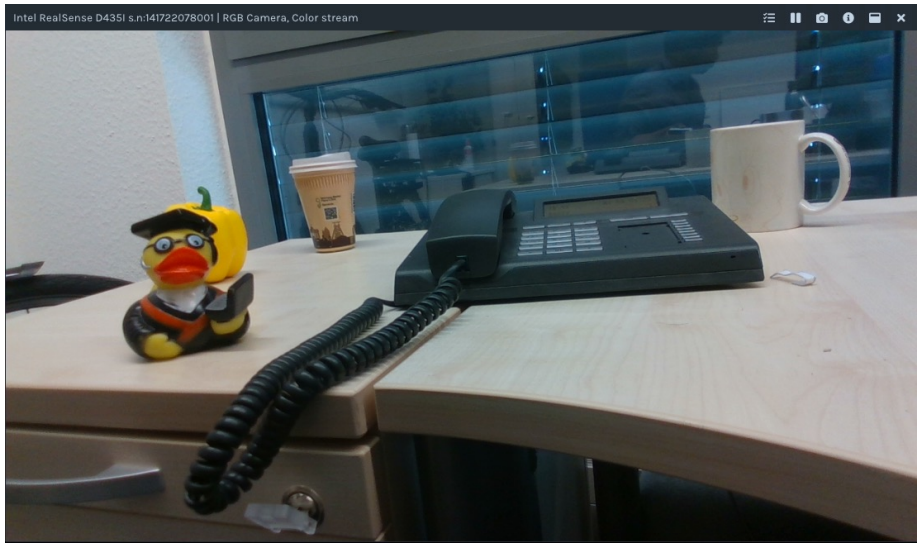
Example: Reaching for an Object

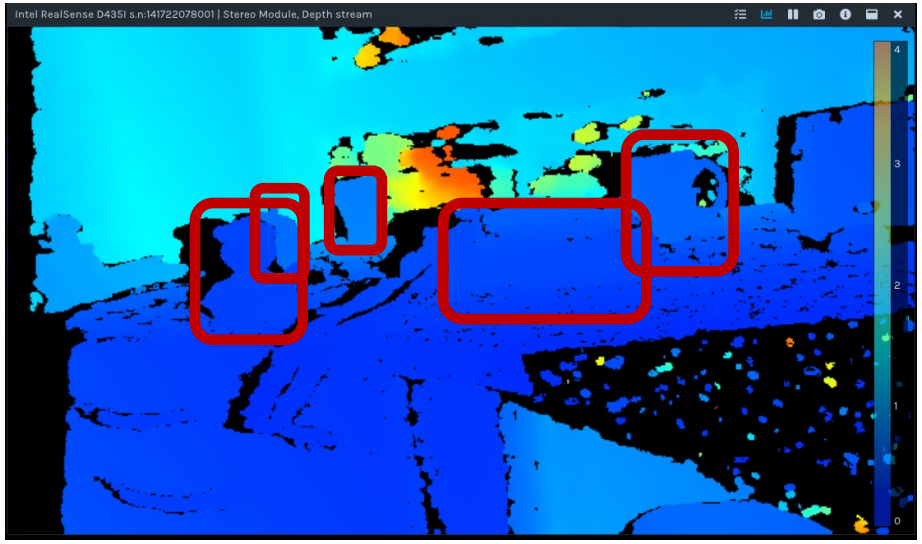
- **World Frame:** Global origin
- **Camera Frame:**
  Coordinates of the camera in the world, needed for accurate perception
- **Body Frame:**
  Coordinates of the robot's torso, needed for motion planning
- **End-Effector Frame:**
  Coordinates of the hand, ensures the robot's hand precisely aligns with object positions



generated by Gemini

Depth Image of RGB-D Camera

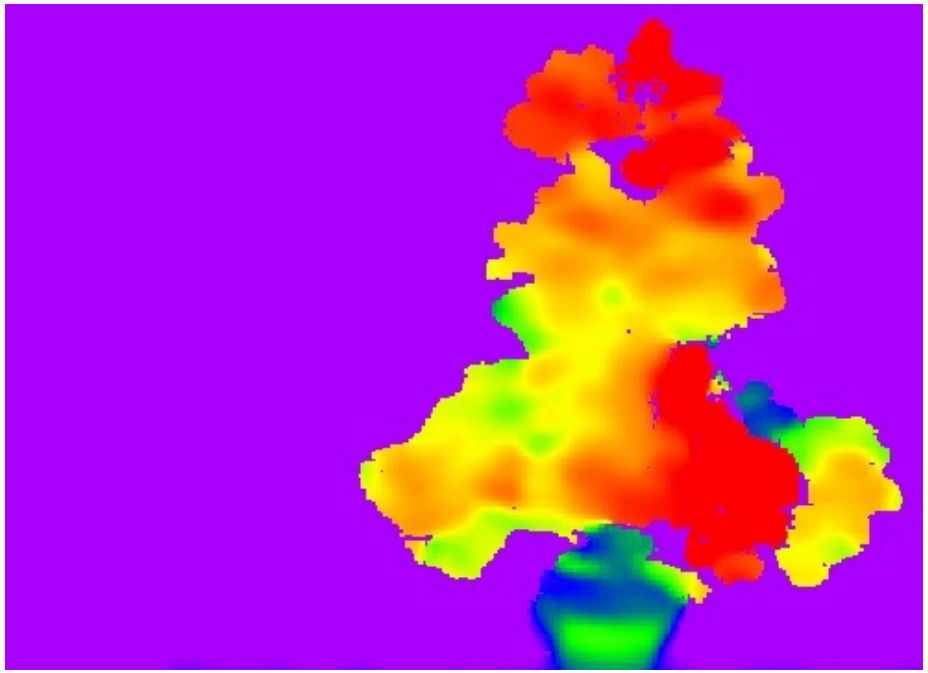




Intel Realsense D435i

Image courtesy: Intel

Example Scene – Depth Data



close, medium, more distant, not visible

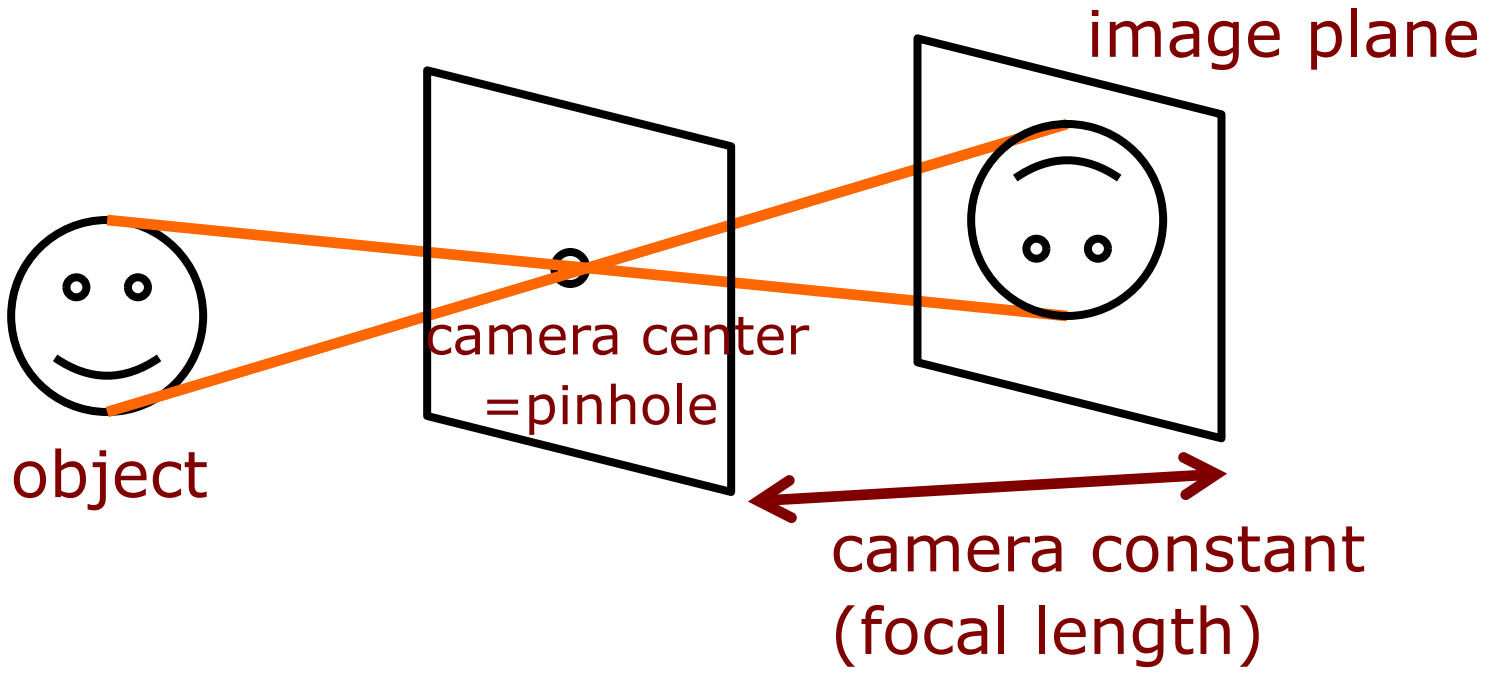
How to Measure Depth

- **Time-of-flight principle**: send laser pulses and measure their return time
- **Structured light**: projects a pattern onto the environment and uses its deformation resulting from hitting objects
- **Stereo vision**: comparing the disparity between objects in the two images

Pinhole Camera Model

- Describes the **projection of a 3D world point** into the camera image
- Assumption: box with an **infinitesimal small hole**
- **Camera center:** Intersection point of the rays, i.e., the pinhole
- **Image plane:** Back wall of box
- **Camera constant:** Distance between the camera center and image plane

Pinhole Camera Model



Projective Mapping

- Straight lines stay straight
- Parallel lines may intersect

Image courtesy: W. Förstner

Pinhole Camera Model: Properties

- **Line-preserving:** straight lines are mapped to straight lines
- **Not angle-preserving:** angles between lines change
- **Not length-preserving:** size of objects is inverse proportional to the distance

Homogeneous Coordinates (H.C.)

- Cameras generate a **projected image** of the **3D world**
- In Euclidian geometry, the math for describing this transformation gets difficult
- So-called **homogeneous coordinates** are typically used in robotics for transformations
- Advantage: **Single matrix** can represent affine and projective transformations

Notation

Point

- in homogeneous coordinates $\mathbf{x}$
- in Euclidian coordinates $\boldsymbol{x}$

2D vs. **3D** space

- lowercase = 2D
- capitalized = 3D

Homogeneous Coordinates

Definition:

The representation $\mathbf{x}$ of a geometric object is **homogeneous** if $\mathbf{x}$ and $\lambda\mathbf{x}$ represent the same object for $\lambda \neq 0$

Example:

$$\mathbf{x} = \lambda\,\mathbf{x}$$

homogeneous

$$\boldsymbol{x} \neq \lambda\,\boldsymbol{x}$$

Euclidian

Homogeneous Coordinates

- H.C. use n+1 dimensional vectors to represent n-dimensional points given in Euclidean coordinates
- Example:

Euclidian → **homogeneous**

$$\boldsymbol{x} = \begin{bmatrix} x \\ y \end{bmatrix} \quad \Rightarrow \quad \mathbf{x} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

From Homogeneous to Euclidian Coordinates

homogeneous **Euclidian**

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} u/w \\ v/w \\ 1 \end{bmatrix} \Rightarrow \begin{bmatrix} u/w \\ v/w \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix}$$

Since

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

Definition

- **Homogeneous coordinates** of a point χ in $\mathbb{R}^2$ is a 3-dimensional vector

$$\chi: \quad \mathbf{x} = \begin{bmatrix} u \\ v \\ w \end{bmatrix} \text{ with } |\mathbf{x}|^2 = u^2 + v^2 + w^2 \neq 0$$

- Corresponding **Euclidian coordinates**

$$\chi: \quad \boldsymbol{x} = \begin{bmatrix} u/w \\ v/w \end{bmatrix} \text{ with } w \neq 0$$

3D Points

homogeneous **Euclidian**

$$\mathbf{x} = \begin{bmatrix} U \\ V \\ W \\ T \end{bmatrix} = \begin{bmatrix} U/T \\ V/T \\ W/T \\ 1 \end{bmatrix} \rightarrow \begin{bmatrix} U/T \\ V/T \\ W/T \end{bmatrix}$$

Origin of the Euclidian Coordinate System in Homogeneous Coordinates

$$\mathbf{O}_2 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \qquad \mathbf{O}_3 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

3D Transformations With Homogenous Transformation Matrices

- Linear mapping

$$\mathbf{X}' = \mathrm{H}\mathbf{X}$$

- 3D translation: 3 parameters

$$\mathrm{H} = \begin{bmatrix} \boldsymbol{I} & \boldsymbol{t} \\ \mathbf{0}^{\mathsf{T}} & 1 \end{bmatrix}$$

$$\boldsymbol{I} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\boldsymbol{t} = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}$$

$$\mathbf{0} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

Rotation Matrices

- 2D:

$$R^{2D}(\theta) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}$$

- 3D:

$$R_x^{3D}(\omega) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\omega) & -\sin(\omega) \\ 0 & \sin(\omega) & \cos(\omega) \end{bmatrix} \quad R_y^{3D}(\phi) = \begin{bmatrix} \cos(\phi) & 0 & \sin(\phi) \\ 0 & 1 & 0 \\ -\sin(\phi) & 0 & \cos(\phi) \end{bmatrix}$$

$$R_z^{3D}(\kappa) = \begin{bmatrix} \cos(\kappa) & -\sin(\kappa) & 0 \\ \sin(\kappa) & \cos(\kappa) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$R^{3D}(\omega, \phi, \kappa) = R_z^{3D}(\kappa) R_y^{3D}(\phi) R_x^{3D}(\omega)$$

3D Transformations With Homogenous Transformation Matrices

- Rigid body transformation: 6 parameters
- 3 translation + 3 rotation

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

Easy Inversion and Chaining

- Inverting a transformation

$$\begin{aligned} \mathbf{X}' &= \mathrm{H}\mathbf{X} \\ \mathbf{X} &= \mathrm{H}^{-1}\mathbf{X}' \end{aligned}$$

- Chaining of transformations via matrix products (not commutative)

$$\begin{aligned} \mathbf{X}' &= \mathrm{H}_1\mathrm{H}_2\mathbf{X} \\ &\neq H_2\mathrm{H}_1\mathbf{X} \end{aligned}$$

Coordinate Frames for Projective Mapping

1. **World** coordinate frame
2. **Camera** coordinate frame
3. **Image** coordinate frame
4. **Sensor** coordinate frame

Coordinate Frames for Projective Mapping

1. **World** coordinate frame S_o
   written as: $[X, Y, Z]^{\mathsf{T}}$
2. **Camera** coordinate frame S_k
   written as: $[\,{}^{k}X,\ {}^{k}Y,\ {}^{k}Z]^{\mathsf{T}}$
3. **Image** coordinate frame S_c
   written as: $[\,{}^{c}x,\ {}^{c}y]^{\mathsf{T}}$
4. **Sensor** coordinate frame S_s
   written as: $[\,{}^{s}x,\ {}^{s}y]^{\mathsf{T}}$

Visualization of the Transformation

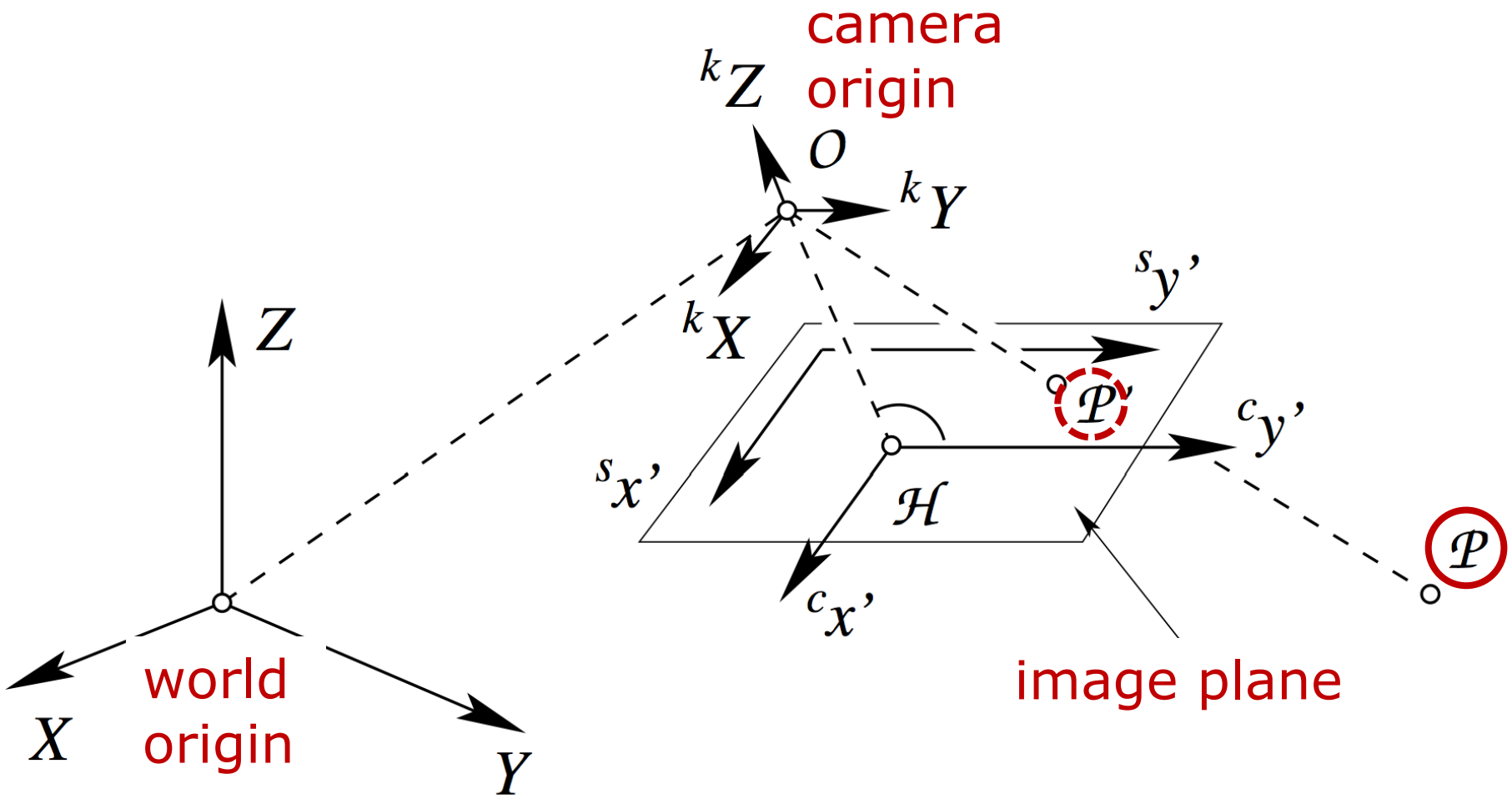
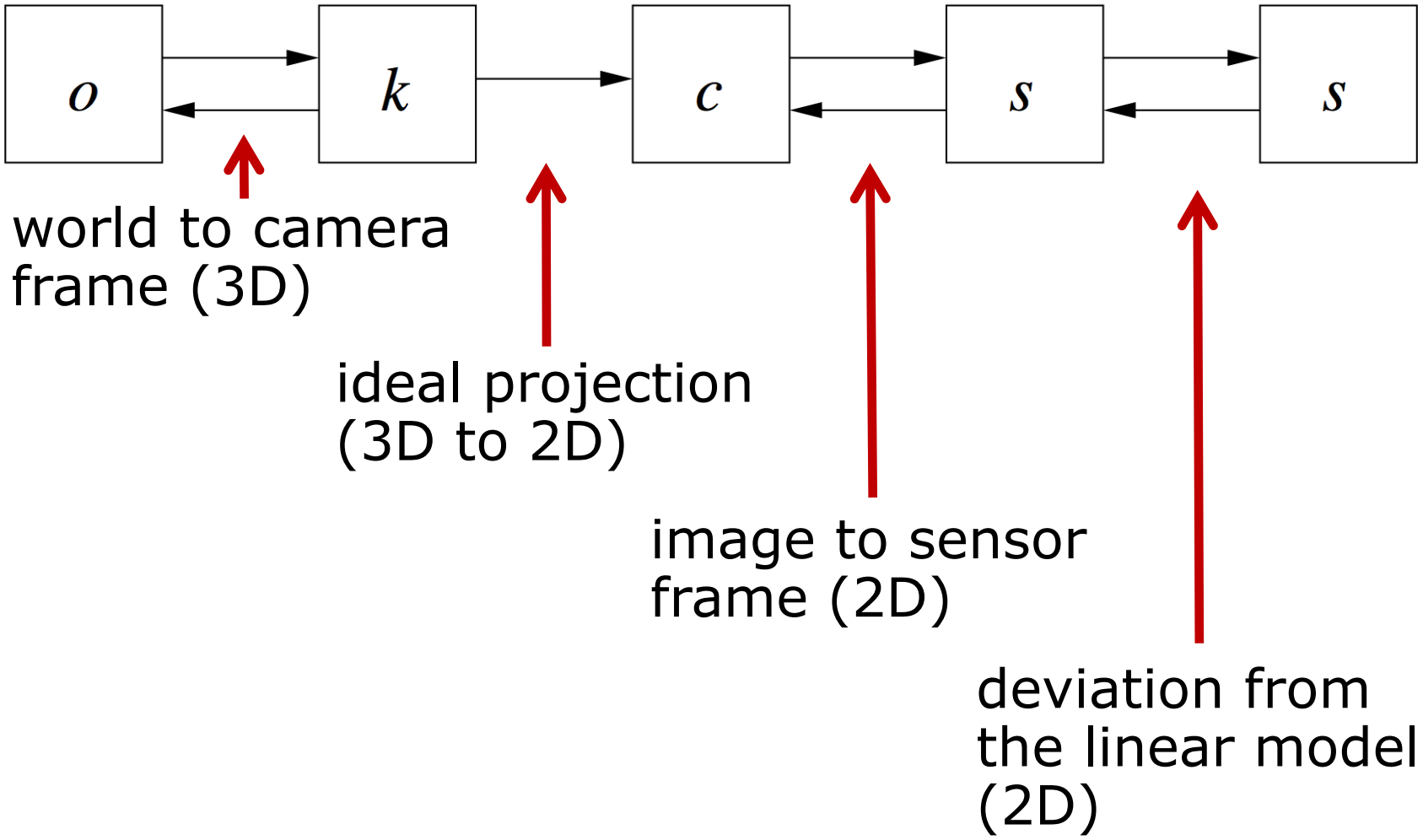
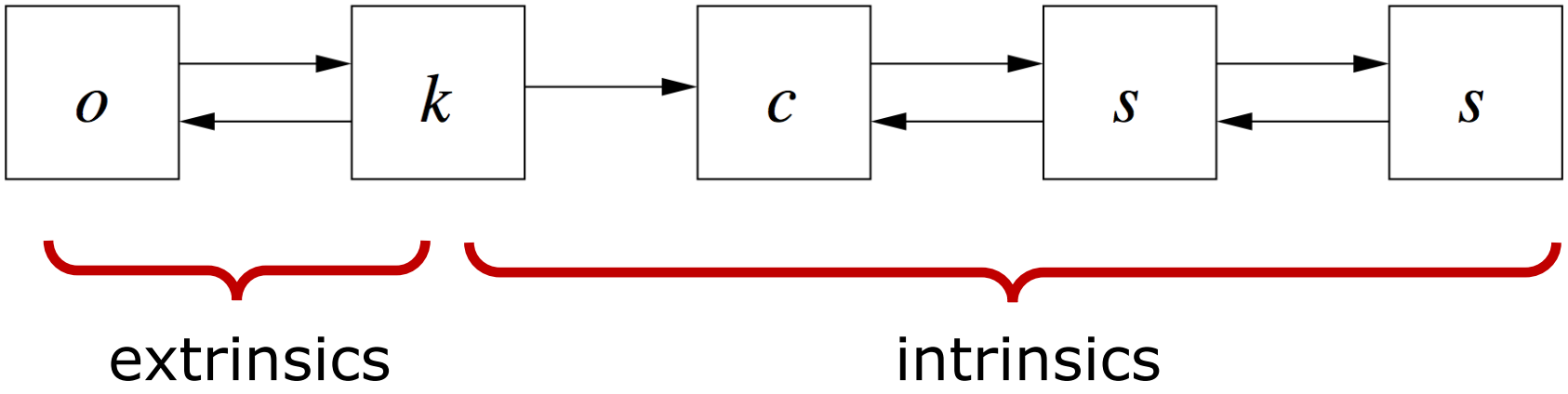


Image courtesy: Förstner

From the World to the Sensor



Extrinsic and Intrinsic Parameters



- **Extrinsic parameters** describe the pose of the camera in the world
- **Intrinsic parameters** describe the mapping of the scene in front of the camera to the pixels in the final image (sensor)

Extrinsic Parameters

- Define the pose of the camera with respect to the world
- Describe **transformation** between **world** and **camera** frame
- **6 parameters**: 3 for the position and 3 for the orientation
- Define **invertible** transformation

3D Transformations With Homogenous Transformation Matrices

- Rotation: 3 parameters

$$\mathrm{H} = \begin{bmatrix} R & \mathbf{0} \\ \mathbf{0}^{\mathrm{T}} & 1 \end{bmatrix}$$

rotation matrix

Extrinsic Parameters for Transformations

- Point $\mathcal{P}$ with coordinates in **world coordinates**

$$\boldsymbol{X}_P = [X_P, Y_P, Z_P]^\mathsf{T}$$

- **Origin** of the **camera frame** (translational component)

$$\boldsymbol{X}_O = [X_O, Y_O, Z_O]^\mathsf{T}$$

- **Rotation** *R* from world to camera frame (orientational comp.)

- Wanted: Coordinates of $\mathcal{P}$ in the **camera frame**

Transformation: World to Camera Frame

- Single **transform** containing translation and rotation
- Yields H.C. of $\mathcal{P}$ in the **camera frame**

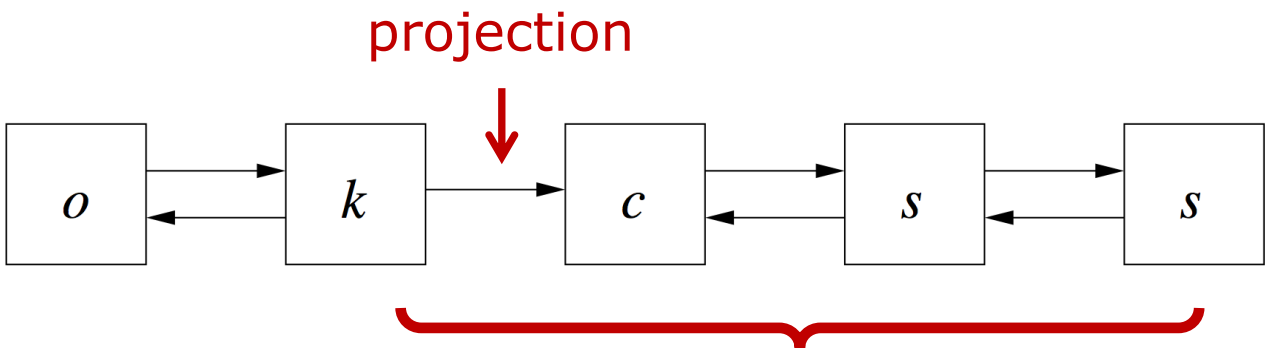
$$ {}^{k}\mathbf{X}_P = {}^{k}\mathrm{H}\,\mathbf{X}_P \quad \text{with} \quad {}^{k}\mathrm{H} = \begin{bmatrix} R & -R\boldsymbol{X}_O \\ \mathbf{0}^\mathsf{T} & 1 \end{bmatrix} $$

H.C. in camera frame

H.C. in world frame

Intrinsic Parameters

- For projecting points from the **camera frame (3D)** to the **sensor frame (2D)**
- Projection is only invertible using depth information
- Directly invertible transformations:
  - Image plane to sensor frame
  - Model deviations

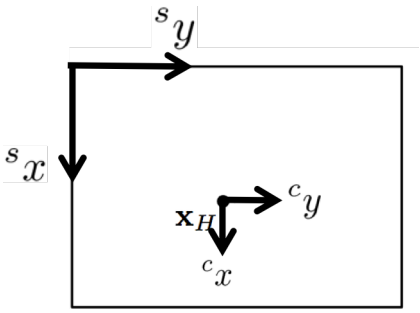


Mapping to Sensor Frame (w/o Derivation)

- Mapping to 2D sensor pixel coordinates using

$$\mathrm{K} = \begin{bmatrix} c & 0 & x_H \\ 0 & c(1+m) & y_H \\ 0 & 0 & 1 \end{bmatrix}$$

- **Calibration matrix** contains 4 parameters:
  - Camera constant: c (pinhole model)
  - Offset from image frame origin: x_H, y_H
  - Scale difference in x and y: m



Non-Linear Errors

- So far, we considered only **linear** parameters
- The real world is **non-linear**
- Reasons for non-linear errors
  - Imperfect lens
  - Non-planarity of the sensor
  - …

Example

not straight-line preserving

rectified image

image courtesy: Abraham

Mapping from Camera to Sensor Frame (2)

- Final transformation to cover non-linear effects
- **Location-dependent** shift in the sensor coordinate system
- Individual shift for each pixel according to the **distance from the image center**

$$ {}^{a}\mathrm{K}(\boldsymbol{x}, \boldsymbol{q}) = \begin{bmatrix} c & 0 & x_H + \Delta x(\boldsymbol{x}, \boldsymbol{q}) \\ 0 & c(1+m) & y_H + \Delta y(\boldsymbol{x}, \boldsymbol{q}) \\ 0 & 0 & 1 \end{bmatrix} $$

- q additional calibration parameter

Complete Transform from World to Sensor

Transform a **3D point in H.C. world coordinates** to **2D sensor pixel coordinates**

$$ {}^{a}\mathbf{x} = {}^{a}\mathrm{P}(\boldsymbol{x}, \boldsymbol{q}) = \underbrace{{}^{a}\mathrm{K}(\boldsymbol{x}, \boldsymbol{q})}_{\text{intrinsics}} \underbrace{R[I| - \boldsymbol{X}_O]}_{\text{extrinsics}} \mathbf{X} $$

Calibrated Camera

- If the intrinsics are **unknown**, the camera is called **”uncalibrated”**
- If the intrinsics are **known**, the camera is **”calibrated”**
- **Camera calibration:** Process of obtaining the intrinsic parameters
- **Complete calibration**: Fit calibration matrix and extrinsic parameters so that projected 3D points match observed pixels

Summary (1)

- Perception and motion planning requires **transforming between** global, sensor, and joint-specific coordinates
- **Homogeneous coordinates** are a representation for transformations
- Transformations as **one single matrix multiplication**
- Allow for **easy chaining** and **inversion** of transformations

Summary (2)

- Today: Mapping from the world frame to the sensor frame
- **Extrinsics** = world to camera frame
- **Intrinsics** = camera to sensor frame
- Assumption: **Pinhole camera model**
- Needed for pose estimation, scene rendering, …
- Mapping from the sensor frame to the world frame: reverse mapping, see exercise

Literature

- Multiple View Geometry in Computer Vision (Chs. 2, 3, 6)
  R. Hartley and A. Zisserman,