UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

> Reinventa el mundo <

Computer Graphics

Class 34. Introductory ideas in Computer Vision.

Professor: Eric Biagioli

TRANSFORMATEC

Today

- Camera models.
- Camera calibration.
- Edge detection.
- Extracting and matching keypoints.

References

- Camera model and calibration:
  - Kaehler, A., and Bradski, G. *Learning OpenCV 3*. O'Reilly Media, Inc., 2016. → **Chapter 18**
  - https://www.mathworks.com/help/vision/ug/camera-calibration.html
  - https://docs.opencv.org/4.x/dc/dbb/tutorial_py_calibration.html
- Border detection:
  - Kaehler, A., and Bradski, G. *Learning OpenCV 3*. O'Reilly Media, Inc., 2016. → **Section "The Canny Edge Detector" of Chapter 12**
  - https://www.geeksforgeeks.org/image-edge-detection-operators-in-digital-image-processing/
  - https://en.wikipedia.org/wiki/Edge_detection
  - https://www.mathworks.com/discovery/edge-detection.html
  - https://en.wikipedia.org/wiki/Canny_edge_detector
  - https://learnopencv.com/edge-detection-using-opencv/

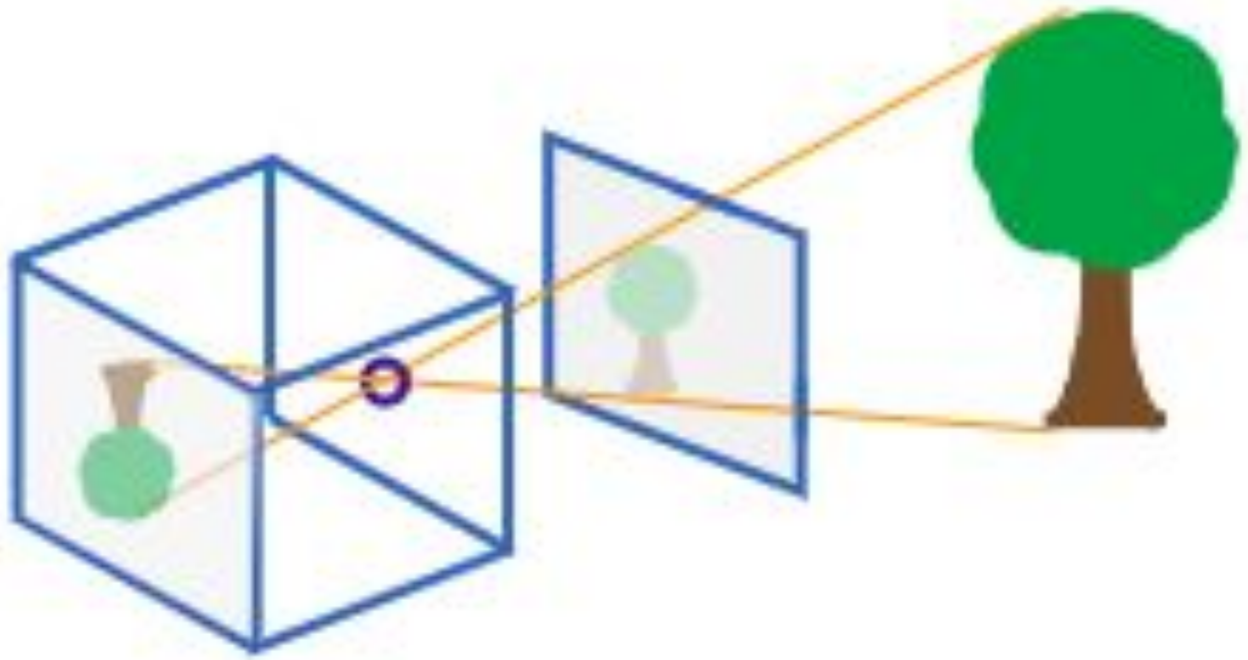
Today

- Camera model.
- Camera calibration.
- Edge detection.
- Extracting and matching features.

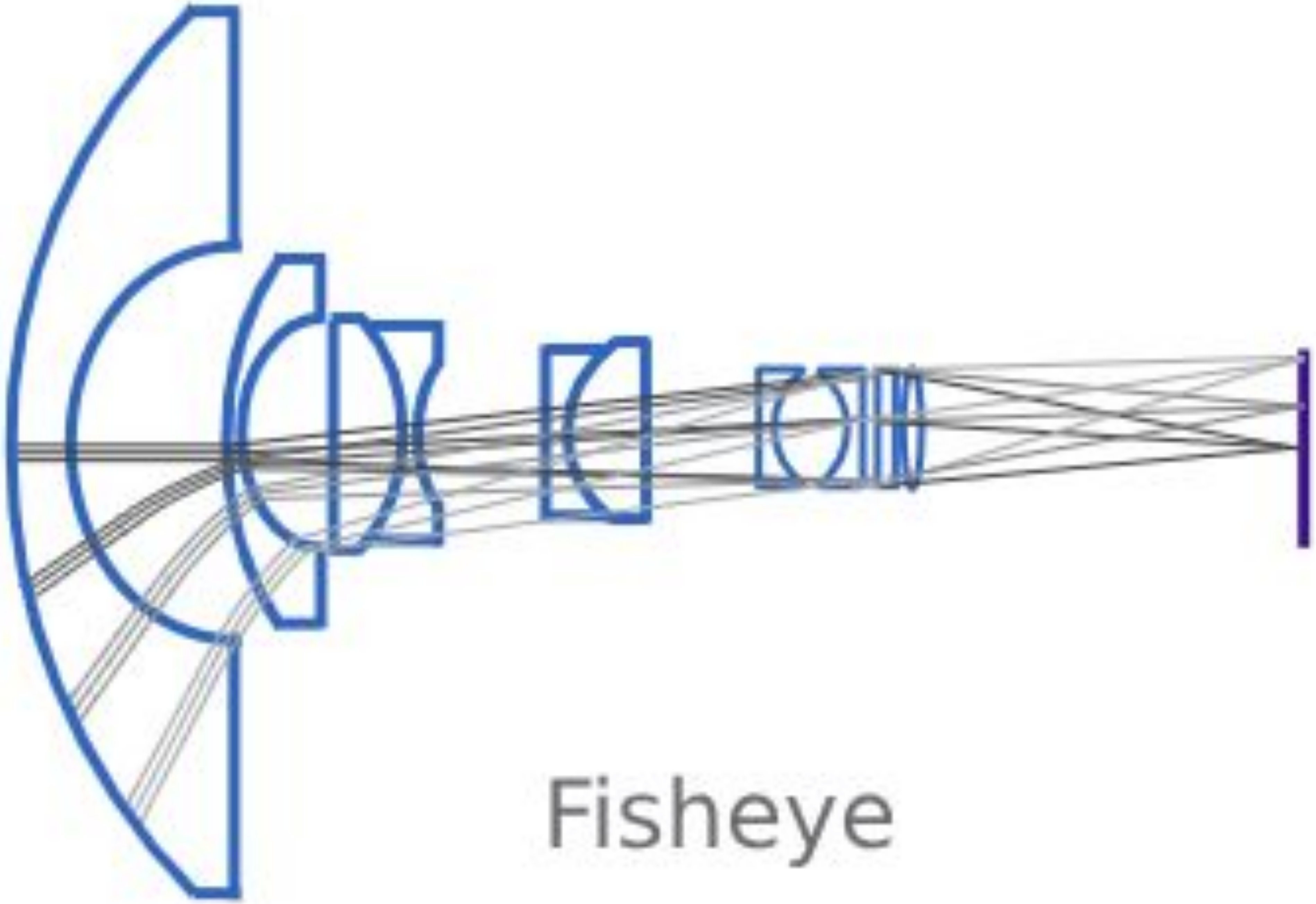
References (continuation)

- Extraction and matching of features:
  - https://www.geeksforgeeks.org/feature-detection-and-matching-with-opencv-python/
  - Kaehler, A., and Bradski, G. *Learning OpenCV 3*. O'Reilly Media, Inc., 2016. → **Chapter 16 (Keypoints and descriptors)**

Camera models

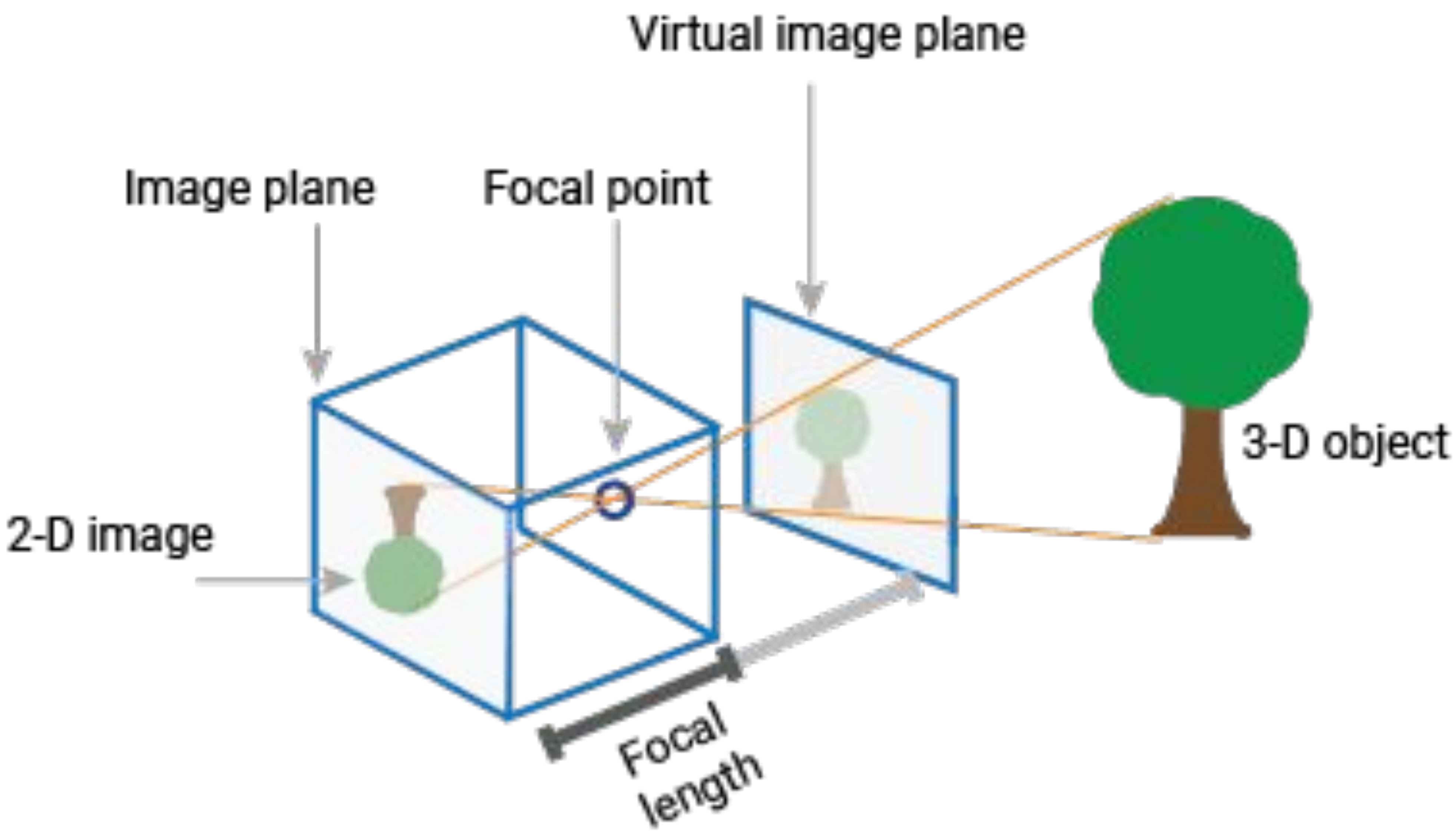


Pinhole



Fisheye

Camera models

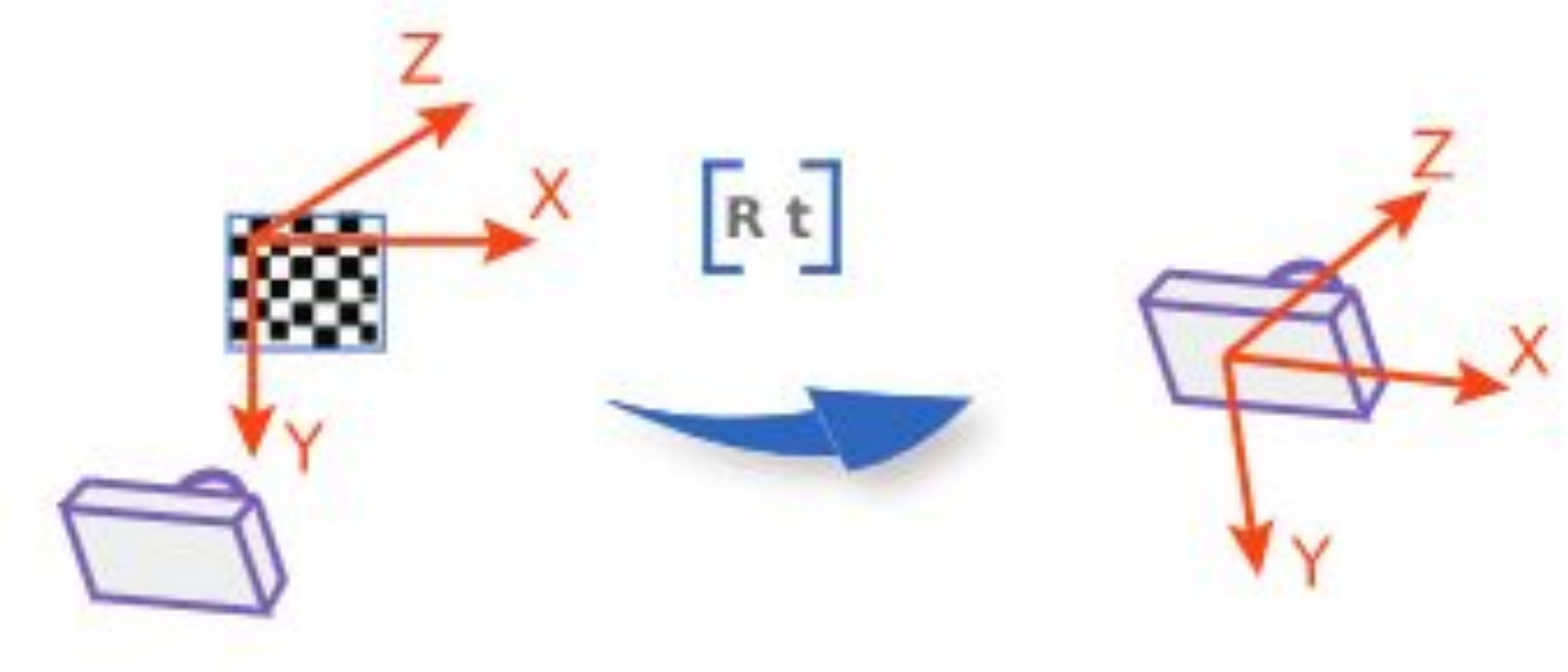


Extrinsic parameters

Intrinsic parameters

Camera models

Extrinsic parameters: rotation, *R*, and a translation, *t*.

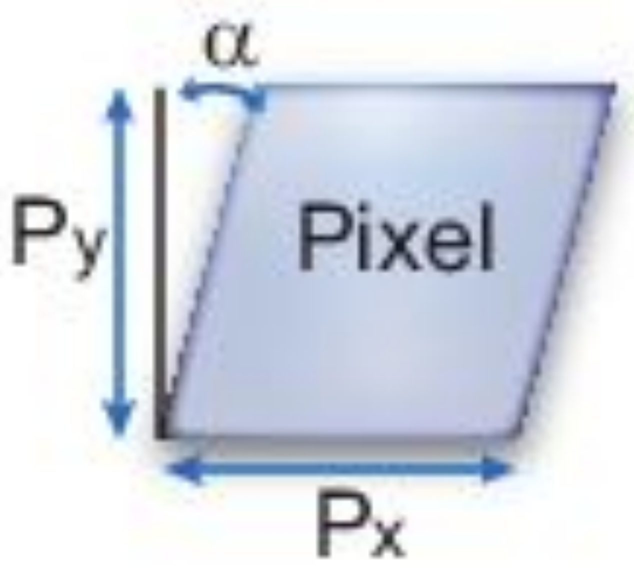


Camera models

Intrinsic parameters: focal length, the optical center, (aka *principal point*), and skew

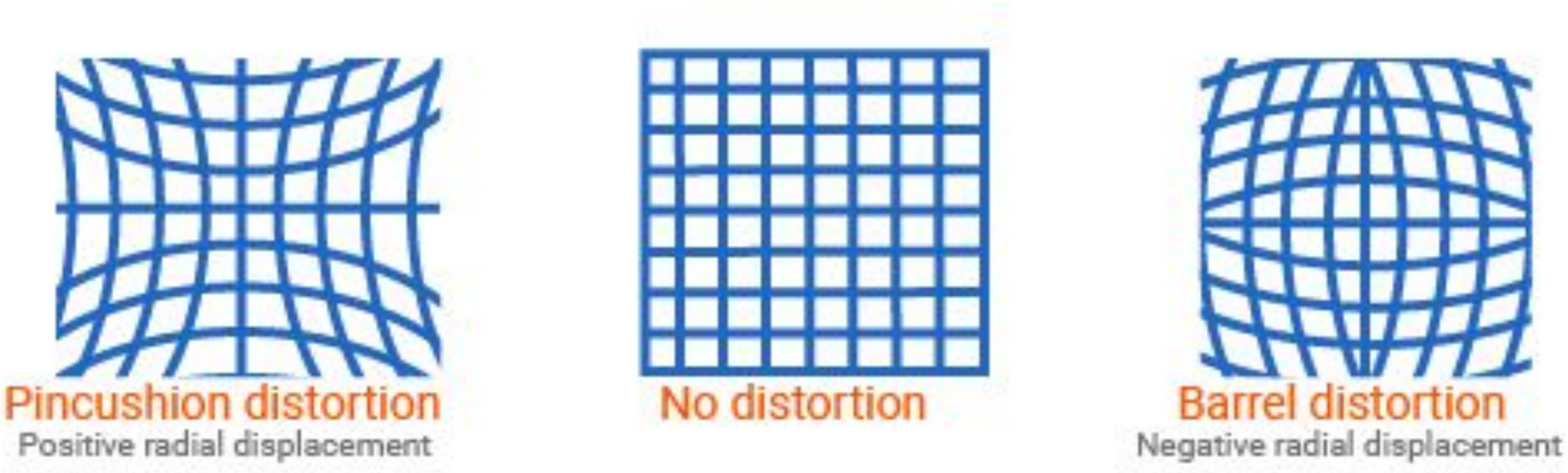
$$\begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

The pixel skew is defined as:

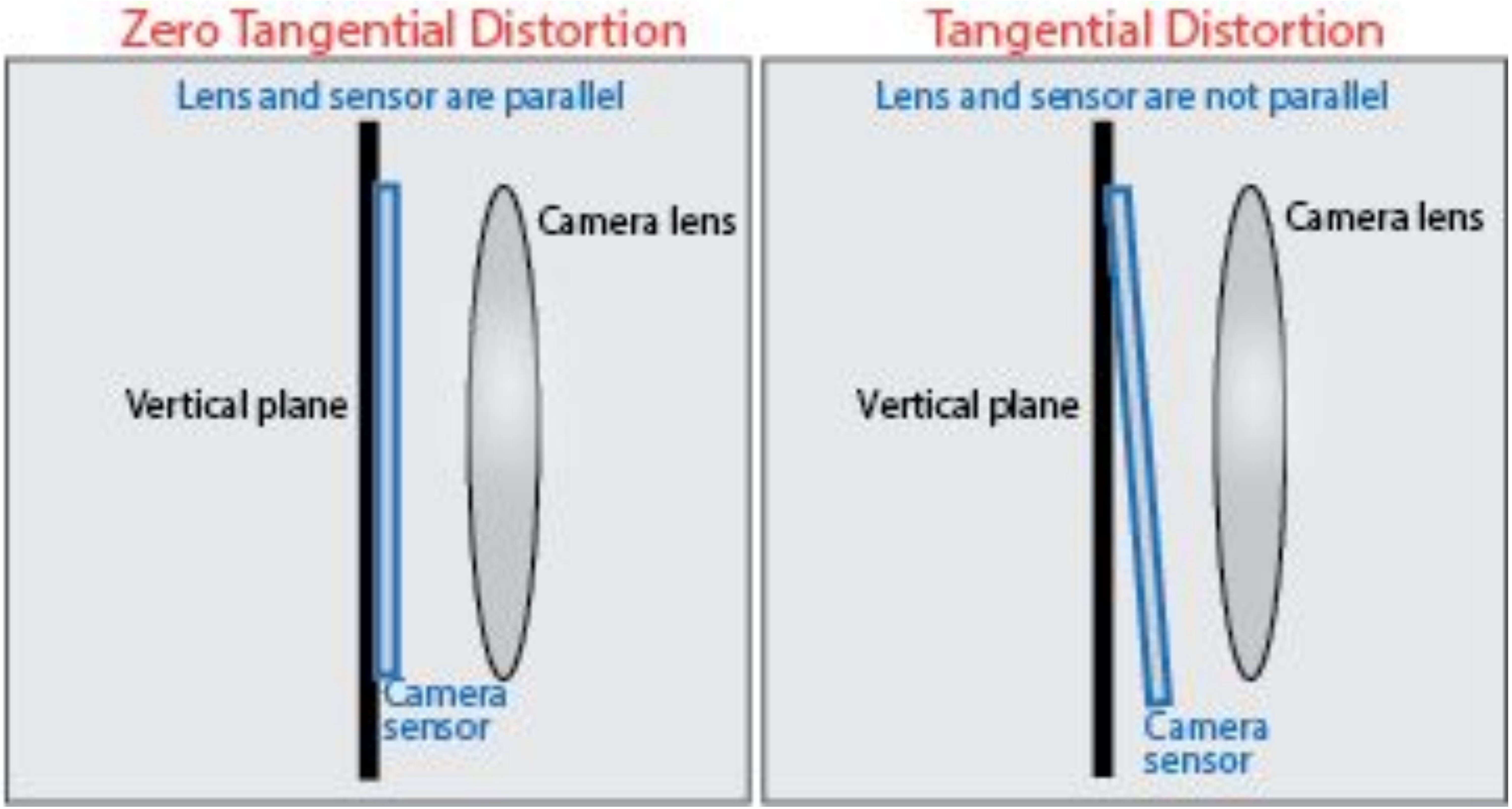


Skew

Radial distortion



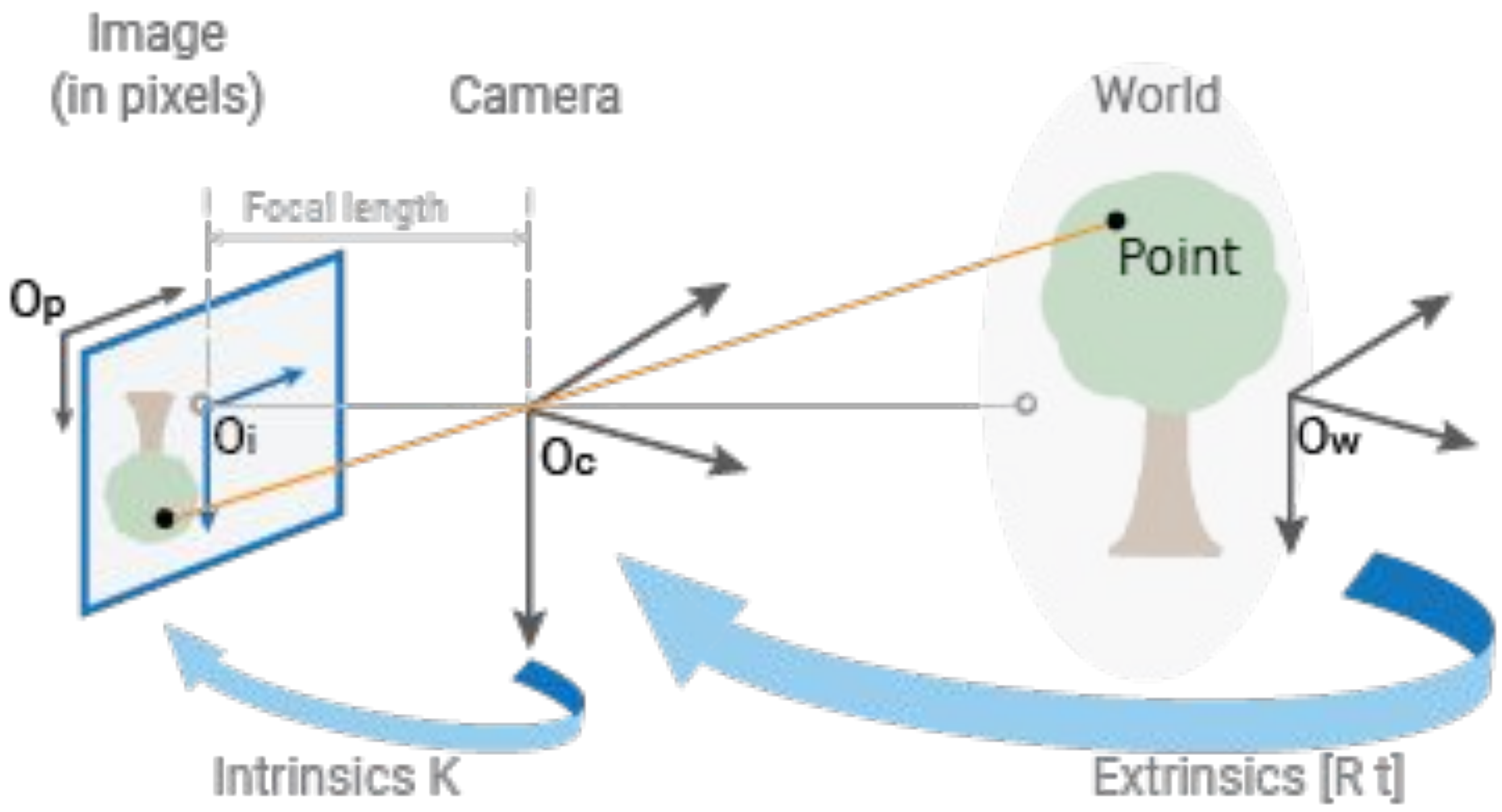
Radial distortion



Camera models

$$\underset{\text{Scale factor}}{w}\underbrace{\begin{bmatrix}x\\y\\1\end{bmatrix}}_{\text{Image points}} = P\underbrace{\begin{bmatrix}X\\Y\\Z\\1\end{bmatrix}}_{\text{World points}}$$

$$\underbrace{P}_{\text{Camera matrix}} = \underbrace{K}_{\text{Intrinsics matrix}}\underbrace{\begin{bmatrix}R & t\end{bmatrix}}_{\substack{\text{Extrinsics}\\ \text{Rotation and Translation}}}$$

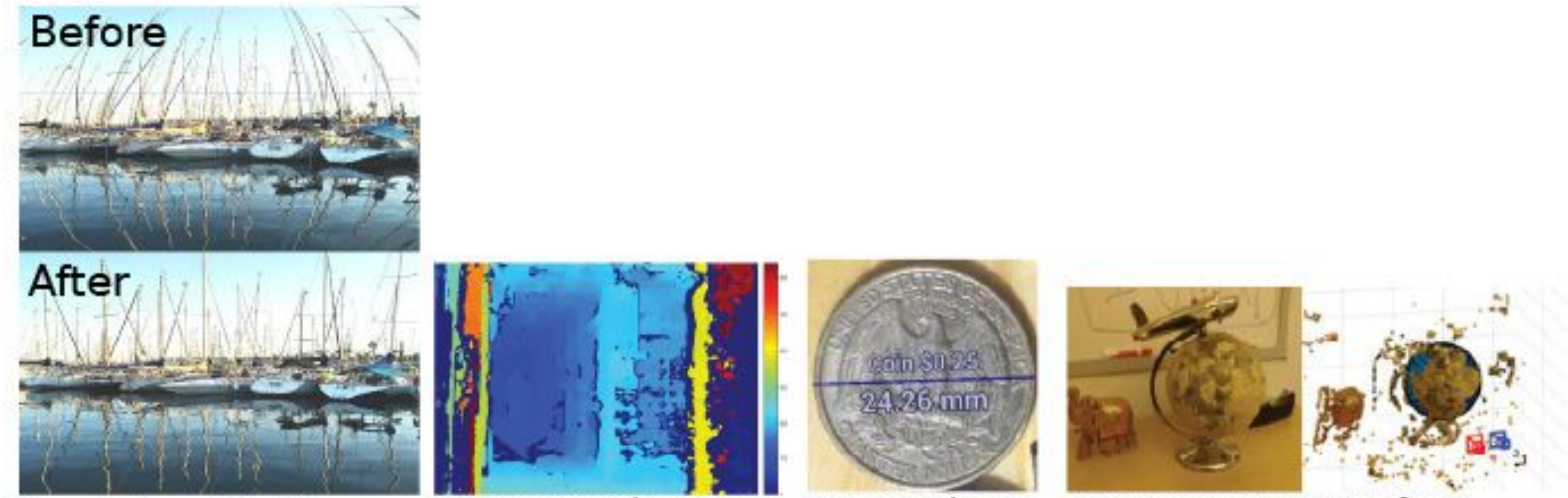


Camera calibration

- What is?
  - Compute the intrinsic and extrinsic parameters of the camera.
  - Goal: provide accurate relationship between real world coordinates (3D) and image coordinates (2D).
  - Often involves using known patterns to adjust the camera's geometry.

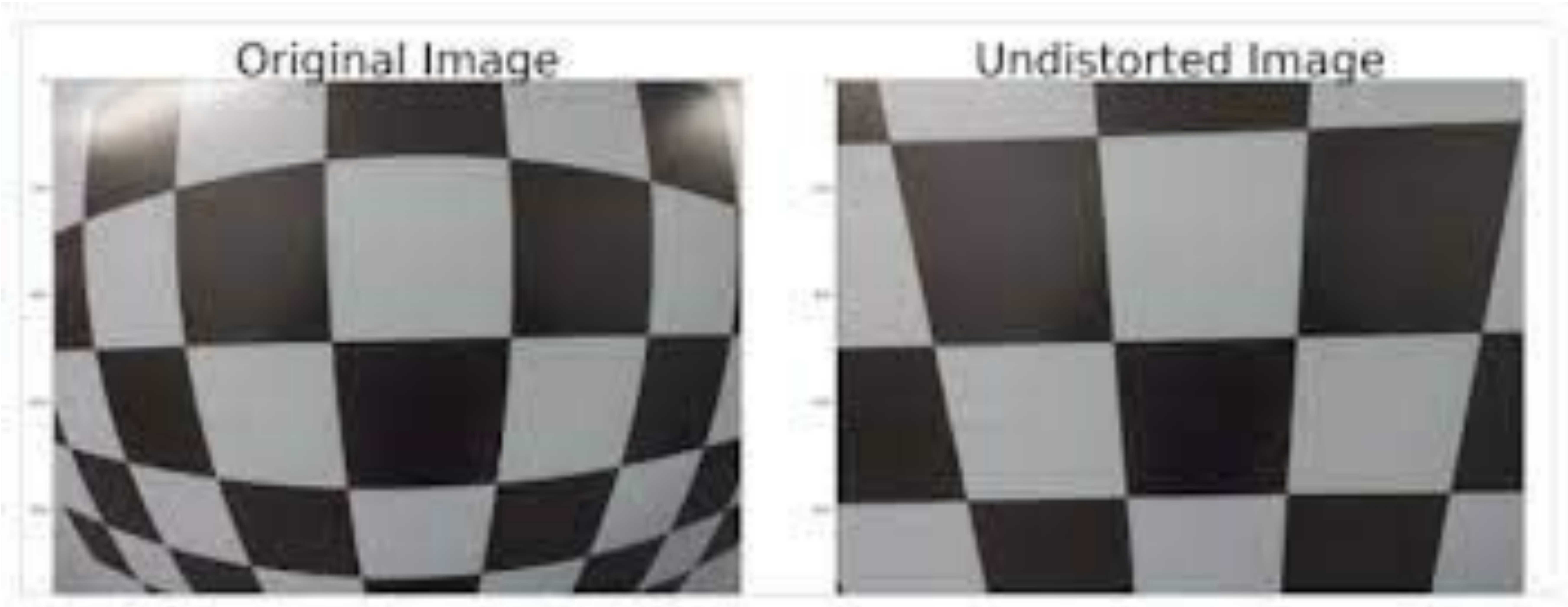
Camera calibration

- Examples of cases where it is useful

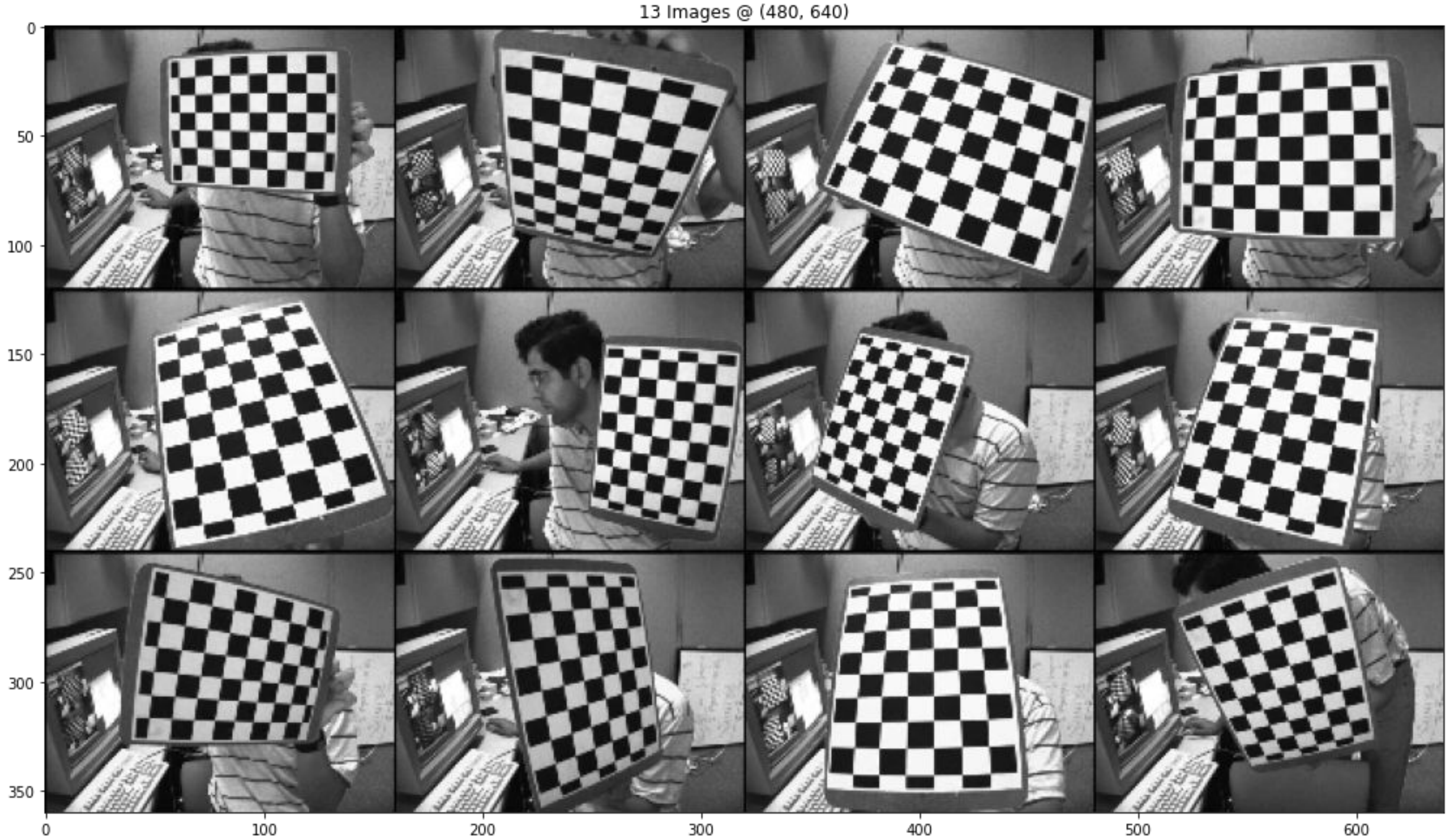


Remove Lens Distortion | Estimate Depth Using a Stereo Camera | Measure Planar Objects | Estimate 3-D Structure from Camera Motion

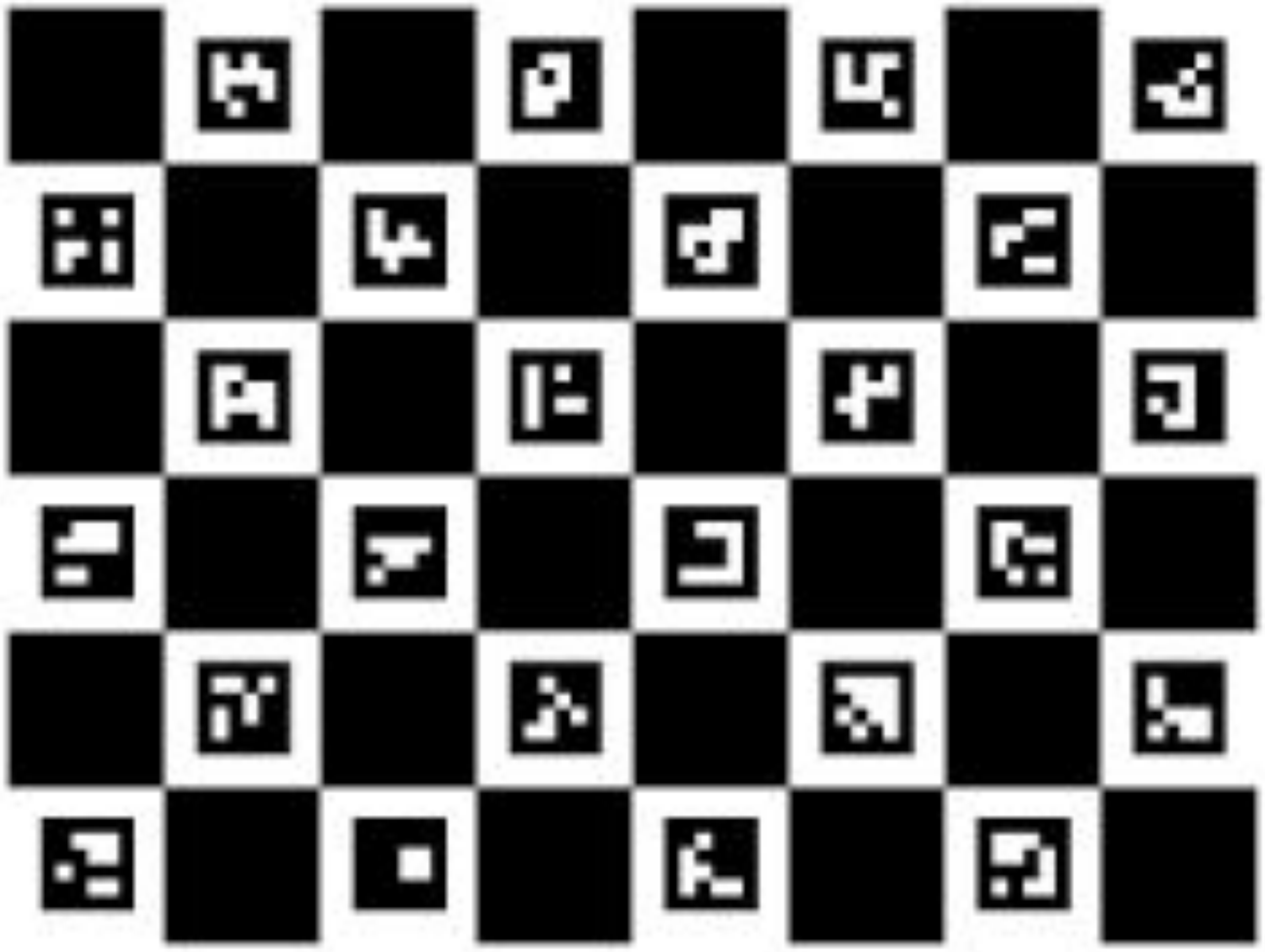
Camera calibration in OpenCV



Camera calibration in OpenCV



Camera calibration in OpenCV



Edge detection

- gradient-based detectors (first order derivative)
  - Sobel
  - Prewitt
  - Robert
- gaussian-based detectors (second order derivative)
  - Laplacian (we saw it several weeks ago)
  - Canny

Edge detection - Gradient-based: Sobel

$$M_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad M_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

- Simple and efficient

- Highly sensitive to noise
- Not very accurate
- Thick edges don't work very well

Edge detection - Gradient-based: Prewitt

$$M_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \qquad M_y = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

- Simple and efficient
- Good for detecting image orientation

- Does not detect diagonals
- Sensitive to noise

Edge detection - Gradient-based: Robert

$$M_x = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \quad M_y = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$$

- Easy
- Detects diagonals

- Sensitive to noise
- Not accurate

Edge detection - Gaussian based: Laplacian of Gaussian

$$\mathrm{LoG} = \frac{\partial^2}{\partial x^2}G(x,y) + \frac{\partial^2}{\partial y^2}G(x,y) = \frac{x^2+y^2-2\sigma^2}{\sigma^4}\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)$$

- Easy
- Detects in all directions

- **Very** sensitive to noise
- False edges (due to noise)

Edge detection - Gaussian based: Canny

- Not as easy as the previous, but still easy
- Not as efficient as previous approaches

- Best edge detection
- Not big sensitivity to noise

Canny

Image segmentation using the Sobel method.

Image segmentation using the Canny method.

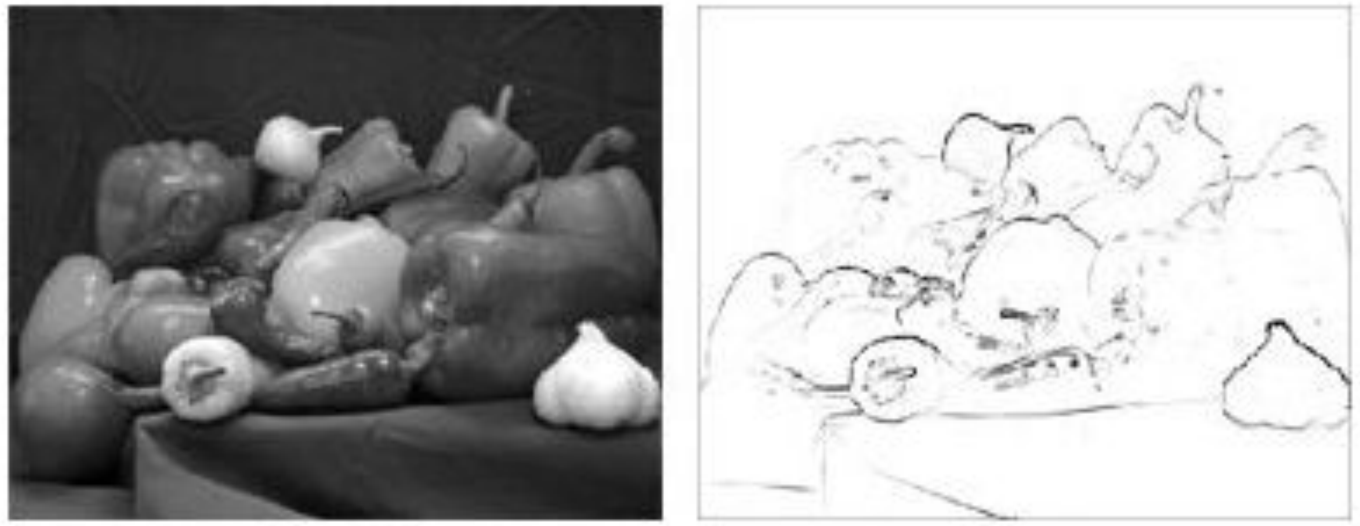


Image segmentation using a Fuzzy Logic method.

Edge detection in OpenCV

```
# Canny Edge Detection
edges = cv2.Canny(image=img_blur, threshold1=100, threshold2=200)

# Display Canny Edge Detection Image
cv2.imshow('Canny Edge Detection', edges)
cv2.waitKey(0)
```

https://learnopencv.com/edge-detection-using-opencv

Extracting and matching keypoints

- Keypoints: points in an image that are distinctive and can be reliably located under various transformations such as scaling, rotation, and illumination changes.
- Descriptors: vectors (or sets) of attributes that describe the local region around a keypoint, in a way that makes it possible to match these regions across different images.

What do we want from a keypoint?

- Distinctiveness: Keypoints are usually located at points of interest like corners, blobs, or edges where the image intensity changes significantly.
- Scale and Rotation Invariance: Good keypoints can be detected regardless of the scale or rotation of the image, making them useful for matching across different viewpoints.
- Localization: They are accurately localized within the image, providing precise points for further analysis.
- Stability: Keypoints should be stable across varying conditions like lighting changes and slight deformations.

Examples of Keypoints detectors

- Harris Corner Detector: Identifies corners in an image by finding points where the intensity gradient is significant in multiple directions.
- SIFT (Scale-Invariant Feature Transform): Detects keypoints that are invariant to scale and rotation, providing robust matching points.
- FAST (Features from Accelerated Segment Test): A high-speed corner detection algorithm that is commonly used in real-time applications.
- ORB (Oriented FAST and Rotated BRIEF): Combines FAST keypoint detection with an orientation component to provide rotationally invariant keypoints.

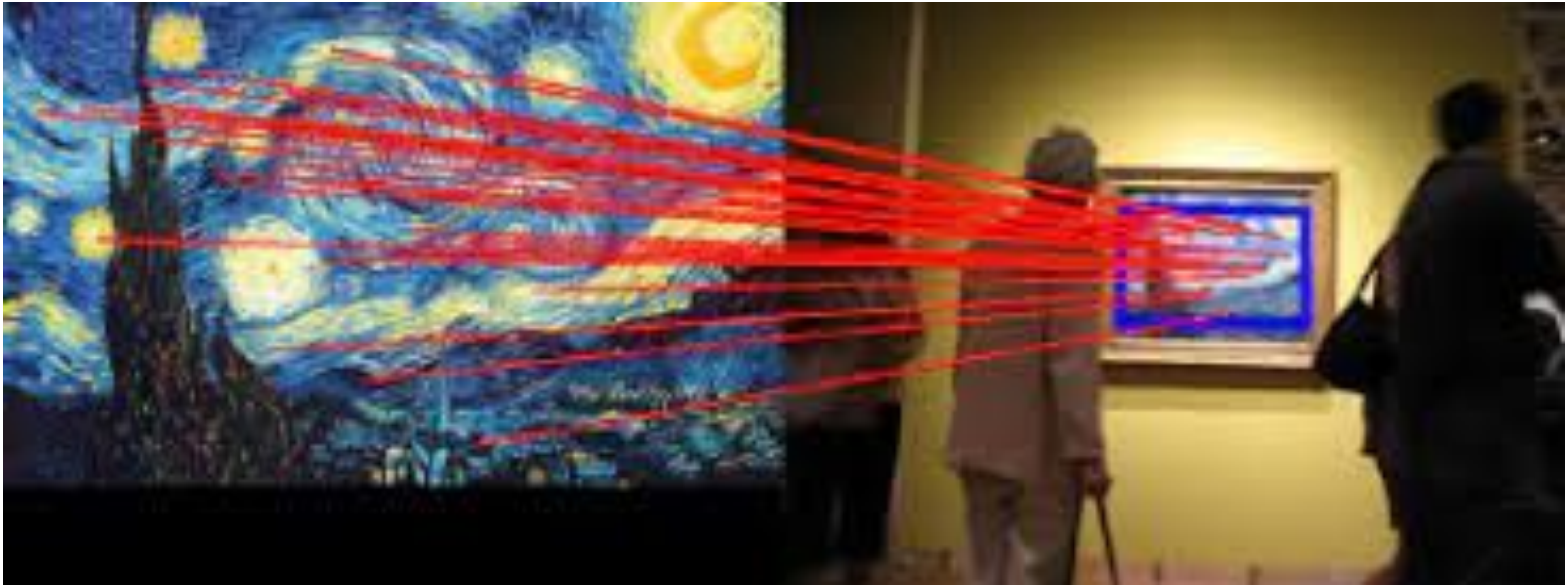
What do we want from a descriptor?

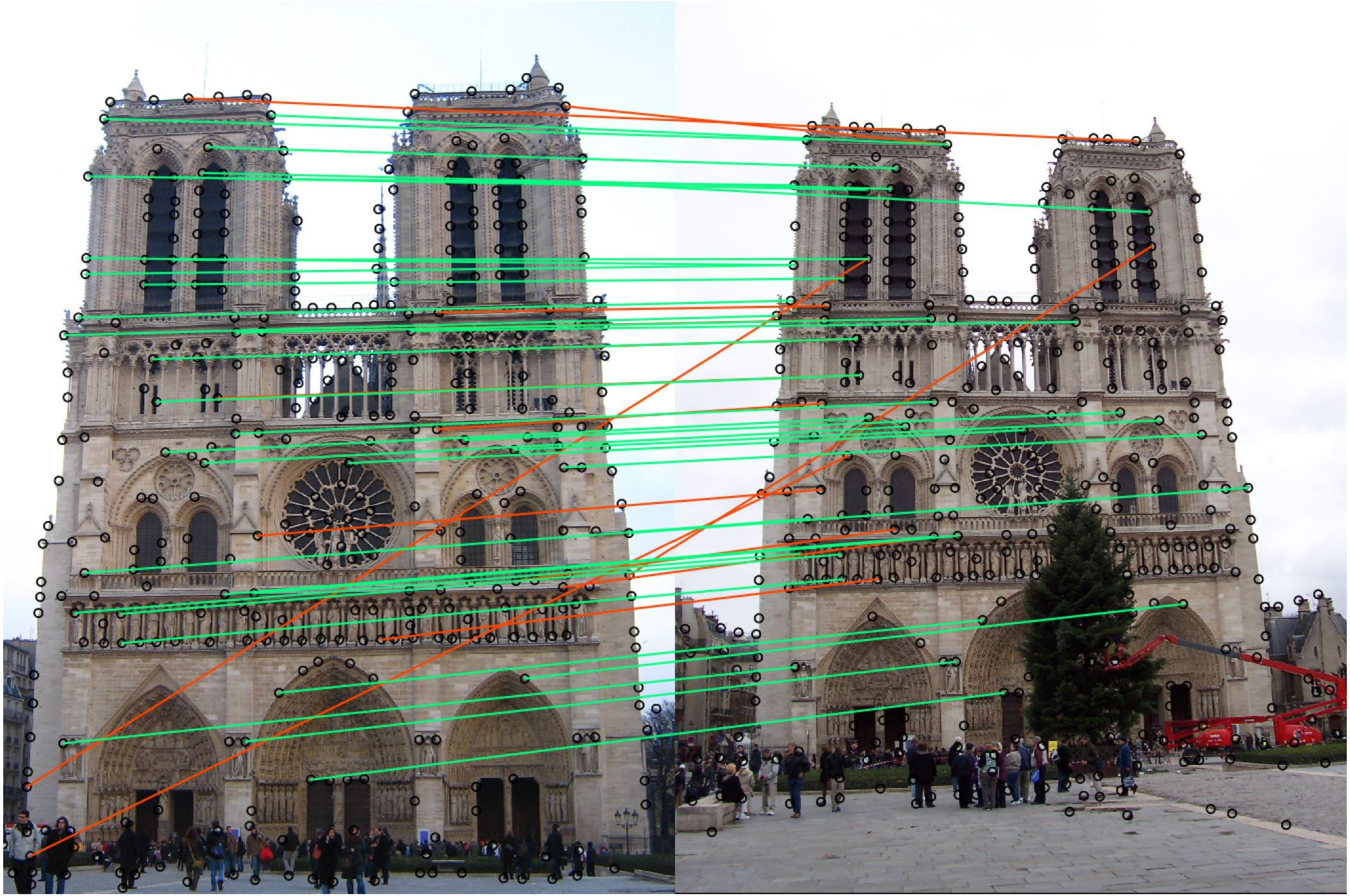
- Robustness: Descriptors should be robust to noise, changes in lighting, and other variations.
- Uniqueness: unique signature for each keypoint (enabling accurate matching).
- Compactness: compact vectors, easily and efficiently comparable.
- Invariance: Good descriptors are invariant to changes in scale, rotation, and, as mentioned before, illumination.

Examples of descriptors

- SIFT Descriptor: Captures gradient information around the keypoint in a multi-scale approach, providing a robust feature vector.
- SURF (Speeded-Up Robust Features): A faster alternative to SIFT that uses integral images for speed, capturing gradient information efficiently.
- BRIEF (Binary Robust Independent Elementary Features): A binary descriptor that is fast to compute and match, suitable for real-time applications.
- ORB Descriptor: Combines the efficiency of BRIEF with orientation information to provide a fast and robust descriptor.

MATCH FOUND





Summary of today

- Cammera model.
- Cammera calibration.
- Edge detection.
- Extracting and matching keypoints.

Activities

- Review the PC3 and, if necessary, ask questions.
- Learn to detect keypoints and compute descriptors in images using OpenCV, and match these keypoints between two images to find correspondences


UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA
> Reinventa el mundo <
Thank you
TRANSFORMATEC