Welcome to Computer Vision

Computer Vision

Dr. Muhammad Tahir

DEPARTMENT OF COMPUTER SCIENCE,

FAST-NUCES, Peshawar

Course Details

LECTURES: Monday & Wednesday

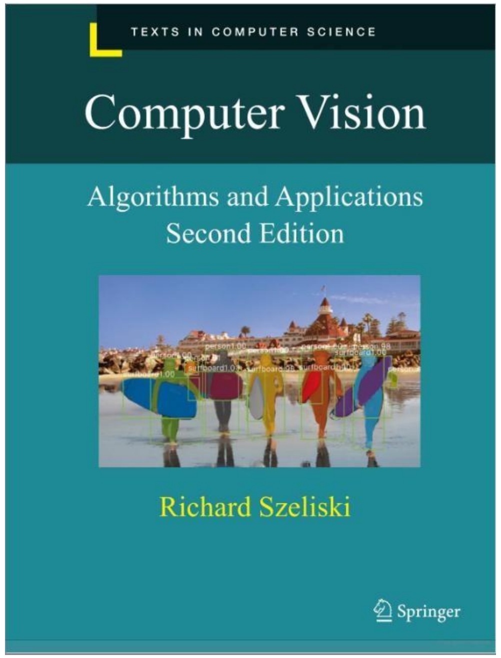
TIMINGS:

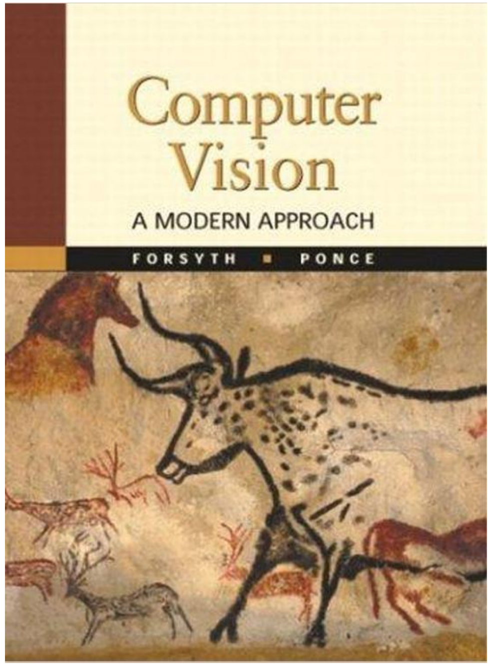
9:30 am – 11:00 am

MY OFFICE:

OFFICE HOURS:

EMAIL: m.tahir@nu.edu.pk





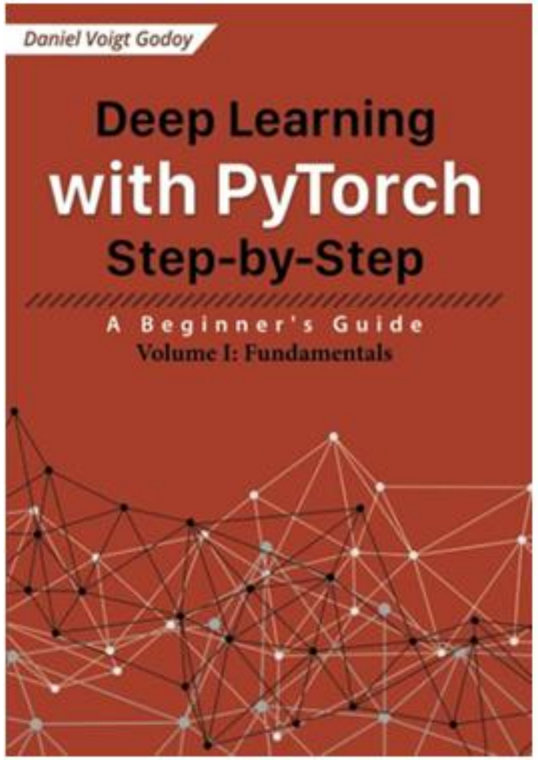
References

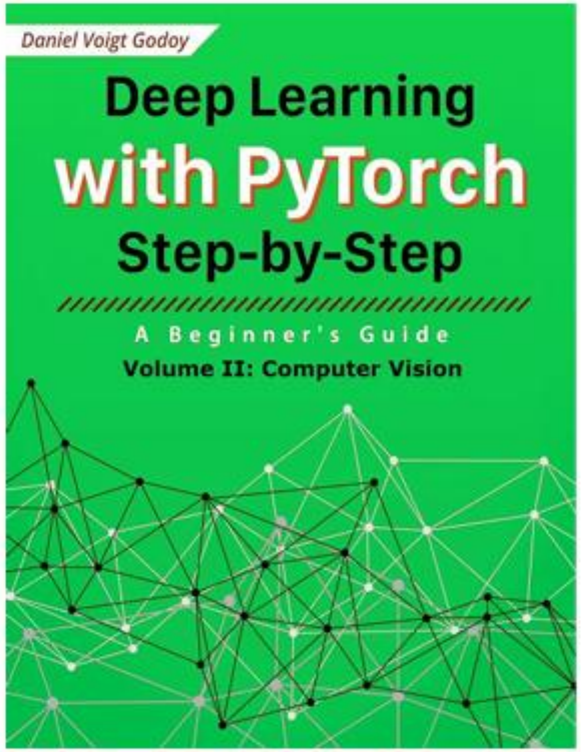
The material in these slides are based on:

1. Rick Szeliski's book: Computer Vision: Algorithms and Applications
2. Forsythe and Ponce: Computer Vision: A Modern Approach

Recommended Books

Deep Learning with PyTorch Step-by-Step by Daniel Voigt Godoy





Course Learning Outcomes

No	CLO (Tentative)	Domain	Taxonomy Level	PLO
1	Understanding basics of Computer Vision: algorithms, tools, and techniques	Cognitive	2	
2	Develop solutions for image/video understanding and recognition	Cognitive	3	
3	Design solutions to solve practical Computer Vision problems	Cognitive	3	

Outline

- Feature Extraction
- Interest Points
- Corner Detectors

A Problem with Edges

- Edges are insensitive to intensity changes, but not to other image transformations

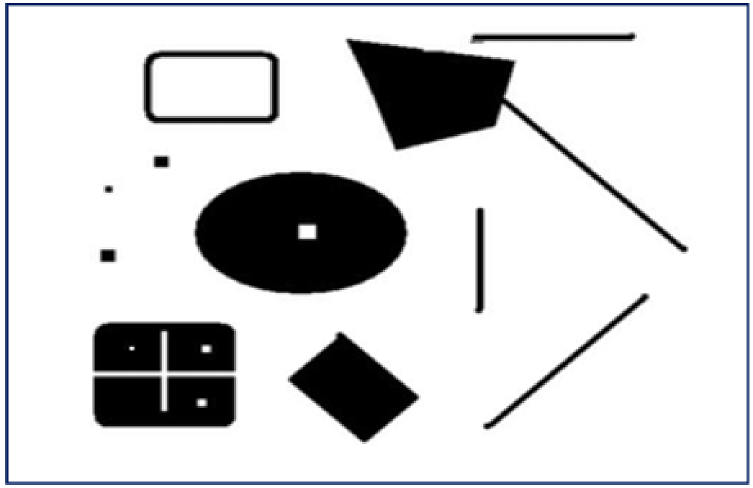
- **Insensitive to Intensity Changes**
  - Even though the **brightness** of the two images is different (left = bright, right = darker), the **edges** (places where pixel intensity changes sharply) are still in the same locations.
  - So, edge detectors (like Sobel, Canny) will detect similar boundaries in both images. This shows that **edges are robust to illumination changes**.

A Problem with Edges

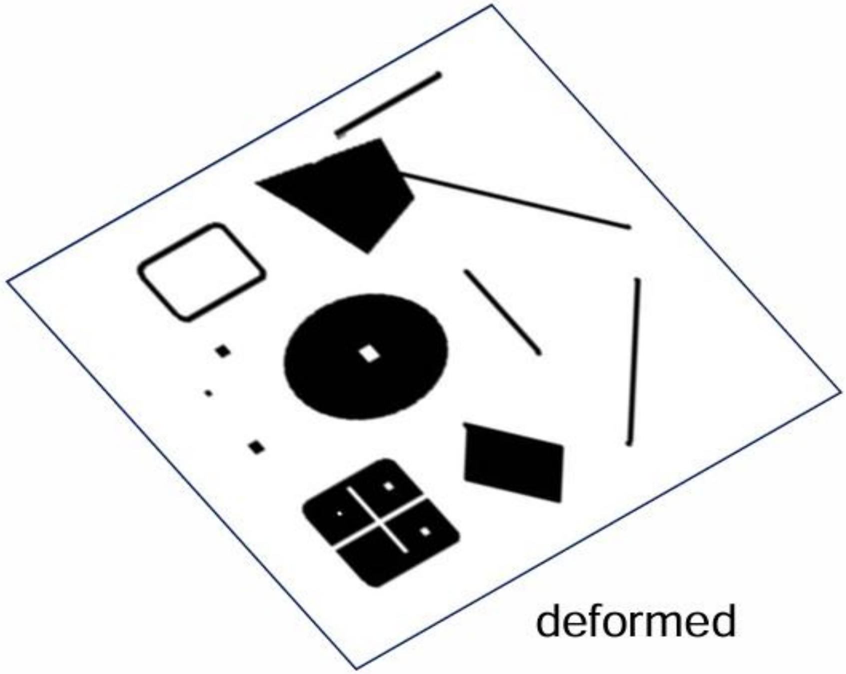
- **Not Invariant to Other Transformations**
  - If the cow was **rotated, scaled, or deformed** (instead of just brightness changed):
    - The positions of the edges would change.
    - Edge detection is not robust to **rotation, scale, or affine transformations**.
- Example: If we rotate the cow 90°, the detected edge map would also rotate, so direct comparison fails.
- This means **edges alone are not good for recognizing objects across different viewpoints**.

Interest Points

- interest points = keypoints = features
  - These are locations in an image that are *distinctive* and can be found again after transformations (rotation, scaling, deformation, etc.).
- A low-level building block in many computer vision applications
- Suppose you have to click on some point, go away and come back after I deform the image, and click on the same points again.
- Which points would you choose?



original



deformed

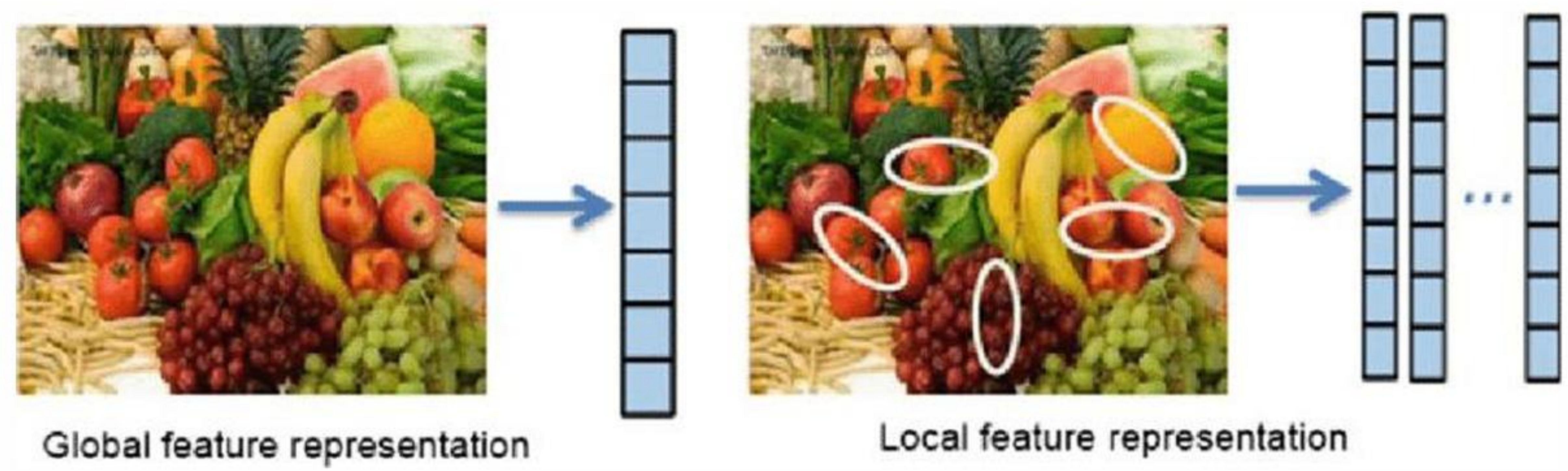
Not all points in an image are equally good choices for **repeatable features**

- **Bad choices** → Points on **flat regions** (inside a white area, or inside a solid black shape).
  - These don't have local uniqueness. After transformation, you won't know *exactly* where you clicked.
- **Bad choices** → Points on **edges/lines**.
  - Along an edge, many points look similar. Small shifts will confuse the detector.
- **Good choices** → Points on **corners, junctions, or blobs**.
  - These are unique, well-defined spots (e.g., the corner of a rectangle, or where lines intersect).
  - Even after deformation (rotation, scaling, affine transformation), you can find these points again.
- That's why **corner detectors (Harris, Shi–Tomasi)** are so important: they automatically find the most reliable points.

Local Image Features

- What Are Local Features / Interest Points?
- Local features refer to a pattern or distinct structure found in an image, such as
  - A point. An edge. Small image patch, Corner, Blob
- They are usually associated with an image patch that differs from its immediate surroundings by texture, color, or intensity.
- What the feature represents does not matter, just that it is distinct from its surrounding

Local Image Features



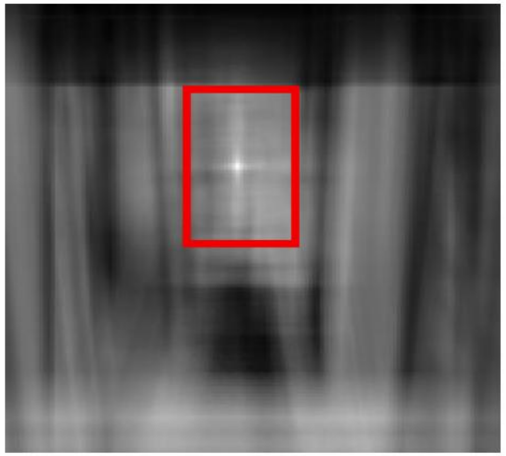
Object recognition: Is it really so hard?

- **Global Template Matching** works by comparing the template image with every possible sub-region of the larger image (using correlation).
- if the object looks exactly the same (same scale, orientation, lighting), this works.

This is a chair

Find the chair in this image

Output of normalized correlation



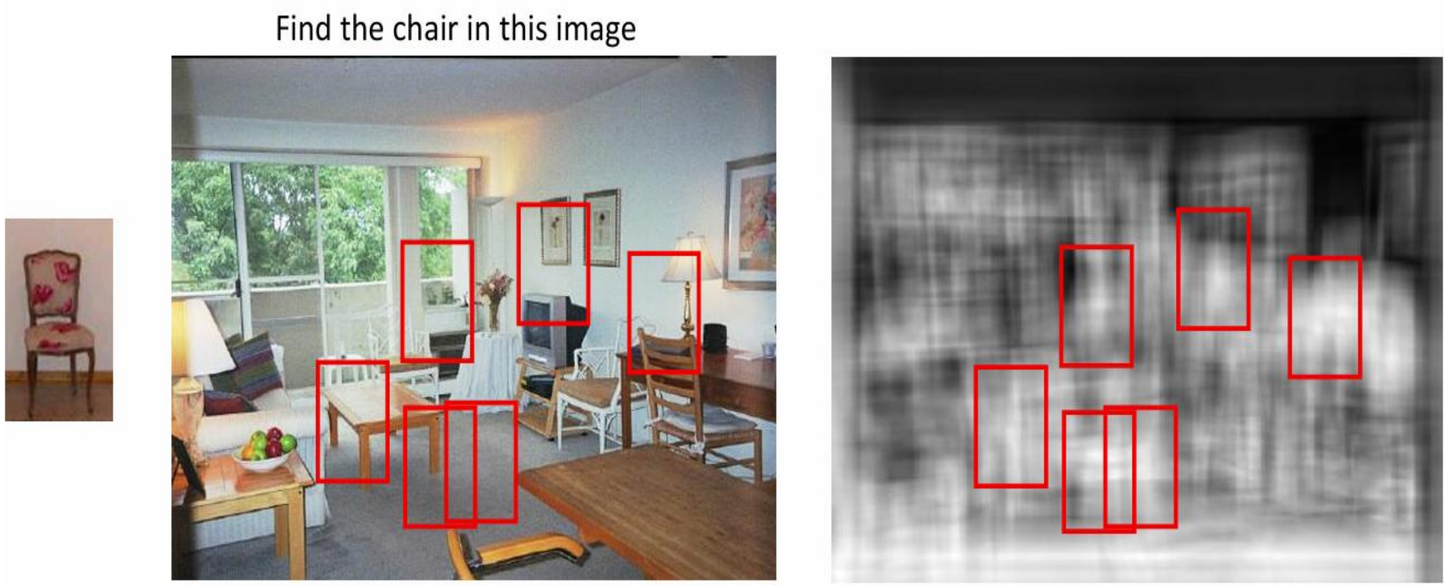
But in reality:

- Objects appear at different **scales** (closer/farther).
- They appear at different **orientations** (rotated, tilted).
- They appear under different **lighting or occlusions**.
- Backgrounds may confuse the correlation (false matches).

That’s why the method often fails or gives misleading results

Object recognition: Is it really so hard?

- Real-world object recognition is hard because objects rarely look identical across scenes.
- These techniques are inadequate for three-dimensional scene analysis for many reasons, such as occlusion, changes in viewing angle, and articulation of parts." Nivatia & Binford, 1977.
- That's why we need better features (corners, keypoints, descriptors like SIFT/SURF/ORB) that are repeatable and invariant.

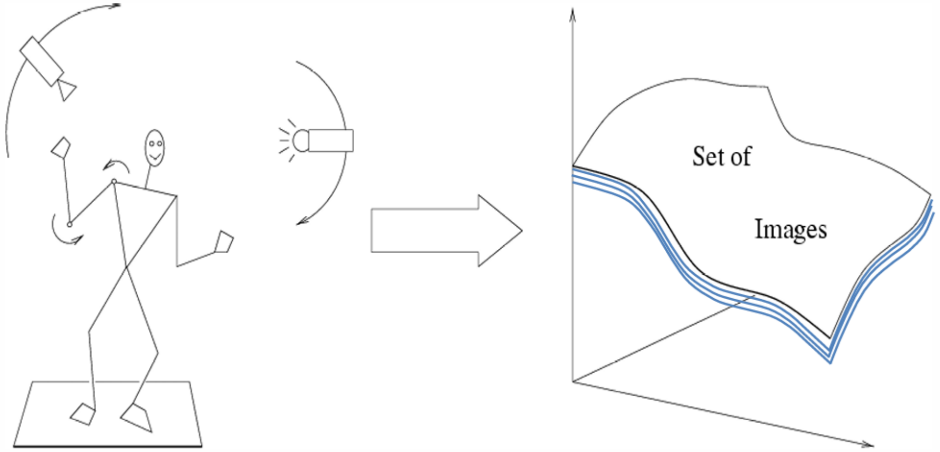


Pretty much garbage
Simple template matching is not going to make it

And it can get a lot harder

Why is this hard?

- Several sources of **variability** affect how the person looks in an image:
  - **Camera position** – if the camera moves (different viewpoints, angles), the same object looks very different.
  - **Illumination** – changing the lighting direction/intensity produces different shadows and highlights.
  - **Shape parameters** – the object itself may deform or move (person raises arms, changes posture).

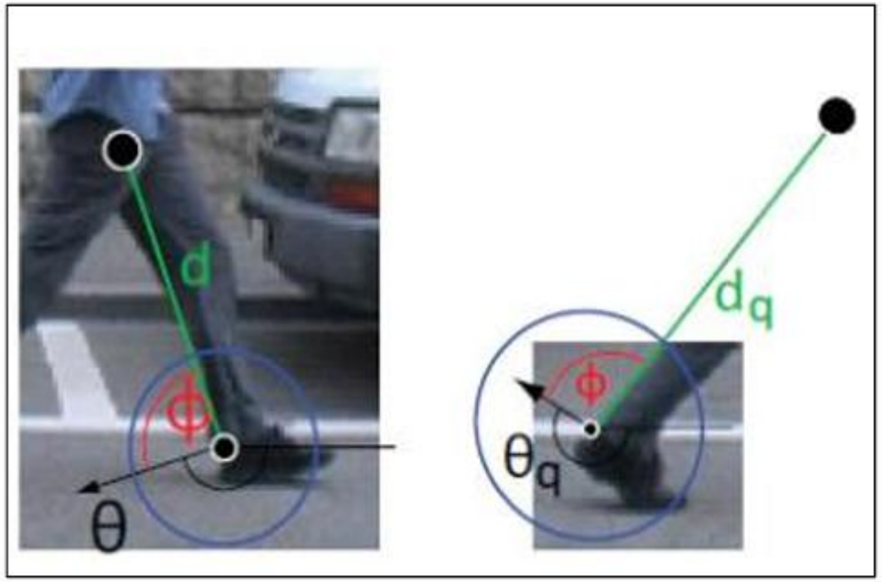


Variability: Camera position
Illumination
Shape parameters

- Object recognition is challenging because we don't just need to match one picture of an object.
- We must recognize the object across its variability (viewpoint, lighting, deformation).
- A robust recognition system should learn to group all these images into the same object identity, even though the raw pixels are different.

Motivation for using Local Features

- Global representation have major limitations
  - Occlusions
  - Articulation
  - Intra-Category Variations

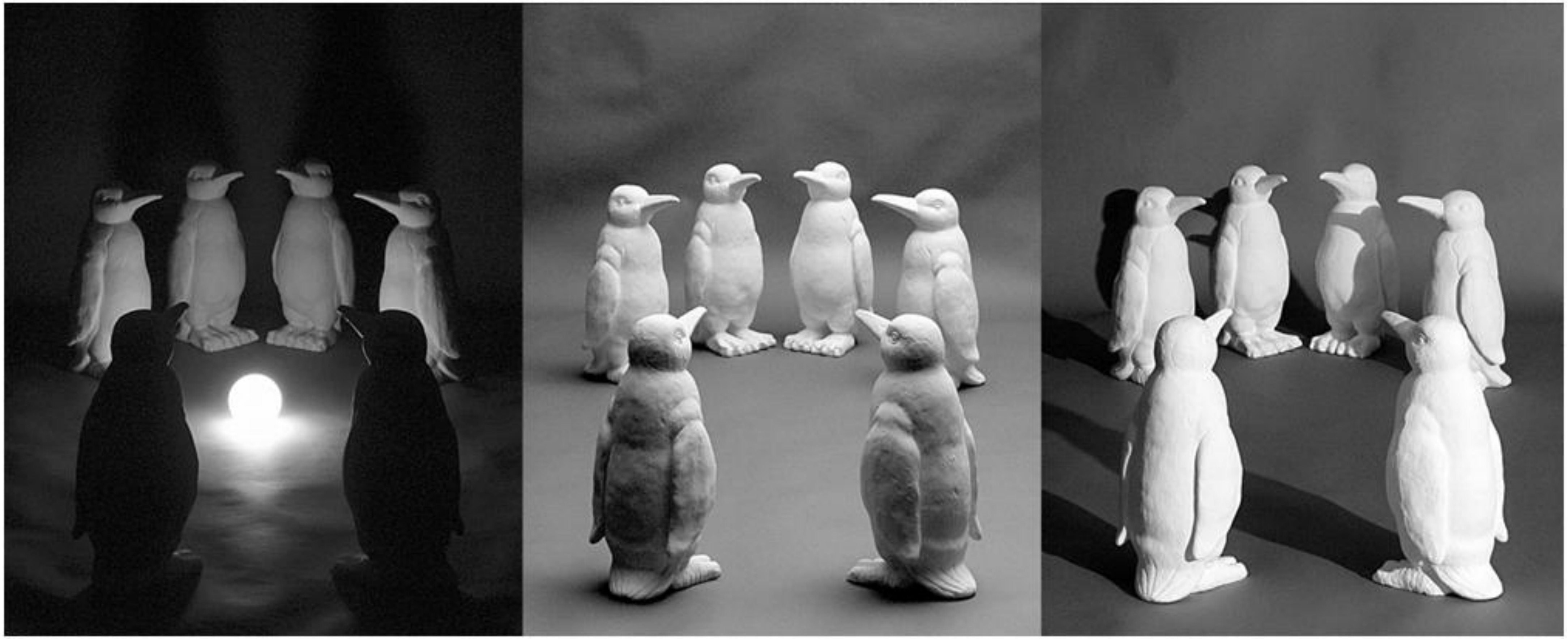


How many object categories are there?

Challenge: variable viewpoint

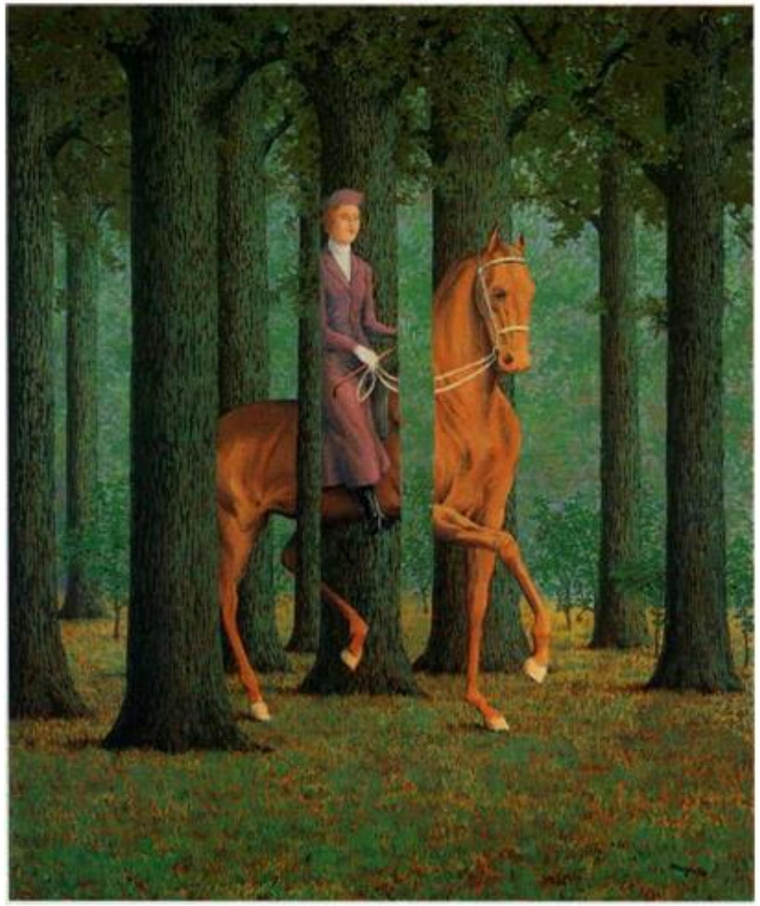
Michelangelo 1475-1564

Challenge: variable illumination



Challenge: deformation

Challenge: Occlusion



Magritte, 1957

Occlusion

Challenge: background clutter

Kilmeny Niland. 1995

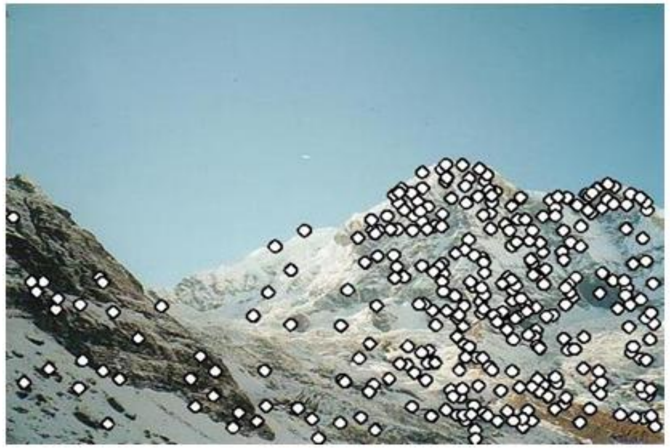
Challenge: intra-class variations

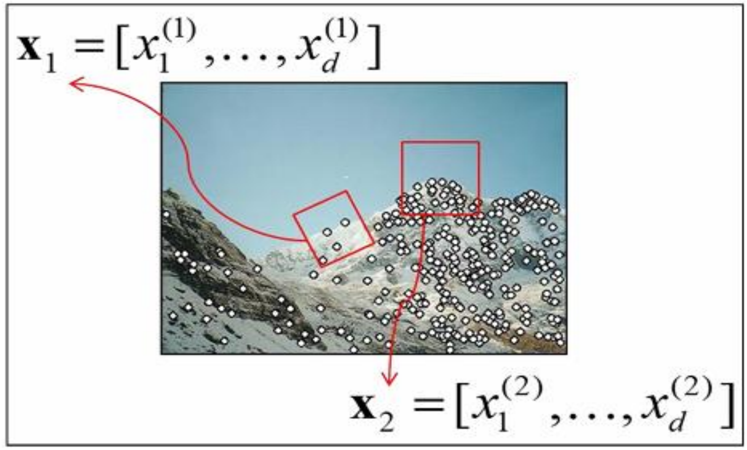


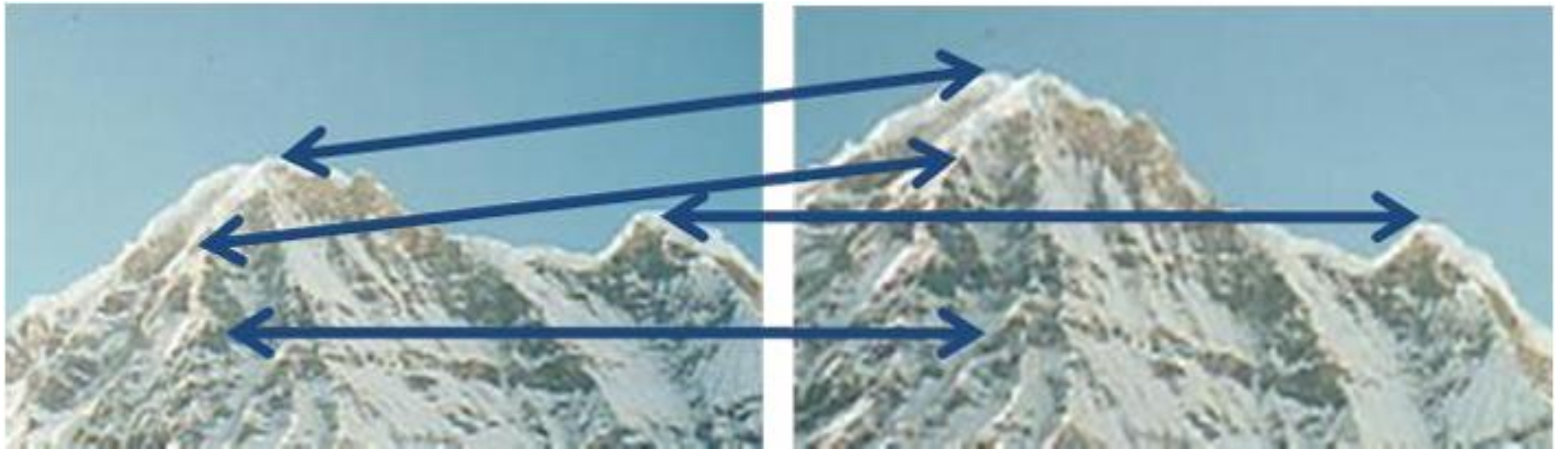



Local or Interest Point Based Matching Components

- **Detection**: Identify the interest points
- **Description**: Extract vector feature descriptor surrounding each interest point.
- **Matching**: Determine correspondence between descriptors in two views

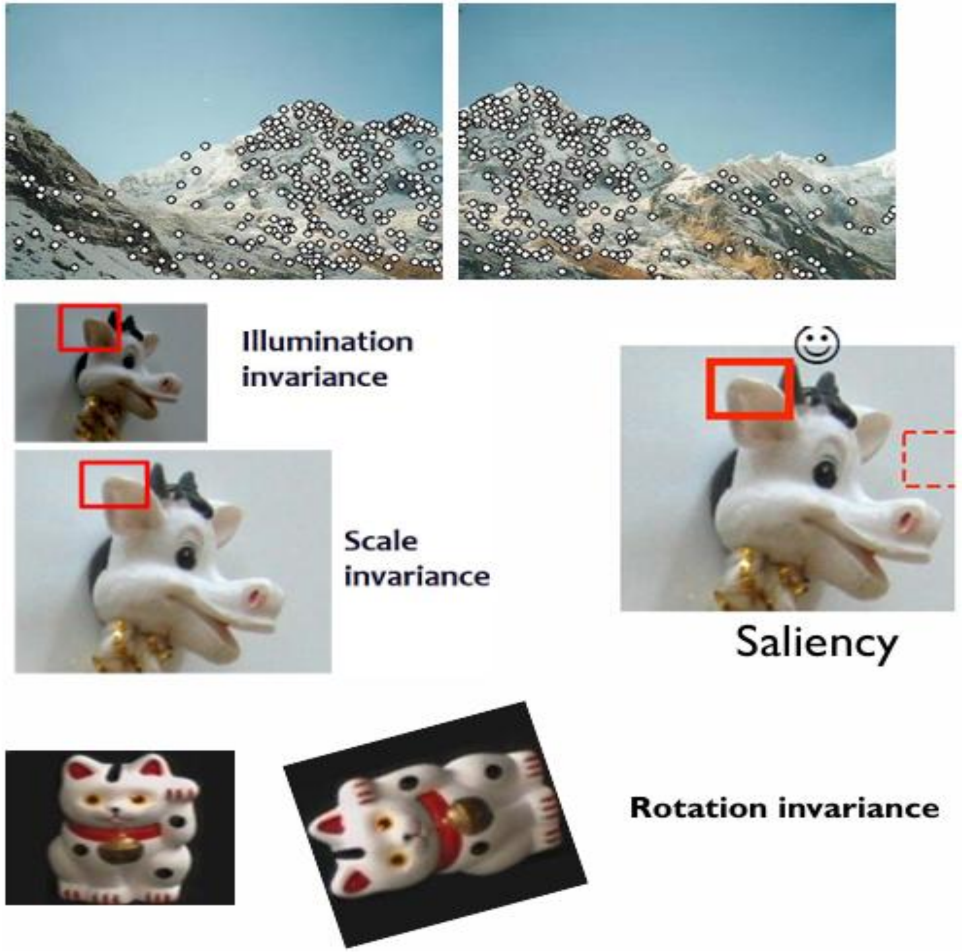






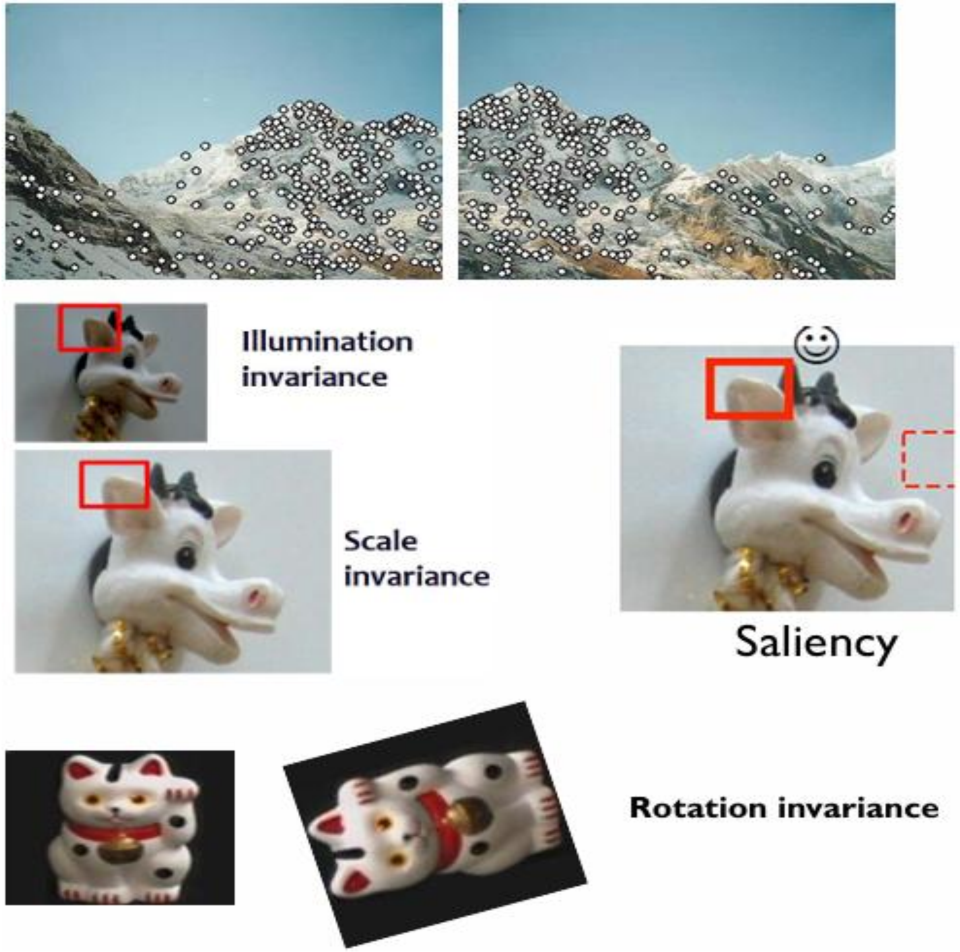
Goals/Characteristics of Good Interest Points

- **Repeatability**
  - A good feature should appear consistently in multiple images of the same object, even under changes in:
    - **Illumination** (lighting conditions),
    - **Scale** (zooming in or out),
    - **Rotation** (object rotated).
  - Example: The same corner on a cow toy can be detected whether the image is bright, dark, zoomed, or rotated.
  - This ensures reliable matching across different conditions.
- **Saliency**
  - Each feature should be **distinctive** (not confused with others).
  - It should provide a **unique signature** that makes it easy to match with its counterpart in another image.
  - Example: The cow’s ear is a unique feature, whereas a flat region may not be distinctive.



Goals/Characteristics of Good Interest Points

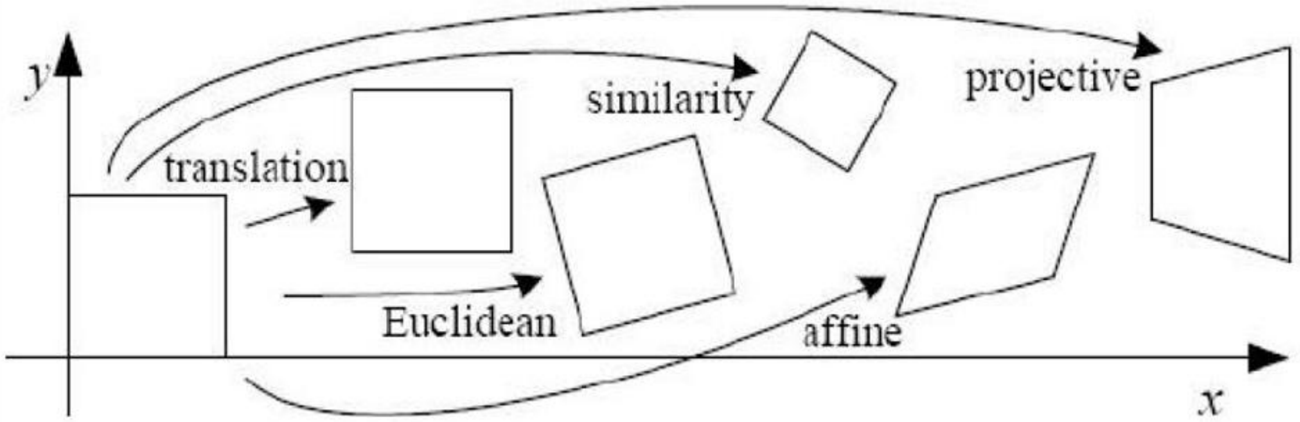
- **Compactness and Efficiency**
  - Ideally, we want **fewer features than pixels**.
  - Storing and matching millions of pixels is impractical, so extracting only the most informative points is efficient.
  - Example: Instead of all pixels, only strong corners or blobs are extracted.
- **Locality**
  - A good feature should occupy a **small, local area** of the image.
  - This makes it more **robust to clutter and occlusion** (e.g., if part of the image is hidden, the visible local features can still help recognition).
  - Example: Even if the cow is partially hidden, some keypoints (like the ear or nose) remain detectable.



Goals/Characteristics of Good Interest Points

- Interest point must provide some invariance to geometric and photometric differences between two views.
- Geometric Transformations
  - Translation
  - Euclidean Transformation = translation + rotation
  - Similarity Transformation = translation + rotation + uniform scaling
  - Affine Transformation = non-uniform scaling, rotation, shear
  - Projective Transformation (Homography) = viewing the object from an angle
- Photometric Variability
  - Includes **illumination changes** (lighting direction, brightness, contrast)

Interest points must be stable under transformations like translation, scaling, rotation, affine, projective distortions, and lighting changes.



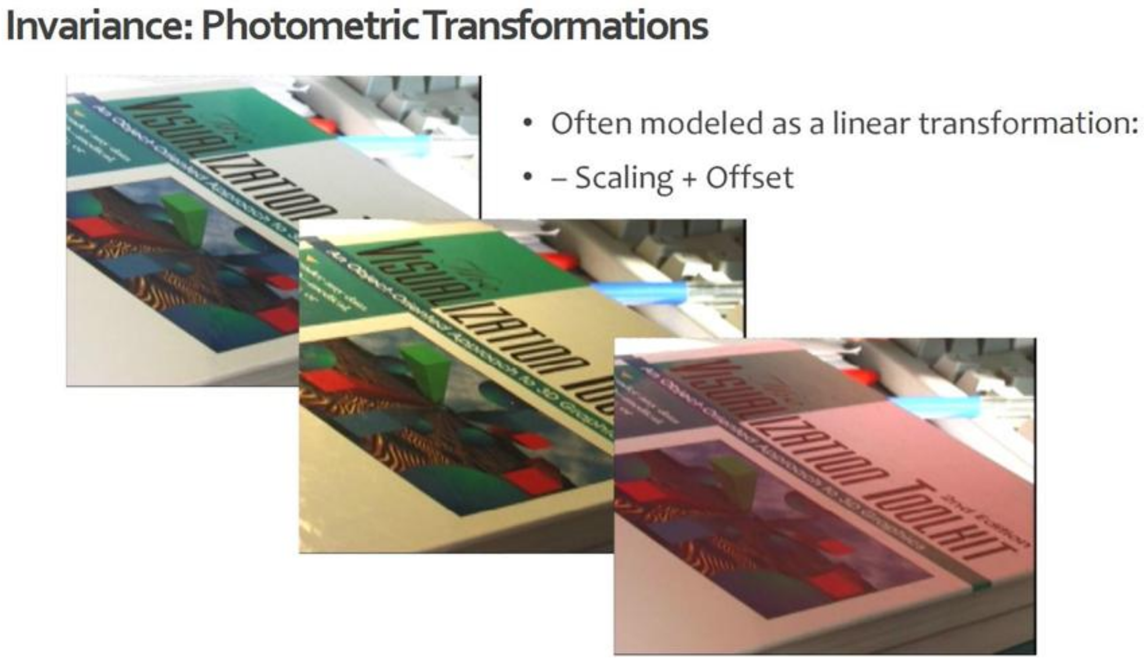
Goals/Characteristics of Good Interest Points

- Same object can look brighter in sunlight, darker in shadows, or tinted under colored lighting

- A simple way to model this is using a **linear transformation** on pixel intensities:

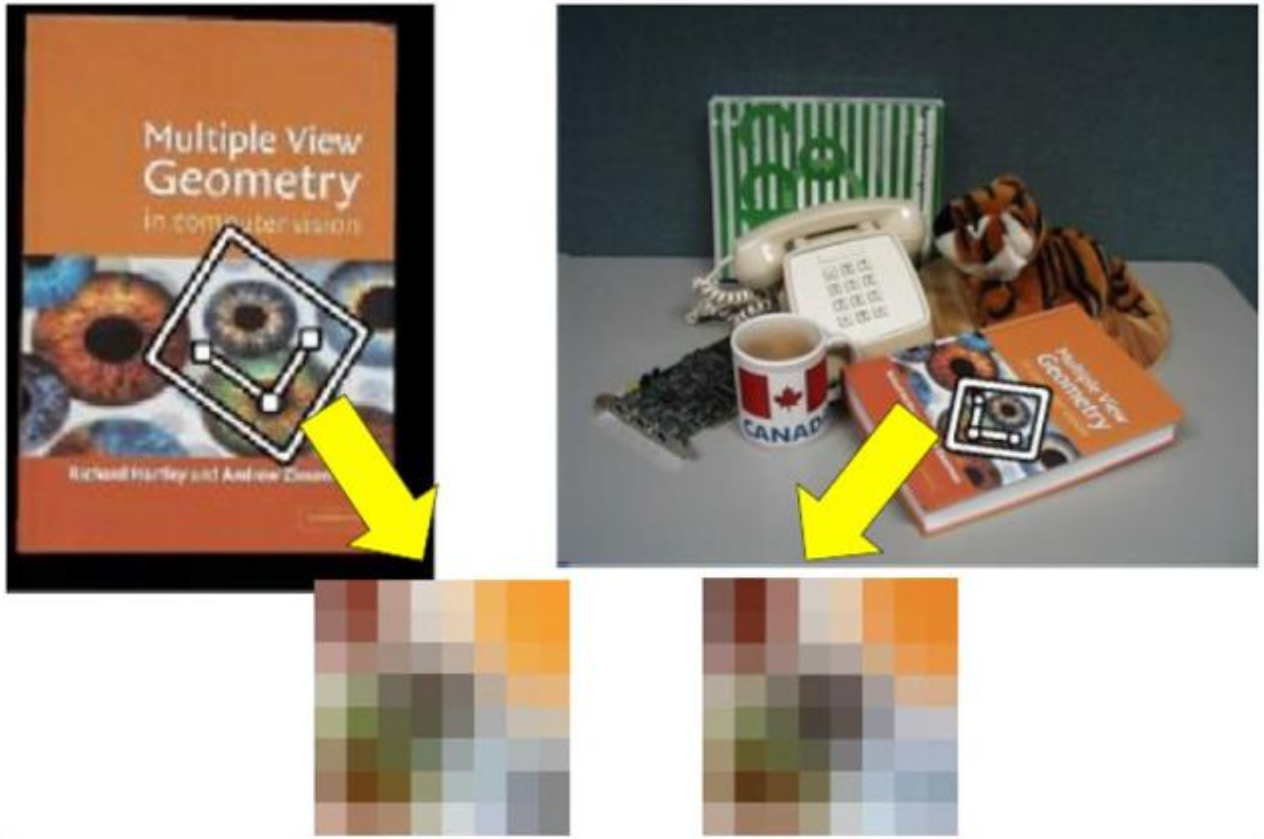
$$\acute{I}(x,y) = a.I(x,y) + b$$

- $I(x,y)$: Original pixel intensity
- a: Scaling (contrast change, e.g. brighter/darker)
- b: Offset (brightness shift, e.g. adding uniform light)
- $I'(x, y)$: New pixel intensity

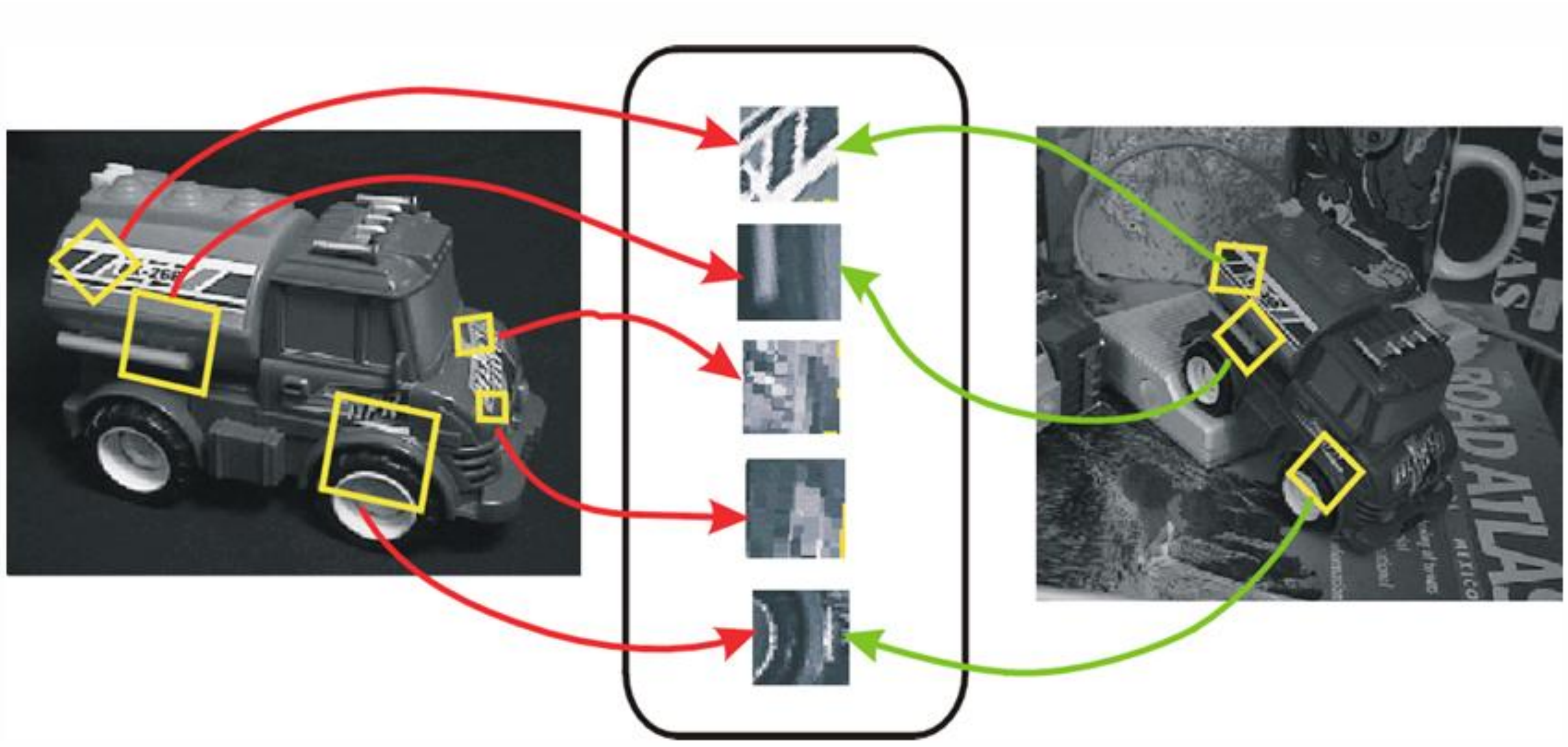


Goals/Characteristics of Good Interest Points

Invariance: Geometric Transformations



Invariant Example



Applications

- Feature points are used for:
  - Image alignment
  - 3D reconstruction
  - Motion tracking
  - Robot navigation
  - Indexing and database retrieval
  - Object recognition

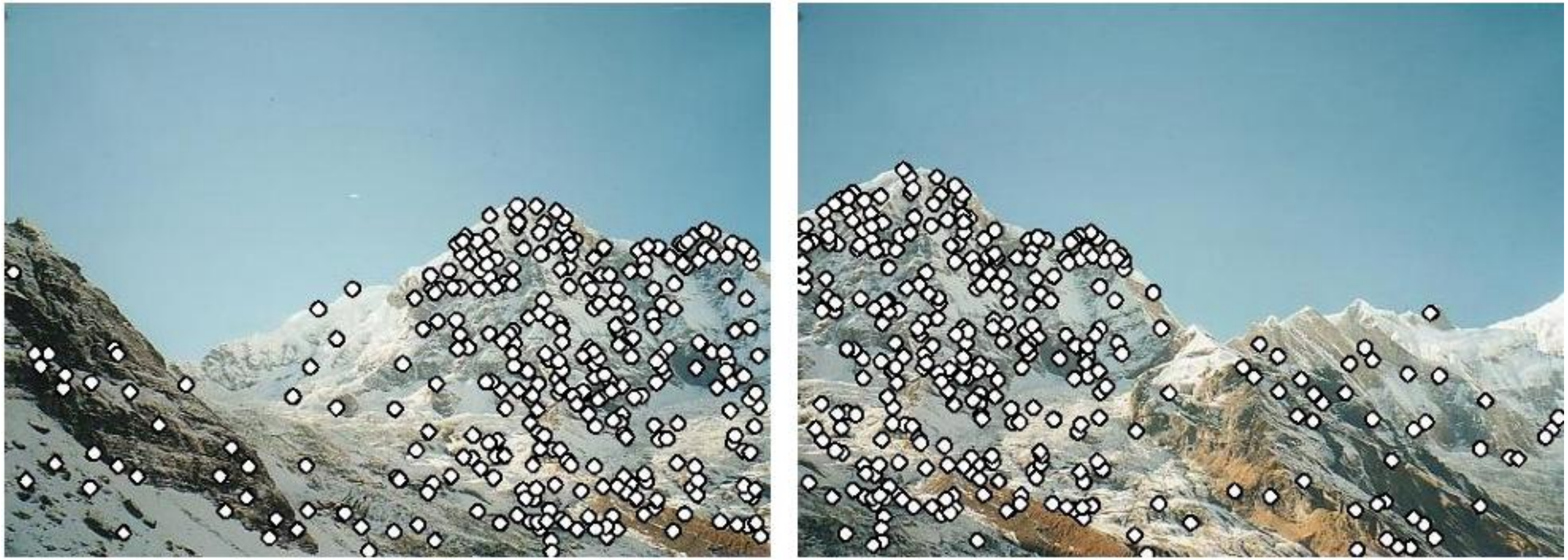
Application 1: Build a Panorama

How do we build panorama?

We need to match (align) images

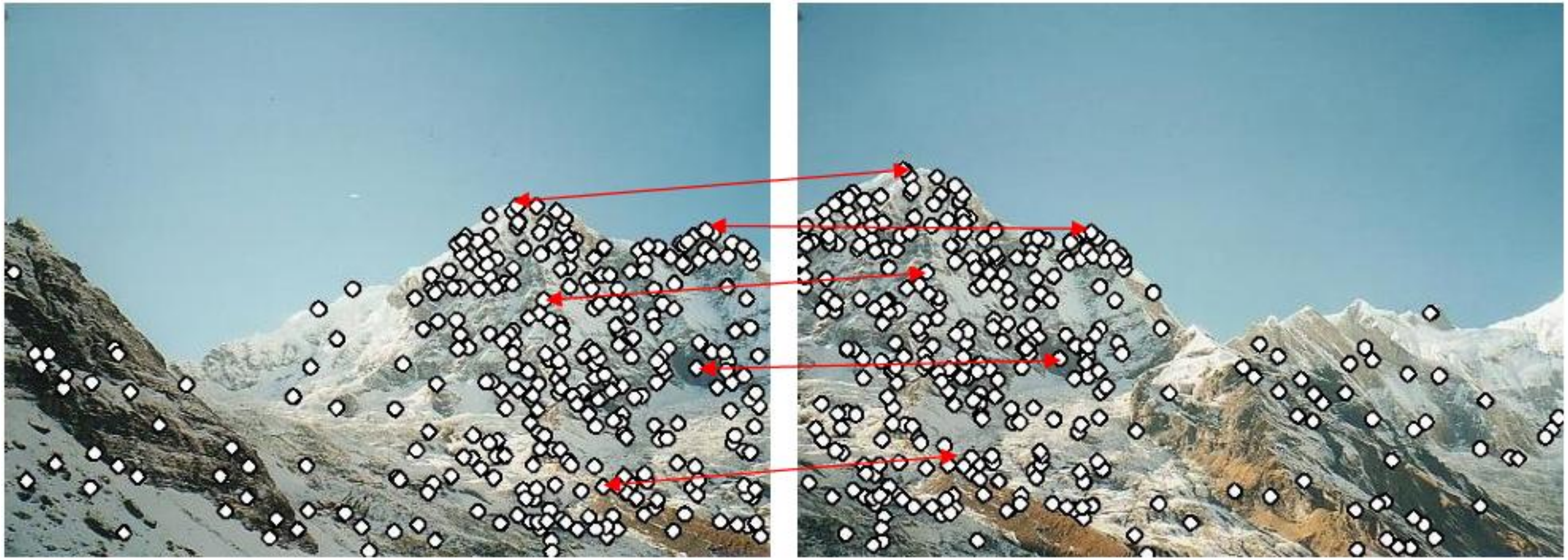
Matching with Features

Detect features (feature points) in both images



Matching with Features

- Detect features (feature points) in both images
- Match features - find corresponding pairs



Matching with Features

- Detect features (feature points) in both images
- Match features - find corresponding pairs
- Use these pairs to align images

Matching with Features

Problem 1:

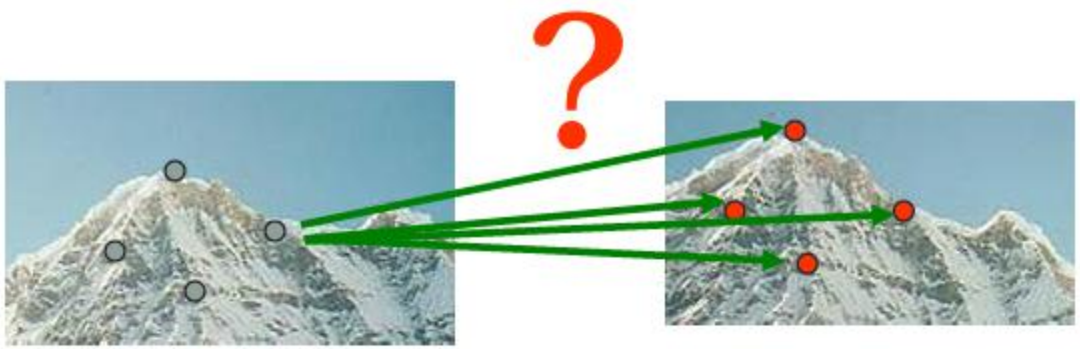
- Detect the same point independently in both images

- no chance to match!
- **We need a repeatable detector**

Matching with Features

Problem 2:

- For each point correctly recognize the corresponding one



- **We need a reliable and distinctive descriptor**

Application 1: Build a Panorama

- **The Problem: Stitching Images Together**
  - You take two overlapping photos of a mountain scene (left and right).
  - To create a wide panorama, we need to **align** them properly.
- **Finding Correspondences (Keypoints)**
  - Detect interest points (corners, distinctive patches) in both images.
  - Match pairs of points across the two images (e.g., the peak of the mountain, rocks, etc.).
  - These **matching pairs** allow us to understand how one image relates to the other.

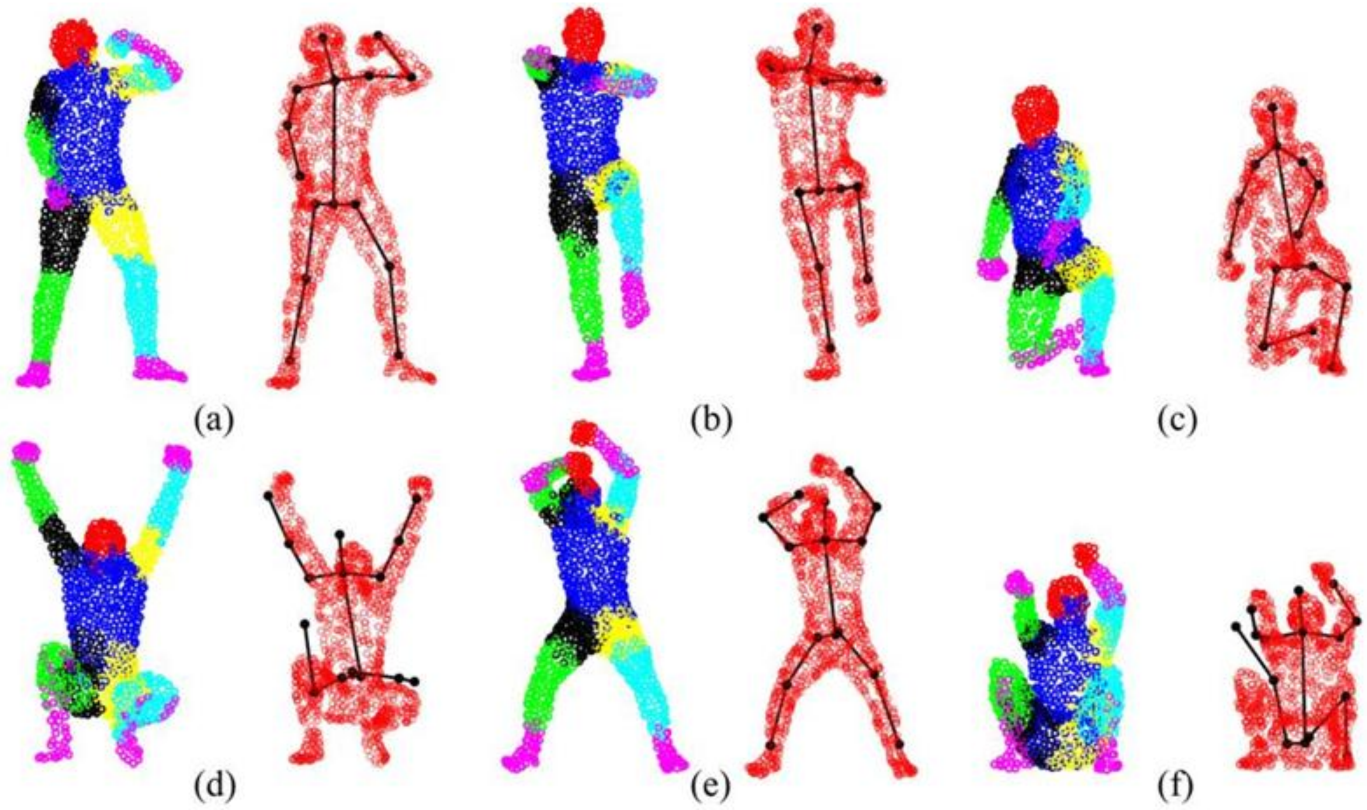
Application 2: Image Registration / Fusion

- **Image Registration:** Aligning two or more images of the same scene taken at different times, from different sensors, or from different viewpoints.
- To make sure that corresponding features (roads, buildings, objects, etc.) overlap correctly.
- Raw images may be misaligned due to scale, ngle, distortion, or sensor differences.

- **Image Fusion:** Combining information from multiple registered (aligned) images into a single image that contains more useful information than any of the individual images.
- To enhance image interpretation or analysis.
- **Example**: Fusing a thermal image with a visible-light image to get both heat information and structural detail.

Application 3: Pose Estimation

- **Pose Estimation** is the process of detecting and tracking the positions of human body joints (like head, arms, legs, torso) from images or videos.
- It essentially maps human movement into a set of **keypoints** (skeleton-like structure).
- These keypoints can then be used to understand the orientation and movement of the person.



Application of local Features

Local image features are widely used in various computer vision applications due to their ability to capture distinctive and repeatable patterns in images. Some common applications include:

1. **Feature Matching:** Local features are used to find correspondences between different images, which is essential for tasks like image alignment, panorama stitching, and object recognition.
2. **Object Detection and Recognition:** Local features are used to detect and recognize objects in images. They provide robustness to changes in scale, rotation, and illumination, making them suitable for object detection in challenging conditions.
3. **Image Registration:** Local features are used to align images from different sources or at different times, such as in medical image analysis or satellite image processing.

Application of local Features

4. **Augmented Reality:** Local features are used to track objects or scenes in real time, enabling the overlay of virtual objects onto the real world in applications like AR games or navigation systems.
5. **Image Retrieval:** Local features are used to index and retrieve images from large databases based on their visual content, enabling content-based image retrieval systems.
6. **3D Reconstruction:** Local features are used to reconstruct the 3D structure of objects or scenes from multiple images, as in structure-from-motion (SfM) and simultaneous localization and mapping (SLAM) systems.
7. **Gesture Recognition:** Local features can be used to recognize hand gestures or body poses in applications such as sign language recognition or human computer interaction.

Available Detectors

These detectors are basic building block for many Computer Vision Applications.

Harris	[Beaudet '78], [Harris '88]
Laplacian, DoG	[Lindeberg '98], [Lowe 1999]
Hessian-Laplace	[Mikolajczyk & Schmid '01]
Harris-/Hessian-Affine	[Mikolajczyk & Schmid '04]
EBR and IBR	[Tuytelaars & Van Gool '04]
MSER	[Matas '02]
Salient Regions	[Kadir & Brady '01]
Others…	

Available Descriptors

- Scale Invariant Feature Transform (SIFT)
- Speed-Up Robust Feature (SURF)
- Histogram of Oriented Gradient (HOG)
- Gradient Location Orientation Histogram (GLOH)
- Pyramidal HOG (PHOG)
- Pyramidal Histogram Of visual Words (PHOW)
- LBP, LTP and variants, HAAR;
- PCA-SIFT, VLAD, MOSIFT,
- Deepfeatures, CNN, Fisher vector,
- SV-DSIFT, BF-DSIFT, LL-MO1SIFT, 1SIFT, VM1SIFT, VLADSIFT,
- DECAF, Fisher vector pyramid, IFV
- Dirichlet Histogram
- Simplex based STV (3-D), MSDR;

Thank you