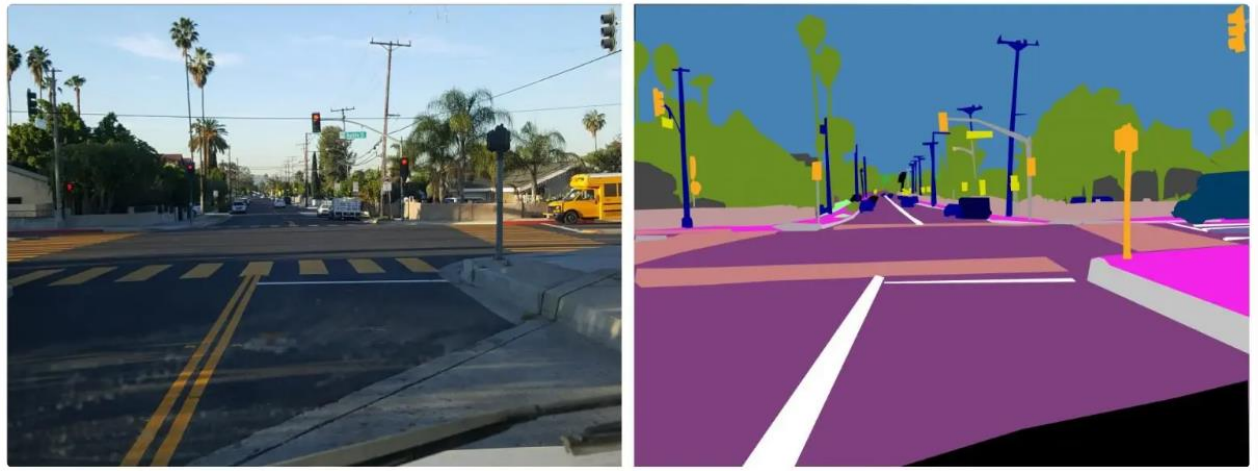
22AIE313 Computer Vision & Image Understanding (2-1-3-4)

Dr. Aiswarya S Kumar
Dept. of Computer Science & Engineering (AI)
Amrita School of Computing

Segmentation

- Image segmentation is a fundamental technique in digital image processing and computer vision.
- It involves partitioning a digital image into multiple segments (regions or objects) to simplify and analyze an image by separating it into meaningful components.



Segmentation

Image Segmentation Techniques

- Thresholding
- Region growing
- Edge-based segmentation
- Clustering
- Watershed segmentation
- Graph-based segmentation
- Deep learning-based segmentation

Read more on segmentation here

Segmentation

Thresholding

- Simple thresholding generates a binary image from a given grayscale image by separating it into two regions based on a threshold value.
- It requires manual selection of the threshold value.
- Otsu's thresholding is computationally more intensive than simple thresholding.
- Otsu selects the threshold that maximizes the between-class variance.

Segmentation

Region Growing

- Region growing is a procedure that groups pixels or subregions into larger regions.
- It starts with a set of “seed” points and from these, regions grow by appending to each seed points those neighboring pixels that have similar properties.
- Region growing based techniques are better than the edge-based techniques in noisy images where edges are difficult to detect.

Segmentation

Watershed Algorithm

- The watershed algorithm is based on the concept of visualizing an image as a topographic surface.
- Primarily used in image processing for segmenting images, especially when dealing with touching or overlapping objects.

Segmentation

The image should be preprocessed before applying Watershed.

Main steps in Watershed

- Identify markers
- Start flooding
- Boundary identification

Good read on https://medium.com/@sasasulakshi/opencv-morphological-dilation-and-erosion-fab65c29efb3

Segmentation

K-Means Clustering

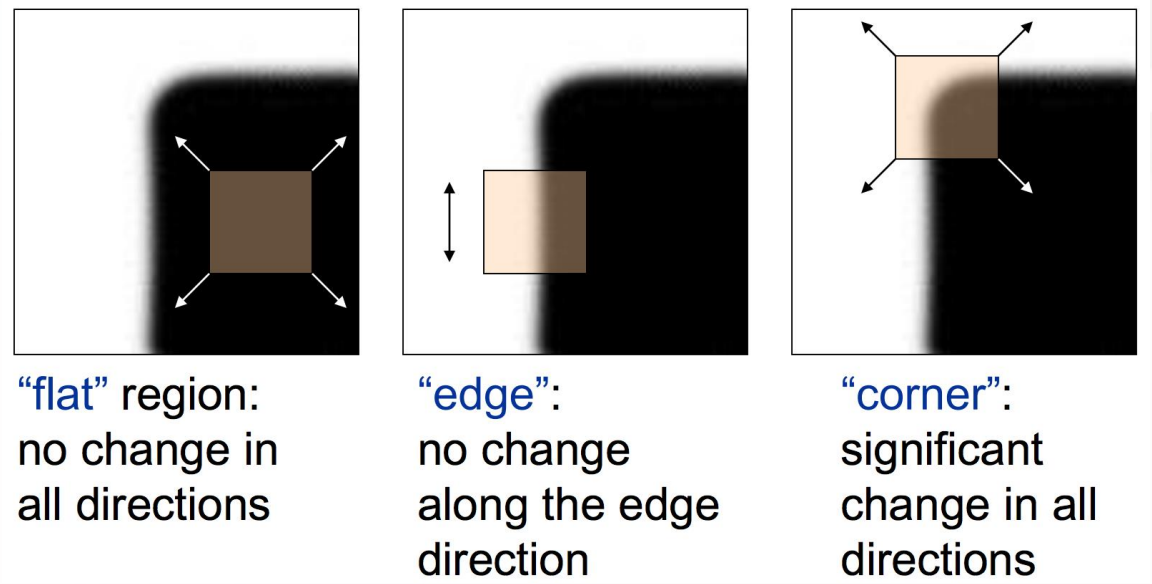
It clusters, or partitions the given image into K clusters or parts

Steps:

1. Choose the number of clusters, K.
2. Select K random pixels, the centroids.
3. Assign each pixel to the closest centroid → that forms K clusters.
4. Compute the new centroid of each cluster.
5. Reassign each pixel to the new closest centroid. If any reassignment took place, go to step 4, otherwise, the output is ready.

Feature Detection

In computer vision, key points and feature descriptors are crucial for tasks like image matching, stitching, object recognition, and tracking.



Feature Detection

Harris Corner Detection Algorithm

- The idea is to consider a small window around each pixel p in an image.
- Uniqueness can be measured by shifting each window by a small amount in a given direction and measuring the amount of change that occurs in the pixel values.

Steps

1. Compute the horizontal and vertical image derivatives
2. Compute the product of the derivatives (dx*dx, dy*dy, dx*dy) and find Harris matix.
3. Determine corner score, C using sensitivity parameter, k (0.04<=k<=0.06).

Feature Detection

The Harris corner score, C will be positive and large in corner regions, negative in edge regions, and small in flat regions.

Pros

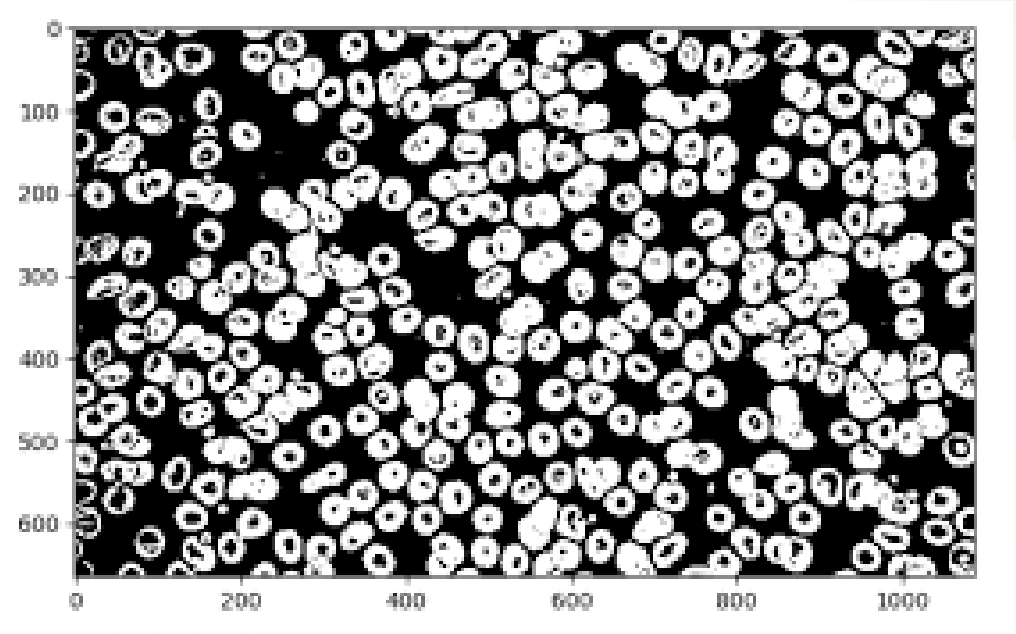
- Accurate corner detection on normal regions
- Rotation invariance
- Computationally efficient
- Simple to implement

Cons

- Not scale-invariant
- Sensitive to noise
- Poor performance on texture-less regions
- Fixed window size

Feature Detection

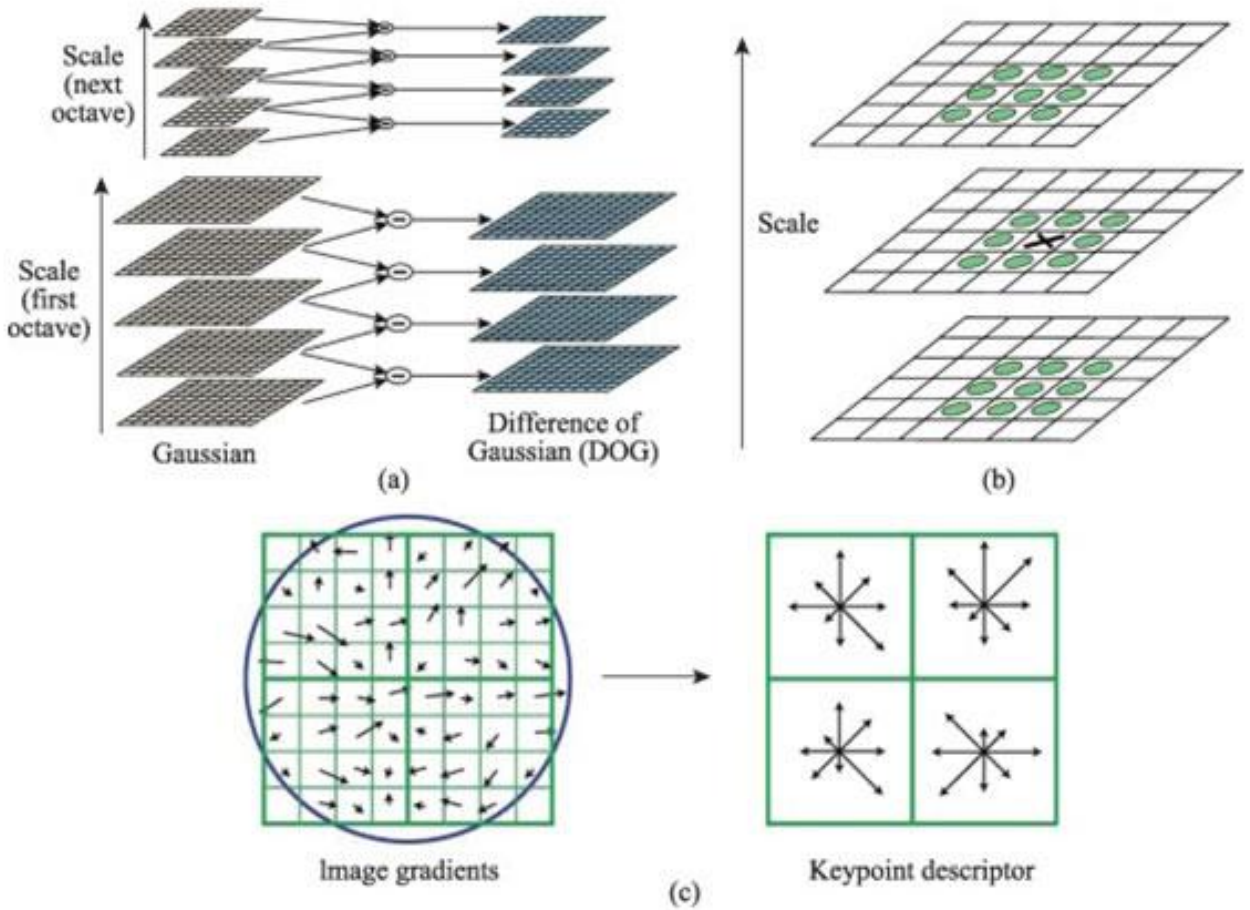
Blob is a group of connected pixels in an image that share some common property.



SIFT – Scale Invariant Feature Transform

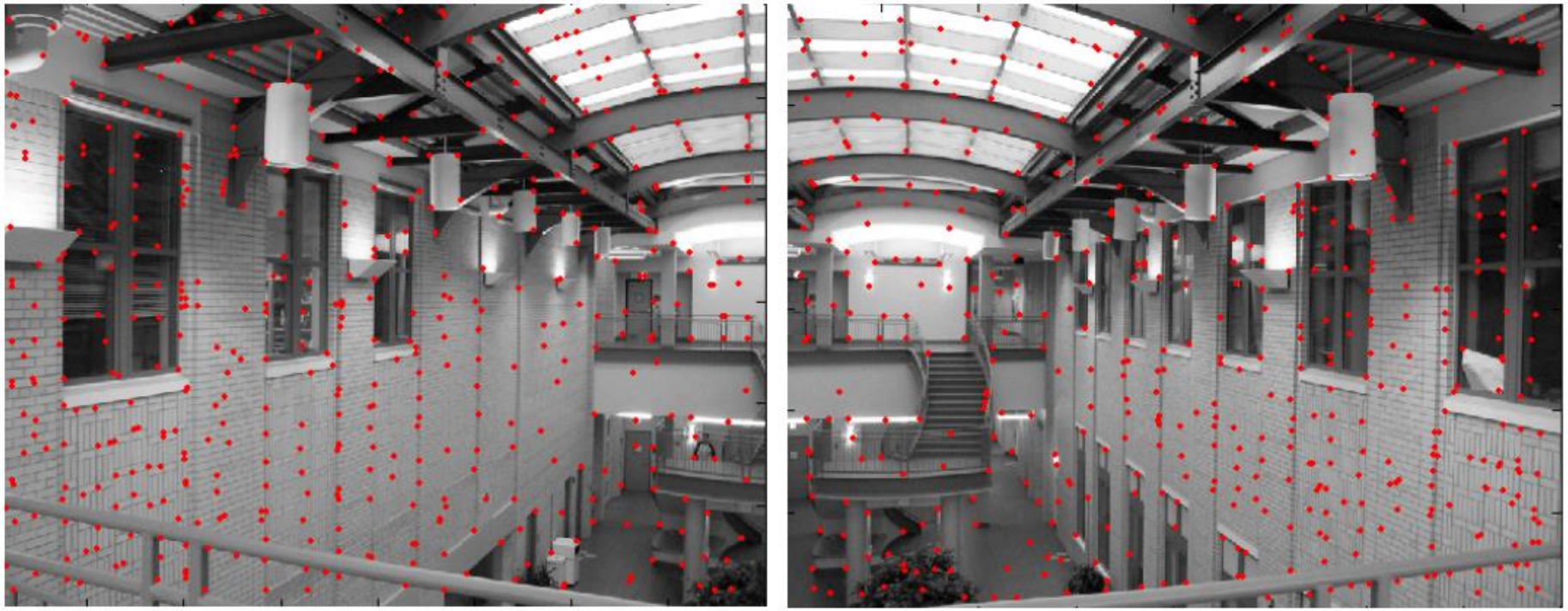
Overview

- Scale-invariant feature transform (SIFT) extracts distinctive features from images for reliable matching between different views of an object or scene.
- SIFT algorithm plays a crucial role in various computer vision tasks involving image analysis and feature detection.
- Developed by David Lowe in 1999, SIFT addresses challenges in recognizing objects across different scales, rotations, and viewpoints.



Construction of SIFT descriptor (a) Image pyramid. (b) Extrema detection for DOG pyramid. (c) Creation of keypoint descriptor.

Feature Matching



Feature Matching

- Feature matching in computer vision refers to the process of finding corresponding keypoints between two images of the same scene or object.
- Once SIFT has detected and described keypoints in two images, the next step is to match them.
- This involves comparing the descriptors of the keypoints from the two images to find the best matches.
- Common methods for comparing descriptors include calculating the Euclidean distance or sum of squared differences between the vectors.

Outliers

Outliers can hurt the quality of our parameter estimates,
e.g.,
– an erroneous pair of matching points from two images
– an edge point that is noise, or doesn't belong to the line we are fitting.

Lowe's Ratio Test

- Feature matching often produces many incorrect or weak matches.
- To remove false positives and improve the accuracy of matching, ***Lowe's Ratio Test*** can be used.

$$\text{Lowe's Ratio} = \frac{\text{Distance to best match}}{\text{Distance to second best match}}$$

- If the ratio is below a threshold, the match is considered good.
- Else, neglected.
- This helps eliminate ambiguous matches caused by repeated patterns or noise.

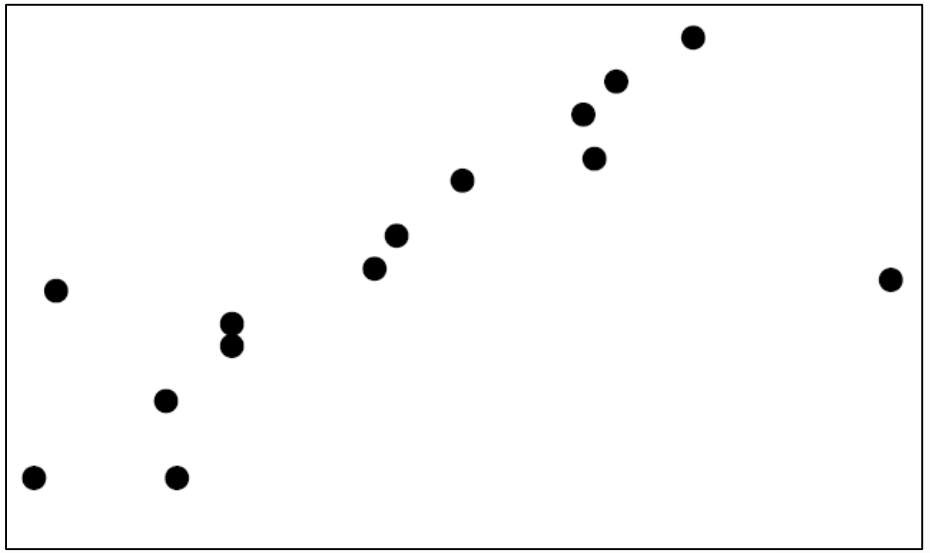
RANSAC

RANSAC (RANdom SAmple Consensus) is an iterative outlier detection algorithm to find the best fit for data that's full of noise or errors.

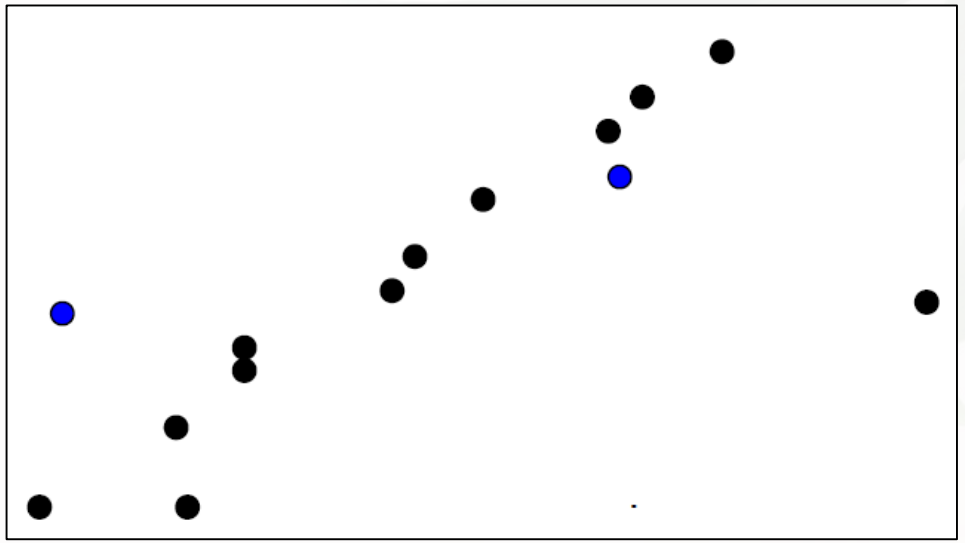
Algorithm

- Start with the input data
- Pick random samples
- Fit a mathematical model (line, curve, 3d shape, ...)
- Compute a cost by checking how many points fit the model
- Repeat until you found the model with the lowest cost (fitting largest number of inliers.

RANSAC Line Fitting Example

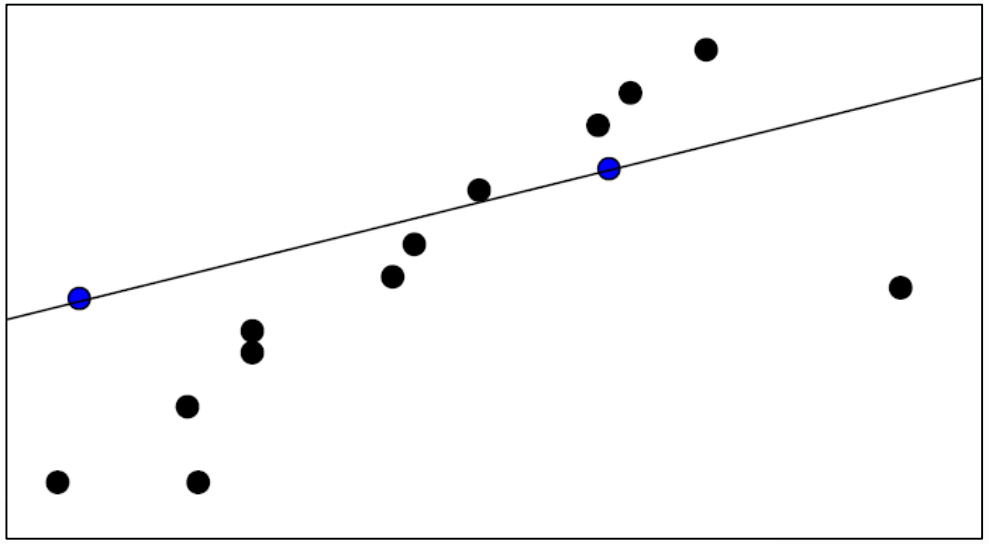


Task: Estimate best line

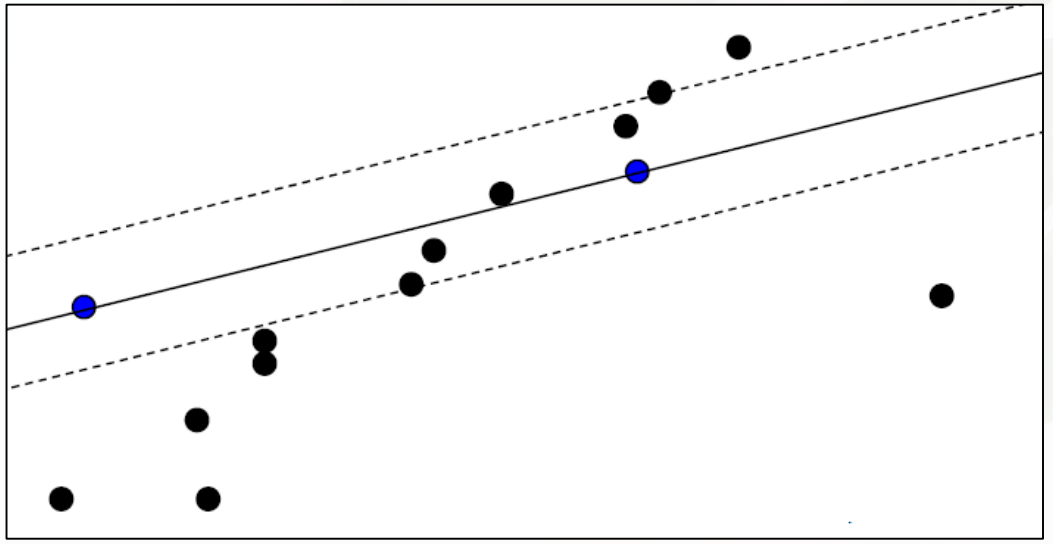


Sample two points

RANSAC Line Fitting Example



Fit a line



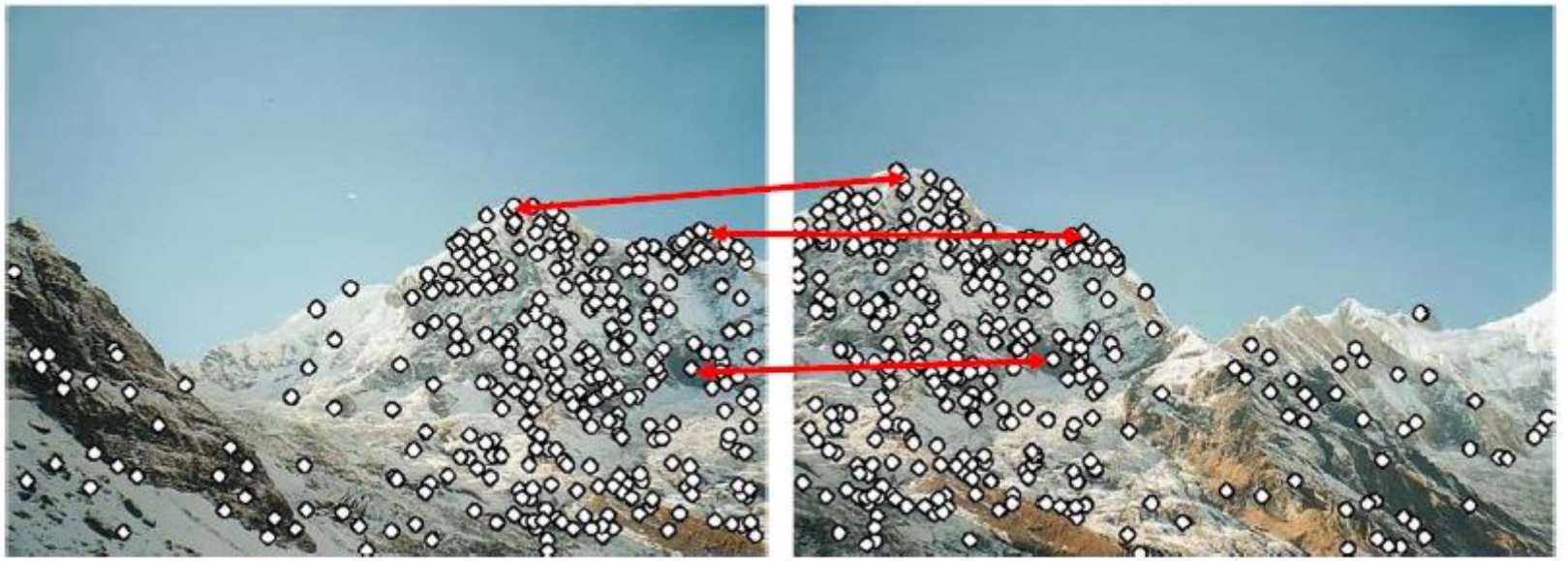
Total number of points
within a threshold of line.

Repeat, until get a good result

Applications

1. Object recognition enables identification of specific objects in complex scenes
2. Image registration aligns multiple images of the same scene taken from different viewpoints
3. 3D scene reconstruction creates three-dimensional models from multiple two-dimensional images
4. Motion tracking follows objects or features across video frames
5. Panorama stitching combines multiple images to create wide-angle panoramic views

Feature alignment



Feature alignment

- Feature matching is a fundamental technique in computer vision that involves identifying and aligning corresponding features across multiple images.
- Features can be distinctive elements in an image, such as edges, corners, or blobs, that can be consistently detected and described.
- By matching these features, computer vision systems can recognize objects, track movement, create panoramic images, and reconstruct 3D scenes from 2D images.

Feature alignment

Two broad approaches:

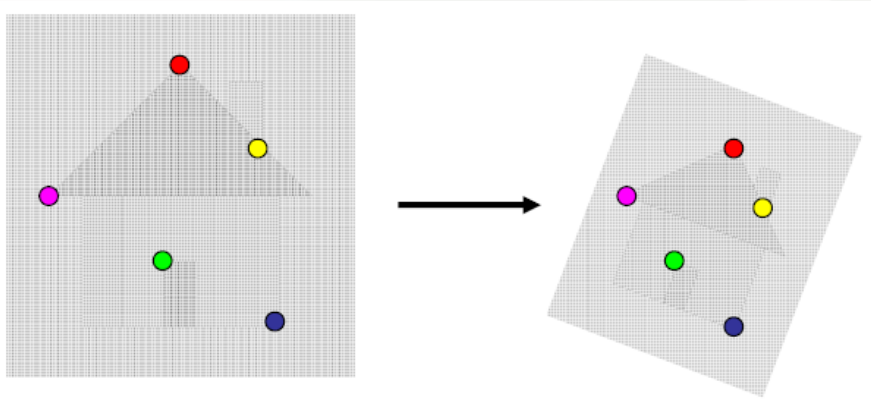
– Direct (pixel-based) alignment

• Search for alignment where most pixels agree

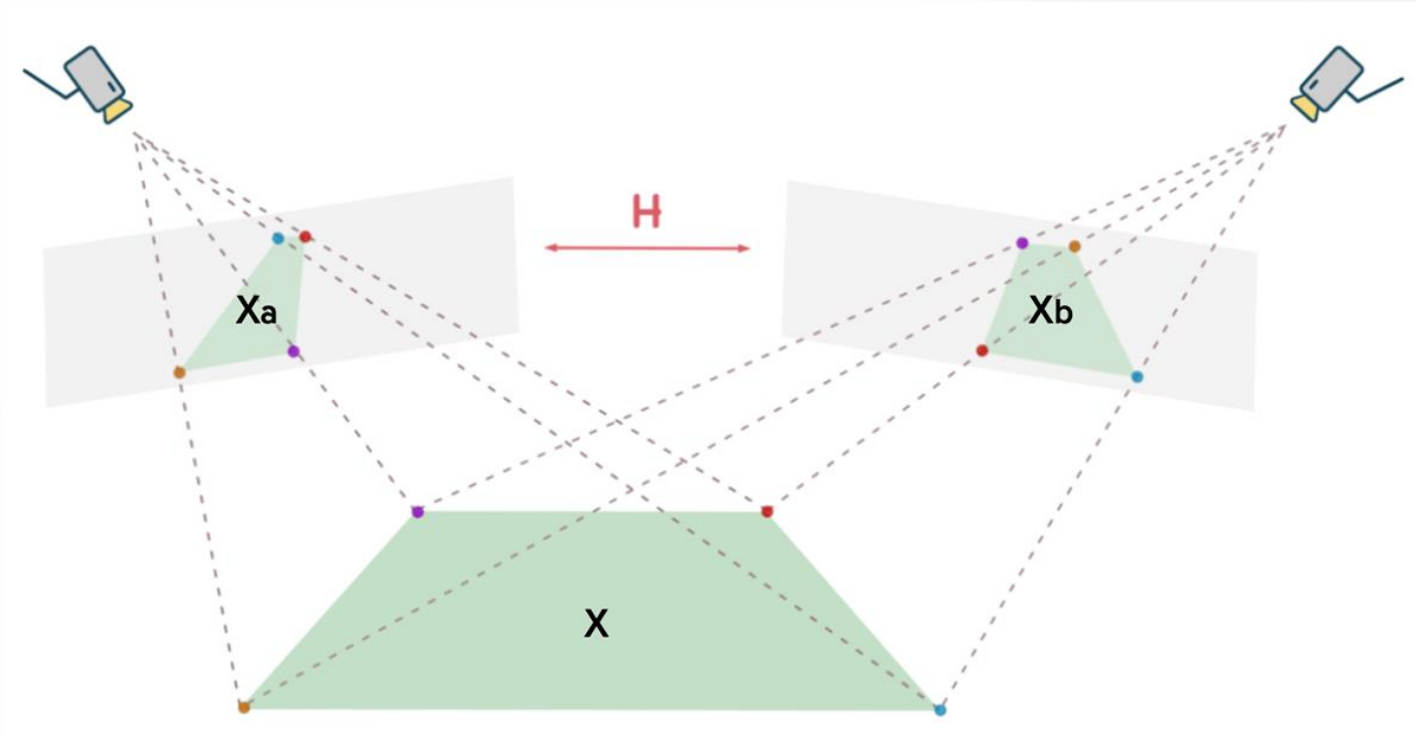
– Feature-based alignment

• Search for alignment where extracted features
agree

• Can be verified using pixel-based alignment



Homography



Homography estimation

To estimate H,

$$\begin{bmatrix} x_2 \\ y_2 \\ z_2 \end{bmatrix} = \begin{bmatrix} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \\ H_{31} & H_{32} & H_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} \Leftrightarrow \mathbf{x}_2 = H\mathbf{x}_1$$

In inhomogenous coordinates,

$$x_2' = \frac{H_{11}x_1 + H_{12}y_1 + H_{13}z_1}{H_{31}x_1 + H_{32}y_1 + H_{33}z_1}$$
$$y_2' = \frac{H_{21}x_1 + H_{22}y_1 + H_{23}z_1}{H_{31}x_1 + H_{32}y_1 + H_{33}z_1}$$

Homography estimation

Set z1 = 1 and rearrange:

$$x_2'(H_{31}x_1 + H_{32}y_1 + H_{33}) = H_{11}x_1 + H_{12}y_1 + H_{13}$$
$$y_2'(H_{31}x_1 + H_{32}y_1 + H_{33}) = H_{21}x_1 + H_{22}y_1 + H_{23}$$

We want to solve for H. Even though these inhomogeneous equations involve the coordinates nonlinearly, the coefficients of H appear linearly.

The form, $Ax = 0$ ..... (1)
is known as the Homogeneous Linear Least Squares problem. It is similar in appearance to the inhomogeneous linear least squares problem,

$Ax = b$ ..... (2)

in which case we solve for x using the pseudoinverse or inverse of A. This won't work with Equation (1). Instead we solve it using Singular Value Decomposition (SVD).

Thank you