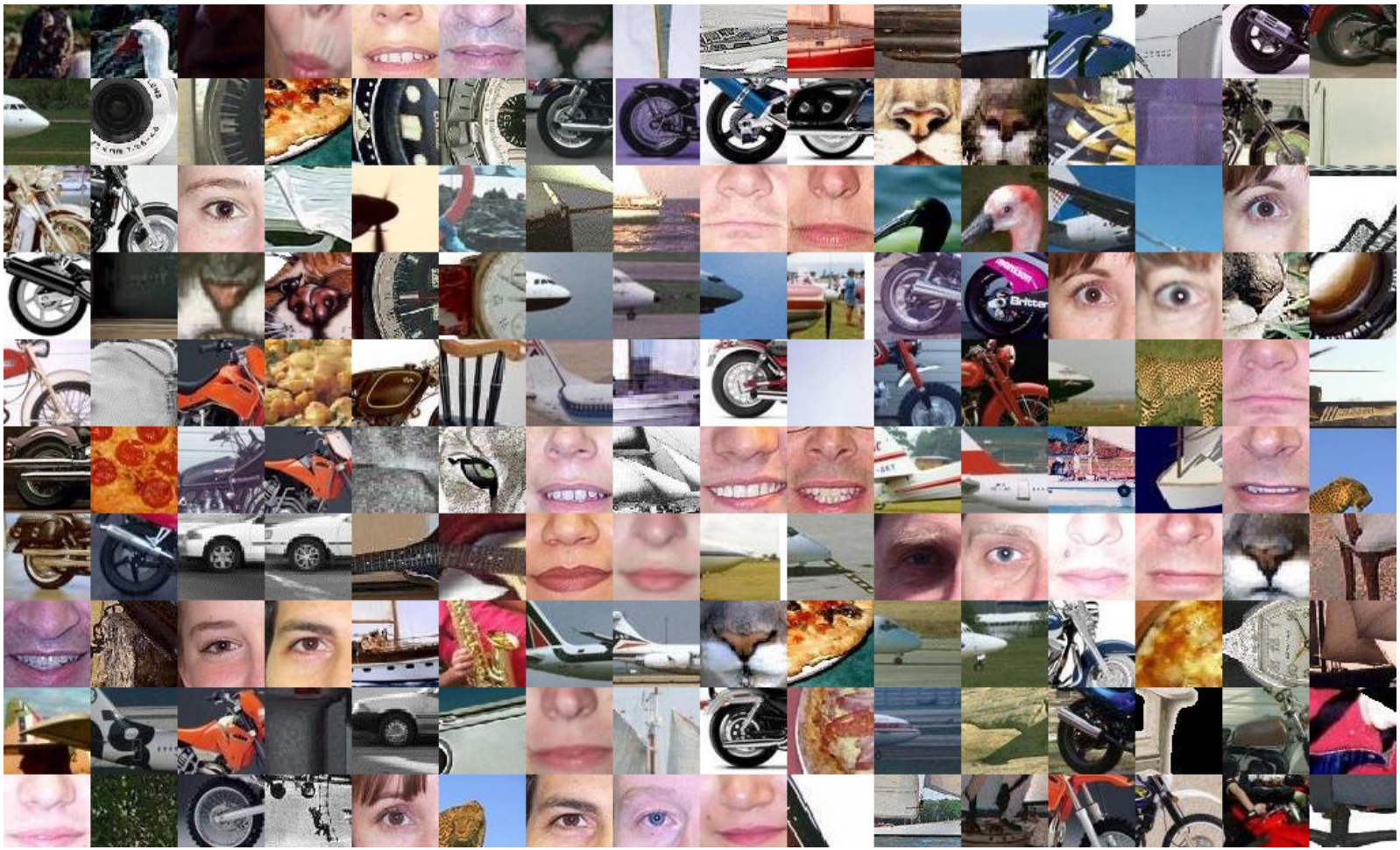
Feature detectors and descriptors

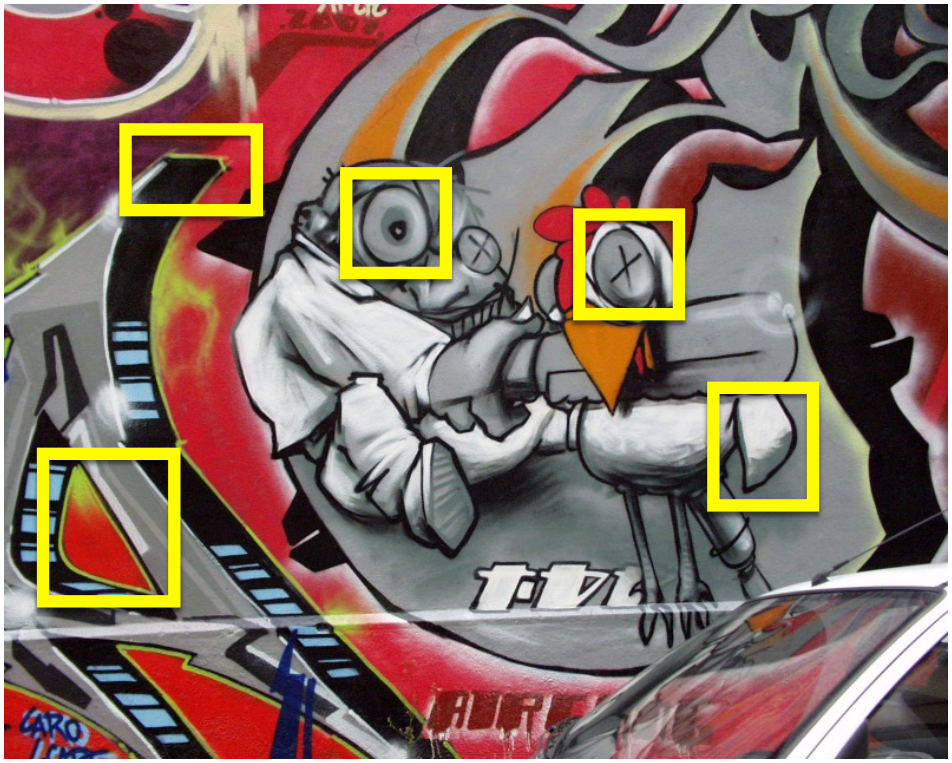


Computer Vision
Fall 2022, Lecture 7

Why do we need feature descriptors?

A

B



If we know where the <u>good</u> features are, how do we <u>match</u> them?


How do we describe an image patch?
Patches with similar content should have similar descriptors.

Designing feature descriptors

Goal: interest operator repeatability

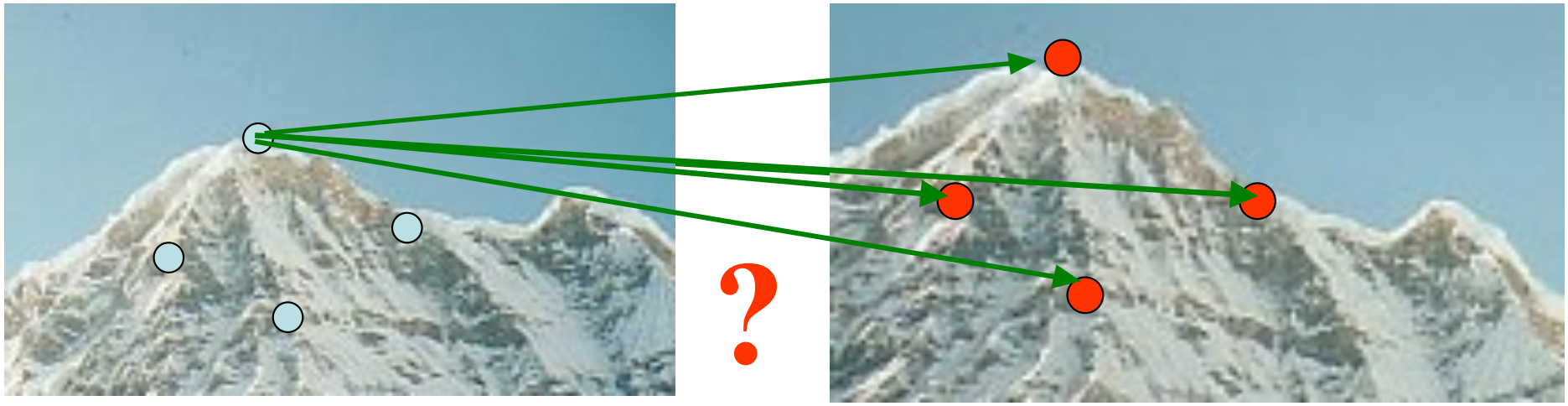
- We want to detect (at least some of) the same points in both images.

No chance to find true matches!

- Yet we have to be able to run the detection procedure independently per image.

Goal: descriptor distinctiveness

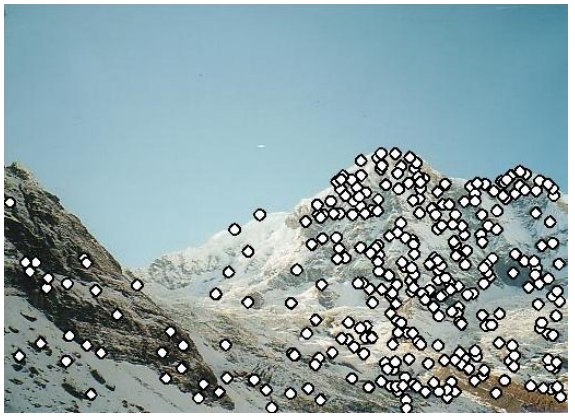
- We want to be able to reliably determine which point goes with which.



- Must provide some invariance to geometric and photometric differences between the two views.

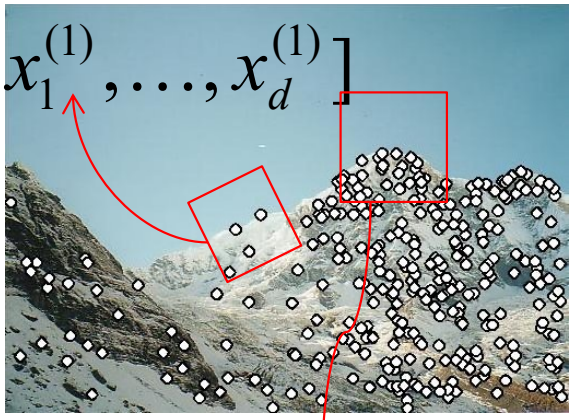
Local invariant features: Outline

1) **Detection**: Identify the interest points



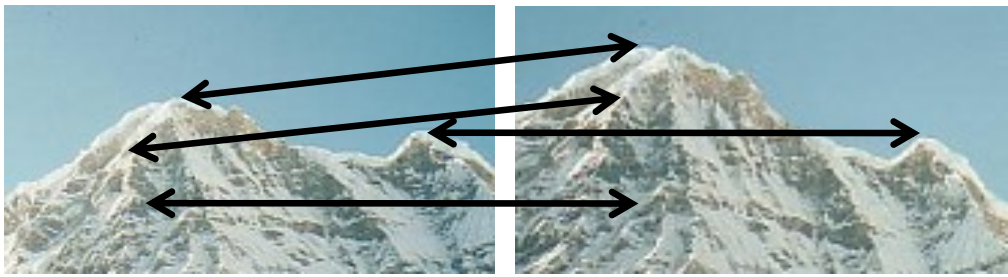
2) **Description**: Extract vector feature descriptor surrounding each interest point.

$$\mathbf{x}_1 = [x_1^{(1)}, \ldots, x_d^{(1)}]$$



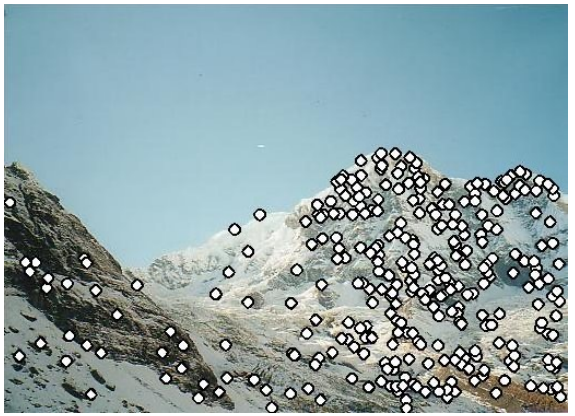
$$\mathbf{x}_2 = [x_1^{(2)}, \ldots, x_d^{(2)}]$$

3) **Matching**: Determine correspondence between descriptors in two views



Local features: main components

1) **Detection**: Identify the interest points



2) Description:Extract vector feature descriptor surrounding each interest point.

3) Matching: Determine correspondence between descriptors in two views

We want invariance!!!

- To illumination
- To scale
- To rotation
- To affine
- To perspective projection

Types of invariance

- Illumination

Types of invariance

- Illumination
- Scale

Types of invariance

- Illumination
- Scale
- Rotation

Types of invariance

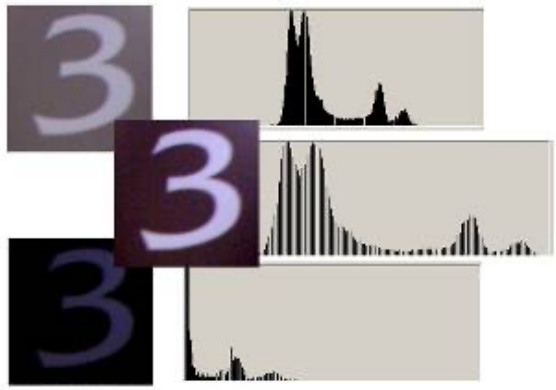
- Illumination
- Scale
- Rotation
- Affine (view point change)

Types of invariance

- Illumination
- Scale
- Rotation
- Affine
- Full Perspective

How to achieve illumination invariance

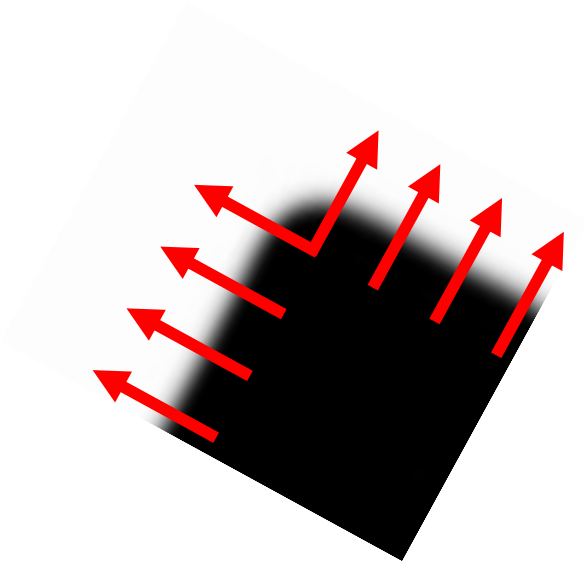
- The easy way (normalized)



Recall: Corners as distinctive interest points

Since M is symmetric, we have

$$M = X \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} X^T$$

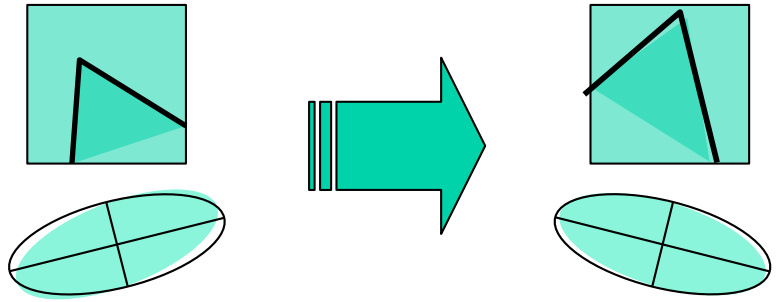


$$Mx_i = \lambda_i x_i$$

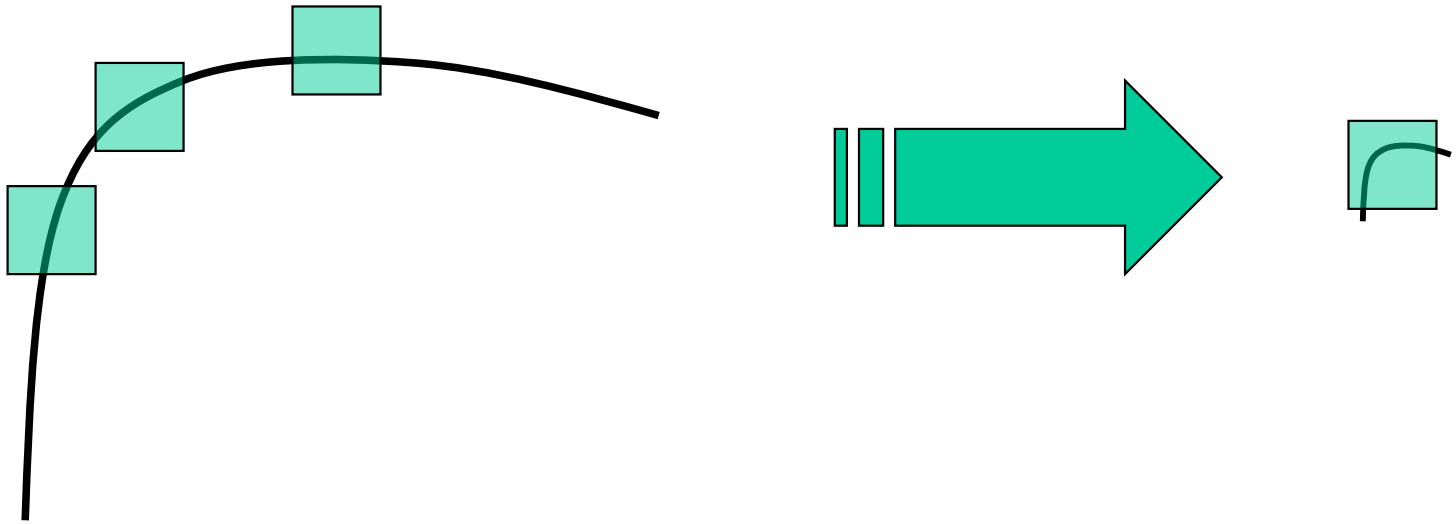
The eigenvalues of M reveal the amount of intensity change in the two principal orthogonal gradient directions in the window.

Properties of the Harris corner detector

Rotation invariant? Yes



Scale invariant? No



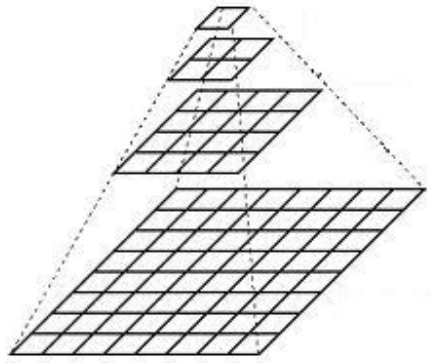
All points will be classified as edges

Corner !

Scale invariant interest points

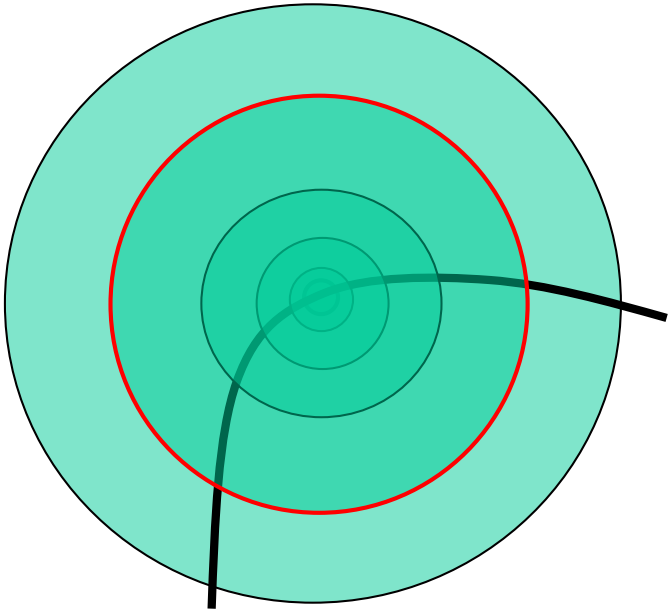
How can we independently select interest points in each image, such that the detections are repeatable across different scales?

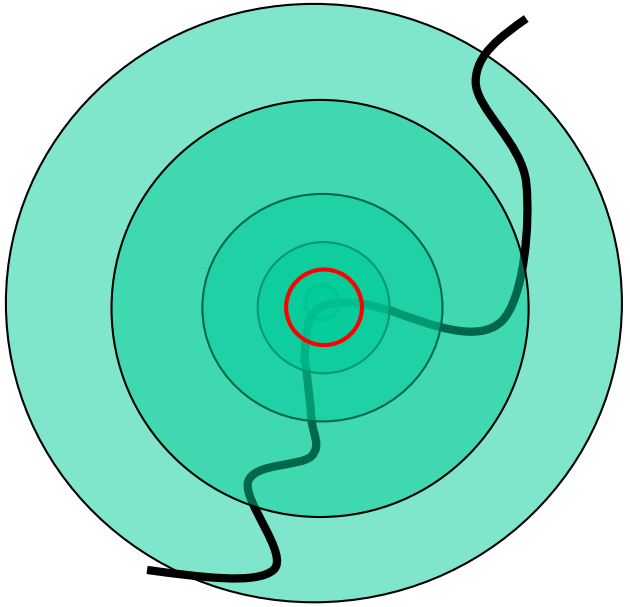
Image Pyramid -> Multi-Scale

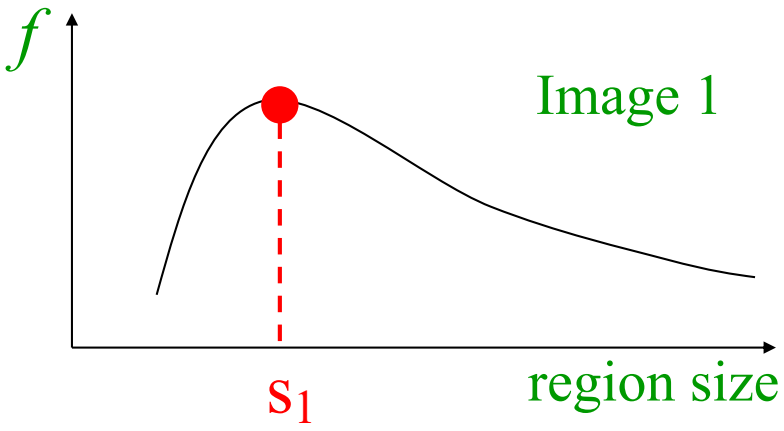


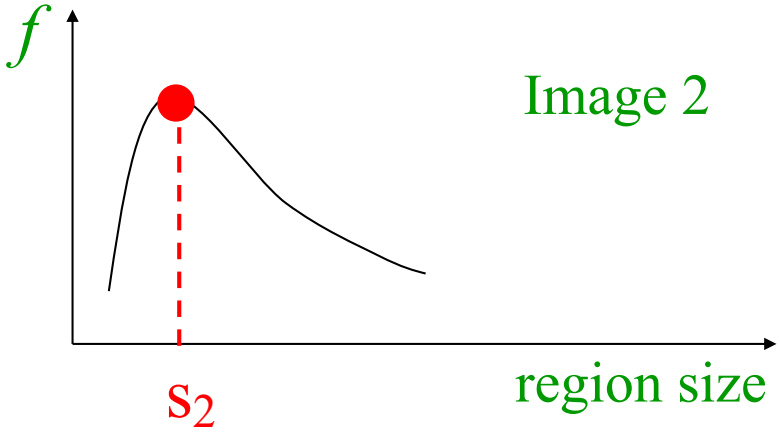
Intuitively…

Find local maxima in both **position** and **scale**



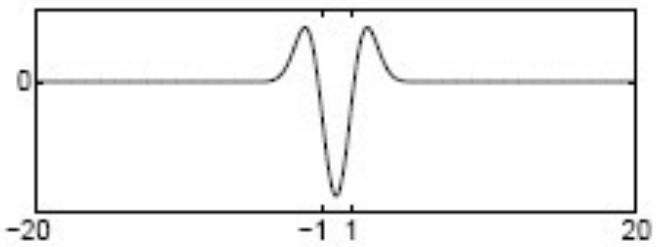


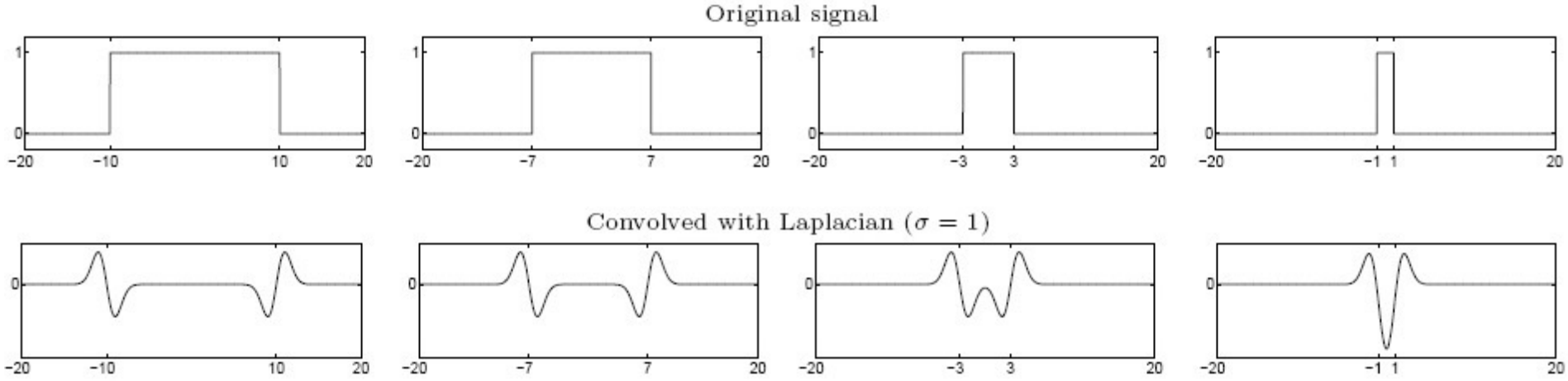




Formally…

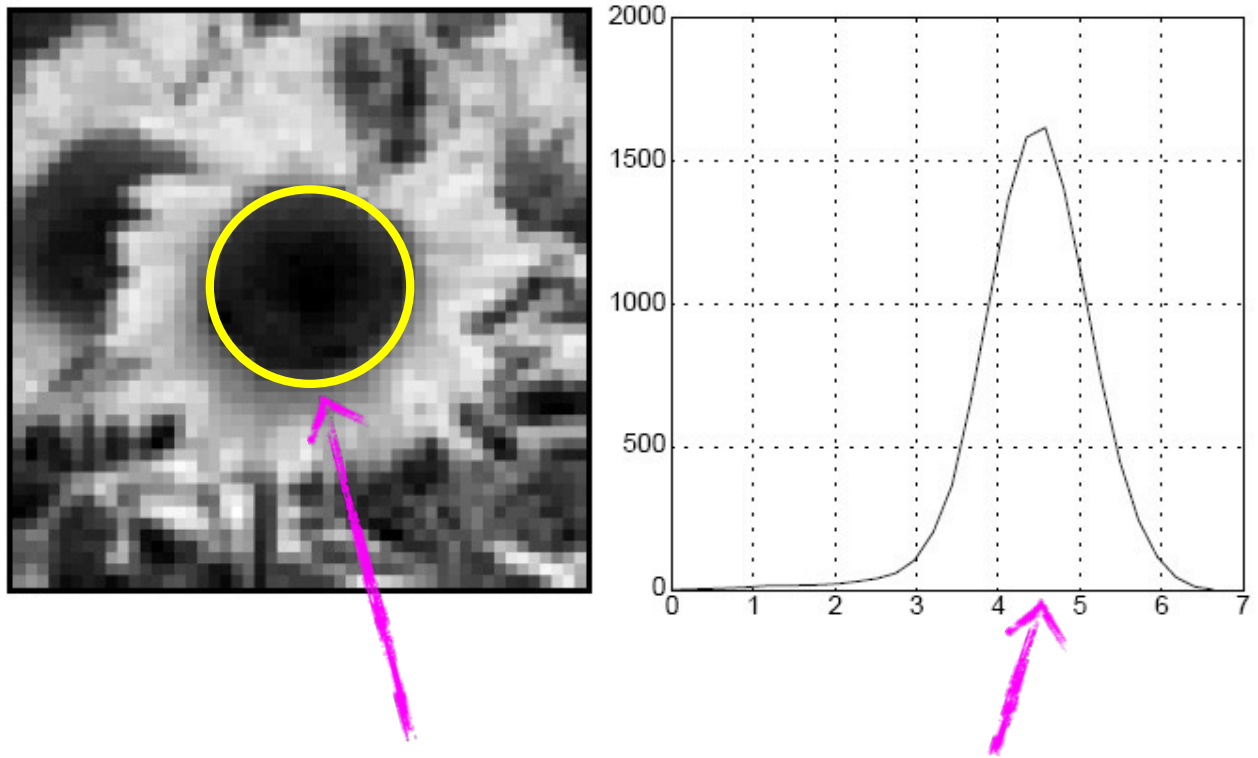
Laplacian filter





Highest response when the signal has the same **characteristic scale** as the filter

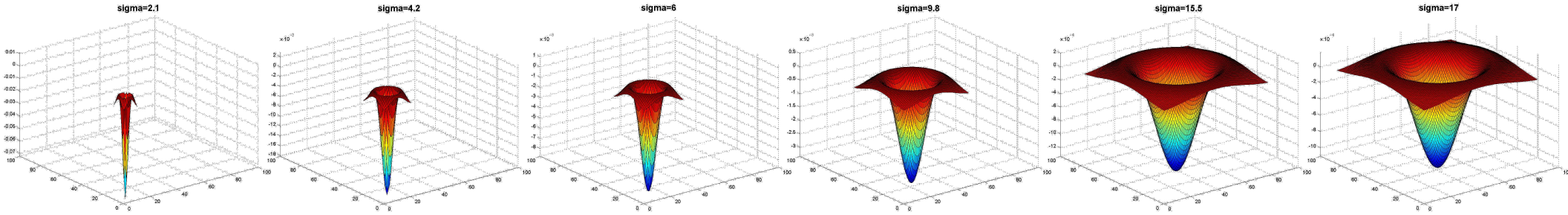
characteristic scale - the scale that produces peak filter response



characteristic scale

we need to search over characteristic scales

What happens if you apply different Laplacian filters?



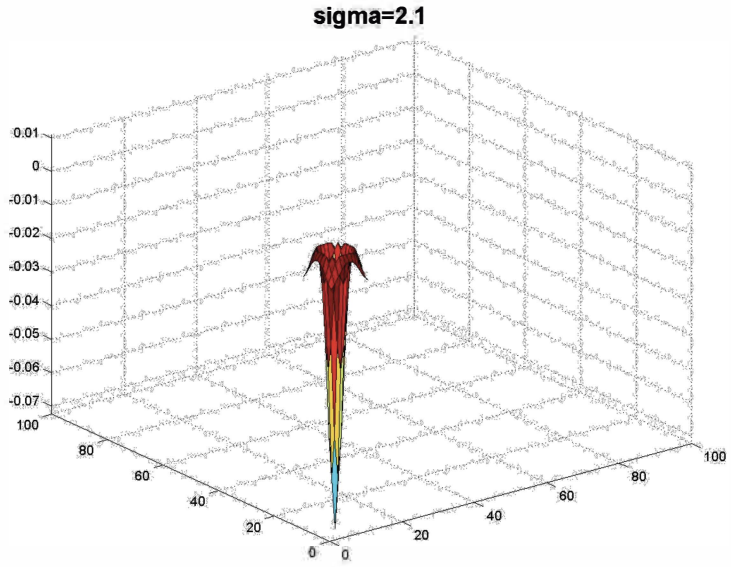
Full size

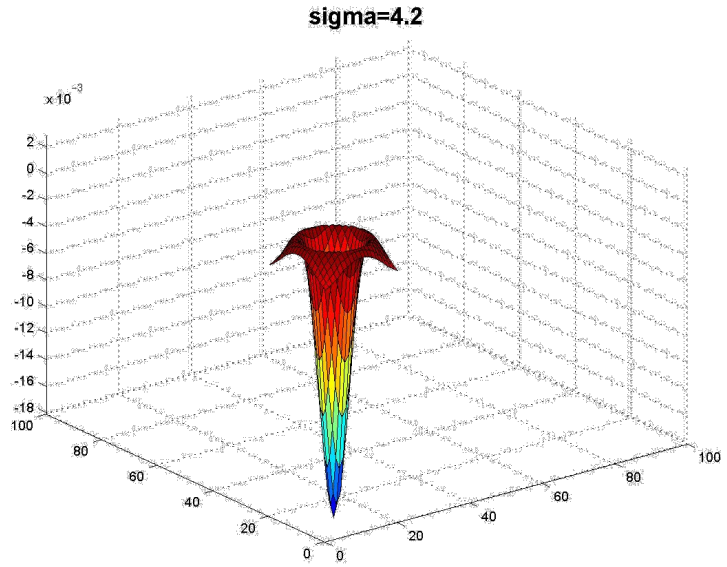
3/4 size

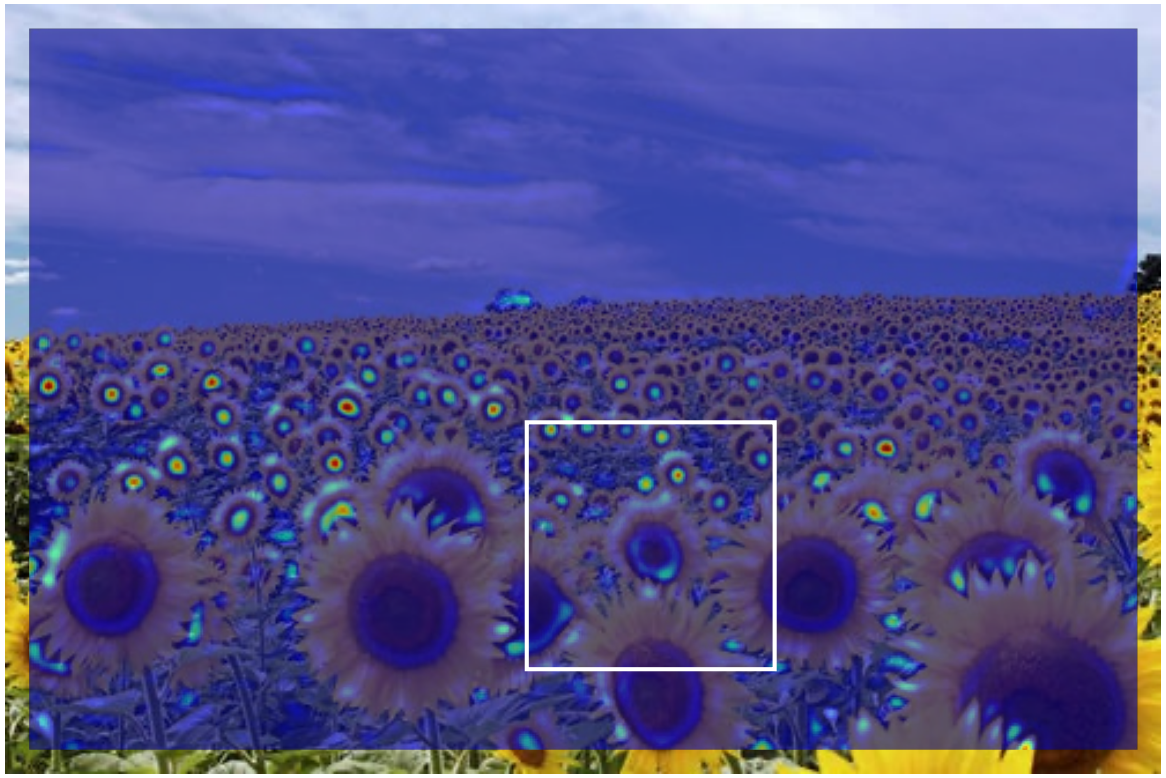
What happened when you applied different Laplacian filters?

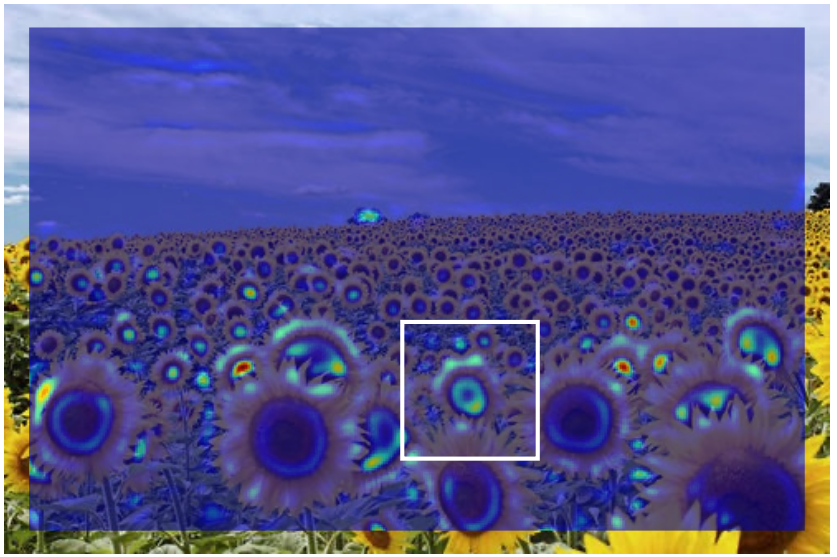
Full size

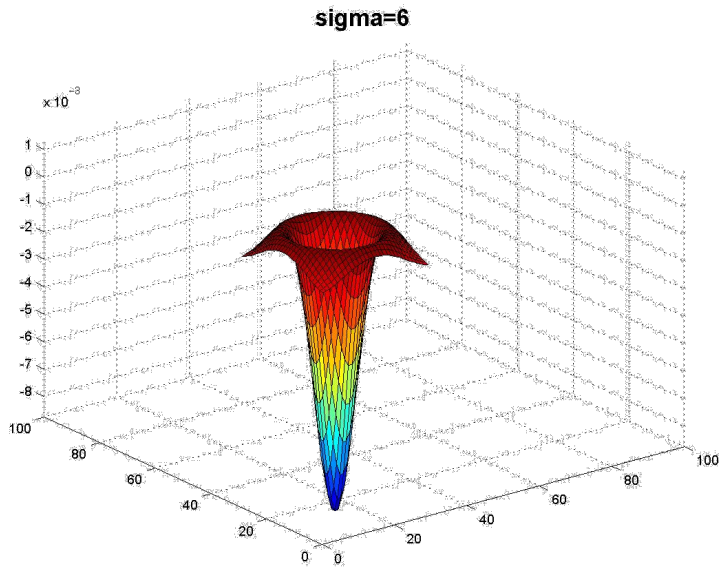
3/4 size

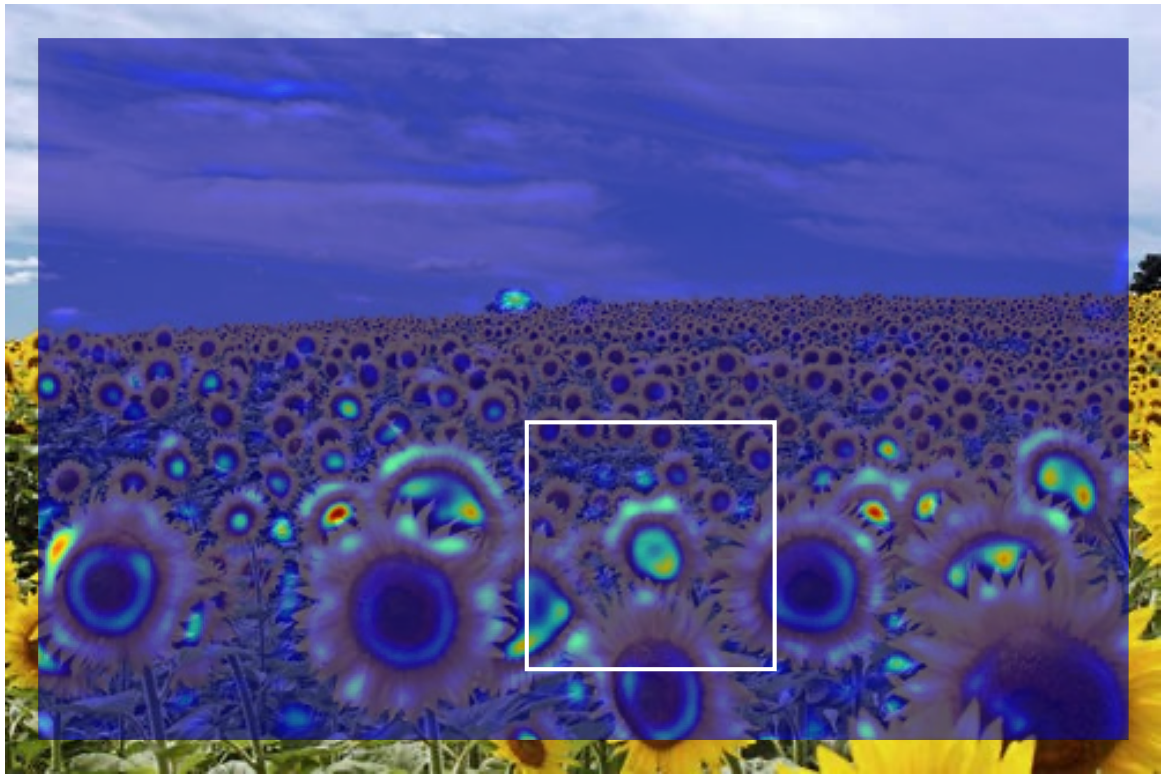

sigma=2.1
0.01
0
-0.01
-0.02
-0.03
-0.04
-0.05
-0.06
-0.07
100
80
60
40
20
0
0
20
40
60
80
100

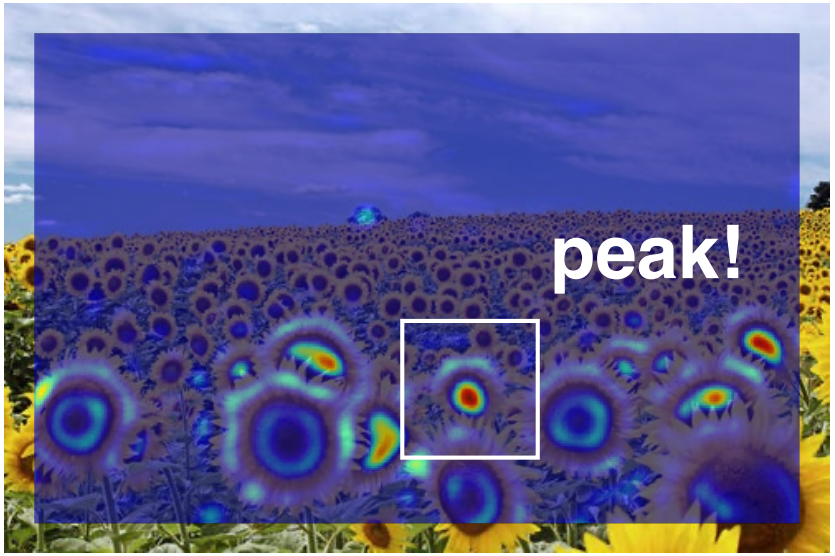

sigma=4.2
x 10^-3
2
0
-2
-4
-6
-8
-10
-12
-14
-16
-18
100
80
60
40
20
0
0
20
40
60
80
100

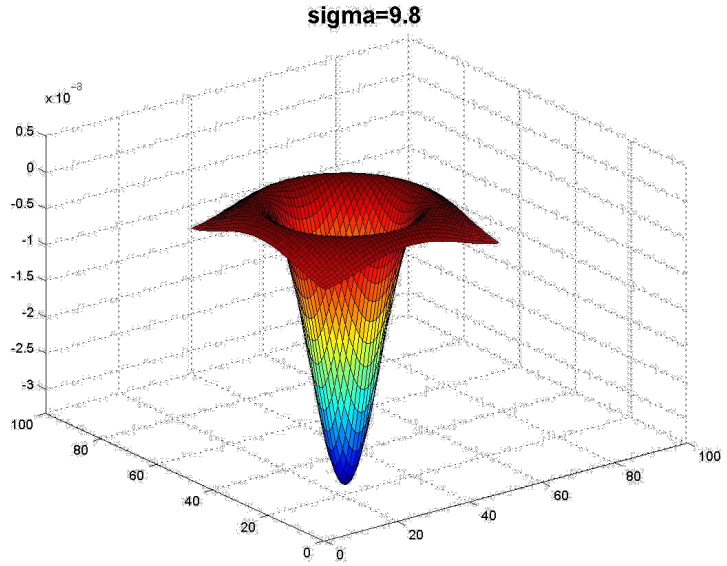




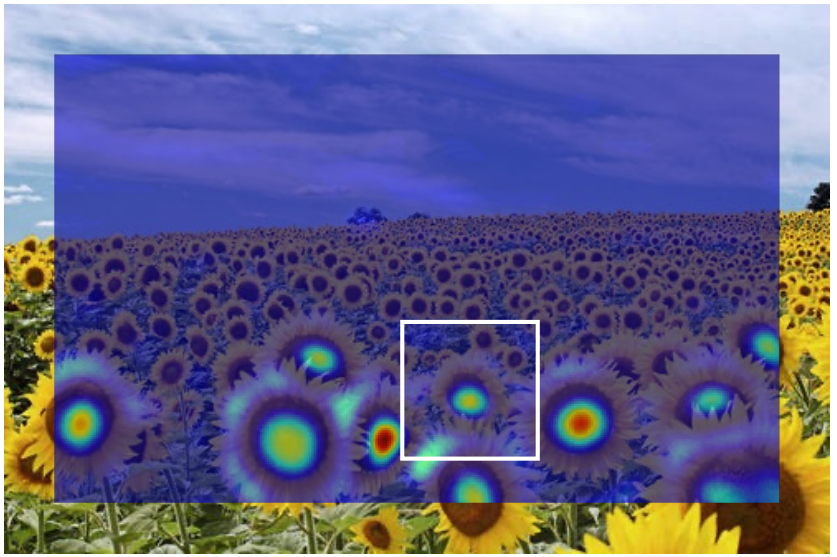

sigma=6
x 10^-3
1
0
-1
-2
-3
-4
-5
-6
-7
-8
100
80
60
40
20
0
0
20
40
60
80
100

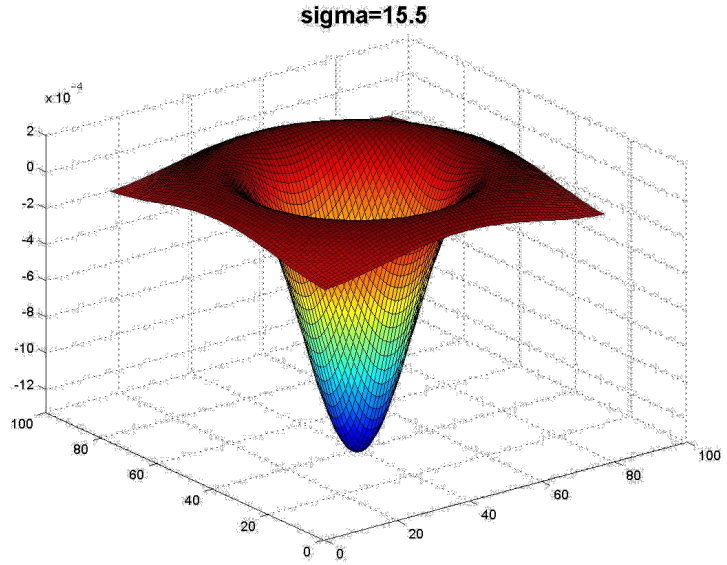


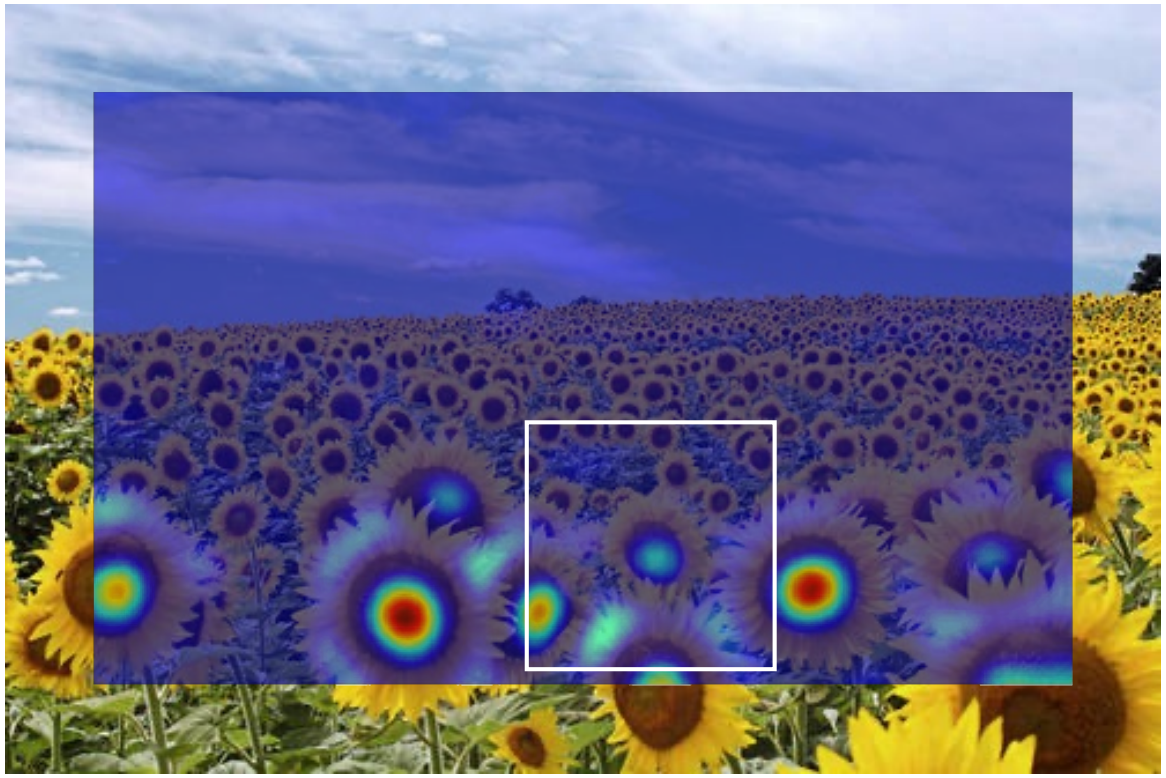

peak!

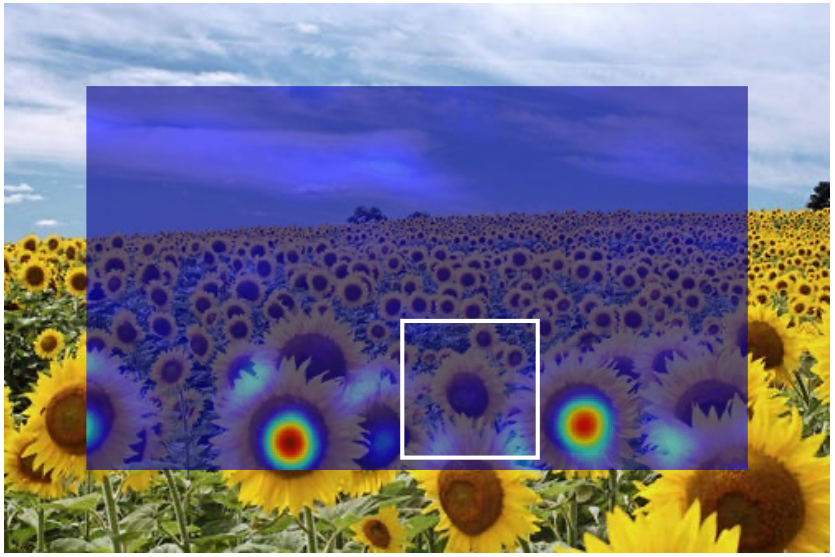

sigma=9.8
x 10^-3
0.5
0
-0.5
-1
-1.5
-2
-2.5
-3
100
80
60
40
20
0
0
20
40
60
80
100

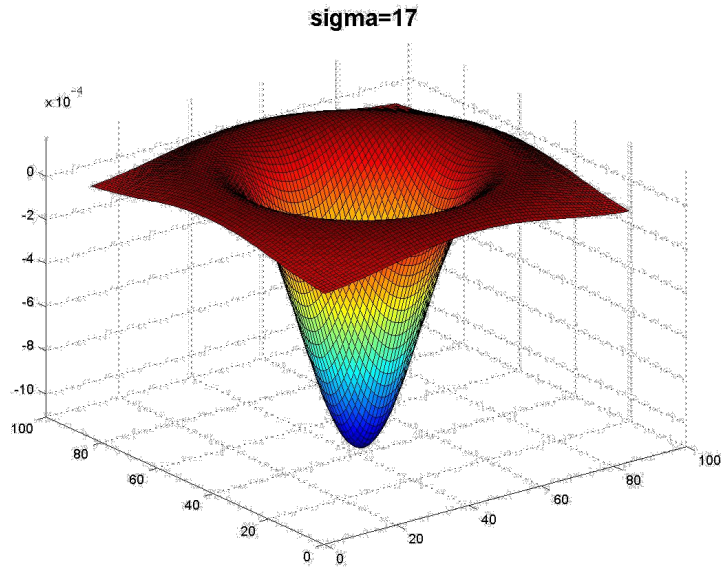

peak!

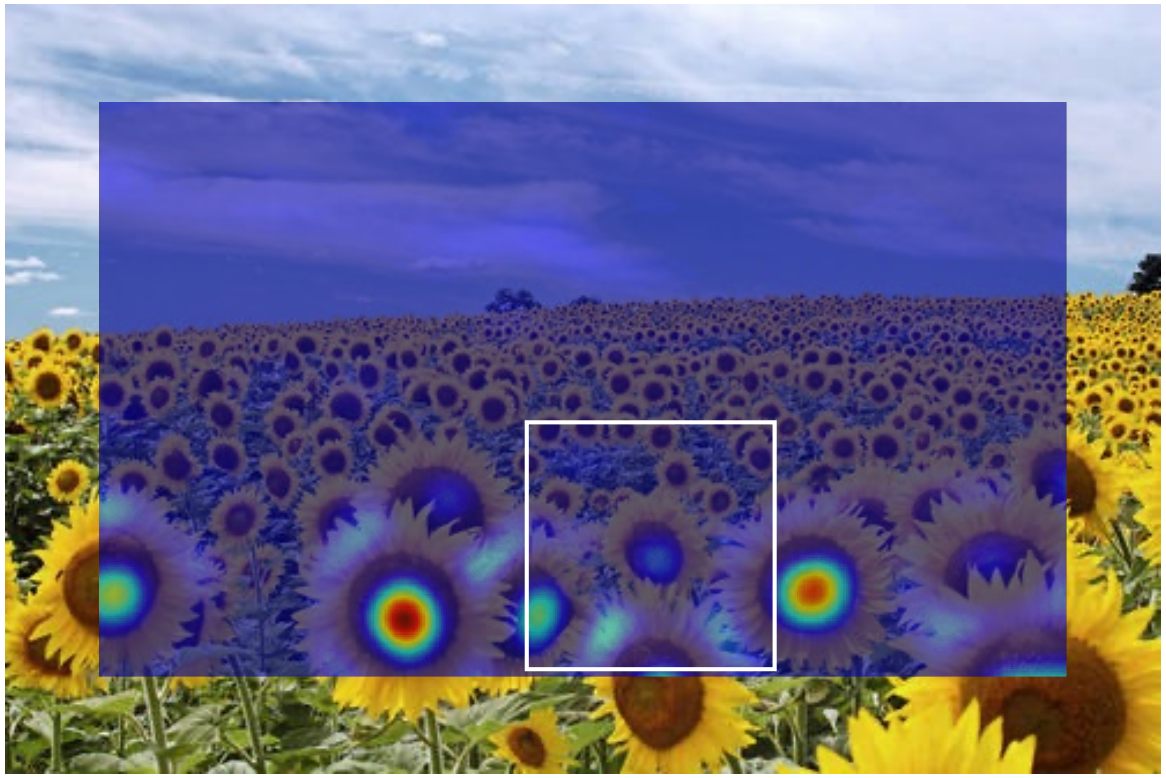


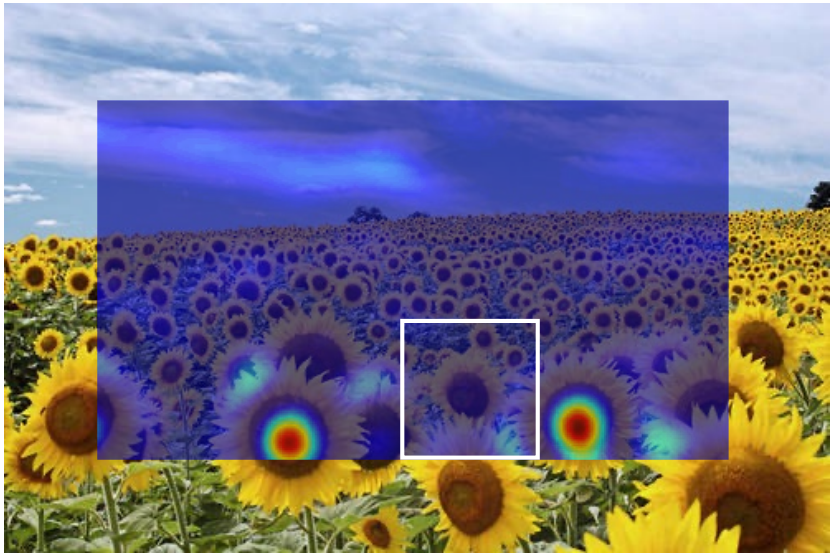

sigma=15.5
x 10^-4
2
0
-2
-4
-6
-8
-10
-12
100
80
60
40
20
0
0
20
40
60
80
100






sigma=17
x 10^-4
0
-2
-4
-6
-8
-10
100
80
60
40
20
0
0
20
40
60
80
100



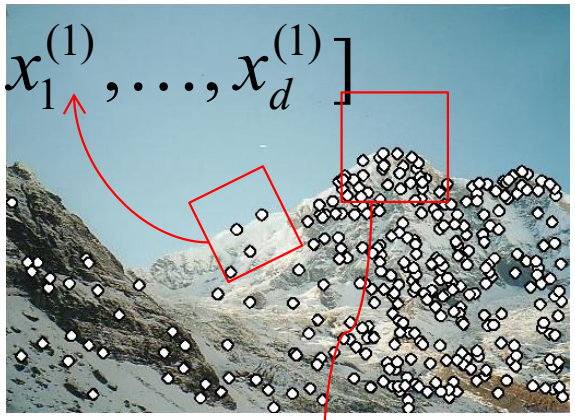


Local features: main components

1) Detection: Identify the interest points

2) **Description**:Extract vector feature descriptor surrounding each interest point.

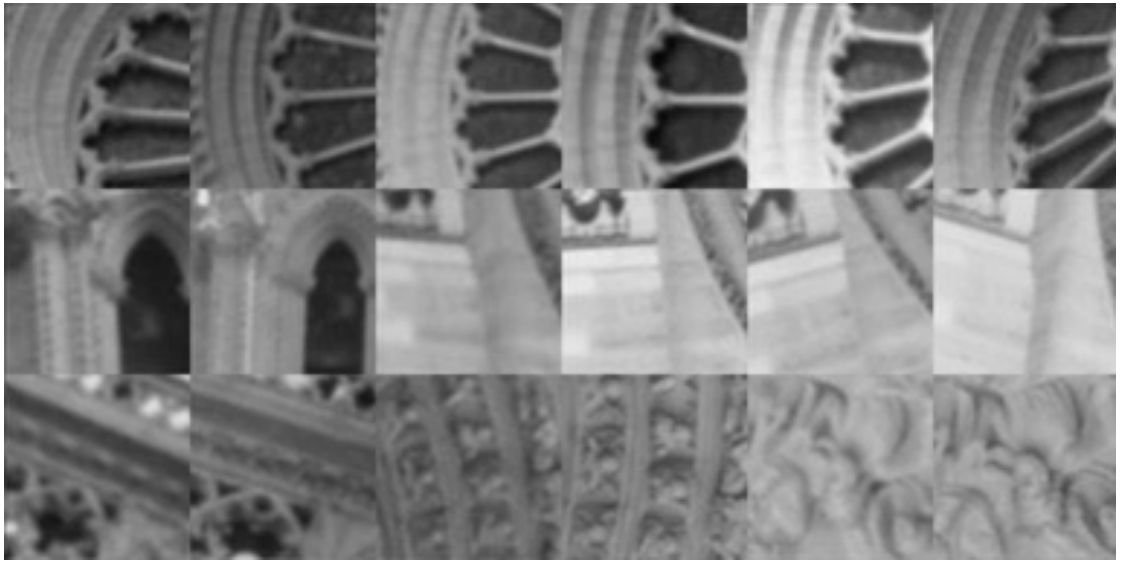
$$\mathbf{x}_1 = [x_1^{(1)}, \ldots, x_d^{(1)}]$$



$$\mathbf{x}_2 = [x_1^{(2)}, \ldots, x_d^{(2)}]$$

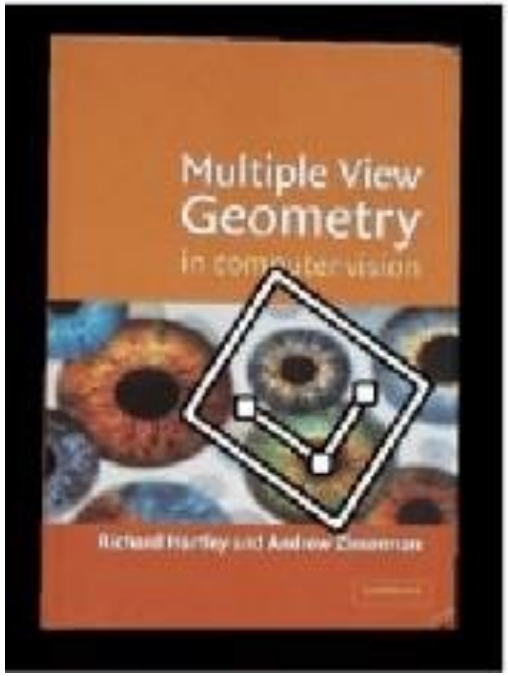
3) Matching: Determine correspondence between descriptors in two views

What is the best descriptor for an image feature?



Photometric transformations

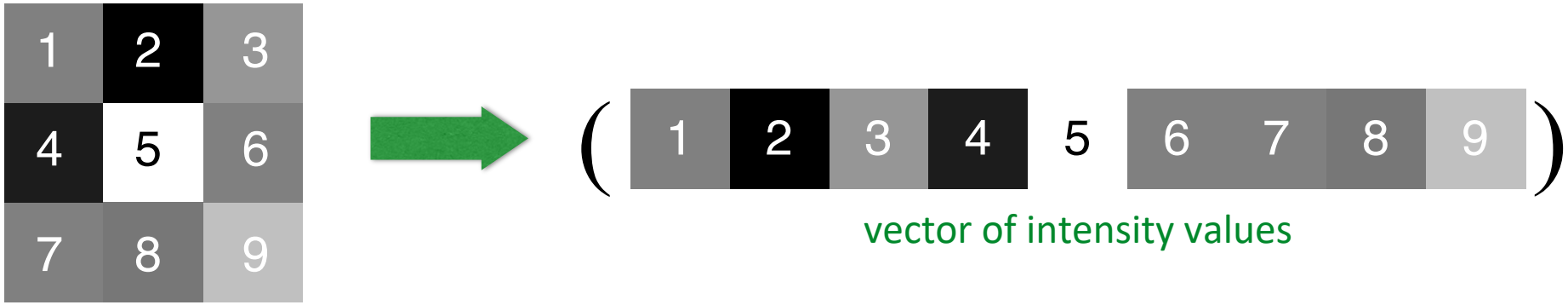
Geometric transformations



objects will appear at different scales,
translation and rotation

Image patch

Just use the pixel values of the patch!

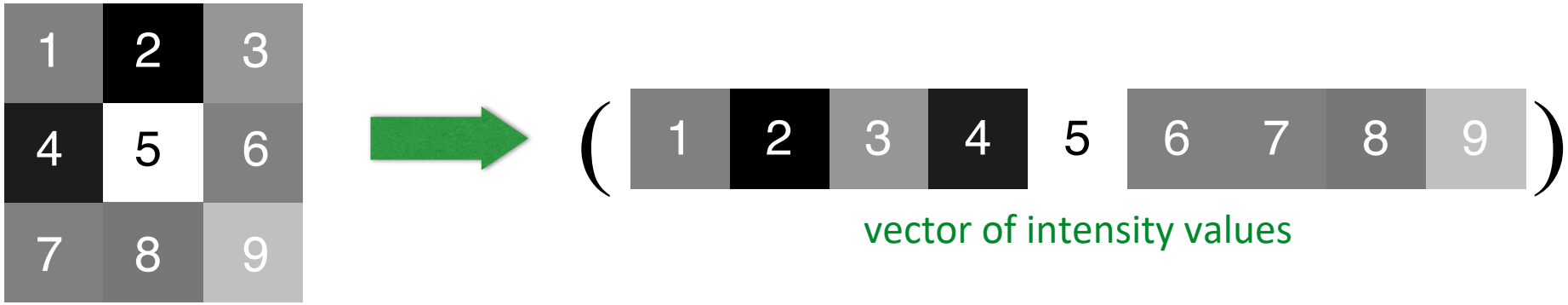


Perfectly fine if geometry and appearance is unchanged

(a.k.a. template matching)

Image patch

Just use the pixel values of the patch!



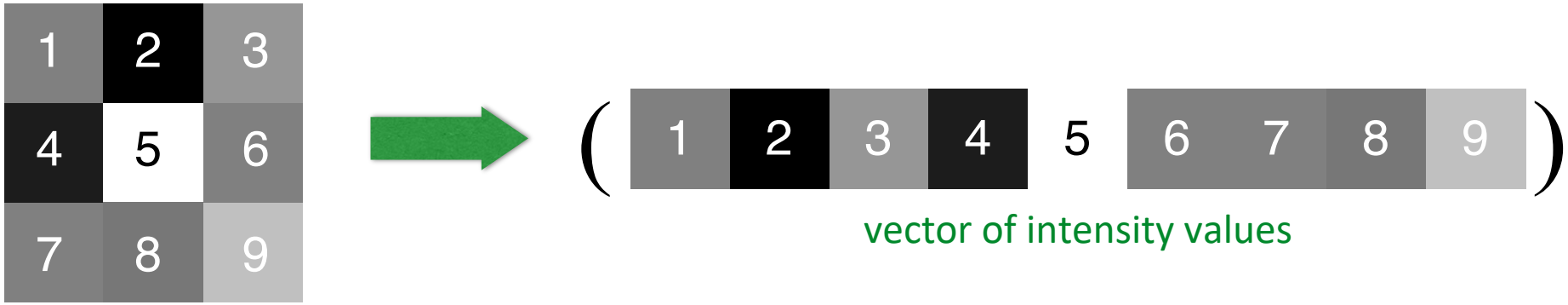
Perfectly fine if geometry and appearance is unchanged

(a.k.a. template matching)

What are the problems?

Image patch

Just use the pixel values of the patch!



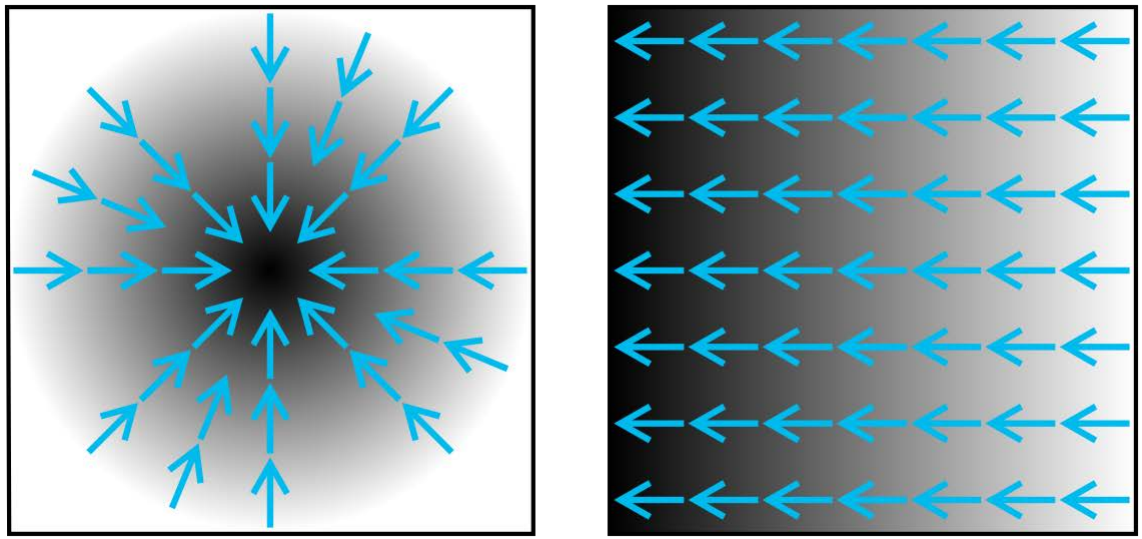
Perfectly fine if geometry and appearance is unchanged

(a.k.a. template matching)

What are the problems?

How can you be less sensitive to absolute intensity values?

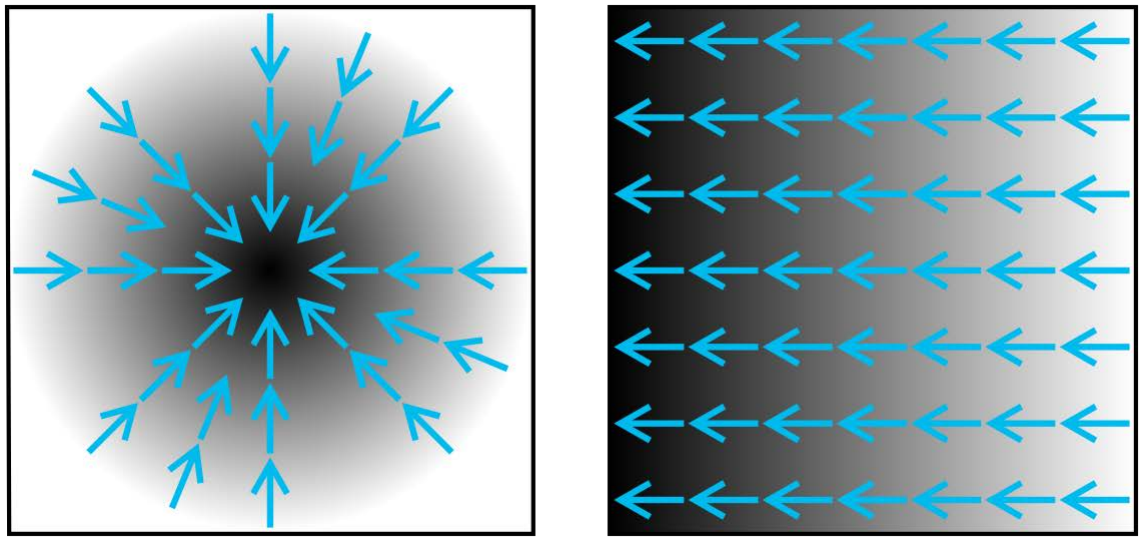
Image gradients



Feature is invariant to absolute intensity values

What are the problems?

Image gradients



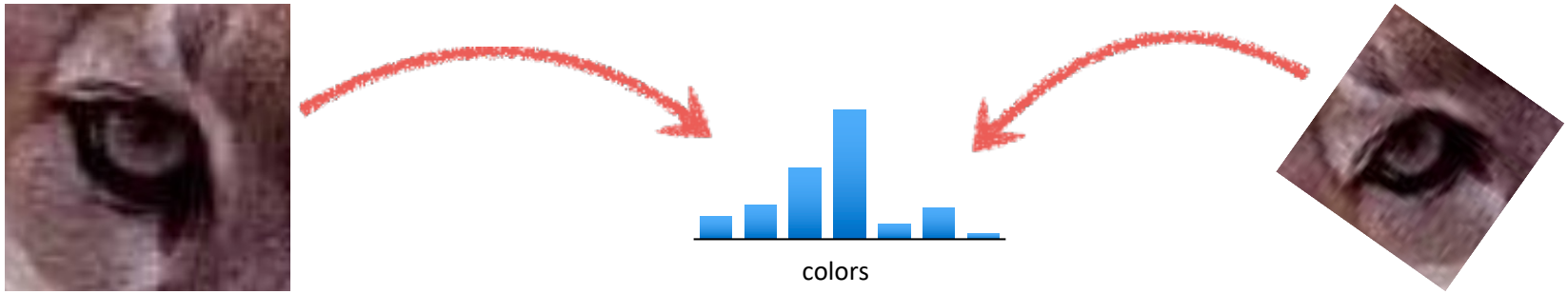
Feature is invariant to absolute intensity values

What are the problems?

How can you be less sensitive to deformations?

Color histogram

Count the colors in the image using a histogram



Invariant to changes in scale and rotation

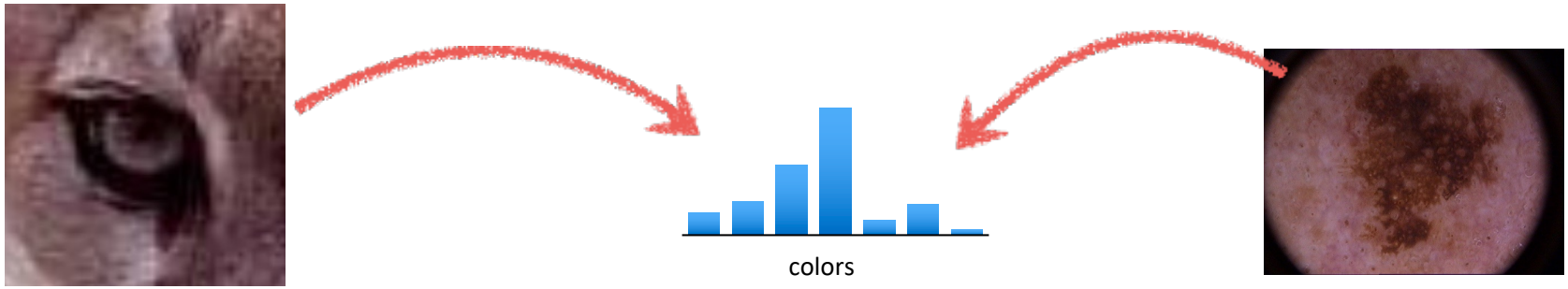
What are the problems?

RGB

HSV

Color histogram

Count the colors in the image using a histogram

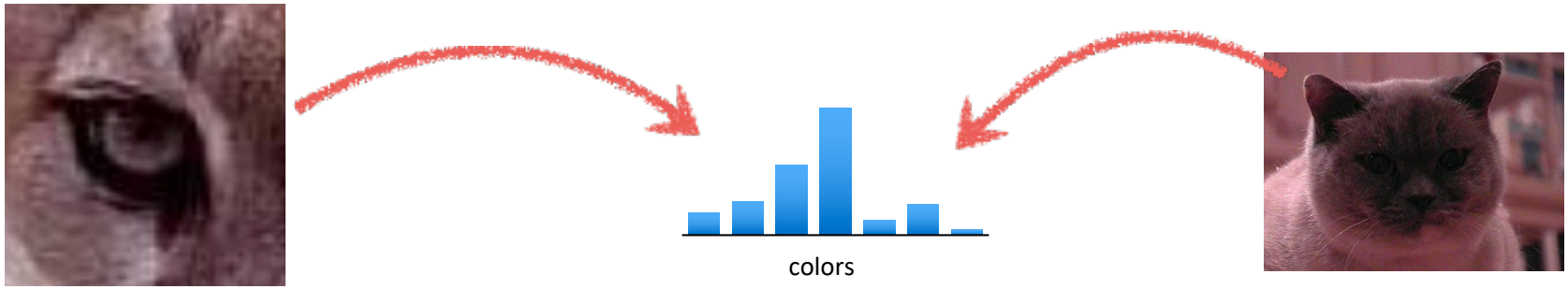


Invariant to changes in scale and rotation

What are the problems?

Color histogram

Count the colors in the image using a histogram



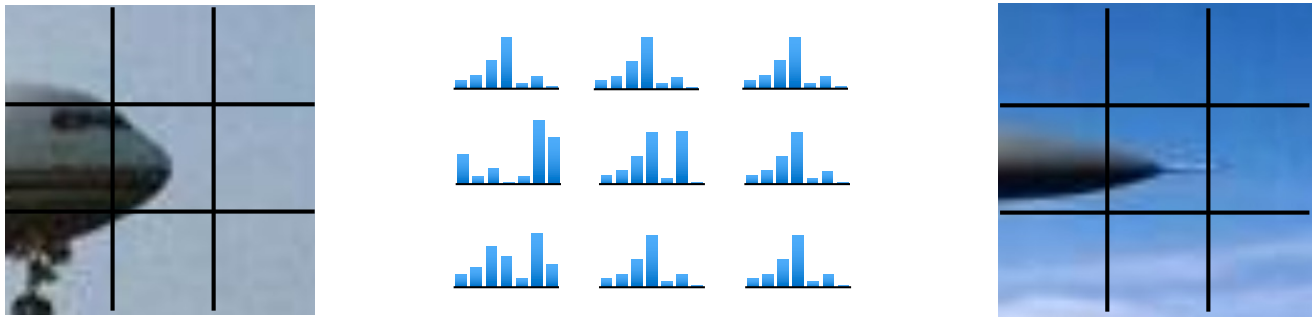
Invariant to changes in scale and rotation

What are the problems?

How can you be more sensitive to spatial layout?

Spatial histograms

Compute histograms over spatial ‘cells’



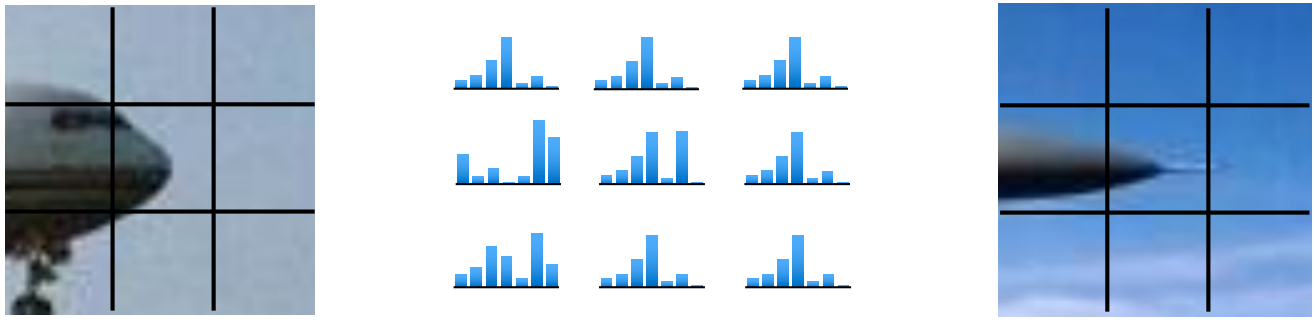
Retains rough spatial layout

Some invariance to deformations

What are the problems?

Spatial histograms

Compute histograms over spatial ‘cells’



Retains rough spatial layout

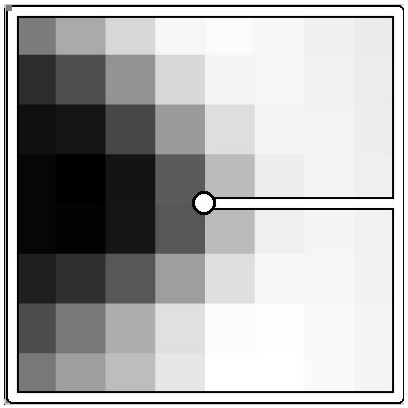
Some invariance to deformations

What are the problems?

How can you be completely invariant to rotation?

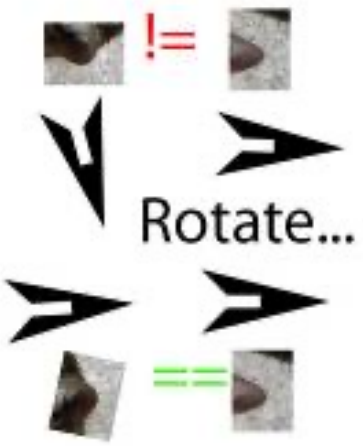
Orientation normalization

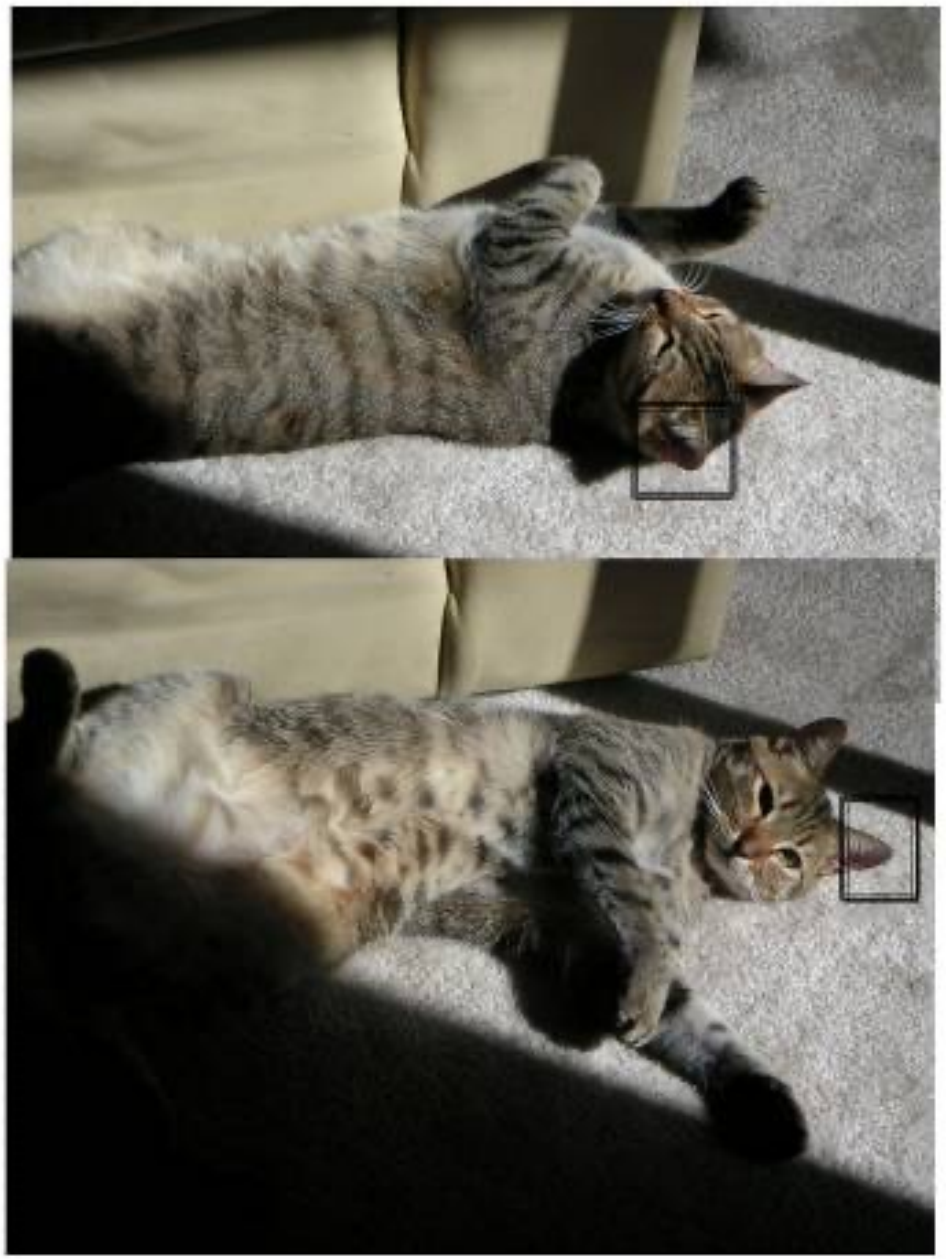
Use the dominant image gradient direction to normalize the orientation of the patch



save the orientation angle θ along with (x, y, s)

Rotation Invariance





SIFT

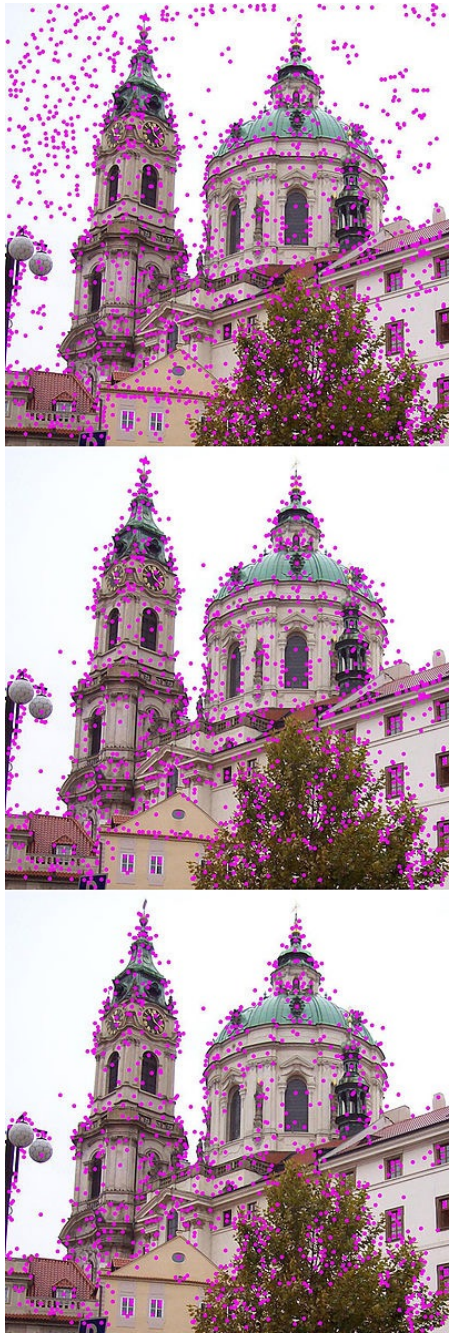
(Scale Invariant Feature Transform)

Distinctive image features from scale-invariant keypoints

DG Lowe - International journal of computer vision, 2004 - Springer

This paper presents a method for extracting distinctive invariant features from images that can be used to perform reliable matching between different views of an object or scene. The features are invariant to image scale and rotation, and are shown to provide robust matching across a substantial range of affine distortion, change in 3D viewpoint, addition of noise, and change in illumination. The features are highly distinctive, in the sense that a single feature can be correctly matched with high probability against a large database of features from …

☆ Save Cite Cited by 67710 Related articles All 164 versions



SIFT

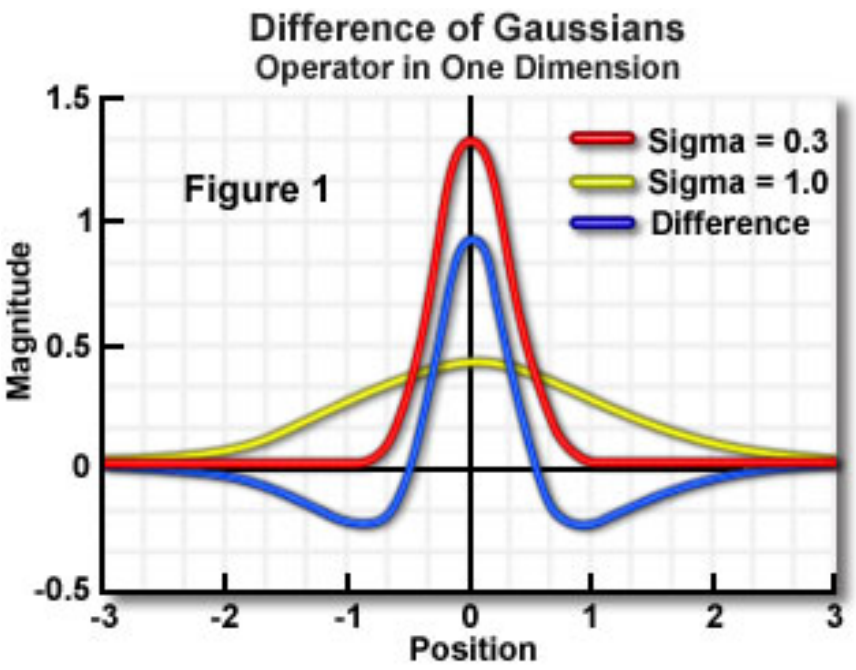
(Scale Invariant Feature Transform)

SIFT describes both a **detector** and **descriptor**

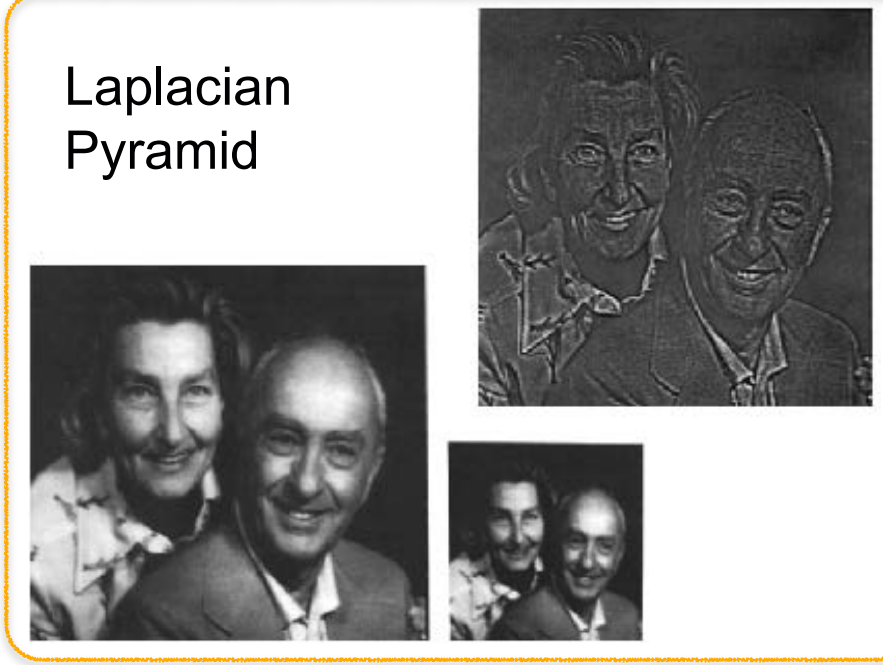
1. Multi-scale extrema detection
2. Keypoint localization
3. Orientation assignment
4. Keypoint descriptor

$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Difference of Gaussians



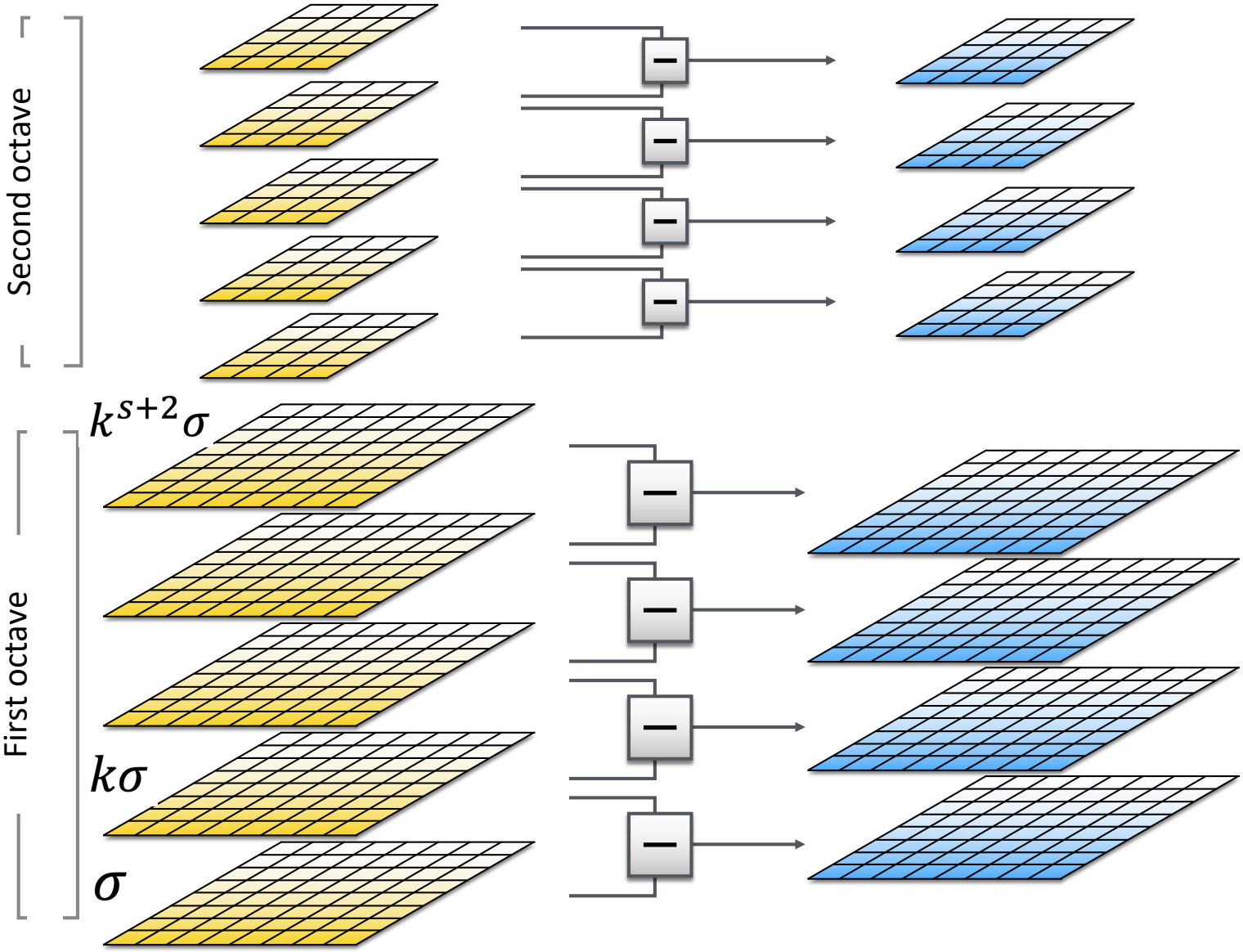
Laplacian Pyramid

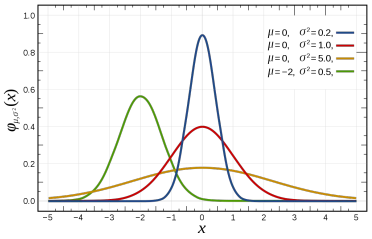


$$DoG = G(x,y,k\sigma) - G(x,y,\sigma)$$

$I(k\sigma)$ $I(\sigma)$ $I(k\sigma) - I(\sigma)$

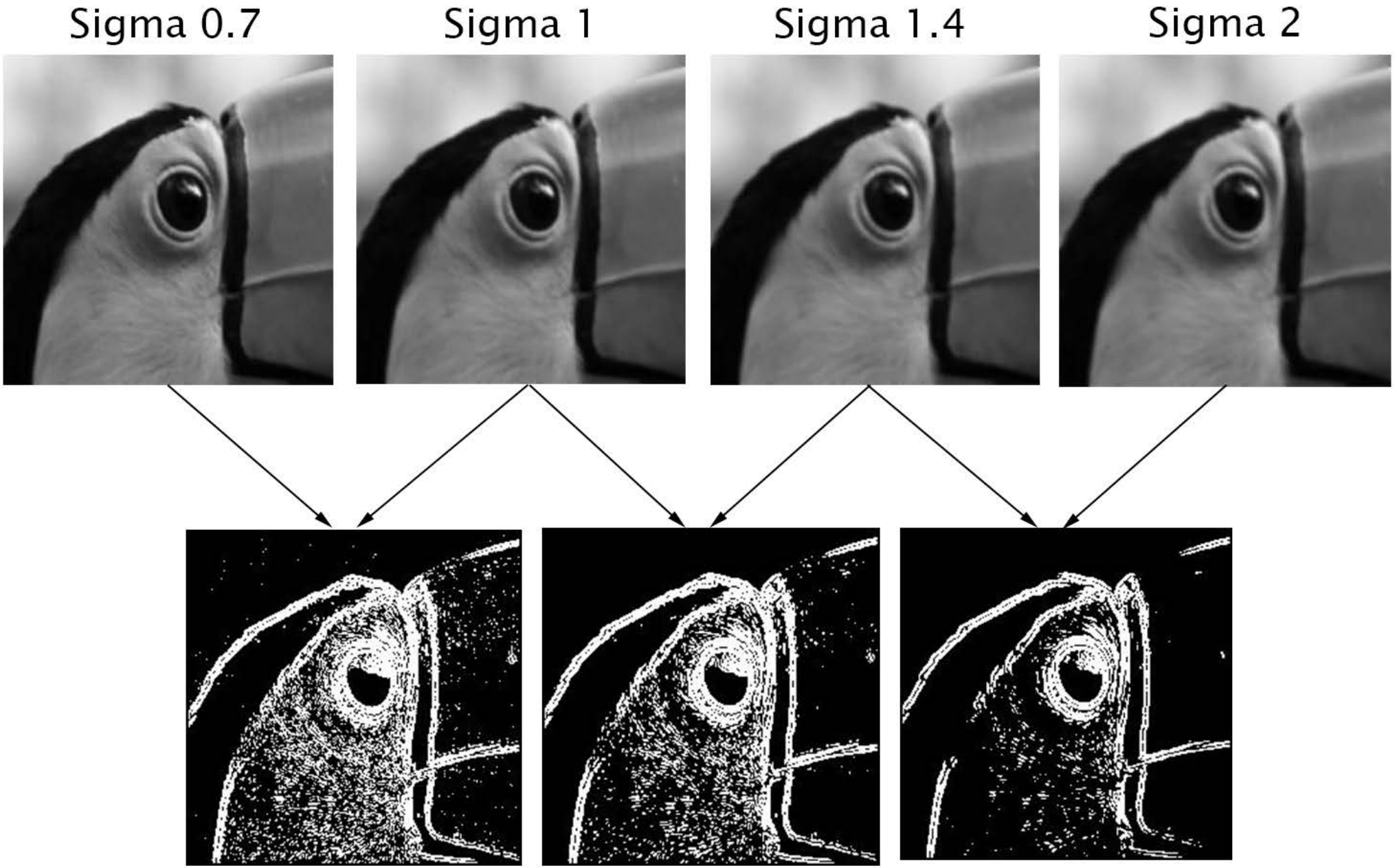
1. Multi-scale extrema detection



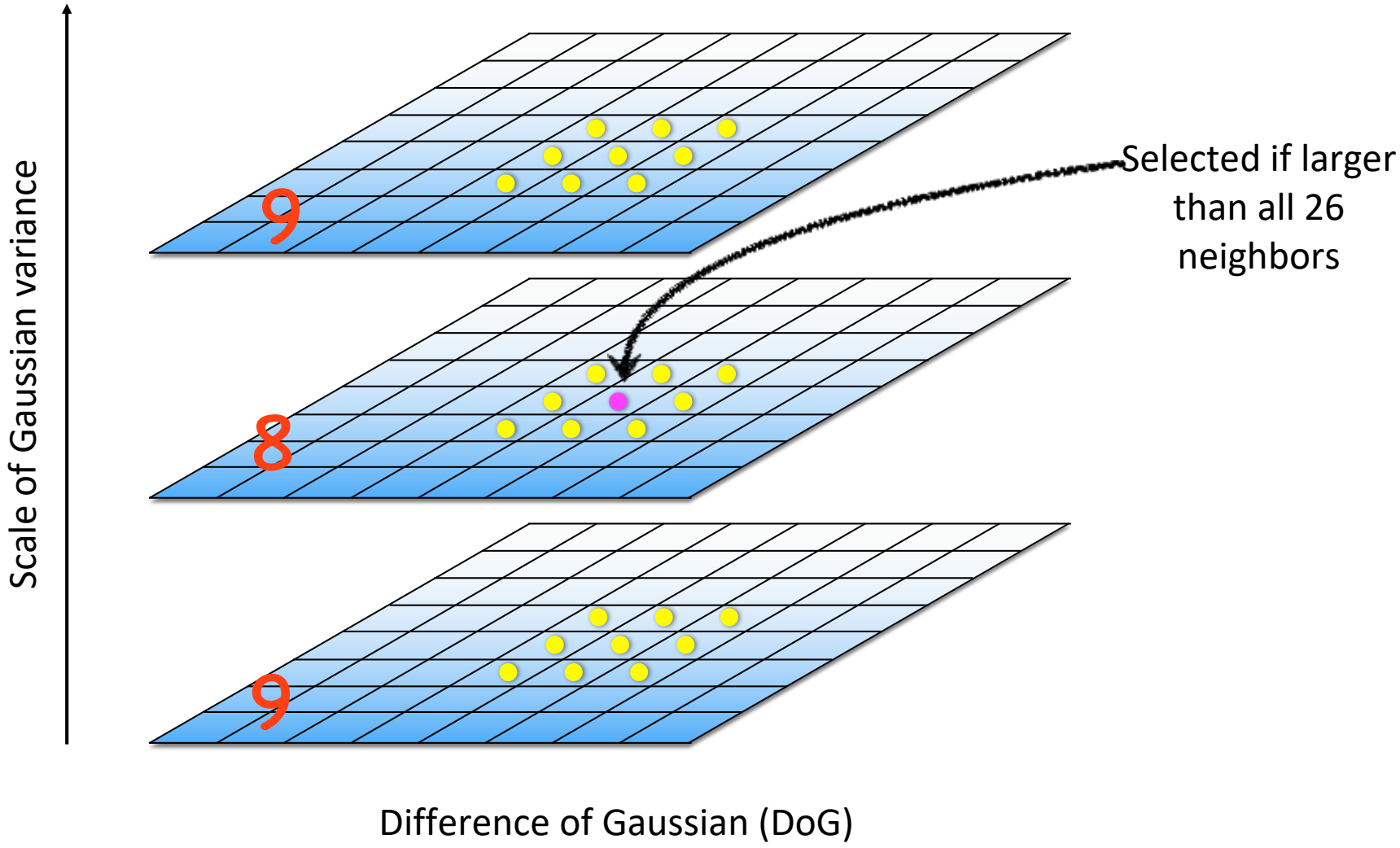


$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

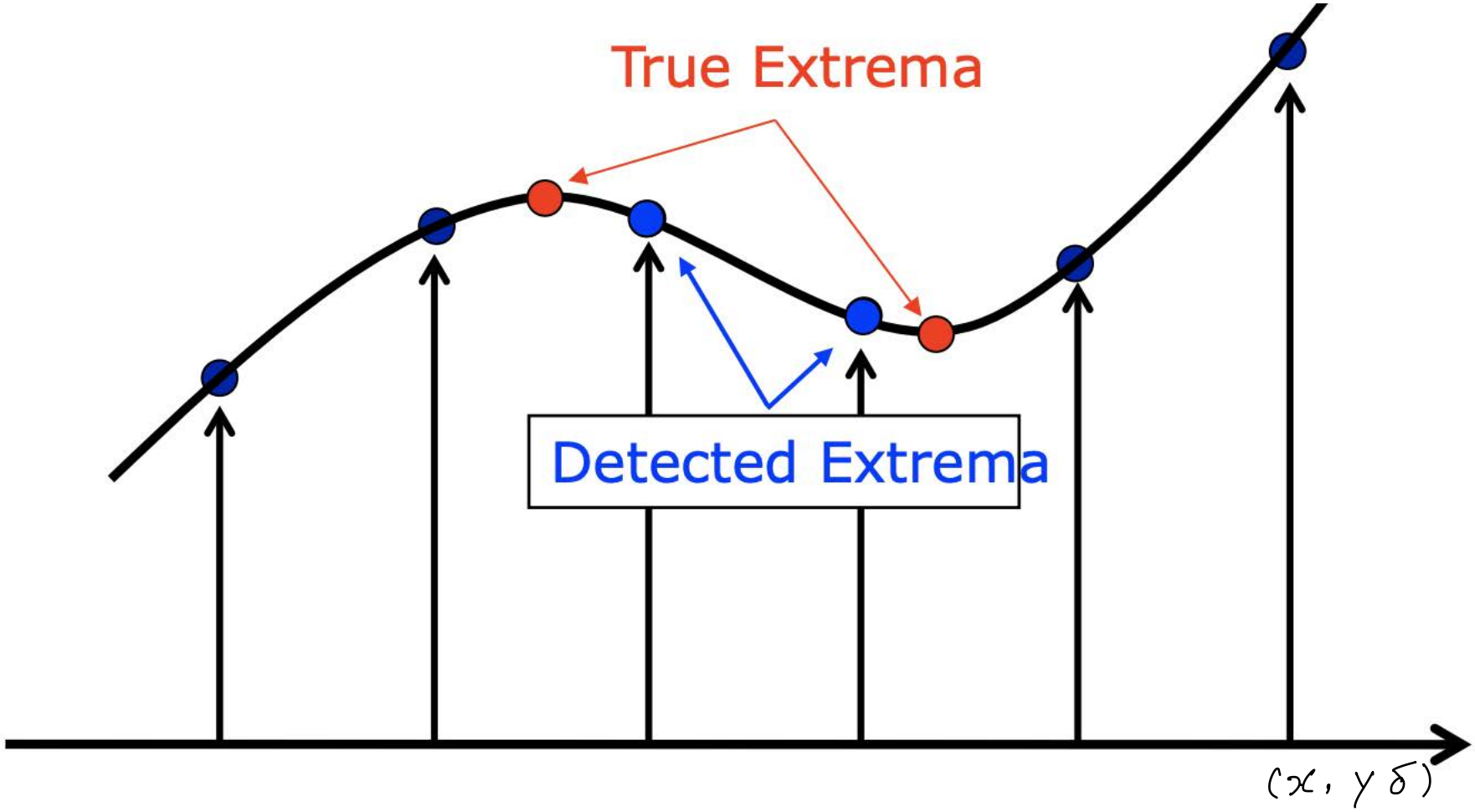
Difference of Gaussians



Scale-space extrema



2. Keypoint localization



Once a keypoint candidate has been found by comparing a pixel to its neighbors, the next step is to perform a detailed fit to the nearby data for location, scale, and ratio of principal curvatures.

2. Keypoint localization

2nd order Taylor series approximation of DoG scale-space

$$f(\mathbf{x}) = f + \frac{\partial f}{\partial \mathbf{x}}^T \mathbf{x} + \frac{1}{2}\mathbf{x}^T \frac{\partial^2 f}{\partial \mathbf{x}^2}\mathbf{x}$$

$$\mathbf{x} = \{x, y, \sigma\}$$

Take the derivative and solve for extrema

$$\mathbf{x}_m = -\frac{\partial^2 f}{\partial \mathbf{x}^2}^{-1} \frac{\partial f}{\partial \mathbf{x}}$$

Additional tests to retain only strong features

3. Orientation assignment

For a keypoint, **L** is the **Gaussian-smoothed** image with the closest scale,

$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2}$$

x-derivative y-derivative

$$\theta(x,y) = \tan^{-1}((L(x,y+1) - L(x,y-1))/(L(x+1,y) - L(x-1,y)))$$

Detection process returns

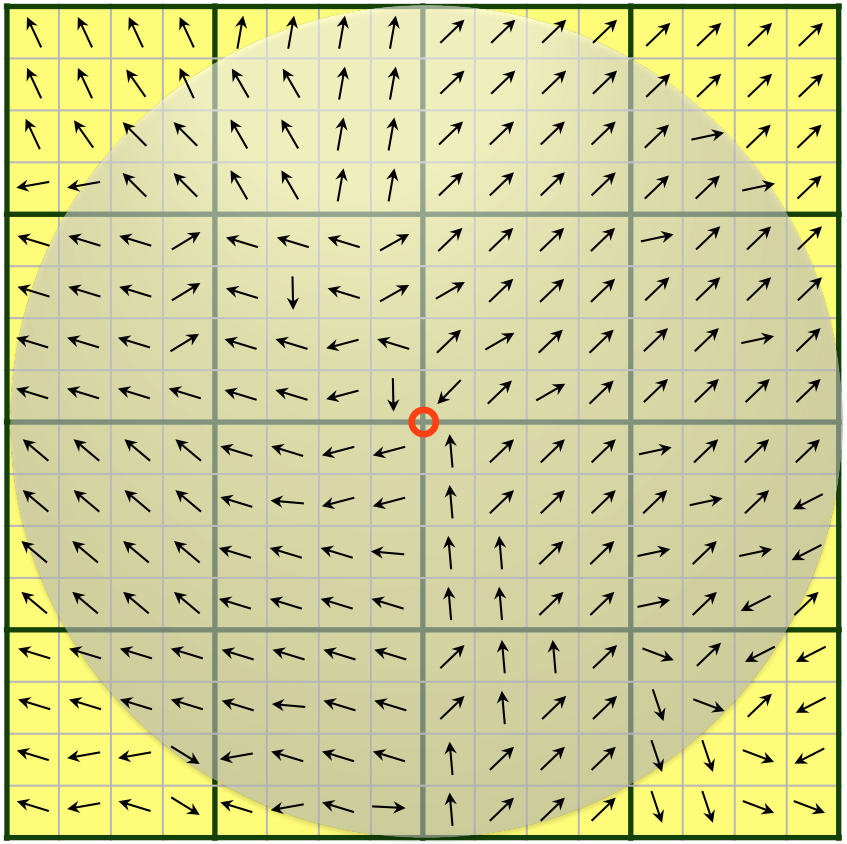
$$\{x, y, \sigma, \theta\}$$

location scale orientation

4. Keypoint descriptor

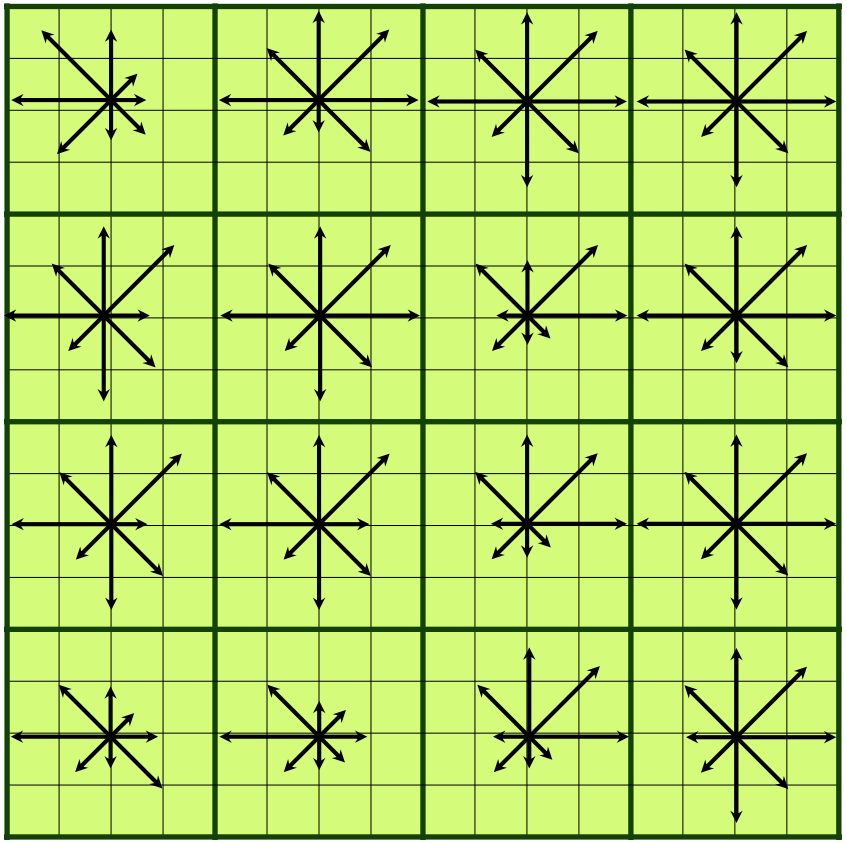
Image Gradients

(4 x 4 pixel per cell, 4 x 4 cells)



SIFT descriptor

(16 cells x 8 directions = 128 dims)



SIFT properties

- Invariant to
  - Scale
  - Rotation

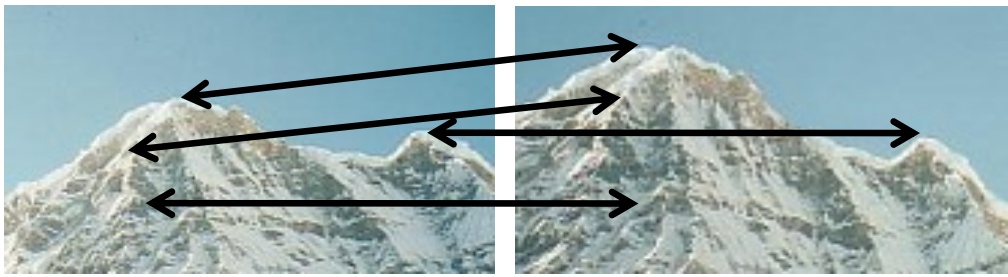
- Partially invariant to
  - Illumination changes
  - Camera viewpoint
  - Occlusion, clutter

Local features: main components

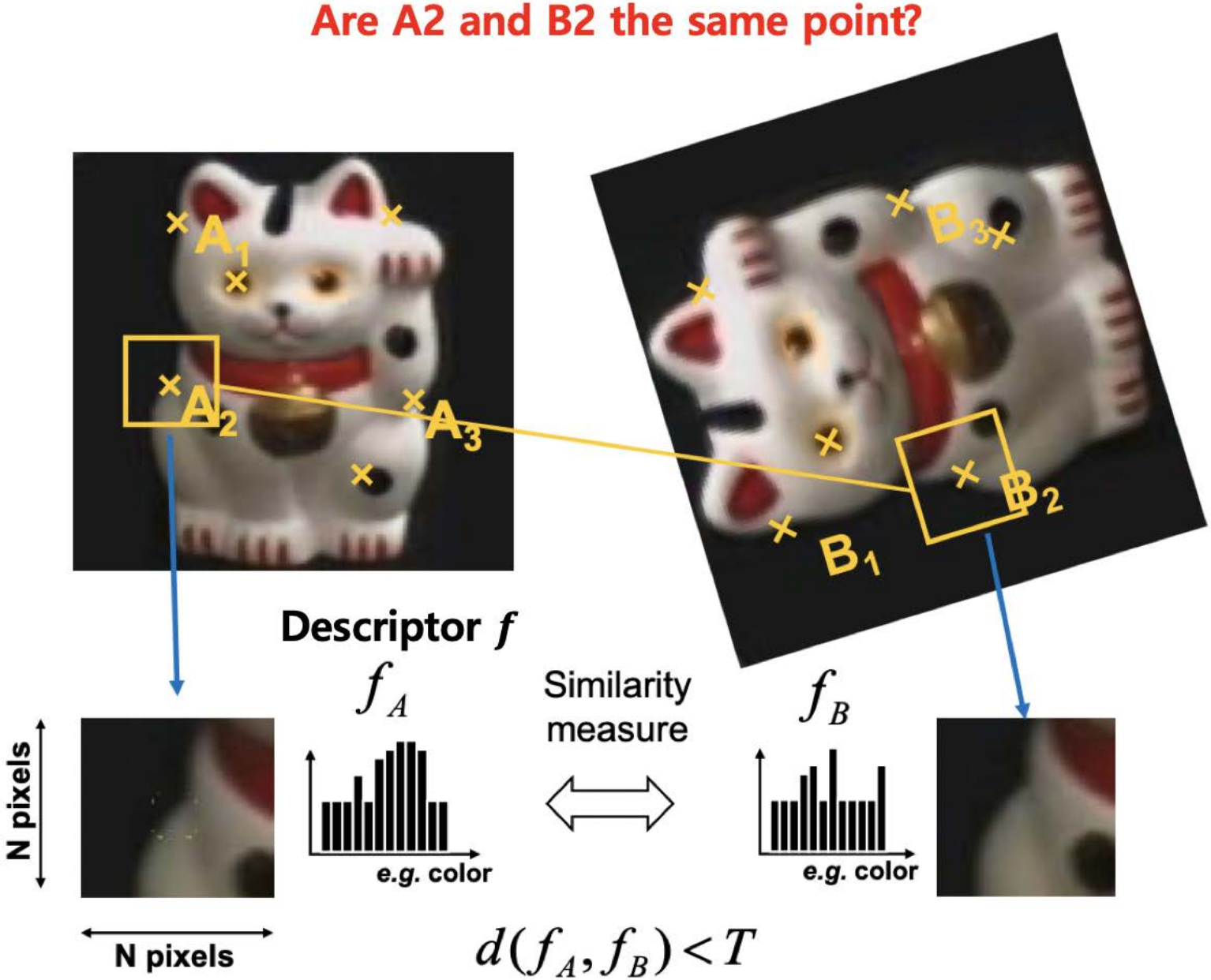
1) Detection: Identify the interest points

2) Description:Extract vector feature descriptor surrounding each interest point.

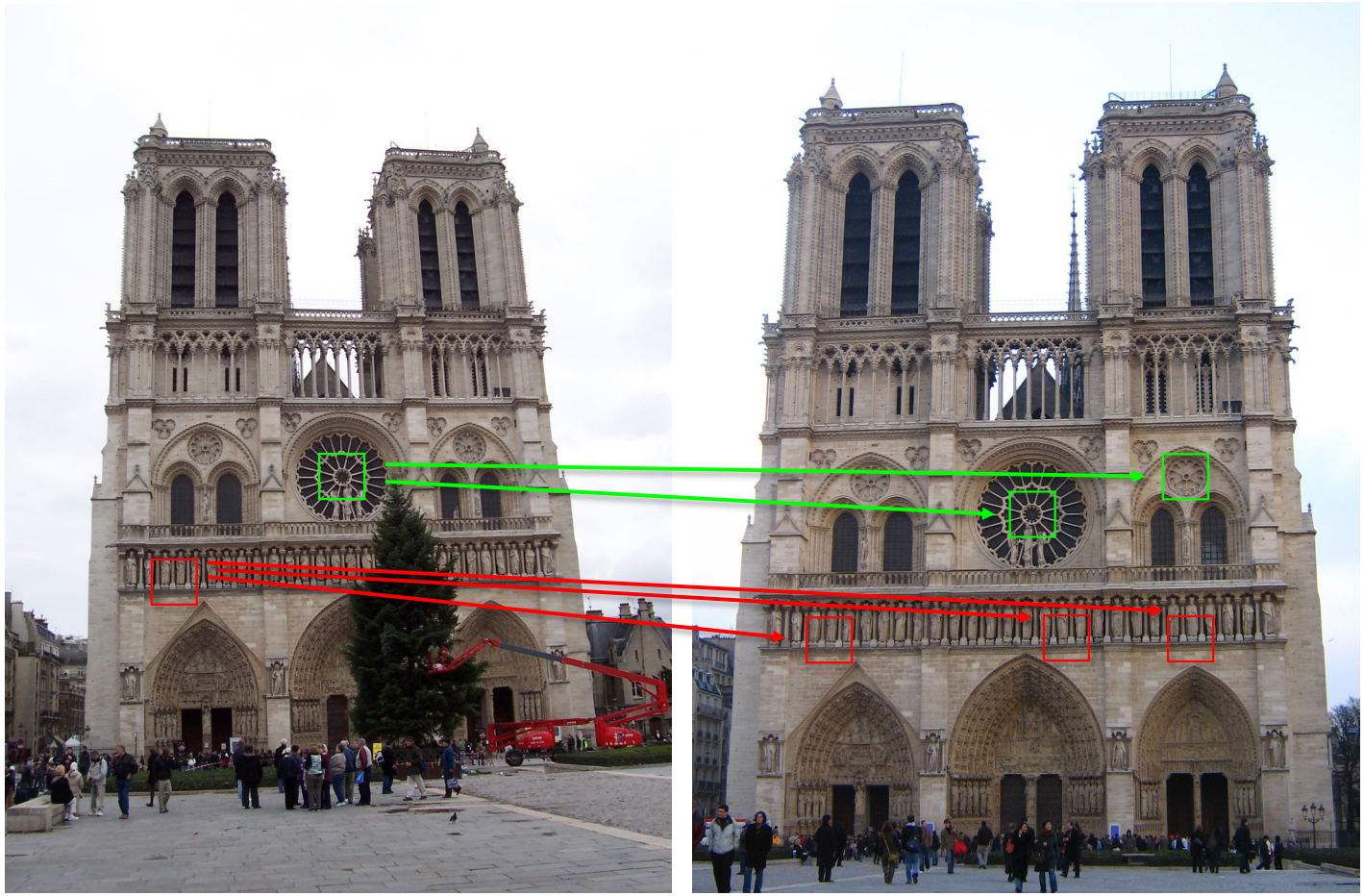
3) Matching: Determine correspondence between descriptors in two views



Matching local features



How do we decide which features match?



Distance: 0.34, 0.30, 0.40 Distance: 0.61, 1.22

Euclidean distance vs. Cosine Similarity

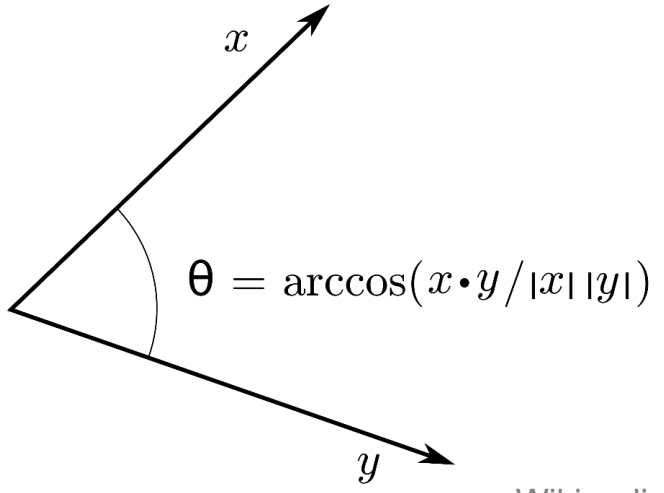
- Euclidean $\mathrm{d}(\mathbf{p}, \mathbf{q}) = \mathrm{d}(\mathbf{q}, \mathbf{p}) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_n - p_n)^2}$

$$= \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}.$$

$$\|\mathbf{q} - \mathbf{p}\| = \sqrt{(\mathbf{q} - \mathbf{p}) \cdot (\mathbf{q} - \mathbf{p})}.$$

- Cosine similarity: $\mathbf{a} \cdot \mathbf{b} = \|\mathbf{a}\|_2 \|\mathbf{b}\|_2 \cos\theta$

$$\mathbf{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|_2 \|\mathbf{B}\|_2}$$



Wikipedia

Feature Matching

- Criteria 1:
  - Compute distance in feature space, e.g., Euclidean distance between 128-dim SIFT descriptors
  - Match point to lowest distance (nearest neighbor)

- Problems:
  - Does everything have a match?

Feature Matching

- Criteria 2:
  - Compute distance in feature space, e.g., Euclidean distance between 128-dim SIFT descriptors
  - Match point to lowest distance (nearest neighbor)
  - Ignore anything higher than threshold (no match!)

- Problems:
  - Threshold is hard to pick
  - Non-distinctive features could have lots of close matches, only one of which is correct

Evaluation

- Suppose our matcher computes 1,000 matches between two images
  - 800 are correct matches, 200 are incorrect (according to an oracle that gives us ground truth matches)
  - A given threshold (e.g., ratio distance = 0.6) gives us 600 correct matches and 100 incorrect matches that survive the threshold
  - True positive rate = 600 / 800 = ¾
  - False positive rate = 100 / 200 = ½

Evaluation

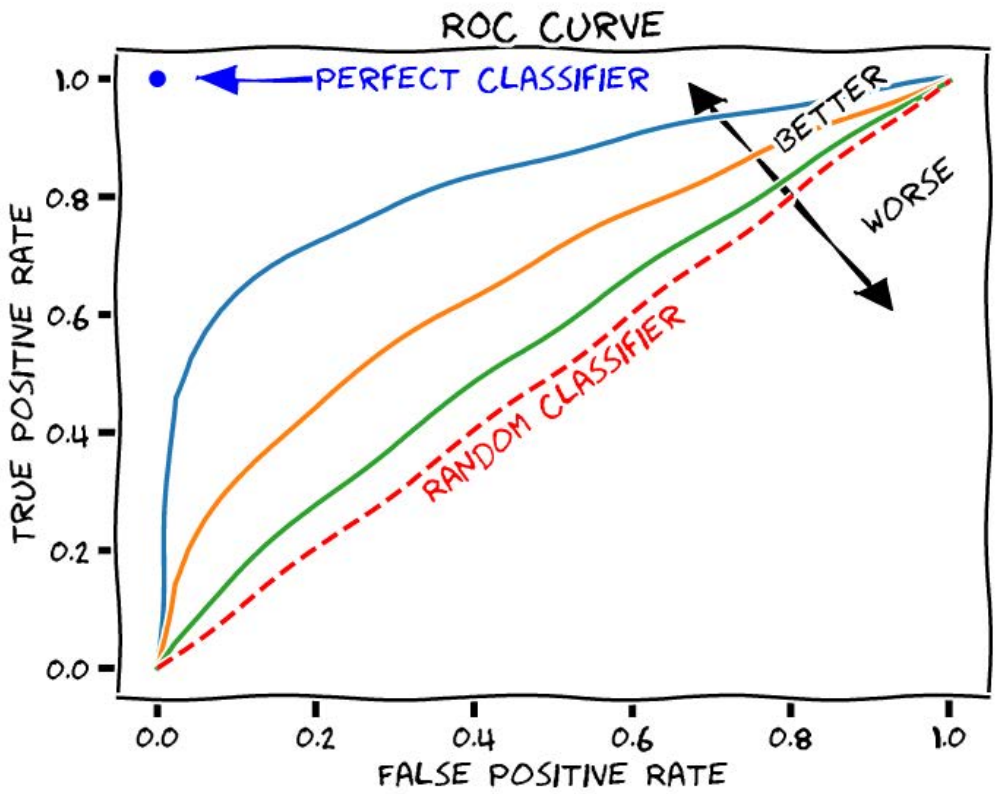
		Actual: Positive	Actual: Negative
Prediction	Positive	TP	FP
Prediction	Negative	FN	TN

Precision: How many retrieved items are relevant?

Recall: How many relevantitems are retrieved?

$$\text{Precision} = \frac{tp}{tp + fp}$$

$$\text{Recall} = \frac{tp}{tp + fn}$$



https://en.wikipedia.org/wiki/Receiver_operating_characteristic