10 – Pattern & Feature Matching

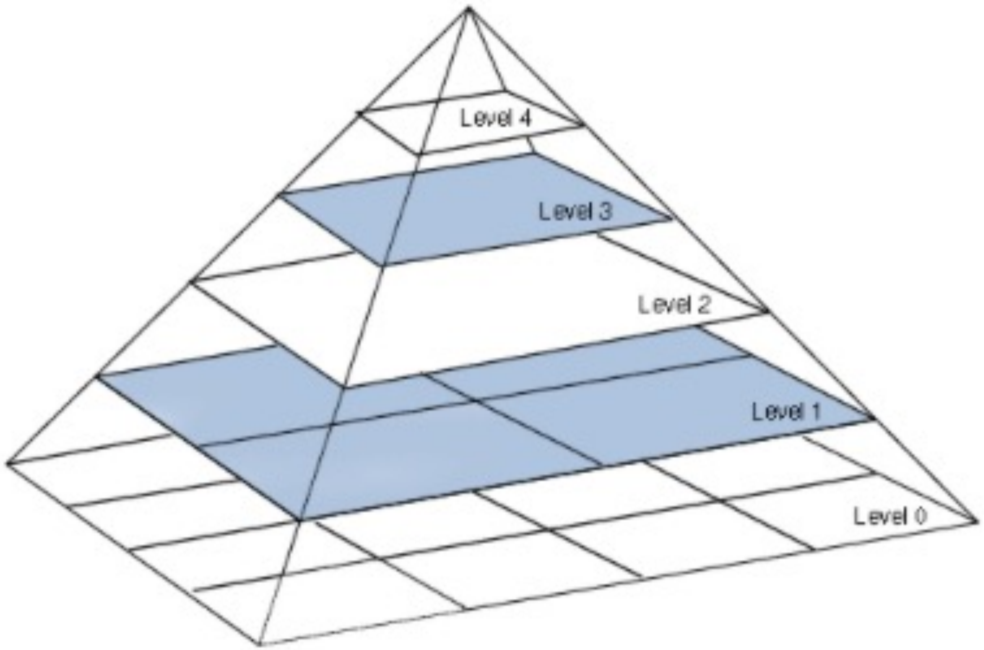
Prof Peter YK Cheung

Dyson School of Design Engineering

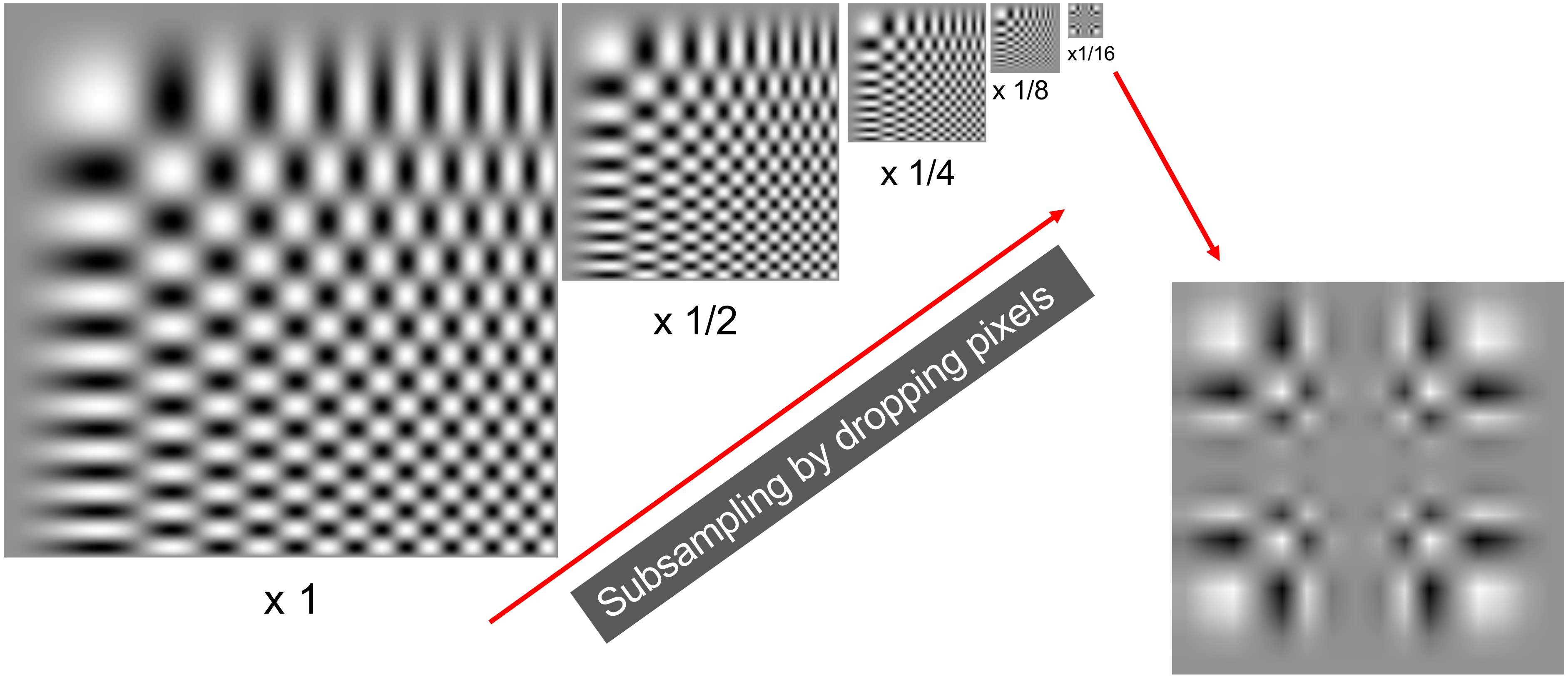
URL: www.ee.ic.ac.uk/pcheung/teaching/DE4_DVS/
E-mail: p.cheung@imperial.ac.uk

Multi-resolution Pyramid

- Important to process image at the **appropriate resolution**.
- Objective: good accuracy with minimal computation.
- Achieved by rescaling the image through **sub-sampling** or **interpolation**.

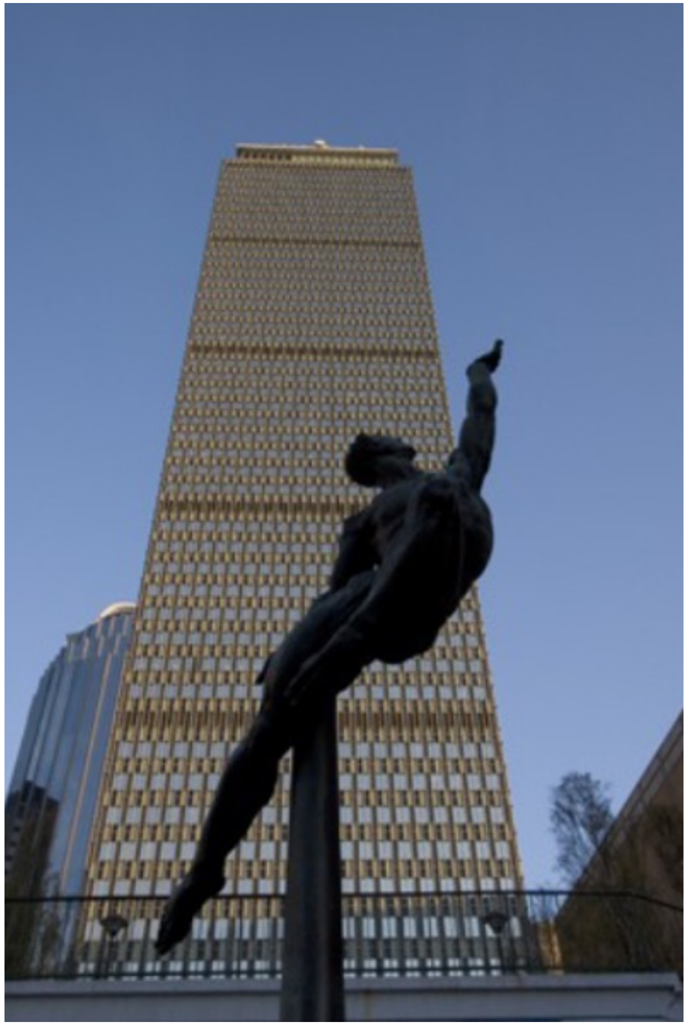


Beware of Aliasing



- Image size halved by taking every other pixel in both directions.
- High frequency patten now appears as low frequency.
- This is the result of ALIASING (DE2 Electronics 2).

Aliasing creates false pattens



Subsampling by dropping pixels

Source: F. Durand

Right way to Down-sample

convolution

*

5 x 5 Gaussian kernel

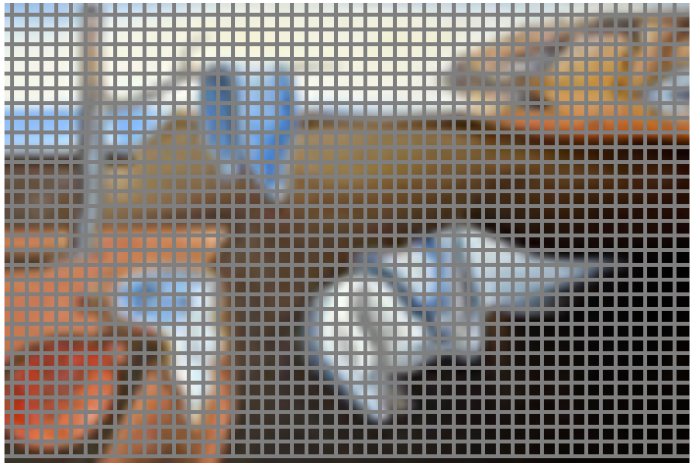
1	4	6	4	1
4	16	24	16	4
6	24	36	24	6
4	16	24	16	4
1	4	6	4	1

Lowpass filtered

subsample

Right way to Up-sample

Expand image



interpolation

- Nearest neighbour
- Bilinear
- Bicubic

Comparison of Right and Wrong way of resizing image

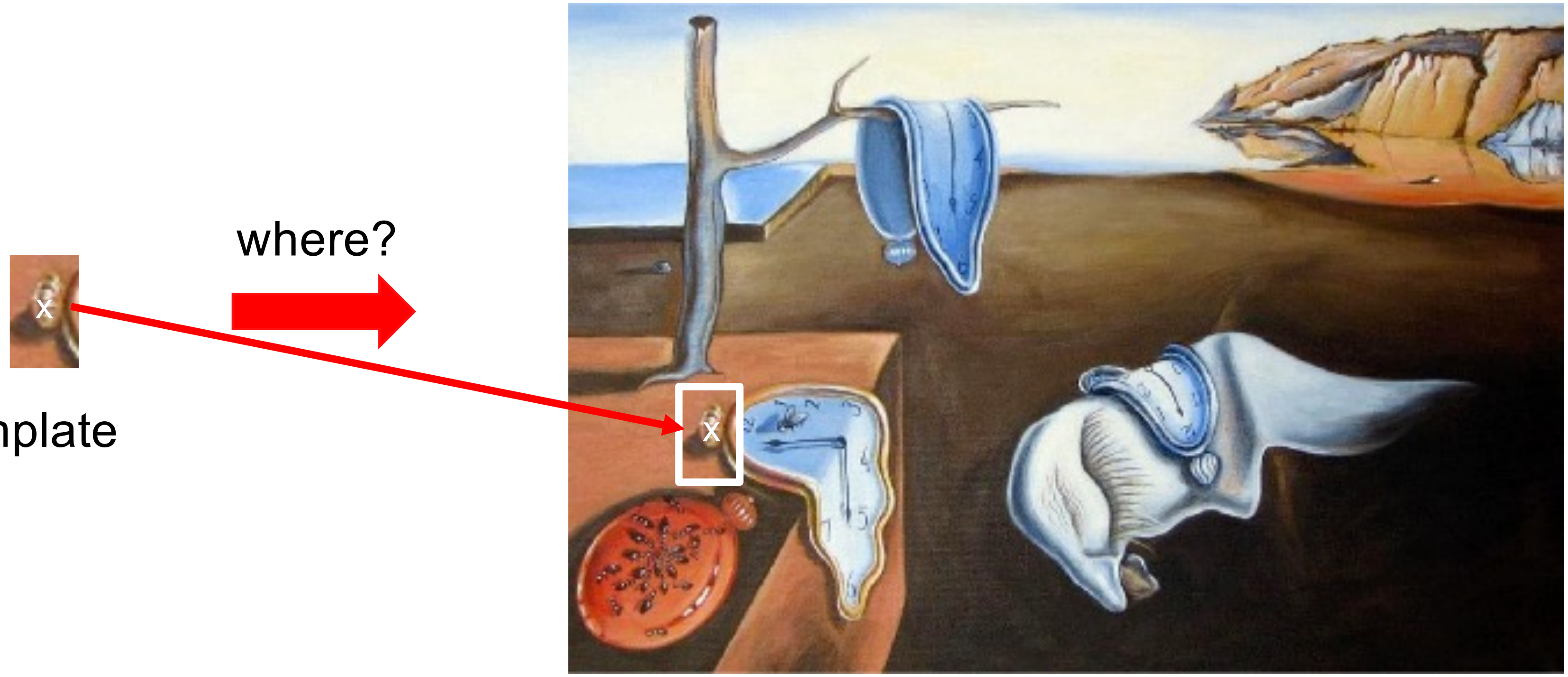
Dropping pixels

Gaussian filter then dropping pxiels

Source: S. Seitz

Template matching Problem

- Given a template (e.g. image of an object), find the location in a large image.
- Usually template is small, image is large.
- Assumption: template is an exact copy of a part of the large image.
- Particularly useful for image alignment (registration).



Template matching by Normalized Cross Correlation

- Given an image $f(x, y)$, and a template $t(u, v)$, cross correlation is the same as performing image filter with $t(u, v)$ as the kernel (see Lecture 5, slides 2-7.
- However, here we use normalized cross correlation (NCC) γ, where γ is normalised to the range of +1 to -1.
- The definition of γ is:

$$\gamma(x, y) = \frac{\sum_{x,y}(f(x, y) - \bar{f}_{u,v})\,(t(x-u, y-v) - \bar{t})}{\sqrt{\sum_{x,y}(f(x, y) - \bar{f}_{u,v})^2 \sum_{x,y}(t(x-u, y-v) - \bar{t})^2}}$$

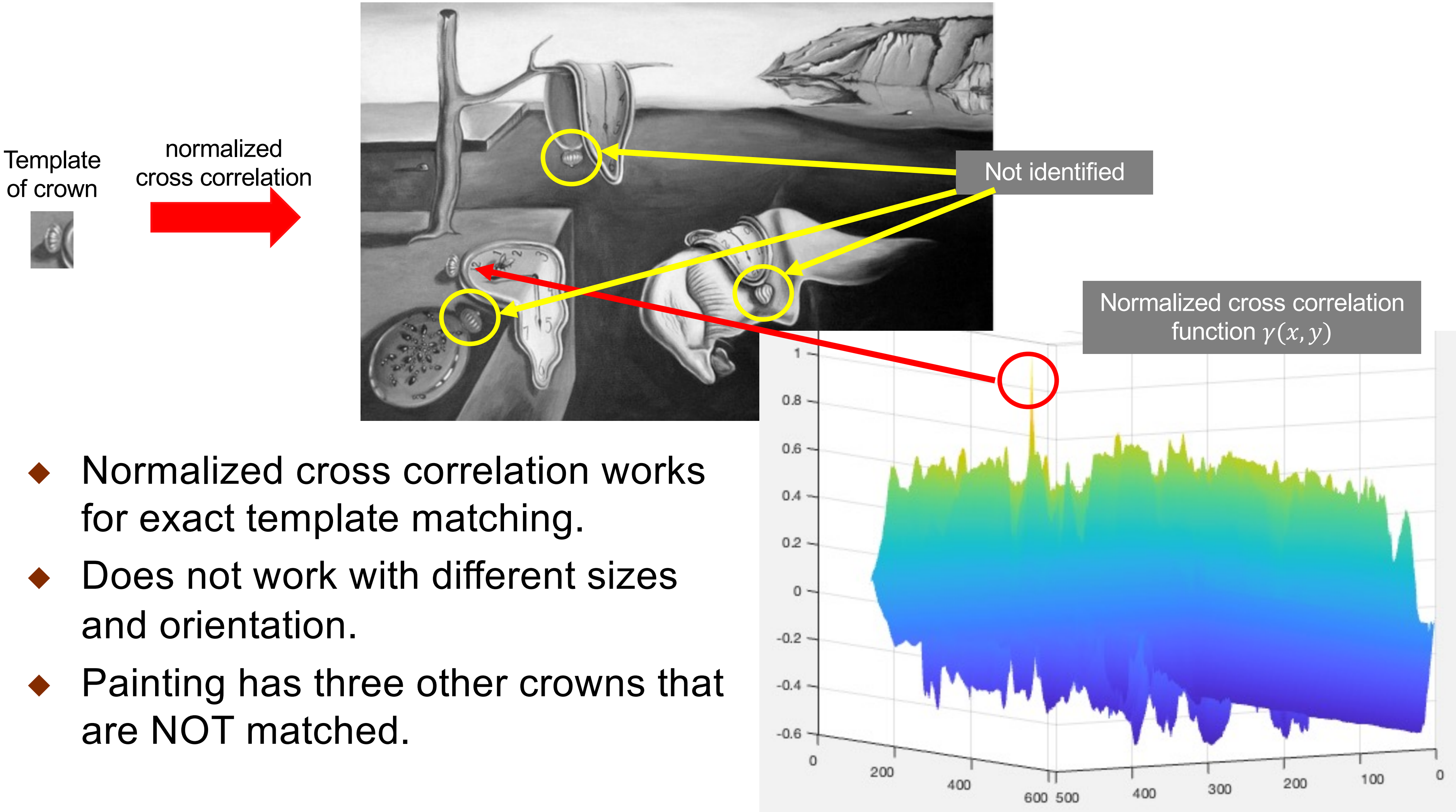
- where:

  $\bar{f}_{u,v}$ is the mean value of $f(x, y)$ within the area of the template $t(u, v)$, and

  $\bar{t}$ is the mean value of the template.

- Using this normalization, $\gamma(x, y)$ is independent of changes in brightness (mean) or contrast (standard deviation) of the image.

The Dali painting example



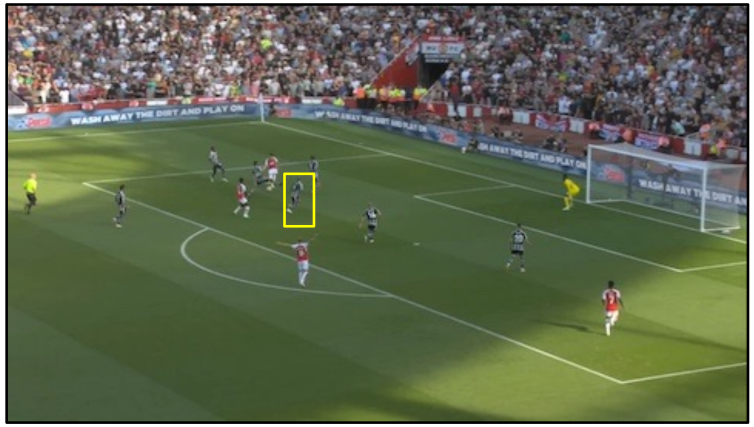
- Normalized cross correlation works for exact template matching.
- Does not work with different sizes and orientation.
- Painting has three other crowns that are NOT matched.

General feature matching problem

Where?

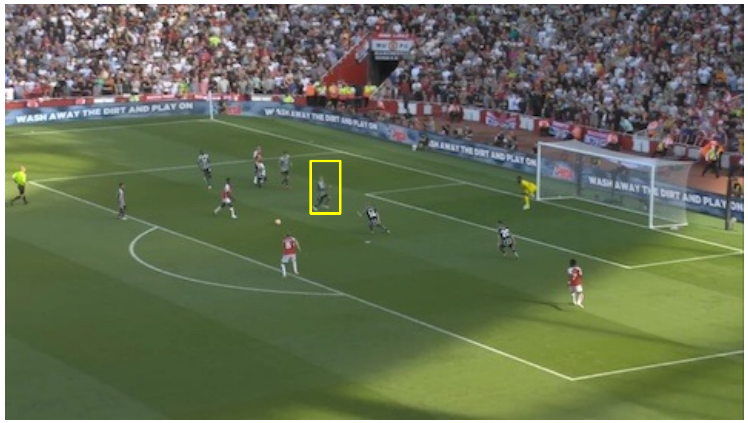
- Problems in matching objects in general:
  1. Different size (or scale)
  2. Different orientation
  3. Different brightness and contrast
  4. Partially covered (occlusion)

Harder Visual Processing



Stitching

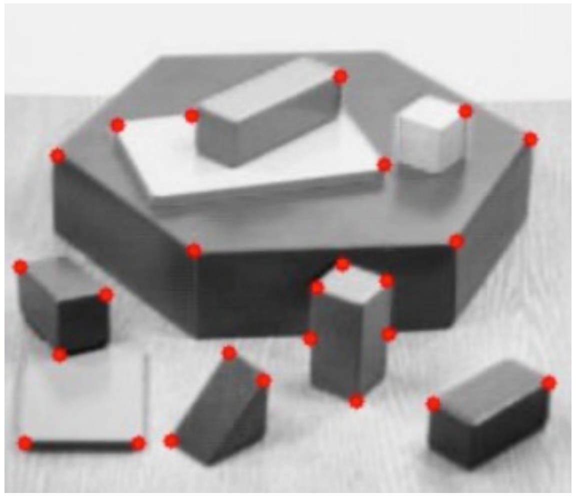
Tracking



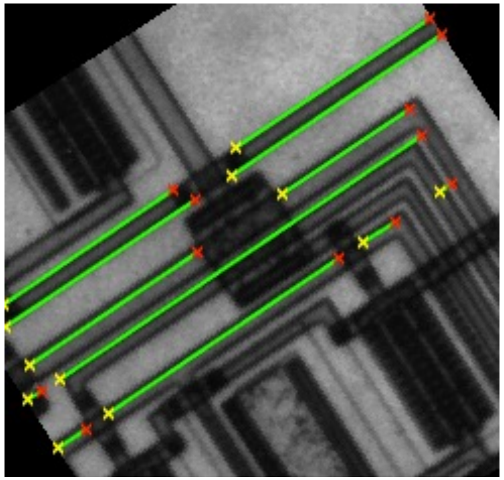
What are interesting features?

- To handle a scene, need to identify and locate interesting features.
- These could be:
  1. Points, particularly corners
  2. Lines or shapes (e.g. circles)
  3. Blobs or patches

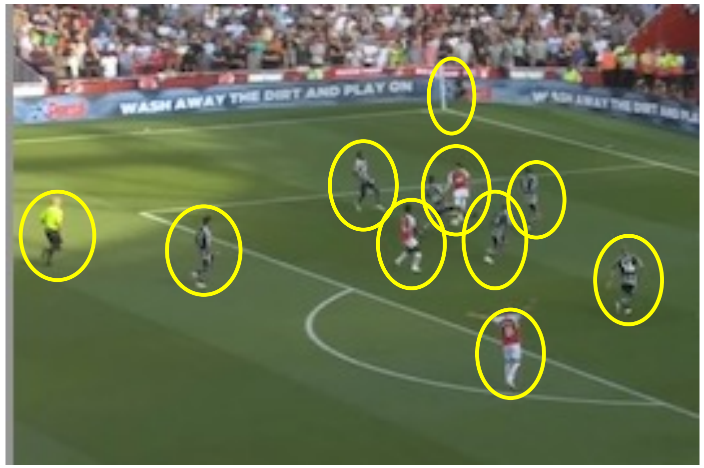
Corners



Line and edges



Blobs

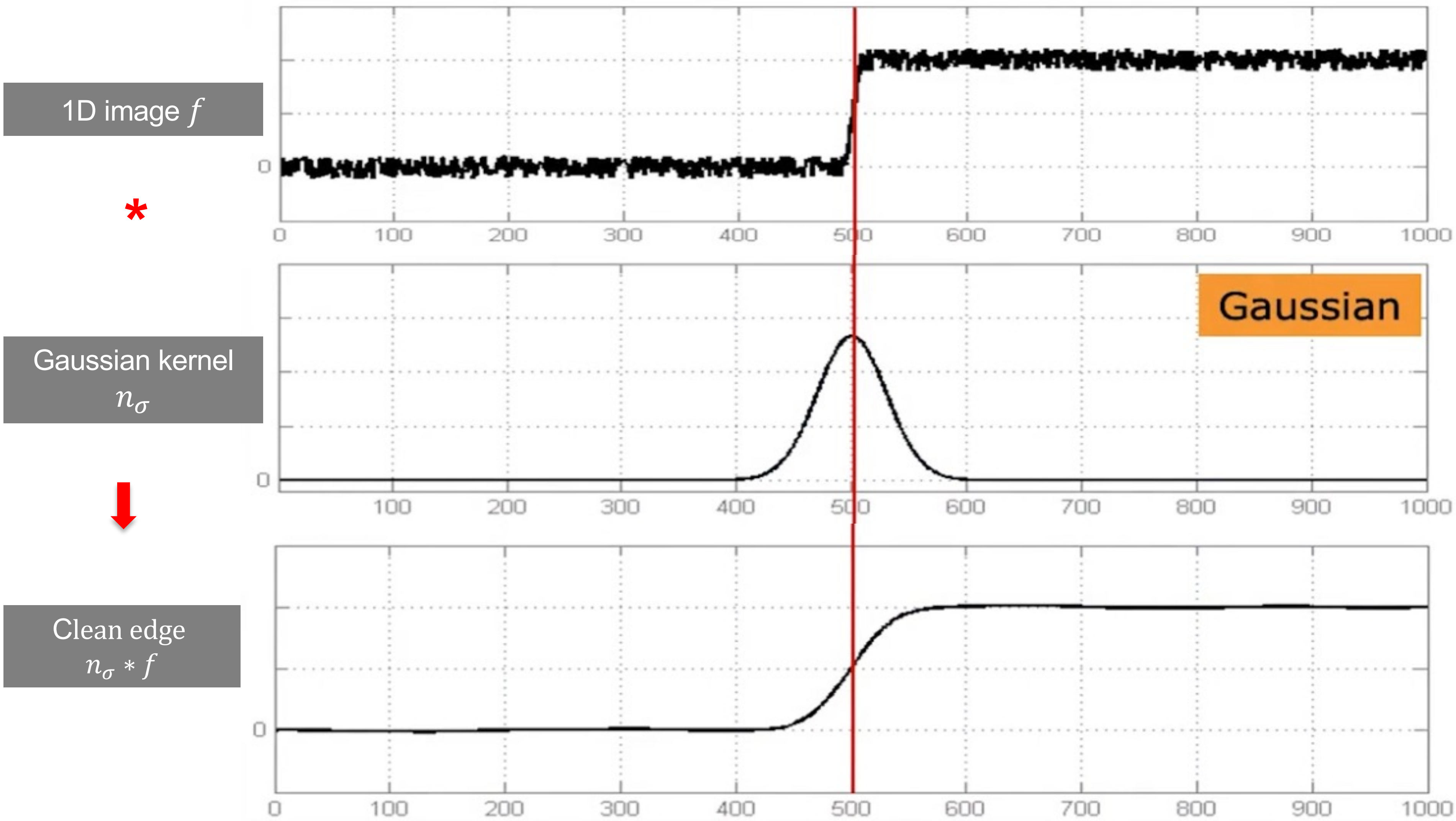


SIFT – A Blob Feature Detection Method

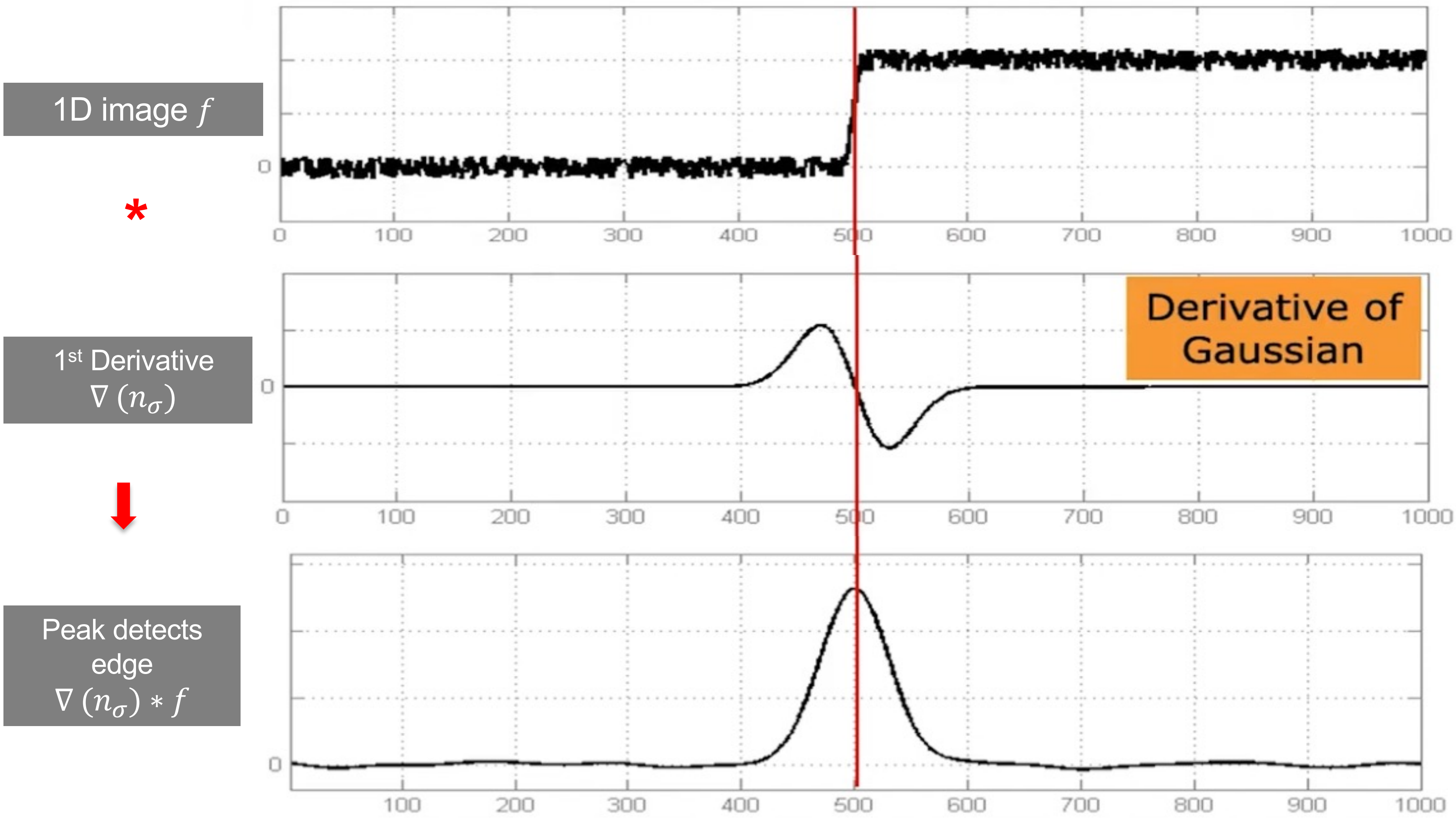
- **SIFT** stands for **Scale Invariant Feature Transform.**
- Useful for image alignment (registration), object tracking and 2D object recognition.
- Proposed by Lowe in 2004 to identify interesting blob features that are independent of their size, orientation and intensity (paper on webpage).
- Output from SIFT detector are these properties of features:
  1. **Locations** of the blobs
  2. **Scales** (or sizes) of the blobs
  3. **Orientations** of the blobs
  4. **Signatures** or descriptors for the blobs

Descriptor or signature

Recap on Gaussian Filter – noise removal



1st Derivative of Gaussian – Edge detection



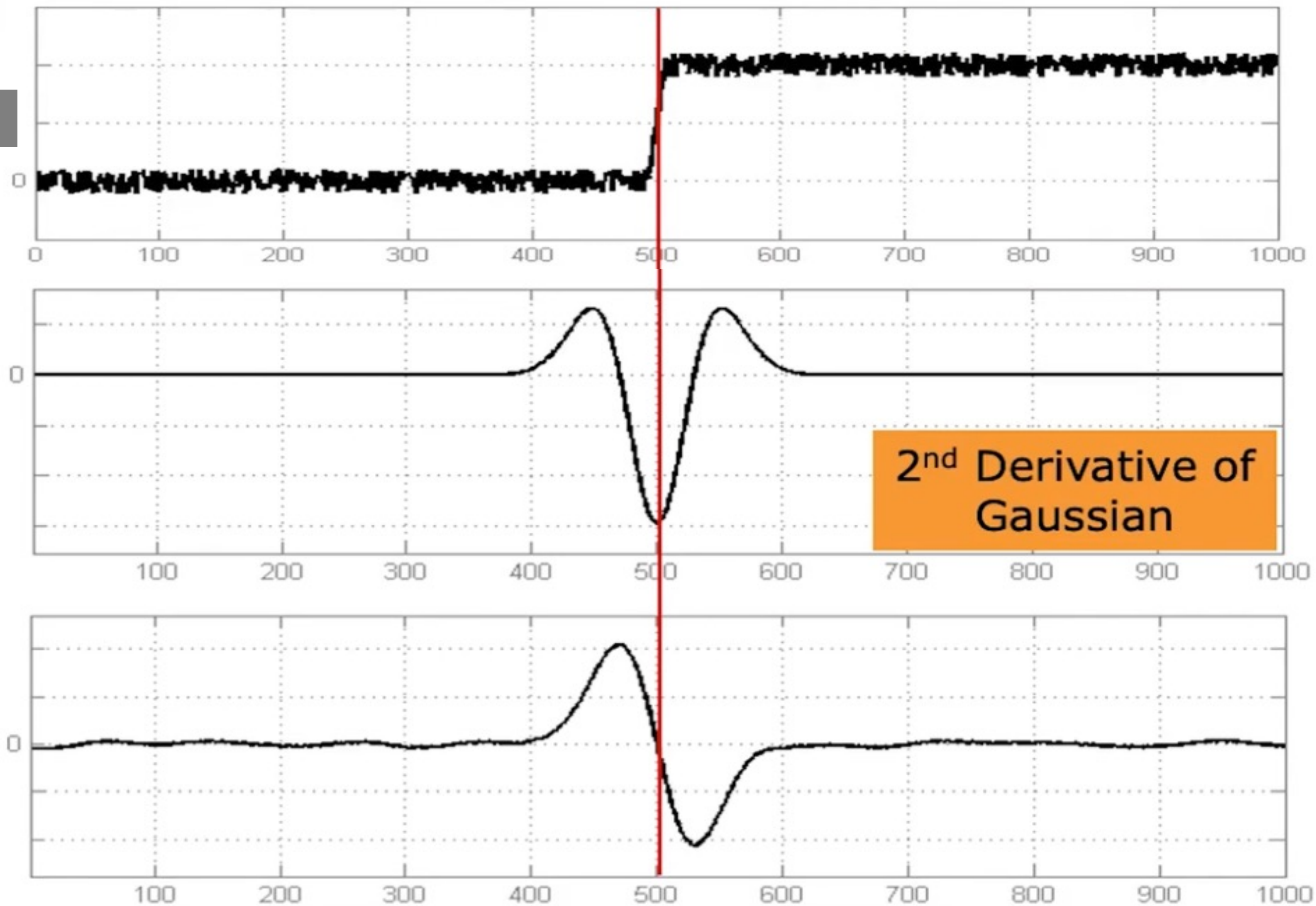
2^{nd} Derivative of Gaussian – Edge detection

1D image f

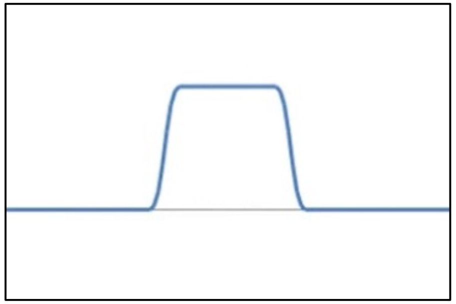
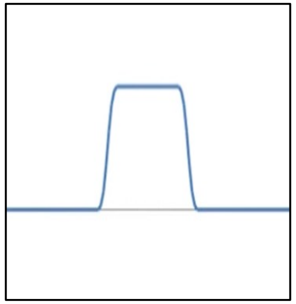
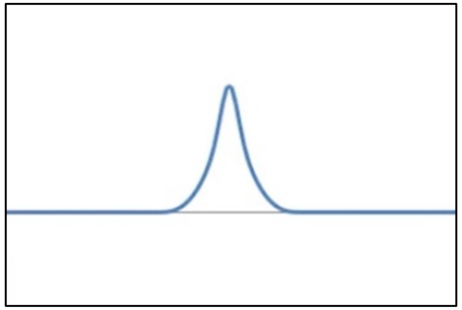
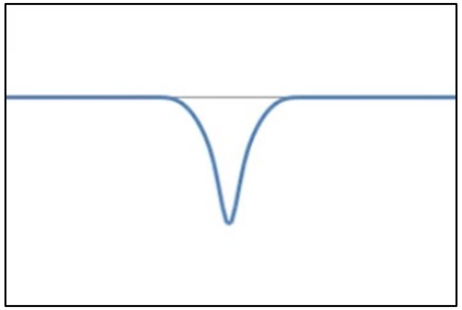
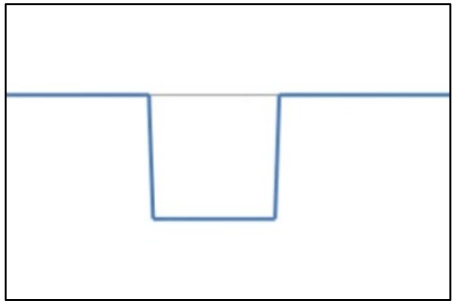
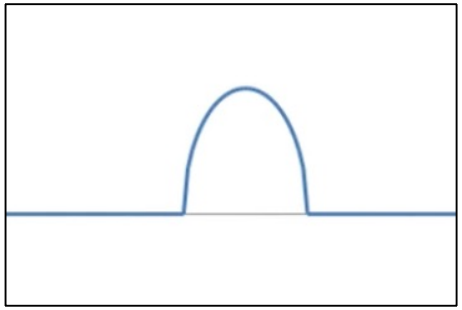
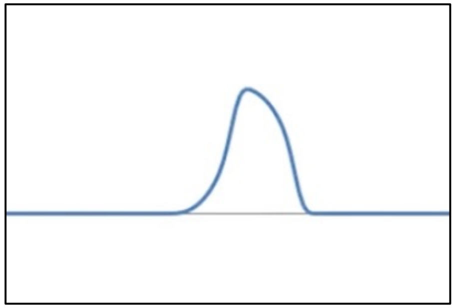
*

2^{nd} derivative
$\nabla^2 \, (n_\sigma)$

Zero crossing
detects edge
$\nabla^2 \, (n_\sigma) * f$

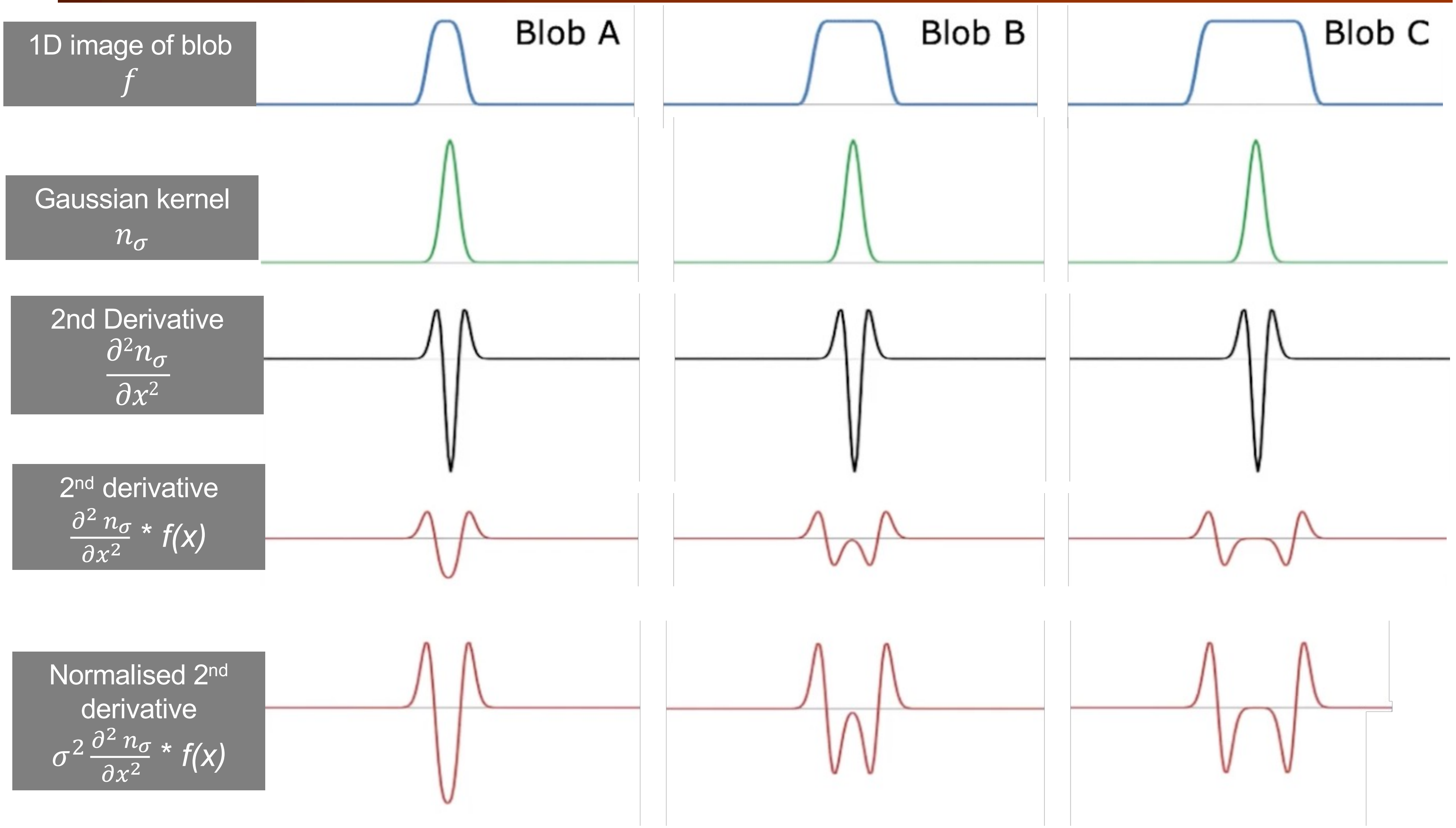


Different types of Blobs in 1D

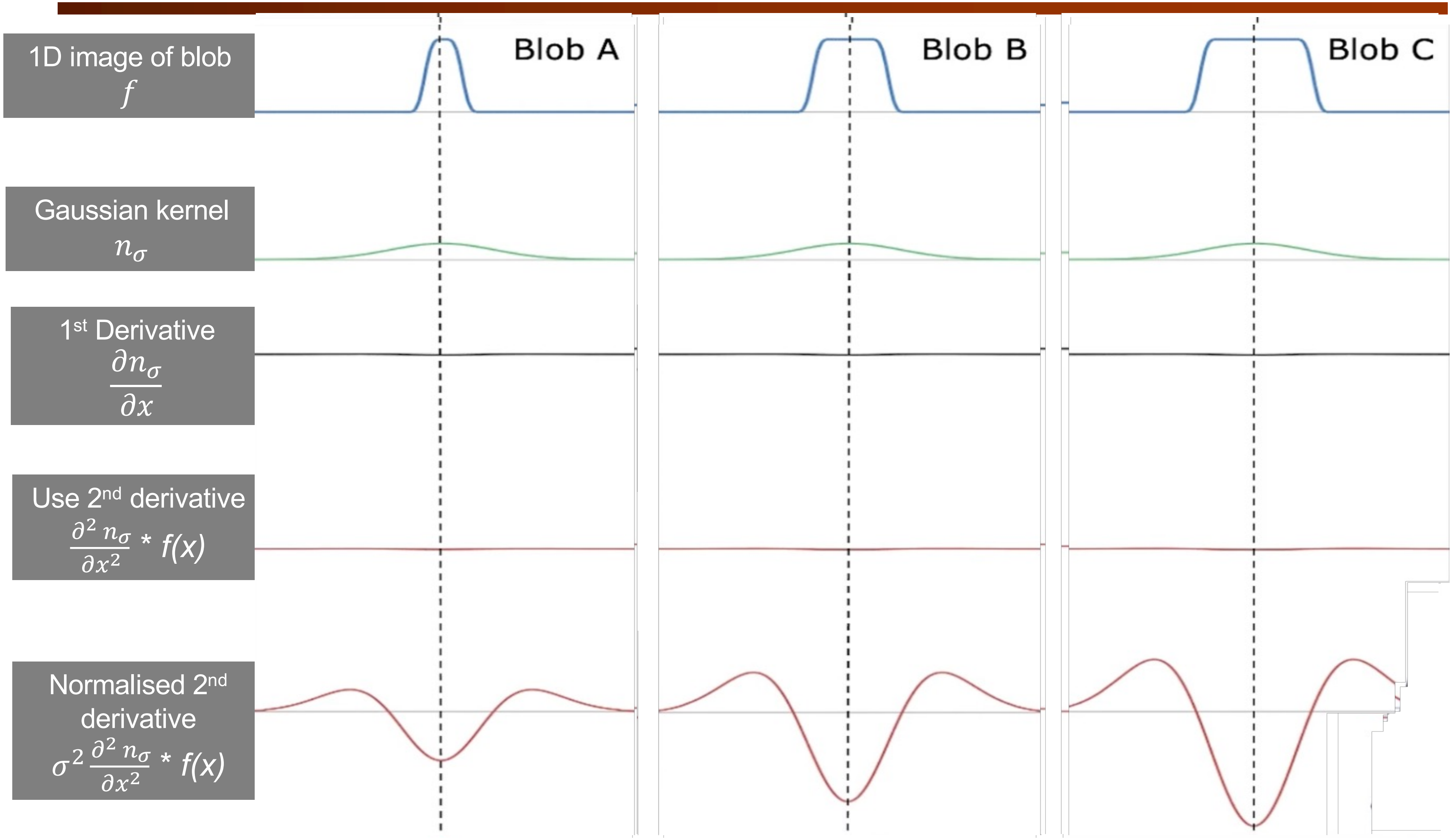








- Different shape and **SIZES**.

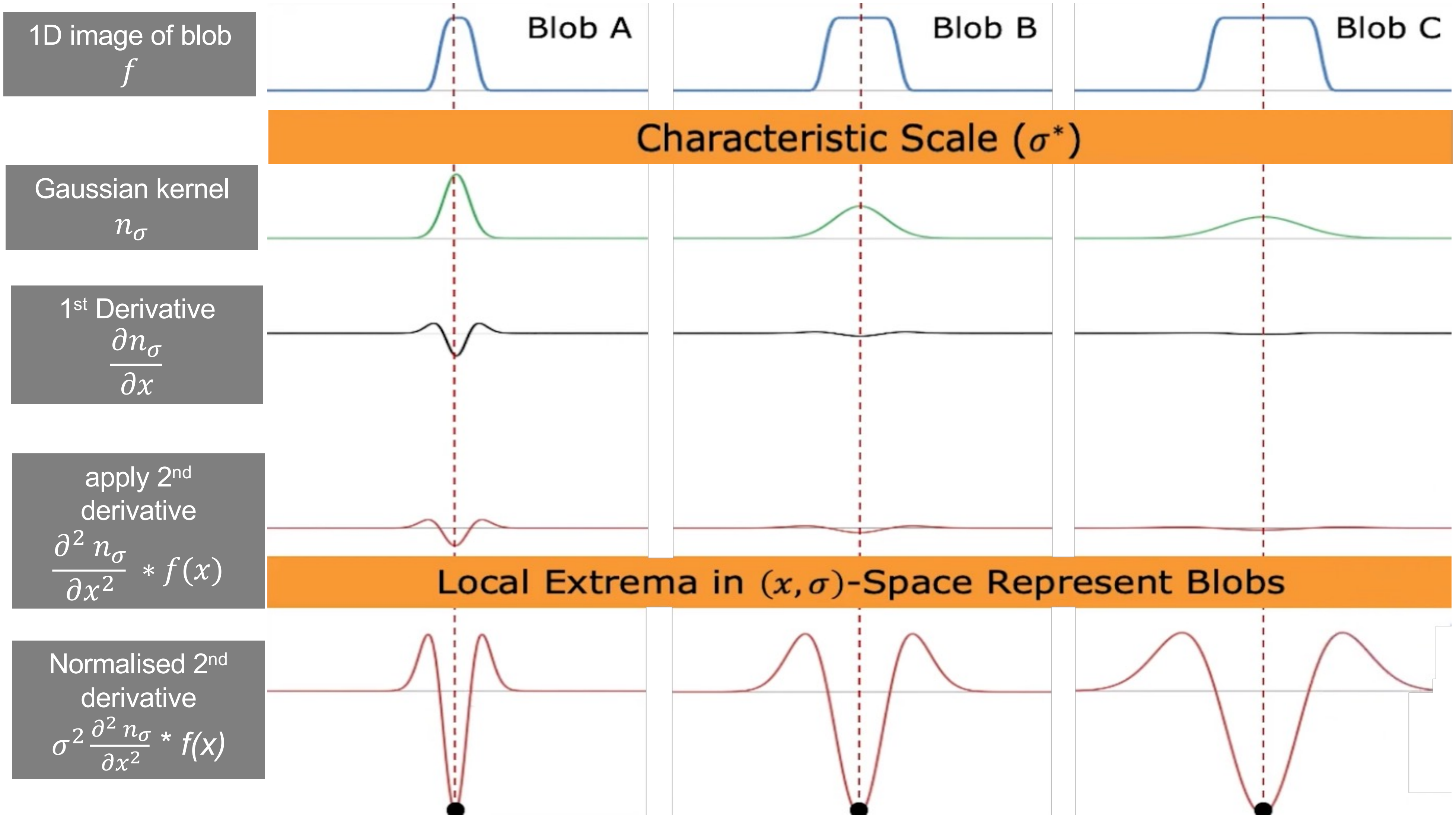
Normalized 2nd Derivative of Gaussian



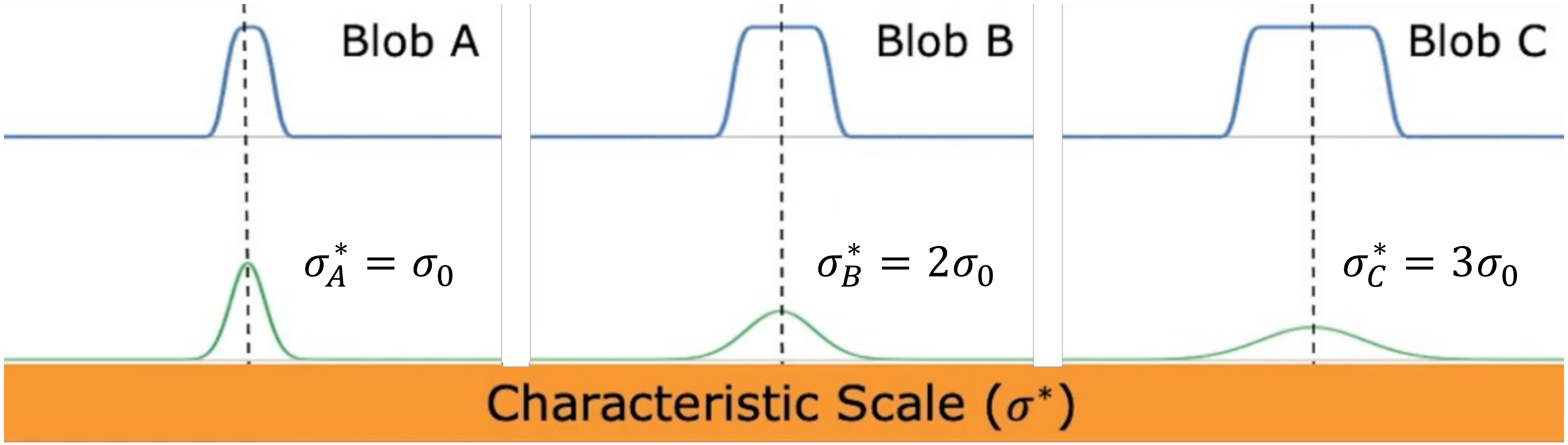
Effect of changing σ on Normalized 2nd Derivatives



Characteristic Scale of Blobs



Characteristic Scale Measures Size of a Blob



- Characteristic Scale: The σ value at which σ-normalised 2nd derivative reaches its peak value.
- Characteristic Scale is a valid measure of the SIZE of the blob. That is:

$$\frac{size\ of\ blob\ A}{size\ of\ blob\ B} \approx \frac{\sigma_A^*}{\sigma_B^*}$$

Summary on Steps of Blob Detection in 1D

1. Given a 1D signal $f(x)$, convolve it with σ-normalized 2nd derivative function:

   Compute: $\sigma^2 \frac{\partial^2 n_\sigma}{\partial x^2} * f(x)$ at different scales $(\sigma_0, \sigma_1, \dots, \sigma_k)$.

2. Find $(x^*, \sigma^*) = \max\limits_{(x,\sigma)} \left| \sigma^2 \frac{\partial^2 n_\sigma}{\partial x^2} * f(x) \right|$

3. Blob position = x^*

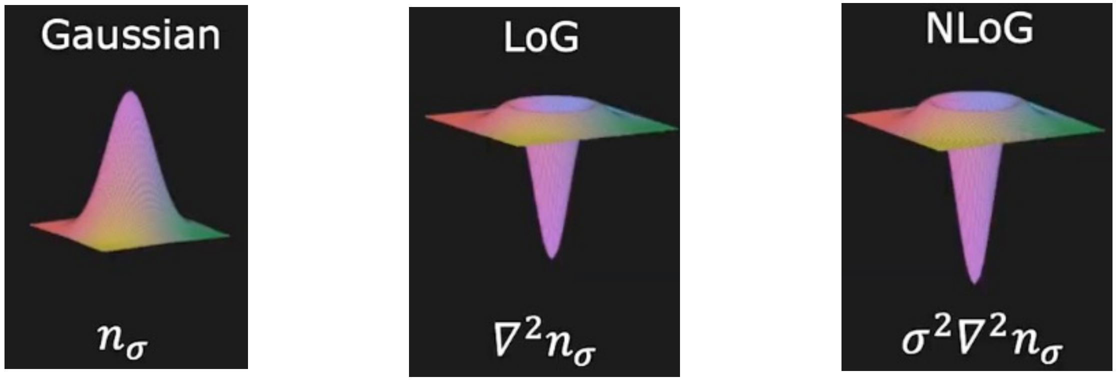
4. Blob size = σ^*

Blob Detection in 2D

- For 2D image $I(x, y)$, use **Normalized Laplacian of Gaussian** (NLoG) for blob detection:

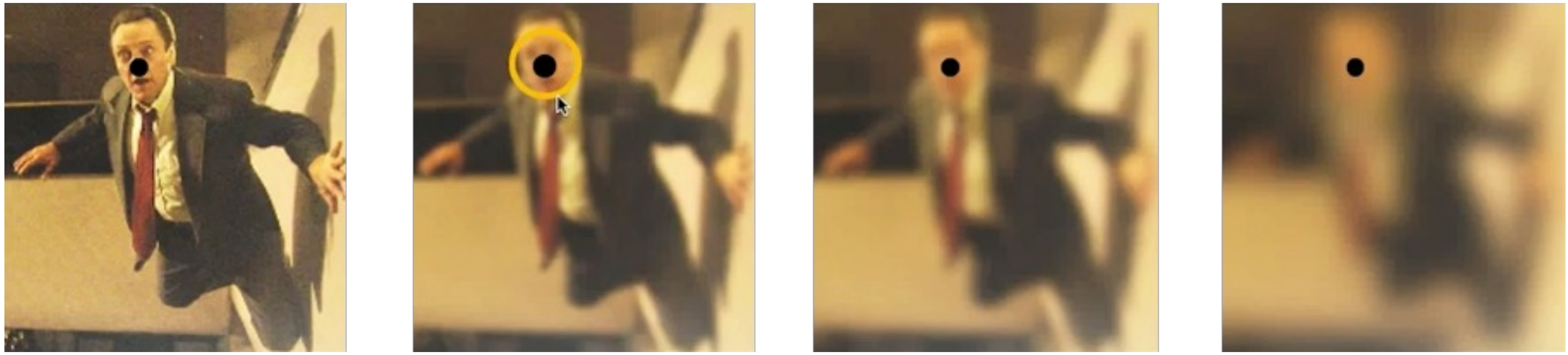
Laplacian

$$\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$$

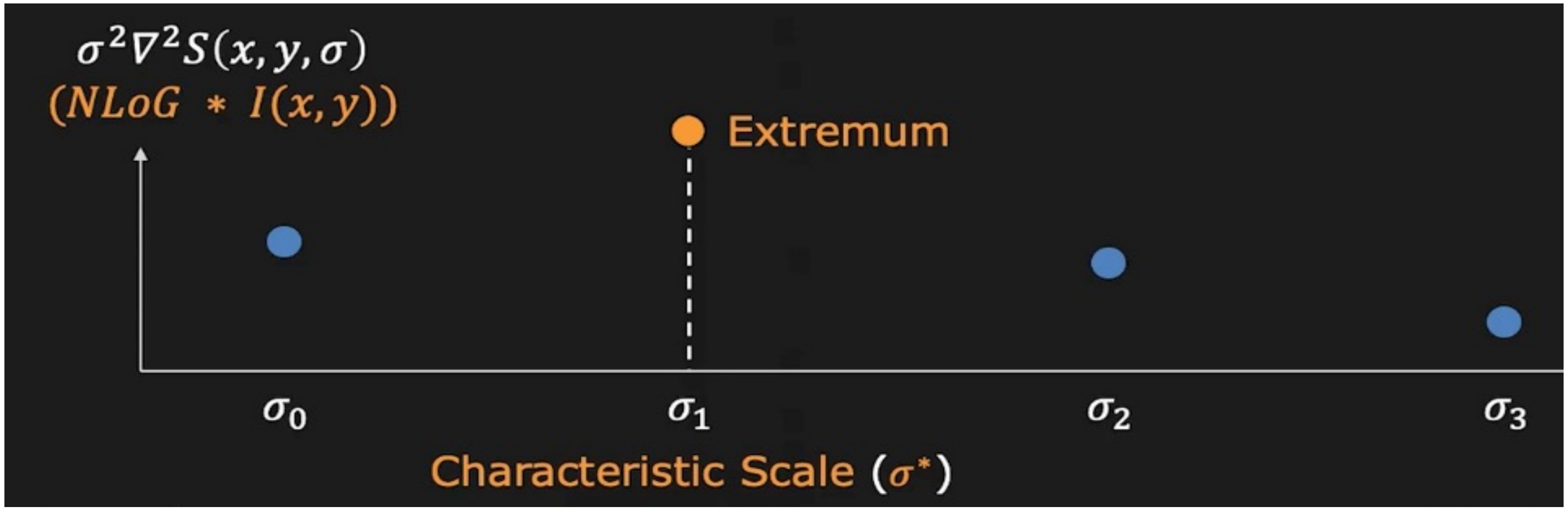


- Location of blobs found by **Local Extrema** after applying **NLoG at many scales**.

Example of Detecting an Interesting Blob



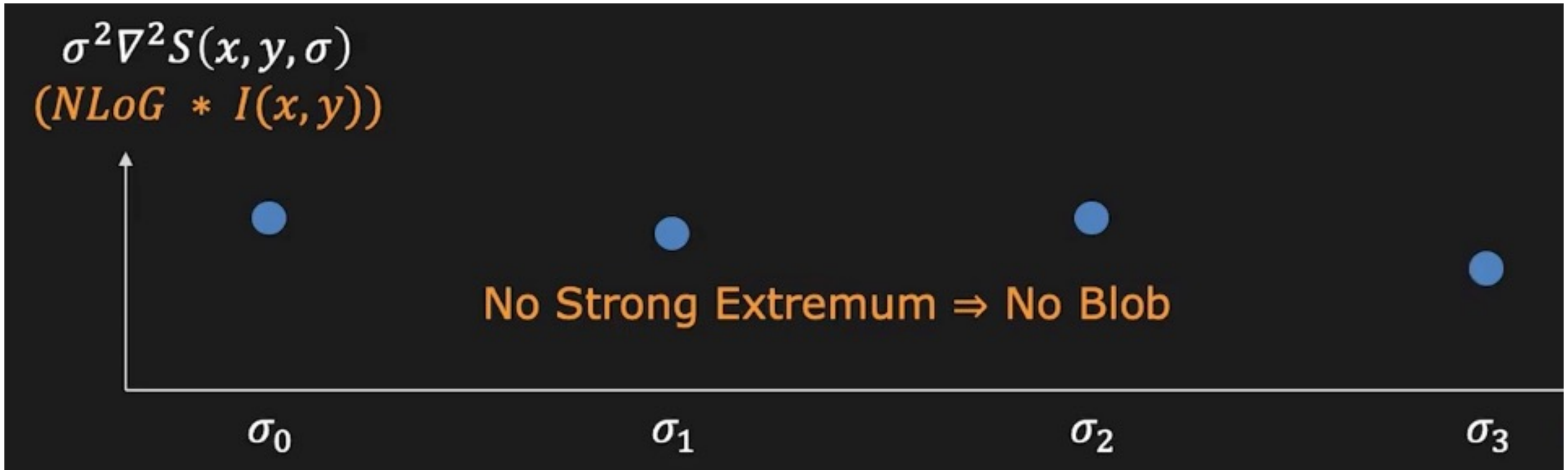
$S(x, y, \sigma_0)$ $S(x, y, \sigma_1)$ $S(x, y, \sigma_2)$ $S(x, y, \sigma_3)$



Source: Lindeberg

Example of Detecting a non-blob

$S(x, y, \sigma_0)$ $S(x, y, \sigma_1)$ $S(x, y, \sigma_2)$ $S(x, y, \sigma_3)$



Source: Lindeberg

Summary on Steps of Blob Detection in 2D

1. Given an image $I(x, y)$, convolve it with $NLoG$ at many scales of σ .

   Compute: $(\sigma^2\, \nabla^2 n_\sigma) * I(x, y)$ at different scale $(\sigma_0, \sigma_1, \dots, \sigma_k)$.

2. Find $(x^*, y^*, \sigma^*) = \max_{(x,y,\sigma)} \left|(\sigma^2\, \nabla^2 n_\sigma) * I(x, y)\right|$

3. Blob position = (x^*, y^*)

4. Blob size = σ^*

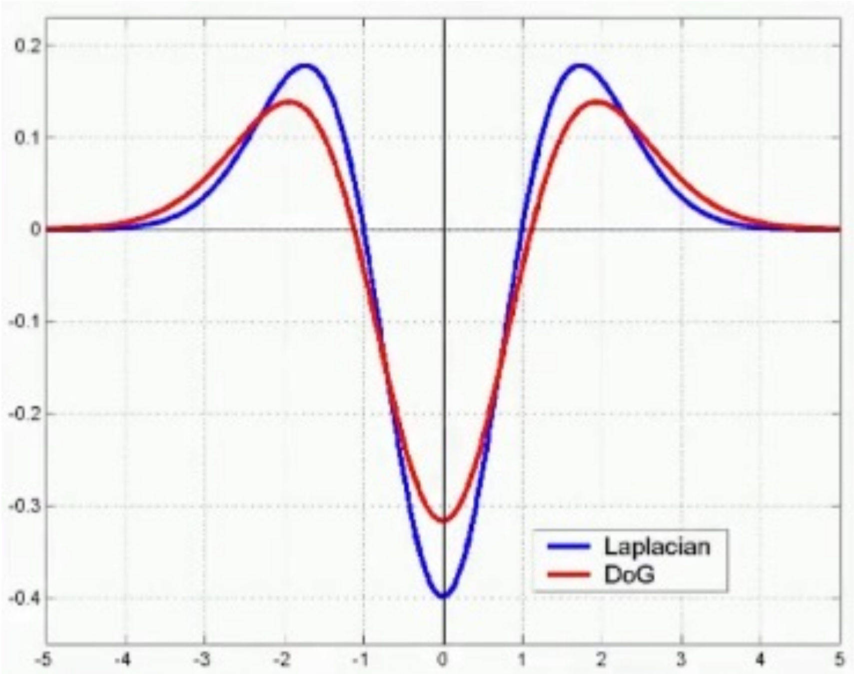
DoG is fast approximation of NLoG

- Is there a faster way to compute NLoG?
- Difference of Gaussian (DoG):

NLoG

$$DoG = (n_{s\sigma} - n_\sigma) \approx (s-1)\sigma^2\nabla^2 n\_\sigma$$

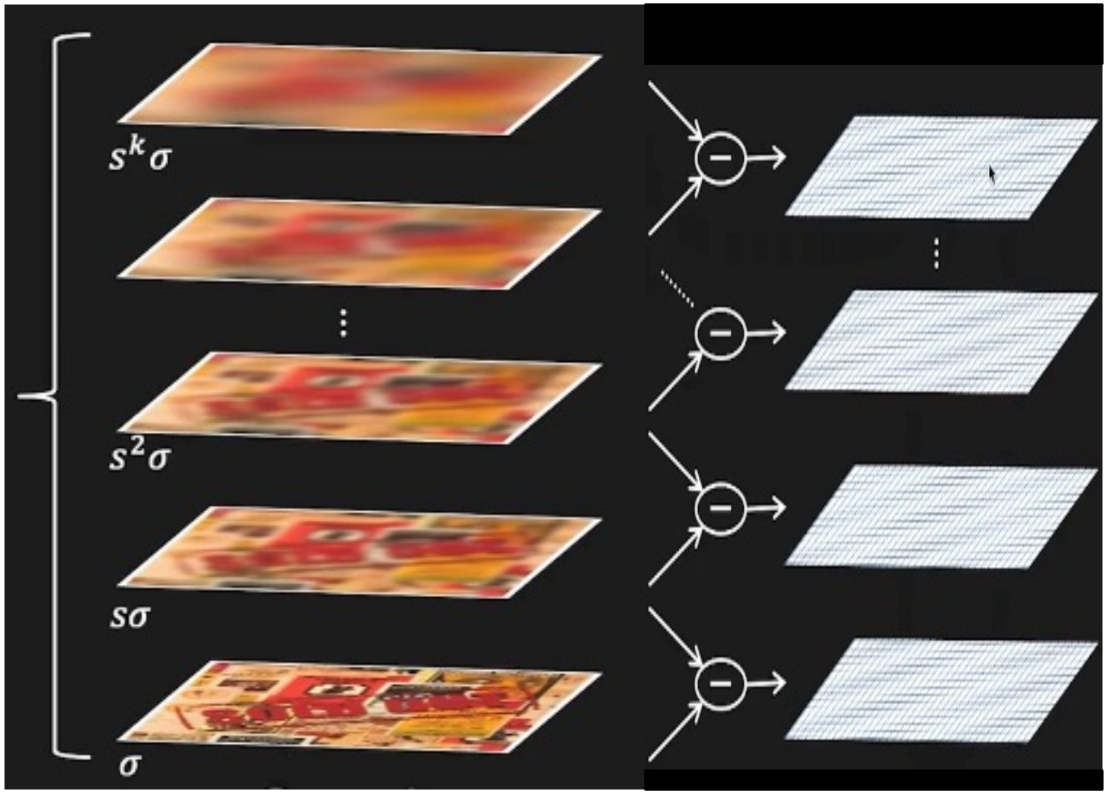
- s is different multipliers (octave) of σ.



$$DoG \approx (s-1)NLoG \qquad s > \mathbf{1}$$

Extracting SIFT Interest Points (1)

Image
$I(x,y)$

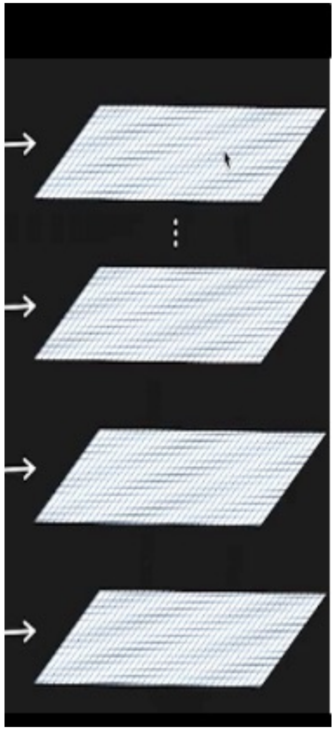


Gaussian Scale Space
$S(x,y,\sigma)$

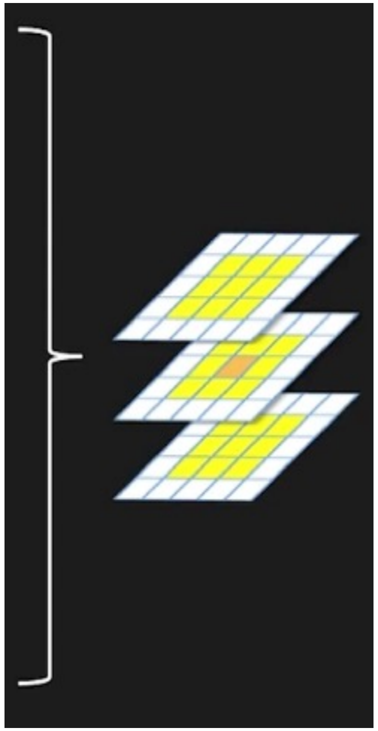
Difference of Gaussians (DoG)
$\approx (s-1)\sigma^2\nabla^2 S(x,y,\sigma)$

Source: Lowe

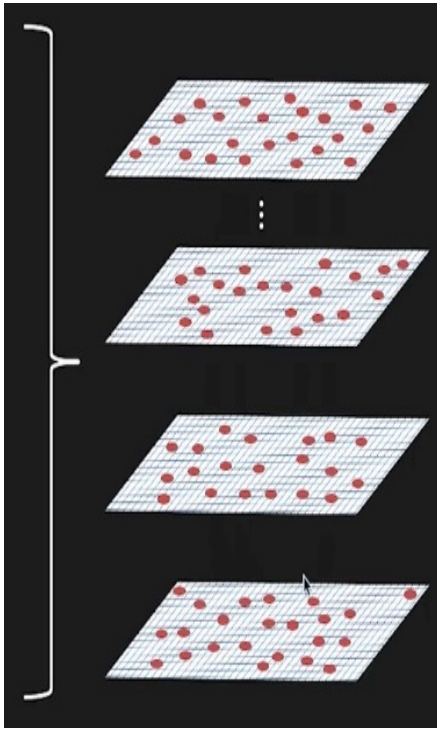
Extracting SIFT Interest Points (2)



Difference of Gaussians (DoG)
$\approx (s-1)\sigma^2\nabla^2 S(x,y,\sigma)$



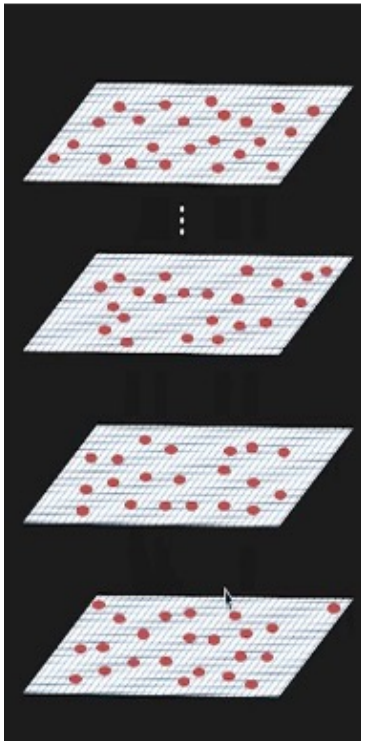
Find peaks (extrema) in every 3x3 grid



Candidates for Interest Point

Source: Lowe

Extracting SIFT Interest Points (3)



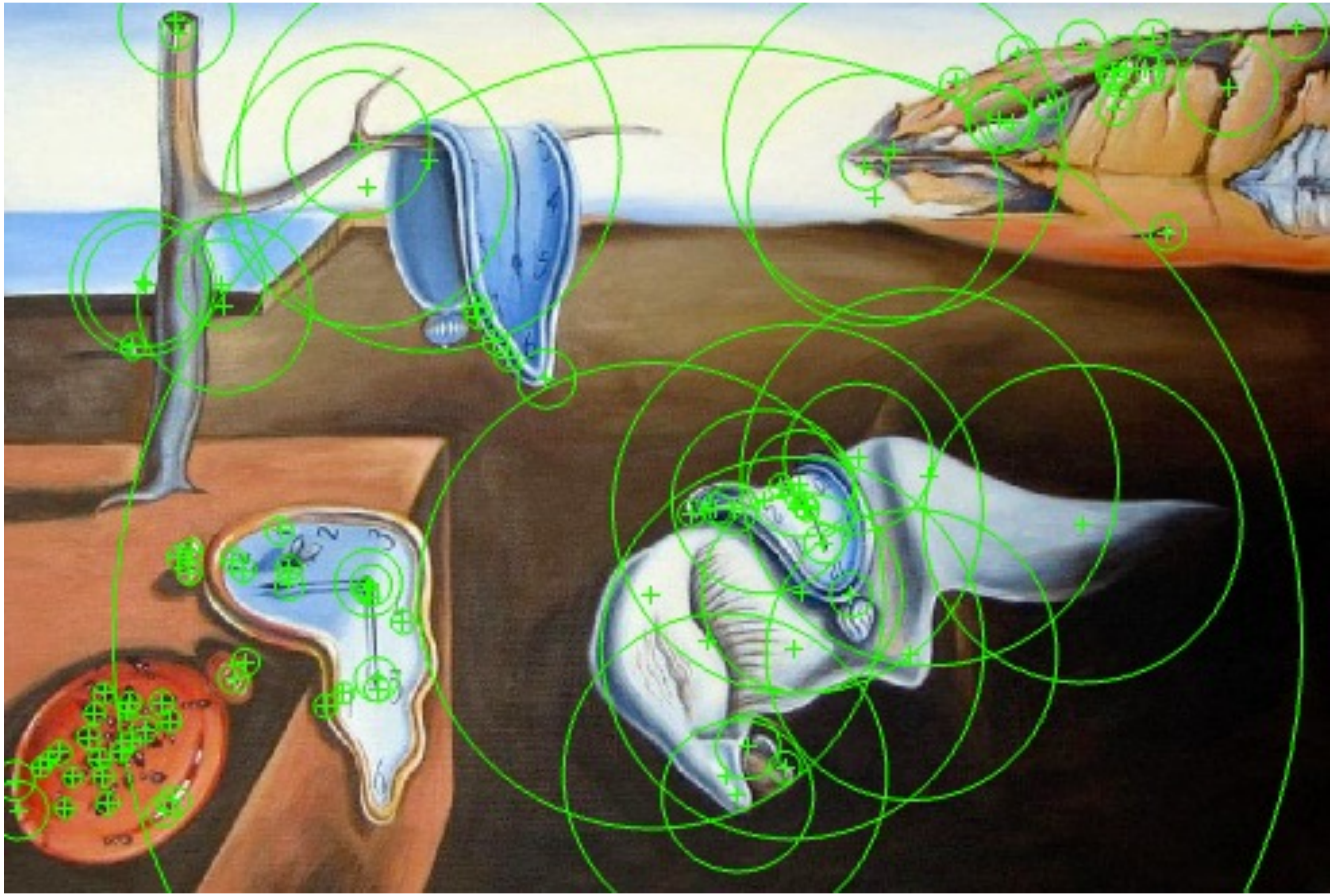
Candidates for Interest Point

SIFT Interest Points after removing weak peaks

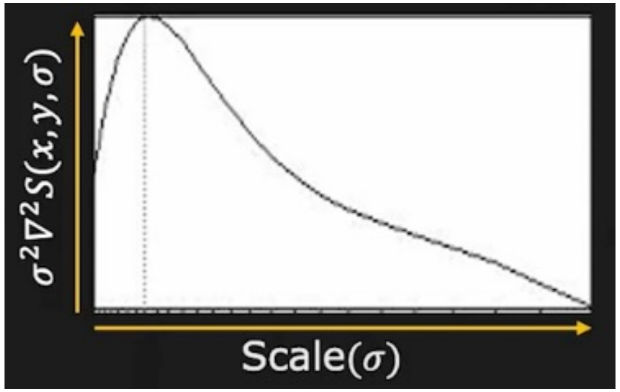
Annotated SIFT Features

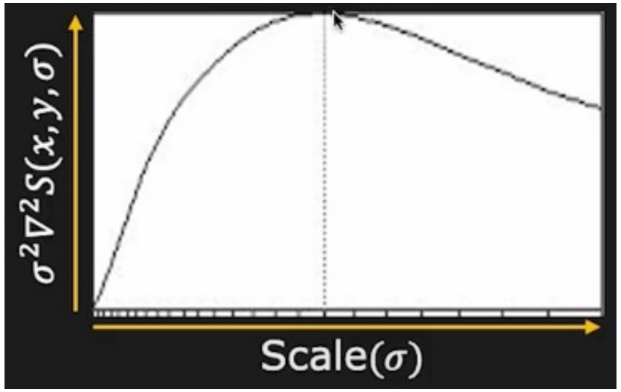
Source: Lowe

Example of SIFT Interest Points Detector



SIFT Scale Invariance



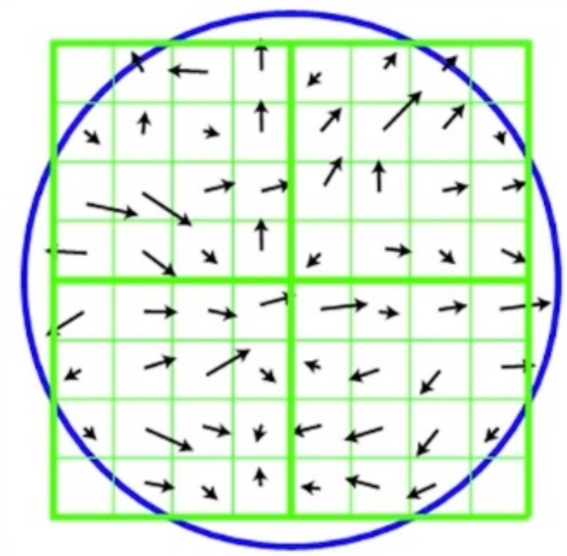


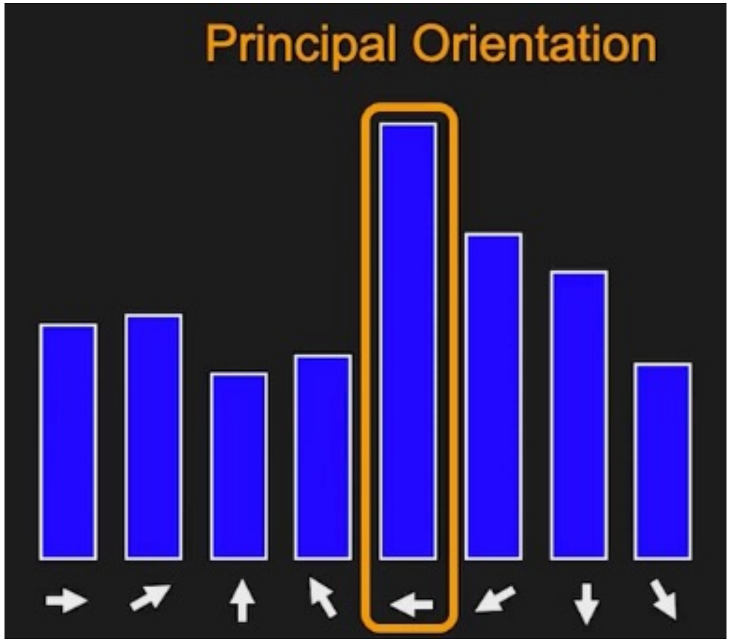
Ratio of Blob Sizes

$$= \frac{\sigma_1^*}{\sigma_2^*}$$

Source: Mikolajczyk

Detect Feature Orientation



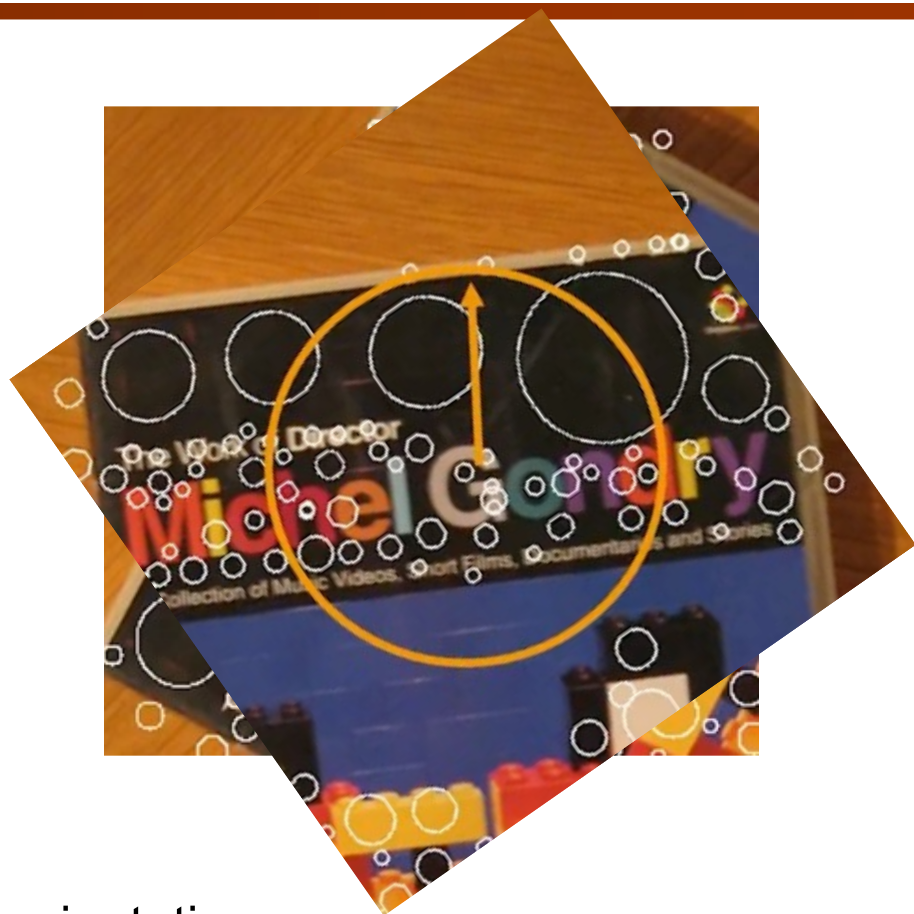


- Image gradient directions is calculated by:

$$\theta = \tan^{-1}\frac{\partial I/\partial y}{\partial I/\partial x}$$

- Build histogram of direction for every pixel (8 directions).
- Principle Orientation is the one with highest count.

SIFT Rotation Invariance



- Correct rotation based on principal orientation.
- We can now match objects of different scale and different orientation.

SIFT Descriptor (signature)

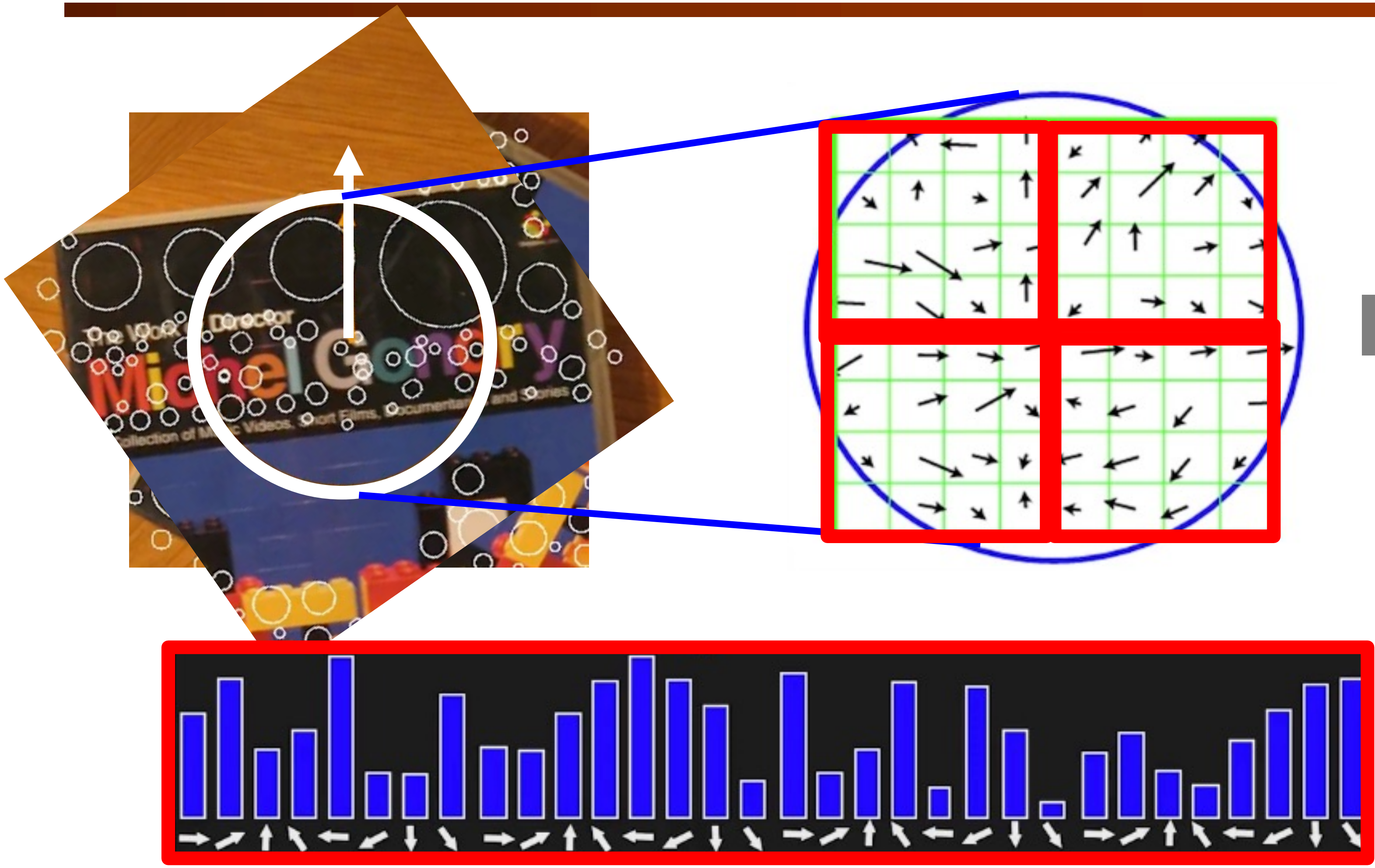


Image gradients

- Normalized Histogram is the featured descriptor or signature.
- It is invariant to Rotation, Scaling and Brightness.

Matching of SIFT Descriptors

- Goal, match two SIFT features from two images, with descriptors $H_1(k)$ and $H_2(k)$. (These are histograms of orientations.)
- Possible measures:
  1. L2 Distance:

     $d(H_1, H_2) = \sqrt{\sum_k \left(H_1(k) - H_2(k)\right)^2}$. (Smaller d = better match.)

  2. Normalized Correlation:

     $$d(H_1, H_2) = \frac{\sum_k [(H_1(k) - \overline{H_1})(H_2(k) - \overline{H_2})]}{\sqrt{\sum_k (H_1(k) - H_1)^2} \sqrt{\sum_k (H_2(k) - H_2)^2}},$$

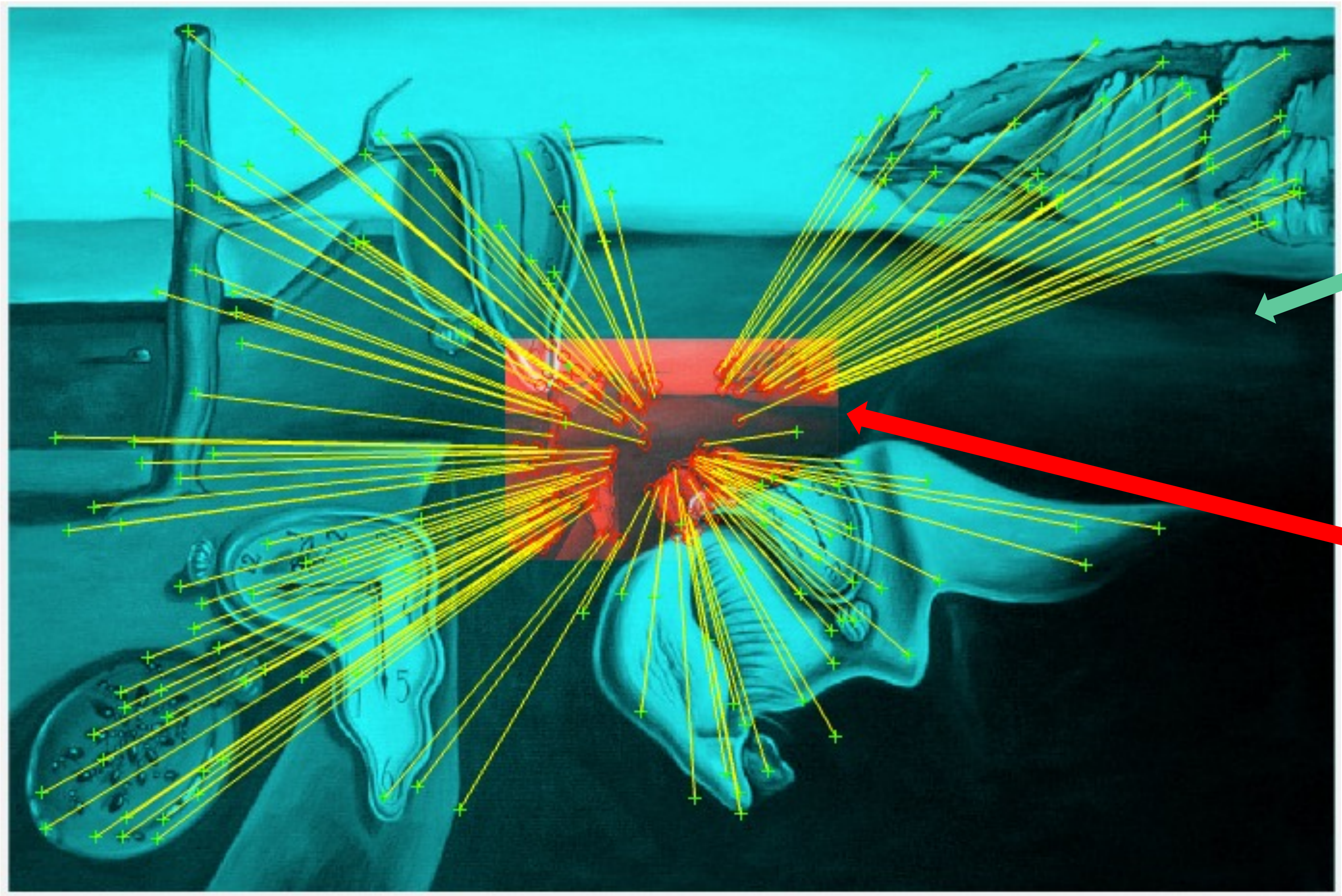
     where $\overline{H_i} = \frac{1}{N} \sum_{k-1}^{N} H_i(k)$. (Larger d = better match.)

  3. **Intersection:**

     $$d(H_1, H_2) = \sum_k \min(H_1(k), H_2(k))$$

     (Larger d = better match.)

SIFT Results: Scale Invariance



Original Image

1/4 size Image

SIFT Results: Rotation Invariance

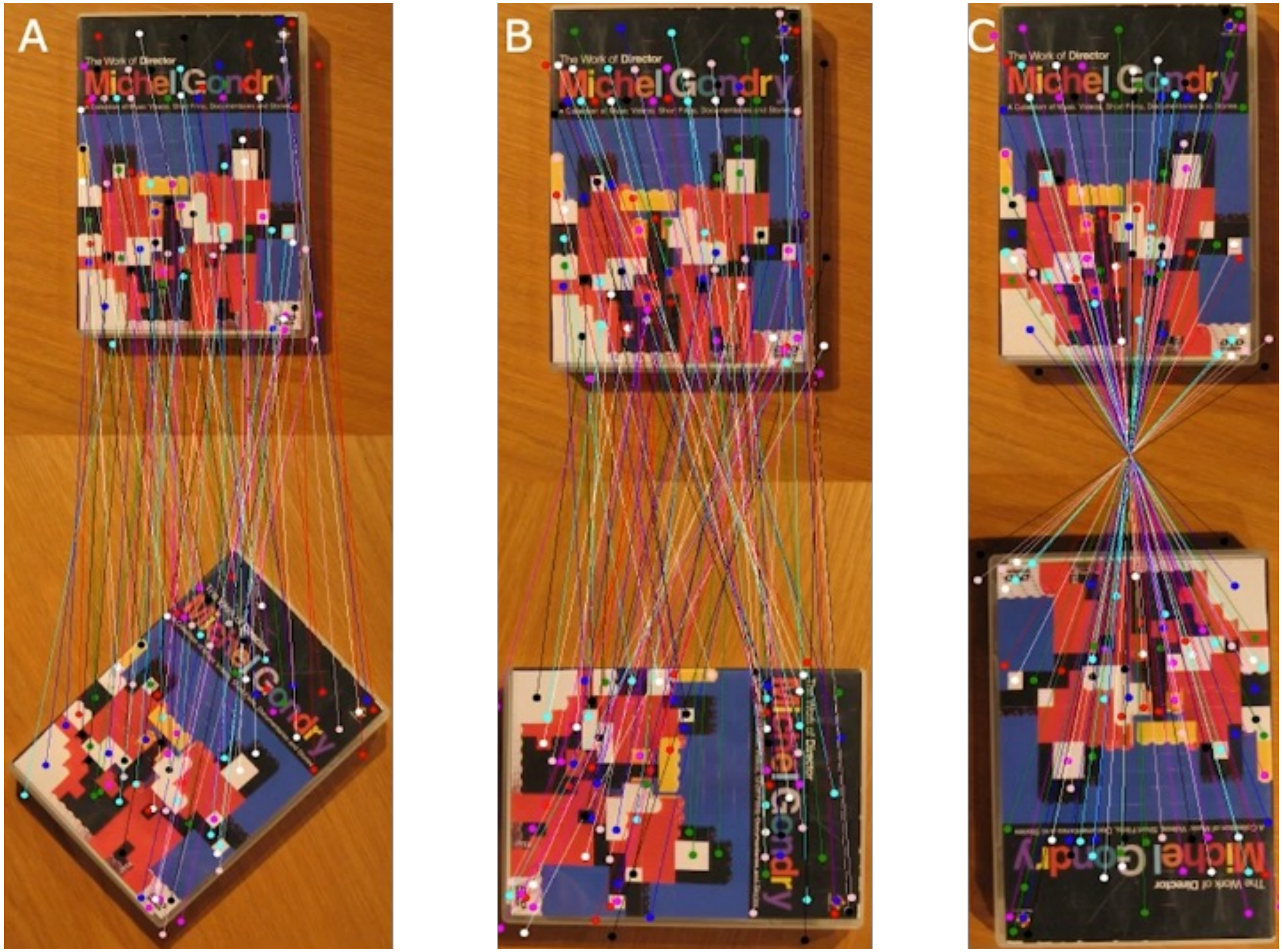
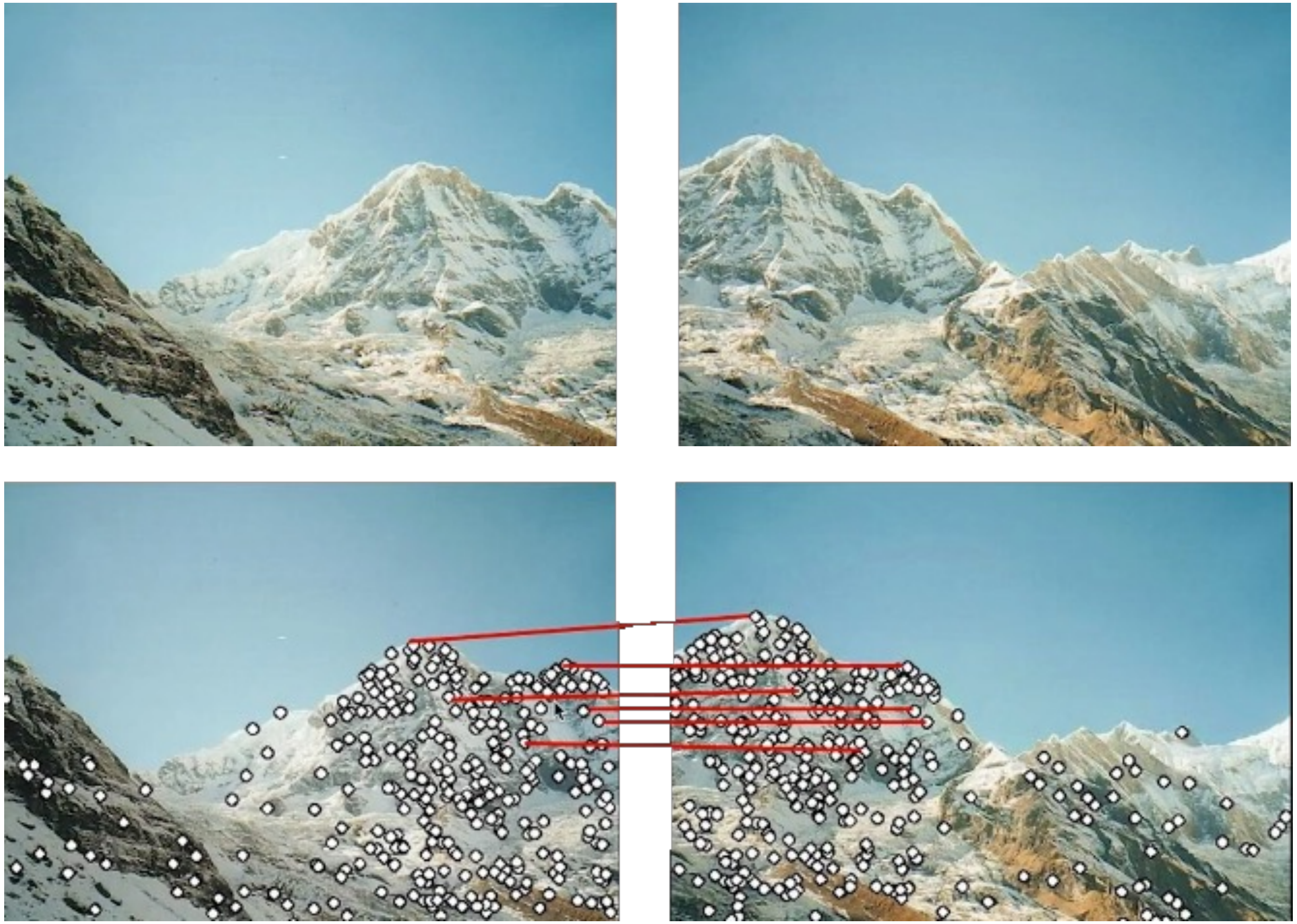


Photo Stitching using SIFT (1)

Image 1

Image 2



Matched SIFT Interest Points

Photo Stitching using SIFT (2)

SIFT for tracking

Matlab Support for Feature Detection

- Many other feature detection methods have been proposed since Lowe's paper in 2004.
- Here are the different algorithms that are implemented in Matlab, some of these are extensions to SIFT.

Detector	Feature Type	Function	Scale Independent
FAST [1]	Corner	detectFASTFeatures	No
Minimum eigenvalue algorithm [4]	Corner	detectMinEigenFeatu	No
Corner detector [3]	Corner	detectHarrisFeature	No
SIFT [14]	Blob	detectSIFTFeatures	Yes
SURF [11]	Blob	detectSURFFeatures	Yes
KAZE [12]	Blob	detectKAZEFeatures	Yes
BRISK [6]	Corner	detectBRISKFeatures	Yes
MSER [8]	Region with uniform intensity	detectMSERFeatures	Yes
ORB [13]	Corner	detectORBFeatures	No