Welcome to Computer Vision

Computer Vision

Dr. Muhammad Tahir

DEPARTMENT OF COMPUTER SCIENCE,

FAST-NUCES, Peshawar

Course Details

LECTURES: Monday
& Wednesday

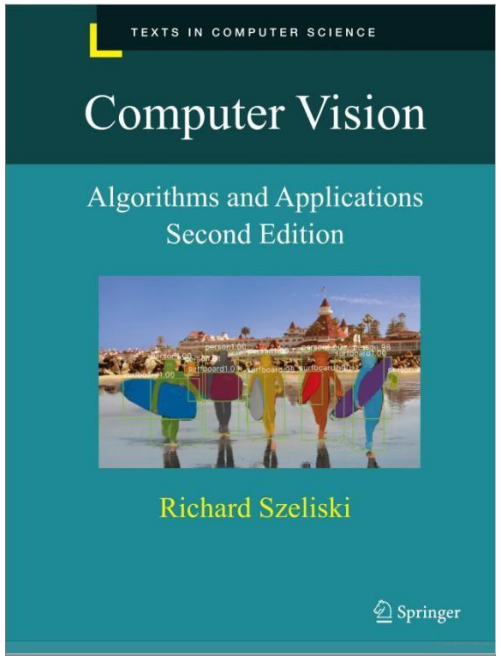
TIMINGS:

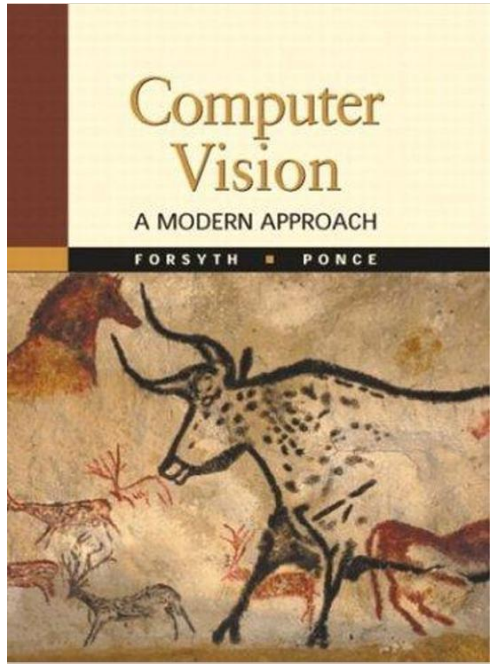
9:30 am – 11:00 am

MY OFFICE:

OFFICE HOURS:

EMAIL: m.tahir@nu.edu.pk





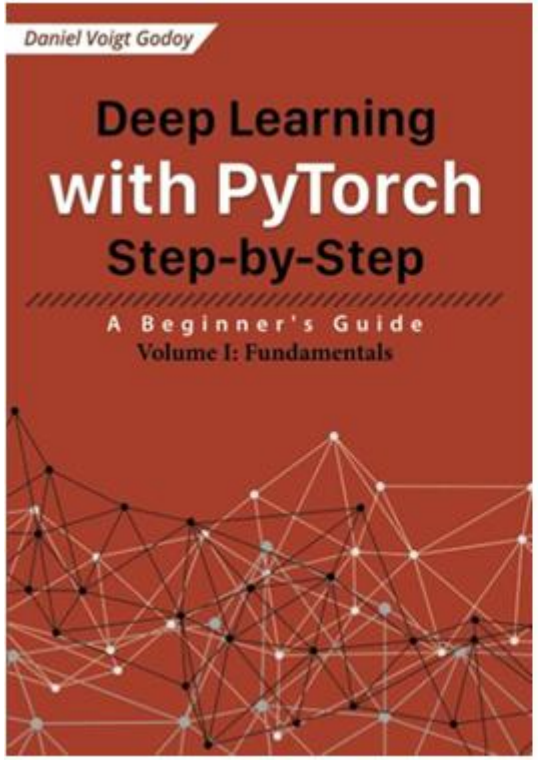
References

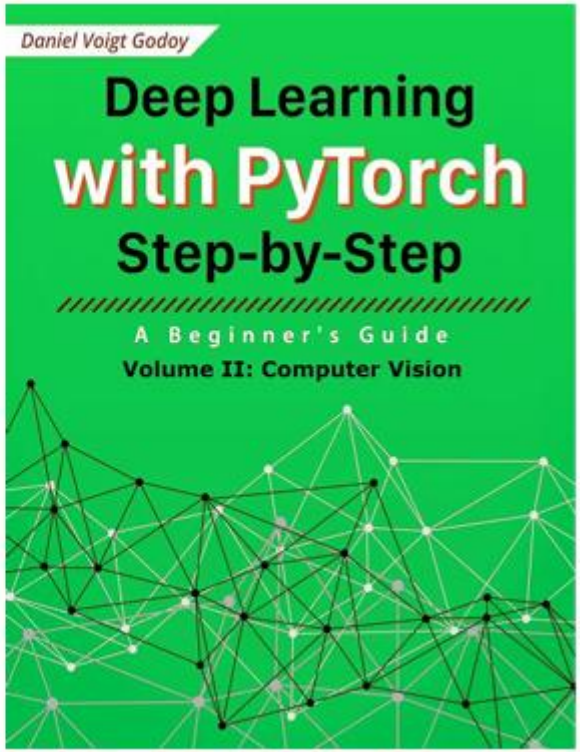
The material in these slides are based on:

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

SIFT

Overview

- We know that how to detect edges and corners in images
- We also know that how to compute the boundaries of objects and images
- We now consider the problem of Object Recognition

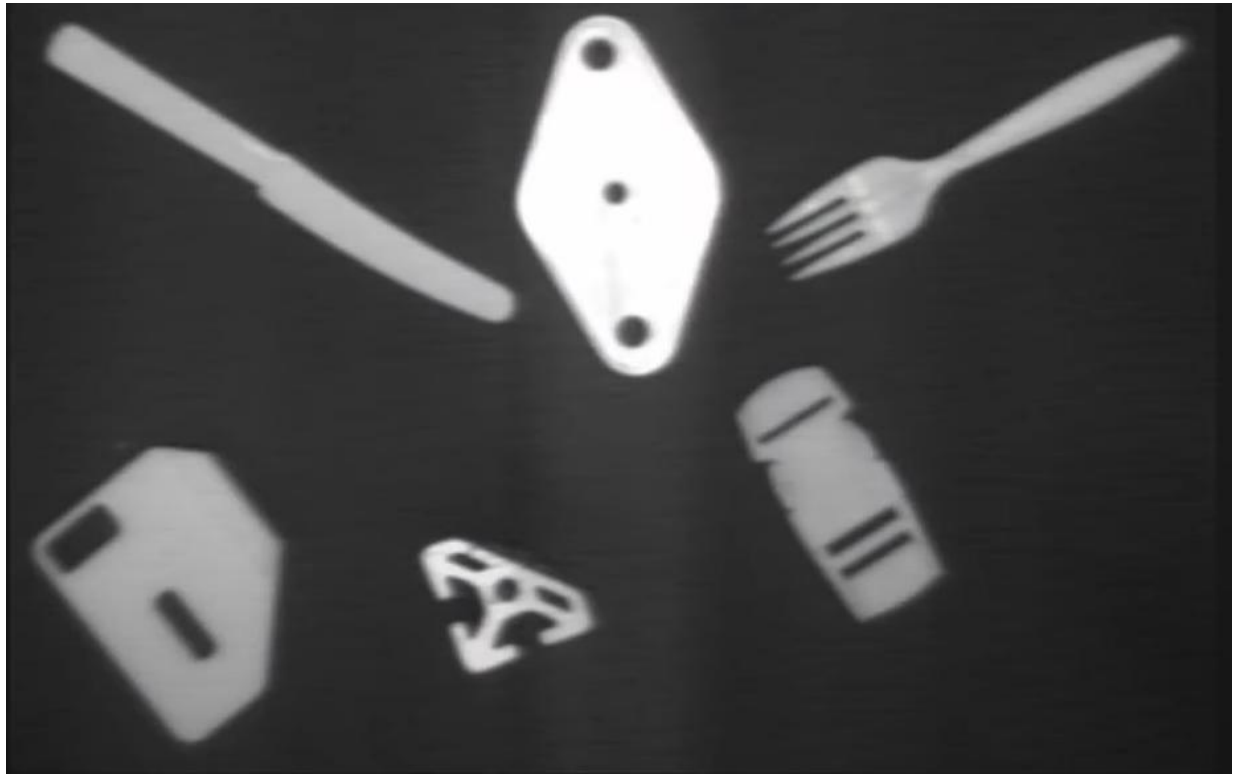
Overview

- Edges and Corners are not interesting enough for a lot of applications
- Objects may appear very complex in different scenes
- We need to detect and match features that are more descriptive and unique
- This will be achieved using SIFT feature detector

Recognizing Objects

- How would you recognize the following types of objects?

- One solution may be to first threshold this image and then using some geometric properties of these objects to recognize them



Objects on an assembly line

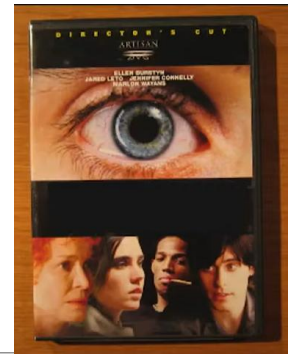
Recognizing Objects

- How would you recognize the following types of objects?

License Plates

Recognizing Objects

Template



- How would you recognize the following types of objects?

- Template Matching may be used
- But the objects do not appear in exactly the same form of the given template
- The objects in the image are **rotated**, **scaled**, and **occluded**
- So, we need to generate a large number of rotated and scaled versions of the template
- That is computationally impractical

2D Image

Solution: Find and Match Interesting Points or Features

- Extract directly from the template some very descriptive and unique features
- The same features can also be computed from the 2D image
- Match those features from both the Template and the image
- This way we can recognize the objects

- This is achieved using SIFT detector

SIFT Detector

- **S**cale **I**nvariant **F**eature **T**ransform (SIFT)
- It is used for image alignment and 2D object recognition

- We are going to address the following:
  - What is an Interest Point?
  - Detecting Blobs
  - SIFT Detector
  - SIFT Descriptor

What is an Interest Point?

What is an Interest Point?

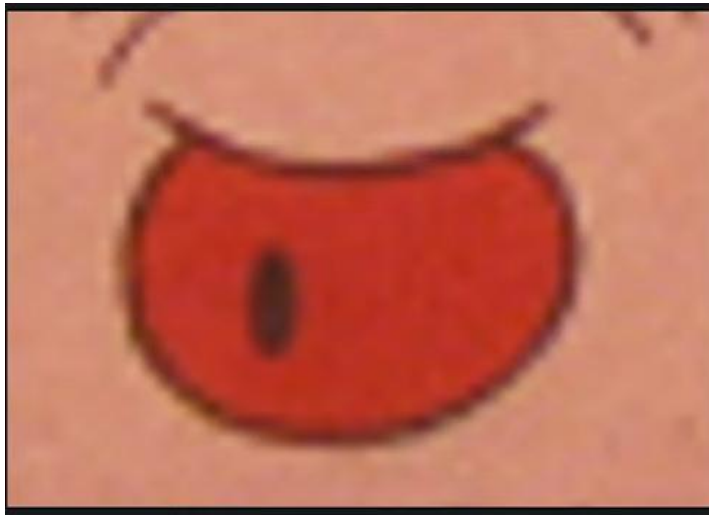
- Raw images are hard to match
- Different size, orientation, lighting, brightness etc.

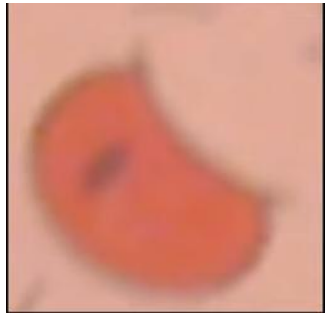
What is an Interest Point?

Consider the two patches:
Different size, orientation, lighting, brightness etc.

Removing Sources of Variation

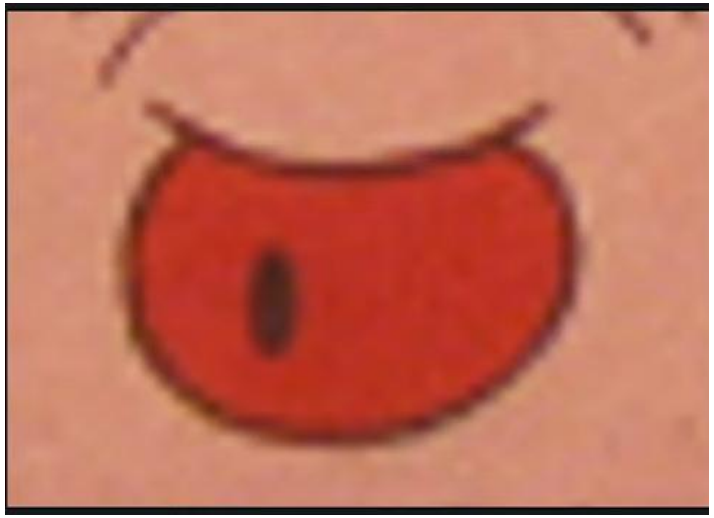
- Any Interest Point Detector should be able to remove or compensate for these variations
- Then the features from different images can be matched





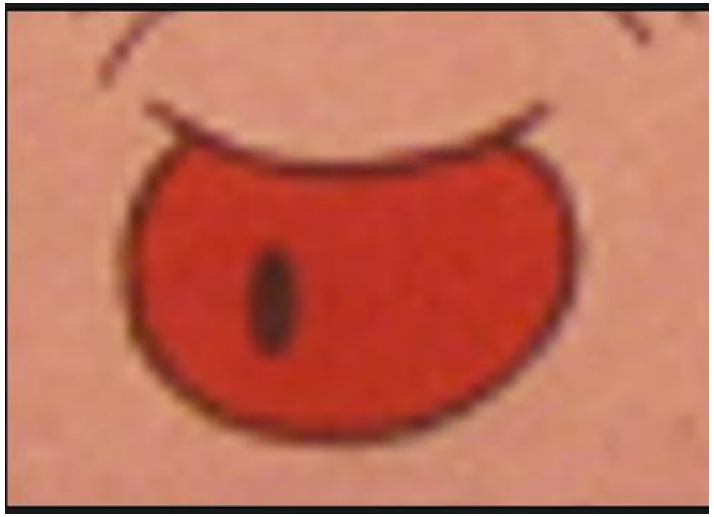
Removing Sources of Variation

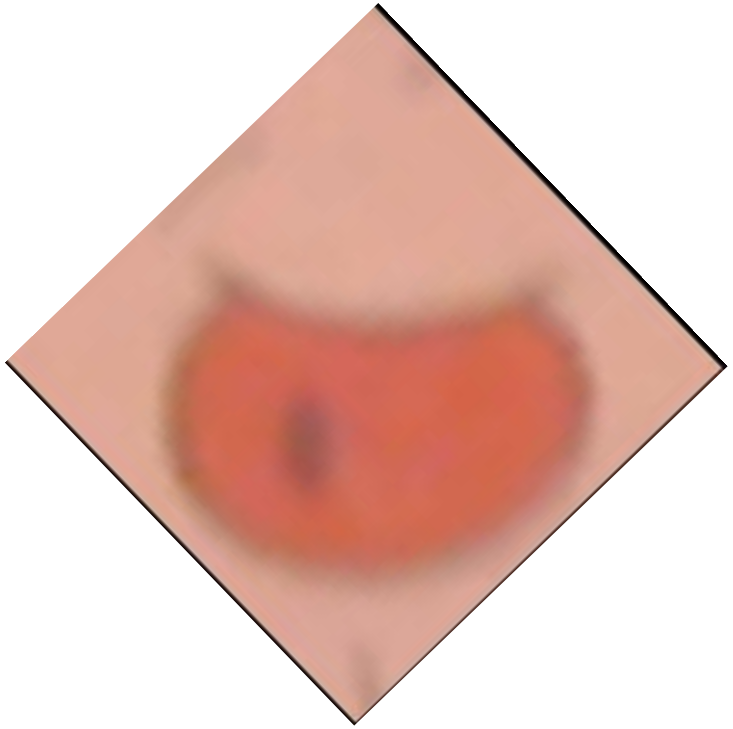
- Smaller patch can be magnified to compensate for scale differences



Removing Sources of Variation

- Patches can also be re-orient to match the orientation of the other





- Matching becomes easier if we can remove variations like size and orientation

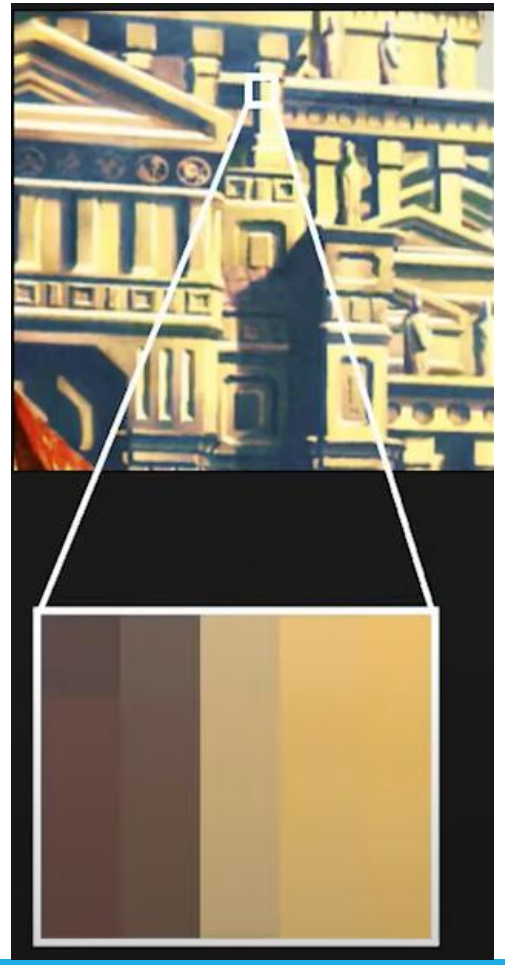
Some Patches are NOT Interesting

Similarly, the Interest Point Detector should not respond to a lot of the things in an image

What is an Interesting Point / Feature?

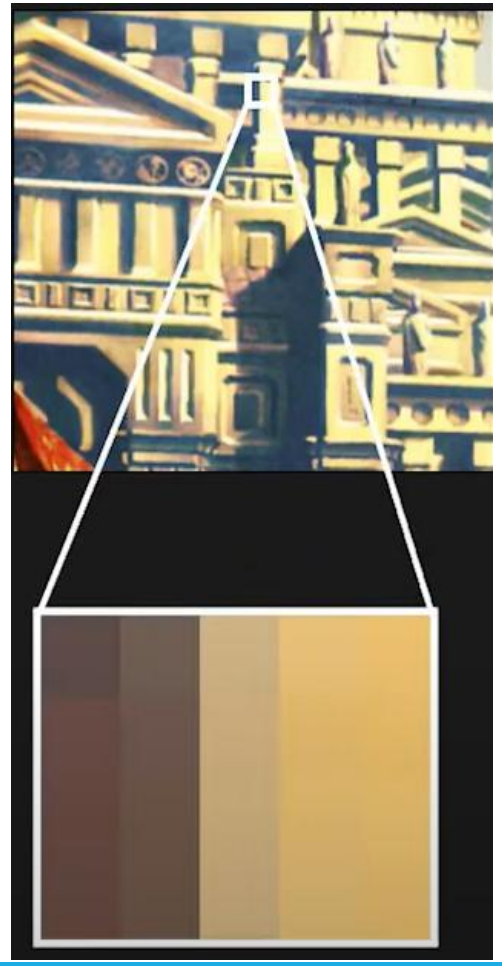
- Has **rich image content** (brightness variation, color variation etc.) within the local window
- Has well-defined **representation** (signature) for matching/comparing with other points
- Has a well-defined **position** in the image
- Should be **invariant to image rotation and scaling**
- Should be **insensitive to lighting** changes

Are Lines/Edges Interesting?



Are these edges interesting enough?

Are Lines/Edges Interesting?



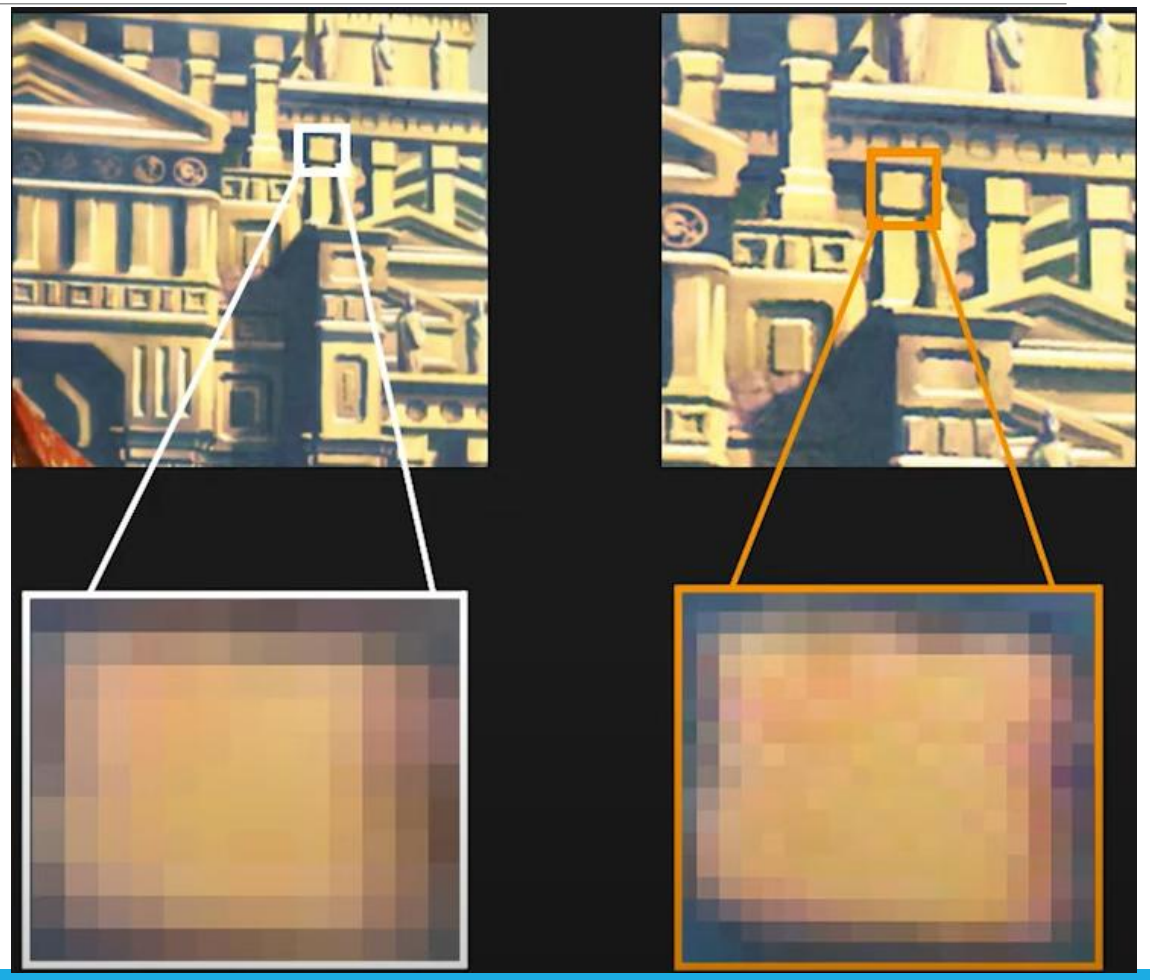
- They are not unique and not descriptive enough
- There are a lot of images with same level of brightness variation
- Cannot **localize** an edge

Are Corners Interesting?

- Yes, they are but for simple applications and not for recognizing objects

Are Blobs Interesting?

Yes, Blobs have fixed position and definite size

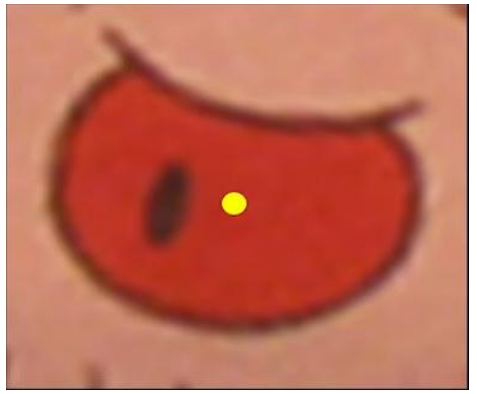


Blobs are Interest Points

- For a Blob-like feature to be useful, we need to be able to:

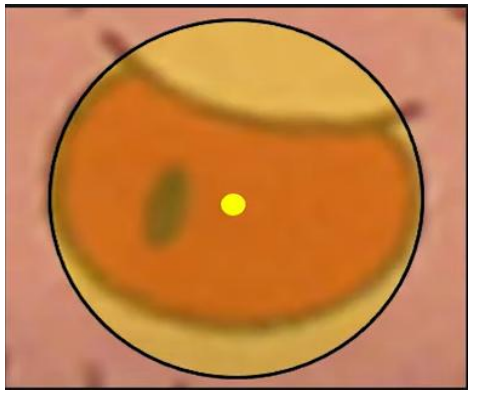
Blobs are Interest Points

- For a Blob-like feature to be useful, we need to be able to:
  - **Locate** the Blob



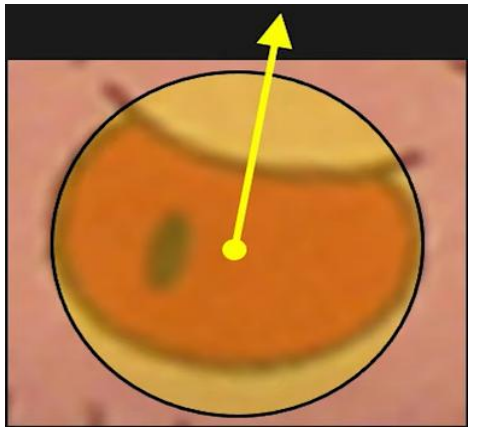
Blobs are Interest Points

- For a Blob-like feature to be useful, we need to be able to:
  - **Locate** the Blob
  - Determine its **size**



Blobs are Interest Points

- For a Blob-like feature to be useful, we need to be able to:
  - **Locate** the Blob
  - Determine its **size**
  - Determine its **orientation**



Blobs are Interest Points

- For a Blob-like feature to be useful, we need to be able to:
  - **Locate** the Blob
  - Determine its **size**
  - Determine its **orientation**
  - Formulate a **description** or signature that is independent of size and orientation

This description is very important for matching purposes

Detecting Blobs

Detecting Blobs

- How can we find blobs in an image?
- We will develop a technique that uses the derivatives of images (we already used them for detecting edges)
- We will start with 1-D signals and then extend the technique to 2-D image

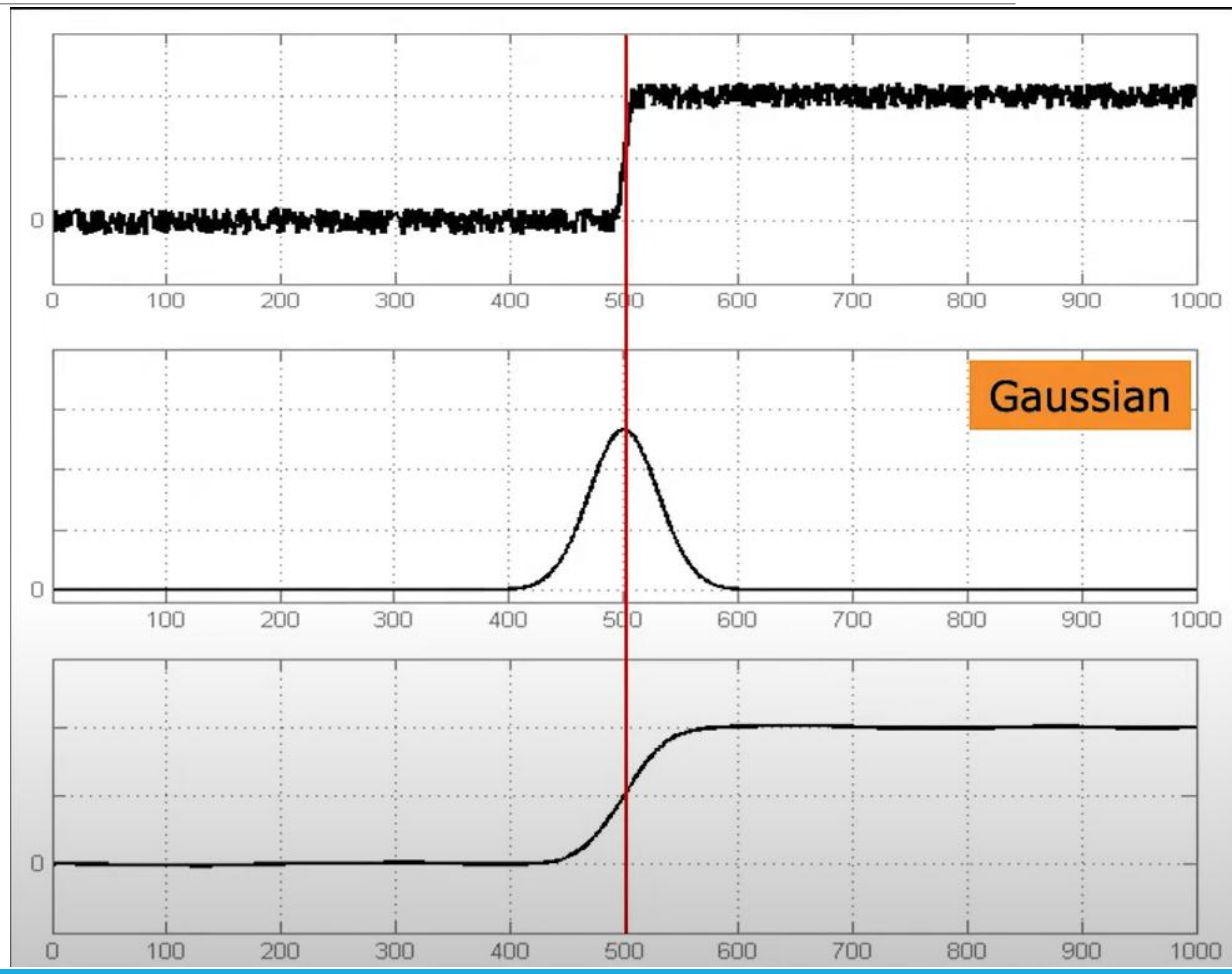
Review: Gaussian Filter

Gaussian Filter is used for removing noise

f

n_σ

$n_\sigma * f$



Review: Derivative of Gaussian

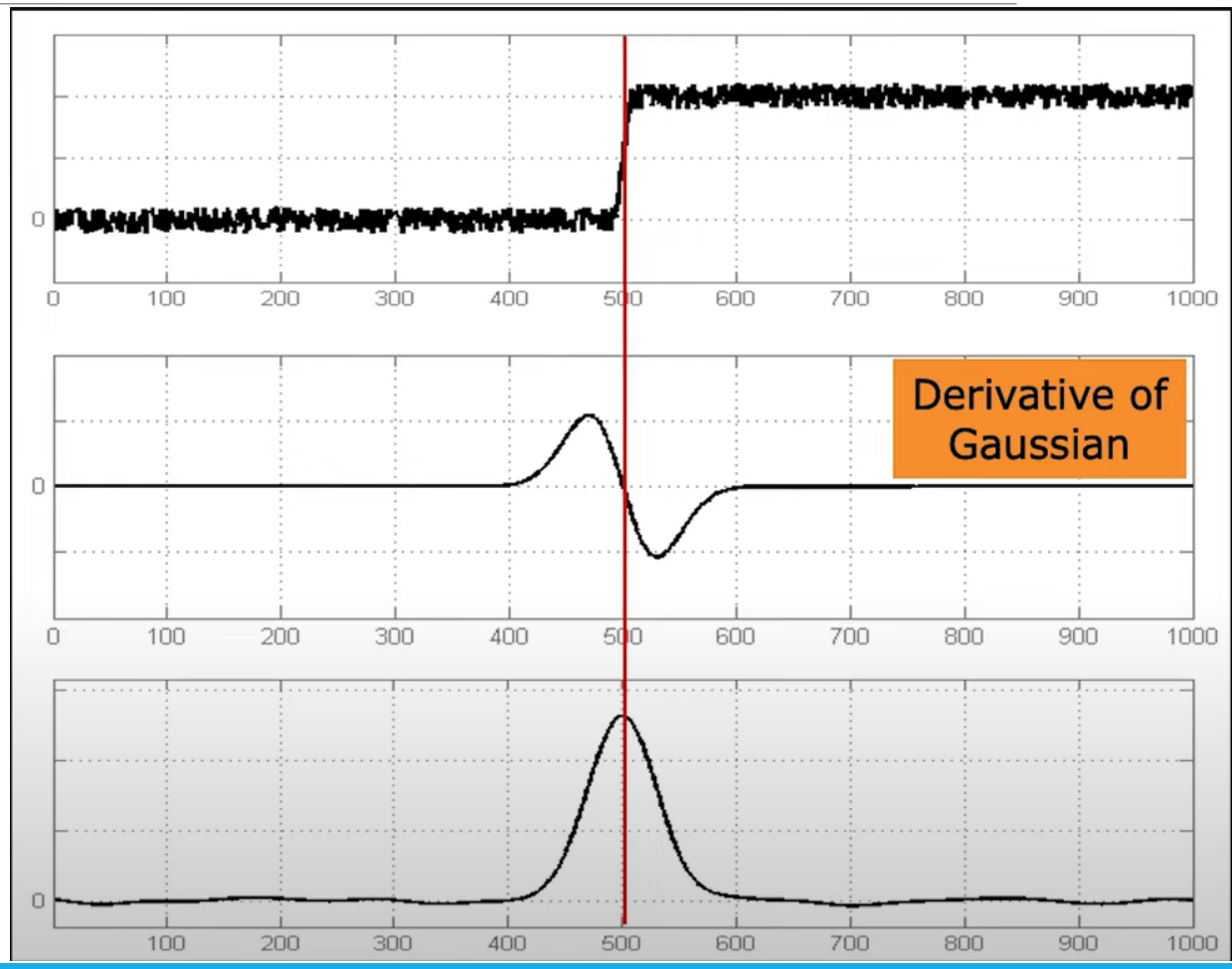
We can find the position of the edge by using the first derivative of the image

f

$\nabla(n_\sigma)$

Extremum of Derivative of Gaussian denotes an edge

$\nabla(n_\sigma) * f$



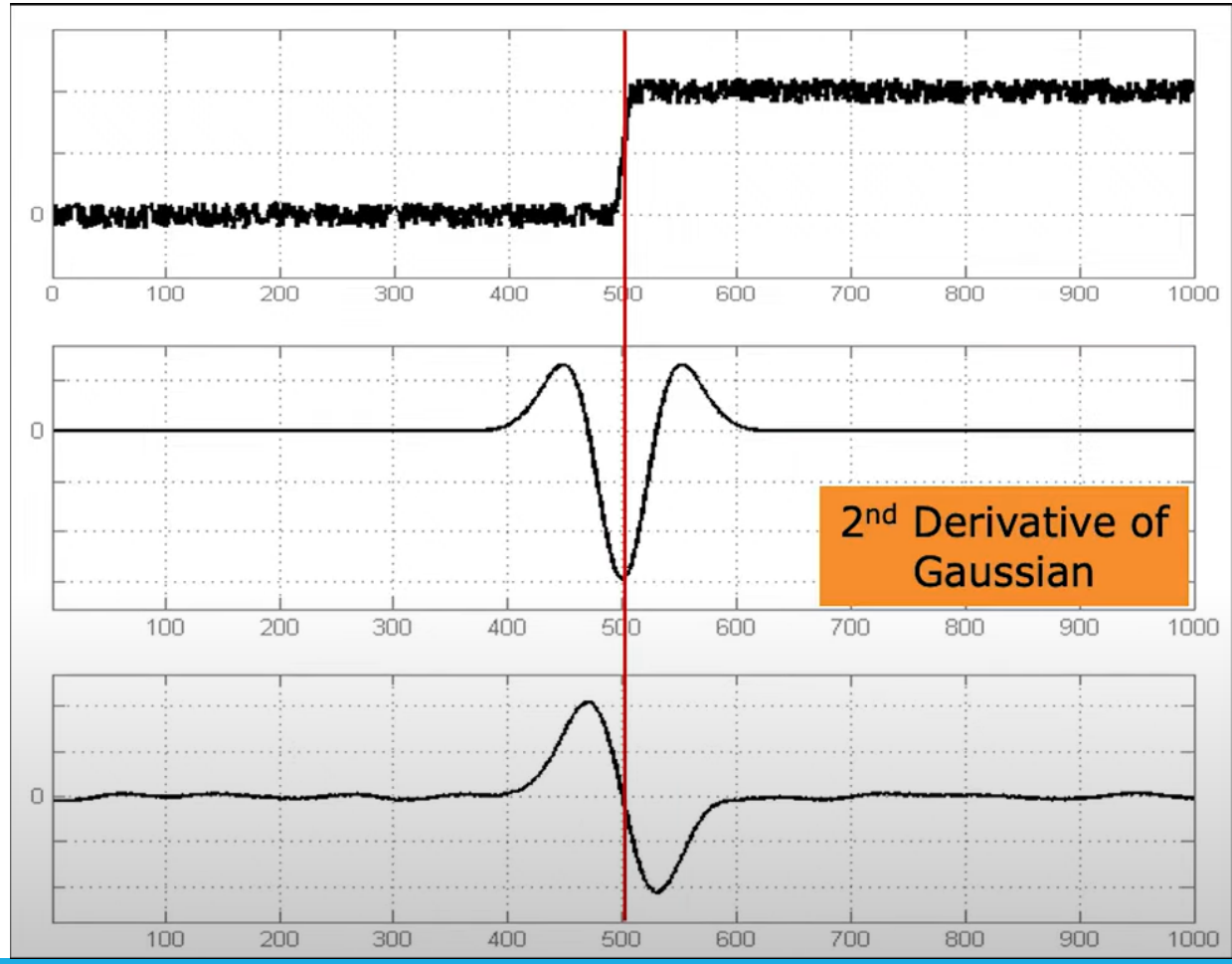
Review: 2nd Derivative of Gaussian

We can also find an edge using the 2nd derivative of an image

f

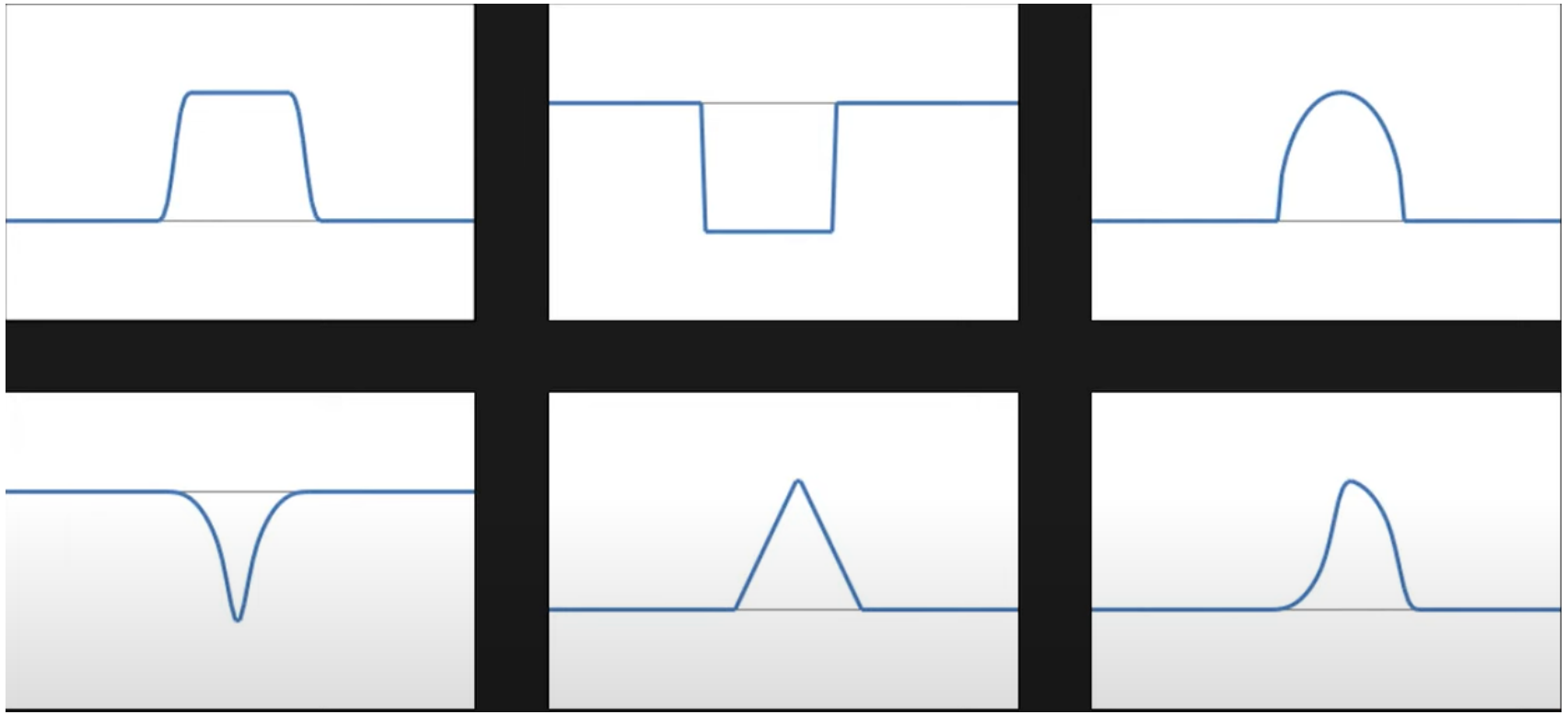
$\nabla^2(n_\sigma)$

$\nabla^2(n_\sigma) * f$



Zero crossing in 2nd derivative of Gaussian denote an Edge

1D Blobs



Examples of 1D Blob-like structures

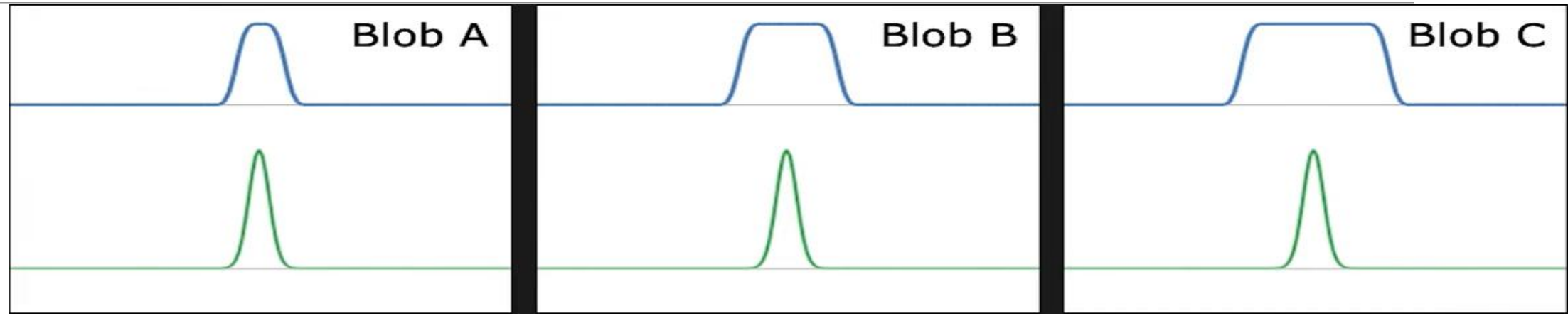
1D Blob and 2nd Derivative of Gaussian

$f(x)$

1D Blob and 2nd Derivative of Gaussian

$f(x)$

n_σ

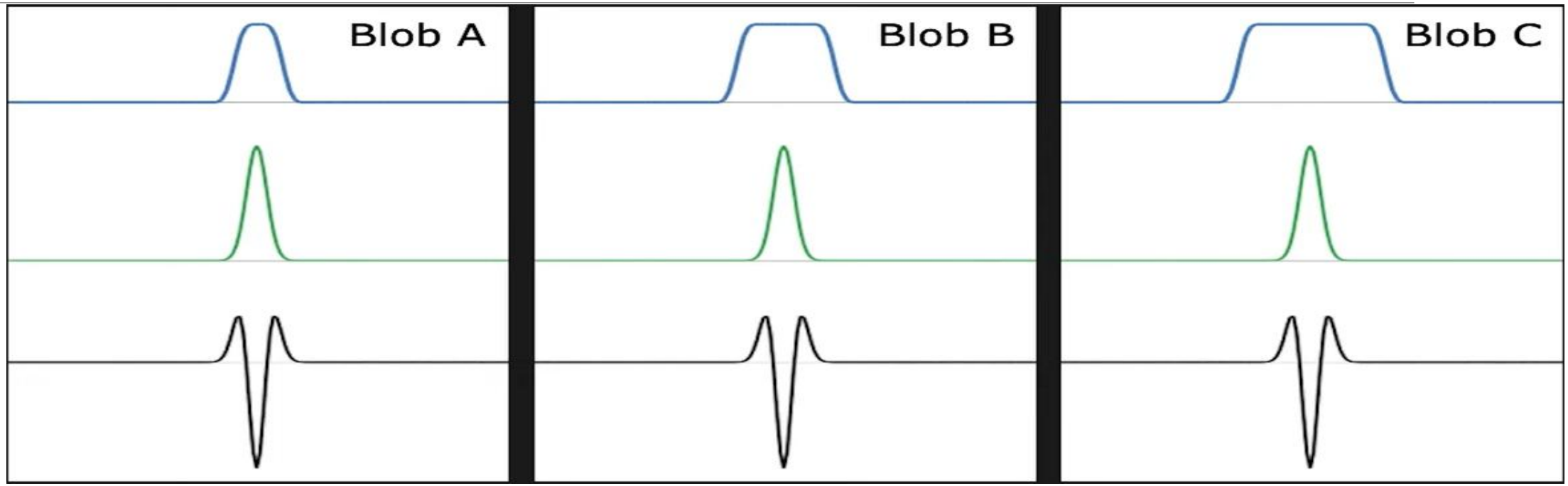


1D Blob and 2nd Derivative of Gaussian

$f(x)$

n_σ

$\frac{\partial^2 n_\sigma}{\partial x^2}$



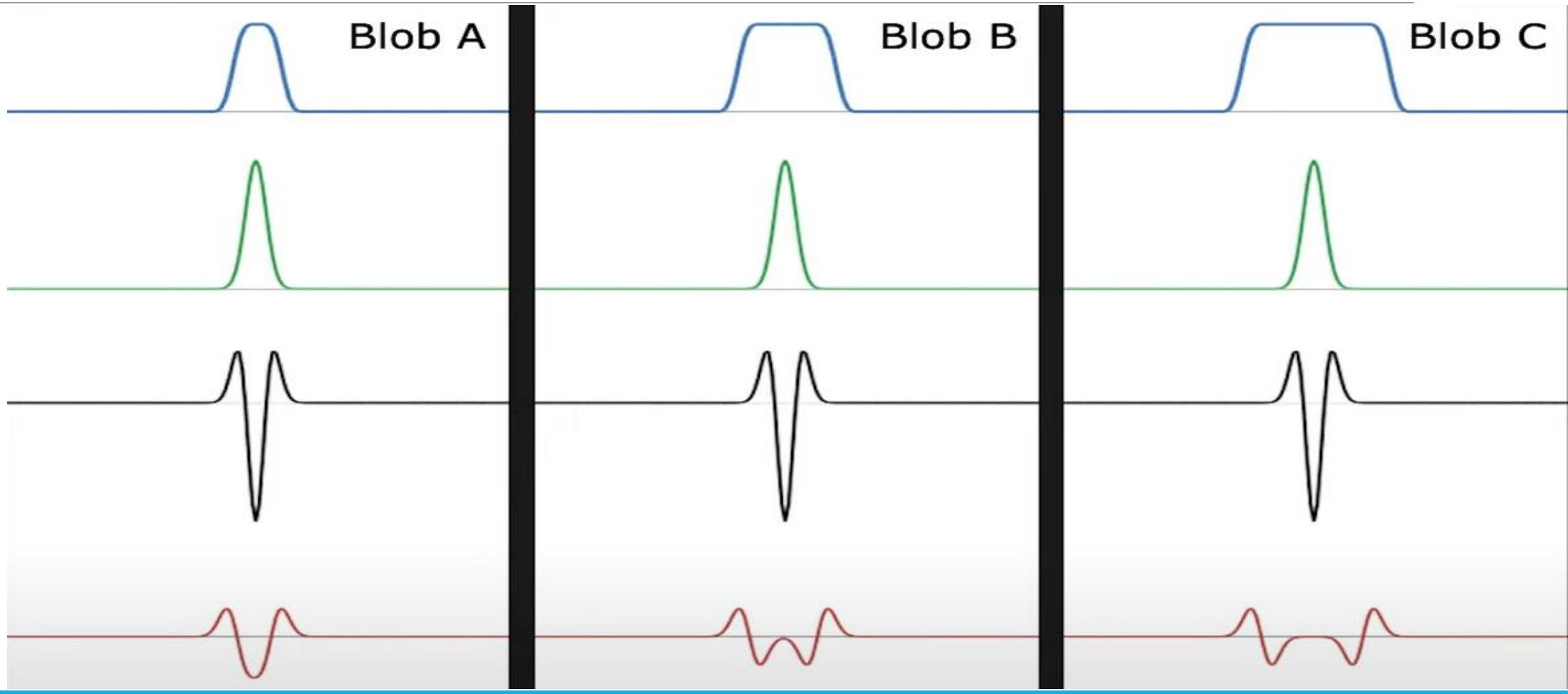
1D Blob and 2nd Derivative of Gaussian

$f(x)$

n_σ

$\frac{\partial^2 n_\sigma}{\partial x^2}$

$\frac{\partial^2 n_\sigma}{\partial x^2} * f(x)$



1D Blob and 2nd Derivative of Gaussian

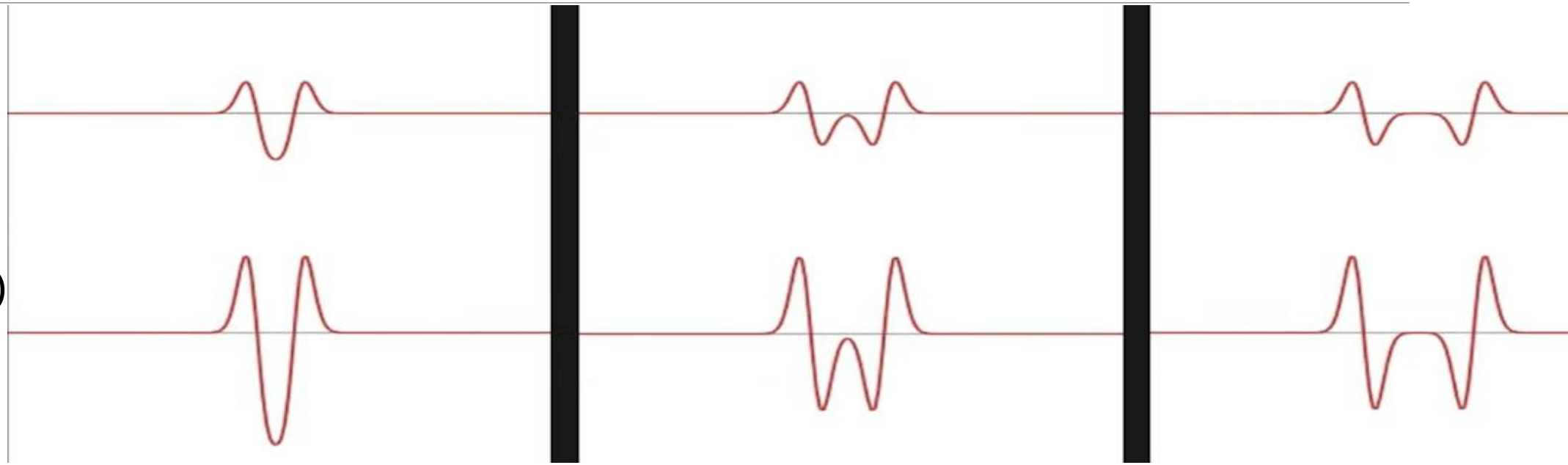
$$\frac{\partial^2 n_\sigma}{\partial x^2} * f(x)$$

- The last results show well separated zero-crossings however the first one and the middle one have some overlapping
- The response of the operator depends on the value of σ
- As σ gets wider, its peak value begins to fall
- Therefore, response of the operator reduces with change in σ
- Changing the value of σ will enable the operator to find blobs at different scales and different resolutions

1D Blob and 2nd Derivative of Gaussian

$\frac{\partial^2 n_\sigma}{\partial x^2} * f(x)$

$\sigma^2 \frac{\partial^2 n_\sigma}{\partial x^2} * f(x)$



- When multiplied by σ^2, we get the σ –normalized output
- So, the scale is now changed for all the three responses

1D Blob and 2nd Derivative of Gaussian

Increasing the value of σ and we are getting more scaling

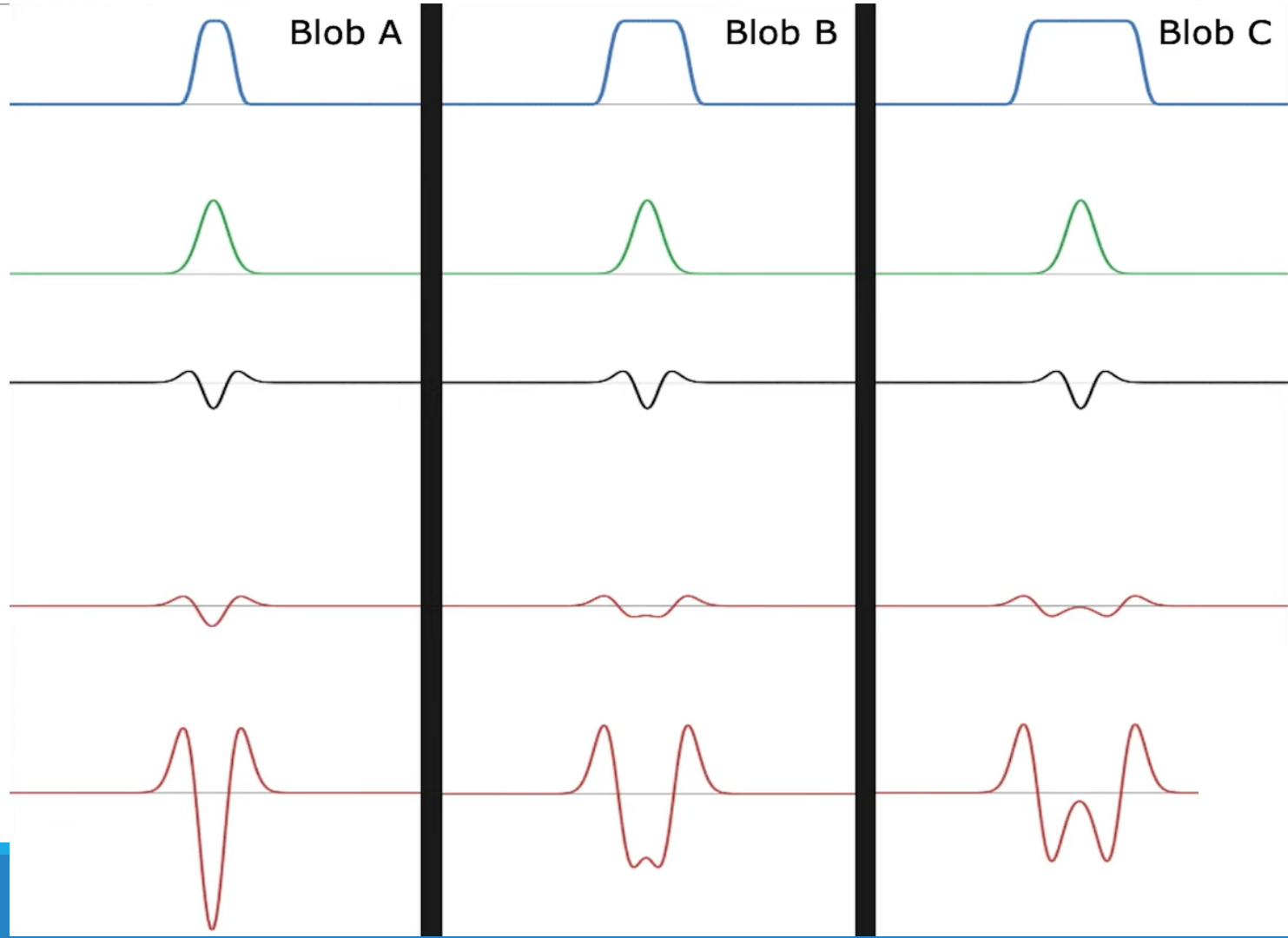
$f(x)$

n_σ

$\frac{\partial^2 n_\sigma}{\partial x^2}$

$\frac{\partial^2 n_\sigma}{\partial x^2} * f(x)$

$\sigma^2 \frac{\partial^2 n_\sigma}{\partial x^2} * f(x)$



1D Blob and 2nd Derivative of Gaussian

Blob A is located due to the peak value after increasing the value of sigma

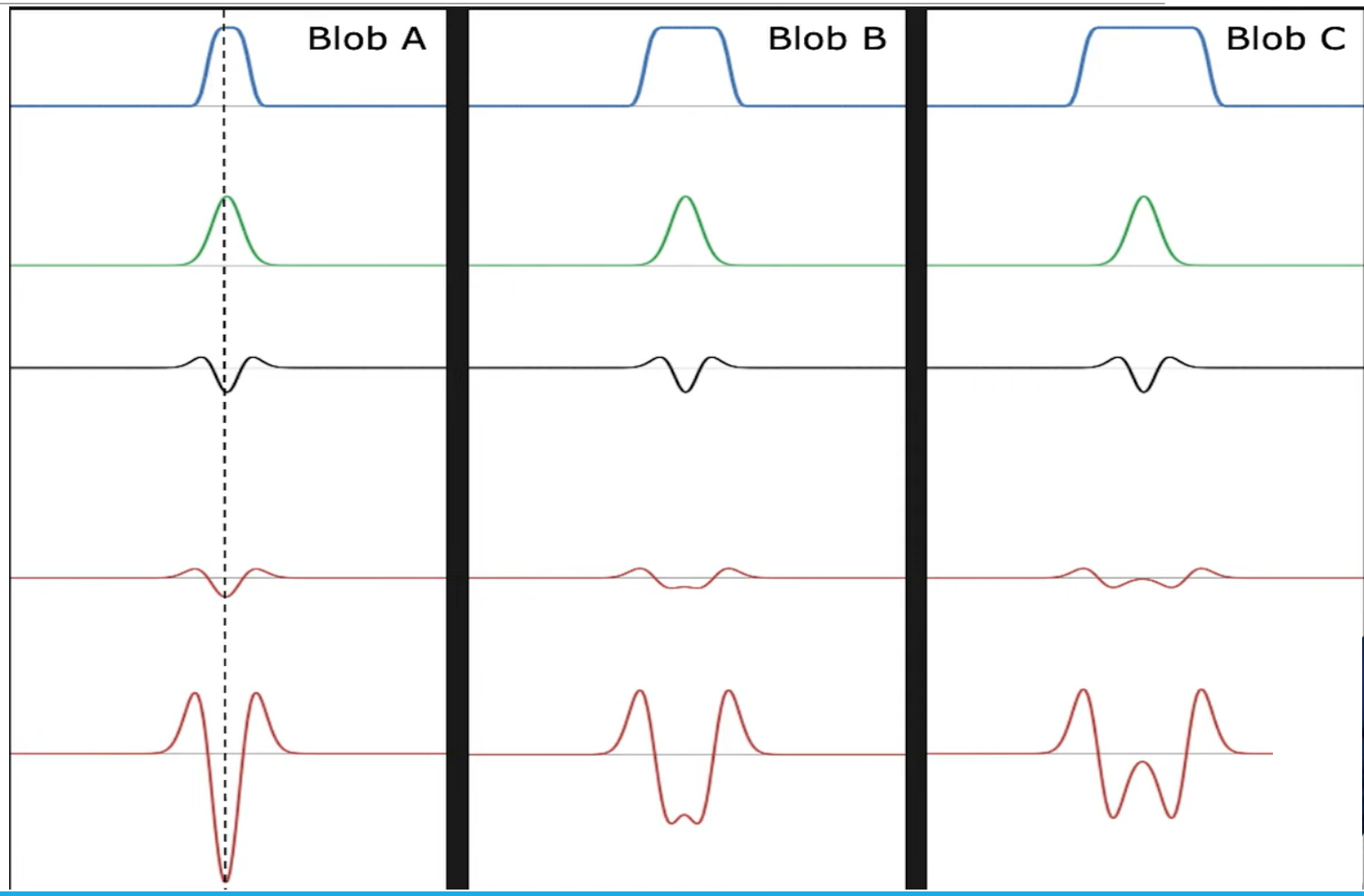
$f(x)$

n_σ

$\frac{\partial^2 n_\sigma}{\partial x^2}$

$\frac{\partial^2 n_\sigma}{\partial x^2} * f(x)$

$\sigma^2 \frac{\partial^2 n_\sigma}{\partial x^2} * f(x)$



1D Blob and 2nd Derivative of Gaussian

We got peak for Blob A, but the other two have not produced the peaks yet

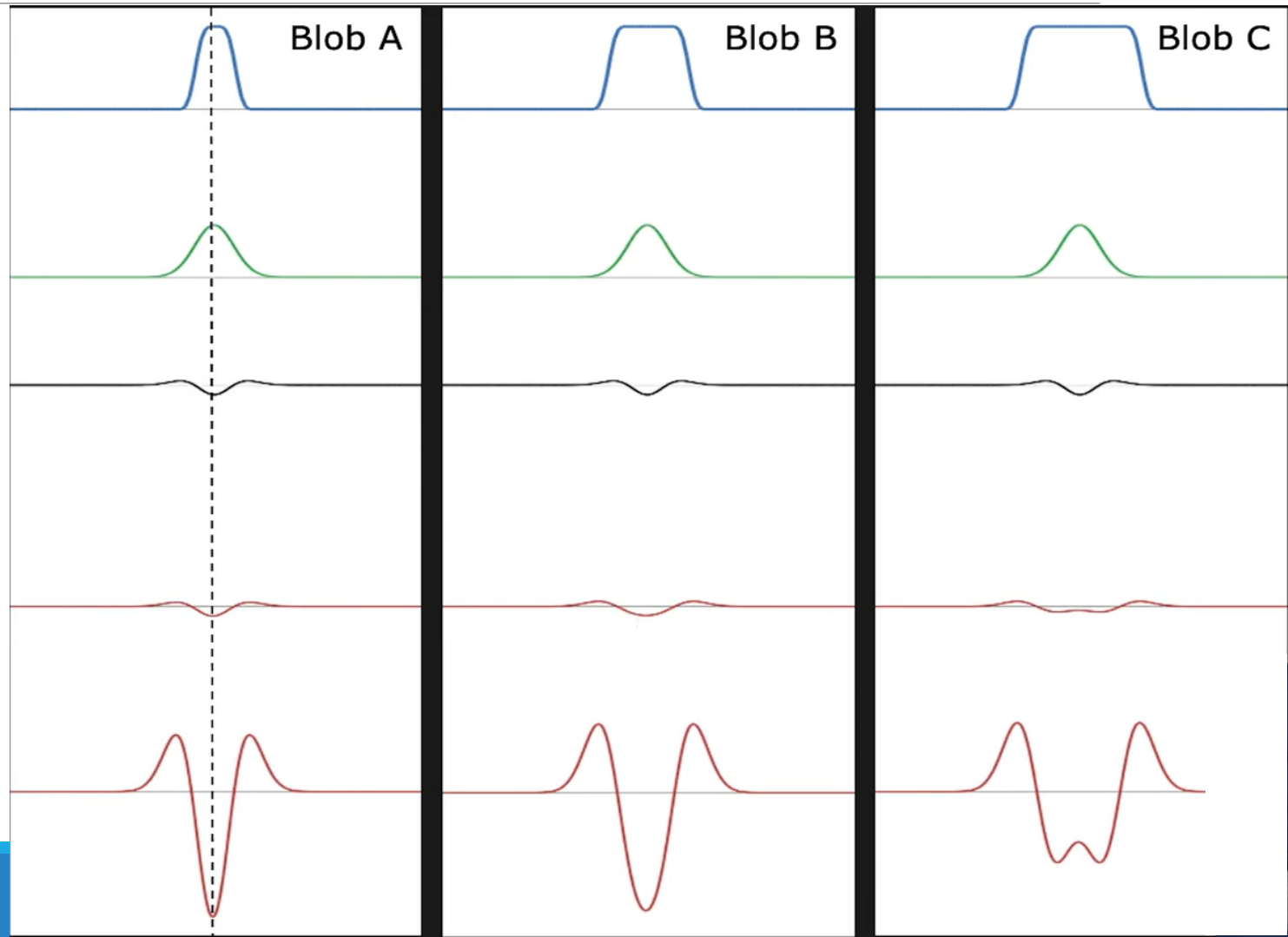
$f(x)$

n_σ

$\frac{\partial^2 n_\sigma}{\partial x^2}$

$\frac{\partial^2 n_\sigma}{\partial x^2} * f(x)$

$\sigma^2 \frac{\partial^2 n_\sigma}{\partial x^2} * f(x)$



1D Blob and 2nd Derivative of Gaussian

Further increasing the σ, we got peak for Blob B, but Blob A peak starts decreasing

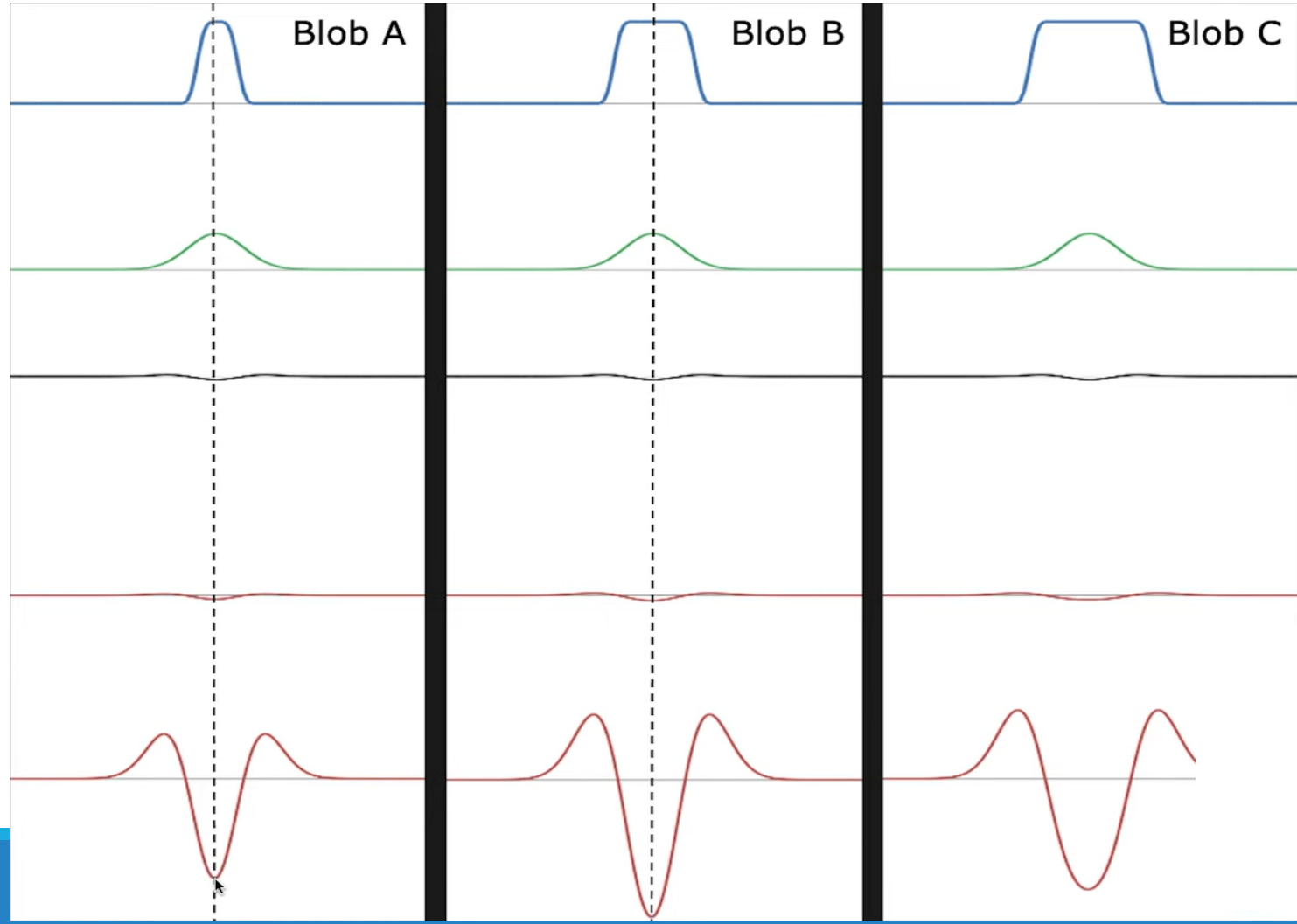
$f(x)$

n_σ

$\frac{\partial^2 n_\sigma}{\partial x^2}$

$\frac{\partial^2 n_\sigma}{\partial x^2} * f(x)$

$\sigma^2 \frac{\partial^2 n_\sigma}{\partial x^2} * f(x)$



1D Blob and 2nd Derivative of Gaussian

Now Blob C is also detecting by further increasing the value of σ

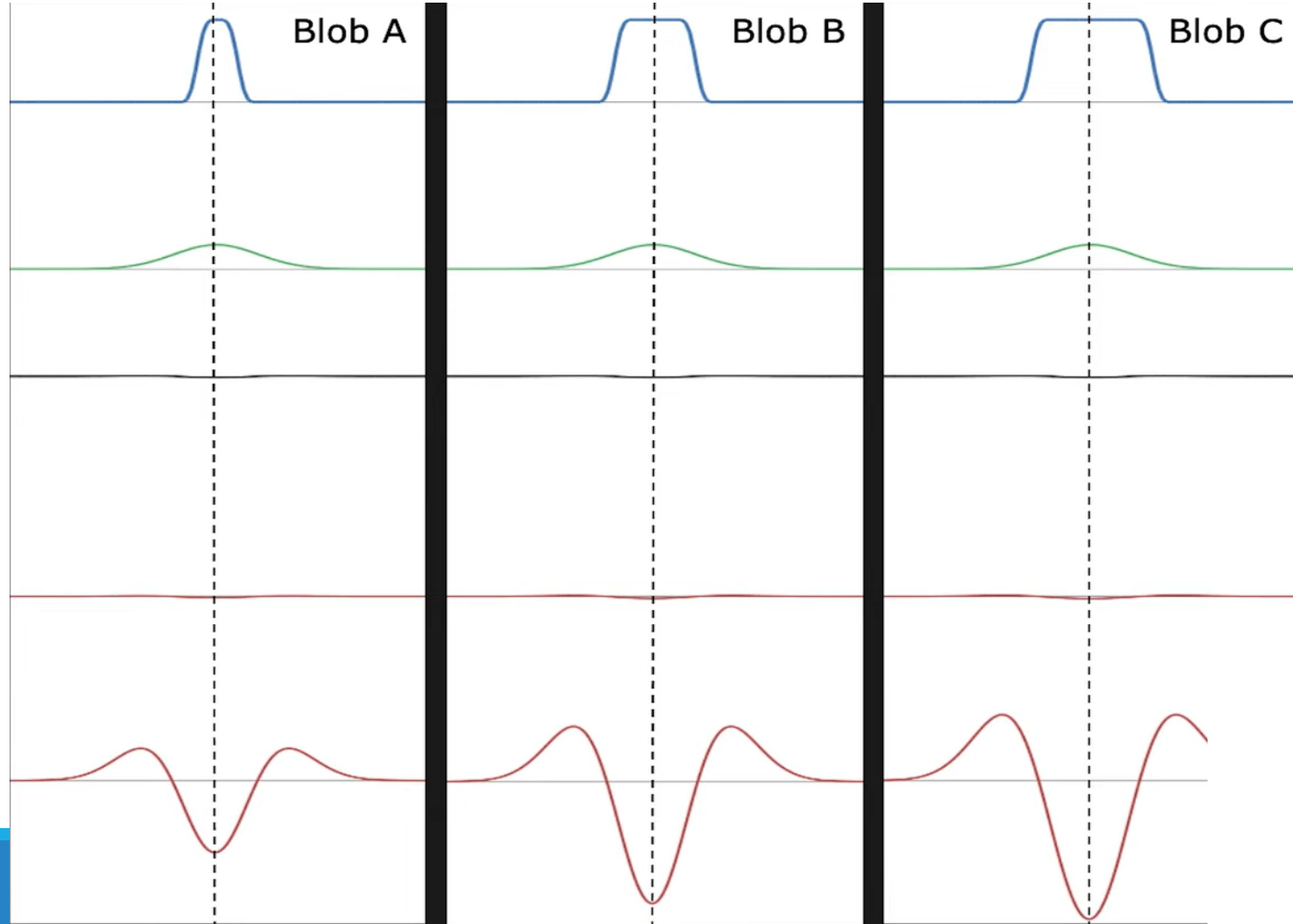
$f(x)$

n_σ

$\frac{\partial^2 n_\sigma}{\partial x^2}$

$\frac{\partial^2 n_\sigma}{\partial x^2} * f(x)$

$\sigma^2 \frac{\partial^2 n_\sigma}{\partial x^2} * f(x)$



1D Blob and 2nd Derivative of Gaussian

Further increasing the value of σ will lead to decreasing peak values

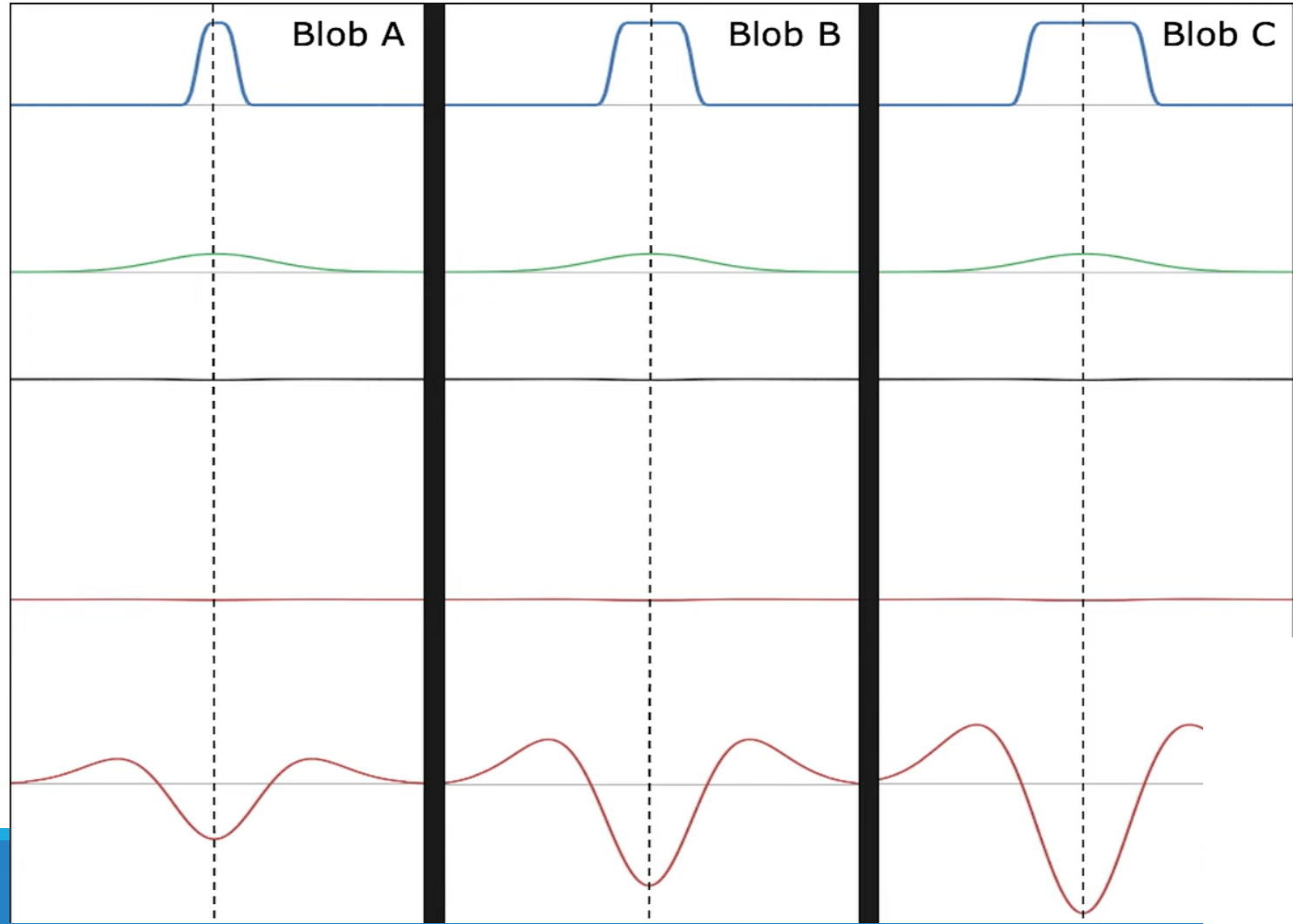
$f(x)$

n_σ

$\frac{\partial^2 n_\sigma}{\partial x^2}$

$\frac{\partial^2 n_\sigma}{\partial x^2} * f(x)$

$\sigma^2 \frac{\partial^2 n_\sigma}{\partial x^2} * f(x)$



1D Blob and 2nd Derivative of Gaussian

- So, for each peak you get a separate Gaussian
- Notice that the width of each Gaussian is proportional to the Blob

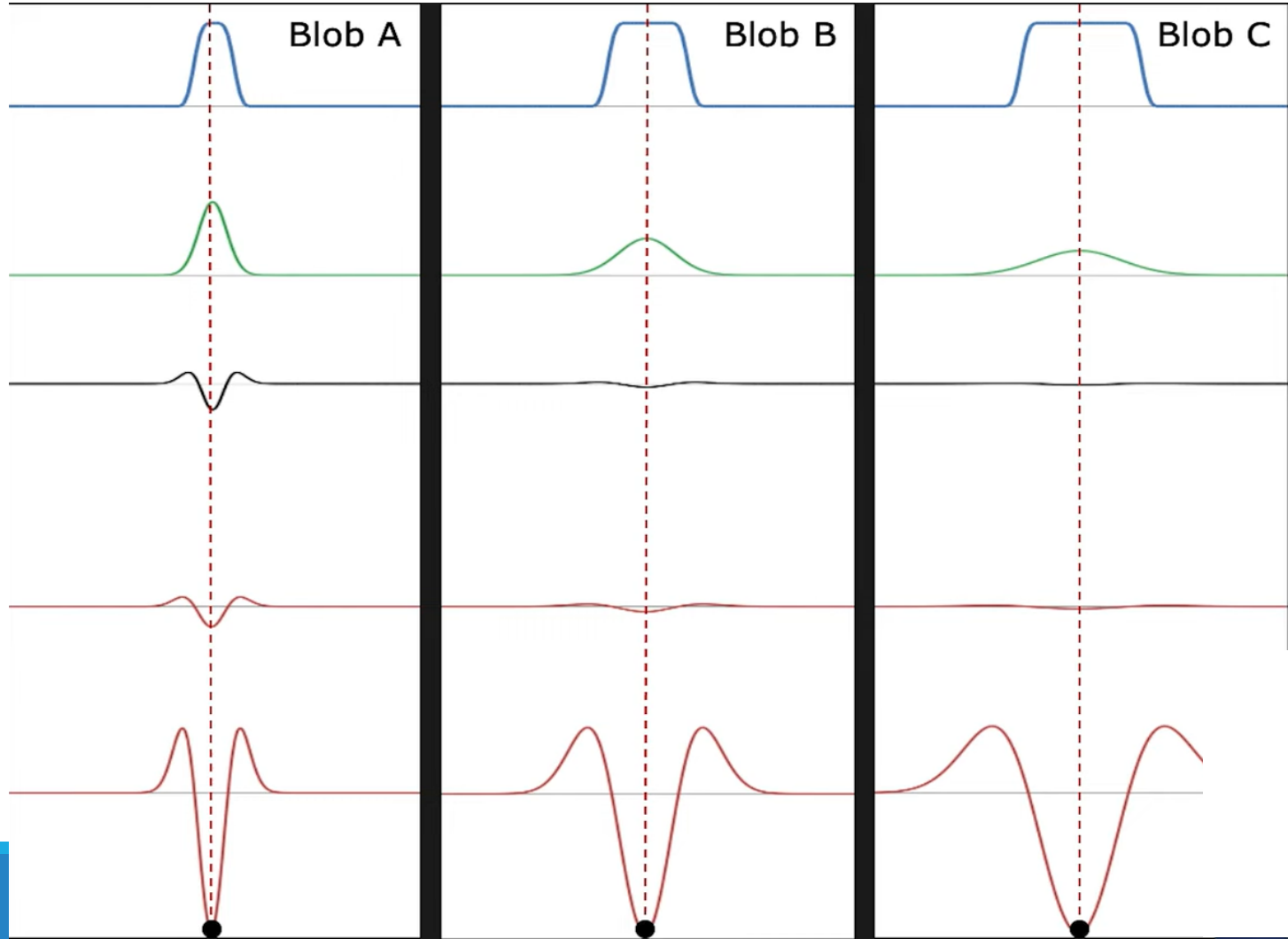
$f(x)$

n_σ

$\frac{\partial^2 n_\sigma}{\partial x^2}$

$\frac{\partial^2 n_\sigma}{\partial x^2} * f(x)$

$\sigma^2 \frac{\partial^2 n_\sigma}{\partial x^2} * f(x)$



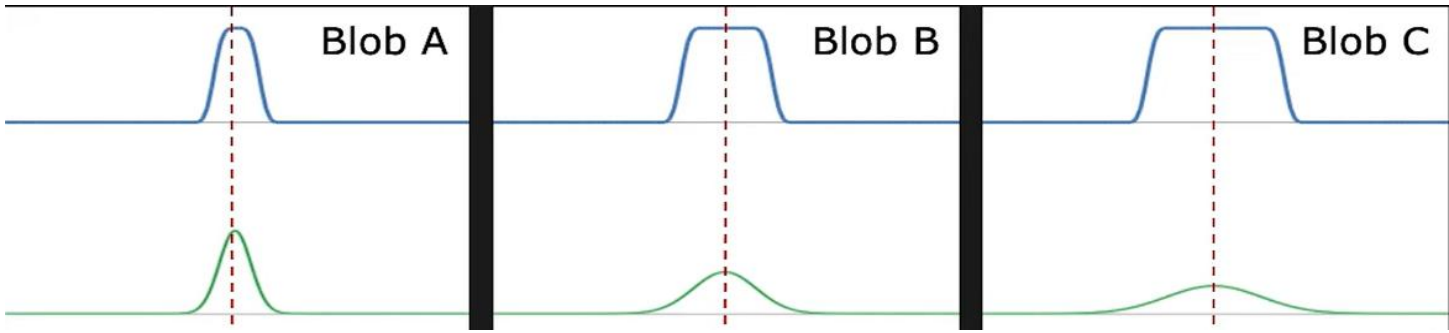
1D Blob and 2nd Derivative of Gaussian

- You have now the stack of the outputs of $\sigma^2 \frac{\partial^2 n_\sigma}{\partial x^2} * f(x)$ for various values of σ
- This stack is described by two parameters that are x and σ
- Local Extremum in $(x, \sigma) - space$ Represent Blobs
- Values of x correspond to the location of the Blobs and σ shows the scale of the Blob

Characteristic Scale and Blob Size

$f(x)$

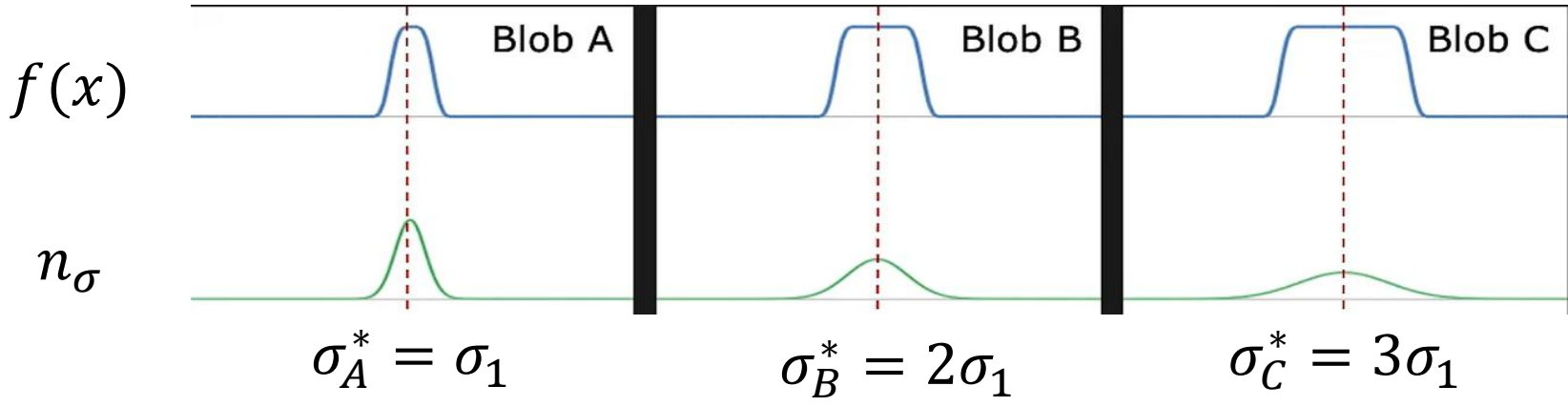
n_σ



Characteristic Scale: The σ at which $\sigma - normalized$ 2nd derivative attains its extreme value

Characteristic Scale $\propto$ Size of Blob

Characteristic Scale and Blob Size



- Characteristic Scale: The σ at which $\sigma - normalized$ 2nd derivative attains its extreme value
- Characteristic Scale is very important to Blob detection

Characteristic Scale $\propto$ Size of Blob

Recap: 1D Blob Detection

Given: 1D signal $f(x)$

Compute: $\sigma^2 \frac{\partial^2 n_\sigma}{\partial x^2} * f(x)$ at many scales $(\sigma_0, \sigma_1, \sigma_2, \dots, \sigma_k)$

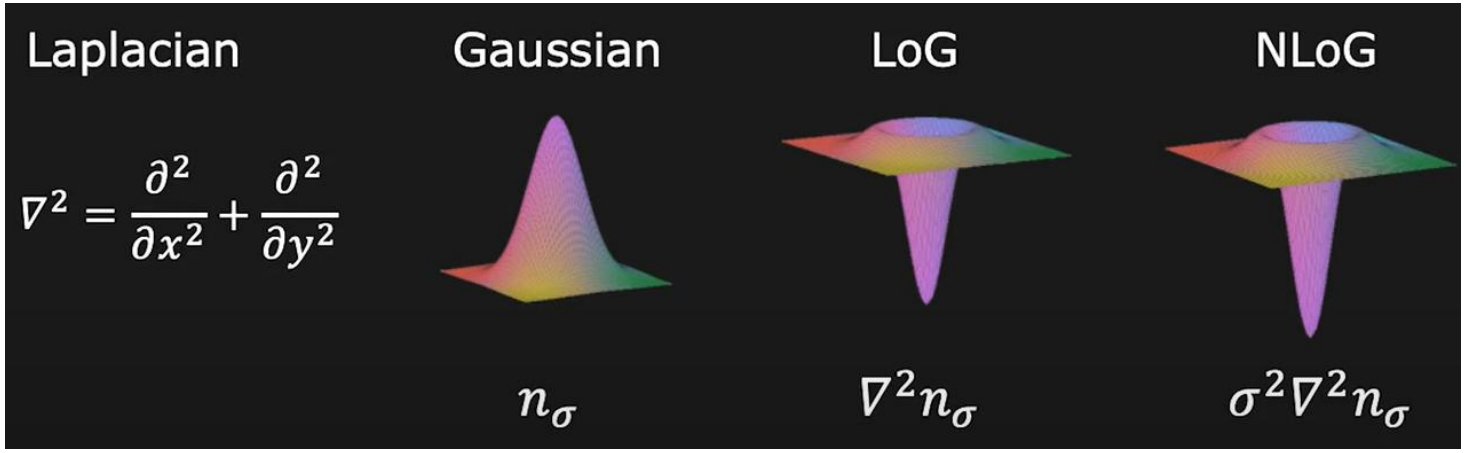
Find: $(x^*, \sigma^*) = \arg\max_{(x,\sigma)} \left| \sigma^2 \frac{\partial^2 n_\sigma}{\partial x^2} * f(x) \right|$

x^*= Blob Position

σ^*= Characteristic Scale (Blob Size)

2D Blob Detector

Normalized Laplacian of Gaussian (NLoG) is used as the 2D equivalent for Blob Detection



Location of Blobs given by **Local Extrema** after applying Normalized Laplacian of Gaussian at many scales

Scale-Space

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

Increasing σ, Higher Scale, Lower Resolution

Scale Space: Stack created by filtering an image with Gaussians of different sigma (σ)

$$S(x, y, \sigma) = n(x, y, \sigma) * I(x, y)$$

Scale-Space

- What sigma (σ) should be used

- Selecting sigma (σ) to generate the scale-space:

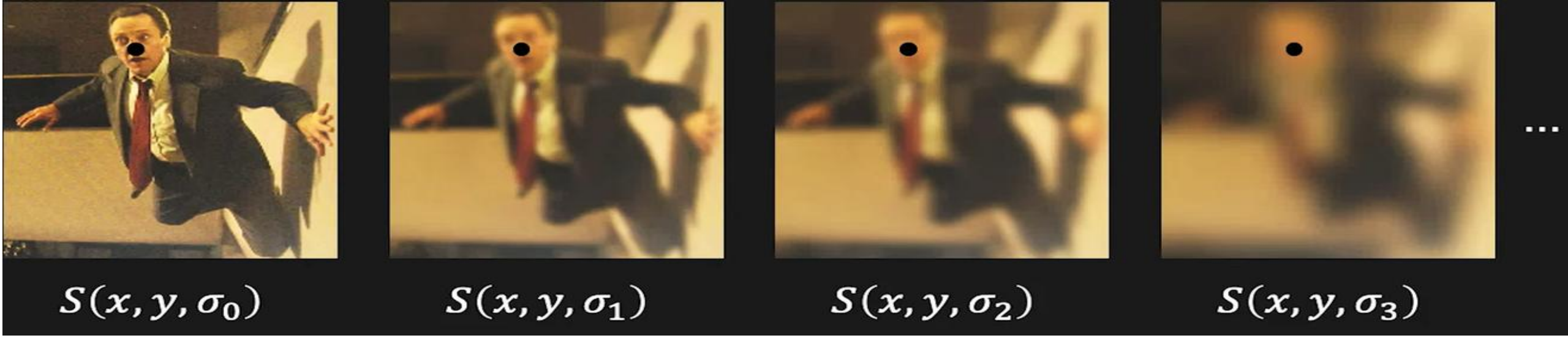
$$\sigma_k = \sigma_0 s^k \quad k = 0,1,2,3, \ldots$$

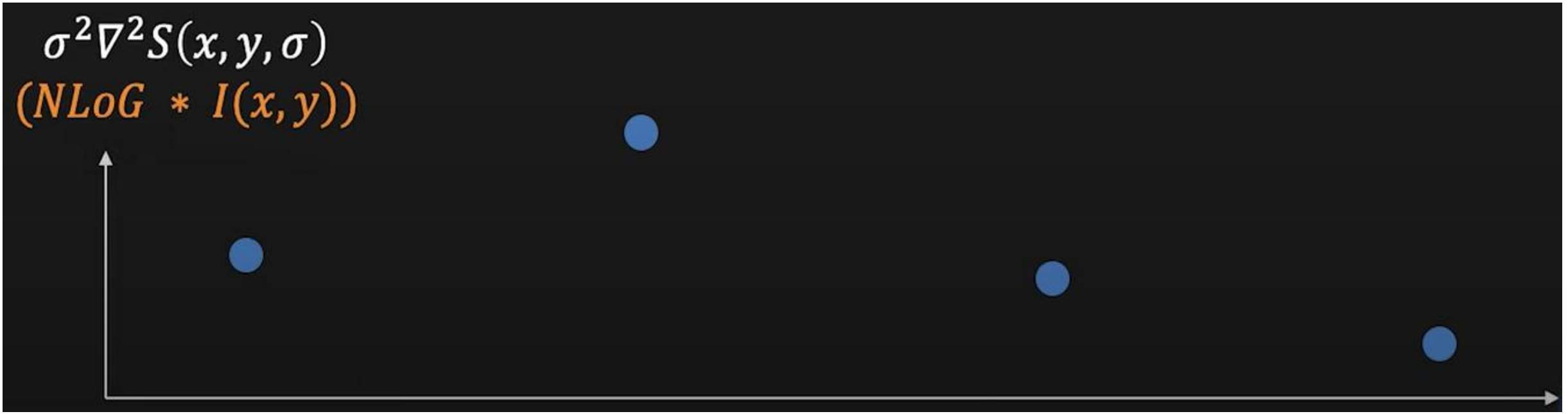
s: Constant multiplier

σ_0: Initial Scale

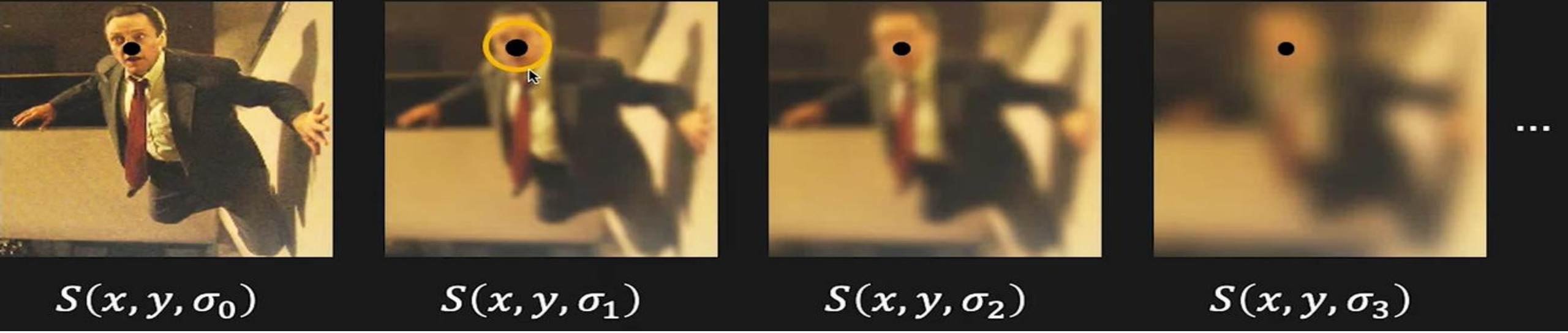
Blob Detection using Local Extrema

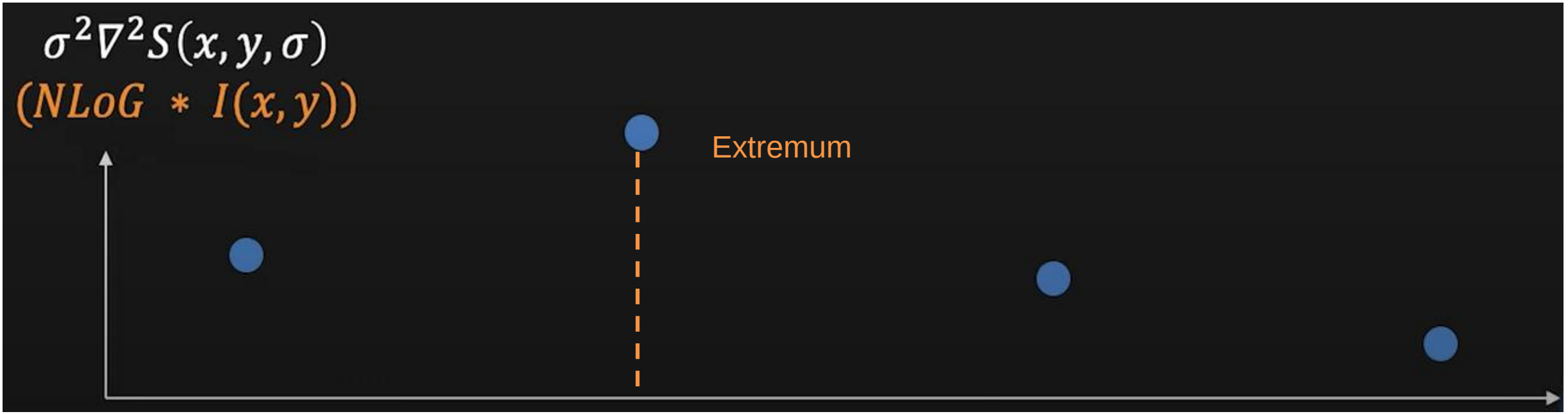
$S(x, y, \sigma_0)$ $S(x, y, \sigma_1)$ $S(x, y, \sigma_2)$ $S(x, y, \sigma_3)$



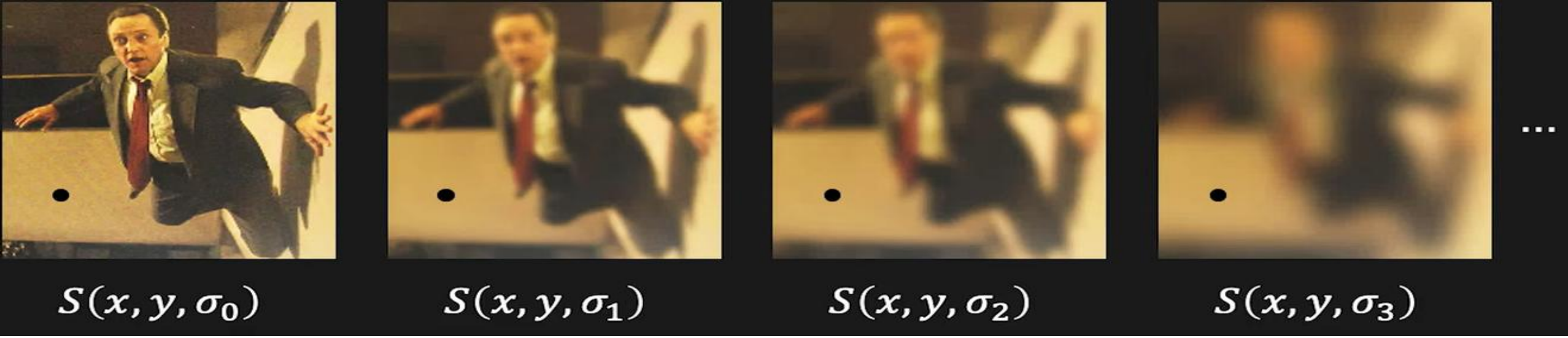


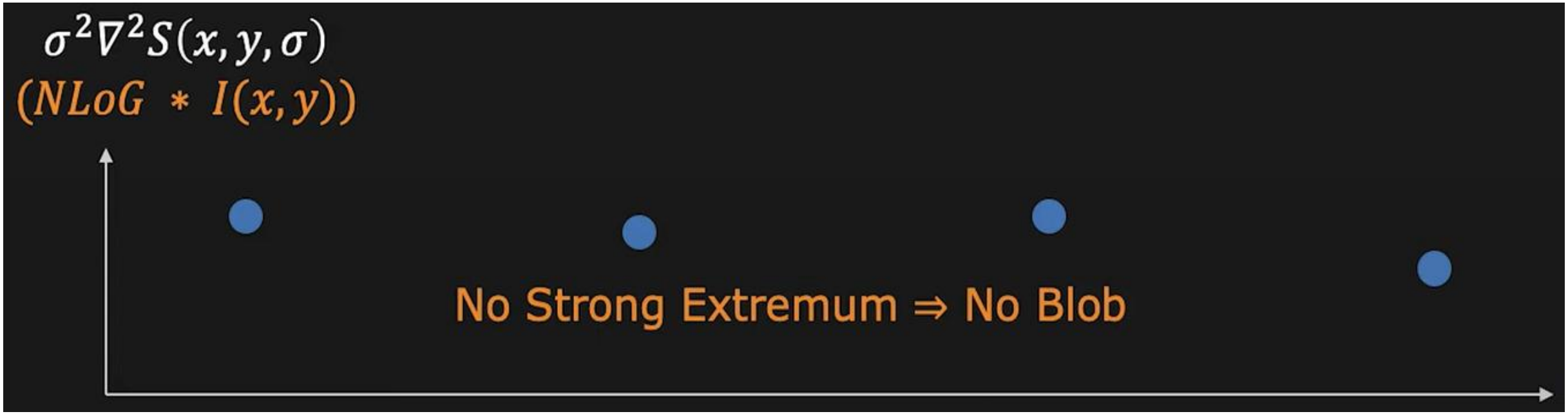
σ_0 σ_1 σ_2 σ_3 $Scale$





Characteristic Scale (σ^*)


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


$\sigma^2 \nabla^2 S(x, y, \sigma)$
$(NLoG * I(x, y))$
No Strong Extremum ⇒ No Blob
σ_0
σ_1
σ_2
σ_3
$Scale$

Recap: 2D Blob Detection

Given an image $I(x, y)$

Convolve the image using NLoG at many scales σ

Find:

$$(x^*, y^*, \sigma^*) = \arg \max_{(x,y,\sigma)} |\sigma^2 \nabla^2 n_\sigma * I(x, y)|$$

(x^*, y^*): Position of the Blob

σ^*: Size of the Blob

SIFT Detector

SIFT Detector

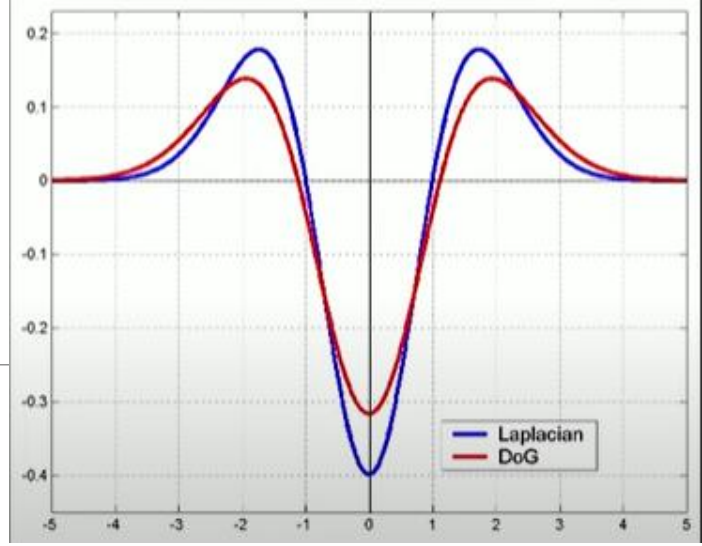
- Now we have all the tools that we need to develop the SIFT detector
- Proposed by David Lowe and widely used in the field of Computer Vision
- SIFT uses some tricks to be reliable and efficient

Fast NLoG Approximation: DoG

- The **Difference of Gaussian (DoG)** is used as a fast approximation for the **Normalized Laplacian of Gaussian (NLoG)** in computer vision.
- DoG is obtained by subtracting two Gaussian-blurred versions of the same image:

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

- where G is a Gaussian filter with different standard deviations (σ). It highlights regions where intensity changes rapidly → edges, blobs, keypoints.
- This shows that **DoG is approximately proportional to LoG**, but computationally cheaper to calculate.

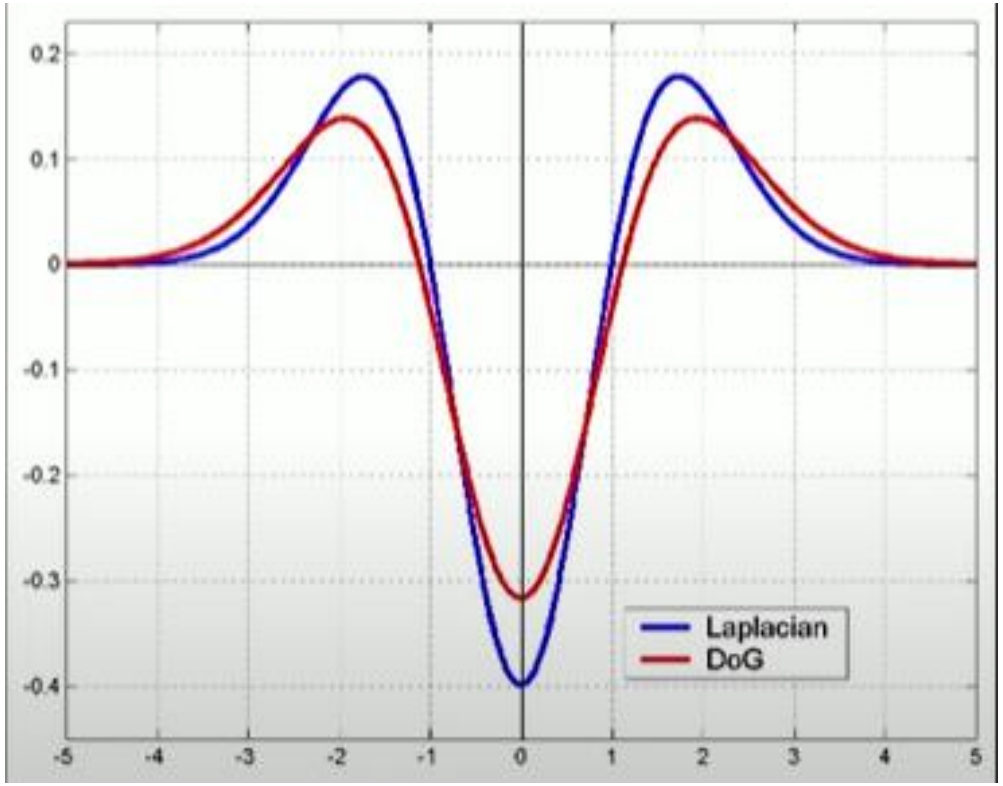


The Laplacian of Gaussian (LoG) $\nabla^2 n_{\sigma}$

Fast NLoG Approximation: DoG

$$\text{Difference of Gaussian } (DoG) = (n_{s\sigma} - n_{\sigma}) \approx (s-1)\sigma^2\nabla^2 n_{\sigma}$$

NLoG



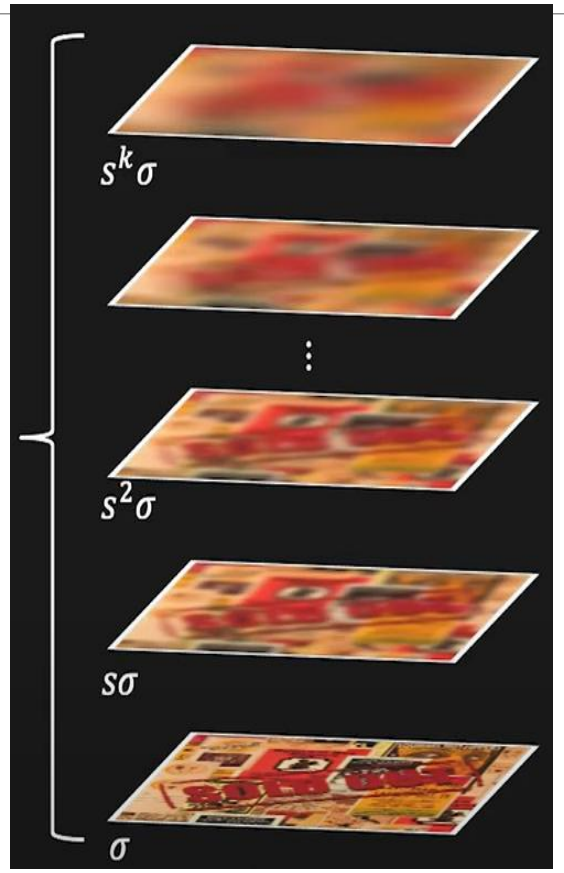
- LoG requires computing second derivatives of Gaussians → computationally expensive.
- DoG only requires two Gaussian smoothings and a subtraction, which is much faster.
- This is why algorithms like SIFT (Scale-Invariant Feature Transform) use DoG to detect scale-space keypoints.

$$DoG \approx (s-1)NLoG$$

Extracting SIFT Interest Points

Extracting SIFT Interest Points

$Image\ I(x, y)$



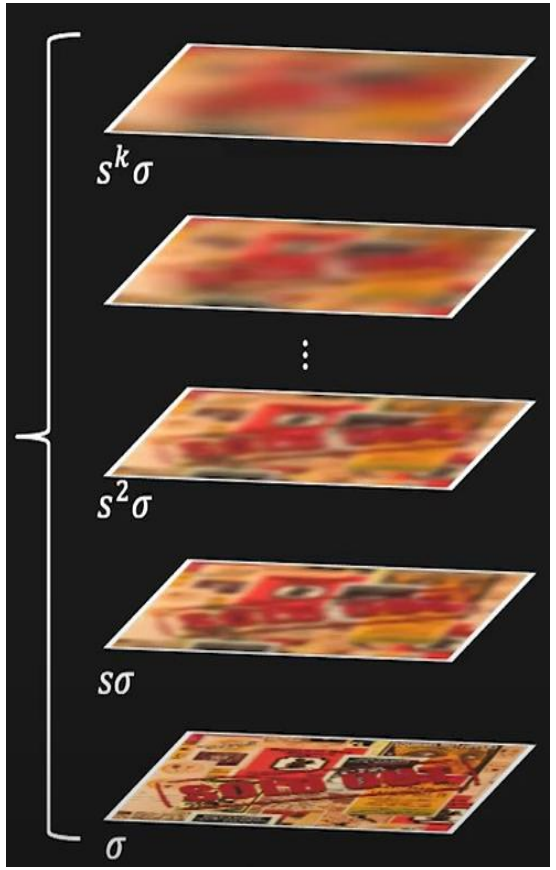
Gaussian Scale-Space
$S(x, y, \sigma)$

Step 1: Build Difference of Gaussian (DoG) images

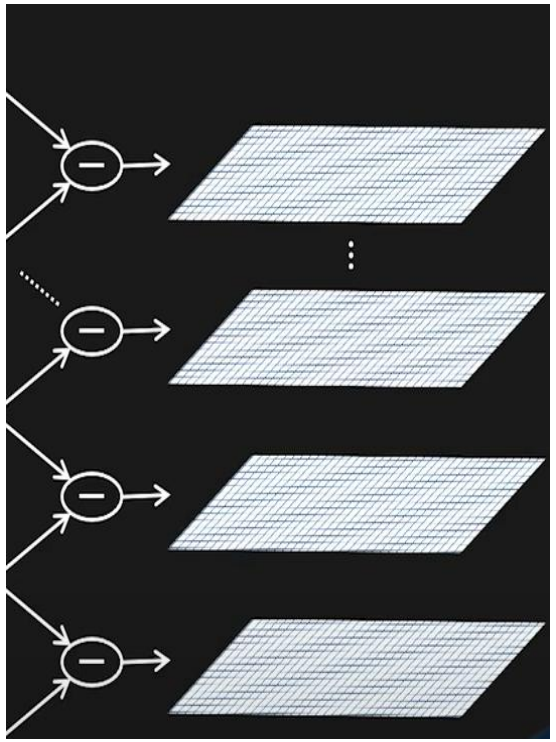
Multiple versions of the image are created by blurring with Gaussians of increasing scale (σ).

Extracting SIFT Interest Points

$Image\ I(x,y)$



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



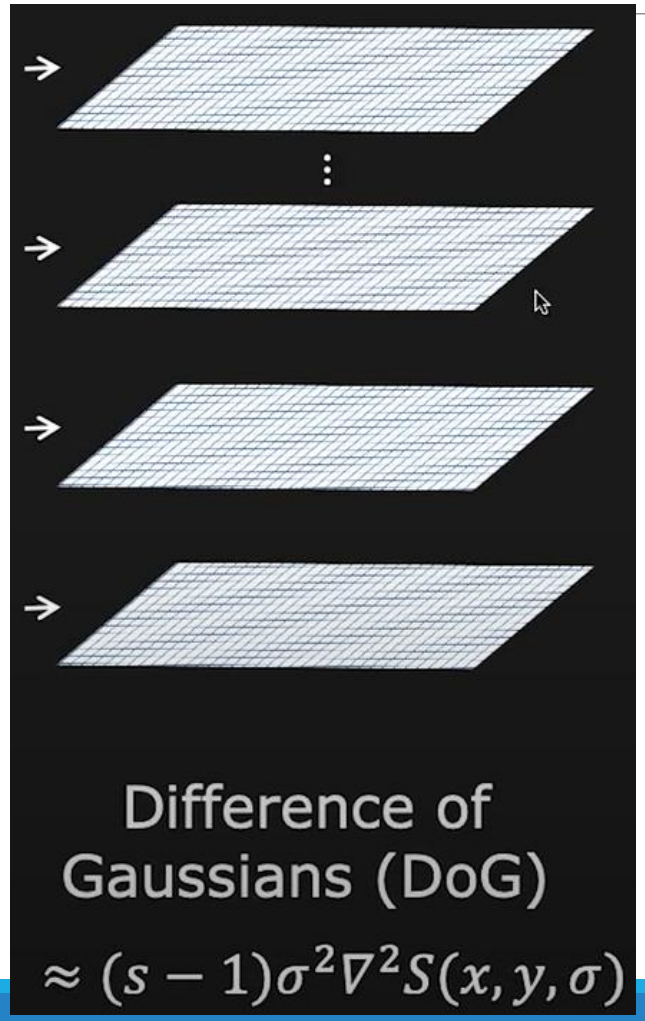
Difference of Gaussian (DoG)

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

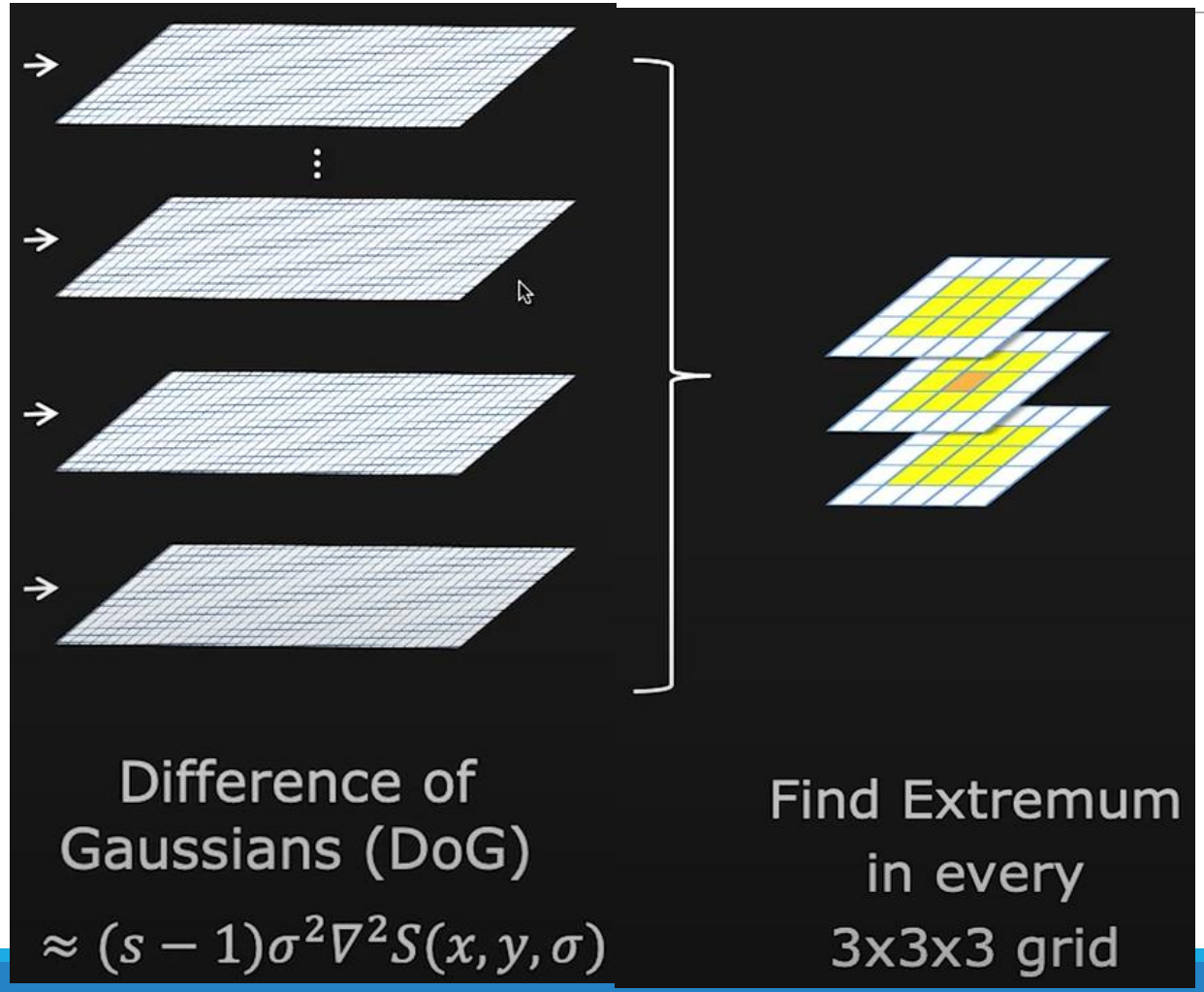
The **Difference of Gaussian (DoG)** is computed between successive blurred images.

This gives a set of DoG images across different scales (shown as stacked layers on the left)

Extracting SIFT Interest Points



Extracting SIFT Interest Points



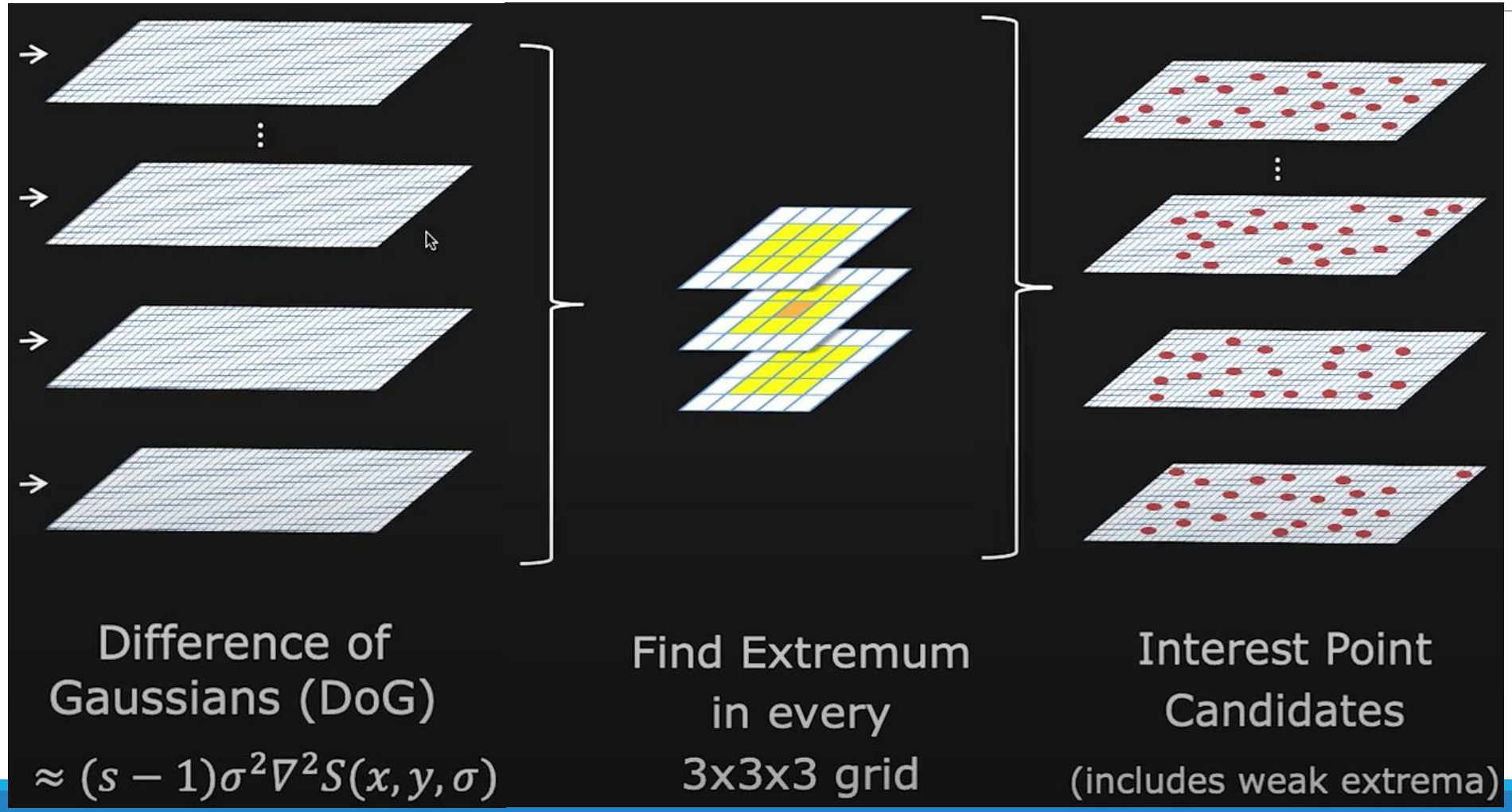
Step 2: Find extrema in scale-space

For each pixel in a DoG image, SIFT checks its value against **26 neighbors**:

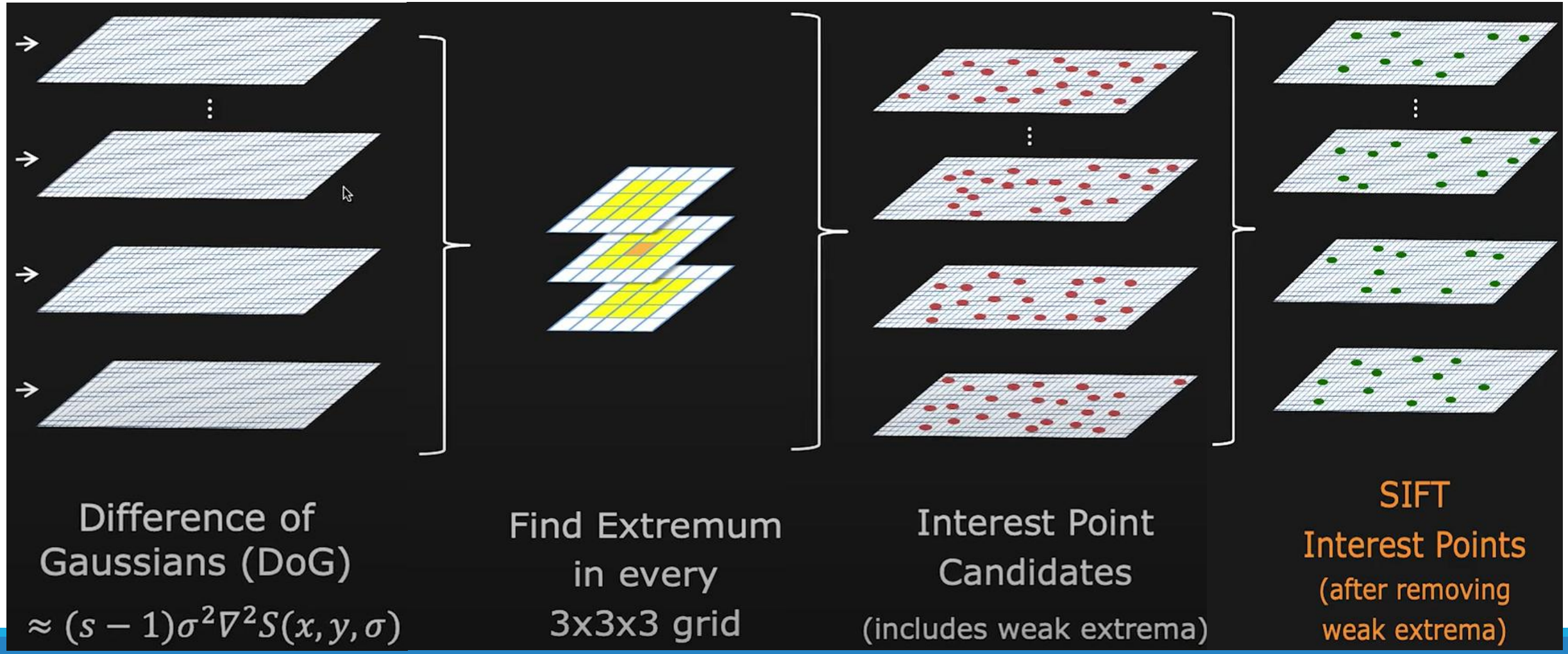
- 8 neighbors in the same 2D image,
- 9 in the scale above,
- 9 in the scale below.

This forms a **3 × 3 × 3 neighborhood** in scale-space (shown as the 3D yellow cubes on the right). If the pixel is a **local maximum or minimum** compared to all 26 neighbors, it is selected as a **keypoint candidate**.

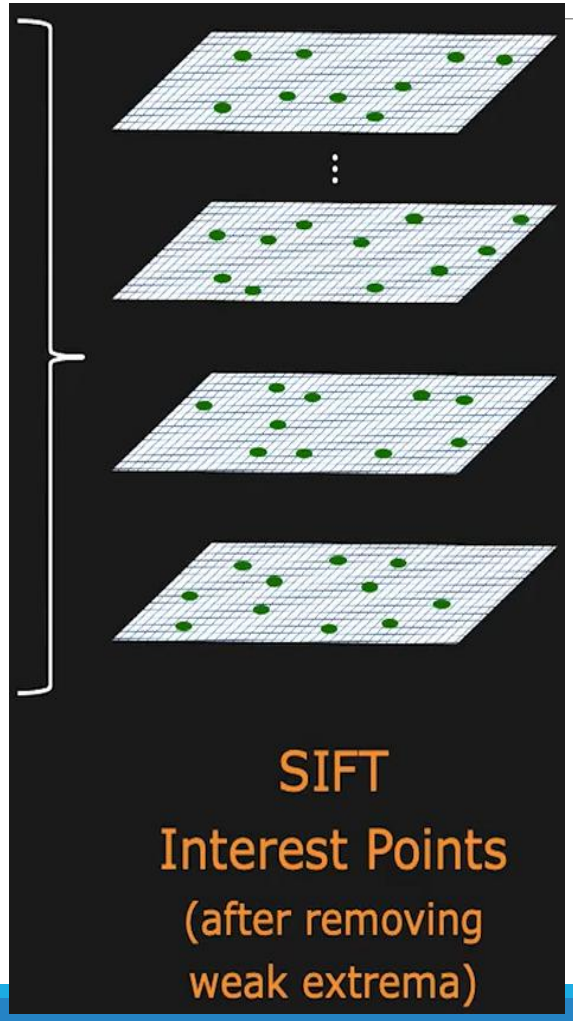
Extracting SIFT Interest Points

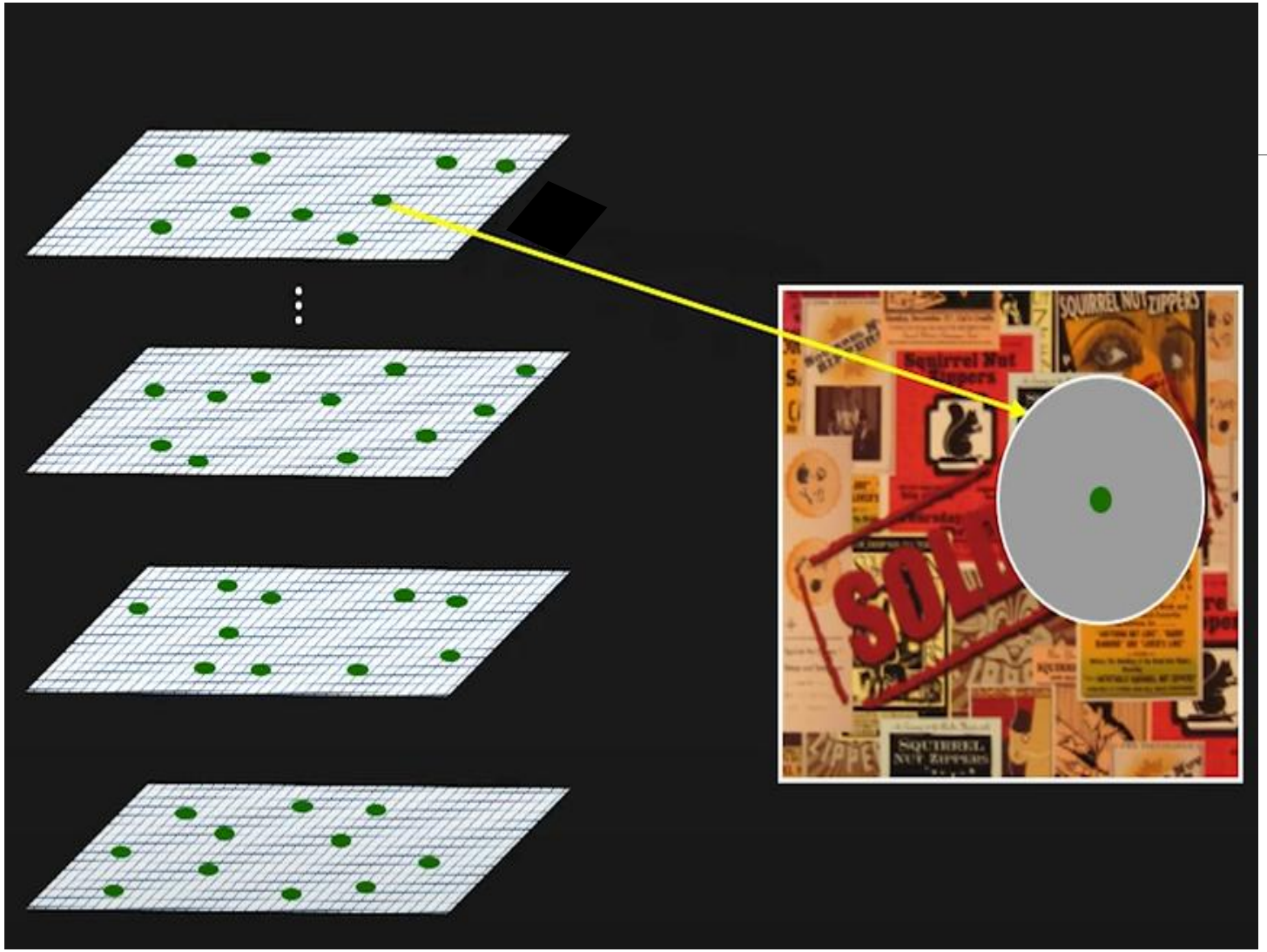


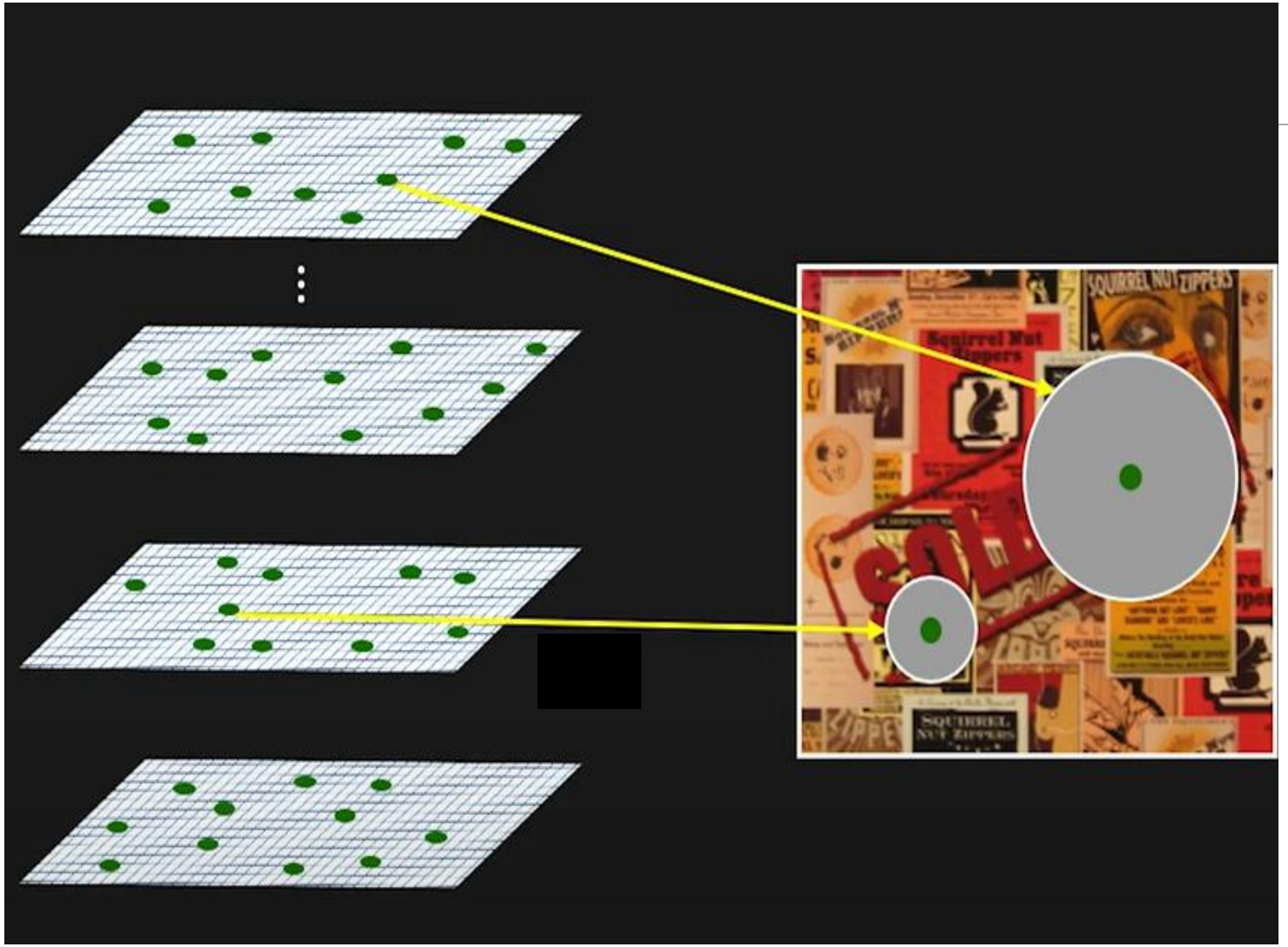
Extracting SIFT Interest Points

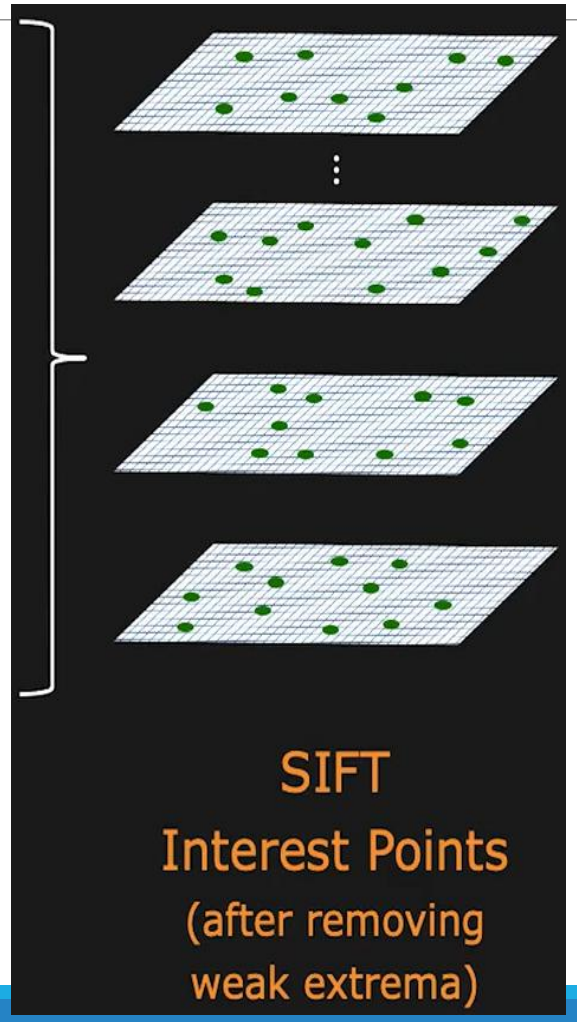


Extracting SIFT Interest Points




SQUIRREL NUT ZIPPERS
Squirrel Nut Zippers
SOLD
SQUIRREL NUT ZIPPERS


SQUIRREL NUT ZIPPERS
Squirrel Nut Zippers
SQUIRREL NUT ZIPPERS



Interest Point Visualization

SIFT Detection Examples

SIFT Detection Examples


PlayStation.
GOD
OUT
DREW BARRYMORE
BETH GRANT
HOLMES OSBORNE
PATRICK SWAYZE
NOAH WYLE
DARKO
DIRECTOR'S CUT

SIFT Scale Invariance

SIFT Scale Invariance

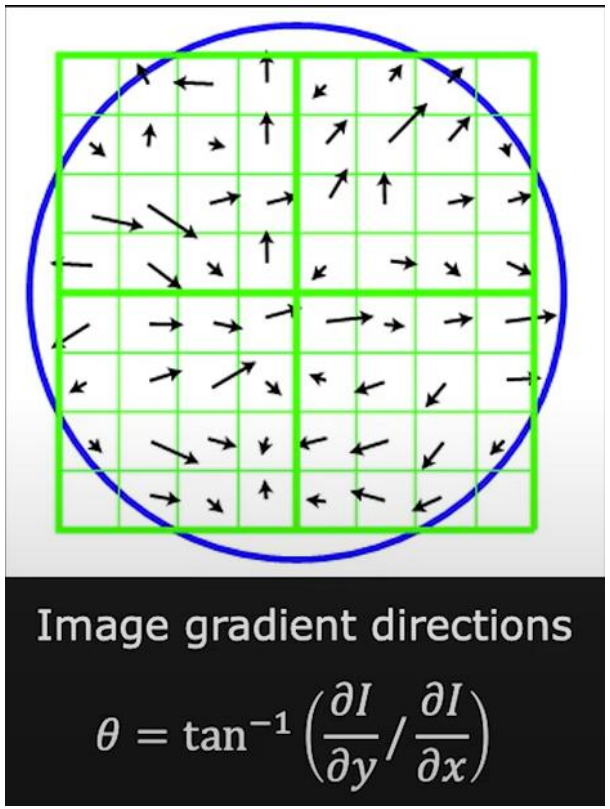
SIFT Scale Invariance

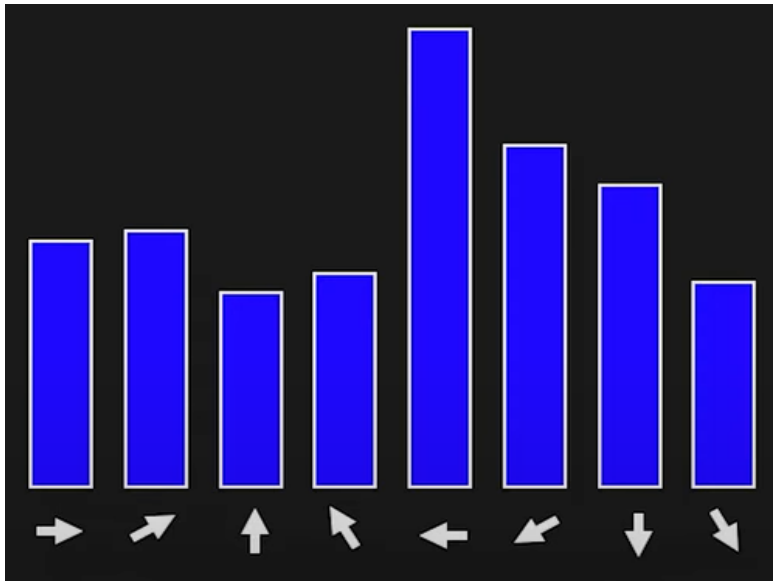
SIFT Scale Invariance

$\frac{\sigma_1^*}{\sigma_2^*}$: Ratio of Blob Sizes

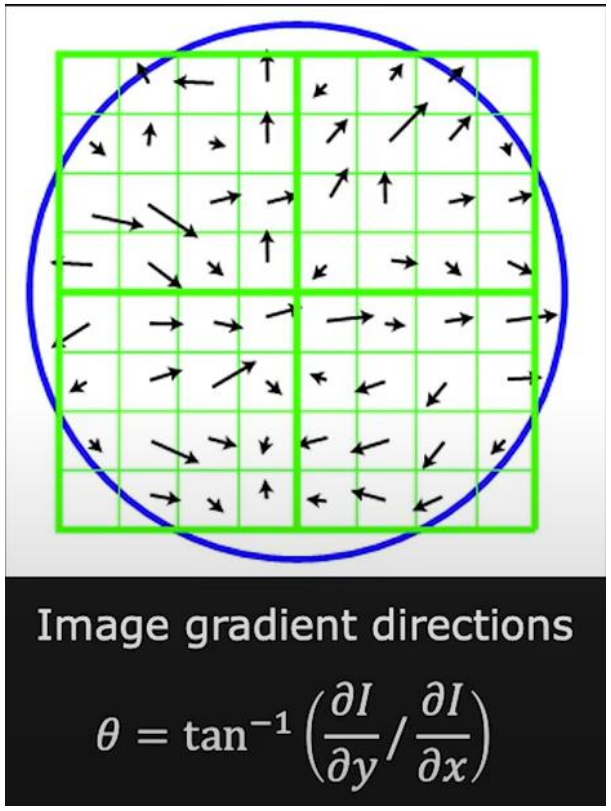
Computing the Principal Orientation

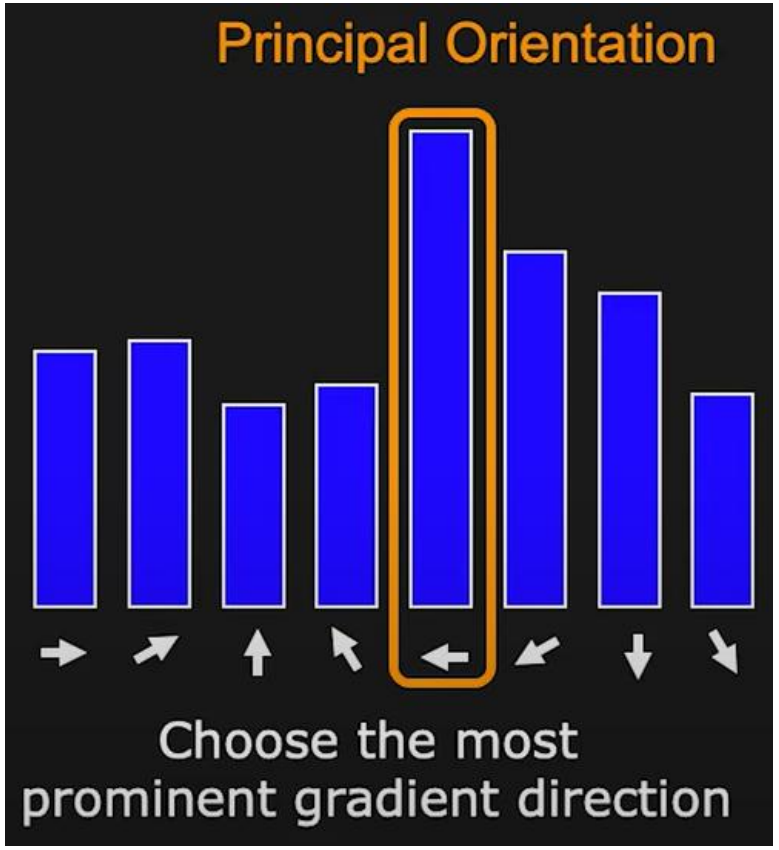
Use the histogram of gradient orientations





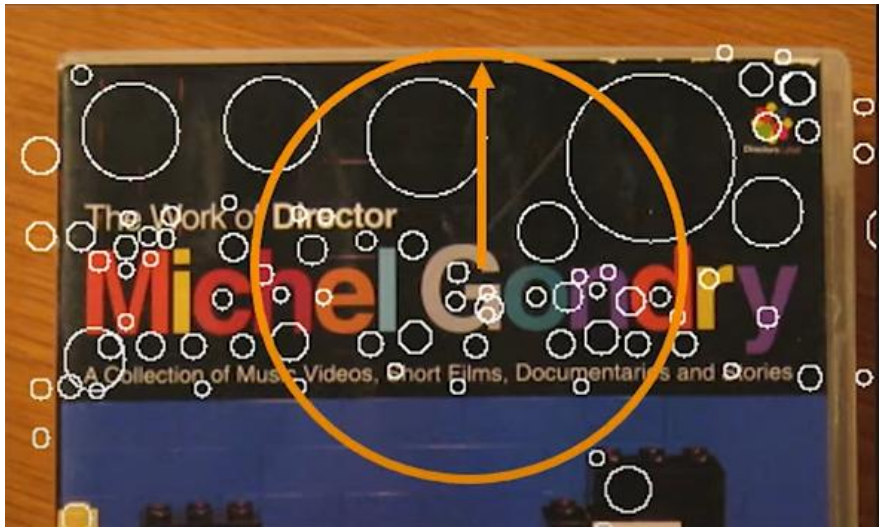
Computing the Principal Orientation

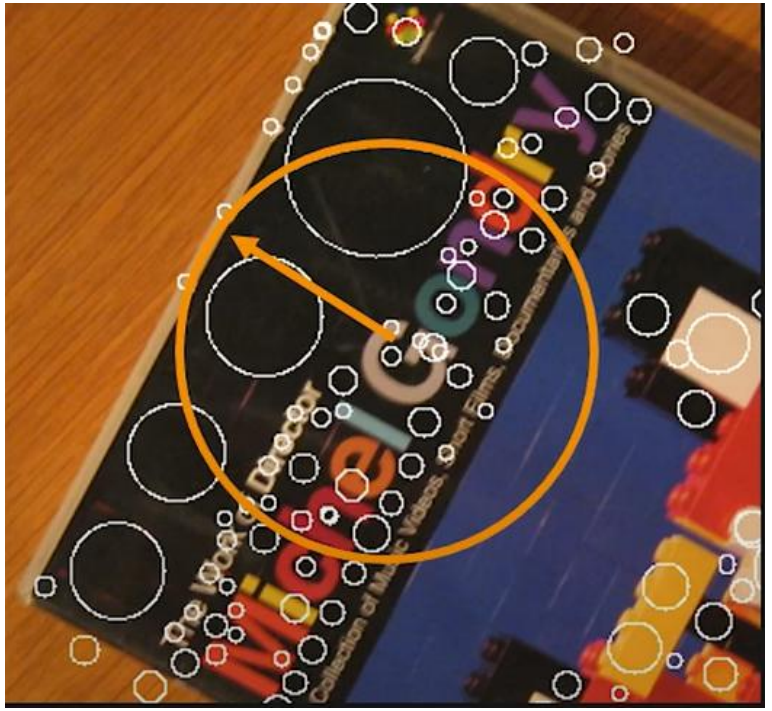




SIFT Rotation Invariance

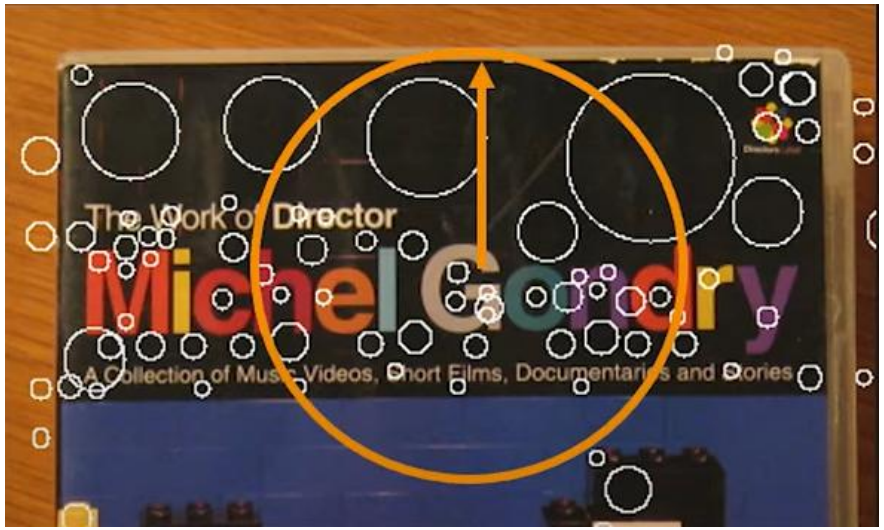
Use the Principal orientation to undo rotation

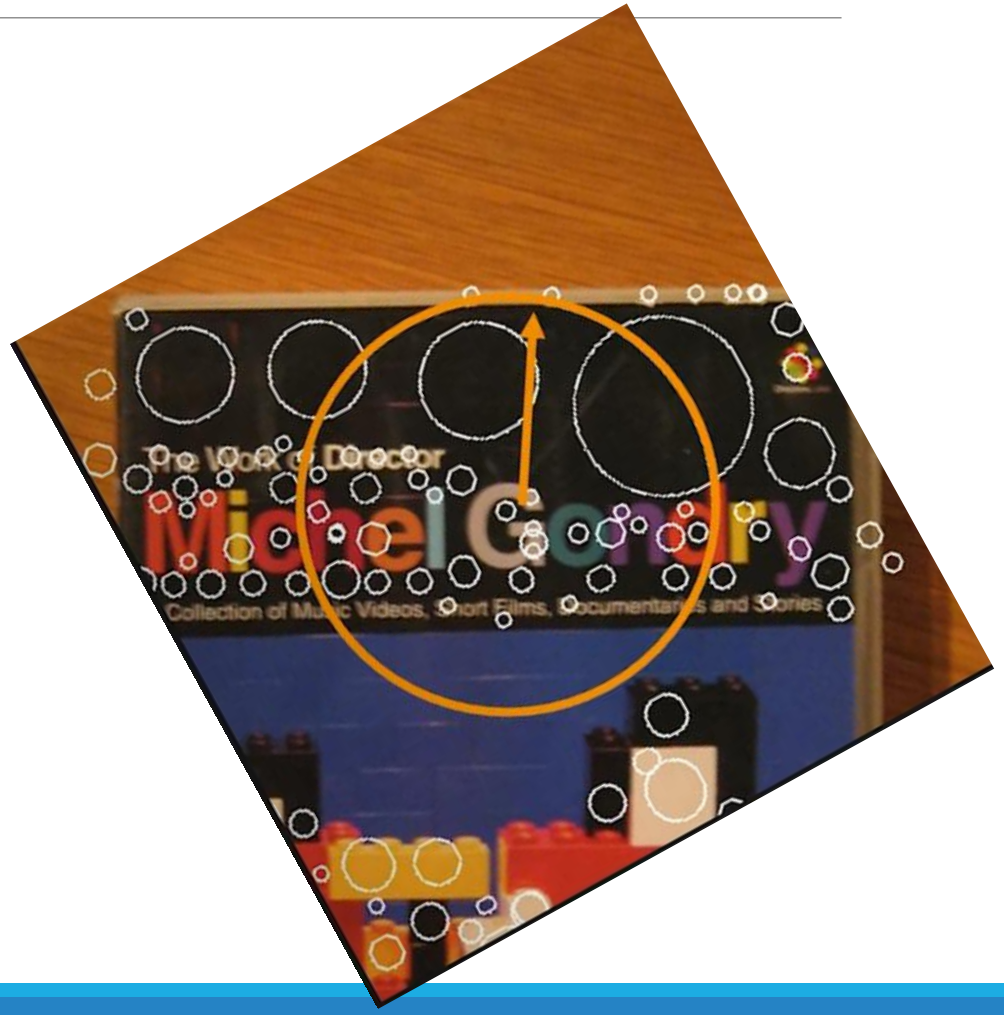




SIFT Rotation Invariance

Use the Principal orientation to undo rotation

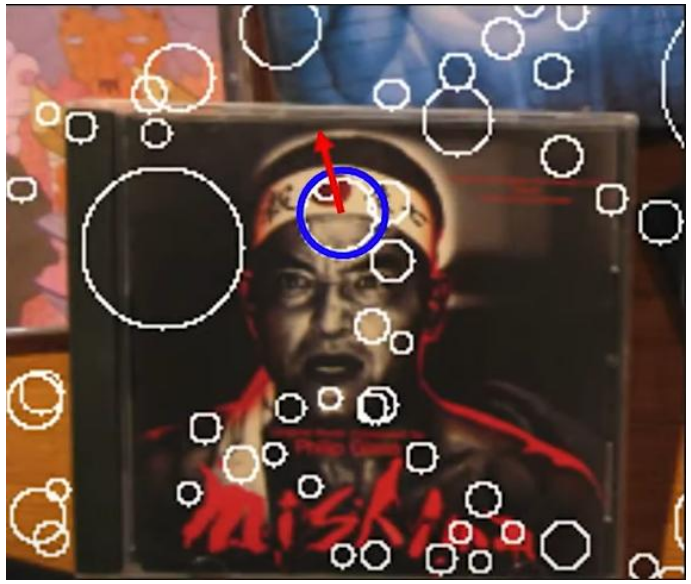




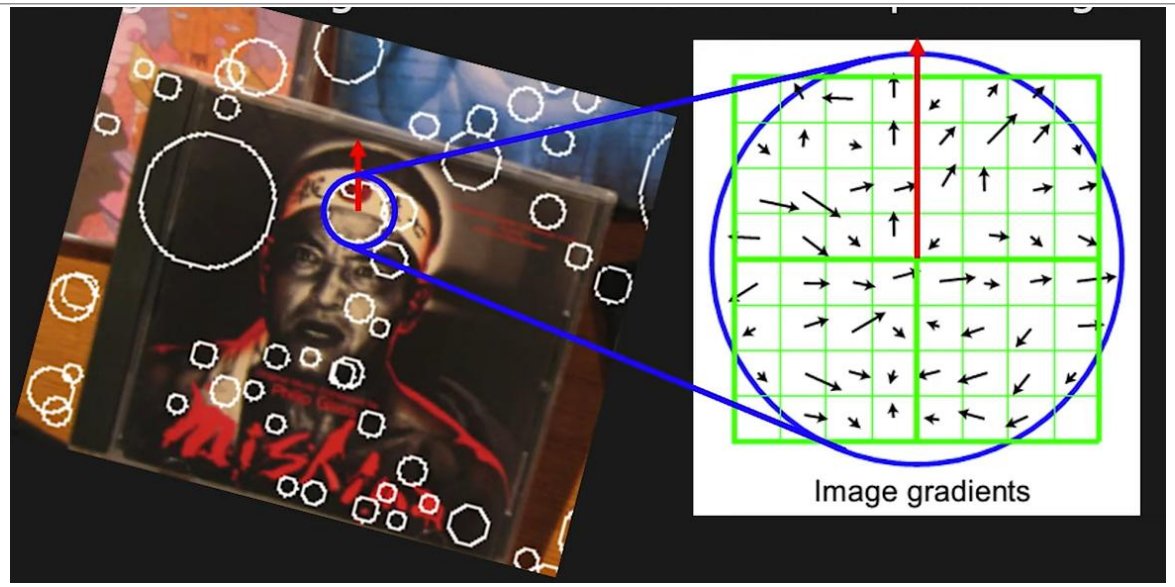
SIFT Descriptor

SIFT Descriptor

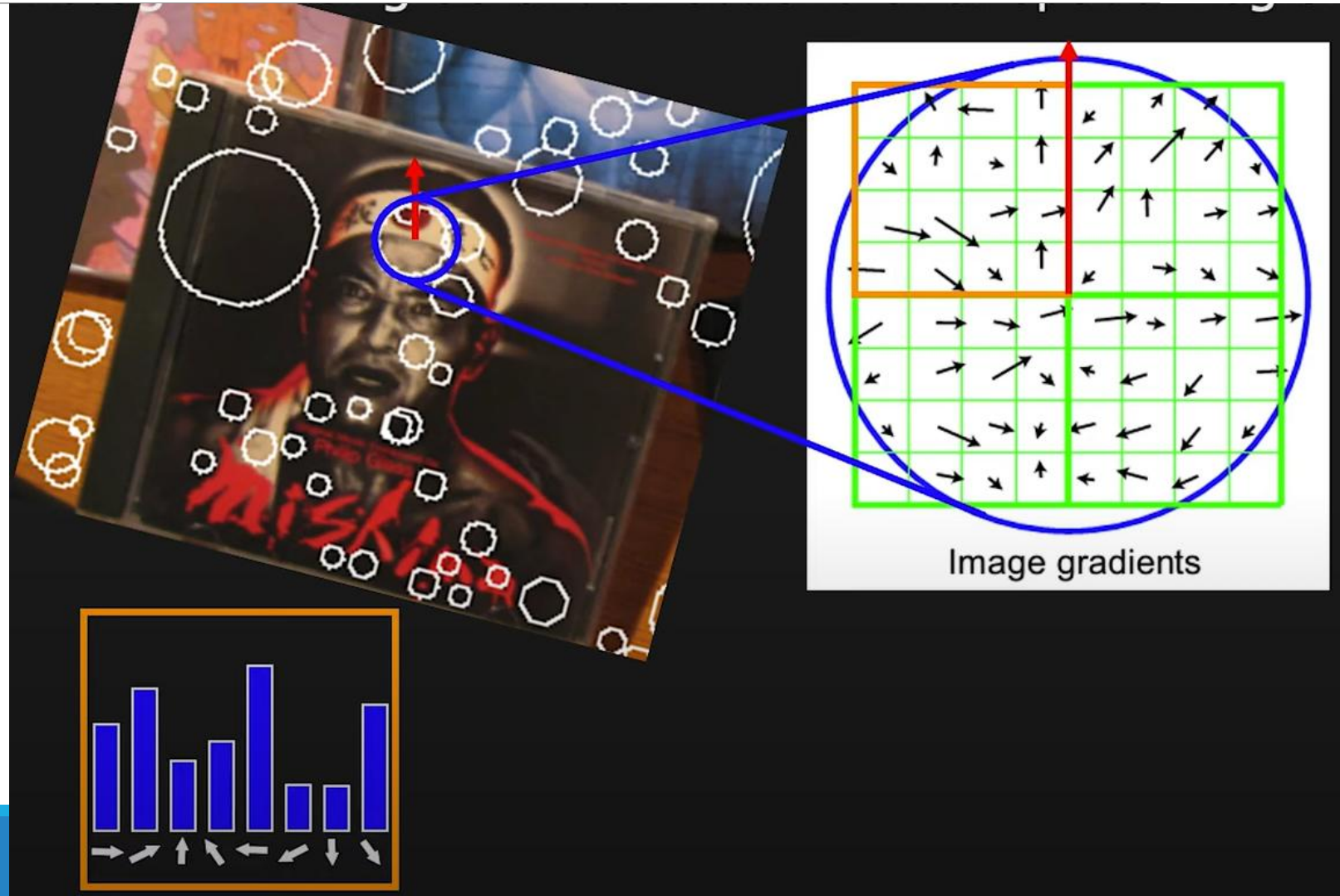
Histogram of gradient directions over spatial regions



SIFT Descriptor



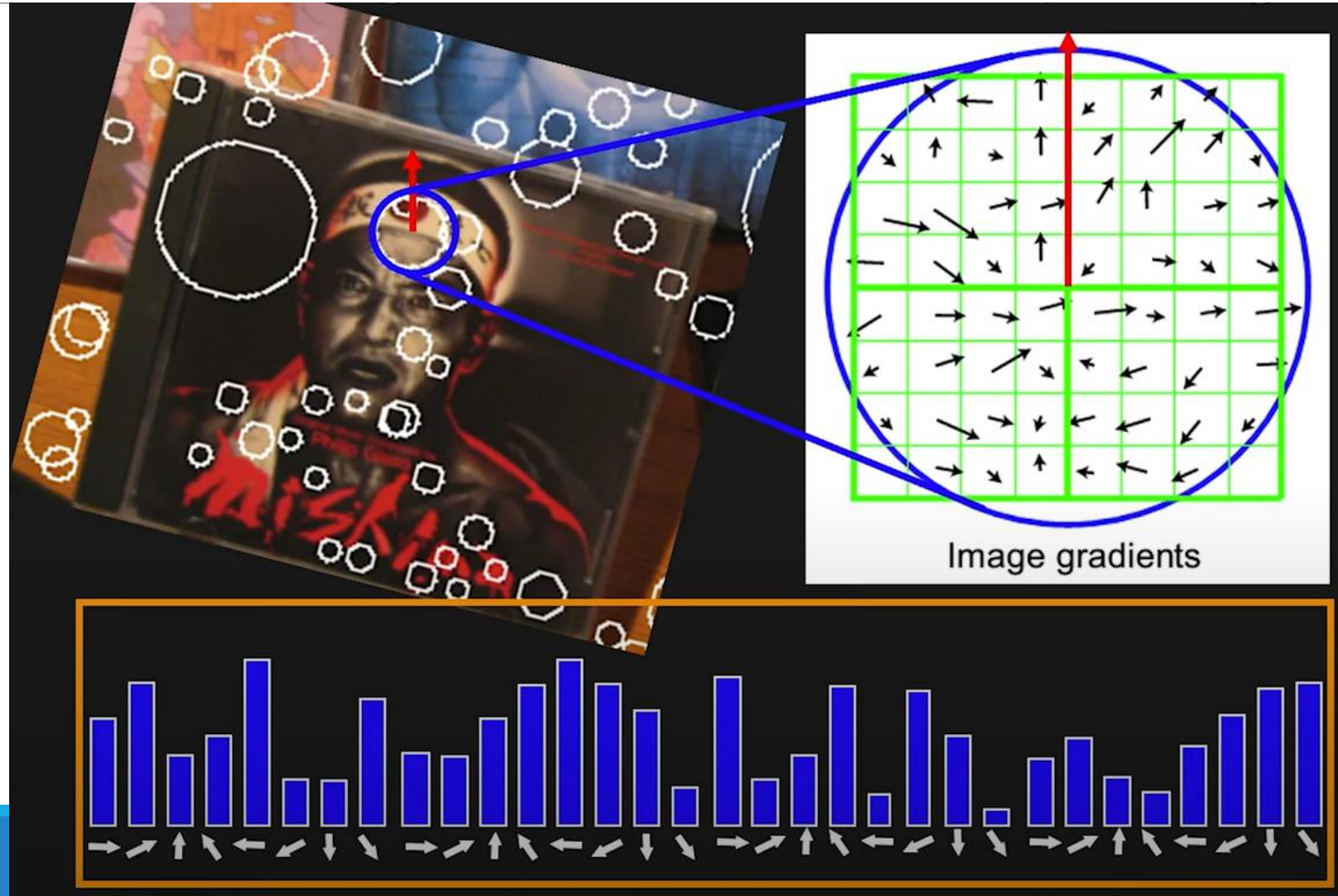
SIFT Descriptor



SIFT Descriptor

Only the orientation is considered, and magnitude of the gradient is ignored that makes it insensitive to Brightness.

Normalized Histogram: Invariant to Rotation, Scale, and Brightness



Comparing SIFT Descriptors

- Essentially comparing two arrays of data
- Let $H_1(k)$ and $H_2(k)$ be two arrays of data of length N
- **L2 Distance:**

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

- Smaller the distance metric, better the match
- Perfect match when $d(H_1, H_2) = 0$

Comparing SIFT Descriptors

- Essentially comparing two arrays of data
- Let $H_1(k)$ and $H_2(k)$ be two arrays of data of length N
- **Normalized Correlation:**

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

- Larger the distance metric, better the match
- Perfect match when $d(H_1, H_2) = 1$

Comparing SIFT Descriptors

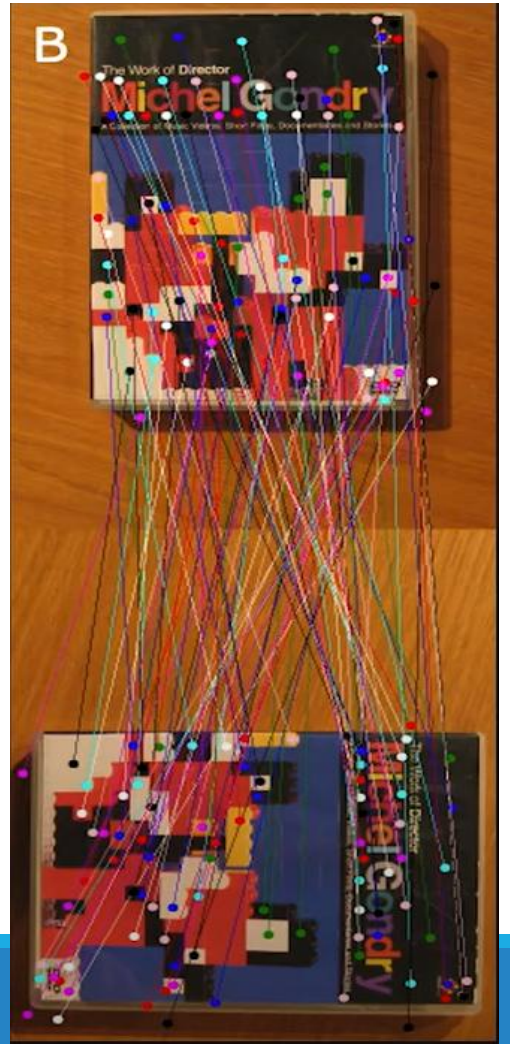
- Essentially comparing two arrays of data
- Let $H_1(k)$ and $H_2(k)$ be two arrays of data of length N
- **Intersection:**

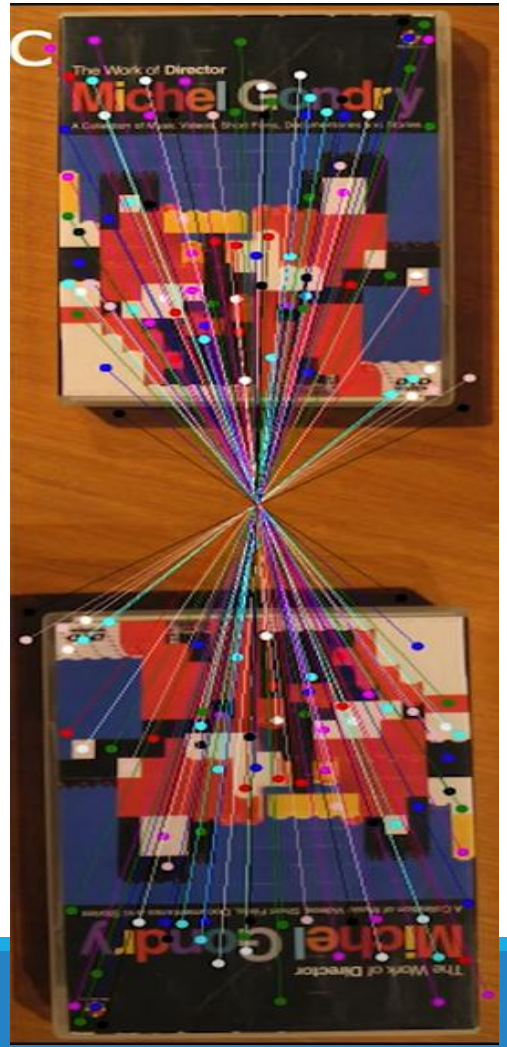
$$d(H_1, H_2) = \sum_k \min(H_1(k), H_2(k))$$

- Larger the distance metric, better the match
- Perfect match when $d(H_1, H_2) = 1$

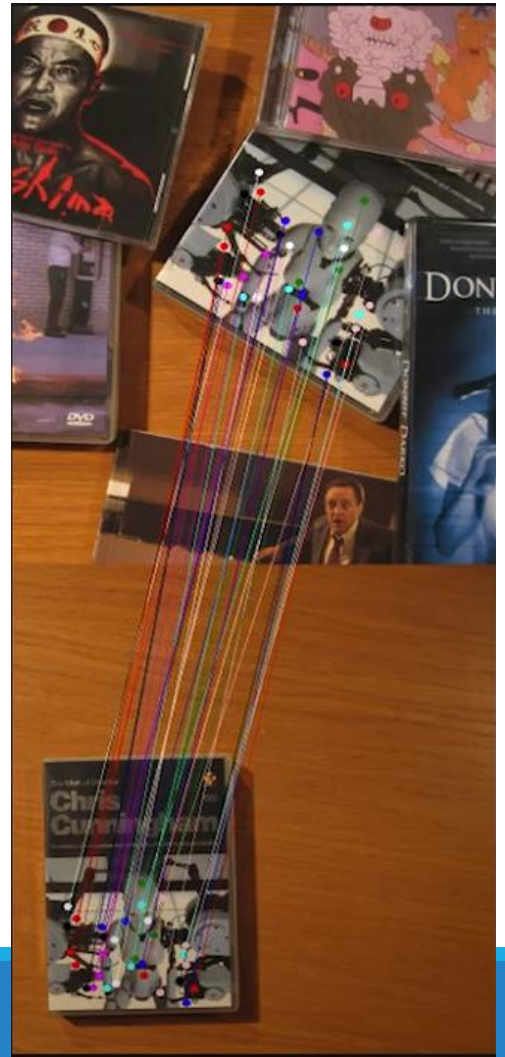
SIFT Results: Scale Invariance

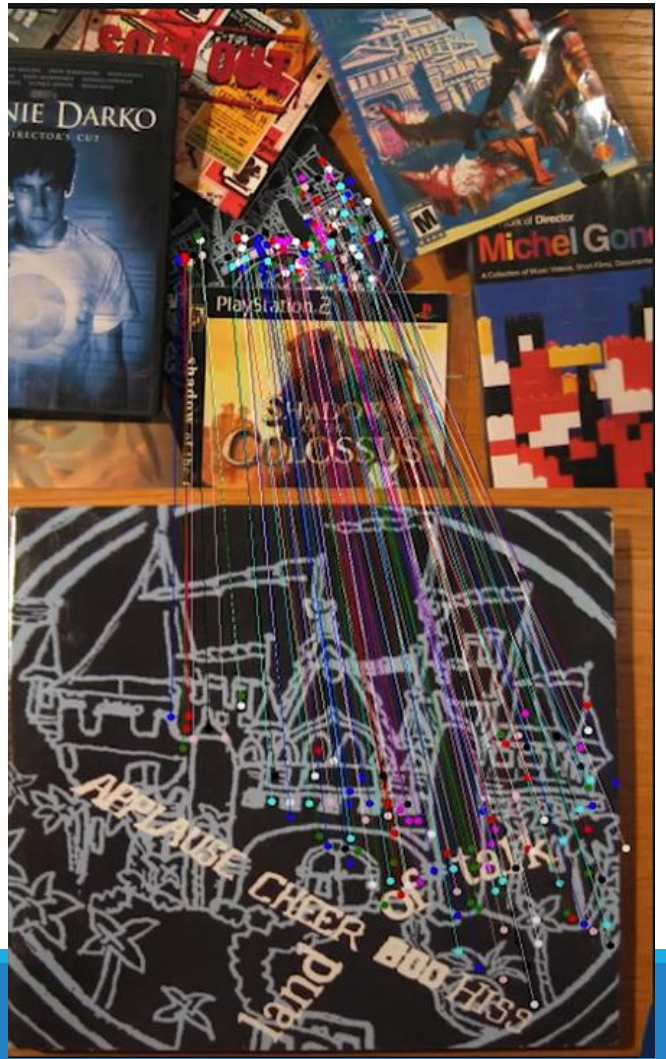
SIFT Results: Rotation Invariance





SIFT Results: Robustness to Clutter





Thank you