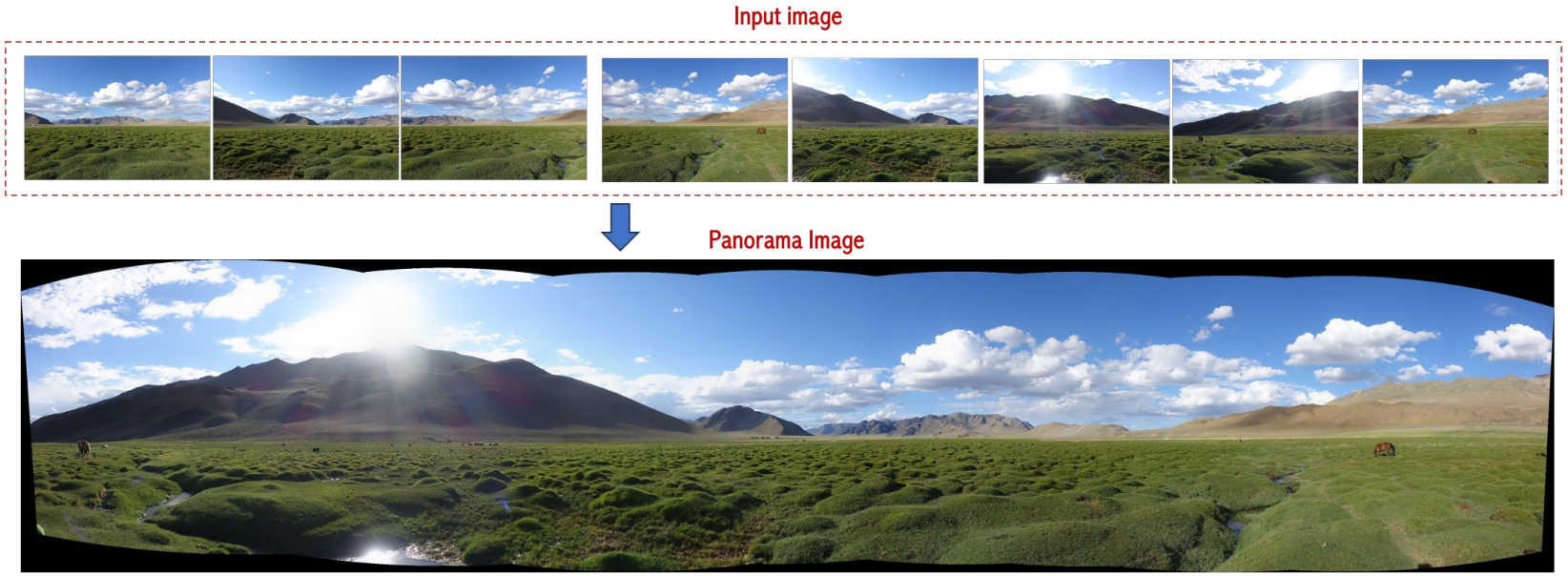
AI VIETNAM
All-in-One Course

Image Stitching: Panorama Image

(Computer Vision Foundation)



Vinh Dinh Nguyen
PhD in Computer Science

Year 2023

Outline

- Image Stitching/Panorama Image
- Edge Detector
- Blob Detector
- SIFT detector
- Image Transformation: 2D & 3D
- Image stitching/panorama Techniques

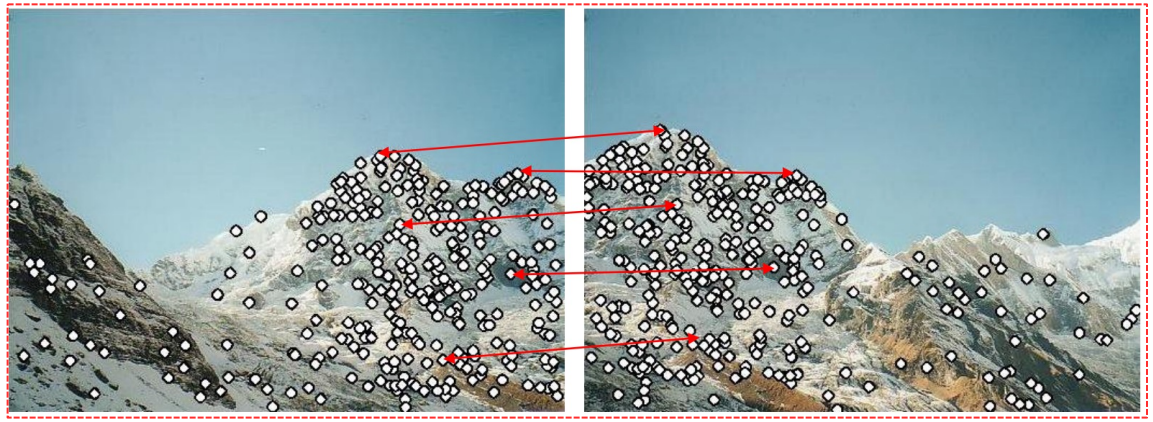
Outline

- Image Stitching/Panorama Image
- Edge Detector
- Blob Detector
- SIFT detector
- Image Transformation: 2D & 3D
- Image stitching/panorama Techniques

How to build a Panaroma

We need to match (align) images

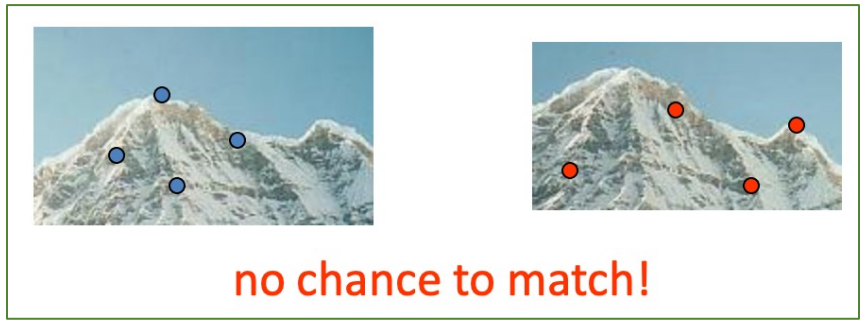
How to build a Panaroma



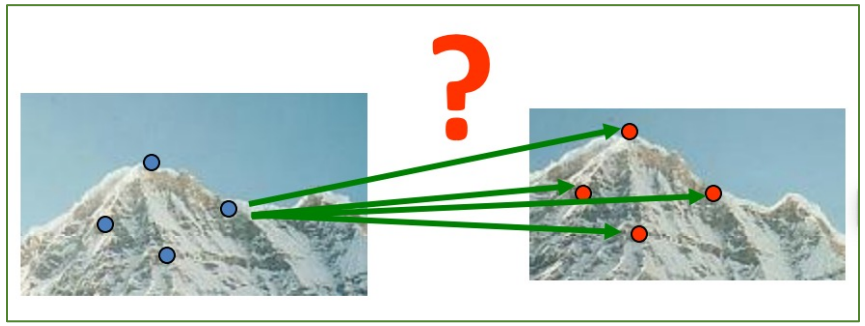
- Detect feature points in both images
- Find corresponding pairs
- Use these pairs to align images

Problems

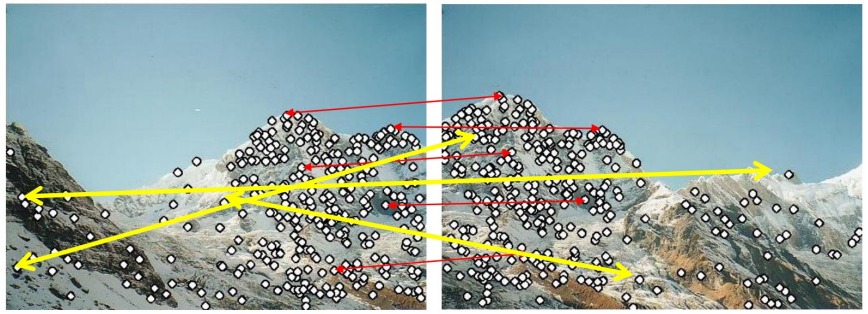
Detect the same point independently in both images



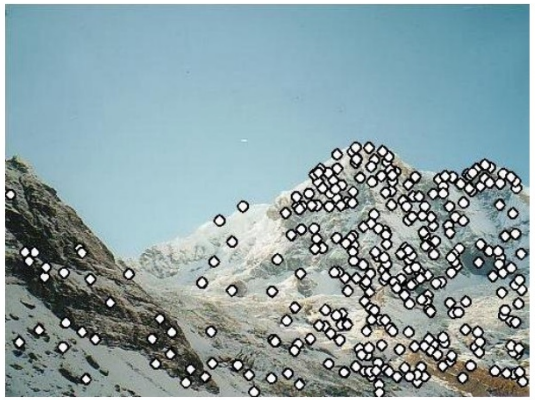
For each point correctly recognize the corresponding one



Need to estimate transformation between images, despite erroneous correspondences



Characteristics of Good Features

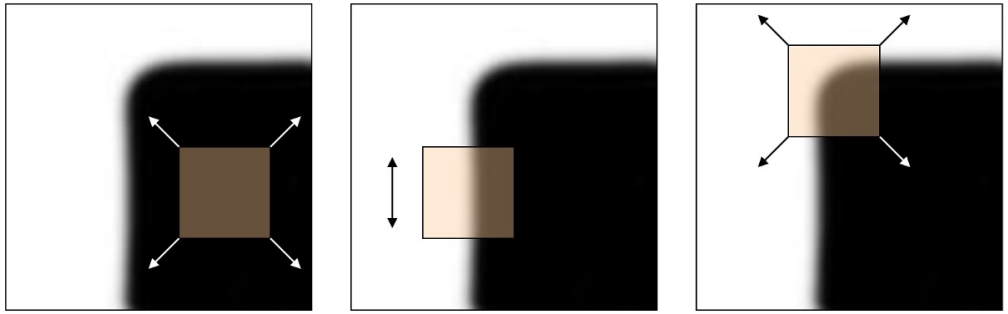


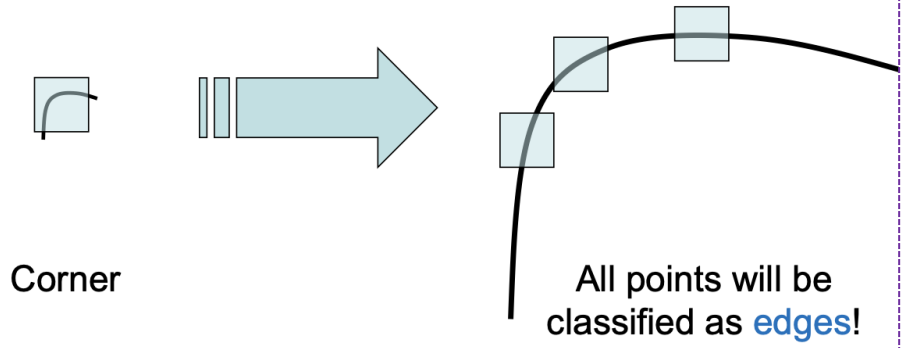

- Repeatability
  - The same feature can be found in several images despite geometric and photometric transformations
- Saliency
  - Each feature has a distinctive description
- Compactness and efficiency
  - Many fewer features than image pixels
- Locality
  - A feature occupies a relatively small area of the image; robust to clutter and occlusion

Feature points are used for:

- Motion tracking
- Image alignment
- 3D reconstruction
- Object recognition
- Indexing and database retrieval
- Robot navigation

Corners





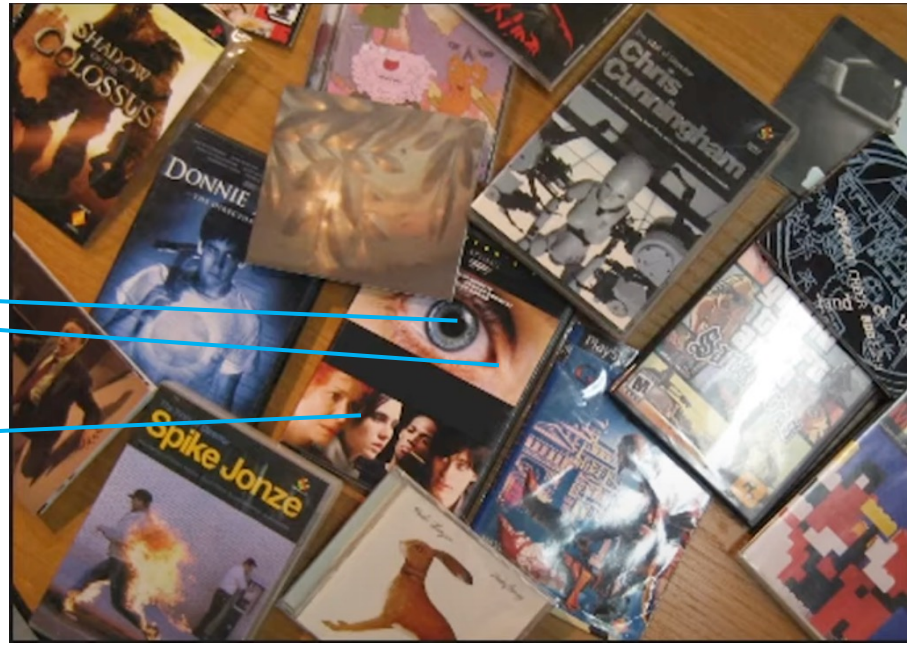
Not invariant to image scale!

Limitations

SIFT Detector: Motivation

How would you recognize the following types of objects

Template



Find and Match "Interesting Points or Features"

Scale Invariant Feature Transformation (SIFT): Image alignment and 2D object recognition

Image credit: Professor Shree Nayar who is faculty in the Computer Science Department, School of Engineering and Applied Sciences, Columbia University.

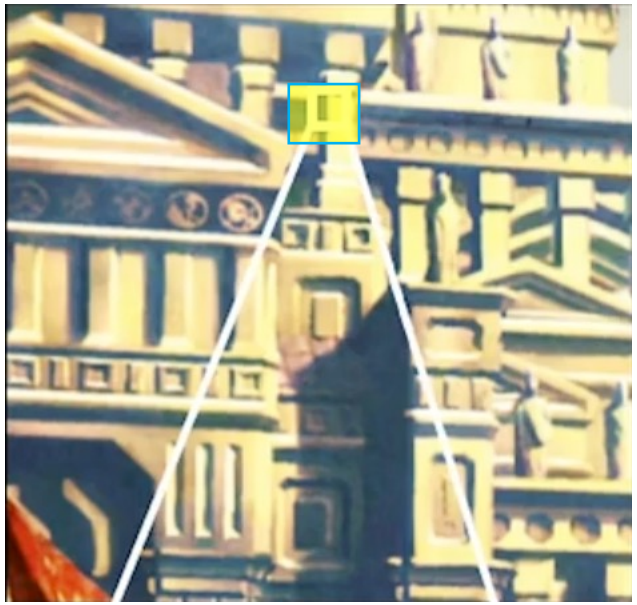
SIFT Detector: Interesting Point

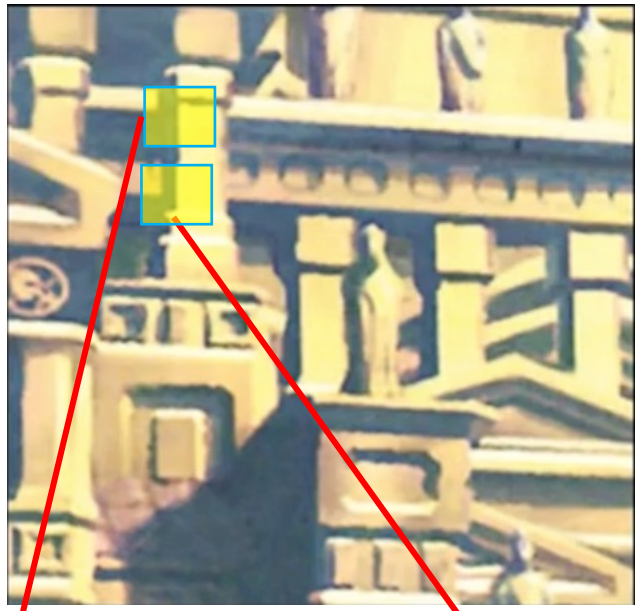
Diffferent size, orientation, lighting, brighness, etc,...

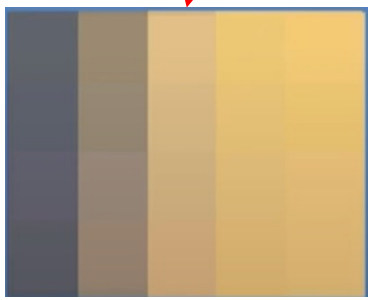
Interesting point: (1) Rich content within the local window. (2) Well-define representation for mathing/comparing with other points. (3) Well-define position in the image. (4) Should be invariant to image rotation and scaling. (5) Should be insenitve to lighting changes

Image credit: Professor Shree Nayar who is faculty in the Computer Science Department, School of Engineering and Applied Sciences, Columbia University.

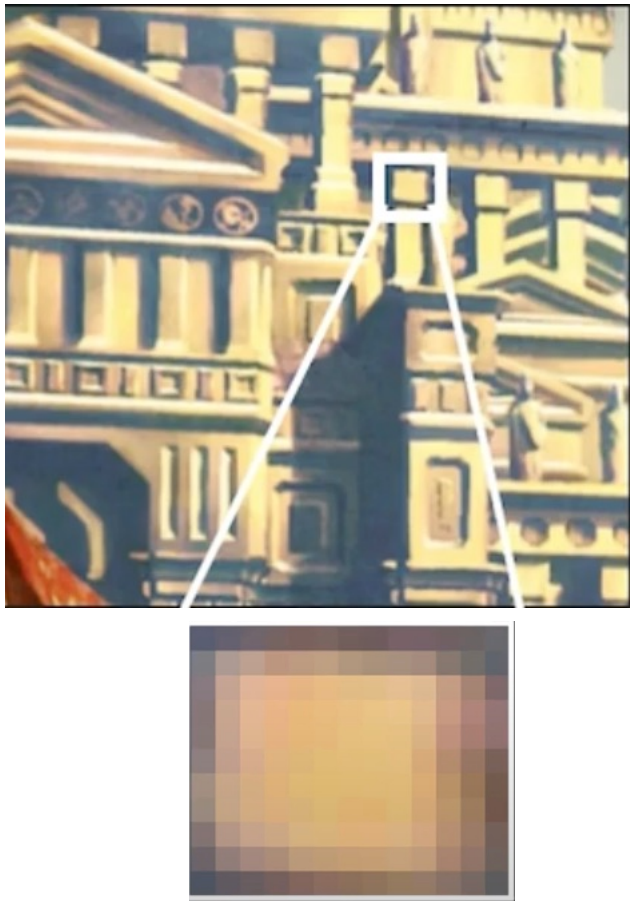
Are Line/Edges Interesting Point?

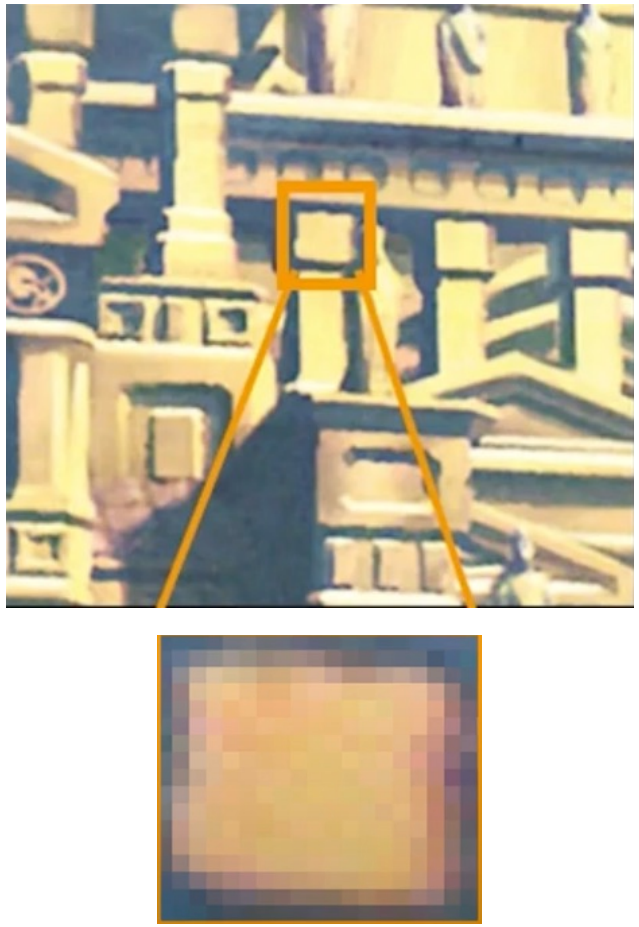






Are Blob Interesting?





Blob has fixed position and definite size: location, size, and orientation

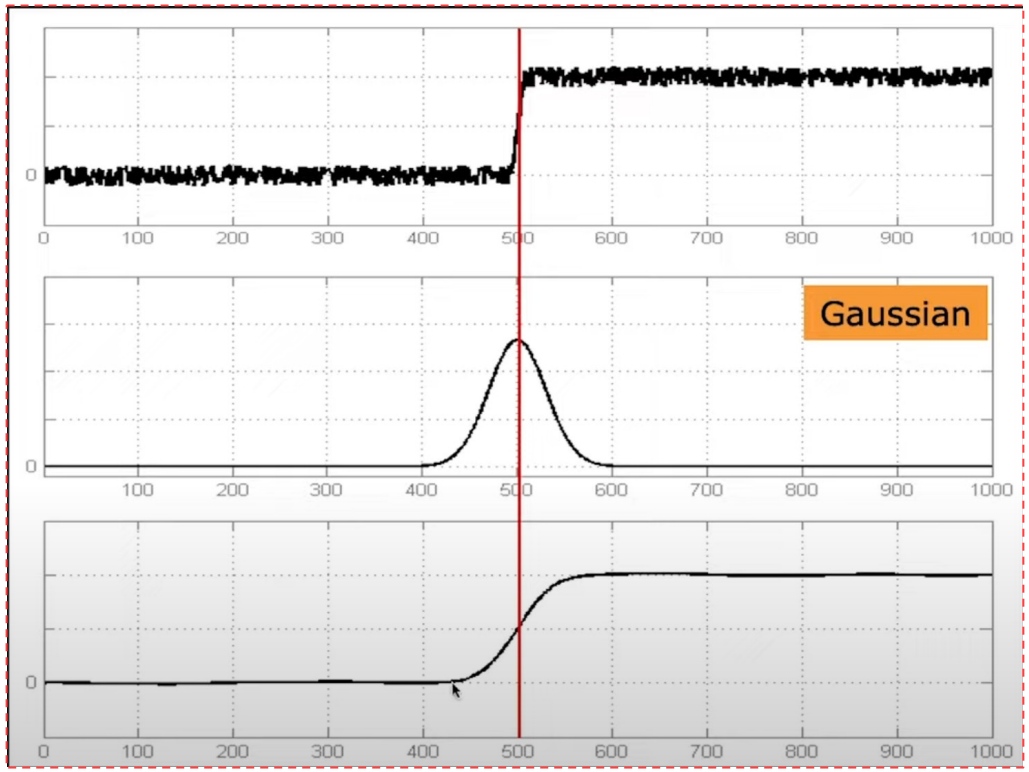
A Blob, in a sense, is anything that is considered a large object or anything bright in a dark background, in images, we can generalize it as a group of pixel values that forms a somewhat colony or a large object that is distinguishable from its background.

Gaussian Filter: Review

f

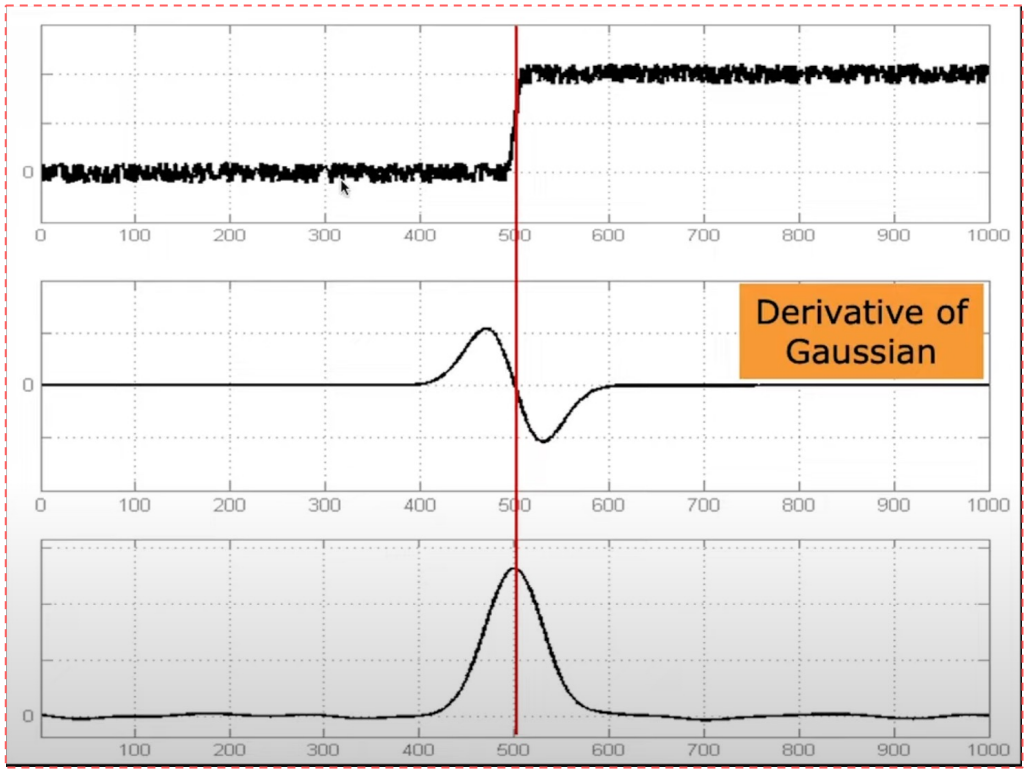
n_σ

n_σ*f



∇n_σ

∇n_σ*f



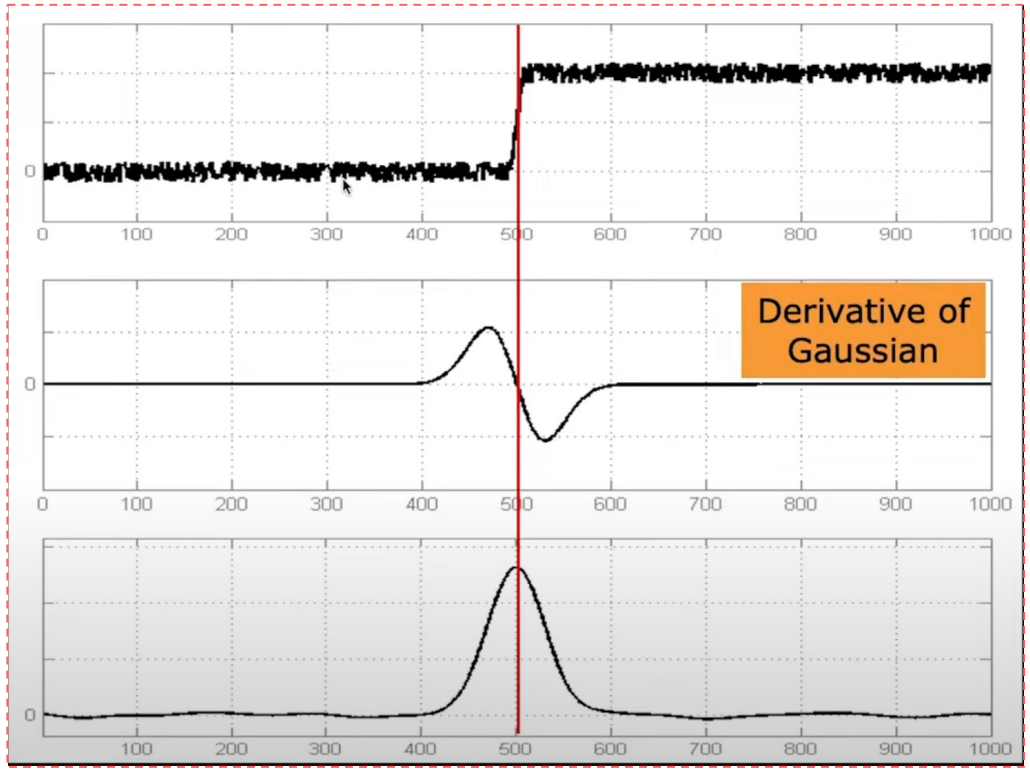
Gaussian Filter is used to remove noise by smoothing

Extremum of Derivative of Gaussian denotes an edge.

Gaussian Filter: Review

∇n_σ

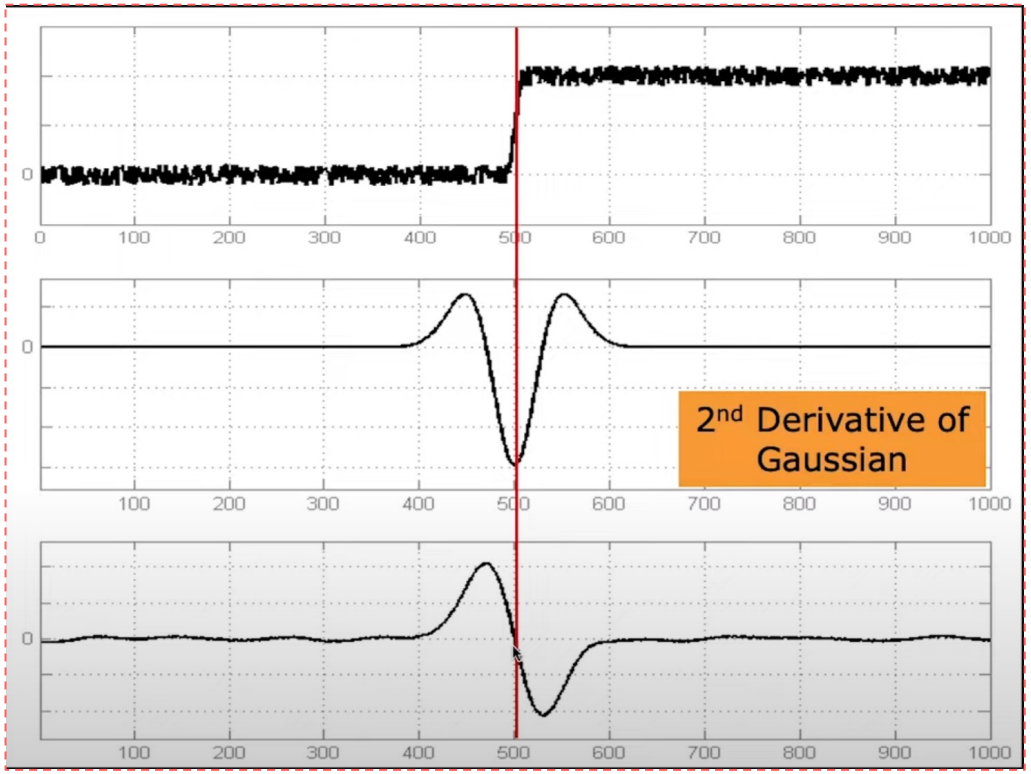
∇n_σ*f



Extremum of Derivative of Gaussian denotes an edge.

$\nabla^2 n_\sigma$

$\nabla^2 n_\sigma$*f

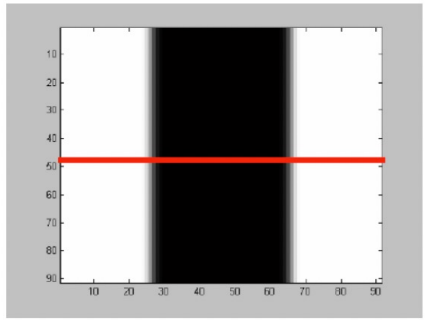


Zero crossing in 2nd derivative of Gaussian denote an Edge

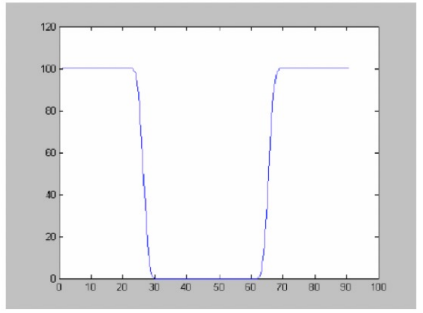
Outline

- What is image stitching/panorama image
- Edge Detector
- Blob Detector
- SIFT detector
- Image Transformation: 2D & 3D
- Image stitching/panorama Techniques

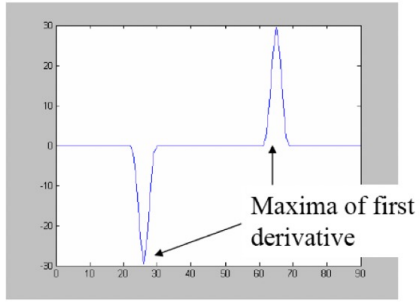
Edge Detection



Image



Intensities of pixels on the red line



1st derivative of pixel intensities

$$\frac{\partial f(x,y)}{\partial x} = \lim_{\varepsilon \to 0} \frac{f(x+\varepsilon, y) - f(x,y)}{\varepsilon}$$

Continuous data

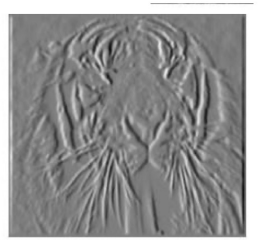
$$\frac{\partial f(x,y)}{\partial x} \approx \frac{f(x+1,y) - f(x,y)}{1}$$

Discrete data

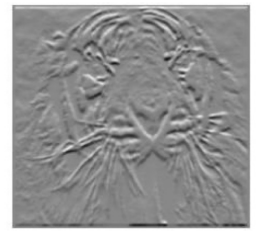
1	-1

Kernel

Image f

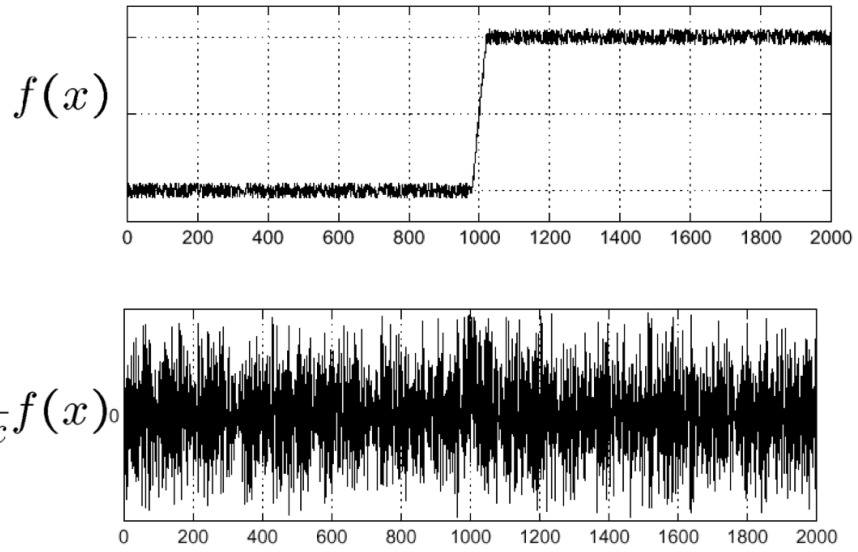


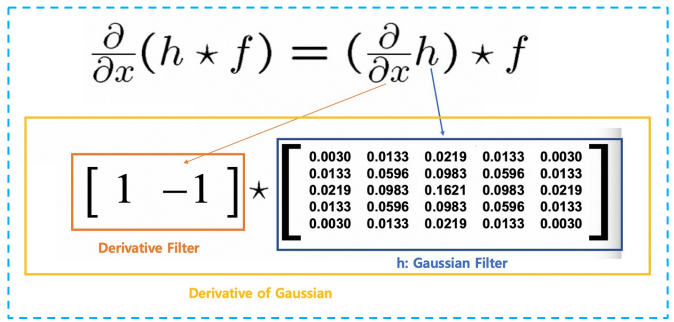
$\frac{\partial f(x,y)}{\partial x}$

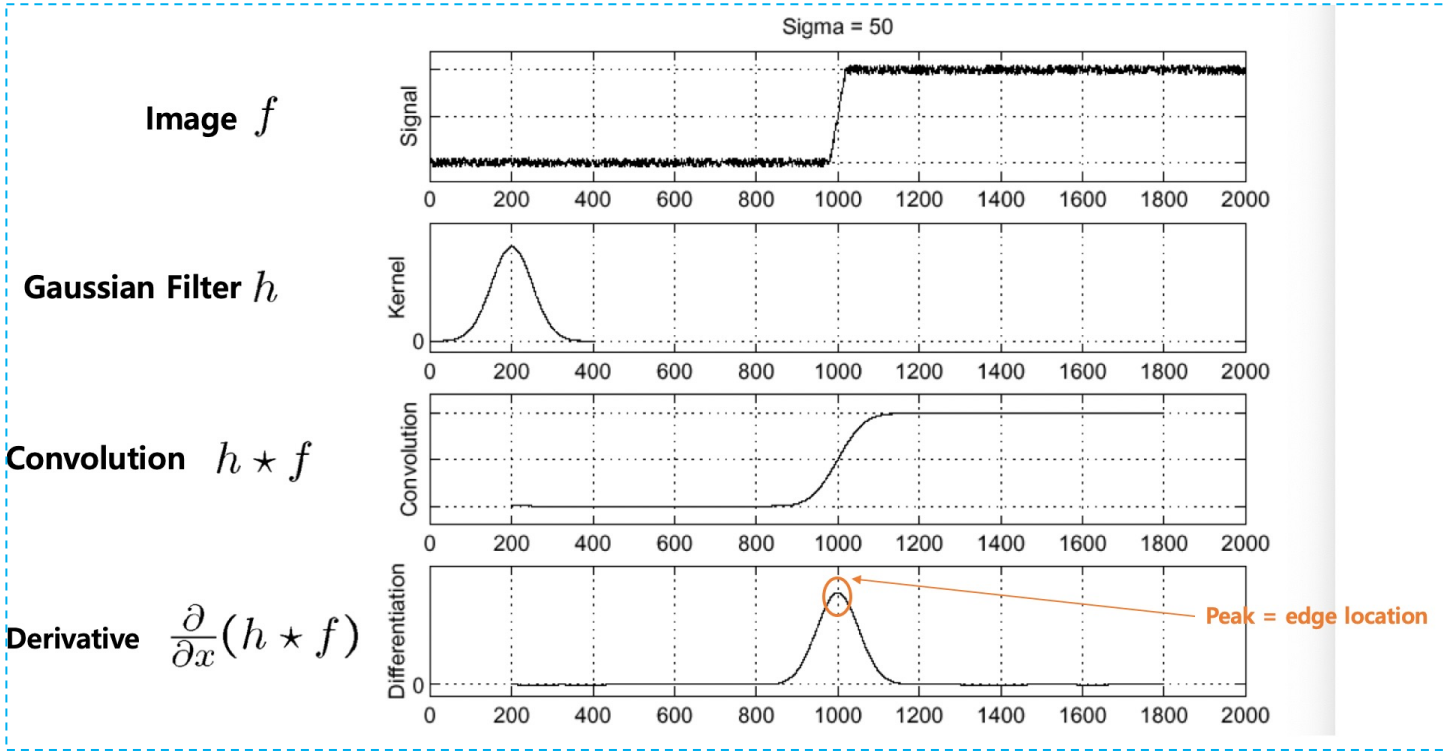


$\frac{\partial f(x,y)}{\partial y}$

Edge Detection with Gaussian Filter

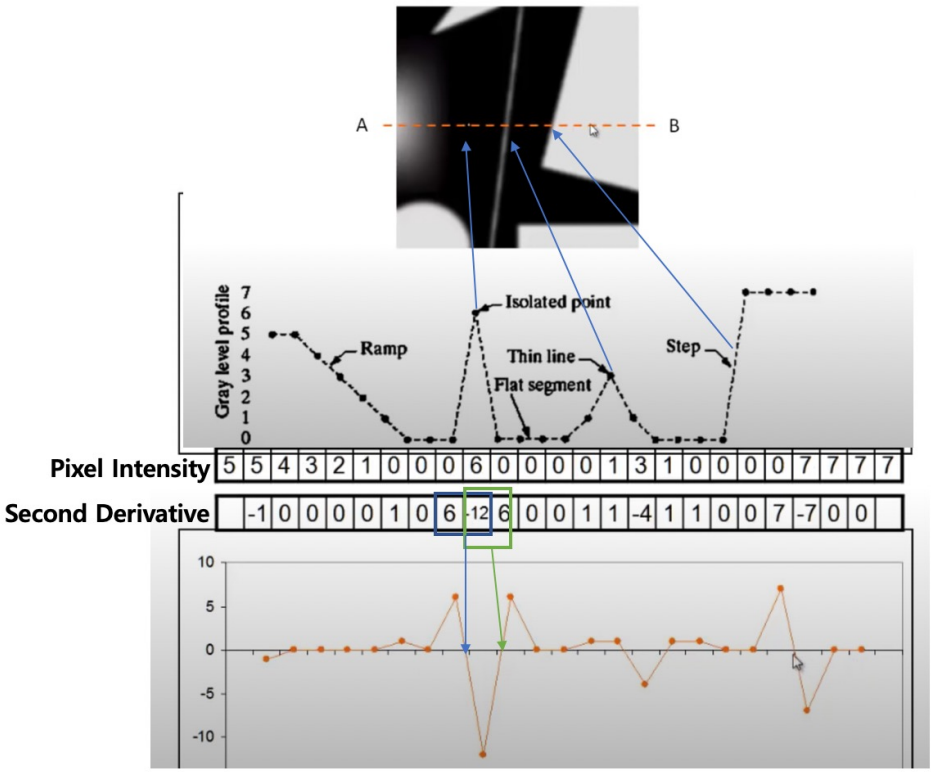






The result of this convolution (Derivative filter $\star h$) is called the **Derivative of Gaussian**

Laplacian as Edge Detector

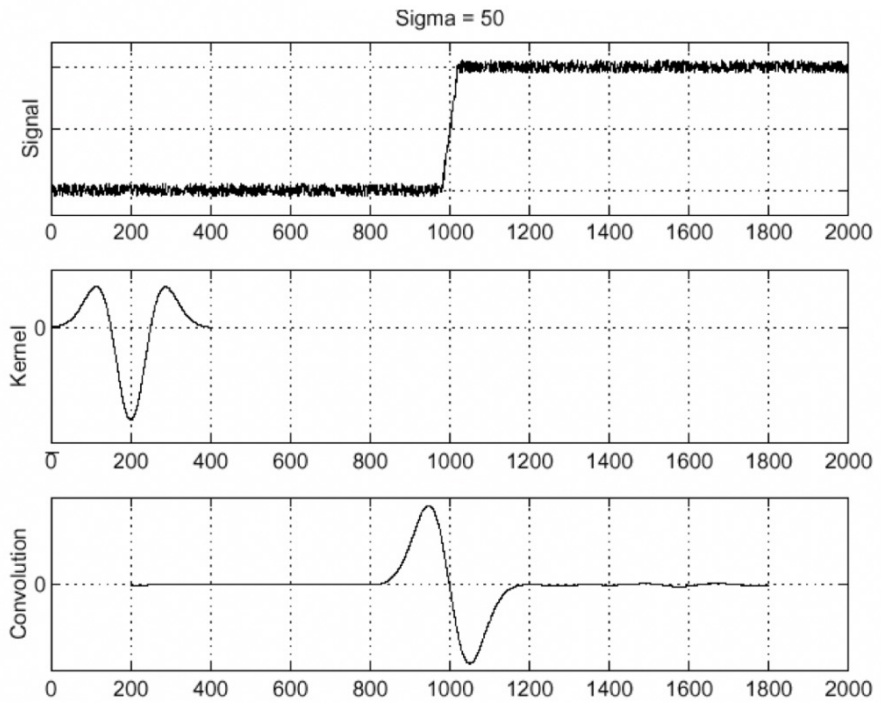


f

$\frac{\partial^2}{\partial x^2} h$

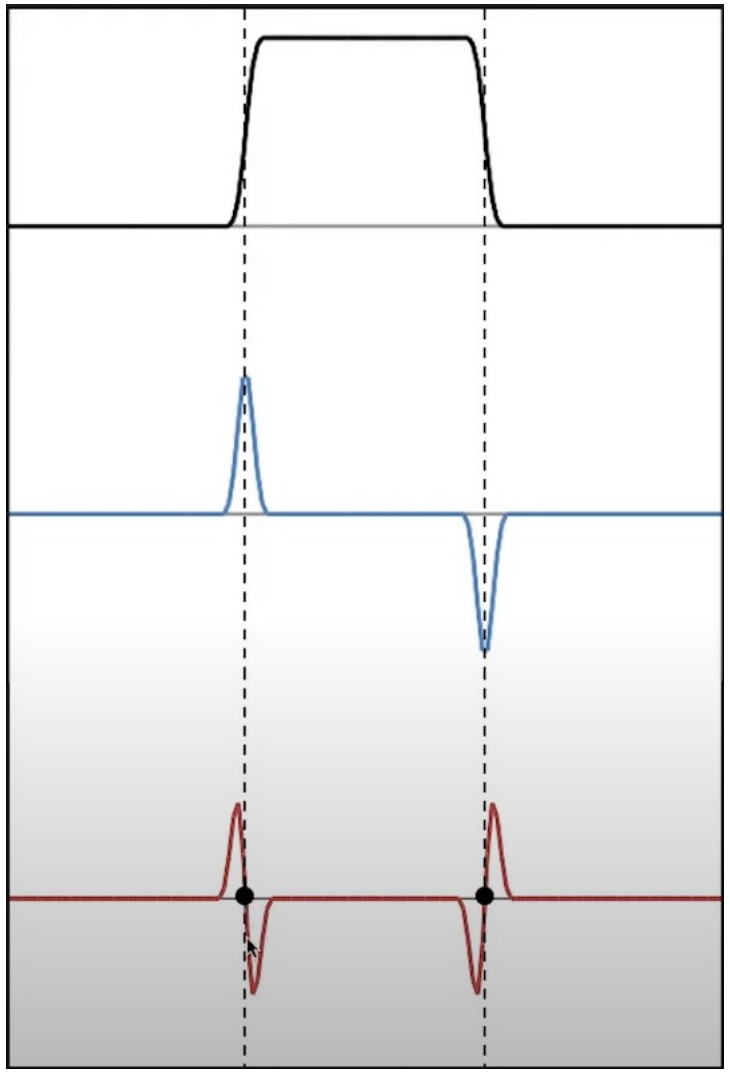
Laplacian of Gaussian

$(\frac{\partial^2}{\partial x^2} h) \star f$



$$\begin{aligned} \textit{Laplacian } \nabla^2 f &= \frac{\partial^2 f}{\partial^2 x} + \frac{\partial^2 f}{\partial^2 y} \\ &= f(x+1,y) + f(x-1,y) - 2f(x,y) + f(x,y+1) + f(x,y-1) - 2f(x,y) \\ &= f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) - 4f(x,y) \end{aligned}$$

Laplacian as Edge Detector



Local extrema indicate Edges

Zero-crossing indicate edges

Laplacian: Sum of Pure Second Derivatives

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

Edges are "zero-crossing" in Laplacian of image
Laplacian does not provide directions of edges

0	1	0
1	−4	1
0	1	0

Kernel

$$\begin{aligned} Laplacian\ \nabla^2 f &= \frac{\partial^2 f}{\partial^2 x} + \frac{\partial^2 f}{\partial^2 y} \\ &= f(x+1,y) + f(x-1,y) - 2f(x,y) + f(x,y+1) + f(x,y-1) - 2f(x,y) \\ &= f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) - 4f(x,y) \end{aligned}$$

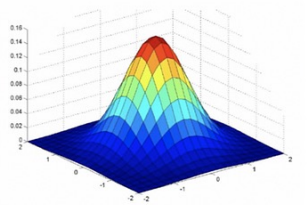
Laplacian of Gaussian

$$LoG = -\frac{1}{\pi\sigma^4}\left[1 - \frac{x^2+y^2}{2\sigma^2}\right]e^{-\frac{x^2+y^2}{2\sigma^2}}$$

0	−1	0
−1	4	−1
0	−1	0

Laplacian Filter

$\star$



Gaussian kernel

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

Gaussian Filter

=

0	1	1	2	2	2	1	1	0
1	2	4	5	5	5	4	2	1
1	4	5	3	0	3	5	4	1
2	5	3	-12	-24	-12	3	5	2
2	5	0	-24	-40	-24	0	5	2
2	5	3	-12	-24	-12	3	5	2
1	4	5	3	0	3	5	4	1
1	2	4	5	5	5	4	2	1
0	1	1	2	2	2	1	1	0

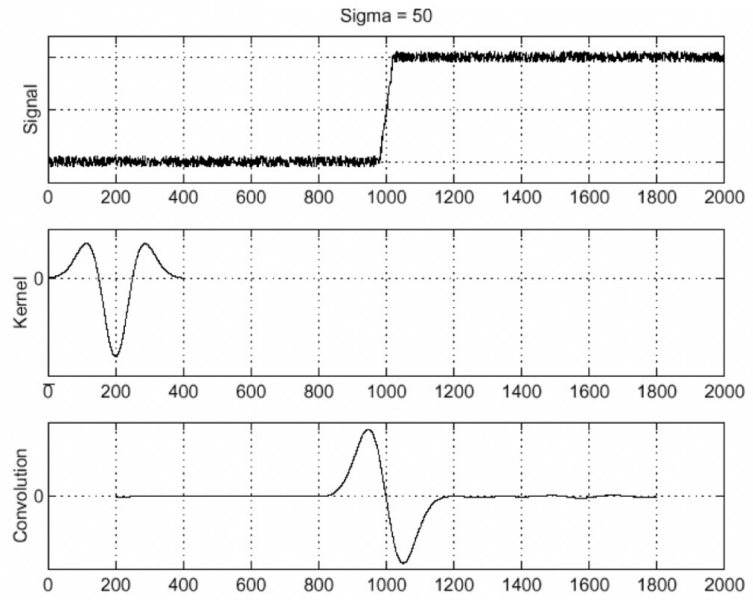
LoG Filter

f

$\frac{\partial^2}{\partial x^2}h$

Laplacian of Gaussian

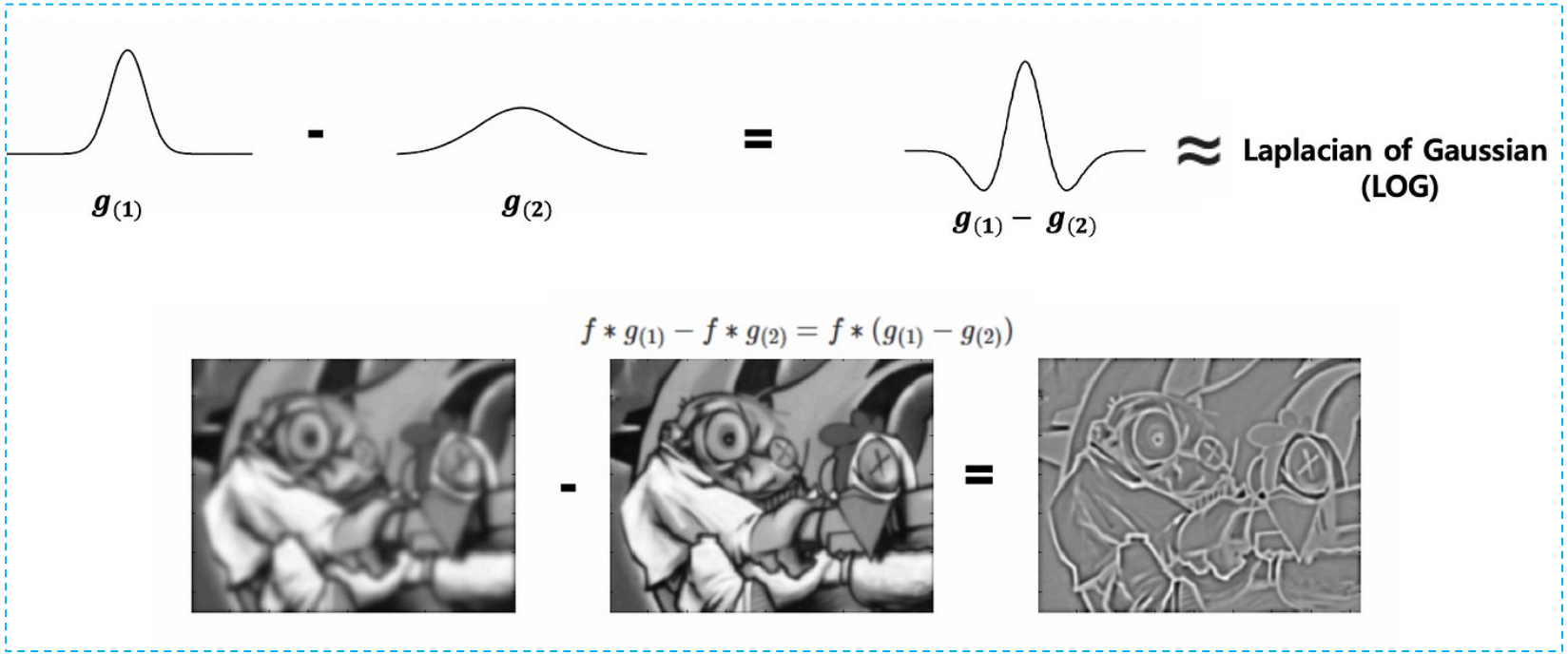
$(\frac{\partial^2}{\partial x^2}h) \star f$



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

$$\text{LoG}(x,y;\sigma) = \Delta_{(x,y)}G(x,y;\sigma) = \frac{\partial^2 G(x,y;\sigma)}{\partial x^2} + \frac{\partial^2 G(x,y;\sigma)}{\partial y^2} = \frac{1}{\pi\sigma^4}\left(\frac{x^2+y^2}{2\sigma^2} - 1\right)e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Difference of Gaussians (DoG)



Subtracting one Gaussian by another approximates the Laplacian of Gaussian

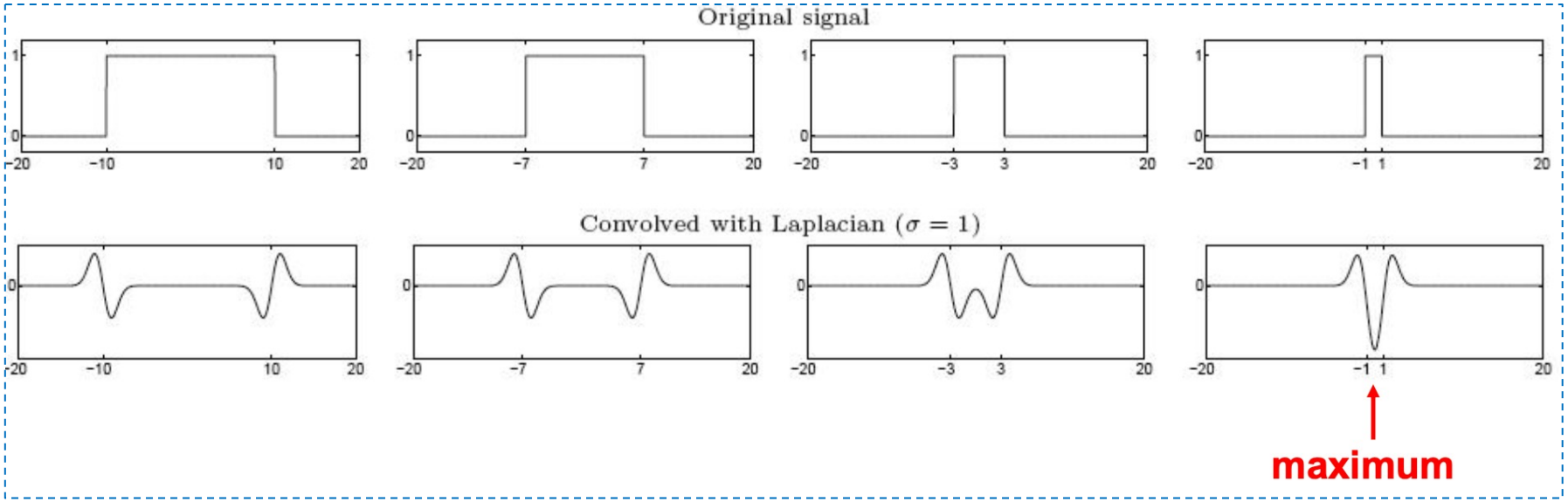
This indicates while approximating the LoG, there is no actual derivative computation needed

Outline

- What is image stitching/panorama image
- Edge Detector
- Blob Detector
- SIFT detector
- Image Transformation: 2D & 3D
- Image stitching/panorama Techniques

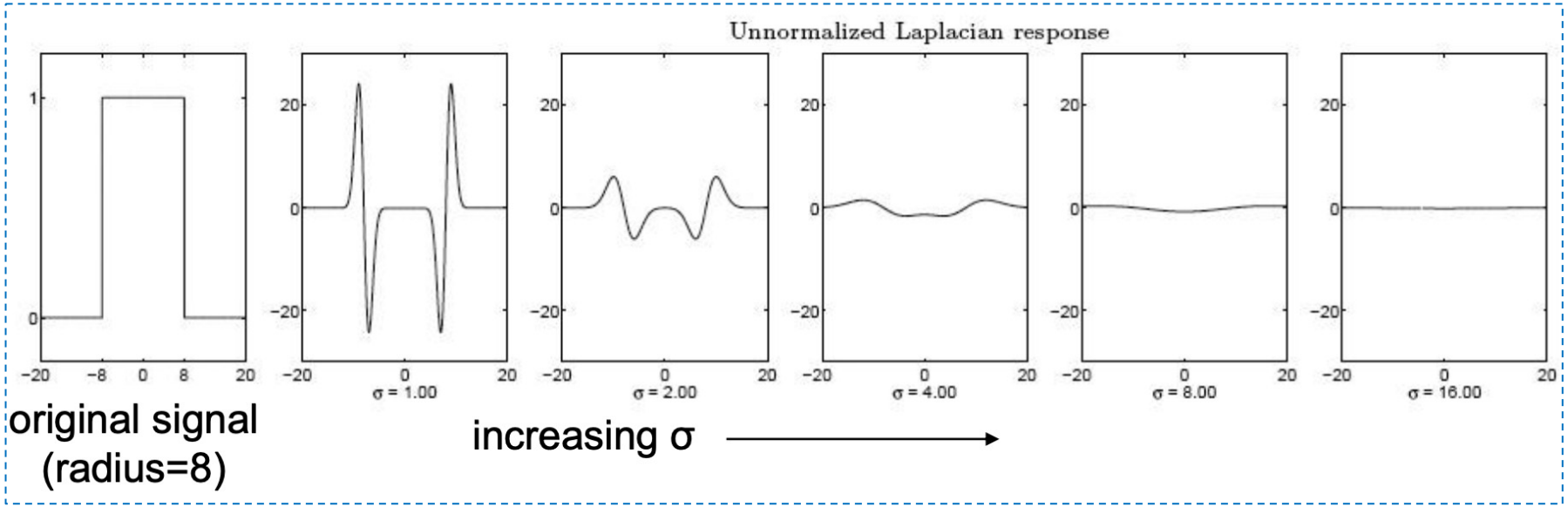
From Edges to Blobs

Edge = ripple
Blob = superposition of two ripples



From Edges to Blobs

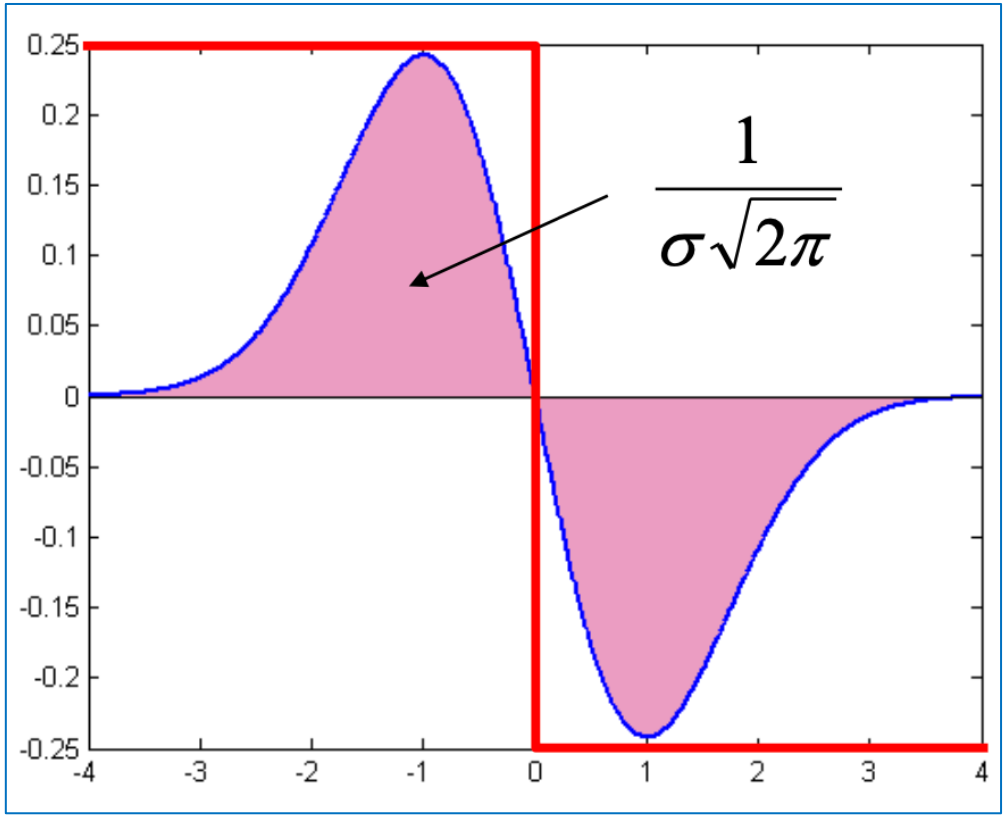
- We want to find the characteristic scale of the blob by convolving it with Laplacians at several scales and looking for the maximum response
- However, Laplacian response decays as scale increases



Why does this happen?

Scale normalization

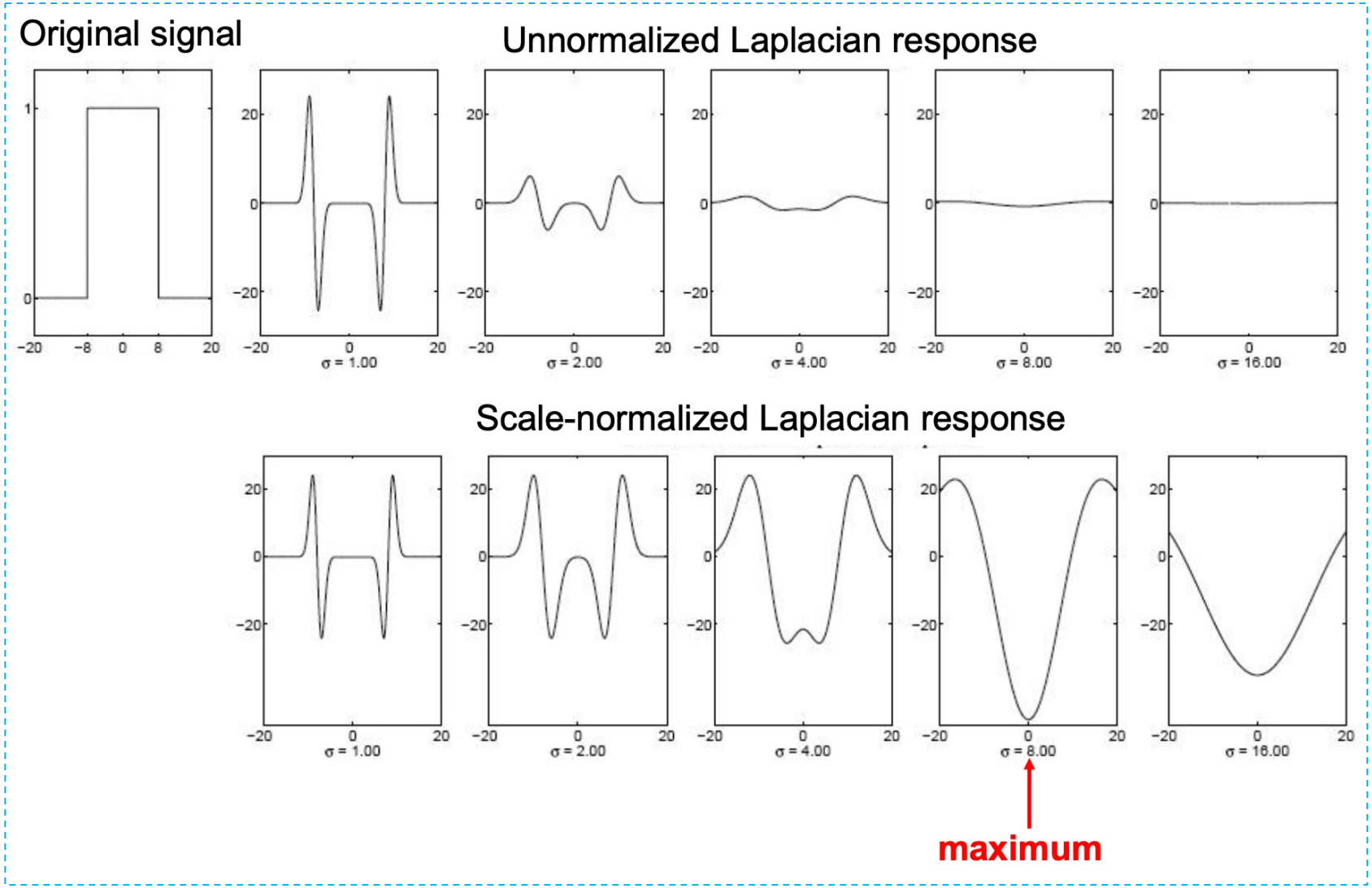
The response of a derivative of Gaussian filter to a perfect step edge decreases as σ increases



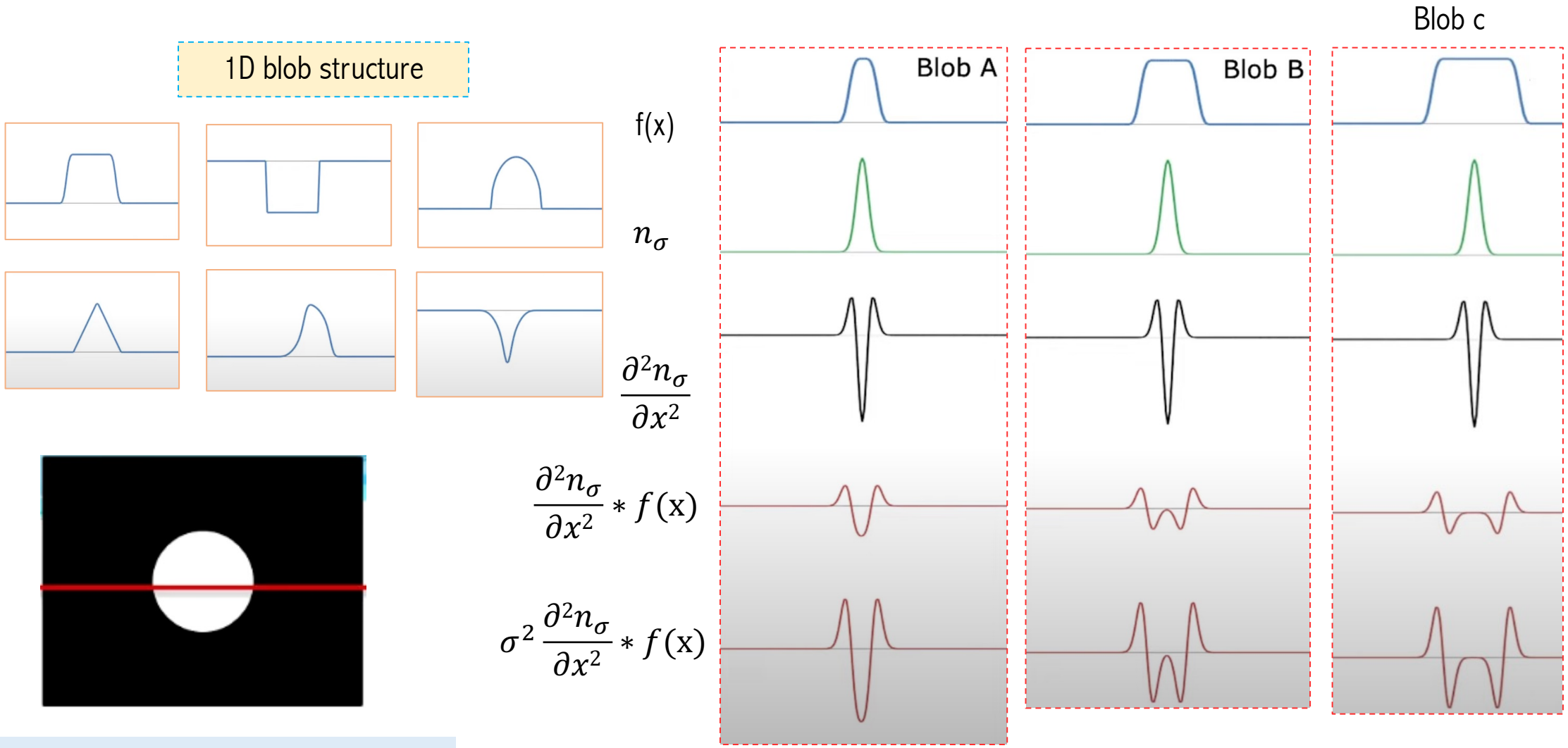
To keep response the same (scale-invariant), must multiply Gaussian derivative by σ

Laplacian is the second Gaussian derivative, so it must be multiplied by σ^2

Effect of Scale Normalization

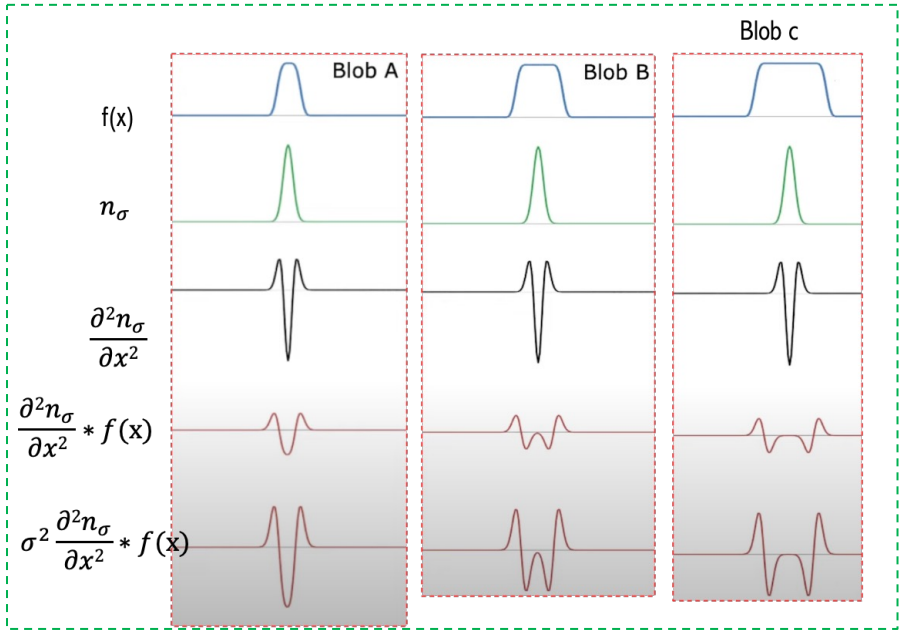


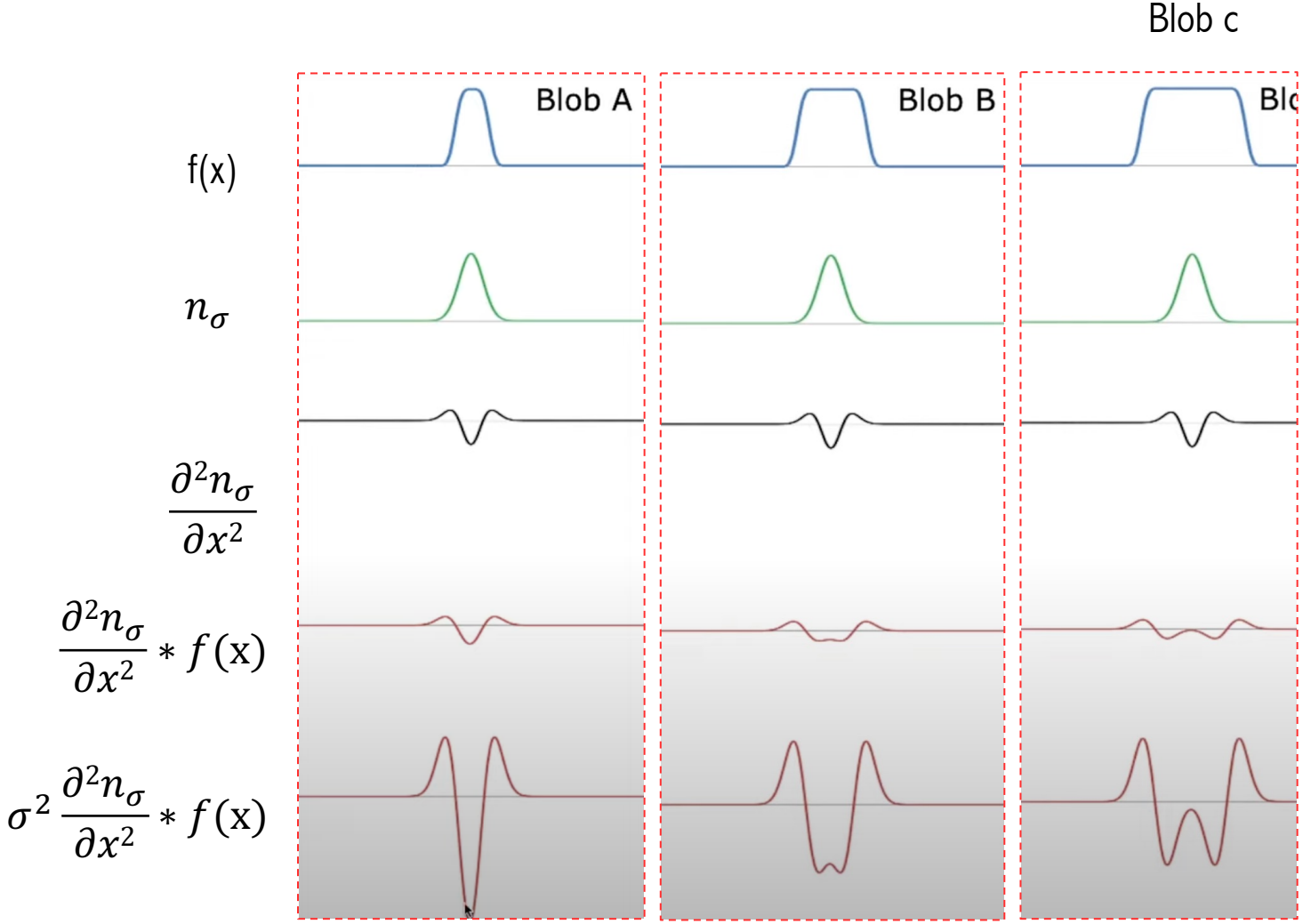
How to detect Blobs



From Prof. Shree Nayar Lecture

How to detect Blobs



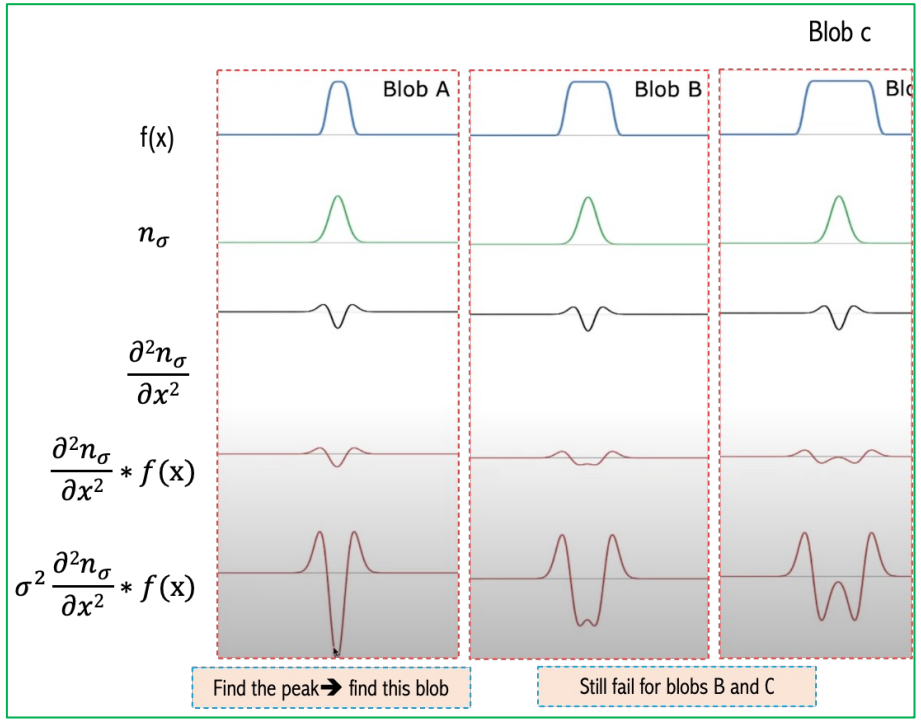


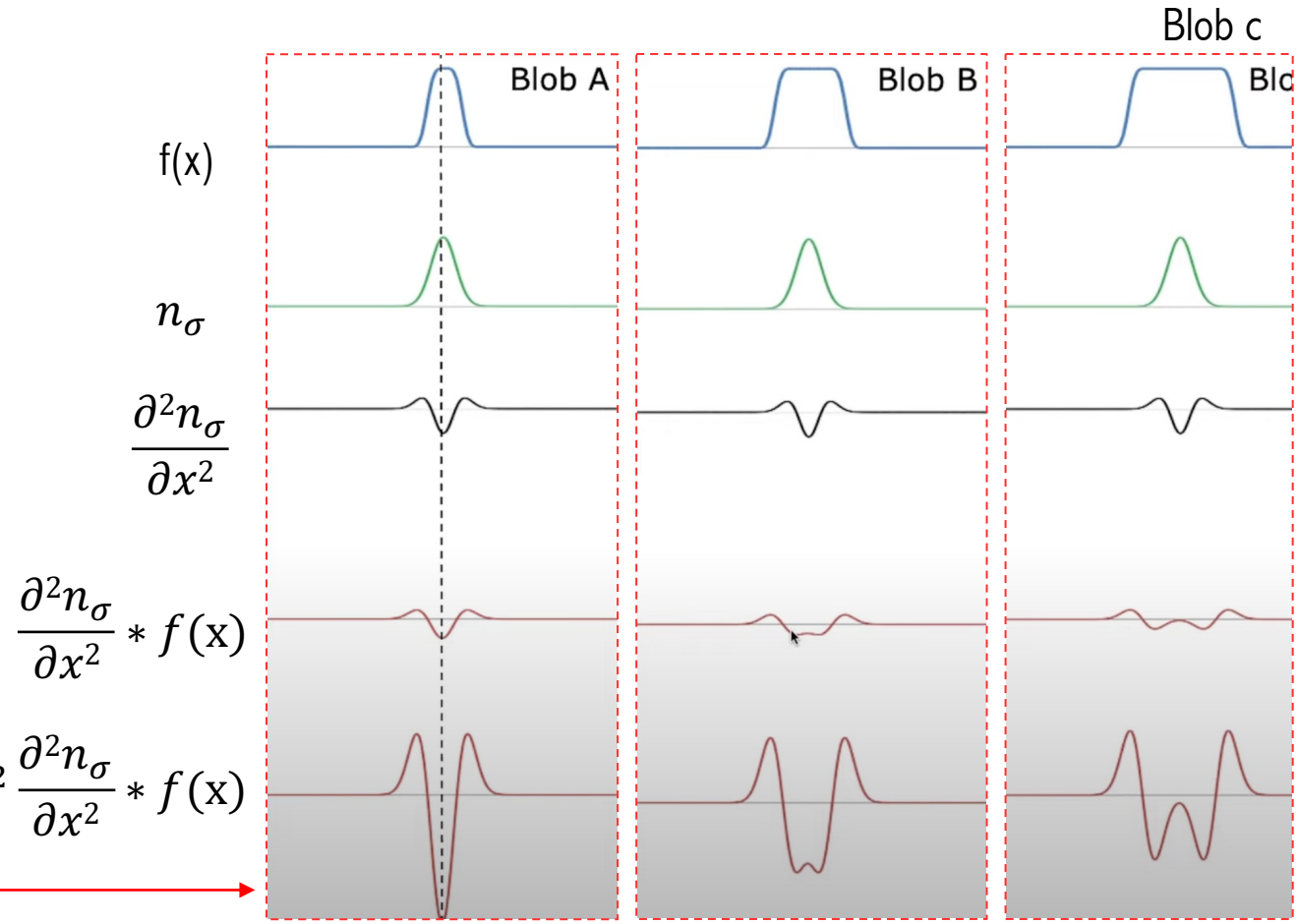
Find the peak➔ find this blob

Still fail for blobs B and C

From Prof. Shree Nayar Lecture

How to detect Blobs

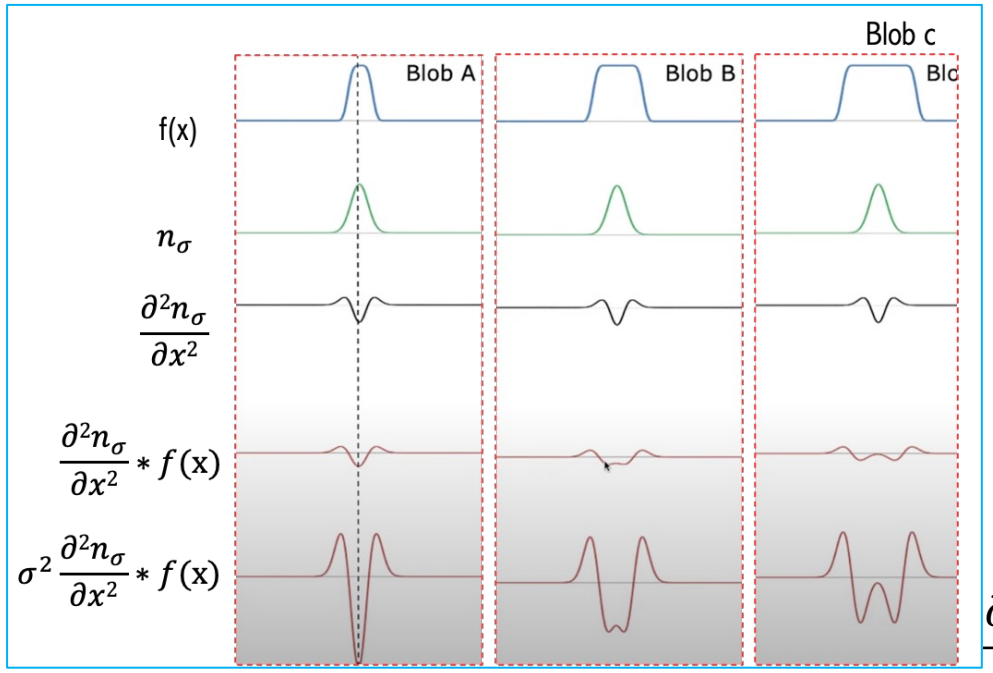




Try to increase σ →

From Prof. Shree Nayar Lecture

How to detect Blobs



$f(x)$

n_σ

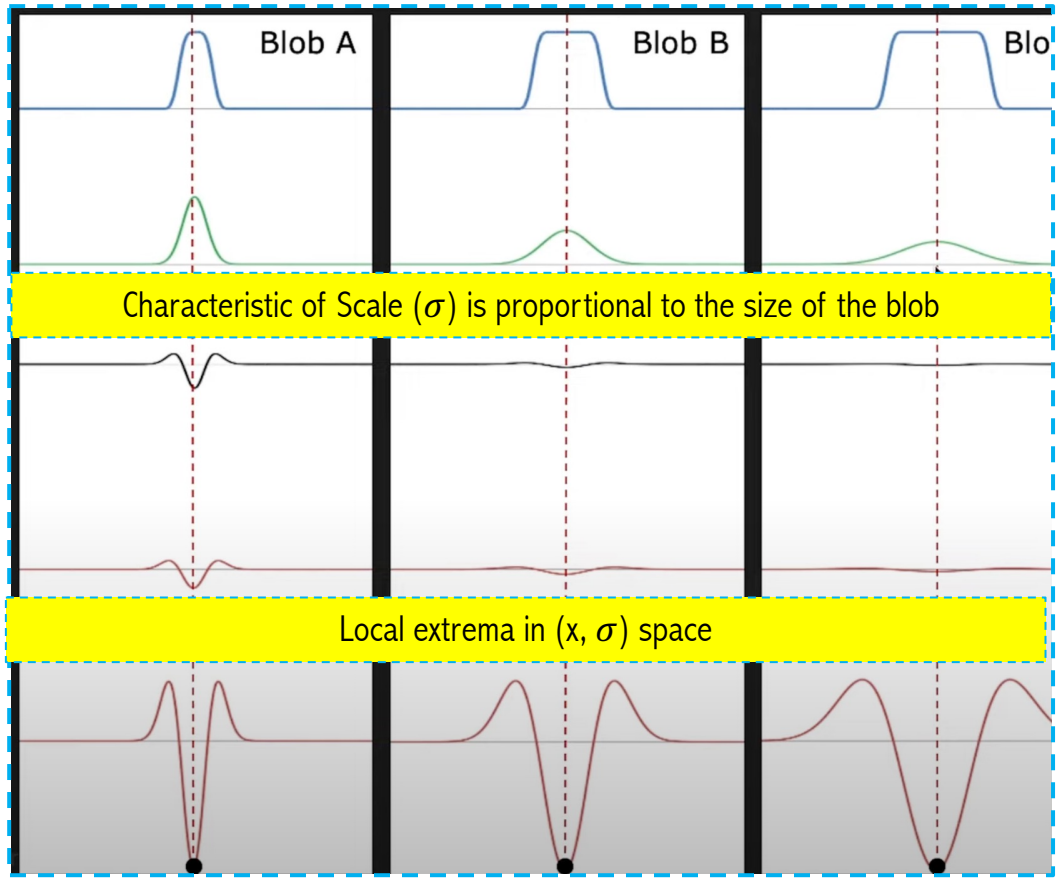
$\frac{\partial^2 n_\sigma}{\partial x^2}$

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

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

Try to increase σ

Local Extrema (x, σ) - Space Represent Blobs



x is the location of the blobs, and σ is the scale of the blob.

From Prof. Shree Nayar Lecture

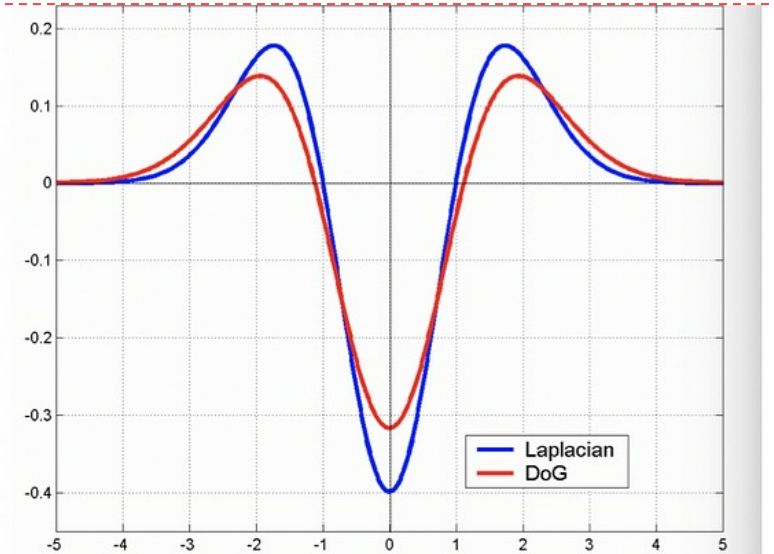
Comparison of Laplacian-of-Gaussian and Difference-of-Gaussian

$$L = \sigma^2 \left(G_{xx}(x, y, \sigma) + G_{yy}(x, y, \sigma) \right)$$

(Laplacian)

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

(Difference of Gaussians)

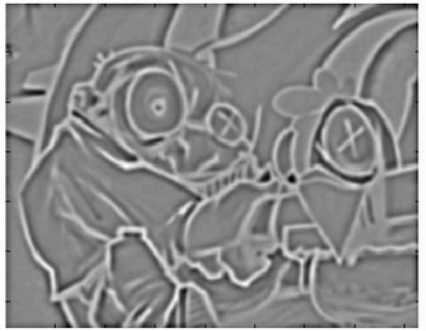


$G(x, y, k\sigma)$

-

$G(x, y, \sigma)$

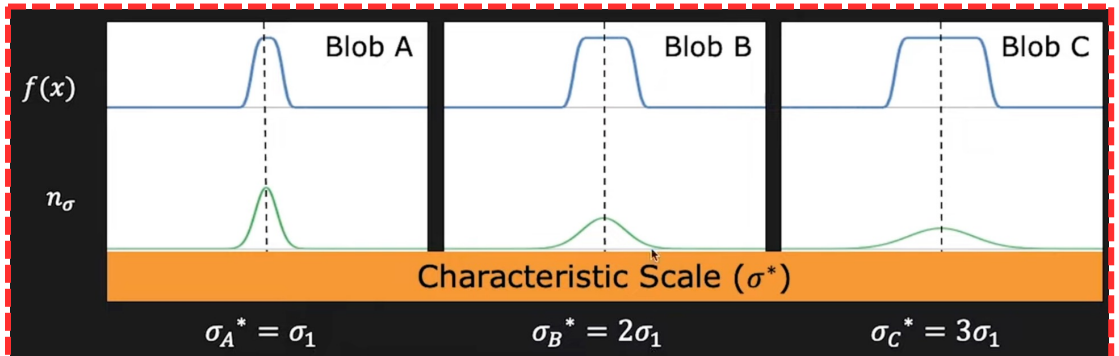
=



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

$\approx LoG$

How to detect 1D Blobs



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

Characteristic Scale $\propto$ Size of Blob

$$\frac{\text{Size of Blob A}}{\text{Size of Blob B}} = \frac{\sigma_A^*}{\sigma_B^*} \; ; \; \frac{\text{Size of Blob B}}{\text{Size of Blob C}} = \frac{\sigma_B^*}{\sigma_C^*}$$

Given: 1D signal $f(x)$

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

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

x^*: Blob Position

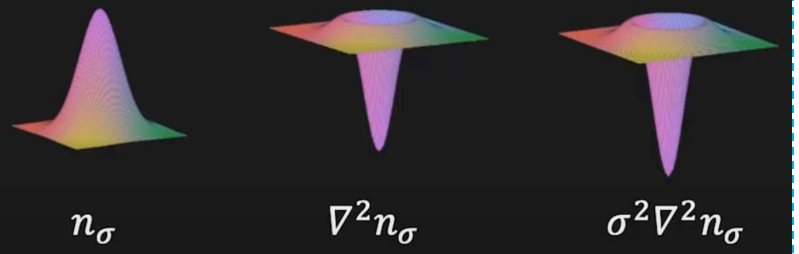
σ^*: Characteristic Scale (Blob Size)

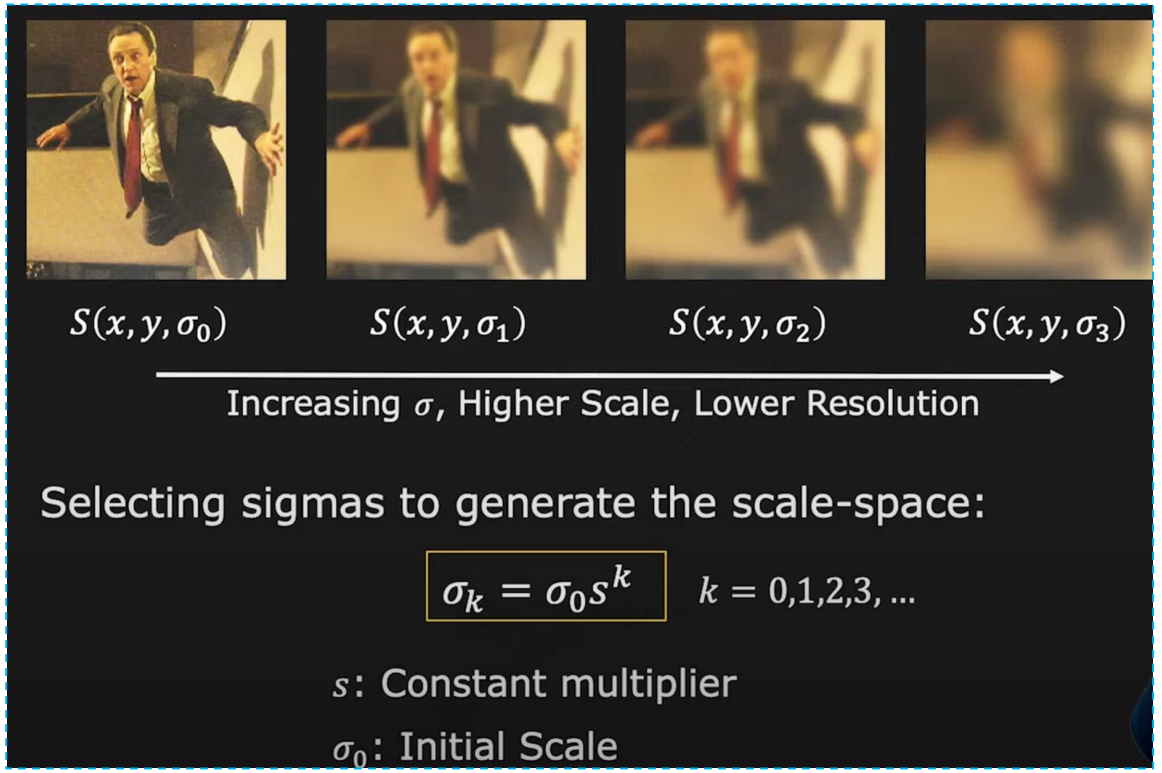
From Prof. Shree Nayar Lecture

How to detect 2D Blobs: Scale-Space

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

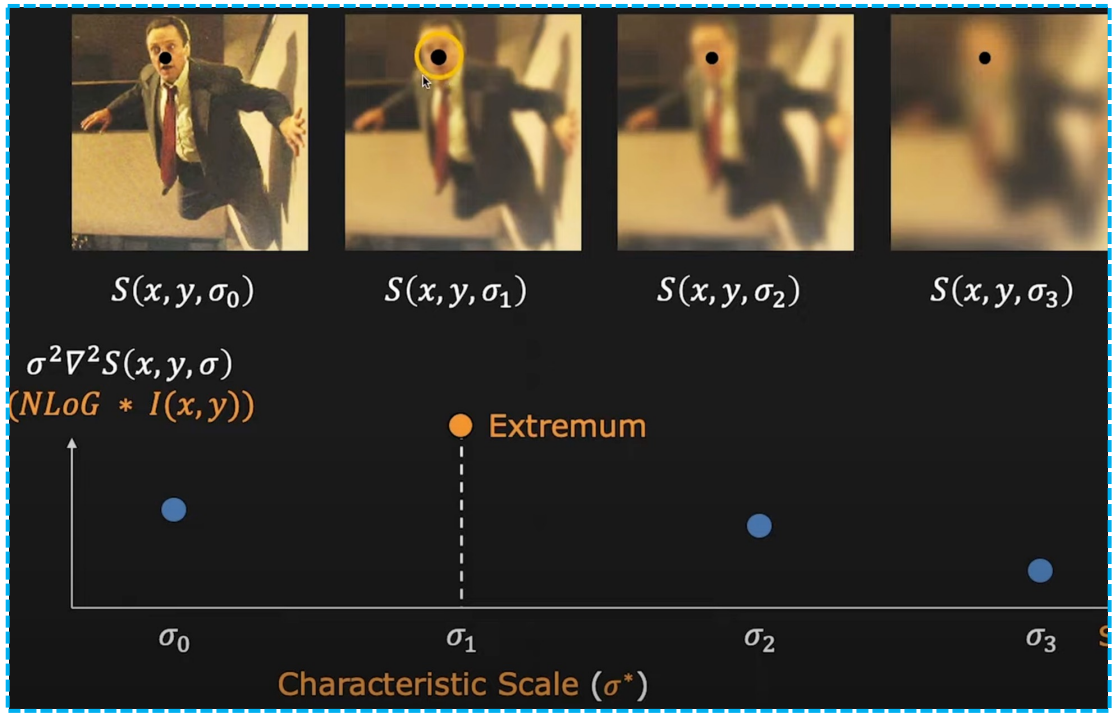
Laplacian $\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$

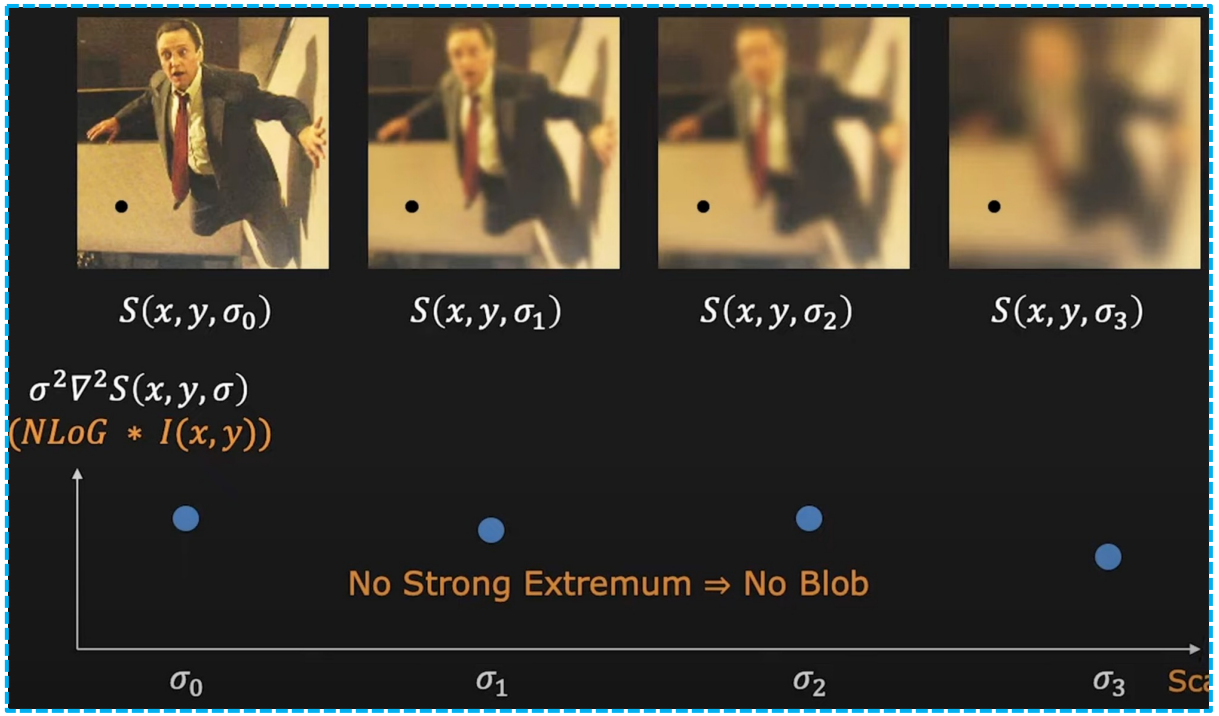




From Prof. Shree Nayar Lecture

How to detect 2D Blobs





2D Blob Detection: Summary

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

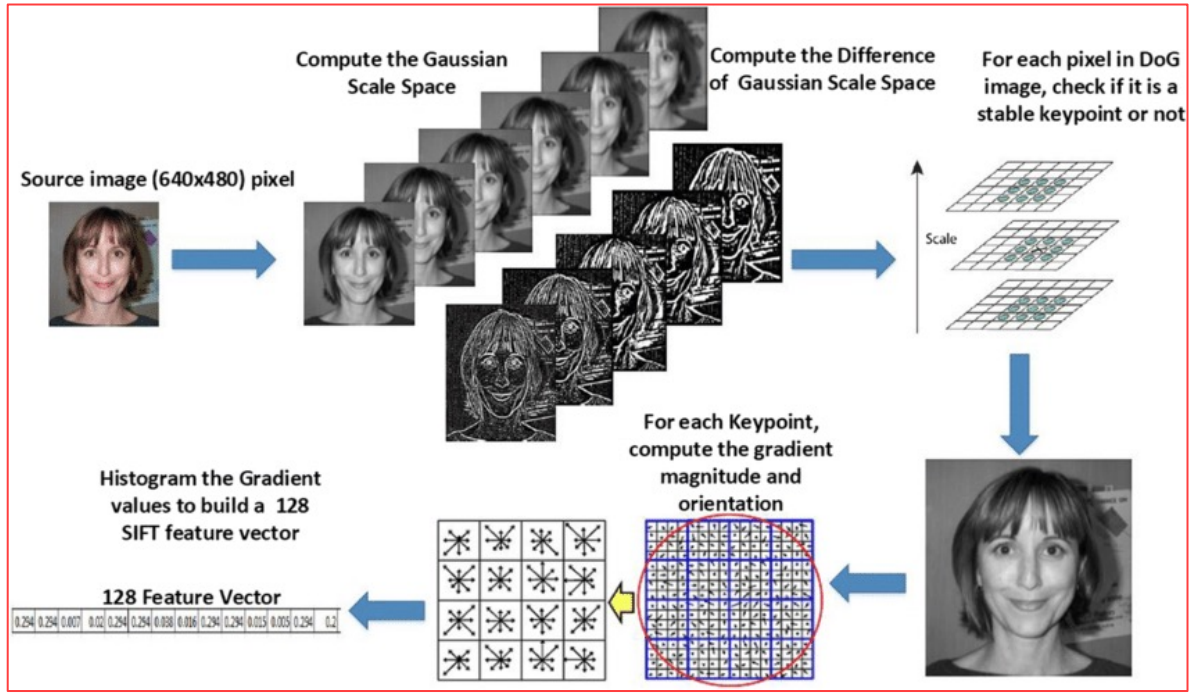
σ^*: Size of the blob

Outline

- What is image stitching/panorama image
- Edge Detector
- Blob Detector
- SIFT detector
- Image Transformation: 2D & 3D
- Image stitching/panorama Techniques

Bird's-eye view of SIFT

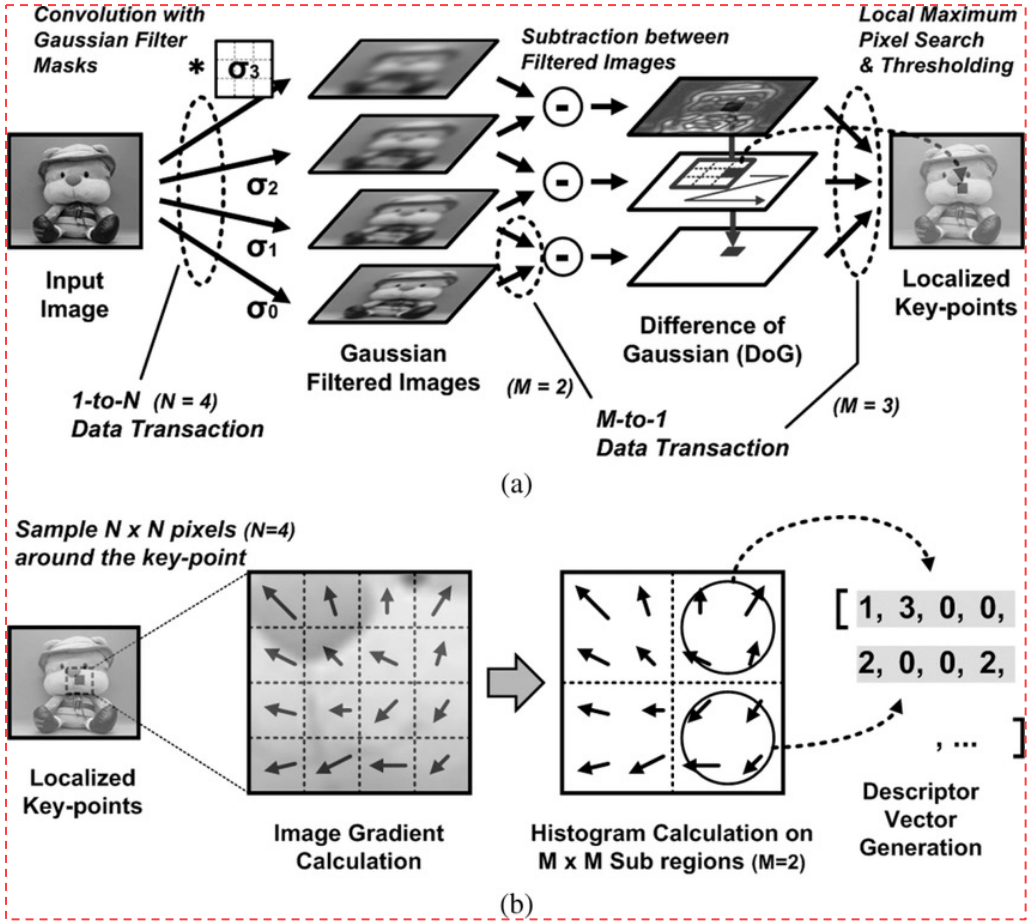
- Scale-space extrema detection
- Keypoint localization and filtering
- Orientation assignment
- Generation of vector representation for interest points



https://www.researchgate.net/publication/331185020_A_novel_SIFT_architecture_and_ASIC_implementation_for_real_time_SOC_application/figures?lo=1

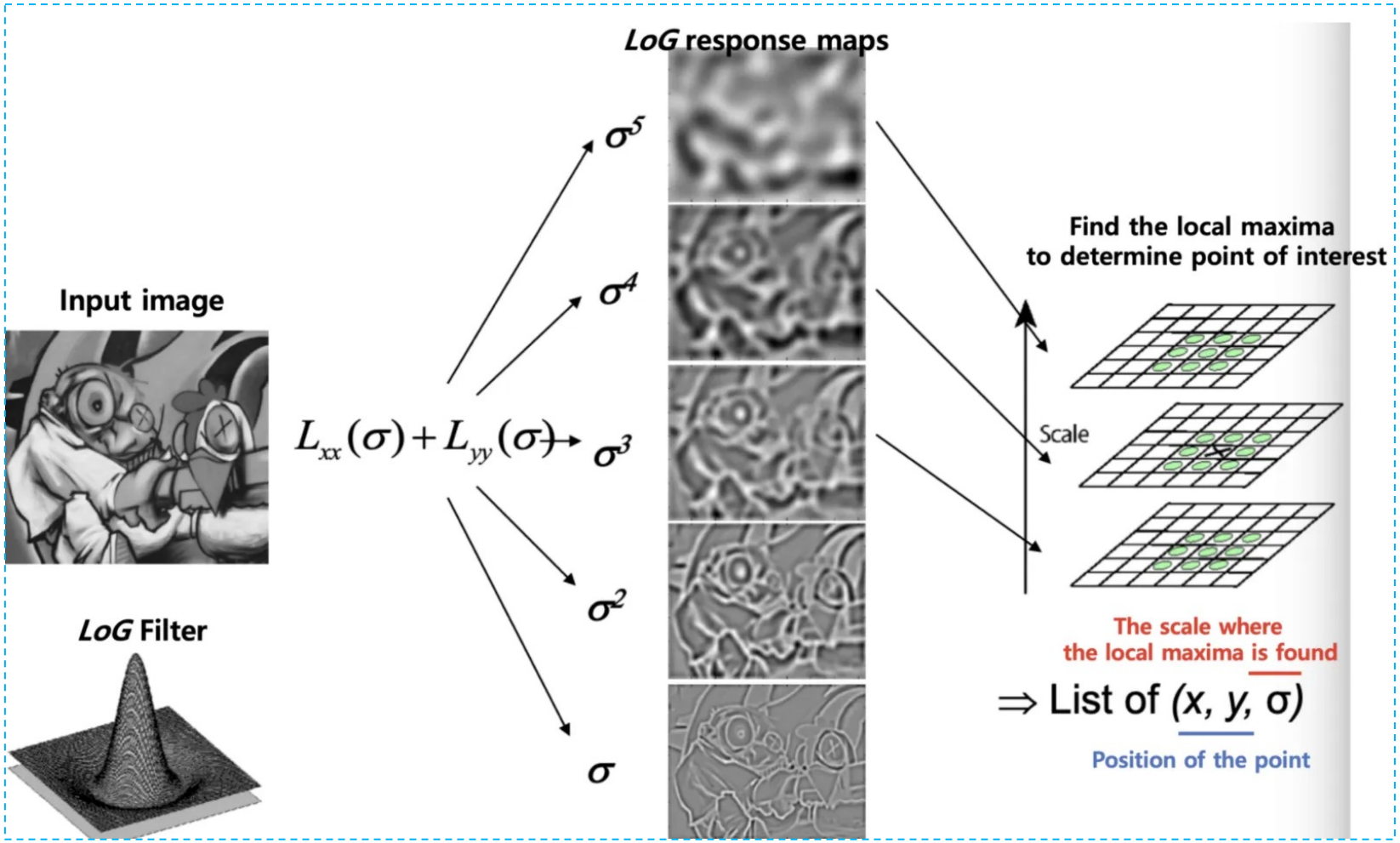
Bird's-eye view of SIFT

- Scale-space extrema detection
- Keypoint localization and filtering
- Orientation assignment
- Generation of vector representation for interest points



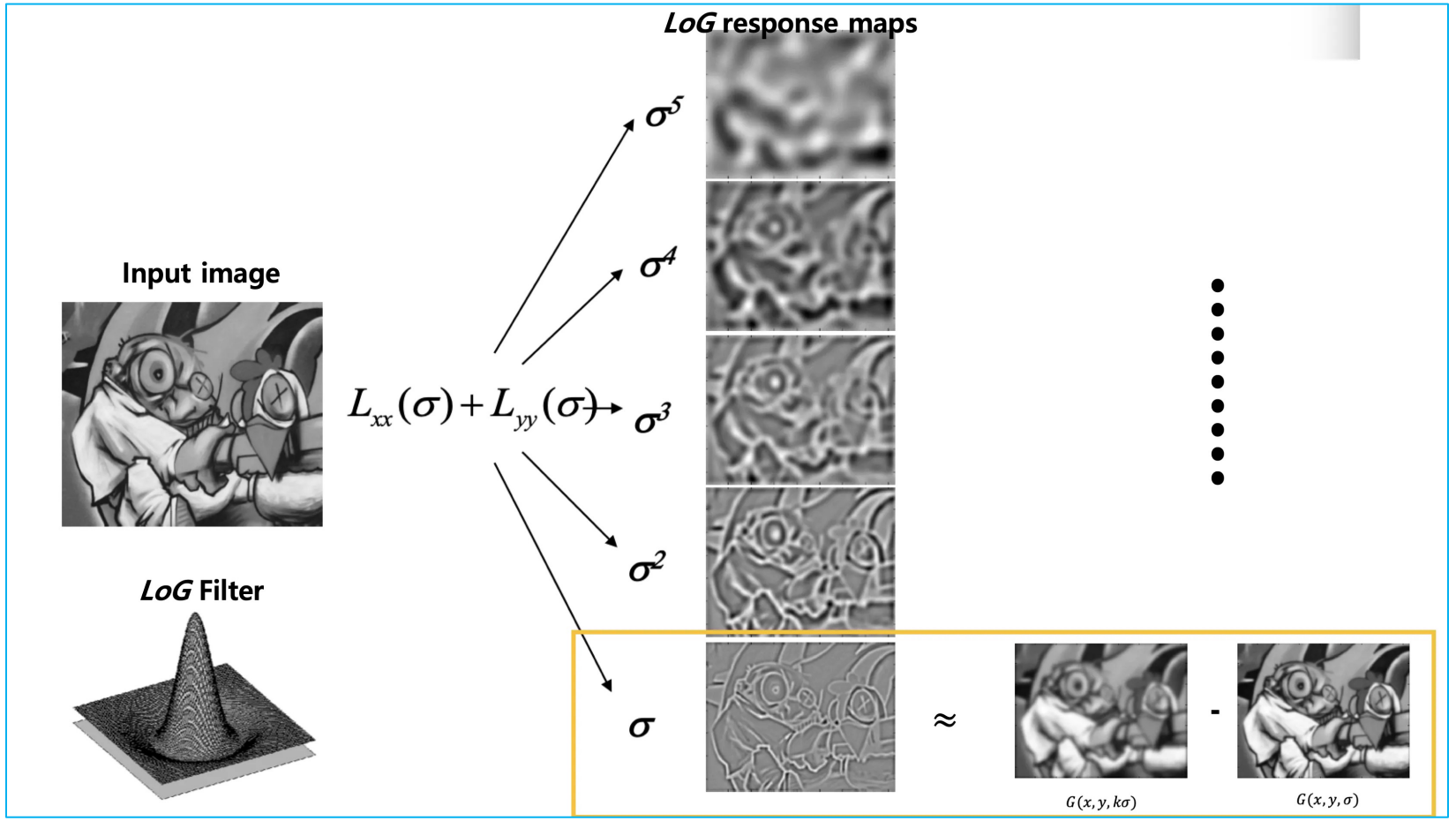
https://www.researchgate.net/publication/331185020_A_novel_SIFT_architecture_and_ASIC_implementation_for_real_time_SOC_application/figures?lo=1

Scale-space extrema detection



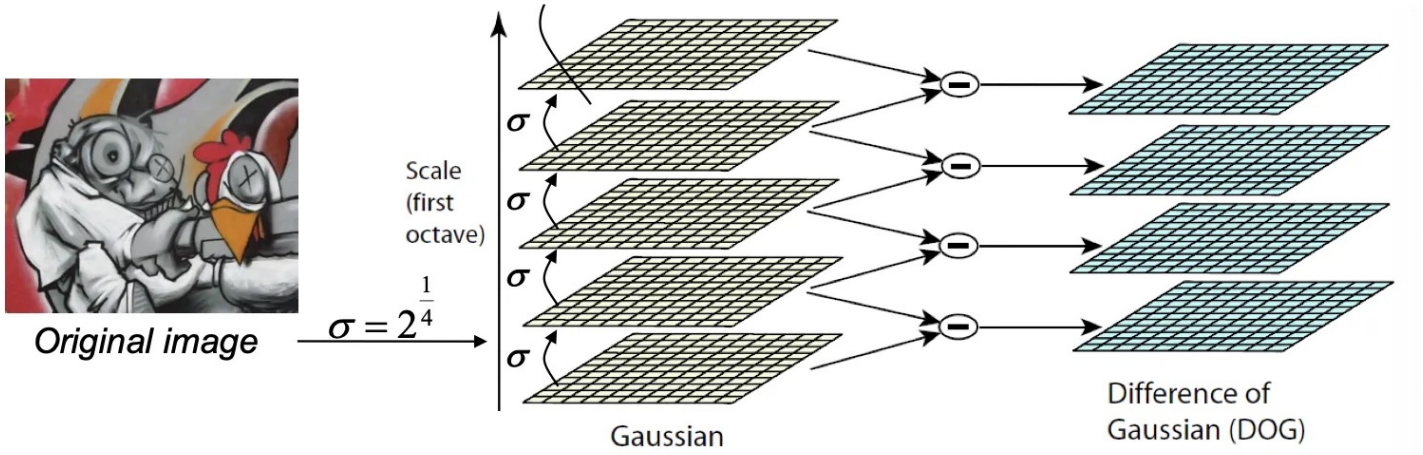
Automatic scale selection with Laplacian of Gaussian

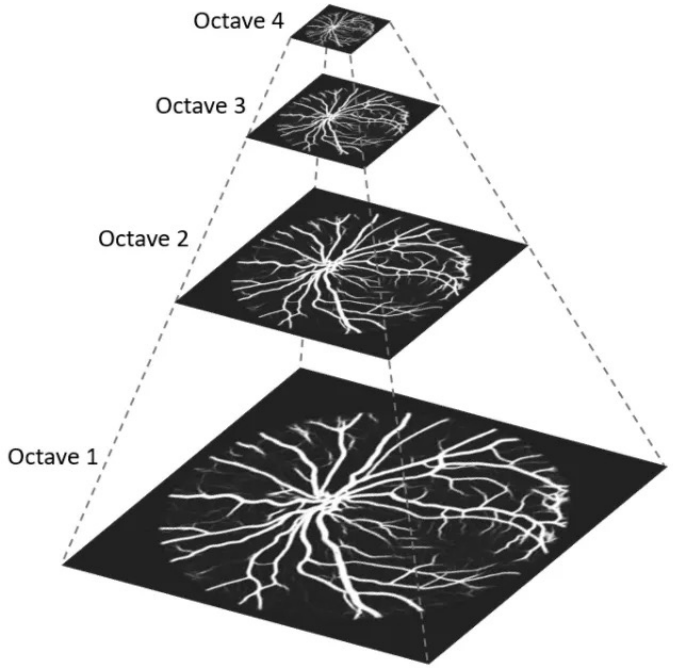
Scale-space extrema detection



Applying DoG instead of computing LoG response map in each scale space

Scale-space extrema detection

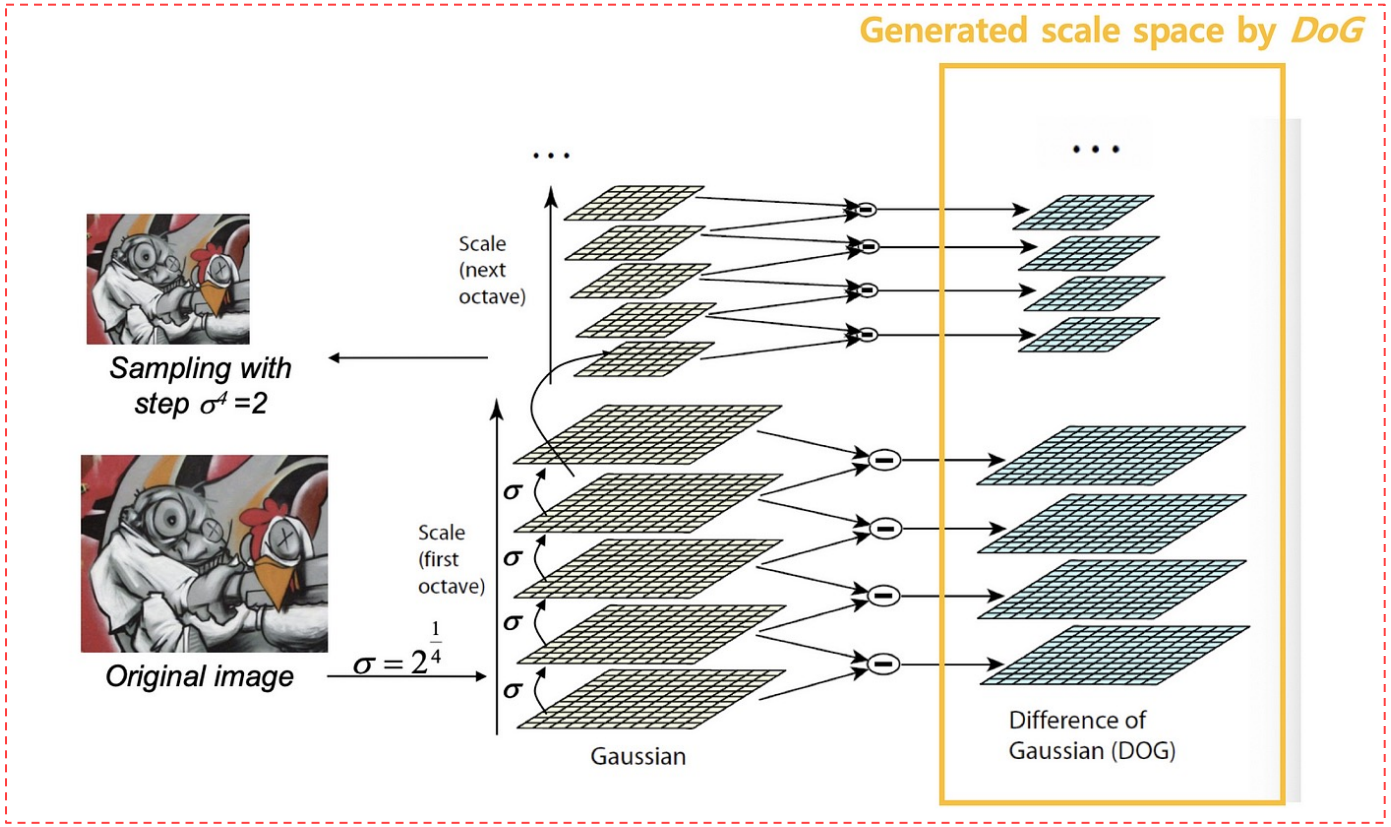


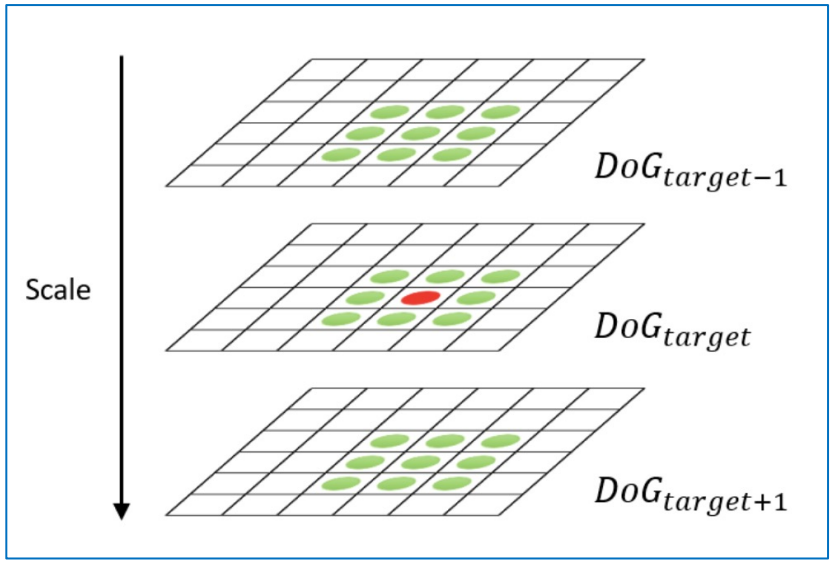


However, just doing this does not guarantee that SIFT is scale-invariant because **the algorithm currently only looks at the image in one fixed scale.** In order to observe the image from many different scales, *SIFT* integrates the idea of *Gaussian Pyramid* into this.

Gaussian pyramid is a multi-scale representation of an image containing a properly re-sized(re-scaled) version of the original image by sampling with Gaussian blurring in order to avoid the aliasing issue

Scale-space extrema detection

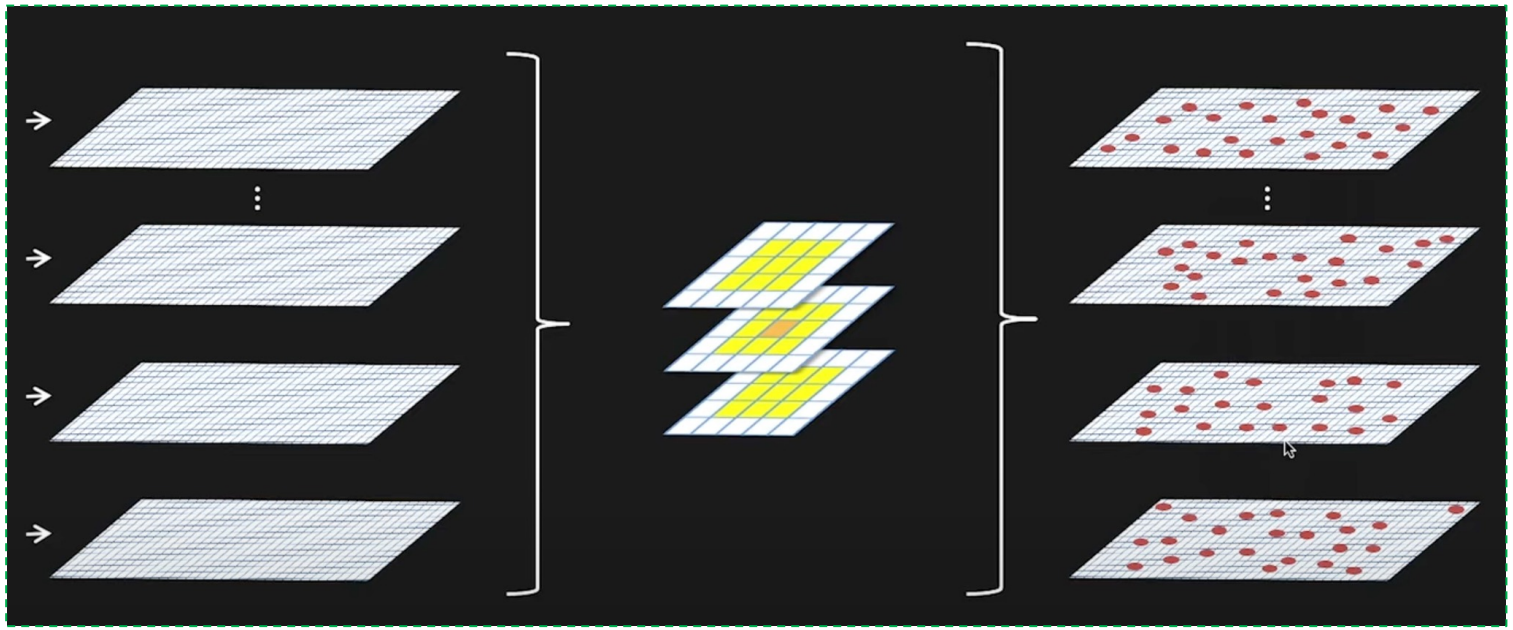




Local extrema detection

Local extrema (either maxima or minima) are detected by comparing a pixel (*red circle*) to its 26 neighbors (*green*) in *3×3* window at the current and adjacent (above and below) scales. More specifically, only a point ***DoG(x, y, σ)*** is selected if it is larger or smaller than all of its neighbors. The list of detected extremum ***(x, y, σ)*** is the **keypoint candidates** which will be processed later on in *SIFT* algorithm.

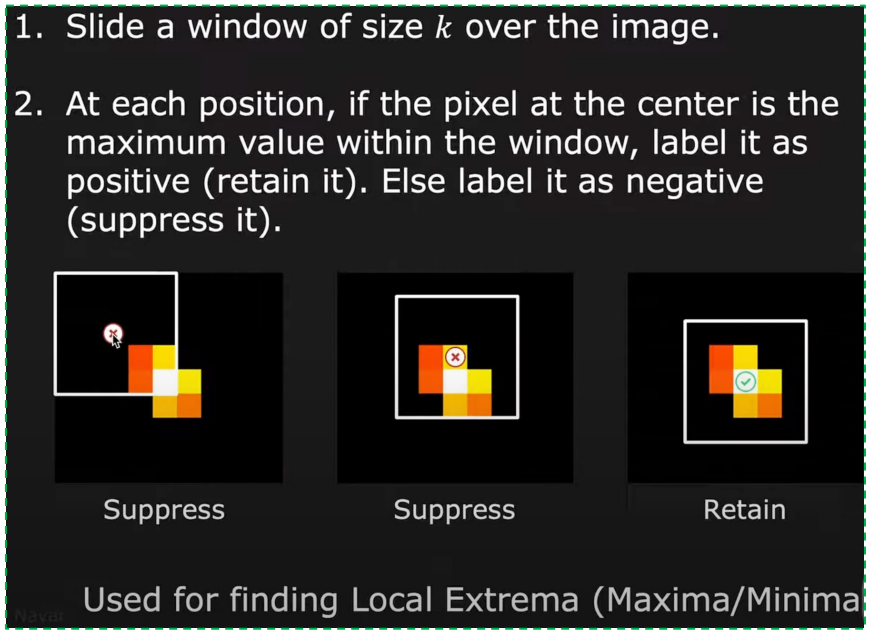
Local Non-Maximal Suppression



Differences of Gaussian

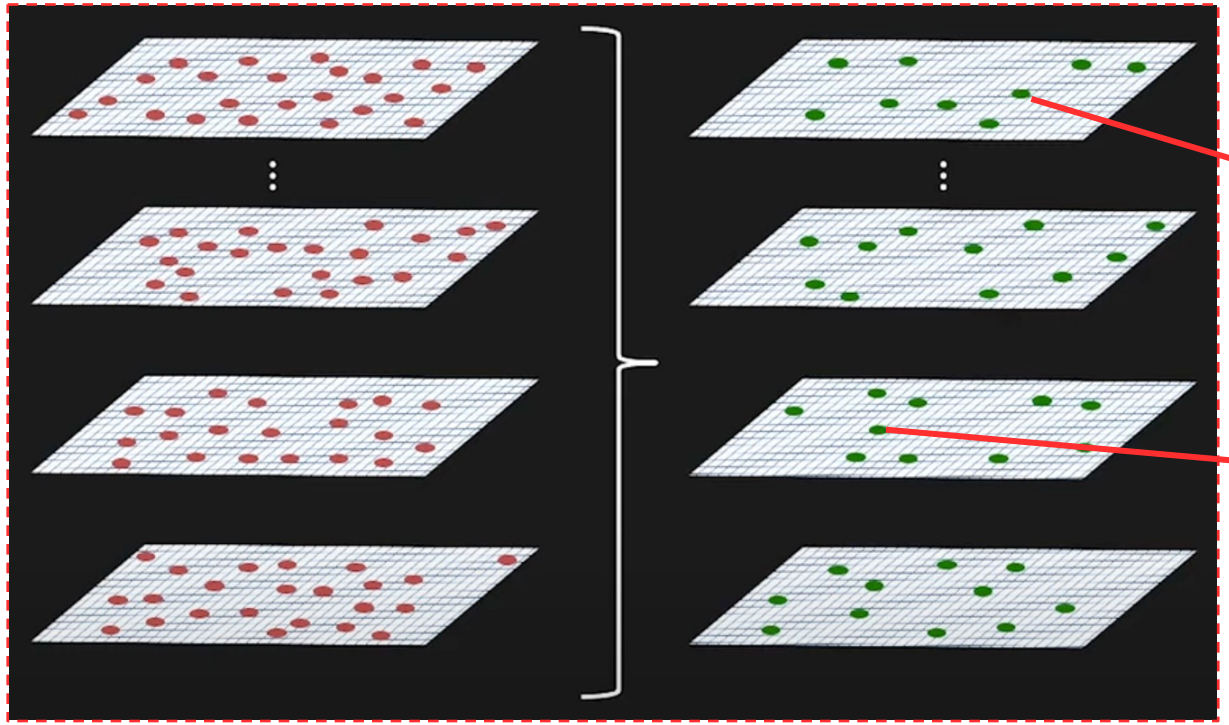
Find extremum in vevy 3x3x3 grid

Interest Point Candidates (includes weak extrema)



Non-maximal suppression

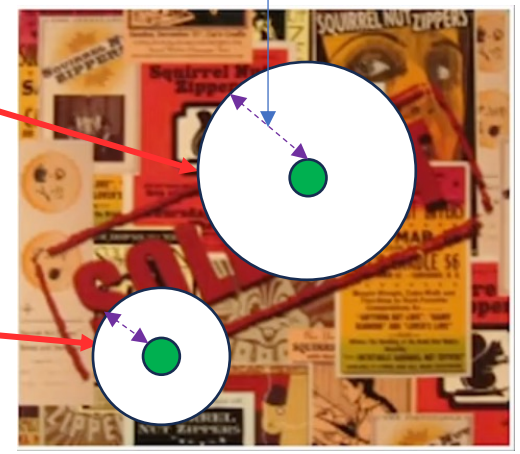
Keypoint localization and filtering



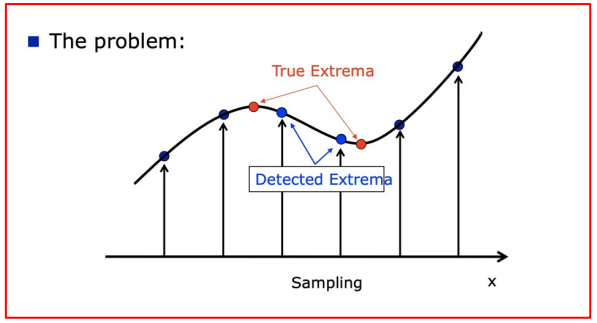
Interest Point Candidates
(includes weeak extrema)

Interest Point Candidates
(remove weak extrema)

Scale: σ

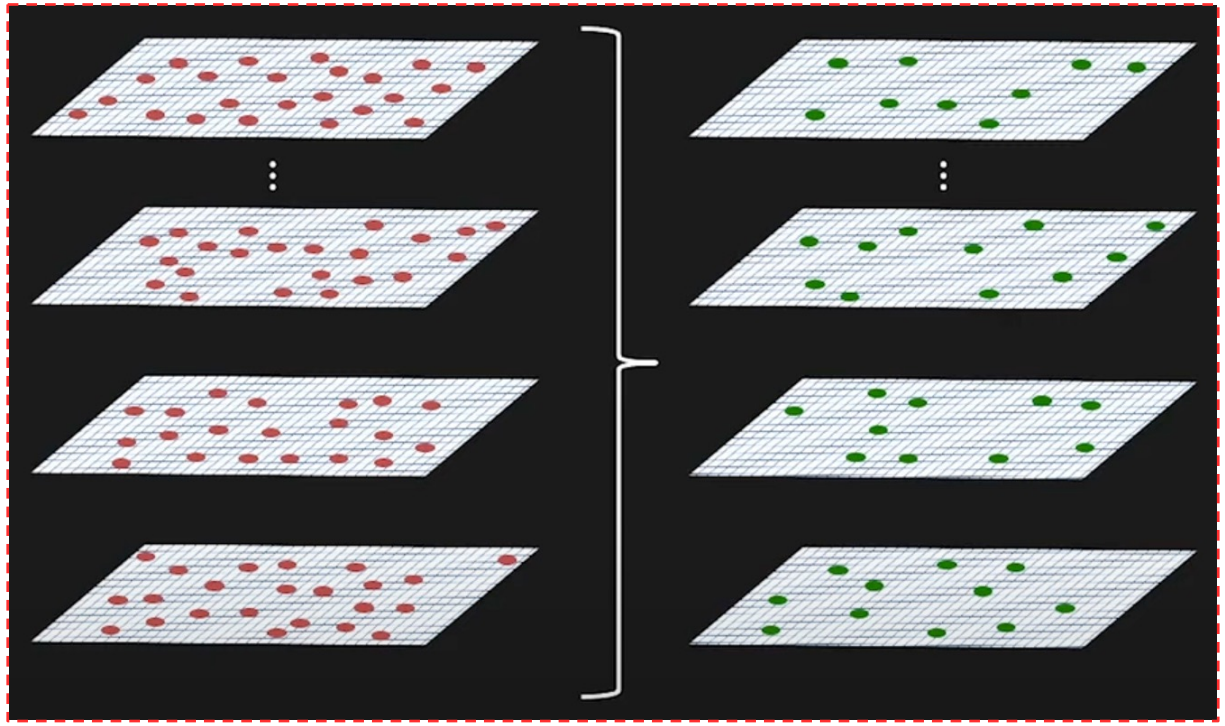


Blobs results



Taylor series expansion of scale space to get a more accurate location of extrema, and if the intensity at this extrema is less than a threshold value (0.03 as per the paper), it is rejected

Keypoint localization and filtering



Interest Point Candidates (includes weeak extrema)

Interest Point Candidates (remove weak extrema)

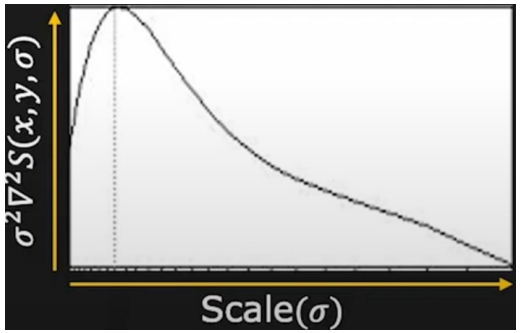
Interest Point Visualization

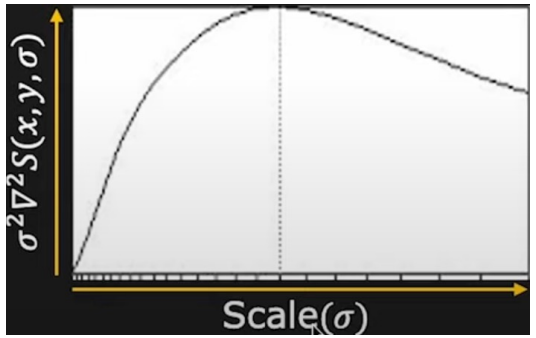
Sift Detector Example

```
import numpy as np
import cv2
import imageio
img = imageio.imread('http://www.ic.unicamp.br\
/~helio/imagens_registro/foto1A.jpg')

gray= cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
sift = cv2.SIFT_create()
kp = sift.detect(gray,None)
img=cv2.drawKeypoints(gray,kp,img, flags=cv2.DRAW_MATCHES_FLAGS_DRAW_RICH_KEYPOINTS)
cv2.imwrite('sift_keypoints.jpg',img)
```

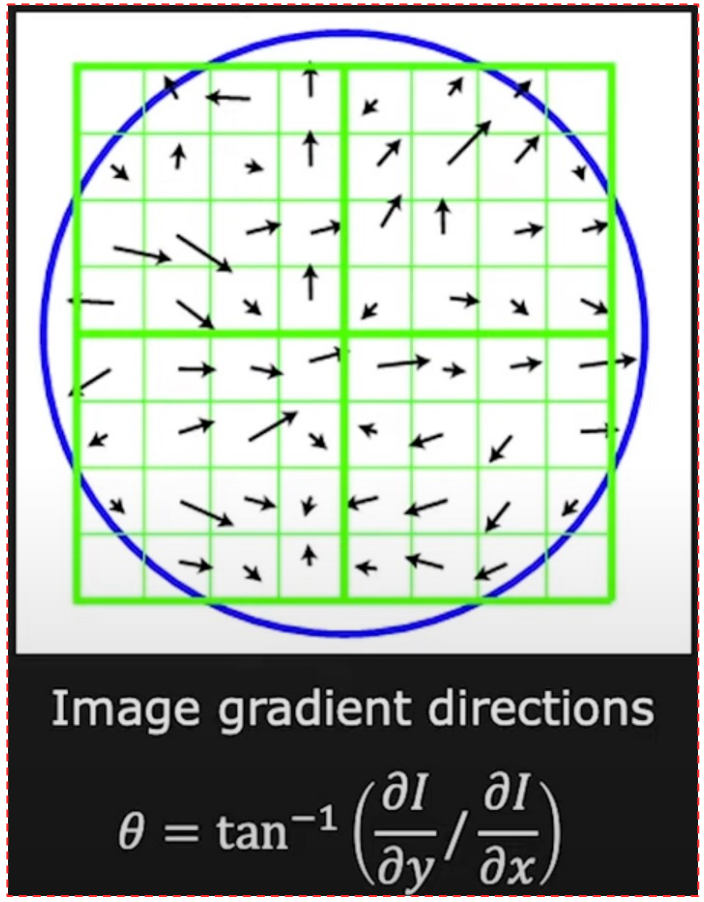
SIFT Scale Invariance

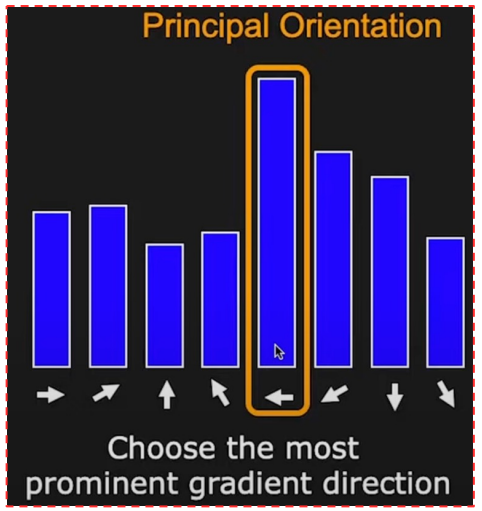


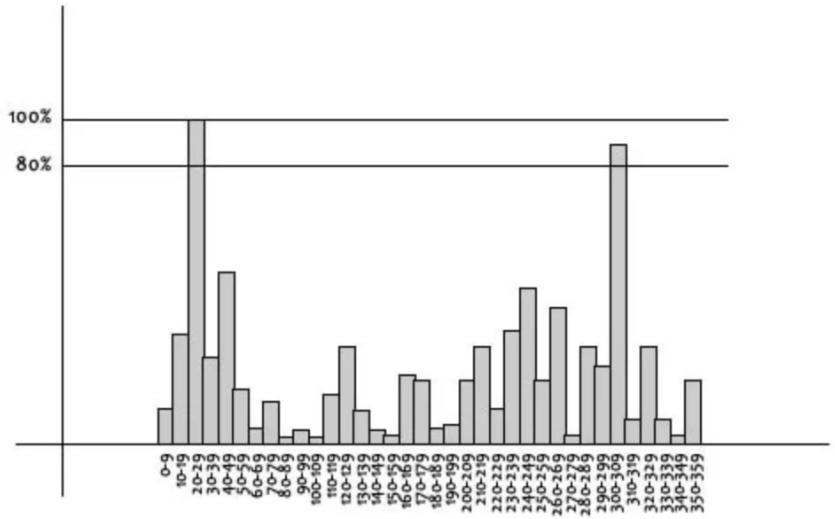


$\frac{\sigma_1^*}{\sigma_2^*}$: Ratio of Blob Sizes

Orientation Assignment



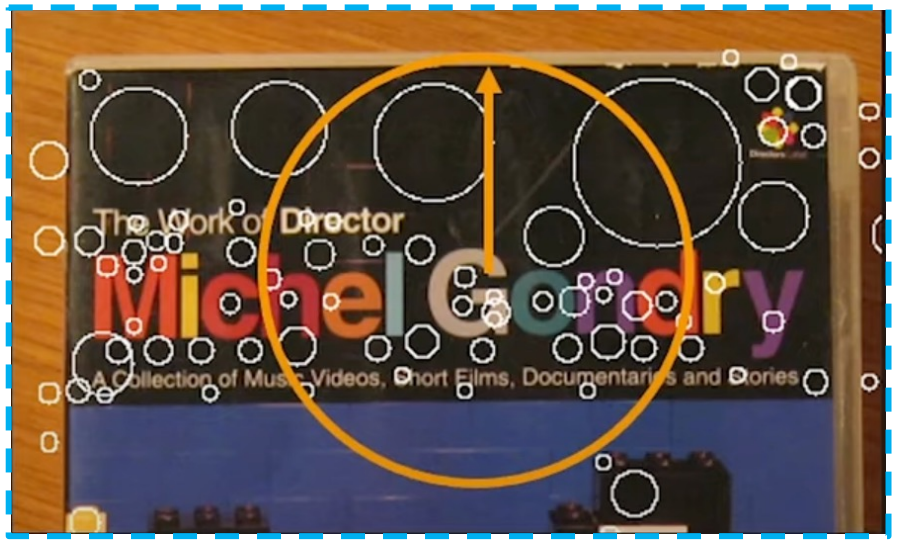


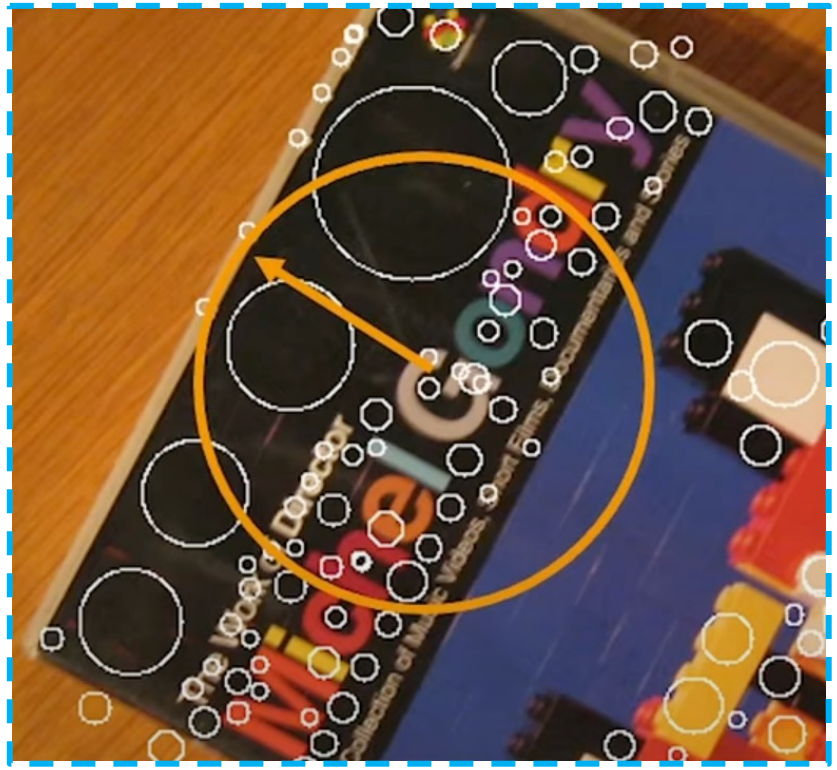


Histogram of oriented gradient

An orientation histogram with 36 bins covering 360 degrees is created

Orientation Assignment

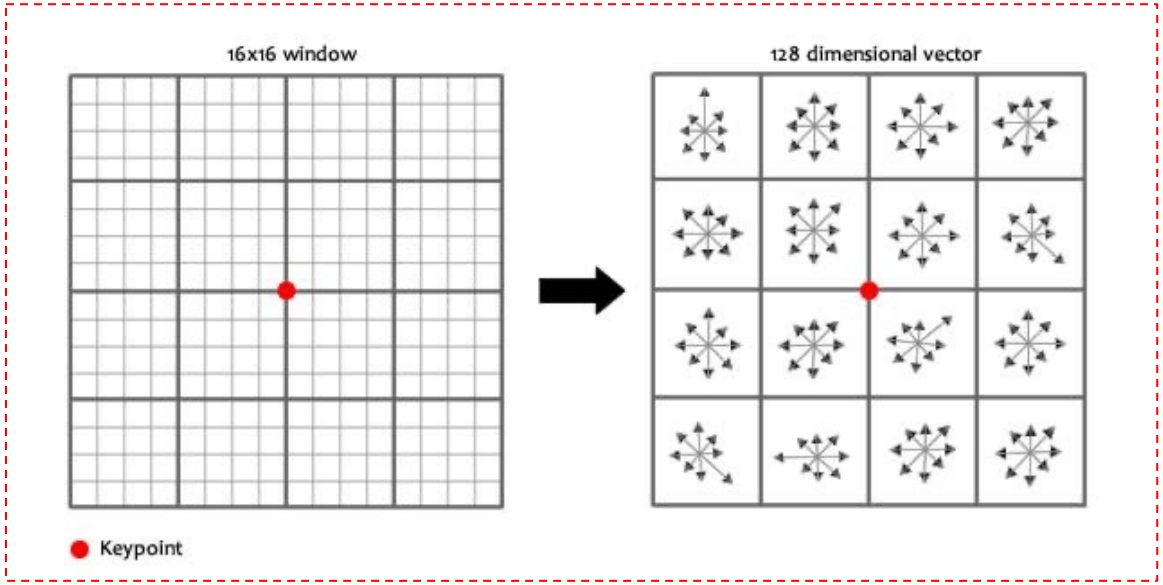


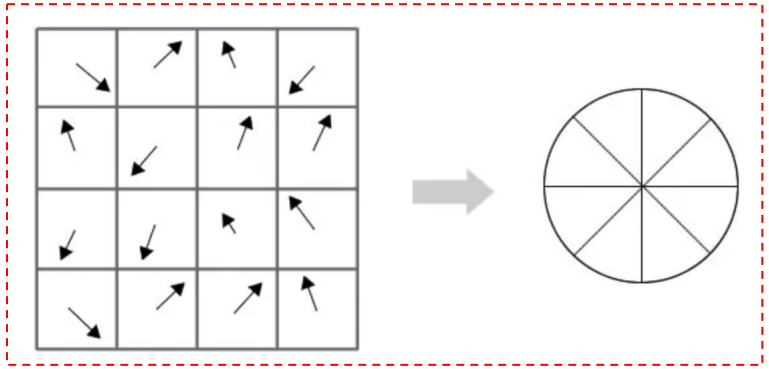


Use the pricipal oritentation to undo the roration

Keypoint descriptor

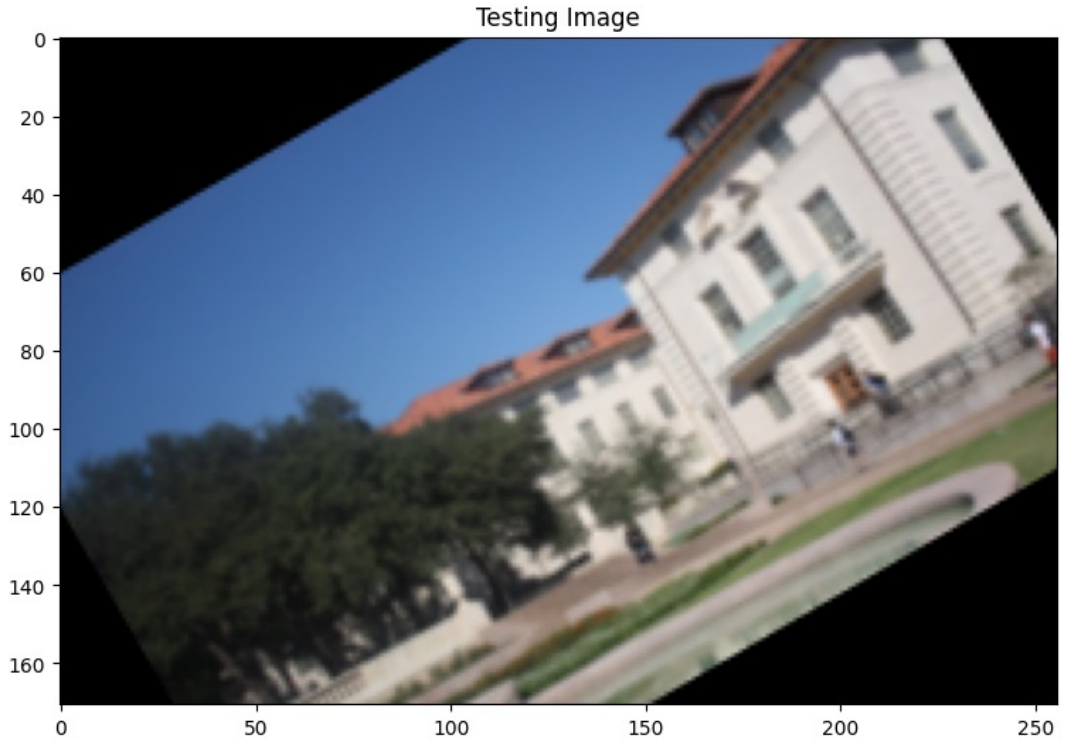
a 16x16 window around the keypoint is taken. It is divided into 16 sub-blocks of 4x4 size.



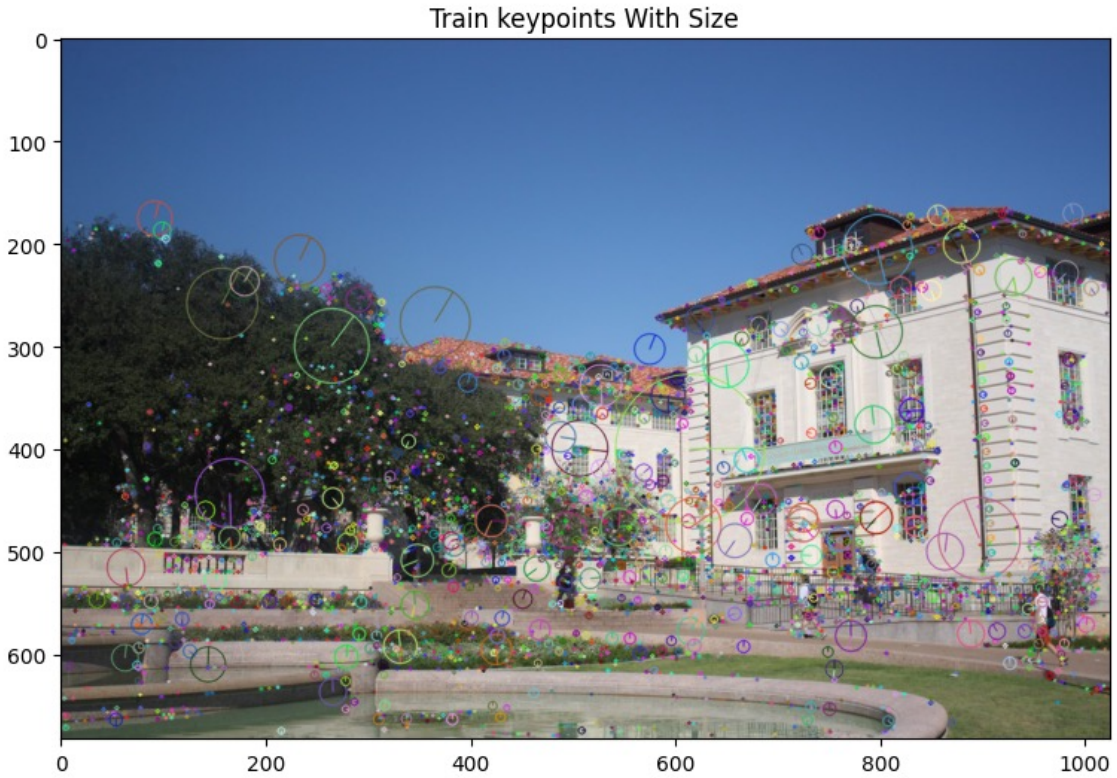


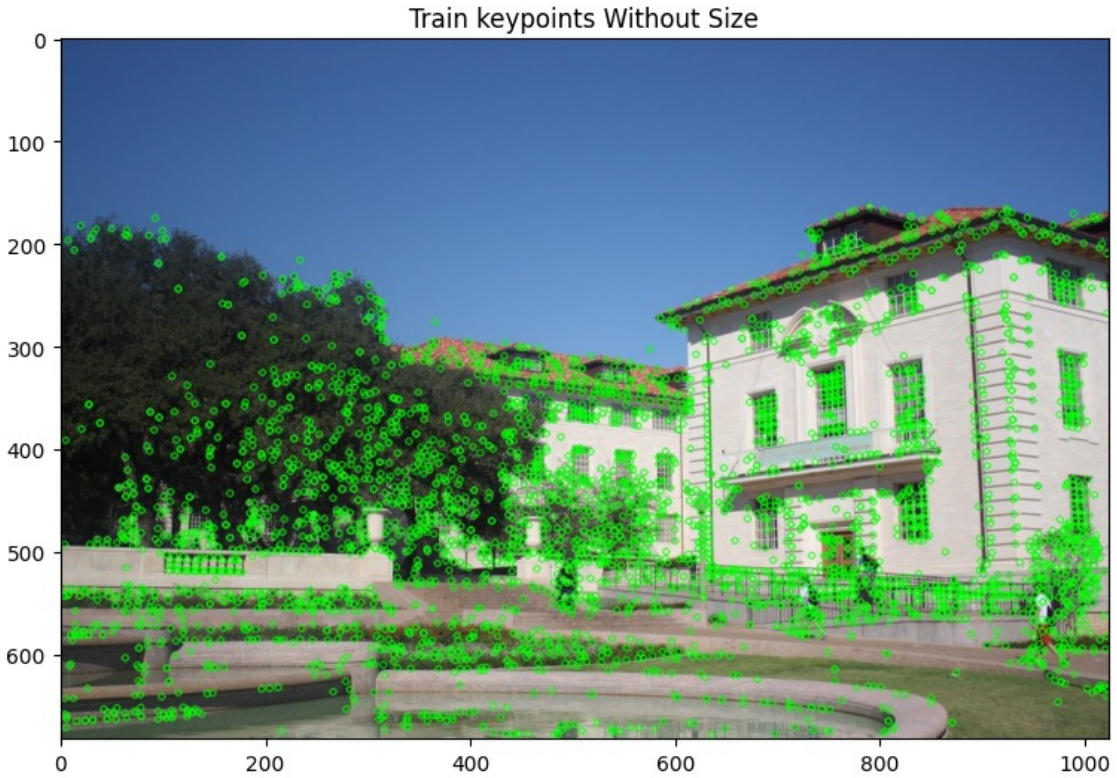
For each sub-block, 8 bin orientation histogram is created.

SIFT Matching



SIFT Matching





SIFT Matching

Let $H_1(k)$ and $H_2(k)$ be two arrays of data of length N.

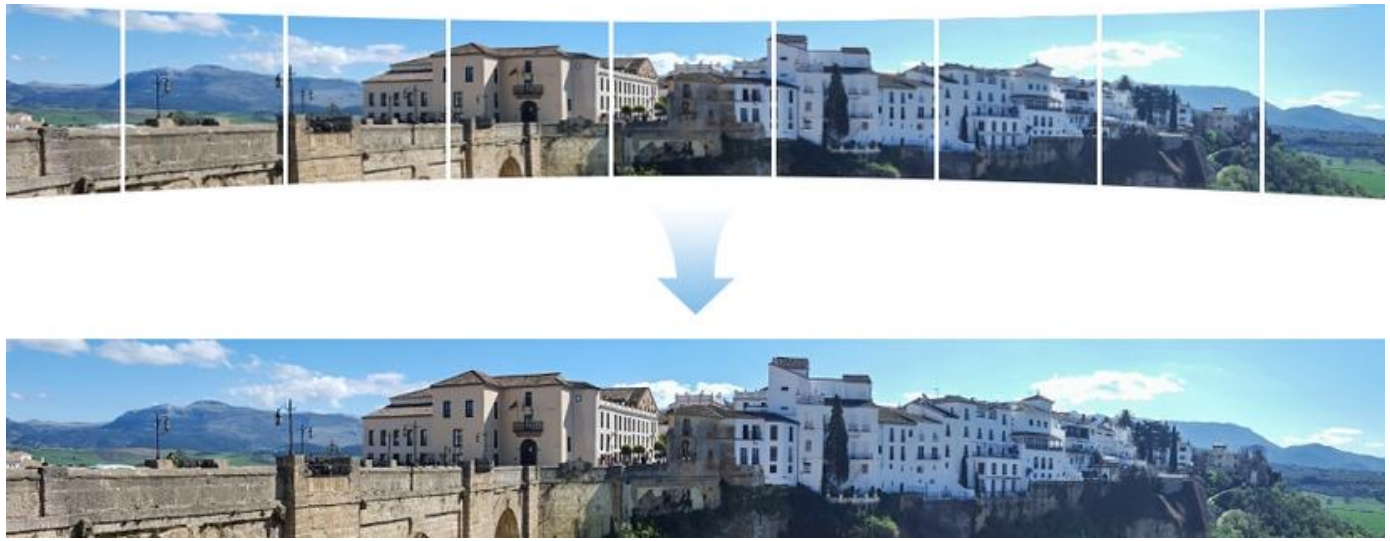
L2 Distance:

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

Smaller the distance metric, better the match.

Perfect match when $d(H_1, H_2) = 0$

Image Stitching



- 2x2 Image Transformation
- 3x3 Image Transformation
- Compute Homography
- Dealing with Outlier: RANSAC
- Warping and Blending Images

Outline

- What is image stitching/panorama image
- Edge Detector
- Blob Detector
- SIFT detector
- Image Transformation: 2D & 3D
- Image stitching/panorama Techniques

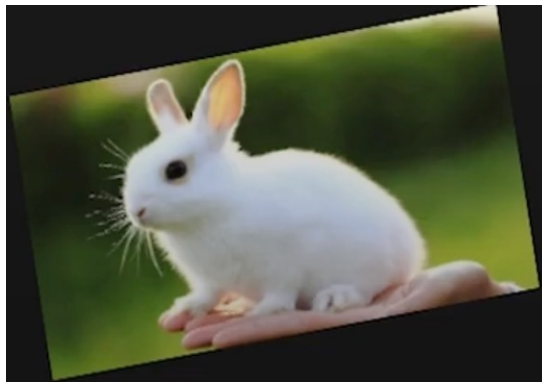
Image Manipulation

$$G(x,y) = Tr(f(x,y)$$

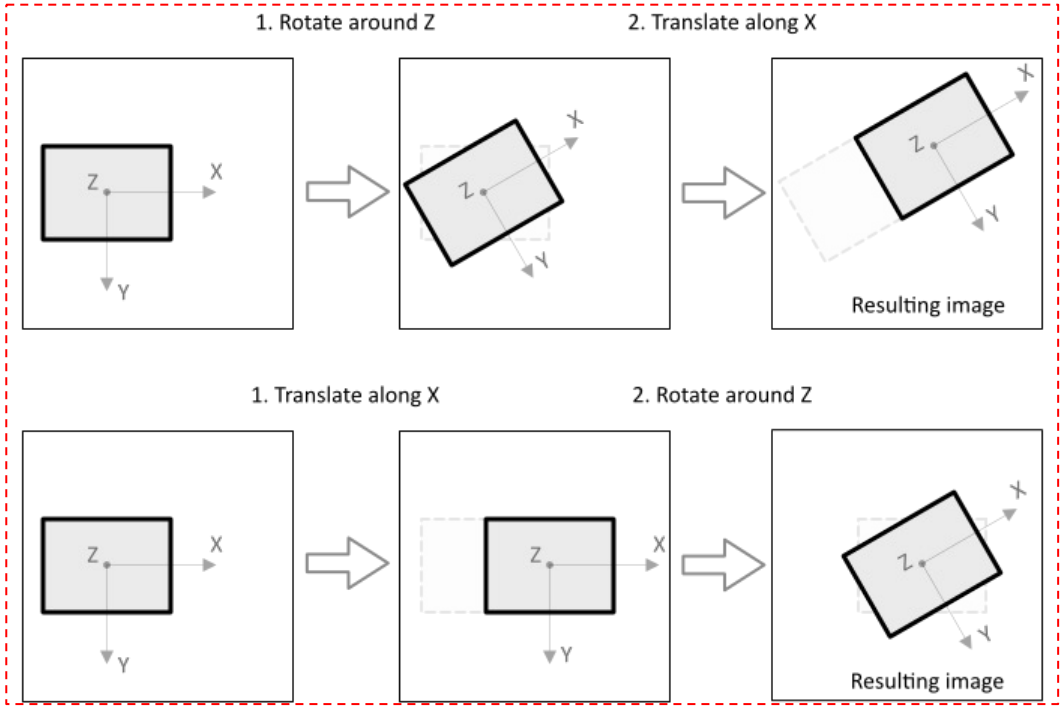
Image filtering: change range(brightness)

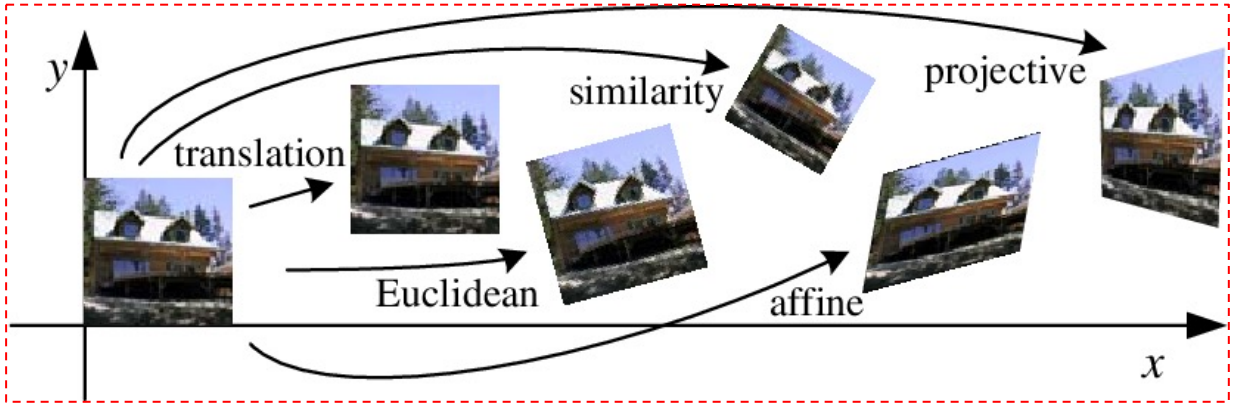
$$G(x,y) = Td(f(x,y)$$

Image filtering: change domain(location)



Global Warping/Transformation



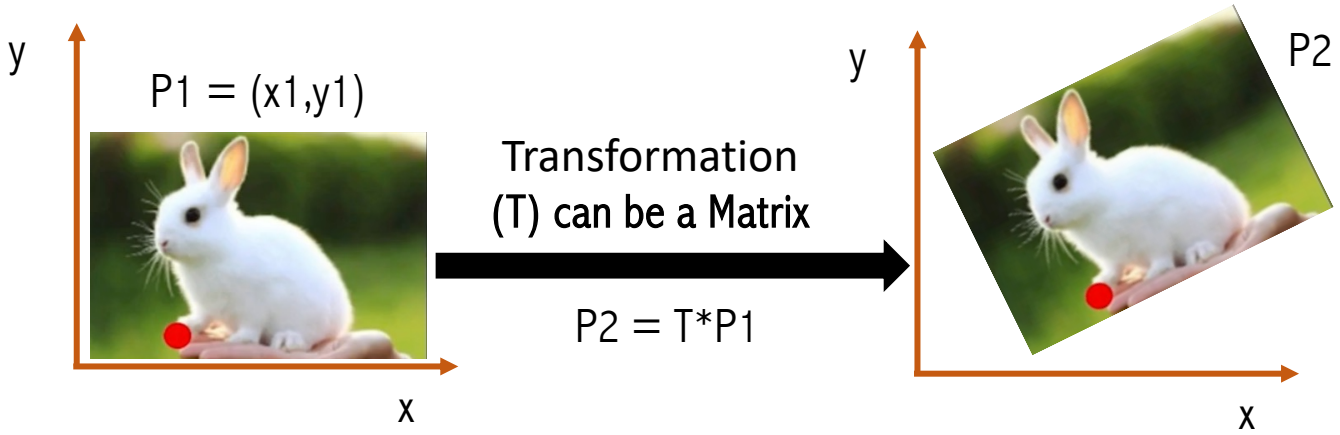


$$\begin{bmatrix} X_{scaled} \\ Y_{scaled} \\ 1 \end{bmatrix} = \begin{bmatrix} Scale_x & 0 & 0 \\ 0 & Scale_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} X_{translated} \\ Y_{translated} \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & D_x \\ 0 & 1 & D_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}$$

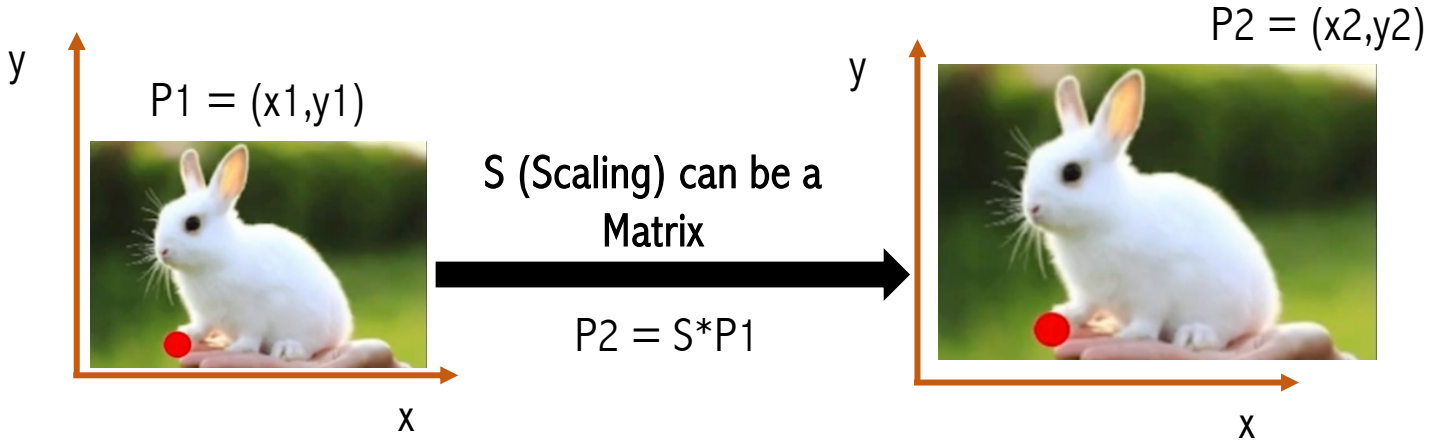
$$\begin{bmatrix} X_{rotated} \\ Y_{rotated} \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}$$

Transformation and Scaling (2D)



$$\begin{bmatrix} x2 \\ y2 \end{bmatrix} = T \begin{bmatrix} x1 \\ y1 \end{bmatrix}$$

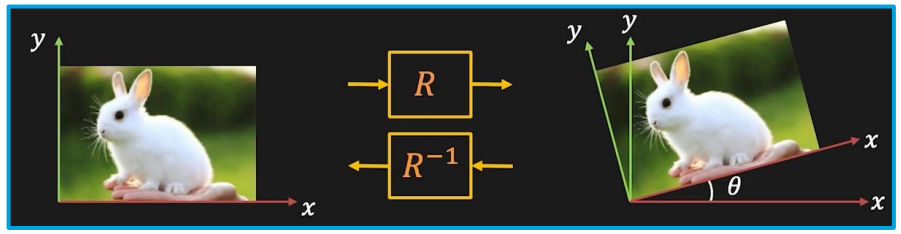
$$\begin{bmatrix} x2 \\ y2 \end{bmatrix} = \begin{bmatrix} a11 & a12 \\ a21 & a22 \end{bmatrix} \begin{bmatrix} x1 \\ y1 \end{bmatrix}$$



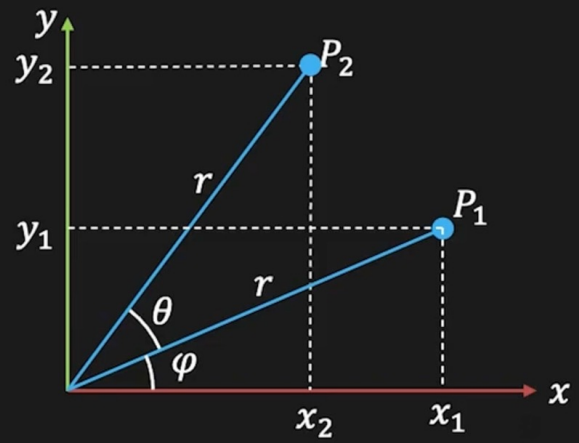
$$\begin{bmatrix} x2 \\ y2 \end{bmatrix} = S \begin{bmatrix} x1 \\ y1 \end{bmatrix}$$

$$\begin{bmatrix} x2 \\ y2 \end{bmatrix} = \begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix} \begin{bmatrix} x1 \\ y1 \end{bmatrix}$$

2D Rotation



$$x_2 = x_1 cos\theta - y_1 sin\theta$$
$$y_2 = x_1 sin\theta + y_1 cos\theta$$
$$\begin{bmatrix} x_2 \\ y_2 \end{bmatrix} = R \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} = \begin{bmatrix} cos\theta & -sin\theta \\ sin\theta & cos\theta \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \end{bmatrix}$$

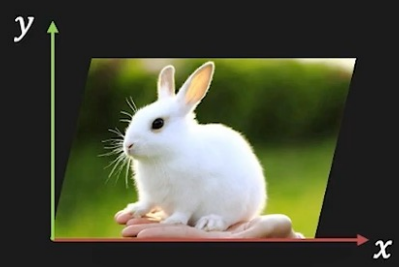


$$x_1 = r\cos(\varphi)$$
$$y_1 = r\sin(\varphi)$$

$$x_2 = r\cos(\varphi + \theta)$$
$$x_2 = r\cos\varphi\cos\theta - r\sin\varphi\sin\theta$$
$$x_2 = x_1\cos\theta - y_1\sin\theta$$

$$y_2 = r\sin(\varphi + \theta)$$
$$y_2 = r\cos\varphi\sin\theta + r\sin\varphi\cos\theta$$
$$y_2 = x_1\sin\theta + y_1\cos\theta$$

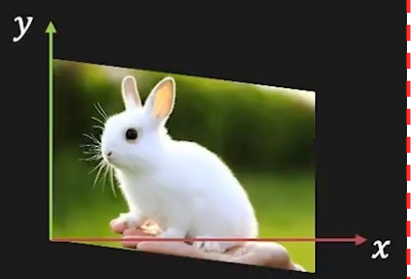
Skew and Mirror



Horizontal Skew:

$x_2 = x_1 + m_x y_1$

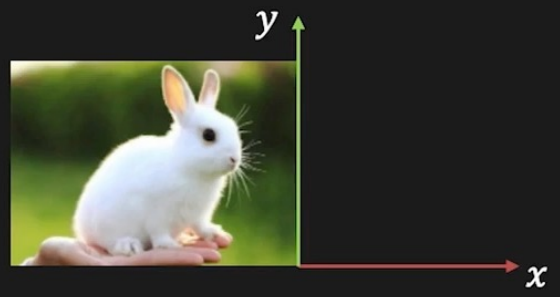
$y_2 = y_1$



Vertical Skew:

$x_2 = x_1$

$y_2 = m_y x_1 + y_1$



Mirror about Y-axis:

$x_2 = -x_1$

$y_2 = y_1$

$M_y = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$

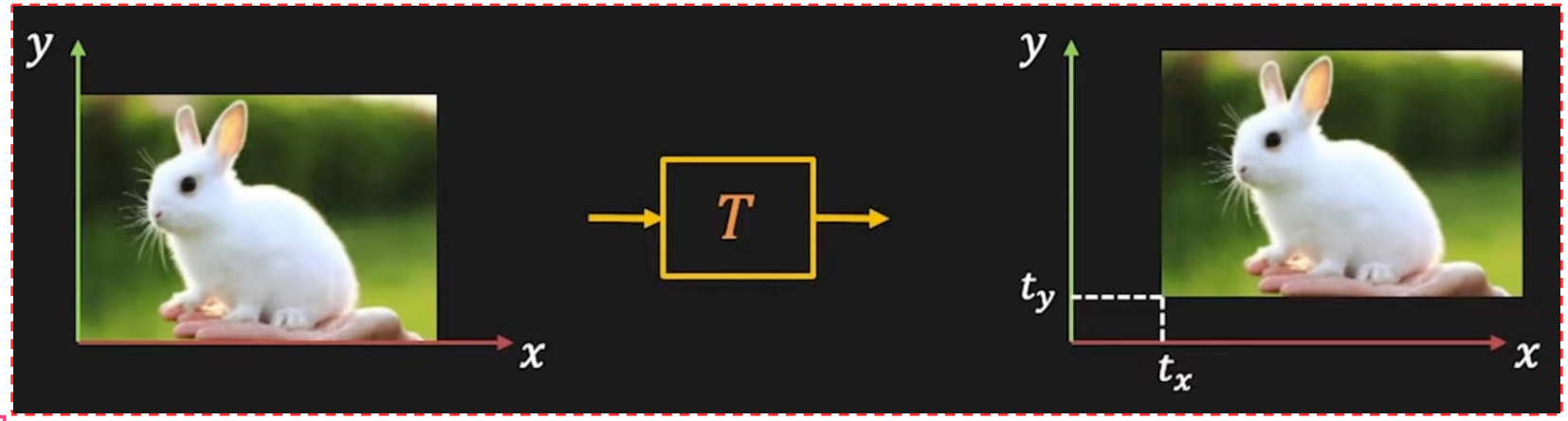
Mirror about line $y = x$:

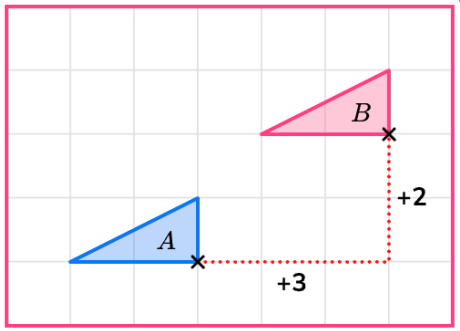
$x_2 = y_1$

$y_2 = x_1$

$M_{xy} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$

Translation





$x2 = x1 + tx$ $y2 = y1 + ty$

Can translation be expressed as 2x2 matrix?

Homogenous Coordinates

$$\begin{bmatrix}2\\4\end{bmatrix} \Rightarrow \begin{bmatrix}2\\4\\1\end{bmatrix}$$

Euclidean homogeneous

$$\begin{bmatrix}4\\8\\2\end{bmatrix} = \begin{bmatrix}4/2\\8/2\\1\end{bmatrix} \Rightarrow \begin{bmatrix}4/2\\8/2\end{bmatrix} = \begin{bmatrix}2\\4\end{bmatrix}$$

homogeneous Euclidean

$$\begin{bmatrix}x\\y\end{bmatrix} \Rightarrow \begin{bmatrix}x\\y\\1\end{bmatrix}$$

Euclidean homogeneous

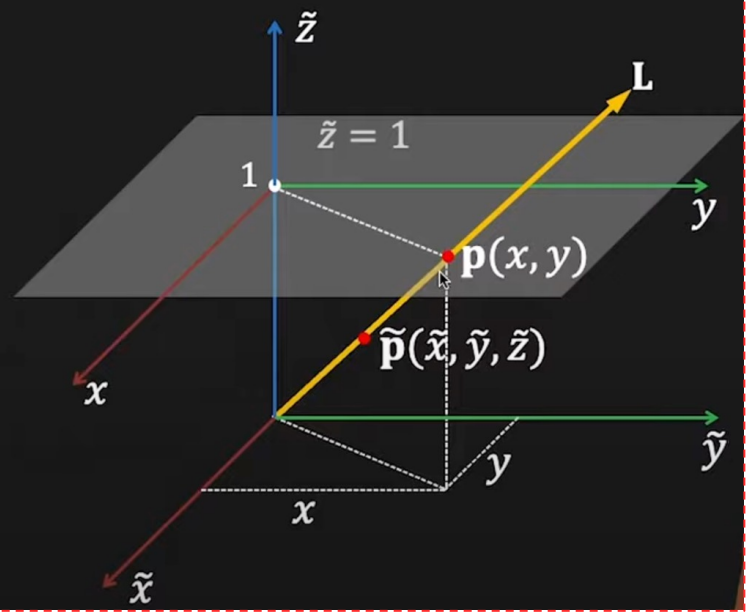
$$\begin{bmatrix}u\\v\\w\end{bmatrix} = \begin{bmatrix}u/w\\v/w\\1\end{bmatrix} \Rightarrow \begin{bmatrix}u/w\\v/w\end{bmatrix} = \begin{bmatrix}x\\y\end{bmatrix}$$

homogeneous Euclidean

The homogenous representation of a 2D point $\mathbf{p} = (x, y)$ is a 3D point $\tilde{\mathbf{p}} = (\tilde{x}, \tilde{y}, \tilde{z})$. The third coordinate $\tilde{z} \neq 0$ is fictitious such that:

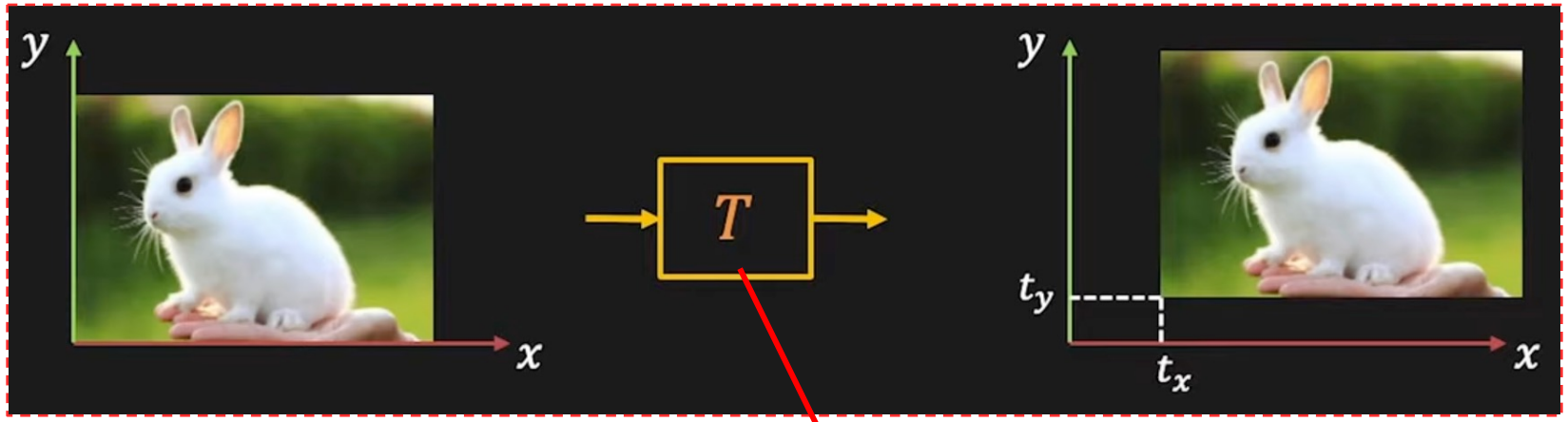
$$x = \frac{\tilde{x}}{\tilde{z}} \qquad y = \frac{\tilde{y}}{\tilde{z}}$$

$$\mathbf{p} \equiv \begin{bmatrix}x\\y\\1\end{bmatrix} \equiv \begin{bmatrix}\tilde{z}x\\\tilde{z}y\\\tilde{z}\end{bmatrix} \equiv \begin{bmatrix}\tilde{x}\\\tilde{y}\\\tilde{z}\end{bmatrix} = \tilde{\mathbf{p}}$$



Every point on line L (except origin) represent the homogeneous coordinate of p(x,y)

Translation (3D)



x2 = x1 + tx y2 = y1 + ty

$$\begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} \equiv \begin{bmatrix} \tilde{x}_2 \\ \tilde{y}_2 \\ \tilde{z}_2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix}$$

Scaling, Translation, Rotation, Skew

$$\begin{bmatrix}\tilde{x}_2\\ \tilde{y}_2\\ \tilde{z}_2\end{bmatrix} = \begin{bmatrix}s_x & 0 & 0\\ 0 & s_y & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}x_1\\ y_1\\ 1\end{bmatrix}$$

Scaling

$$\begin{bmatrix}\tilde{x}_2\\ \tilde{y}_2\\ \tilde{z}_2\end{bmatrix} = \begin{bmatrix}1 & m_x & 0\\ 0 & 1 & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}x_1\\ y_1\\ 1\end{bmatrix}$$

Skew

$$\begin{bmatrix}\tilde{x}_2\\ \tilde{y}_2\\ \tilde{z}_2\end{bmatrix} = \begin{bmatrix}1 & 0 & t_x\\ 0 & 1 & t_y\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}x_1\\ y_1\\ 1\end{bmatrix}$$

Translation

$$\begin{bmatrix}\tilde{x}_2\\ \tilde{y}_2\\ \tilde{z}_2\end{bmatrix} = \begin{bmatrix}cos\theta & -sin\theta & 0\\ sin\theta & cos\theta & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}x_1\\ y_1\\ 1\end{bmatrix}$$

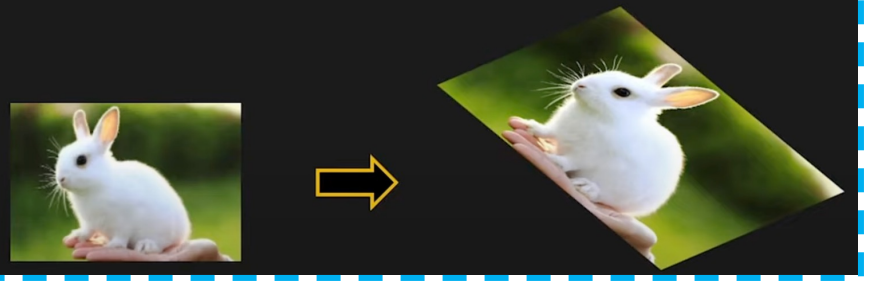
Rotation

Composition of these transformation (scaling + skew + translation + rotation)?

Affine Transformation

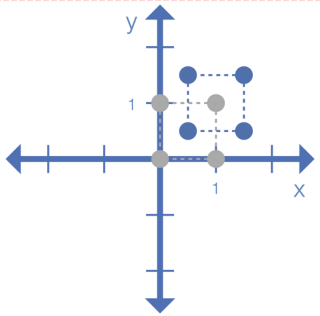
Any transformation of the form:

$$\begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} \equiv \begin{bmatrix} \tilde{x}_2 \\ \tilde{y}_2 \\ \tilde{z}_2 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \tilde{x}_1 \\ \tilde{y}_1 \\ \tilde{z}_1 \end{bmatrix}$$



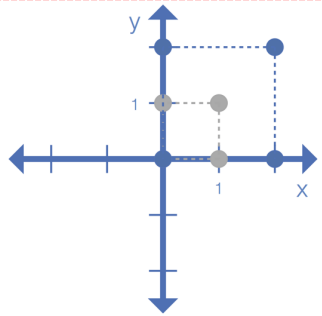
Translate

$$\begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$



Scale

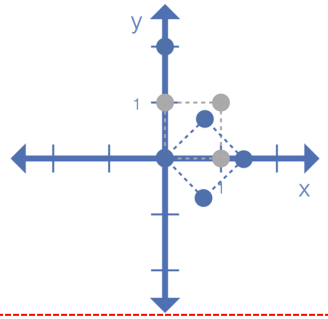
$$\begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



Rotate

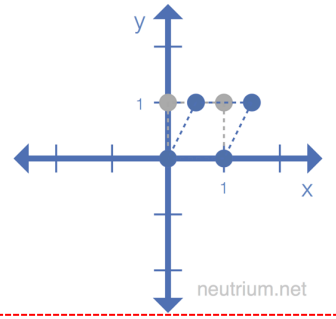
$$\begin{bmatrix} c & s & 0 \\ -s & c & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

c = s = sin(45°)



Shear

$$\begin{bmatrix} 1 & 0.5 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$



neutrium.net

Outline

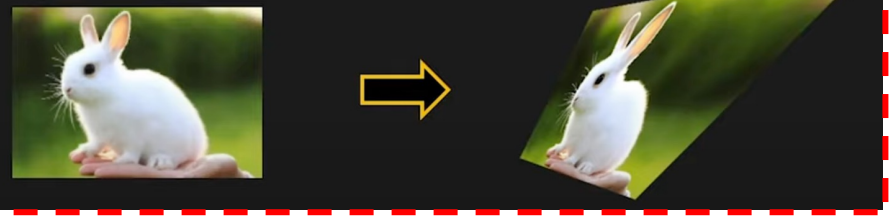
- What is image stitching/panorama image
- Edge Detector
- Blob Detector
- SIFT detector
- Image Transformation: 2D & 3D
- Image stitching/panorama Techniques

Projective Transformation

Mapping of one plane to another throught a point

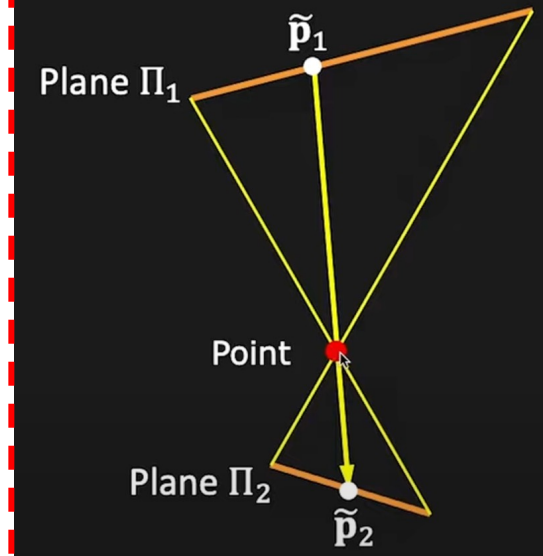
Any transformation of the form:

$$\begin{bmatrix}\tilde{x}_2\\ \tilde{y}_2\\ \tilde{z}_2\end{bmatrix} = \begin{bmatrix}h_{11} & h_{12} & h_{13}\\ h_{21} & h_{22} & h_{23}\\ h_{31} & h_{32} & h_{33}\end{bmatrix}\begin{bmatrix}\tilde{x}_1\\ \tilde{y}_1\\ \tilde{z}_1\end{bmatrix} \qquad \tilde{\mathbf{p}}_2 = H\tilde{\mathbf{p}}_1$$



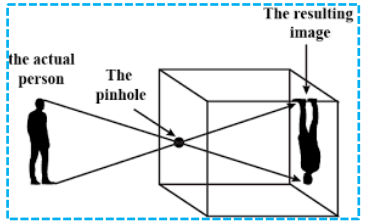
Homography can only be defined up to a scale.

$$\begin{bmatrix}h_{11} & h_{12} & h_{13}\\ h_{21} & h_{22} & h_{23}\\ h_{31} & h_{32} & h_{33}\end{bmatrix}\begin{bmatrix}\tilde{x}_1\\ \tilde{y}_1\\ \tilde{z}_1\end{bmatrix} \equiv \begin{bmatrix}\tilde{x}_2\\ \tilde{y}_2\\ \tilde{z}_2\end{bmatrix} \equiv k\begin{bmatrix}h_{11} & h_{12} & h_{13}\\ h_{21} & h_{22} & h_{23}\\ h_{31} & h_{32} & h_{33}\end{bmatrix}\begin{bmatrix}\tilde{x}_1\\ \tilde{y}_1\\ \tilde{z}_1\end{bmatrix}$$



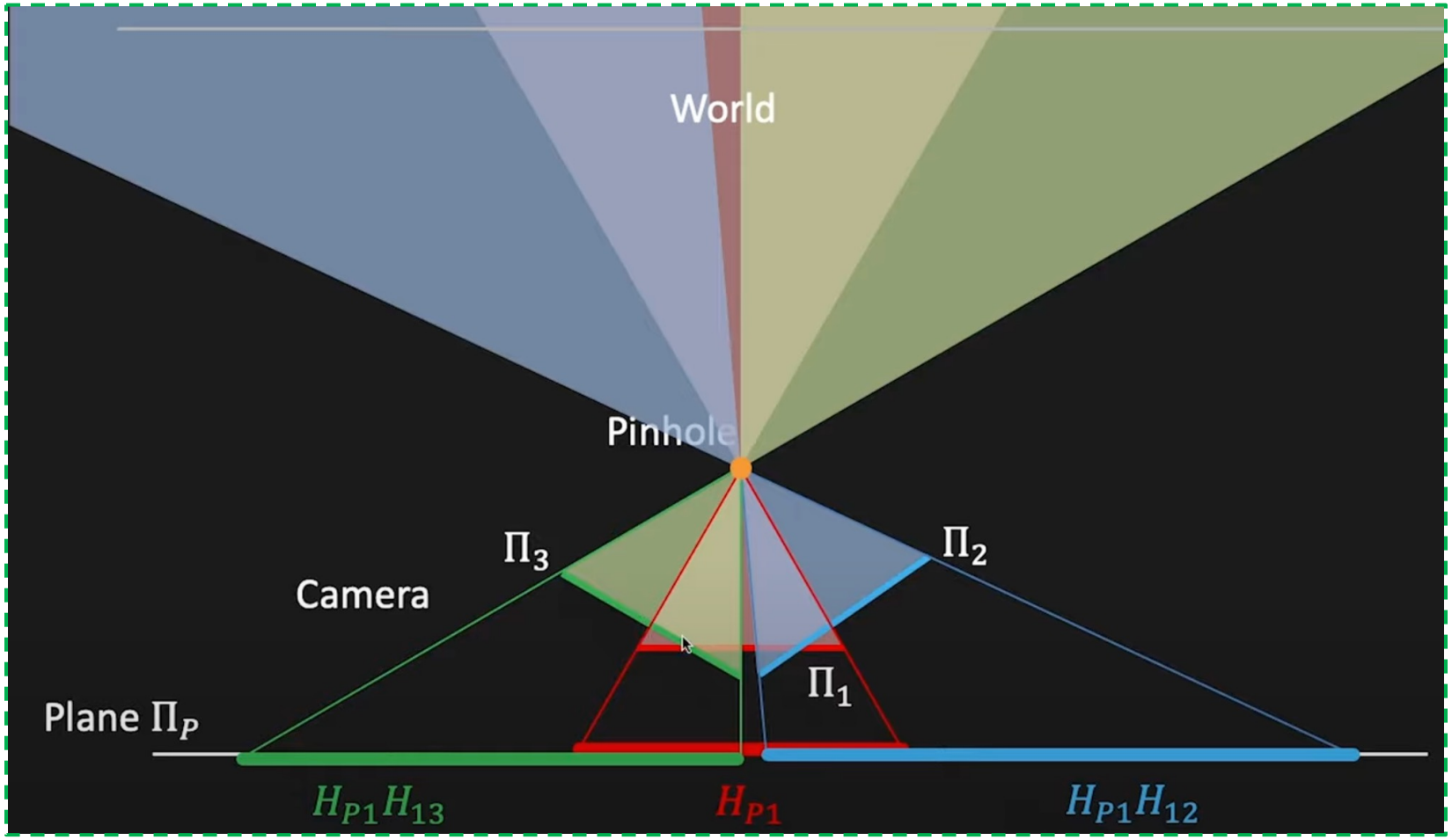
$$\tilde{\mathbf{p}}_2 = H\tilde{\mathbf{p}}_1$$

$$\begin{bmatrix}\tilde{x}_2\\ \tilde{y}_2\\ \tilde{z}_2\end{bmatrix} = \begin{bmatrix}h_{11} & h_{12} & h_{13}\\ h_{21} & h_{22} & h_{23}\\ h_{31} & h_{32} & h_{33}\end{bmatrix}\begin{bmatrix}\tilde{x}_1\\ \tilde{y}_1\\ \tilde{z}_1\end{bmatrix}$$

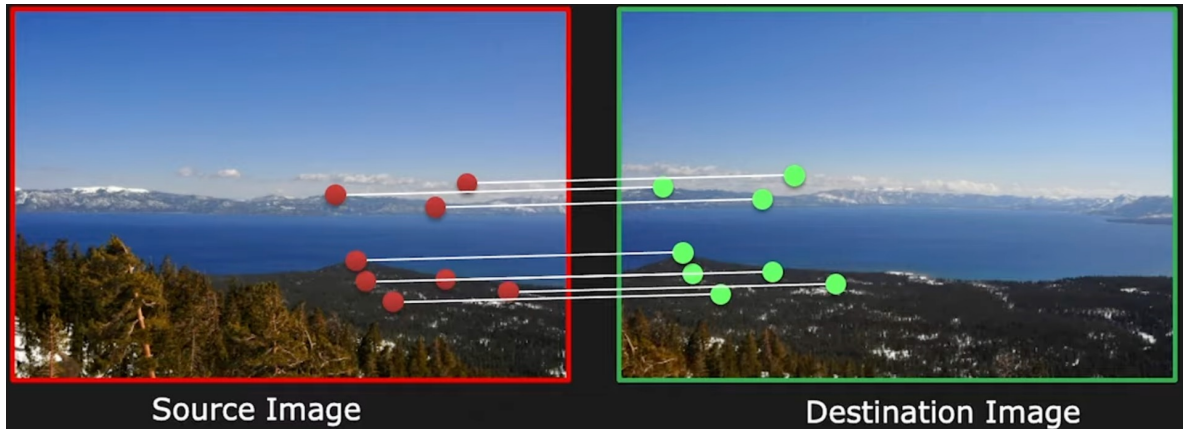


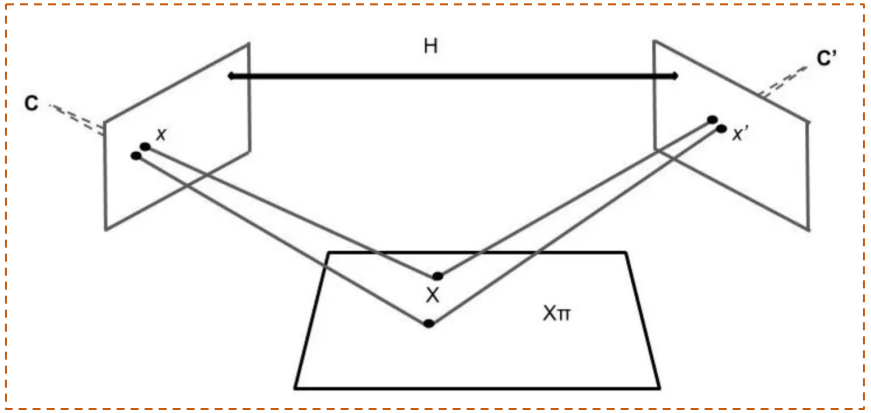
A homography is a projective transformation between two planes. How to compute it?

Homography Composition



Computing Homograpy





Given a set of matching features/points between image 1 and image 2, find the homography H that best "agrees" with the matches. How many mininum pairs of matching points?

For a given pair i of corresponding points

$$\begin{bmatrix} x_d \\ y_d \\ 1 \end{bmatrix} \equiv \begin{bmatrix} \tilde{x}_d \\ \tilde{y}_d \\ \tilde{z}_d \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x_s \\ y_s \\ 1 \end{bmatrix}$$

$$x_d^{(i)} = \frac{\tilde{x}_d^{(i)}}{\tilde{z}_d^{(i)}} = \frac{h_{11}x_s^{(i)} + h_{12}y_s^{(i)} + h_{13}}{h_{31}x_s^{(i)} + h_{32}y_s^{(i)} + h_{33}}$$

$$y_d^{(i)} = \frac{\tilde{y}_d^{(i)}}{\tilde{z}_d^{(i)}} = \frac{h_{21}x_s^{(i)} + h_{22}y_s^{(i)} + h_{23}}{h_{31}x_s^{(i)} + h_{32}y_s^{(i)} + h_{33}}$$

Computing Homograpy

$$x_d^{(i)} = \frac{\tilde{x}_d^{(i)}}{\tilde{z}_d^{(i)}} = \frac{h_{11}x_s^{(i)} + h_{12}y_s^{(i)} + h_{13}}{h_{31}x_s^{(i)} + h_{32}y_s^{(i)} + h_{33}}$$

$$y_d^{(i)} = \frac{\tilde{y}_d^{(i)}}{\tilde{z}_d^{(i)}} = \frac{h_{21}x_s^{(i)} + h_{22}y_s^{(i)} + h_{23}}{h_{31}x_s^{(i)} + h_{32}y_s^{(i)} + h_{33}}$$

$$x_d^{(i)}\left(h_{31}x_s^{(i)} + h_{32}y_s^{(i)} + h_{33}\right) = h_{11}x_s^{(i)} + h_{12}y_s^{(i)} + h_{13}$$

$$y_d^{(i)}\left(h_{31}x_s^{(i)} + h_{32}y_s^{(i)} + h_{33}\right) = h_{21}x_s^{(i)} + h_{22}y_s^{(i)} + h_{23}$$

$$\begin{bmatrix} x_s^{(1)} & y_s^{(1)} & 1 & 0 & 0 & 0 & -x_d^{(1)}x_s^{(1)} & -x_d^{(1)}y_s^{(1)} & -x_d^{(1)} \\ 0 & 0 & 0 & x_s^{(1)} & y_s^{(1)} & 1 & -y_d^{(1)}x_s^{(1)} & -y_d^{(1)}y_s^{(1)} & -y_d^{(1)} \\ \vdots & & & & & & & & \\ x_s^{(i)} & y_s^{(i)} & 1 & 0 & 0 & 0 & -x_d^{(i)}x_s^{(i)} & -x_d^{(i)}y_s^{(i)} & -x_d^{(i)} \\ 0 & 0 & 0 & x_s^{(i)} & y_s^{(i)} & 1 & -y_d^{(i)}x_s^{(i)} & -y_d^{(i)}y_s^{(i)} & -y_d^{(i)} \\ \vdots & & & & & & & & \\ x_s^{(n)} & y_s^{(n)} & 1 & 0 & 0 & 0 & -x_d^{(n)}x_s^{(n)} & -x_d^{(n)}y_s^{(n)} & -x_d^{(n)} \\ 0 & 0 & 0 & x_s^{(n)} & y_s^{(n)} & 1 & -y_d^{(n)}x_s^{(n)} & -y_d^{(n)}y_s^{(n)} & -y_d^{(n)} \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}$$

A (Known) — $\mathbf{h}$ (Unknown)

All corresponding points

$$\begin{bmatrix} x_s^{(i)} & y_s^{(i)} & 1 & 0 & 0 & 0 & -x_d^{(i)}x_s^{(i)} & -x_d^{(i)}y_s^{(i)} & -x_d^{(i)} \\ 0 & 0 & 0 & x_s^{(i)} & y_s^{(i)} & 1 & -y_d^{(i)}x_s^{(i)} & -y_d^{(i)}y_s^{(i)} & -y_d^{(i)} \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

(Known) — $\mathbf{h}$ (Unknown)

One corresponding point

Constraint Least Square

$$\begin{bmatrix} x_s^{(1)} & y_s^{(1)} & 1 & 0 & 0 & 0 & -x_d^{(1)}x_s^{(1)} & -x_d^{(1)}y_s^{(1)} & -x_d^{(1)} \\ 0 & 0 & 0 & x_s^{(1)} & y_s^{(1)} & 1 & -y_d^{(1)}x_s^{(1)} & -y_d^{(1)}y_s^{(1)} & -y_d^{(1)} \\ \vdots & & & & & & & & \\ x_s^{(i)} & y_s^{(i)} & 1 & 0 & 0 & 0 & -x_d^{(i)}x_s^{(i)} & -x_d^{(i)}y_s^{(i)} & -x_d^{(i)} \\ 0 & 0 & 0 & x_s^{(i)} & y_s^{(i)} & 1 & -y_d^{(i)}x_s^{(i)} & -y_d^{(i)}y_s^{(i)} & -y_d^{(i)} \\ \vdots & & & & & & & & \\ x_s^{(n)} & y_s^{(n)} & 1 & 0 & 0 & 0 & -x_d^{(n)}x_s^{(n)} & -x_d^{(n)}y_s^{(n)} & -x_d^{(n)} \\ 0 & 0 & 0 & x_s^{(n)} & y_s^{(n)} & 1 & -y_d^{(n)}x_s^{(n)} & -y_d^{(n)}y_s^{(n)} & -y_d^{(n)} \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}$$

A (Known) $\quad$ $\mathbf{h}$ (Unknown)

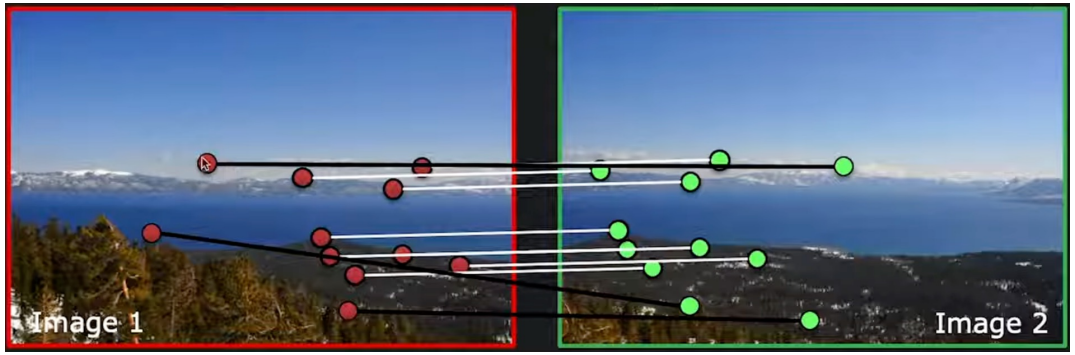
Solve for $\mathbf{h}$: $A\,\mathbf{h} = \mathbf{0}$ such that $\|\mathbf{h}\|^2 = 1$

Define least squares problem:

$$\min_{\mathbf{h}} \|A\mathbf{h}\|^2 \quad \text{such that } \|\mathbf{h}\|^2 = 1$$

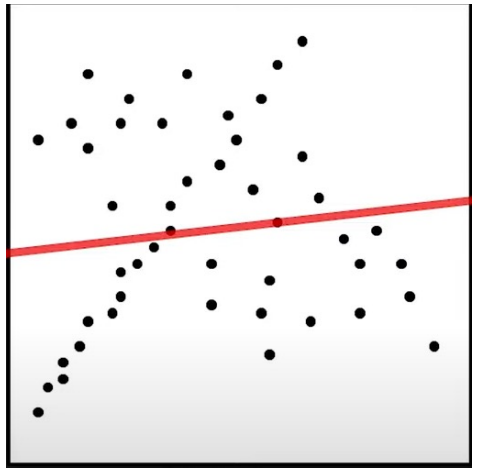
Deal with outlier: RANSAC

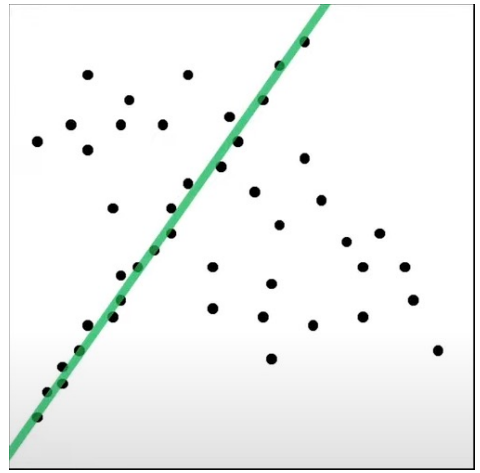
How to compute the homography in the presence of outliers



1. Randomly choose s samples. Typically s is the mininum samples to fit a model
2. Fit the model to the randomly choosen samples
3. Count the number M of data points (inliers)that fit the model within a measure of error ε
4. Repeat steps 1-3 N times
5. Choose the model that has the largest number M of inliers

Random Sample Consensus





Warping Images

Given a transformation T and a image $f(x,y)$, compute the transformed image $g(x,y)$

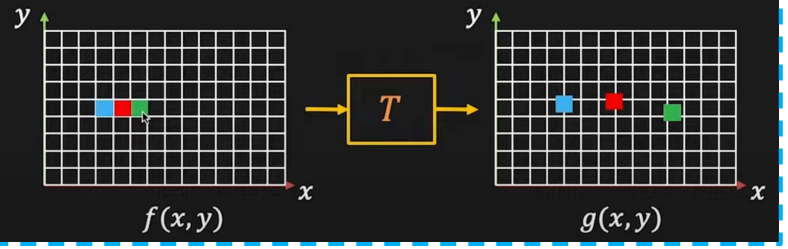
$$g(x,y) = f(T(x,y))$$

Result in holes

Forward Warping

Send each pixel (x,y) in $f(x,y)$ to its corresponding location $T(x,y)$ in $g(x,y)$

$$g(x,y) = f(T(x,y))$$



Get each pixel (x,y) in $g(x,y)$ from its corresponding location $T^{-1}(x,y)$ in $f(x,y)$

$$g(x,y) = f(T(x,y))$$

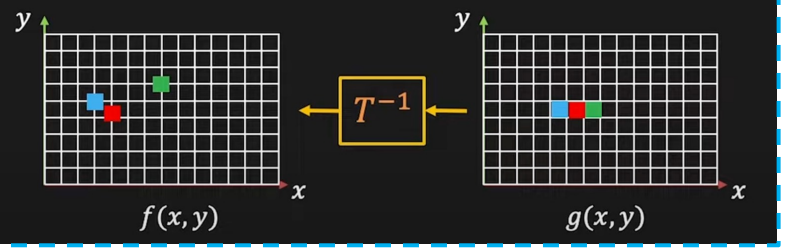
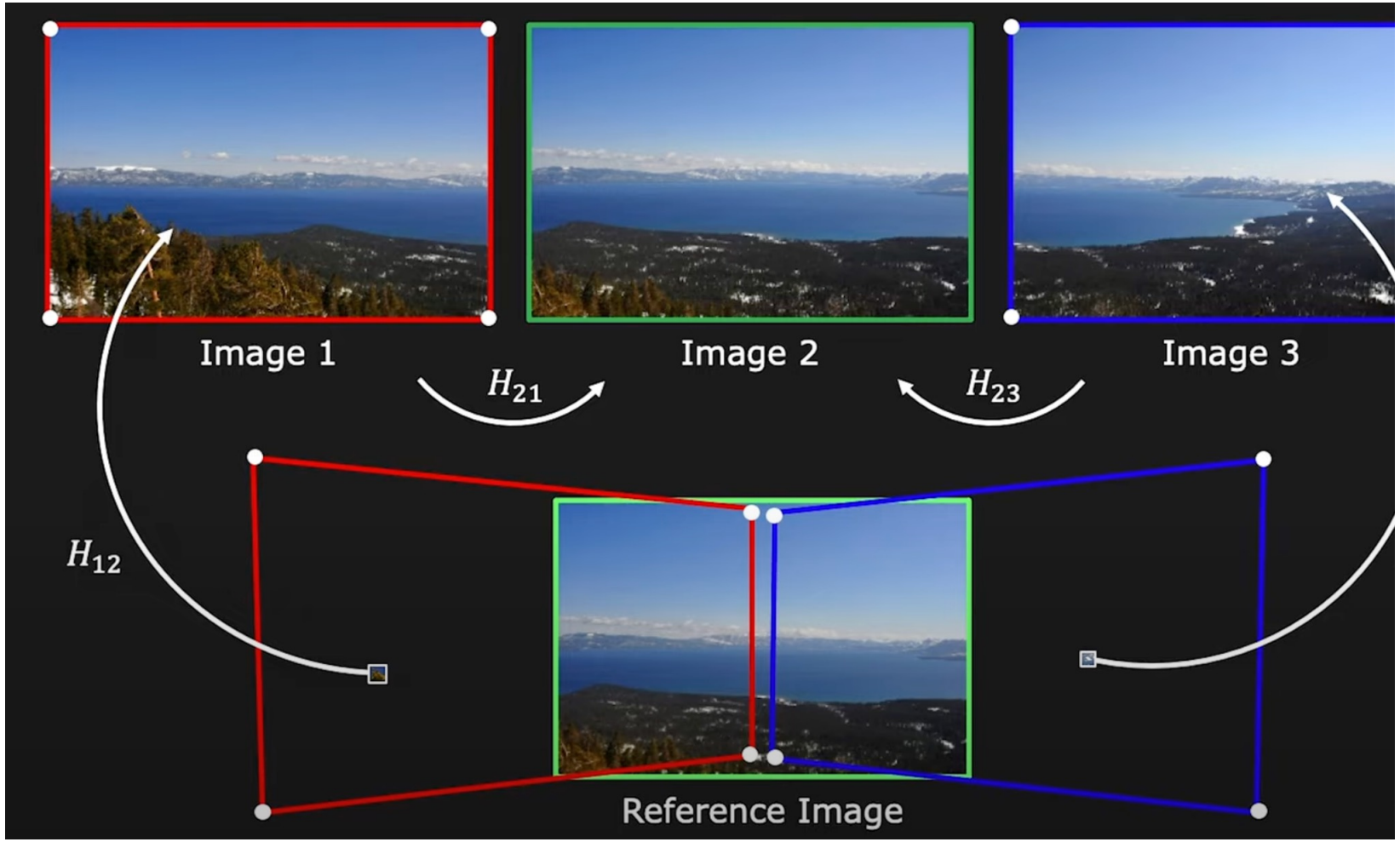
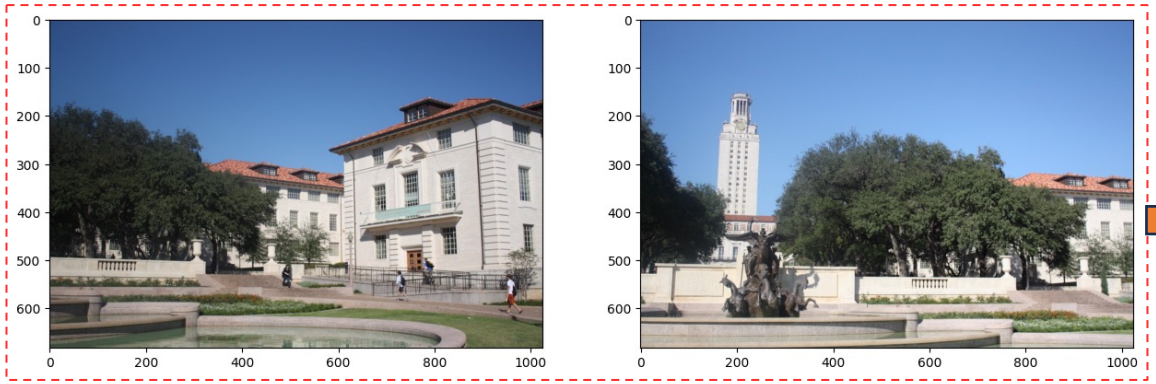
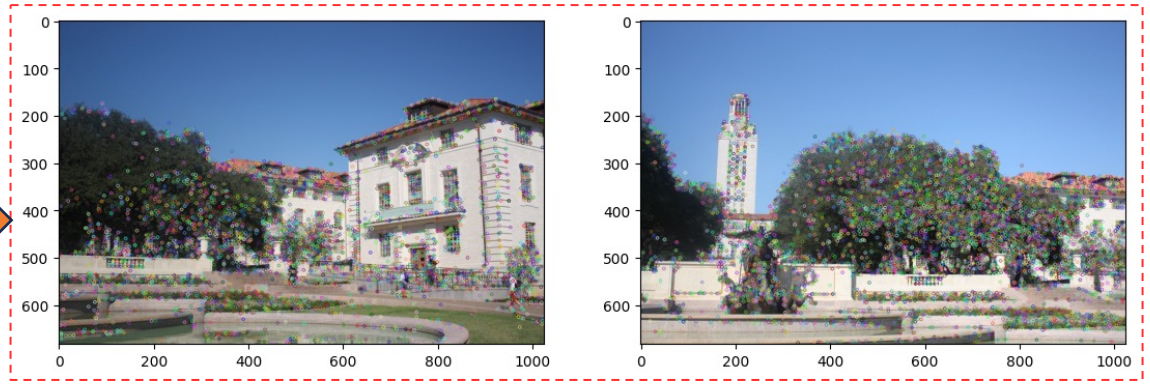


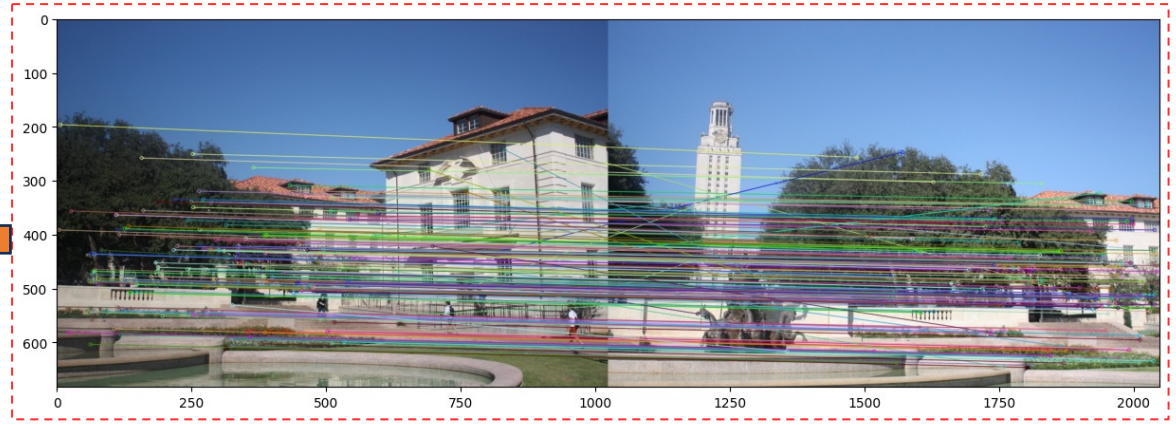
Image Alignment Process



Blending Images

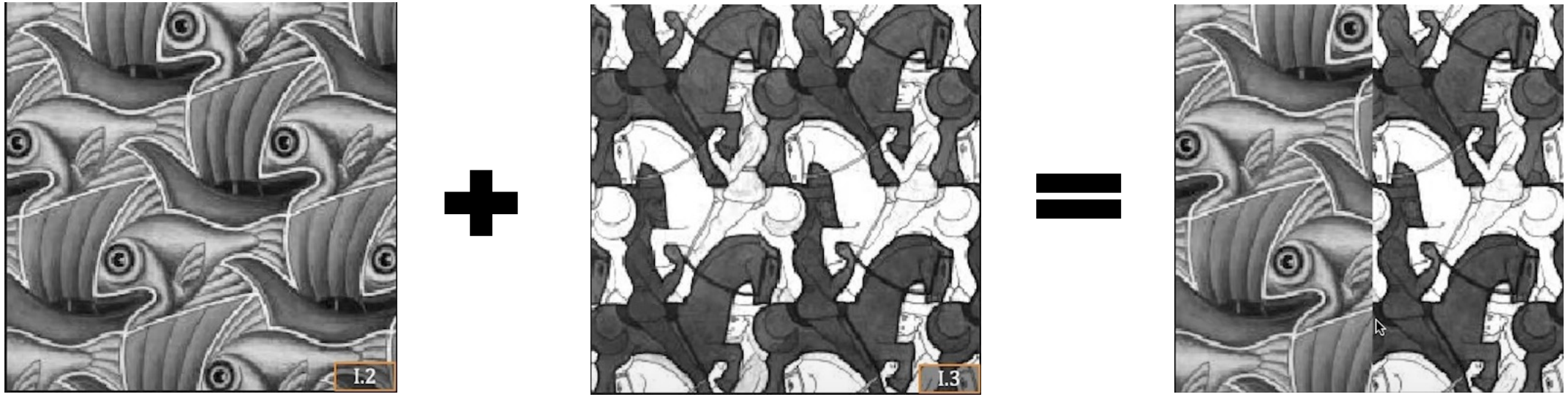




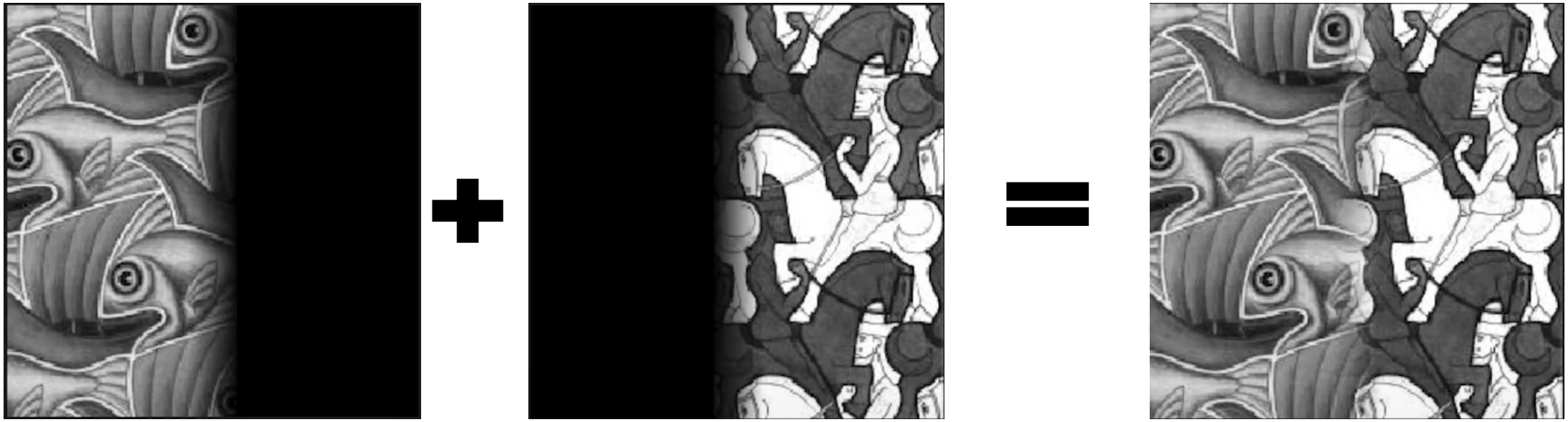


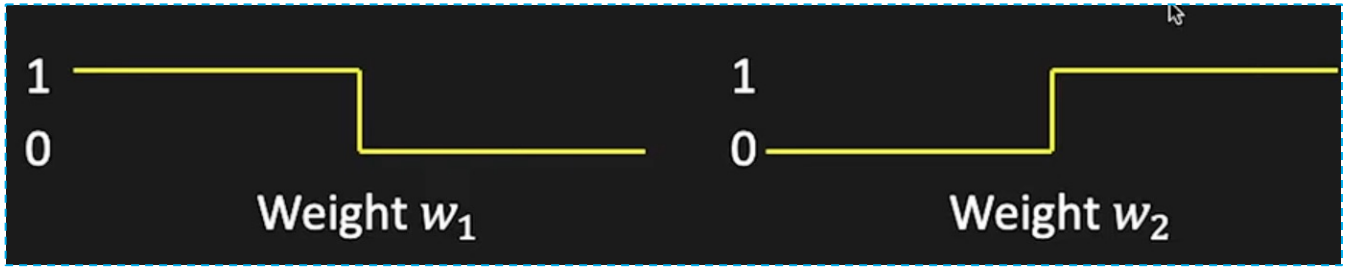
Seams still visible

Blending Images



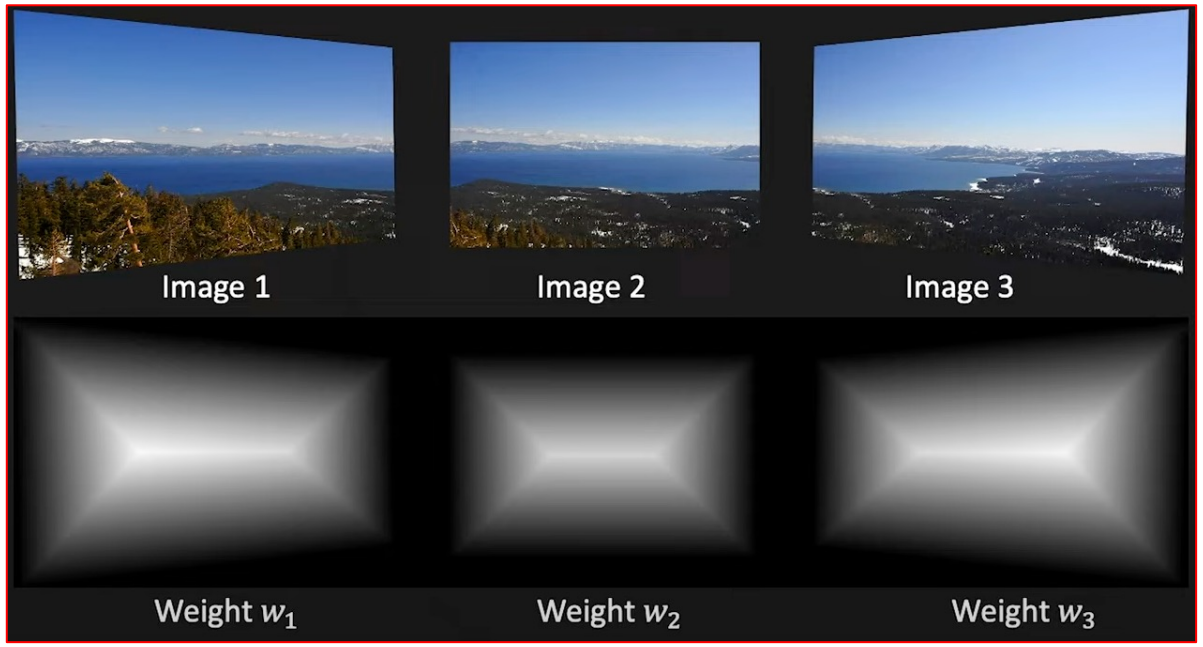
Blending Images



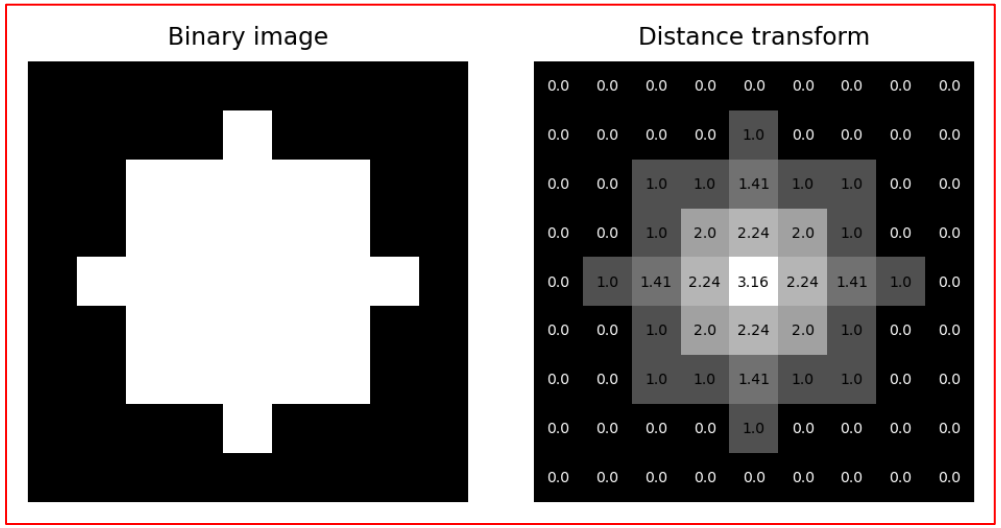


$$I_{blend} = \frac{w_1 I_1 + w_2 I_2}{w_1 + w_2}$$

Blending Images (Distance Transform)



Pixels closer to the edge get lower weight



The distance transform is an operator normally only applied to binary images. The result of the transform is a graylevel image that looks similar to the input image, except that the graylevel intensities of points inside foreground regions are changed to show the distance to the closest boundary from each point

Distance Transform

Euclidean

0	0	0
0	1	0
0	0	0

Image

1.41	1.0	1.41
1.0	0.0	1.0
1.41	1.0	1.41

Distance Transform

Chessboard

0	0	0
0	1	0
0	0	0

Image

1	1	1
1	0	1
1	1	1

Distance Transform

City Block

0	0	0
0	1	0
0	0	0

Image

2	1	2
1	0	1
2	1	2

Distance Transform

Quasi-Euclidean

0	0	0	0	0
0	0	0	0	0
0	0	1	0	0
0	0	0	0	0
0	0	0	0	0

Image

2.8	2.4	2.0	2.4	2.8
2.4	1.4	1.0	1.4	2.4
2.0	1.0	0	1.0	2.0
2.4	1.4	1.0	1.4	2.4
2.8	2.4	2.0	2.4	2.8

Distance Transform

Blending Images (Distance Transform)

Input image

Panorama Image


Thank you