Google Developers Group Manipal

DevFest 2017

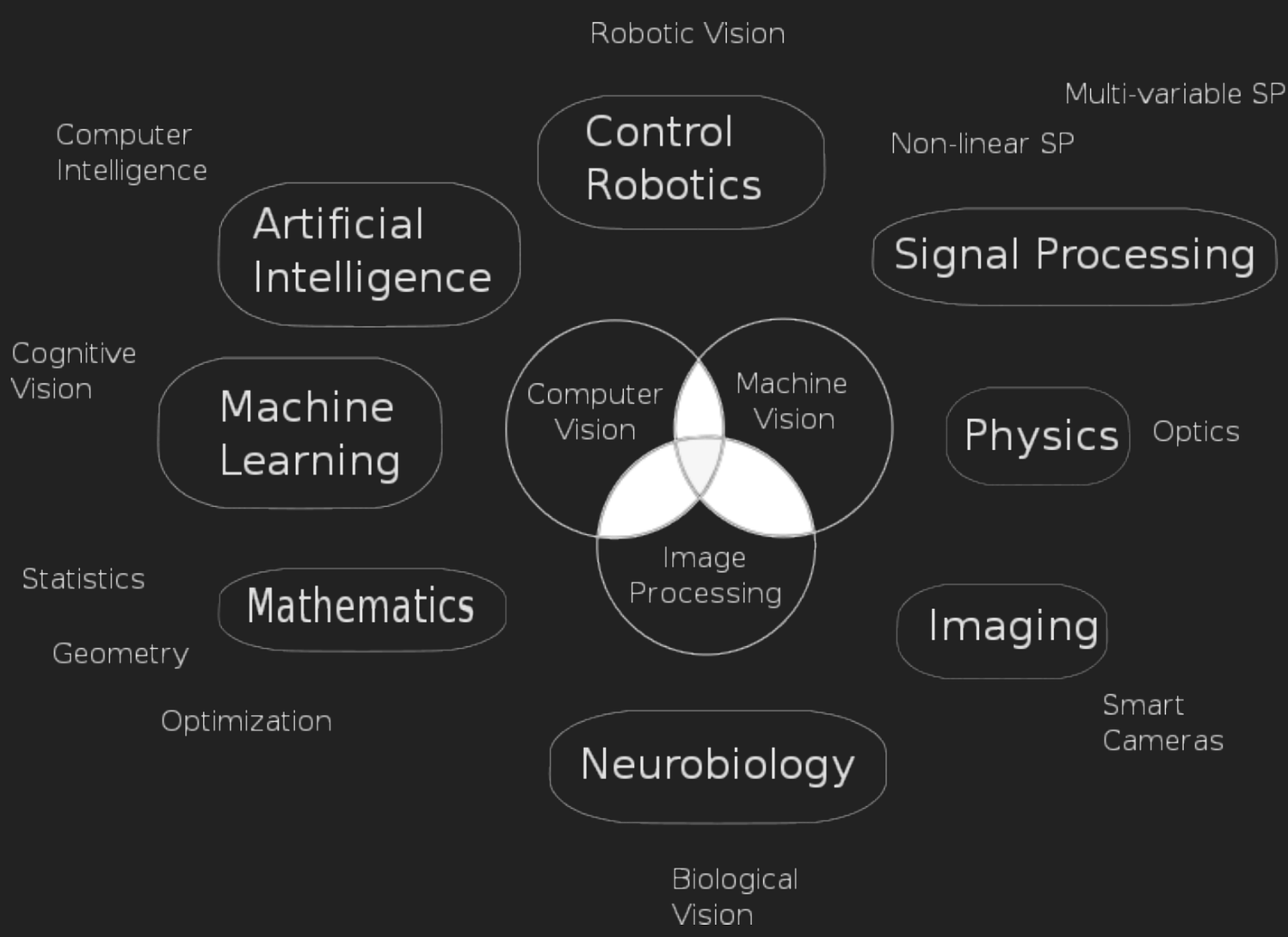
SIDDHARTH KOTHIYAL • AVIKANT SAINI

IMPLEMENTATION OF COMPUTER VISION ALGORITHMS

WHAT IS COMPUTER VISION?

WHAT IS COMPUTER VISION

- Image → Sensor → Interpreter → Information



GOAL OF COMPUTER VISION

- Extract meaning from pixels
- Hoomans be good at this
- Comphooters not so much

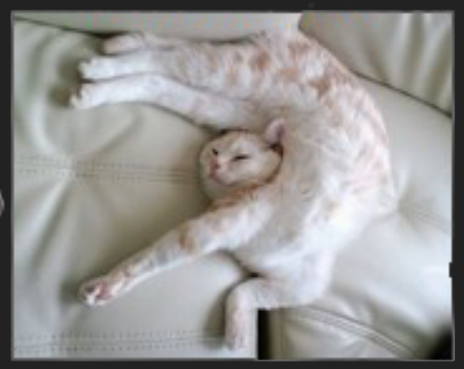
KINDS OF COMPOOTHER VISHUN

- LOW LEVEL
  - Measurements
  - Enhancements
  - Region segmentation
  - Features
- MID LEVEL
  - Reconstruction
  - Depth
  - Motion Estimation
- HIGH LEVEL
  - Category Detection
  - Activity Recognition
  - Deep Understanding

WHY BE COMPUTER VISION DIFFICULT

- Viewpoint variation
- Illumination changes
- Scale
- Deformation
- Occlusion
- Background Clutter
- Motion
- Object class variation
- Local ambiguity





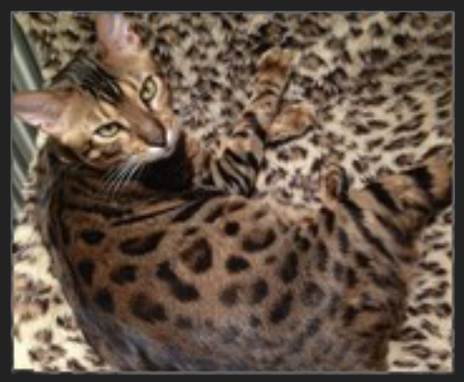
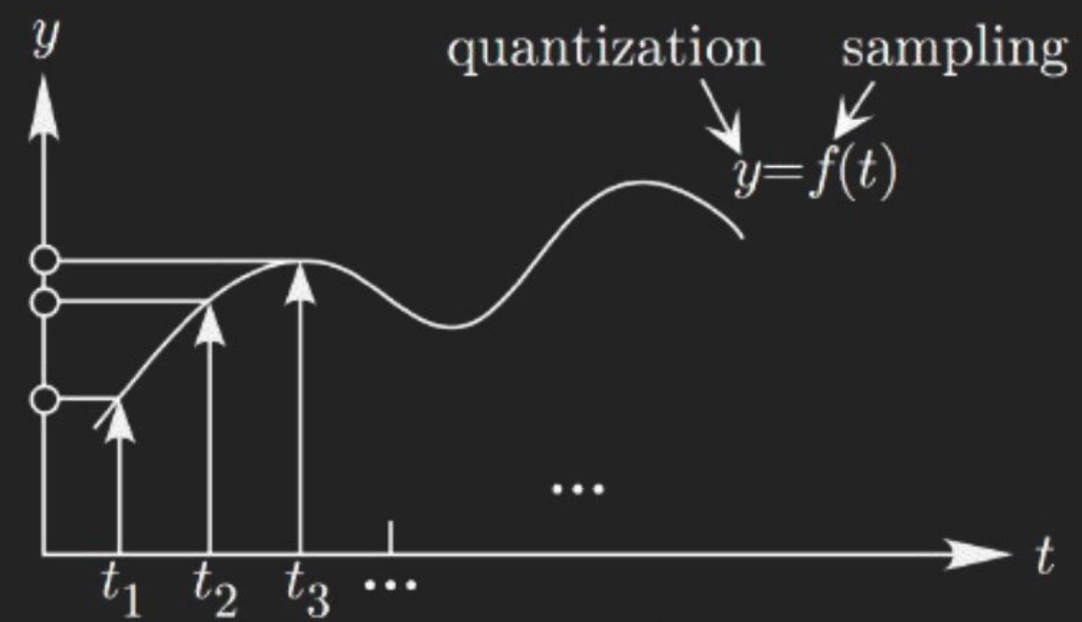



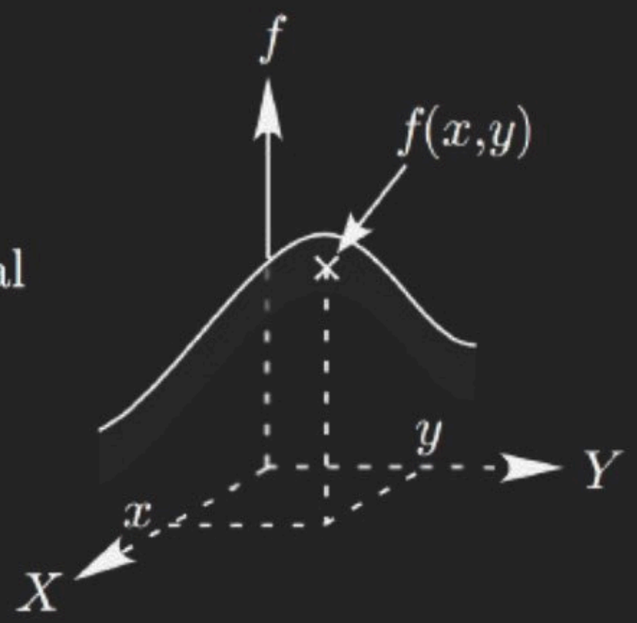

IMAGE FILTERING

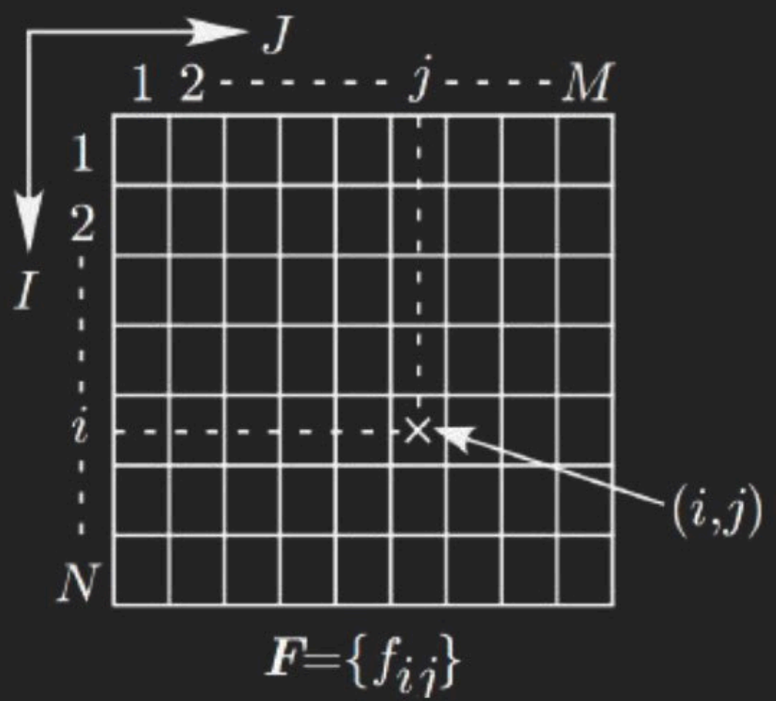
one-dimensional



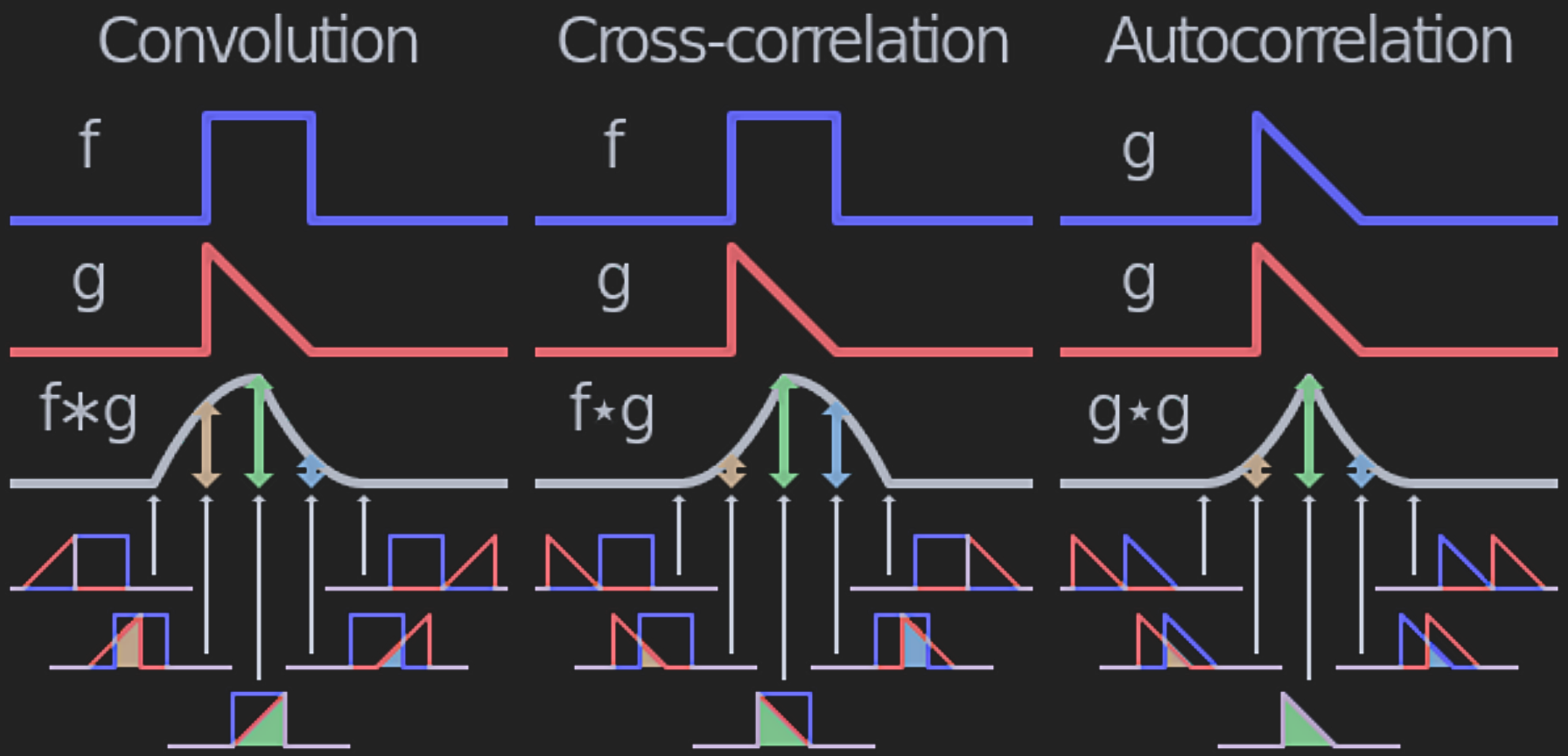
continuous signal ⟶ digitized signal

two-dimensional





CONVOLUTION AND CORRELATION



CORRELATION – LINEAR RELATIONSHIP

$$f \otimes h = \sum_k \sum_l f(k,l)h(k,l)$$

f = Image

h = Kernel

f

f_1	f_2	f_3
f_4	f_5	f_6
f_7	f_8	f_9

$\otimes$

h

h_1	h_2	h_3
h_4	h_5	h_6
h_7	h_8	h_9

$$f \otimes h = f_1h_1 + f_2h_2 + f_3h_3 + f_4h_4 + f_5h_5 + f_6h_6 + f_7h_7 + f_8h_8 + f_9h_9$$

CONVOLUTION

$$f * h = \sum_k \sum_l f(k,l)h(-k,-l)$$

f = Image

h = Kernel

h

h_7	h_8	h_9
h_4	h_5	h_6
h_1	h_2	h_3

$X - flip$

h_1	h_2	h_3
h_4	h_5	h_6
h_7	h_8	h_9

$Y - flip$

f

f_1	f_2	f_3
f_4	f_5	f_6
f_7	f_8	f_9

$*$

h_9	h_8	h_7
h_6	h_5	h_4
h_3	h_2	h_1

$$f * h = f_1h_9 + f_2h_8 + f_3h_7 + f_4h_6 + f_5h_5 + f_6h_4 + f_7h_3 + f_8h_2 + f_9h_1$$

FILTER = ?

- Image contains a discrete number of pixels
- Filtering is forming a new image whose pixel values are a combination of original image's

Not these pixels →

IMAGE FILTERING: MOVING AVERAGE

$$g[\cdot,\cdot]\ \frac{1}{9}\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix}$$

$f[.,.]$

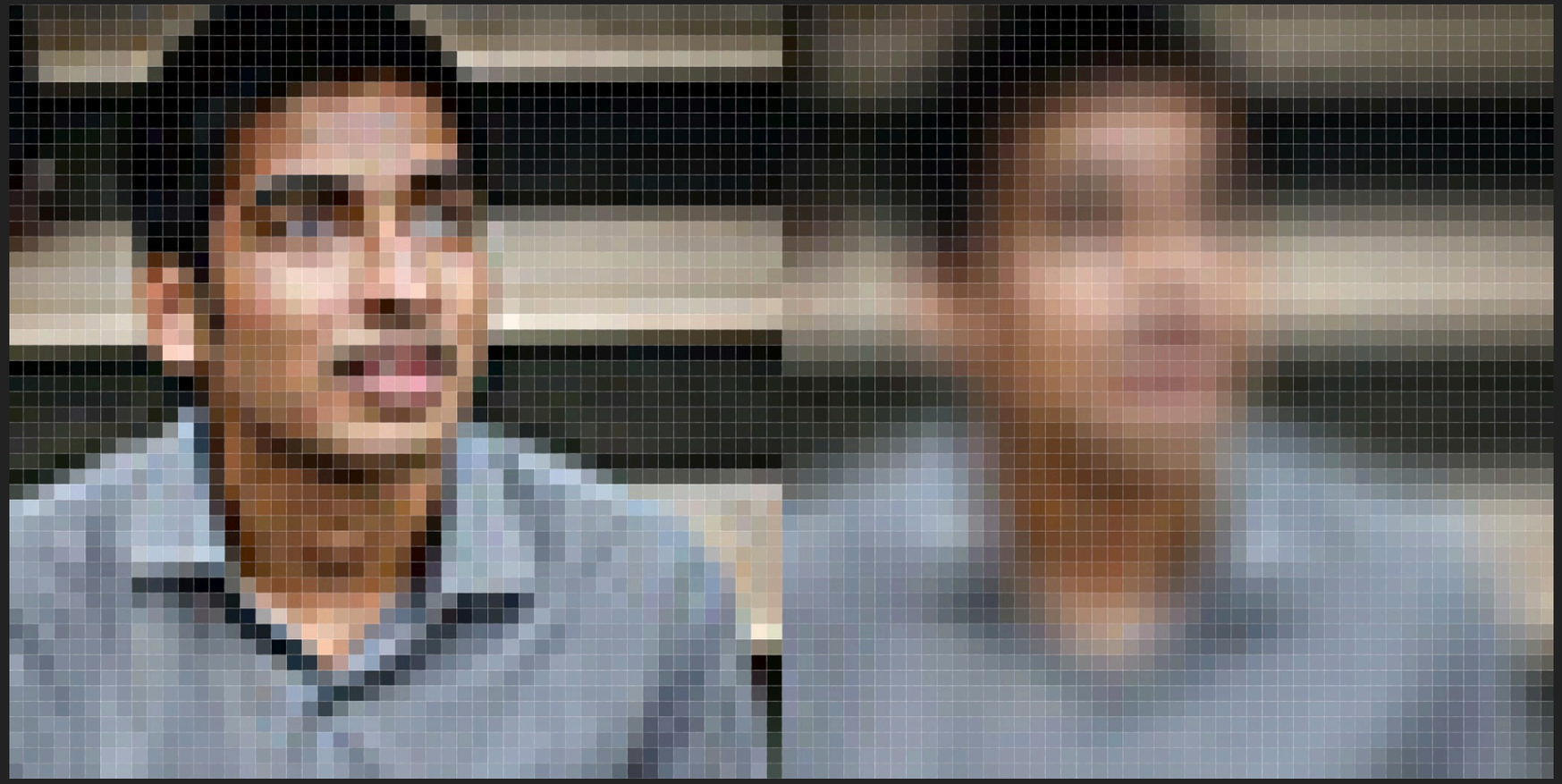
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$h[.,.]$

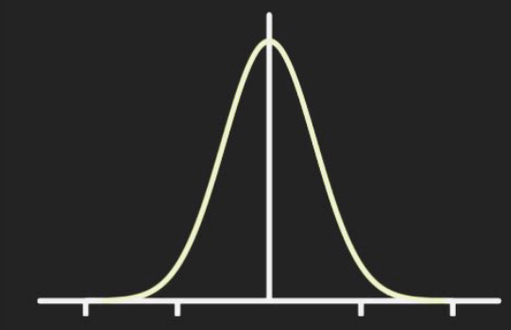
$$h[m,n]=\sum_{k,l} g[k,l]\ f[m+k,n+l]$$

AVERAGE FILTER

$$\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

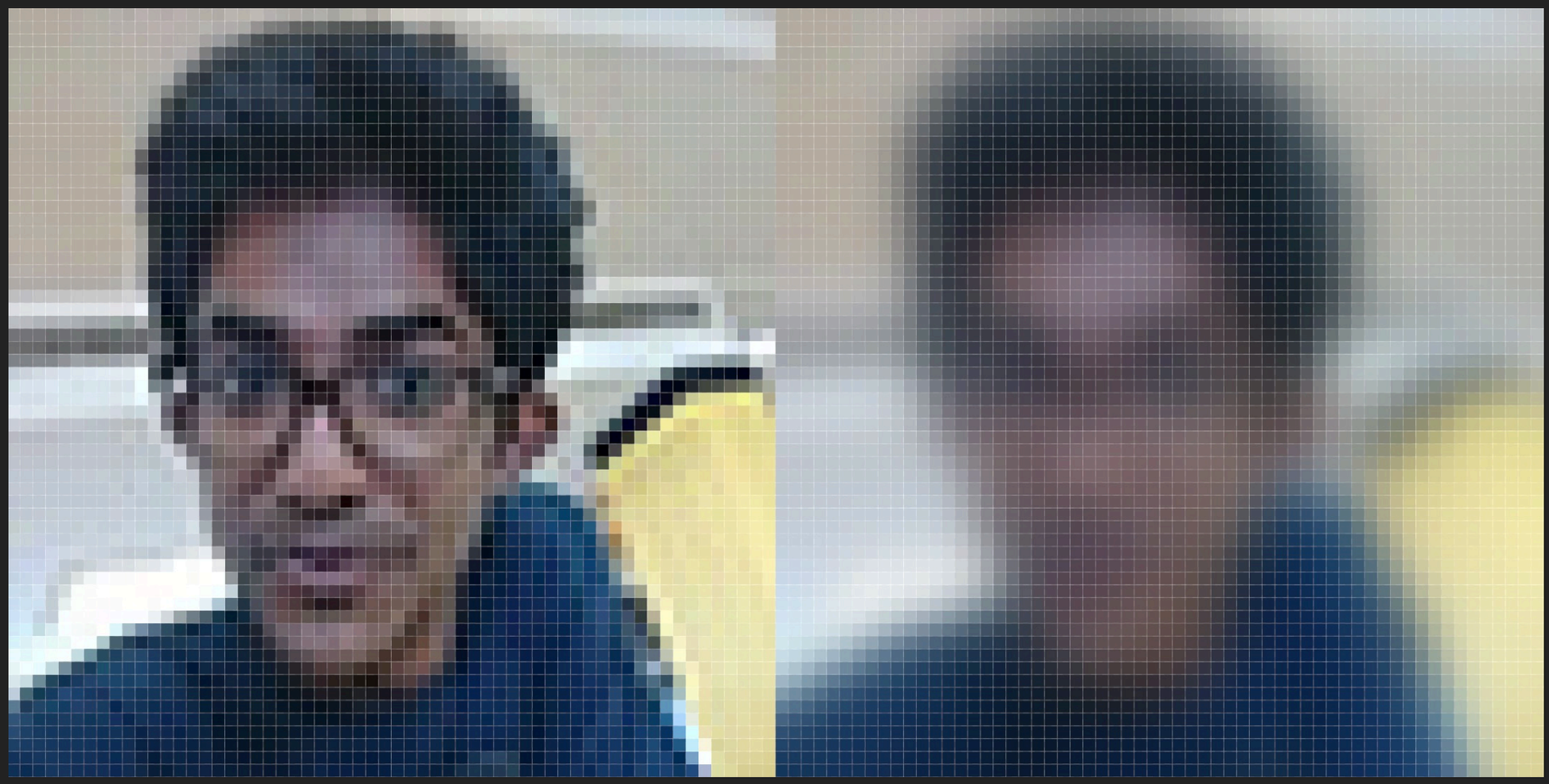


GAUSSIAN FILTER



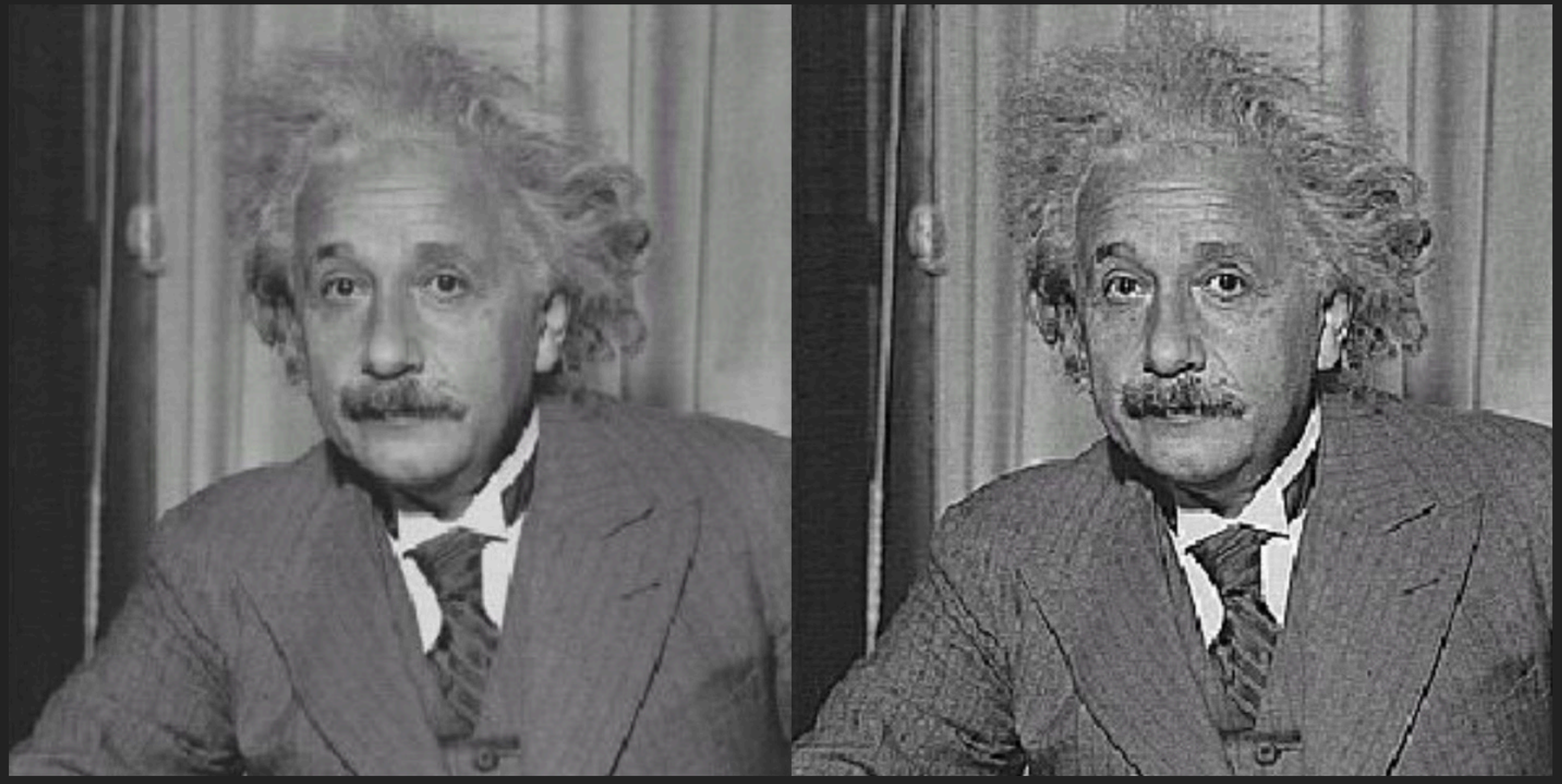
$$h_\sigma(u,v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}}$$

$$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

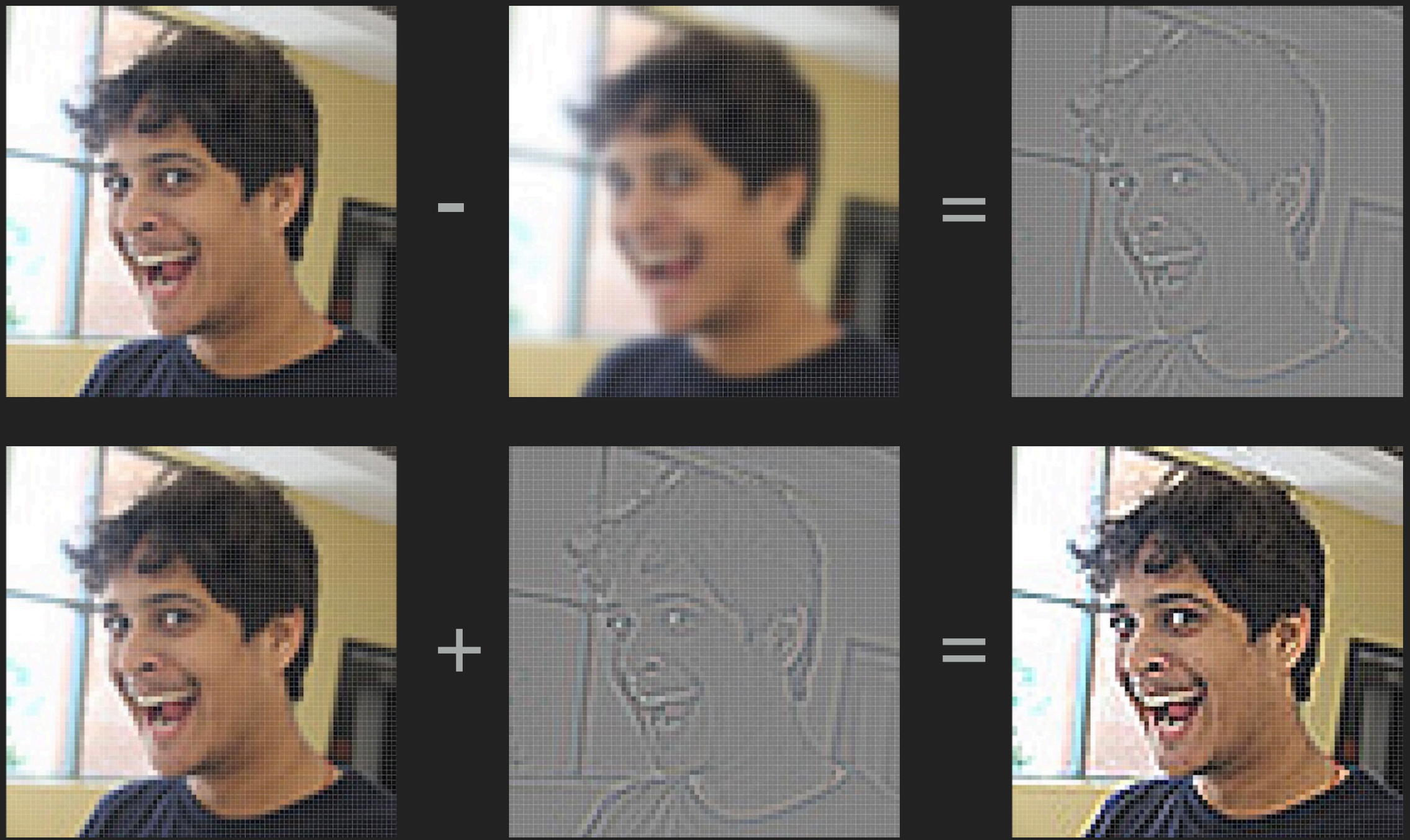


SHARPENING

-1	-1	-1
-1	9	-1
-1	-1	-1



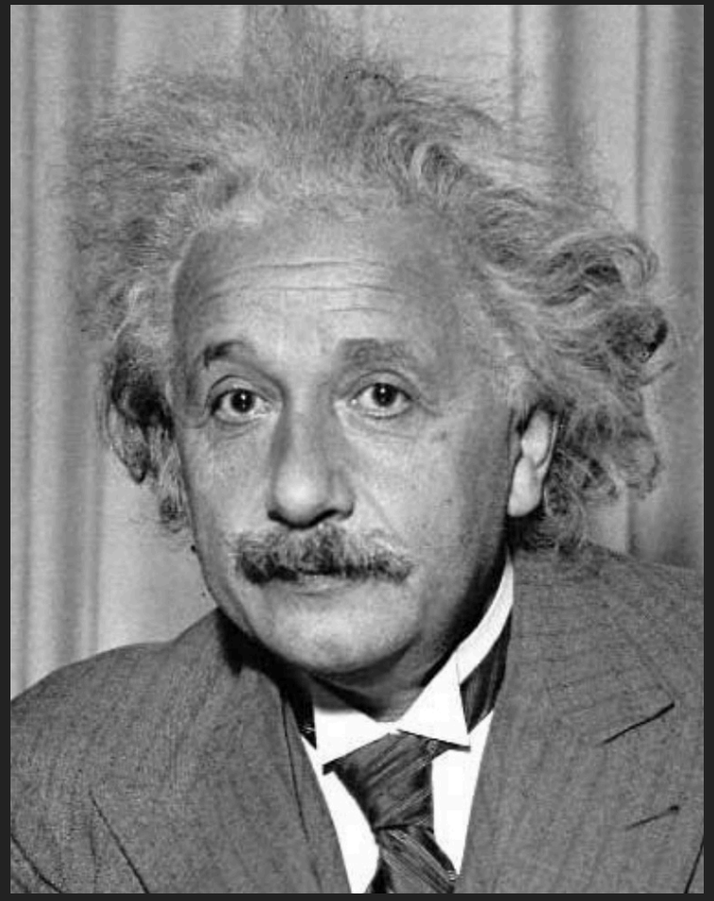
OPERATIONS ON FILTERS



SOBEL FILTER

SOBEL X

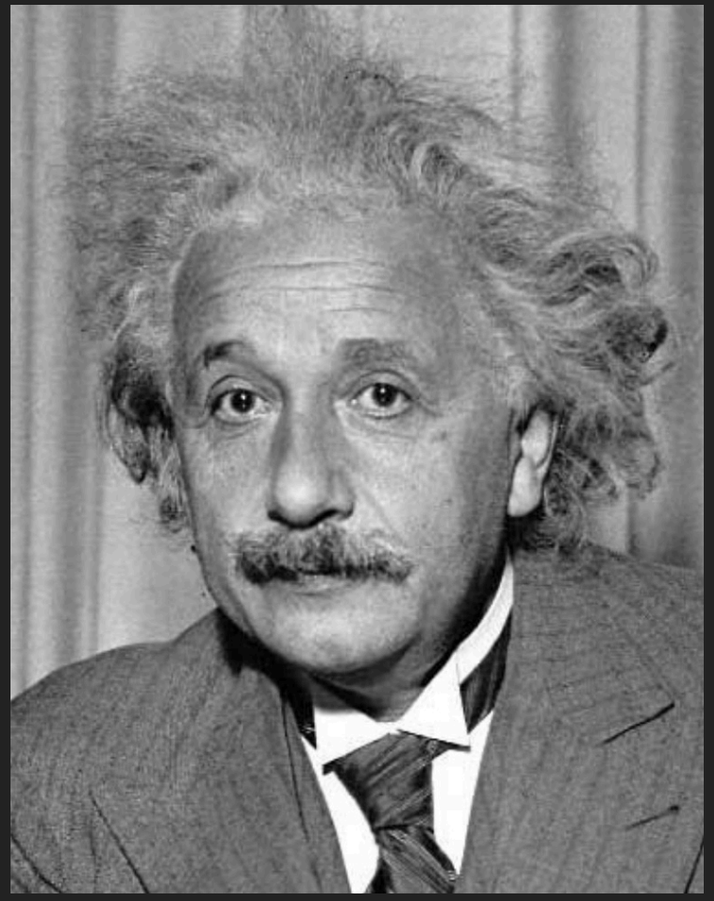
1	2	1
0	0	0
-1	-2	-1



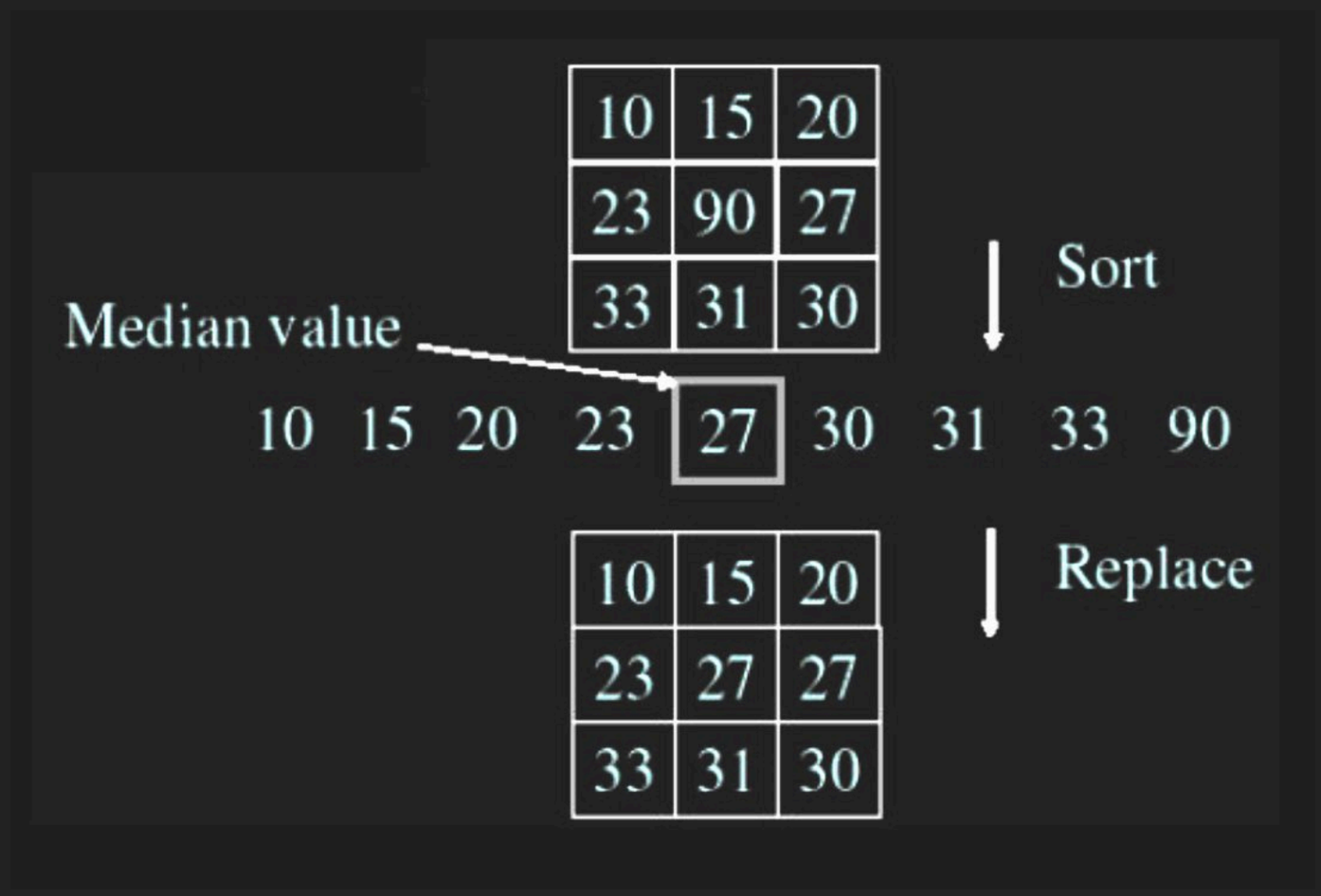
SOBEL FILTER

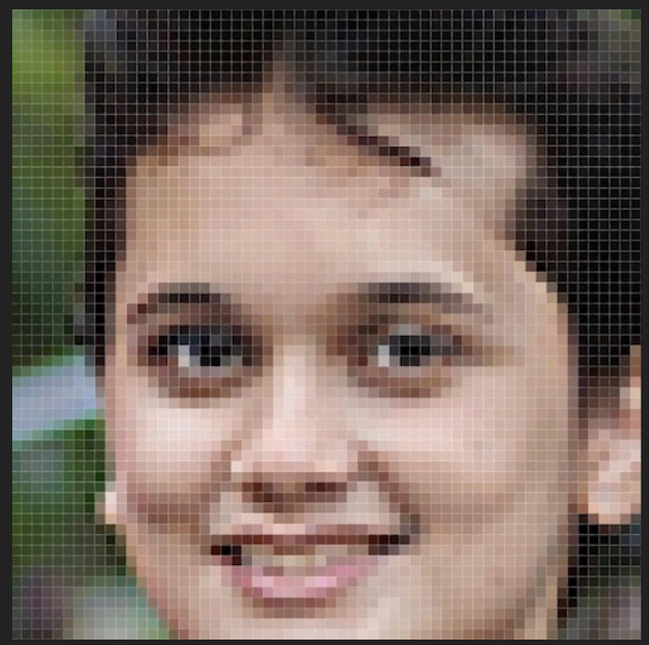
SOBEL Y

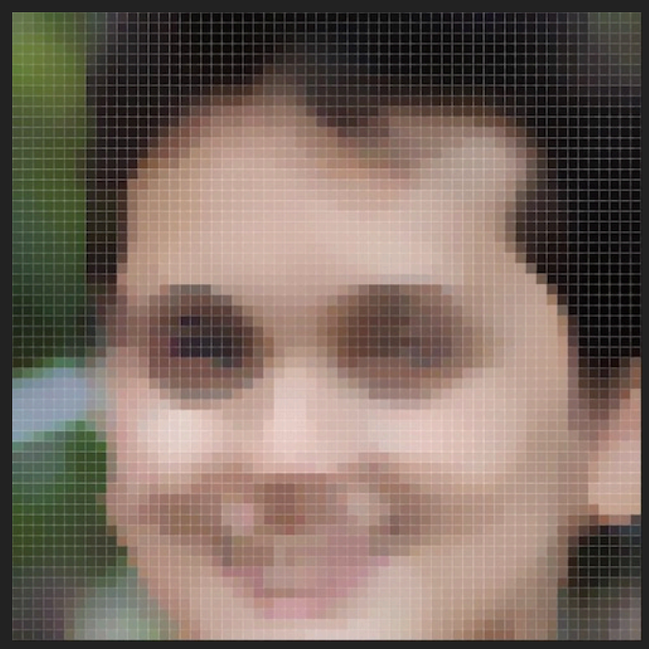
1	0	-1
2	0	-2
1	0	-1

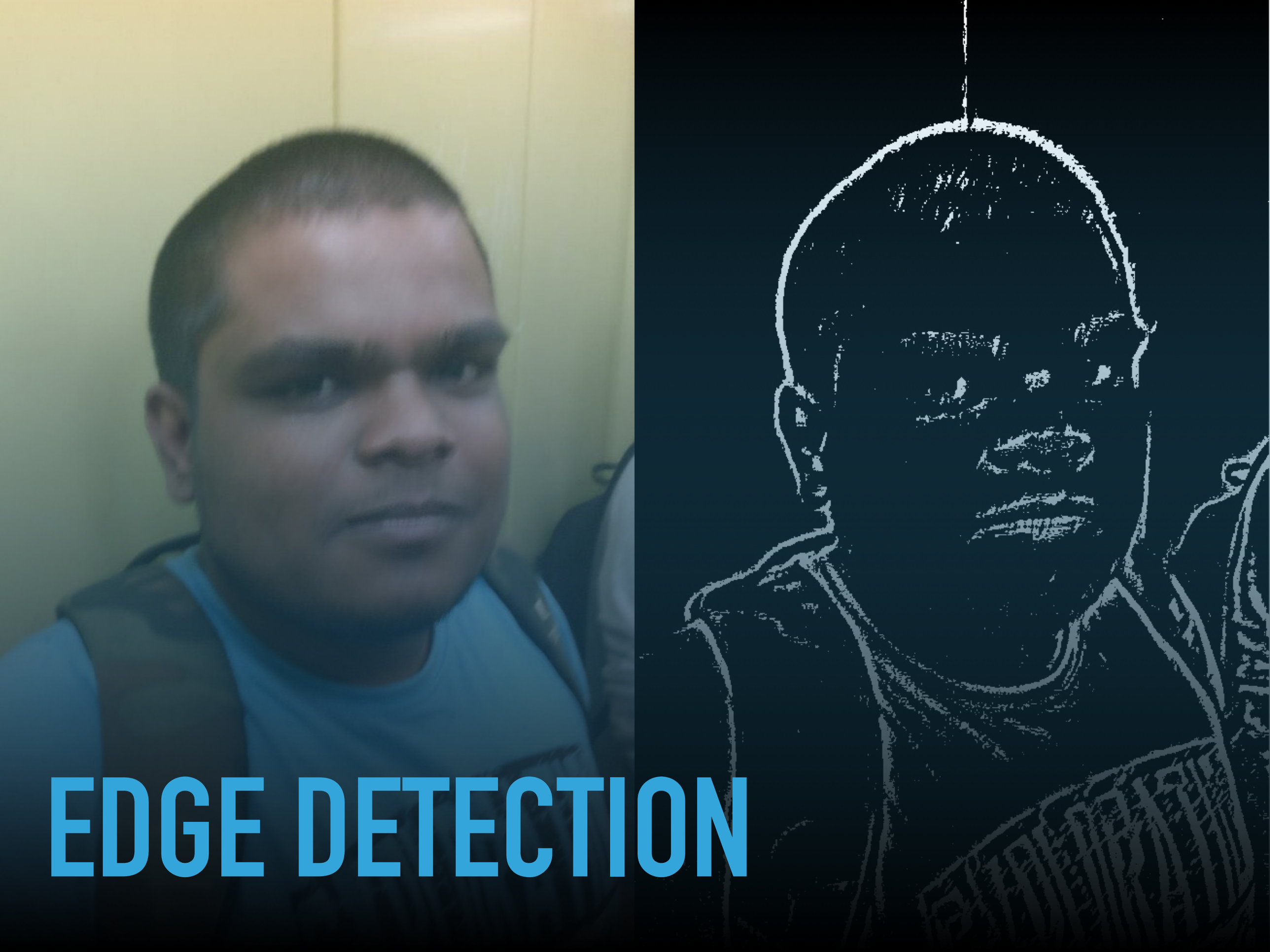


MEDIAN FILTER









EDGE DETECTION

EDGE DETECTION

- Identify sudden changes (discontinuities) in an image.
- Intuitively, most semantic and shape information from the image can be encoded in the edge.
- Edges extract information and and recognize edges
- Recover geometry and viewpoint

WHAT EXACTLY IS AN EDGE

An edge is a place of rapid change in the image intensity function

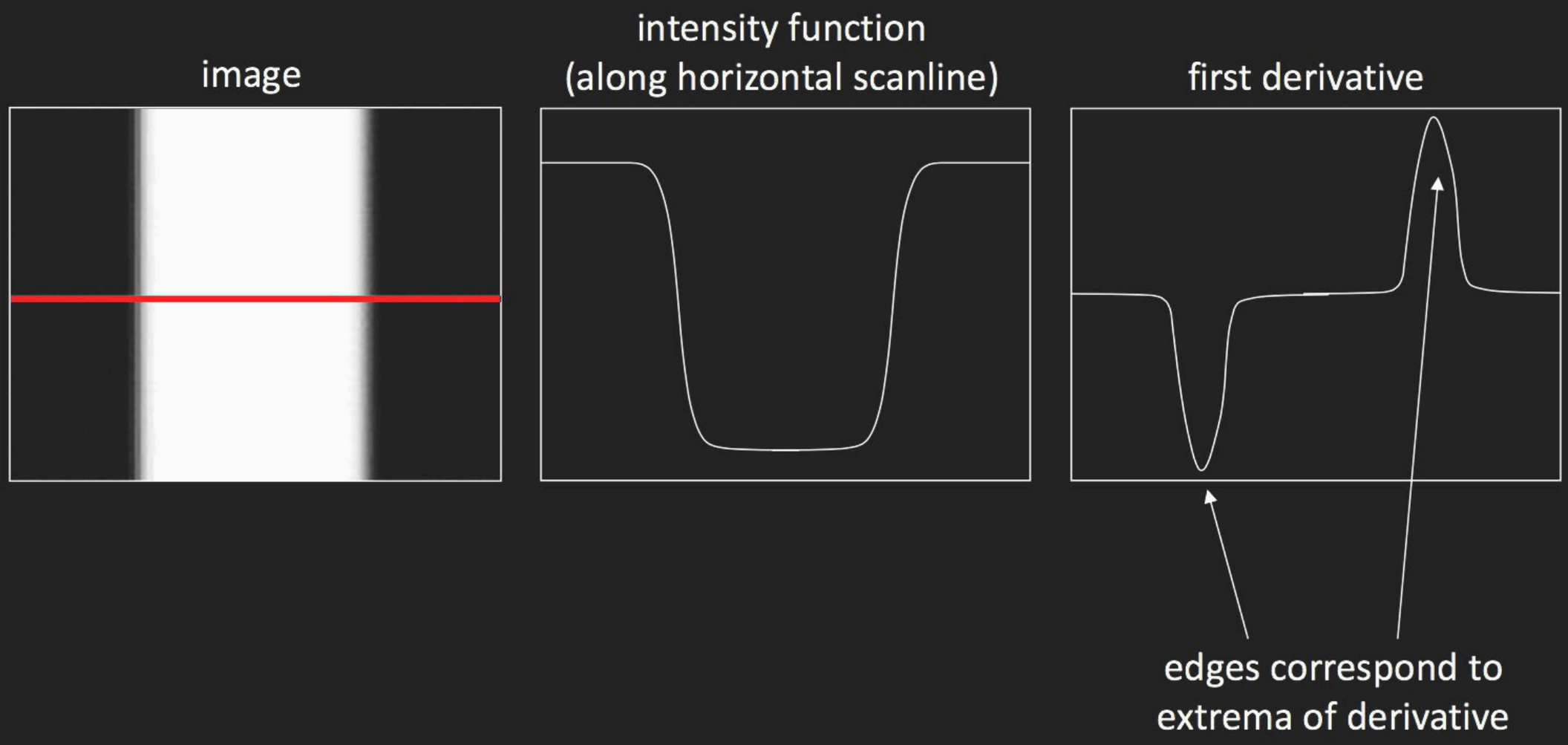
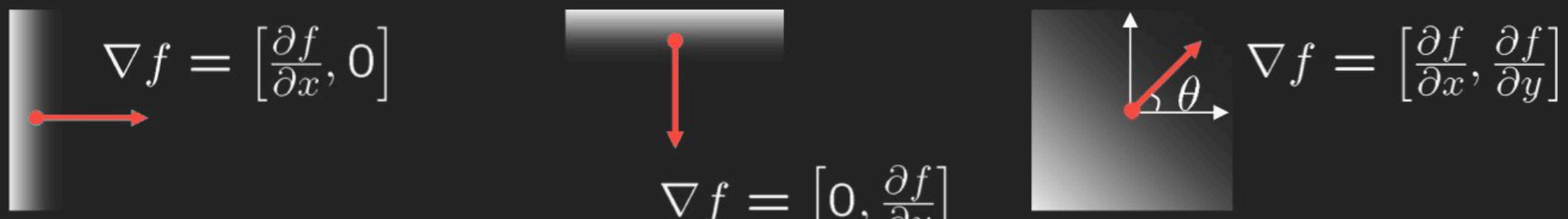


IMAGE GRADIENT

- The gradient of an image: $\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$



The gradient points in the direction of most rapid increase in intensity

- How does this direction relate to the direction of the edge?

The gradient direction is given by $\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$

The *edge strength* is given by the gradient magnitude

$$\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

Source: Steve Seitz

CANNY EDGE DETECTOR

- Smooth image with a *Gaussian*
  - optimizes the trade-off between noise filtering and edge localization
- Compute the *Gradient* magnitude using approximations of partial derivatives. (2x2 filters)
- *Thin edges* by applying non-maxima suppression to the gradient magnitude.
- Detect edges by *double thresholding*

GRADIENT

- Convolve with Gx and Gy at each point.
- Magnitude and gradients are computed

$$G_x = \begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix}$$

$$G_y = \begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix}$$

$$M[i,j] = \sqrt{P[i,j]^2 + Q[i,j]^2}$$

$$\theta[i,j] = \tan^{-1}(Q[i,j], P[i,j])$$

- Floats are dropped for faster computation

NON MAXIMA SUPPRESSION

- Thin edges by keeping values of gradient
  - Not always at the location of the edge
  - There are many thick edges

GRADIENT ORIENTATION

- Get the four basic directions of the edges orientation: horizontal, vertical, 45° and -45°.
- Find the direction closest to α(x, y).
- If the value of $M(x, y)$ is less than at least one of its two neighbors along d_k, let $g_N(x, y) = 0$ (suppression); otherwise, let $g_N(x, y) = M(x, y)$

THRESHOLDING

- Final operation is to threshold $g_N(x, y)$ to reduce false edge points.

Hysteresis thresholding:

$$g_{NH}(x, y) = g_N(x, y) \geq T_H$$

$$g_{NL}(x, y) = g_N(x, y) \geq T_L$$

and

$$g_{NL}(x, y) = g_{NL}(x, y) - g_{NH}(x, y)$$

gradient X

gradient Y

gradient magnitude

Before non maxima suppression

After non maxima suppression

CANNY EDGE DETECTOR

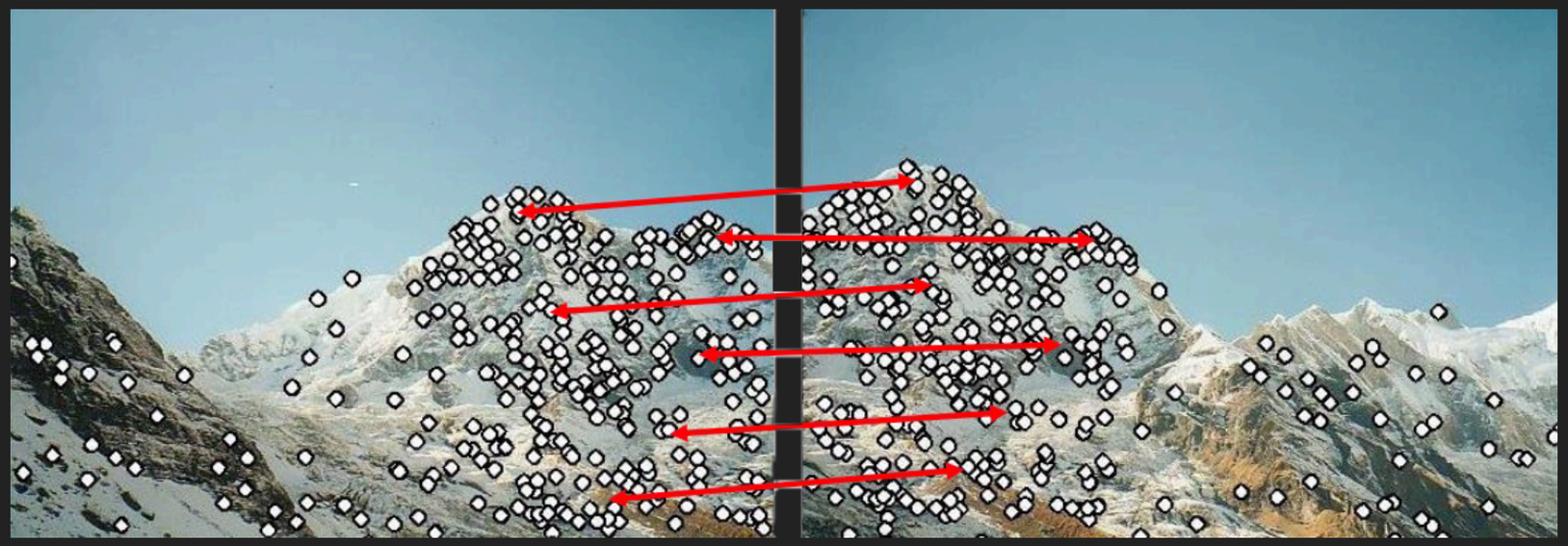
FEATURE DETECTION

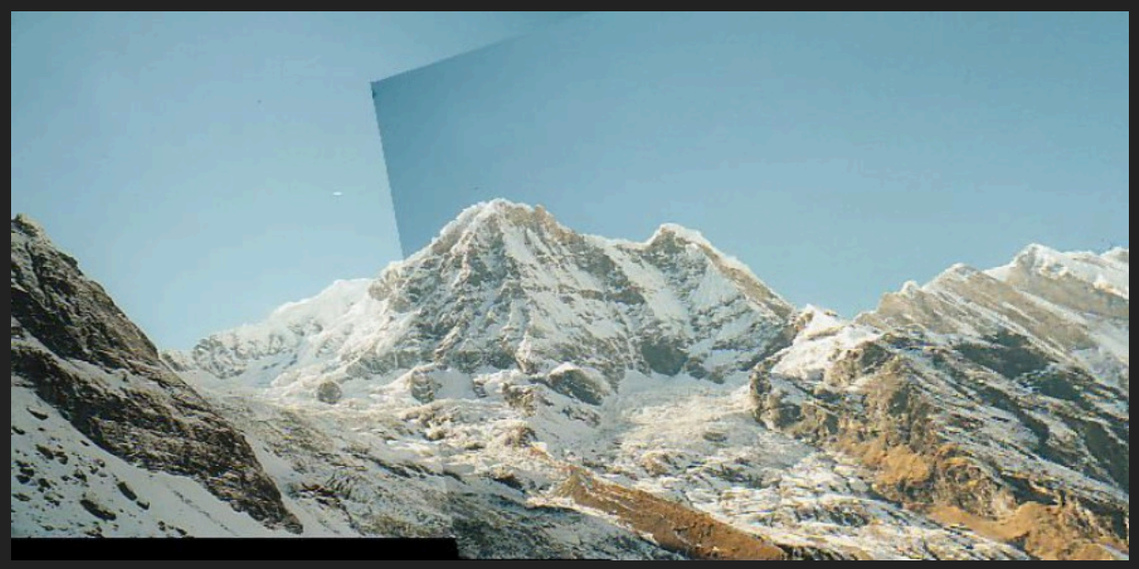
GENERAL STEPS

- Find a set of distinctive key points
- Define a region around each key point
- Extract and normalize the region content
- Compute a local descriptor from a normalized region
- Match local descriptors

IMAGE STITCHING

- Extract features, combine feature, and align image





HARRIS DETECTOR FORMULATION

Change of intensity for the shift [u,v]:

$$E(u,v) = \sum_{x,y} w(x,y)\left[I(x+u,y+v) - I(x,y)\right]^2$$

- Window function
- Shifted intensity
- Intensity

- Window function w(x, y) can be linear or gaussian

This measure of change can be approximated by:

$$E(u,v) \approx [u \;\; v] \; M \begin{bmatrix} u \\ v \end{bmatrix}$$

where M is a 2 ? 2 matrix computed from image derivatives:

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

Gradient with respect to x, times gradient with respect to y

Sum over image region – the area we are checking for corner

$$M = \begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} = \sum \begin{bmatrix} I_x \\ I_y \end{bmatrix} [I_x \; I_y]$$

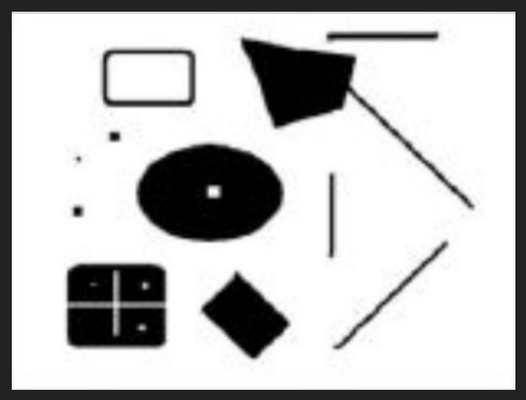
where M is a 2 ? 2 matrix computed from image derivatives:

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

Gradient with respect to x, times gradient with respect to y

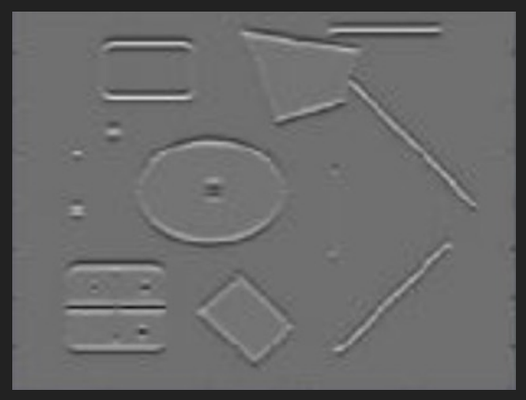
Sum over image region – the area we are checking for corner

$$M = \begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} = \sum \begin{bmatrix} I_x \\ I_y \end{bmatrix} [I_x \; I_y]$$

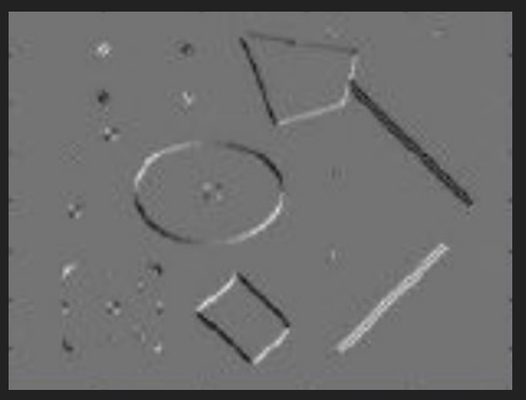


I

I_X

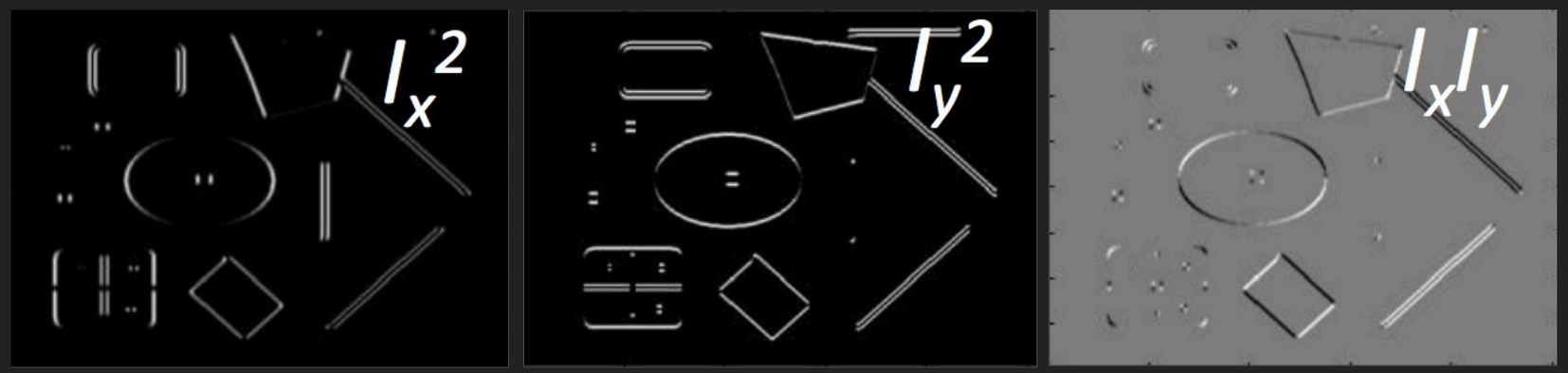


I_Y



$I_X I_Y$

- Gaussian filter $g(\sigma)$ on I_x^2, I_y^2, and $I_x I_y$.



- Cornerness function – both eigenvalues are strong

$$har = \det[M(\sigma_I, \sigma_D)] - \alpha[\text{trace}(M(\sigma_I, \sigma_D))^2]$$
$$= g(I_x^2)g(I_y^2) - [g(I_x I_y)]^2 - \alpha[g(I_x^2) + g(I_y^2)]^2$$

THANK YOU!

@sidkothiyal

@avikantz