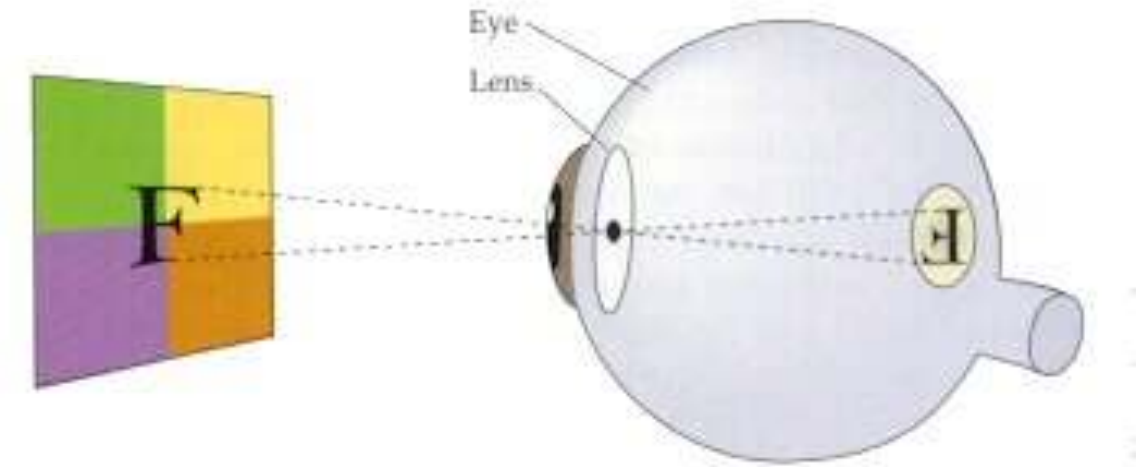
Image Formation (Cont'd)
&
Edge Detection

P. Fua
(Taught by M. Salzmann)
IC-CVLab
EPFL

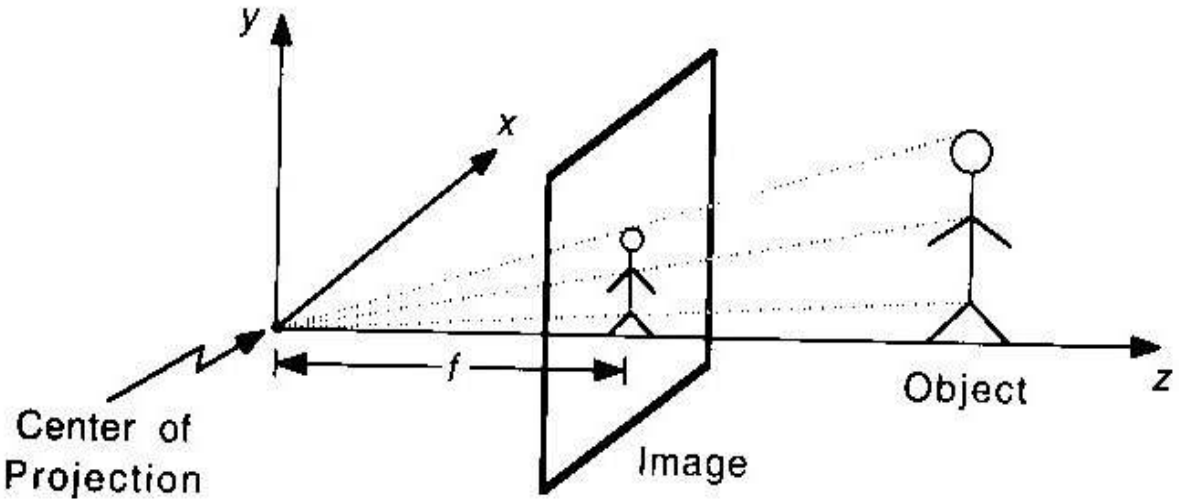
Reminder: Image Formation



Projection from surfaces to 2-D sensor.

- Where: Geometry
- How bright: Radiometry
- Stored how: Sensing

Reminder: Pinhole Camera Model



In pixels

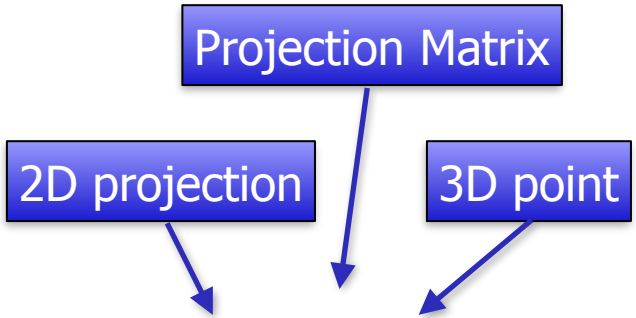
$$u \propto x_i = f\frac{x_c}{z_c}$$

$$v \propto y_i = f\frac{y_c}{z_c}$$

In meters

→Reformulate it as a linear operation using homogeneous coordinates.

Reminder: Projection in Homogeneous Coordinates



$$\mathbf{x} = \mathbf{P}\mathbf{X}$$

$$\mathbf{P} = \mathbf{K} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \mathbf{Rt}$$

$$\text{with } \mathbf{K} = \begin{bmatrix} \alpha_u & s & p_u \\ 0 & \alpha_v & p_v \\ 0 & 0 & 1 \end{bmatrix}, \mathbf{Rt} = \begin{bmatrix} \mathbf{R} & \mathbf{T} \\ \mathbf{0} & 1 \end{bmatrix}, \text{ and } \mathbf{R}^T\mathbf{R} = \mathbf{I}\,.$$

Intrinsics

Extrinsics

Hartley, Chap 6.

Reminder: Camera Calibration

Internal Parameters:

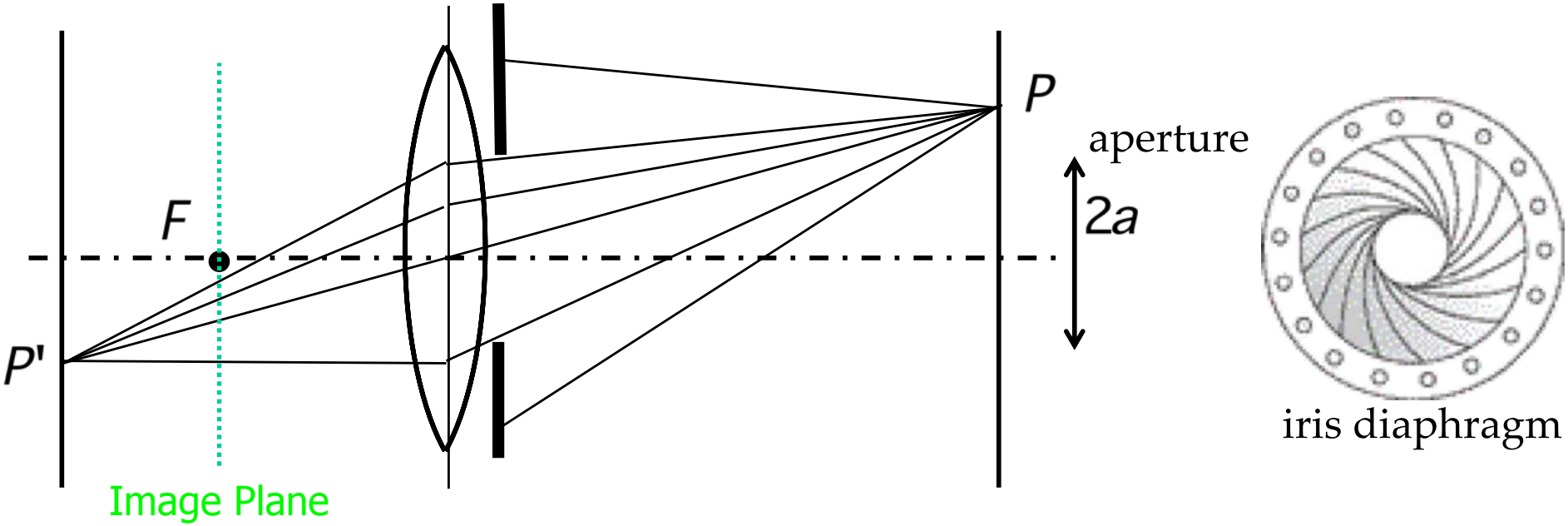
- Horizontal and vertical scaling (2)
- Principal points (2)
- Skew of the axis (1)

External Parameters:

- Rotations (3)
- Translations (3)

→There are 11 free parameters to estimate. This is known as **calibrating** the camera.

Reminder: Thin Lens



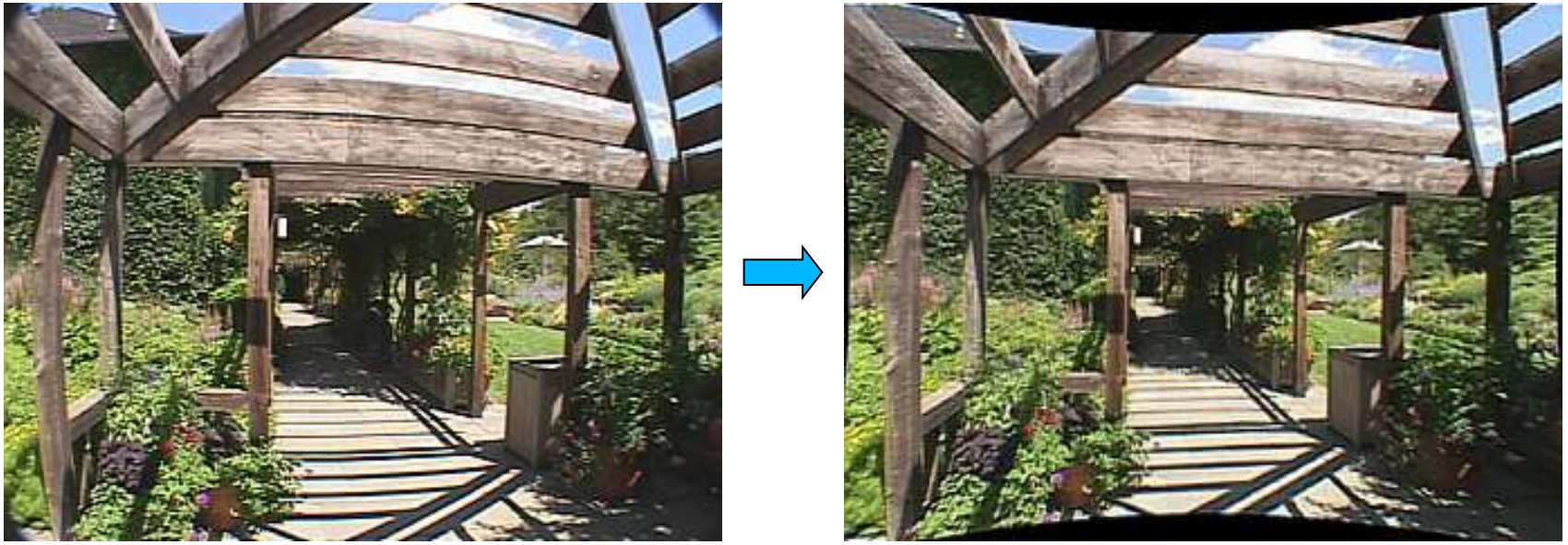
- Diameter d=2a of the lens that is exposed to light.
- The image plane is not located exactly where the rays meet.
- The greater a, the more blur there will be.

Reminder: Distortions

The lens is not exactly a "thin lens:"

- Different wave lengths are refracted differently,
- Barrel Distortion.

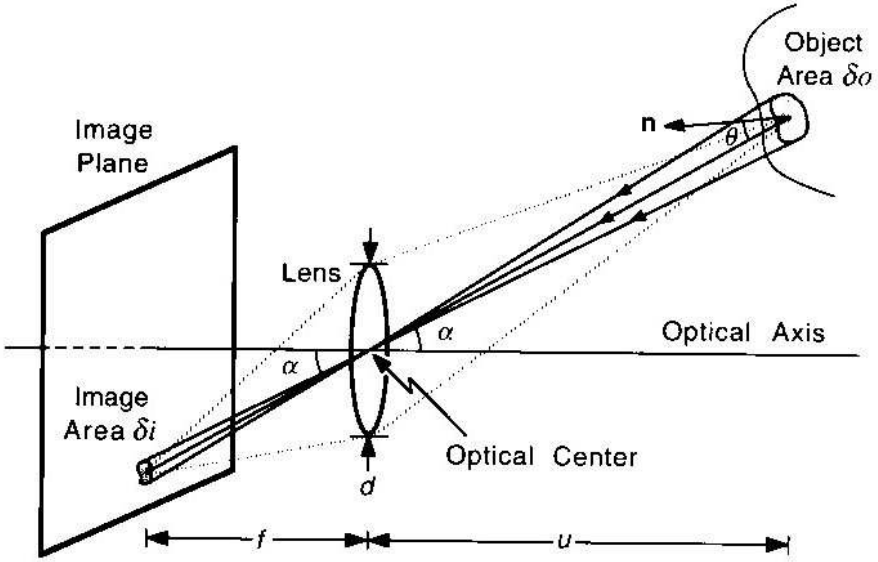
Undistorting



Once the image is undistorted, the camera projection can be formulated as a projective transform.

→ The pinhole camera model applies.

Fundamental Radiometric Equation



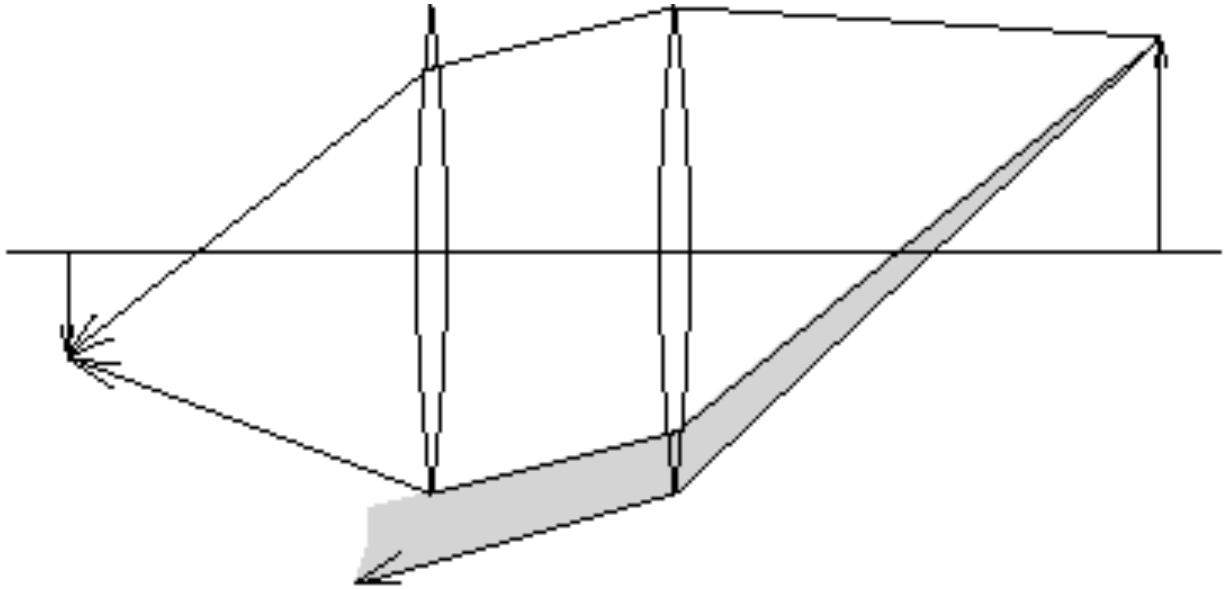
Scene Radiance (Rad) : Amount of light radiation emitted from a surface point (Watt / m2 / Steradian).

Image Irradiance (Irr): Amount of light incident at the image of the surface point (Watt / m2).

$$\text{Irr} = \frac{\pi}{4}\left(\frac{d}{f}\right)^2 \cos^4(\alpha)\,\text{Rad}\,,$$

$$\Rightarrow \text{Irr} \propto \text{Rad for small values of } \alpha\,.$$

Vignetting



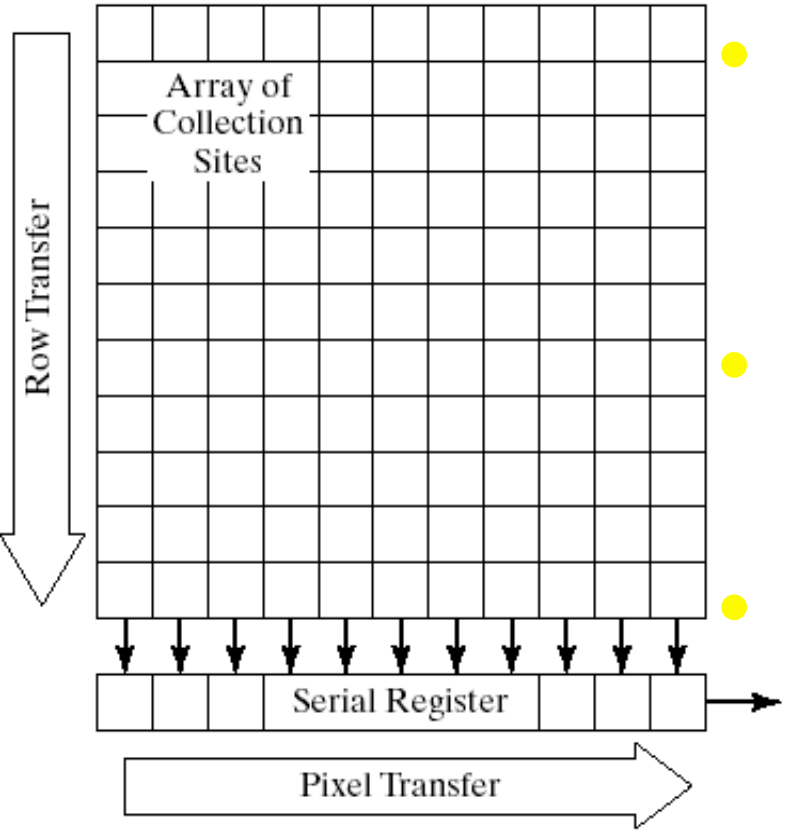
Images can get darker towards their edges because some of the light does not go through all the lenses.

De Vignetting

—> As for geometric undistortion, undo vignetting to create an image that an ideal camera would have produced.

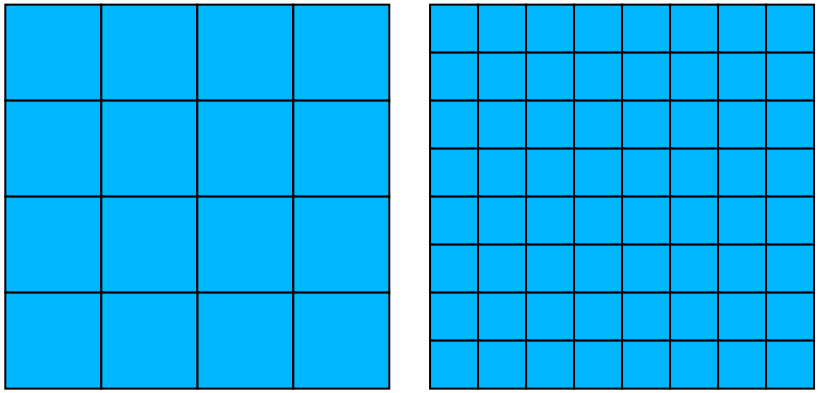
Zheng et al., CVPR'08

Sensor Array



- Photons free up electrons that are then captured by a potential well.
- Charges are transferred row by row wise to a register.
- Pixel values are read from the register.

Sensing



Conversion of the "optical image" into an "electrical image":

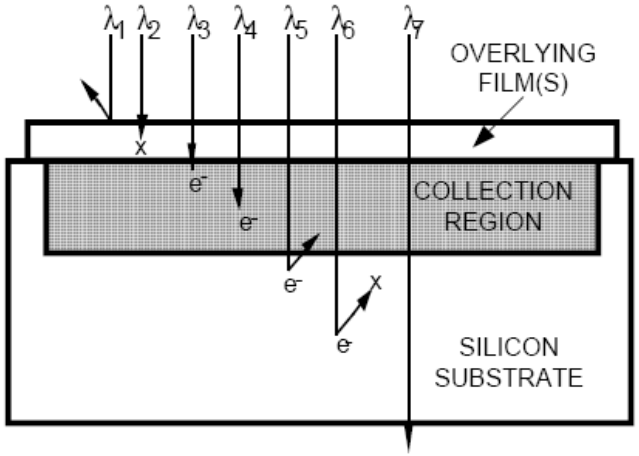
$$E(x,y) = \int_{t_0}^{t_1} \int_{0}^{\Lambda} \mathrm{Irr}(x,y,t,\lambda) s(\lambda) dt d\lambda$$

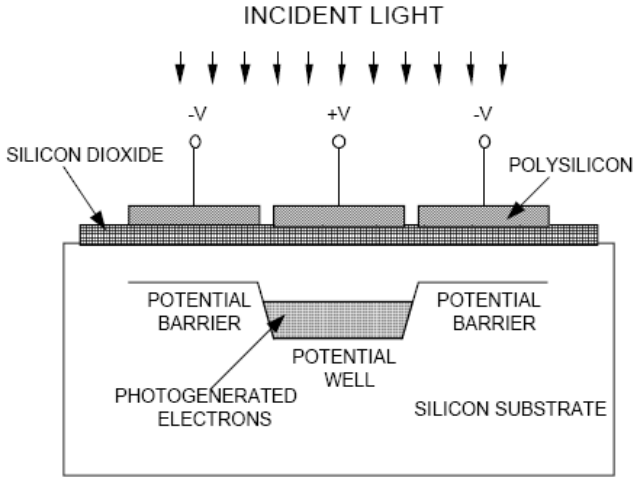
$$I(m,n) = \mathrm{Quantize}\left(\int_{x_0}^{x_1} \int_{y_0}^{y_1} E(x,y) dx dy \right)$$

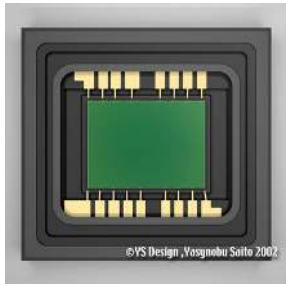
→ Quantization in

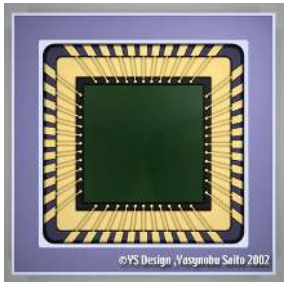
- Time
- Space

Sensors









CCD

CMOS

- Charged Coupling Devices (CCD): Made through a special manufacturing process that allows the conversion from light to signal to take place in the chip without distortion.
- Complimentary Metal Oxide Semiconductor (CMOS): Easier to produce and similar quality. Now used in most cameras except when quantum efficient pixels are needed, e.g. for astronomy.

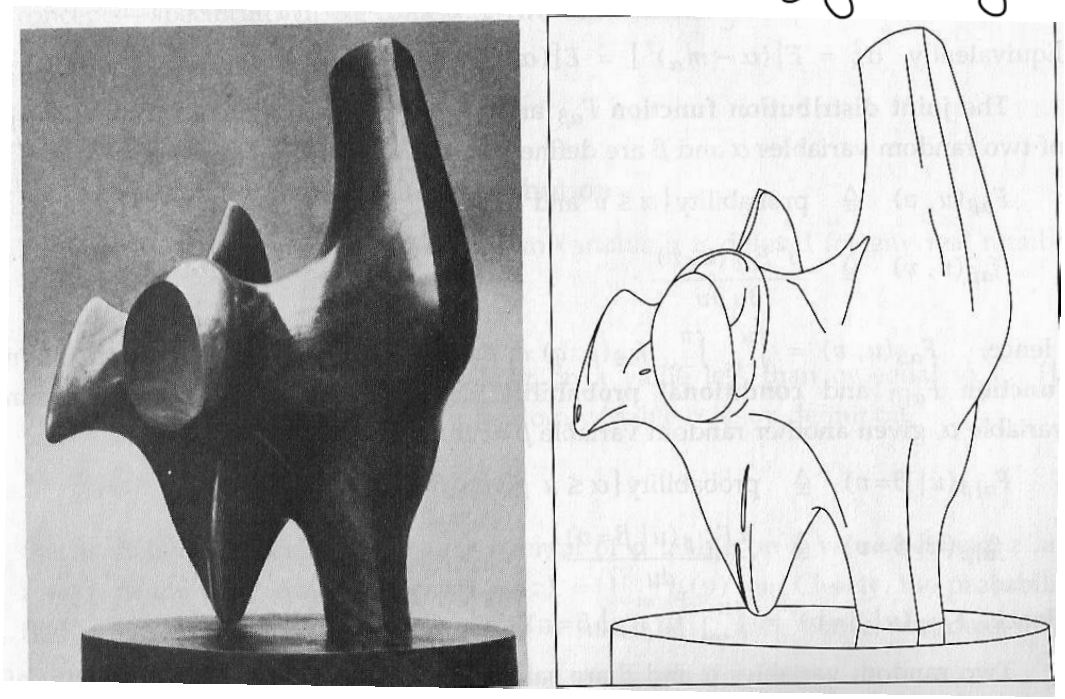
In Short

- Camera geometry can be modeled in terms of the pinhole camera model, which is linear in projective space.

- Image radiance is roughly proportional to surface radiance and the two can be used interchangeably for our purposes.

Edge Detection

salient contours



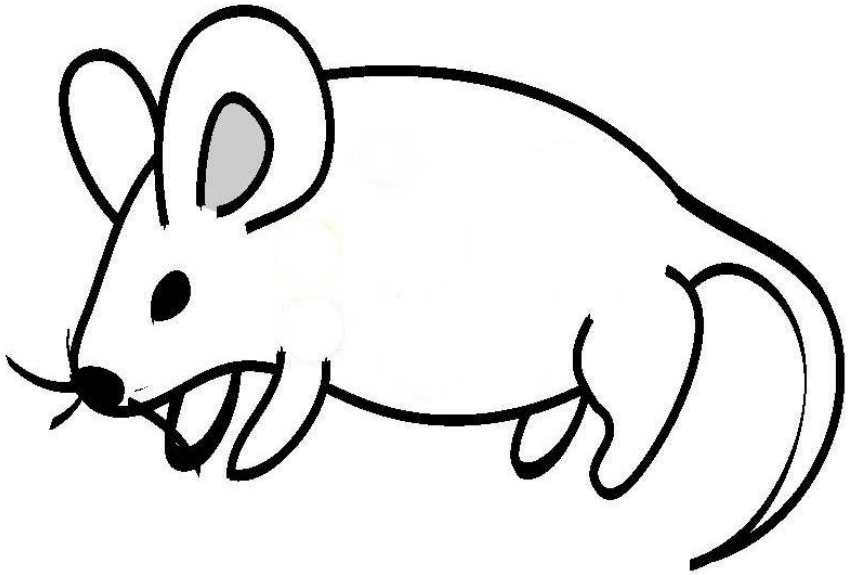
Edge image

- What's an edge
- Image gradients
- Edge operators

v info

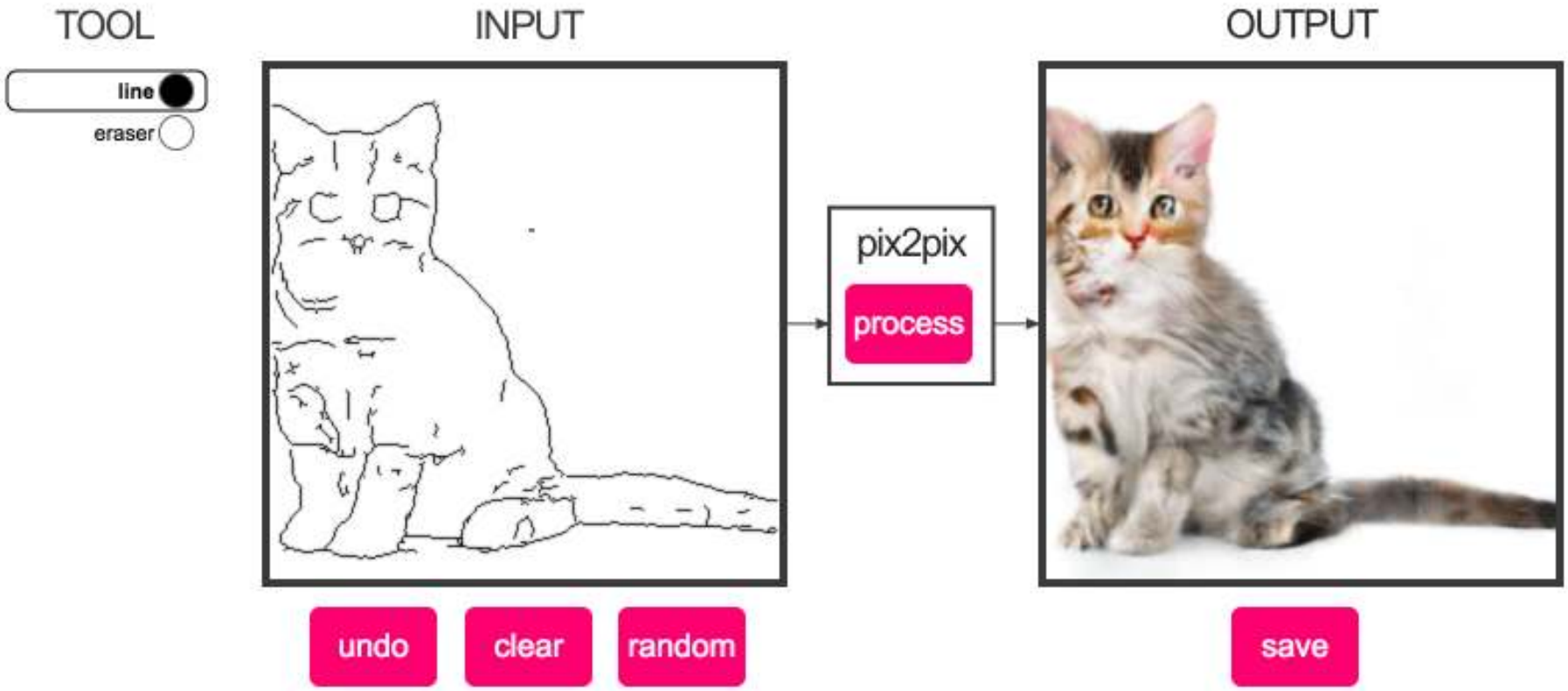
v contours carry a lot of info

Line Drawings



- Edges seem fundamental to human perception.
- They form a compressed version of the image.

From Edges To Cats



Deep-Learning based generative model.

Demo

https://affinelayer.com/pixsrv/

Maps and Overlays

roads → are contours

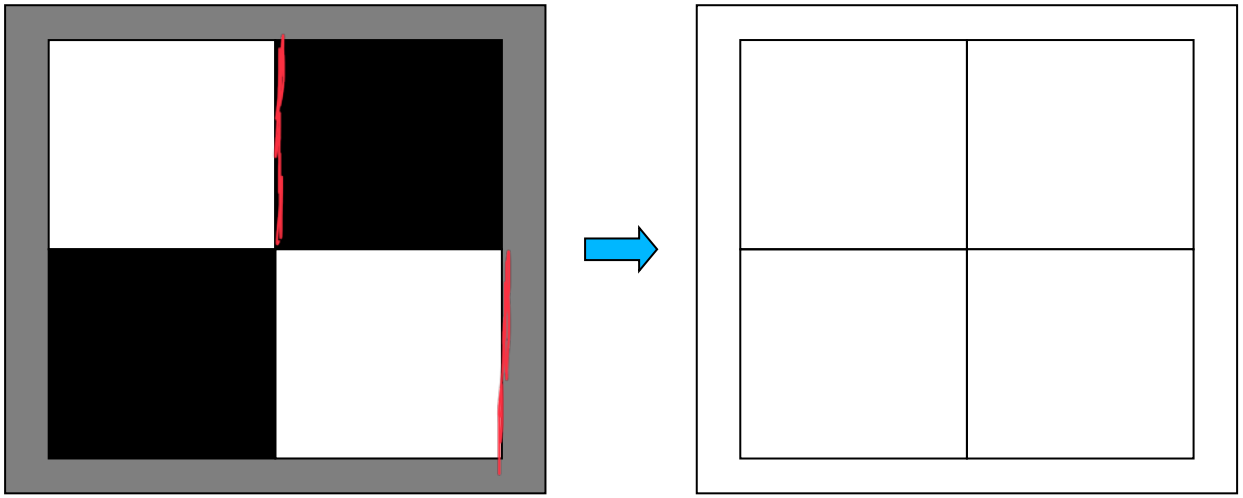
Distance/Road between A and B

Corridor

in Real-time, it calculates projection matrix so that 3D contours get projected in the right place

Corridor

Edges and Regions



Edges:

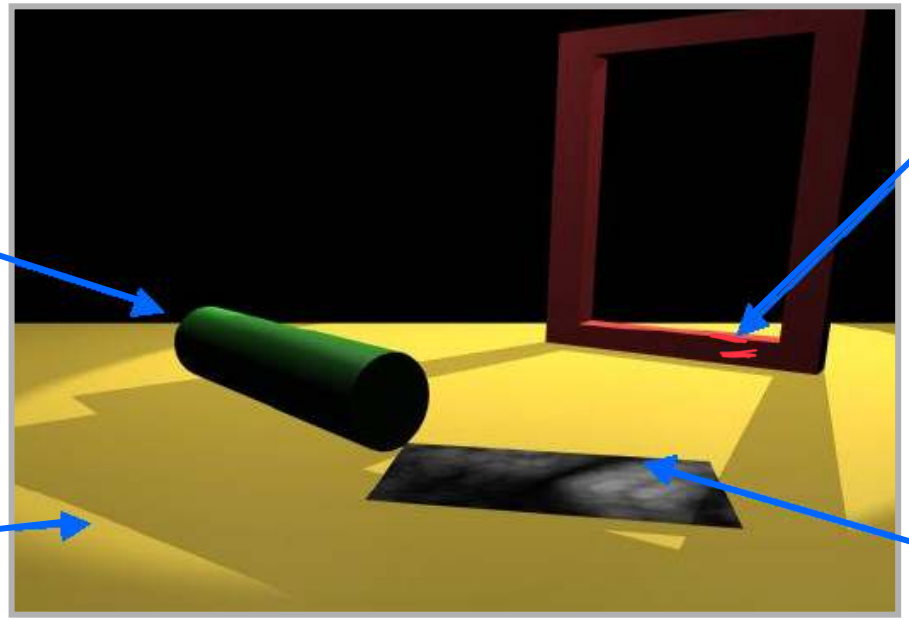
- Boundary between bland image regions.

Regions:

- Homogenous areas between edges.

→ Edge/Region Duality.

Discontinuities

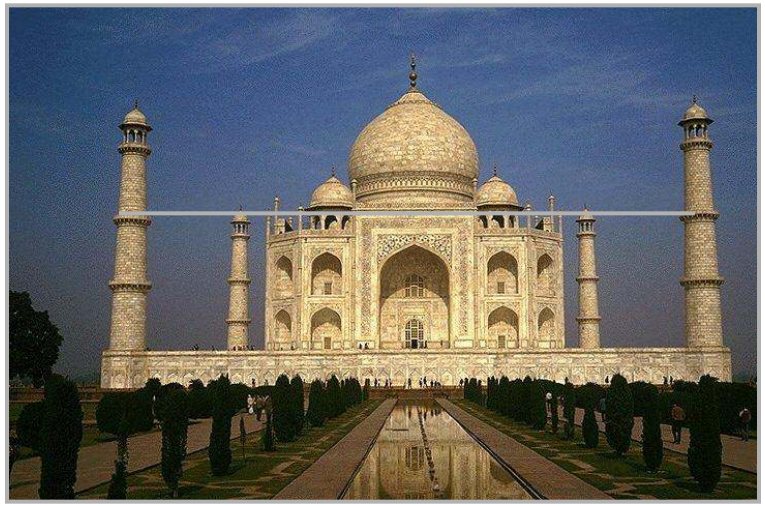


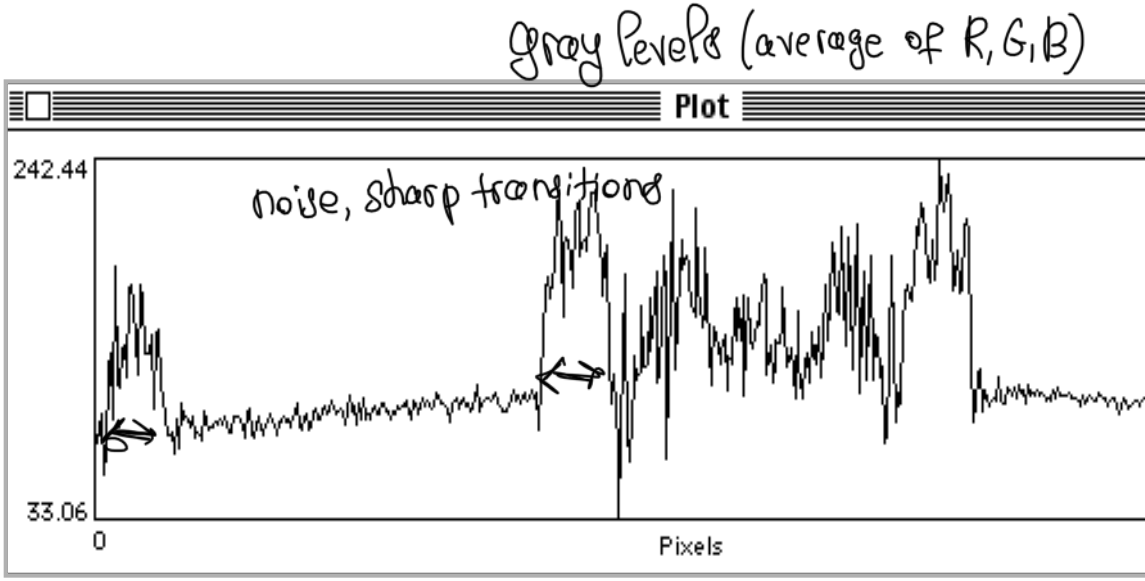
Depth changes abruptly in projection

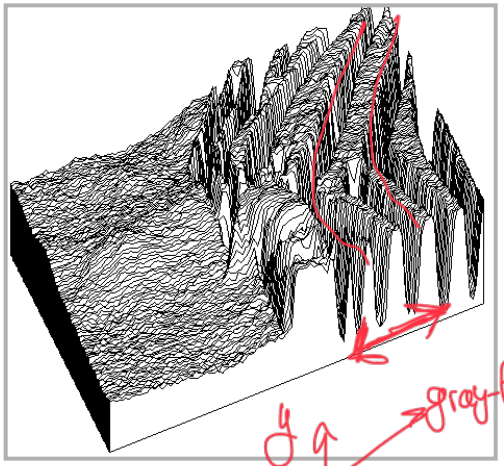
- A. Depth discontinuity: Abrupt depth change in the world
- B. Surface normal discontinuity: Change in surface orientation Change of contrast
- C. Illumination discontinuity: Shadows, lighting changes
- D. Reflectance discontinuity: Surface properties, markings

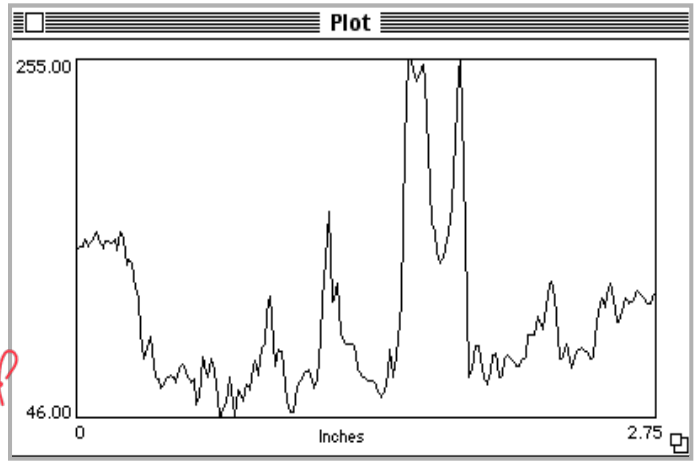
→ Sharply different Gray levels on both sides

REALITY

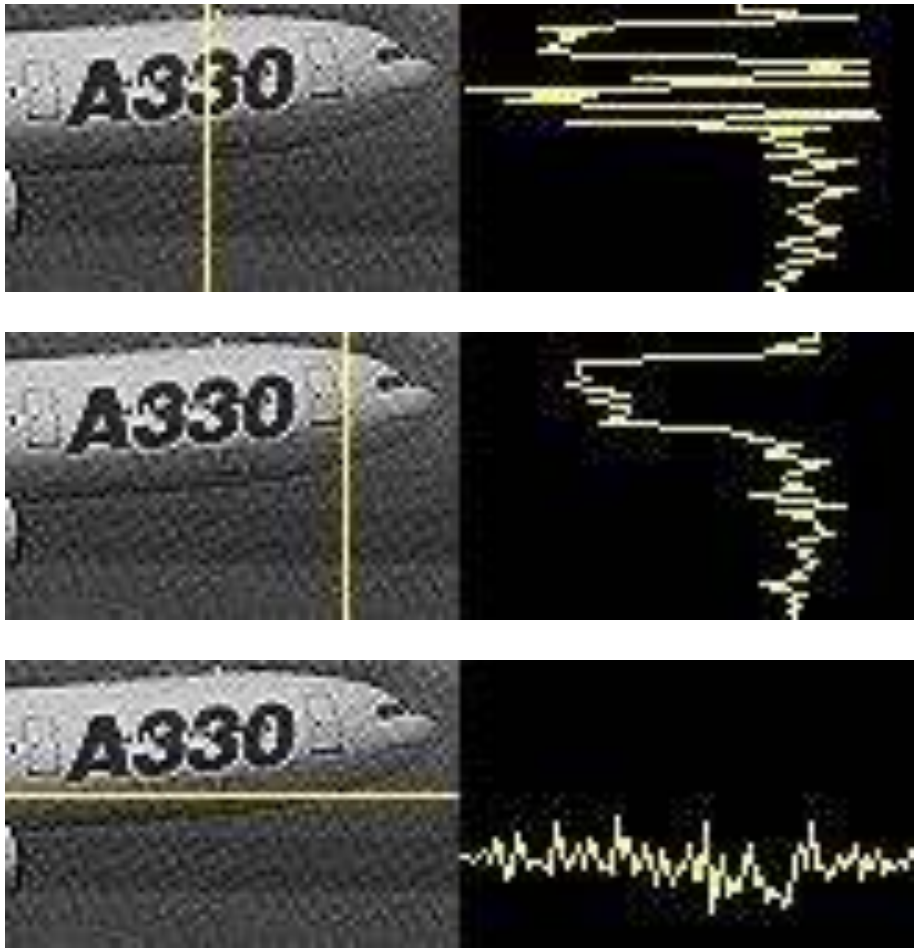



gray levels (average of R, G, B)
Plot
242.44
noise, sharp transitions
33.06
0
Pixels


gray-level
y
x


Plot
255.00
46.00
0
Inches
2.75

More Reality

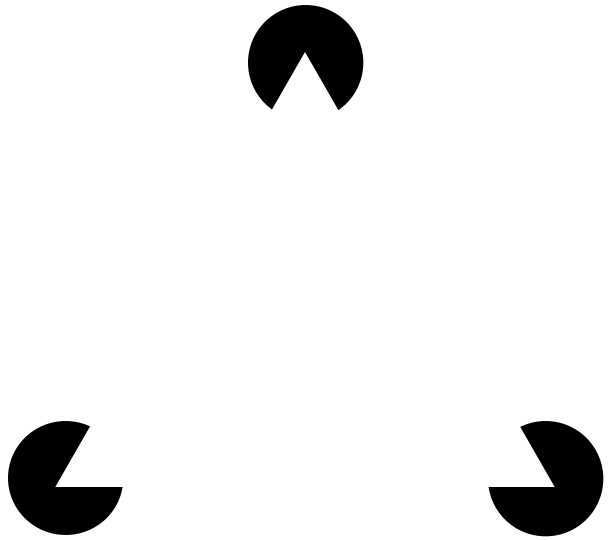


Finding meaningful physical discontinuities in noisy signals → hard

Very noisy signals

→ Prior knowledge is required!!

Optional: Illusory Contours

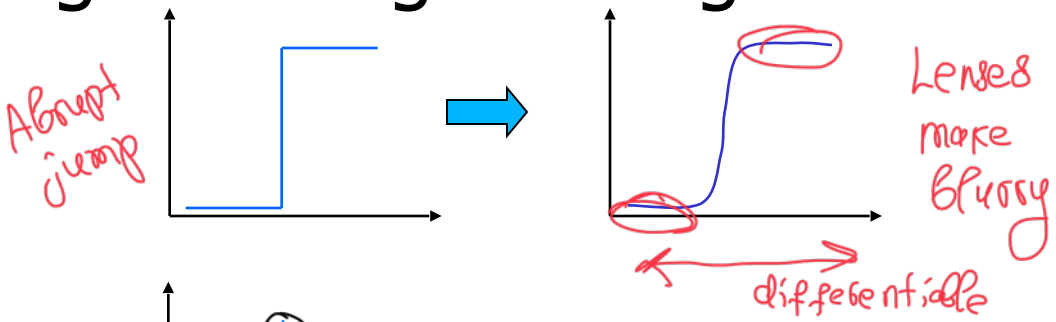


- No closed contour, but we still perceived an edge.
- This will not be further discussed in this class.

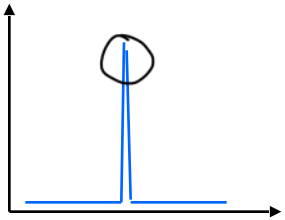
Ideal Step Edge

Rapid change in image => High local gradient

f(x) = step edge

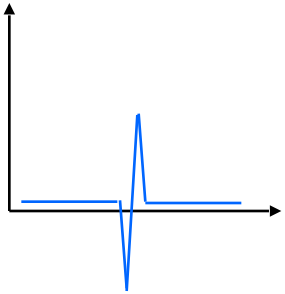


1st Derivative f'(x)



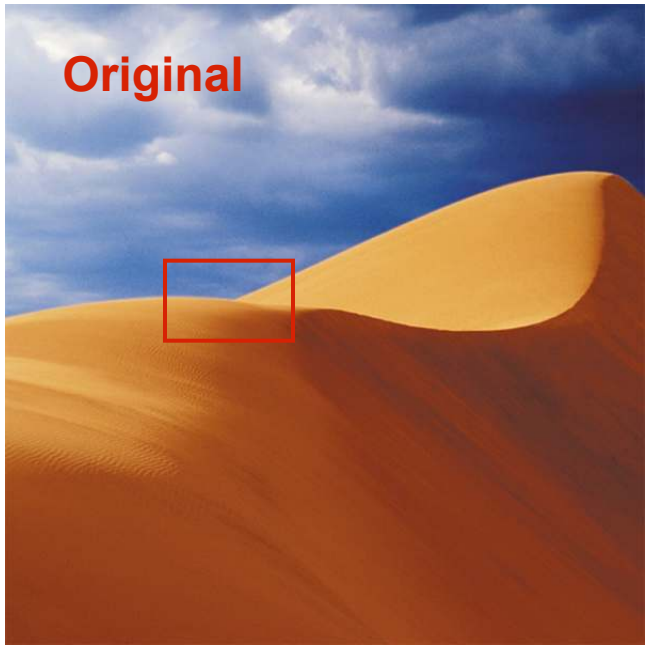
maximum of 1st derivative

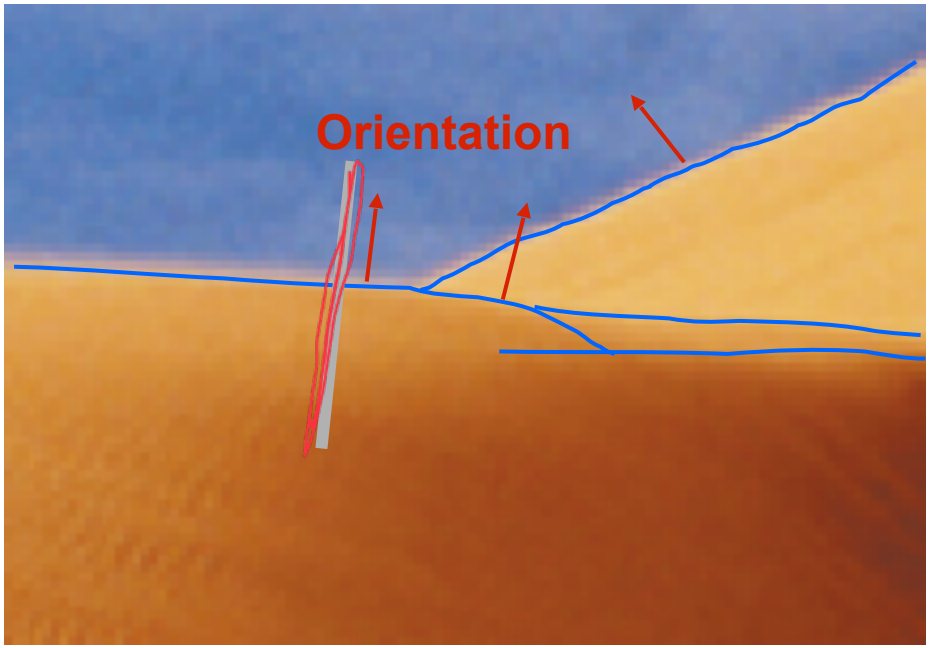
2nd Derivative f''(x)

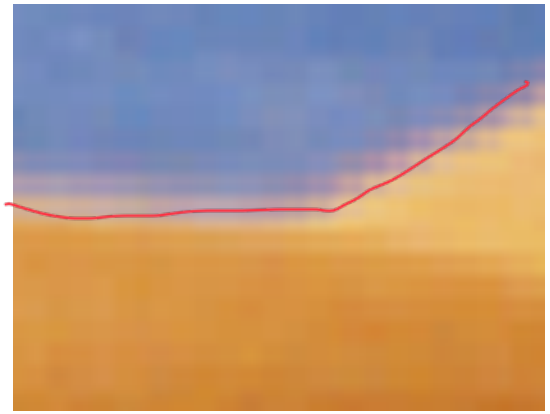


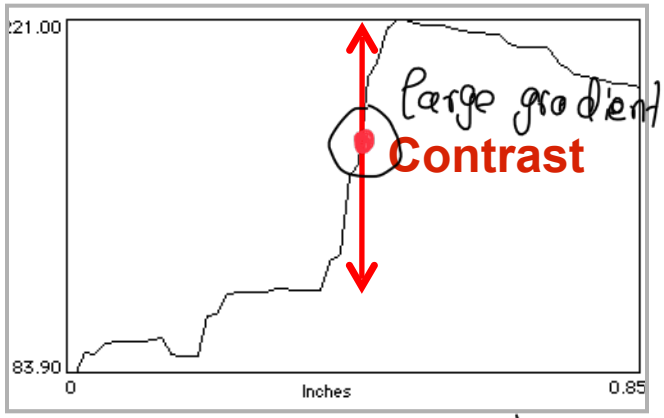
zero crossing of 2nd derivative

Edge Properties



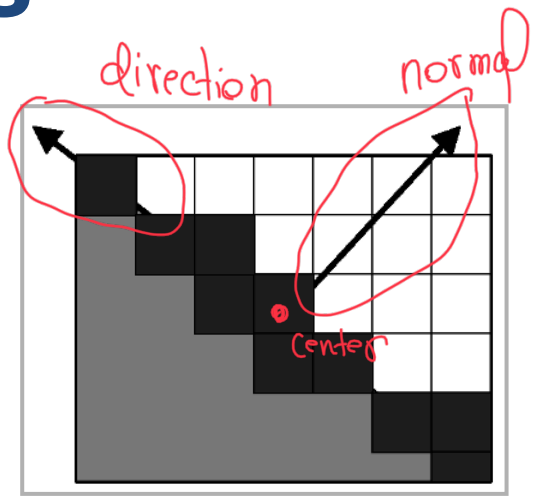






edge is located at a place where the slope in the direction of normal to a edge is largest

Edge Descriptors

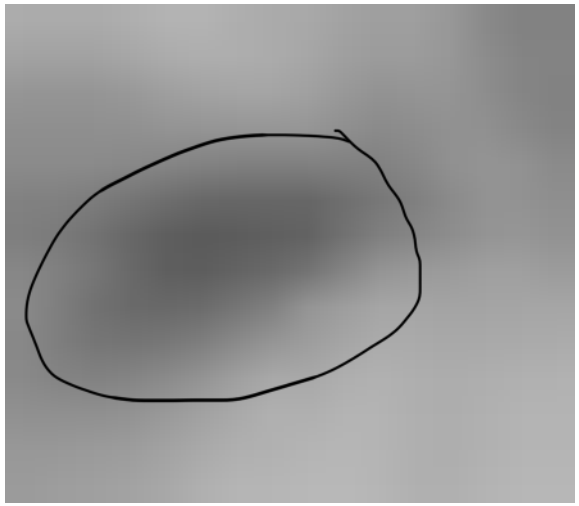


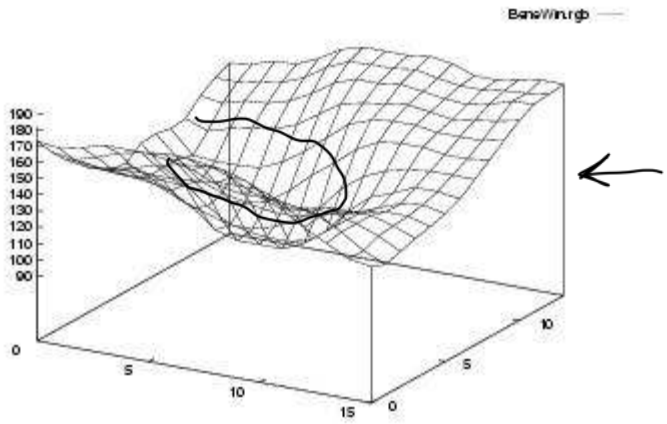
- Edge Normal:
  - Unit vector in the direction of maximum intensity change
- Edge Direction:
  - Unit vector perpendicular to the edge normal
- Edge position or center
  - Image location at which edge is located
- Edge Strength
  - Speed of intensity variation across the edge.

derivative of gray level in edge normal direction

Images as 3-D Surfaces

as a Landscape

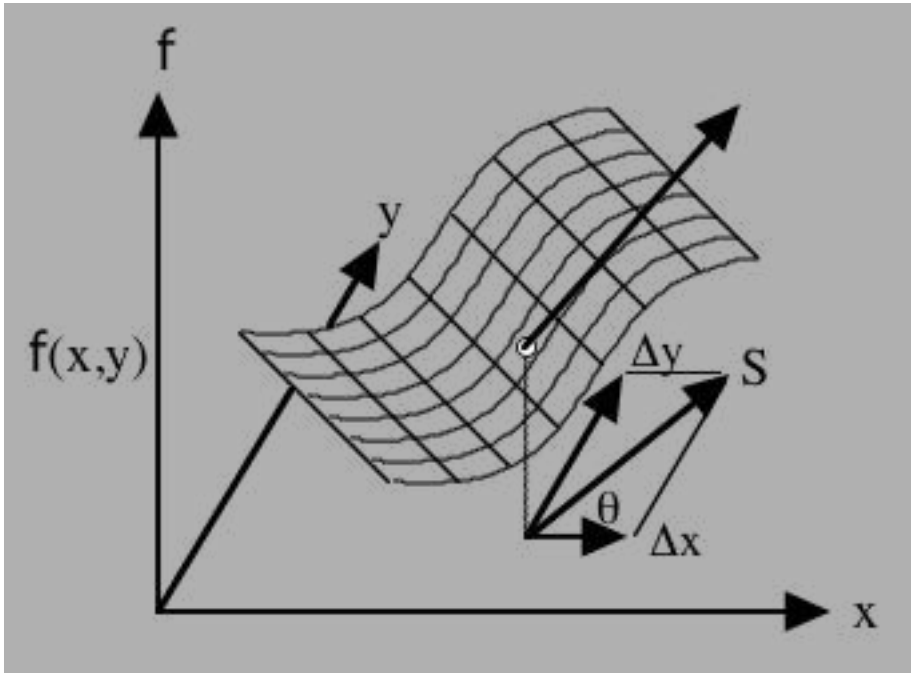




highest slopes in gray-level landscapes

elevation → gray-level for pixel (x,y)

Geometric Interpretation

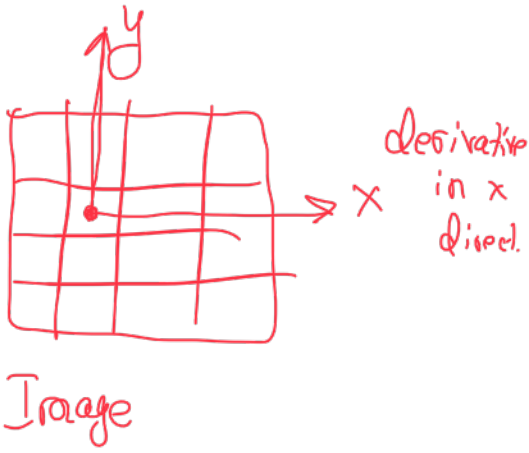


Since I(x,y) is not a continuous function:

1. Locally fit a smooth surface.
2. Compute its derivatives.

and find where derivatives high ↳ dealing with edges

Image Gradient



The gradient of an image

$$\nabla I = \left[\frac{\delta I}{\delta x}, \frac{\delta I}{\delta y}\right]$$

points in the direction of most rapid change in intensity.

$$\nabla I = \left[\frac{\delta I}{\delta x}, 0\right]$$

perpendicular to edge

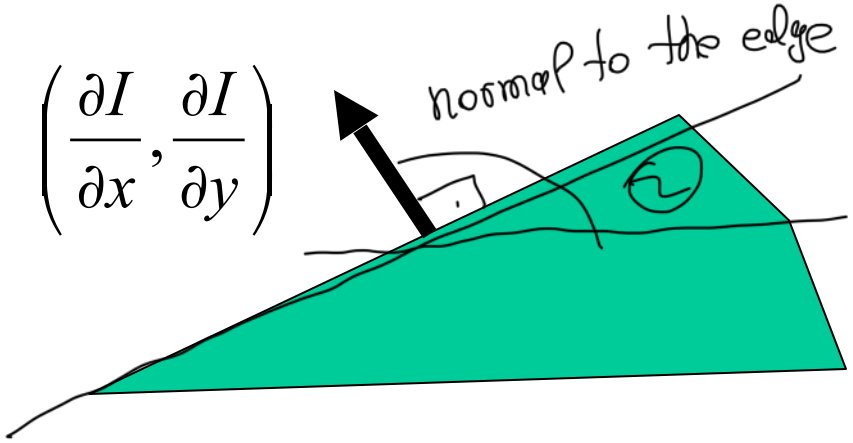
changes rapidly in y direction
constant in x direction

$$\nabla I = \left[0, \frac{\delta I}{\delta y}\right]$$

$$\nabla I = \left[\frac{\delta I}{\delta x}, \frac{\delta I}{\delta y}\right]$$

vector in the direction of slope

Magnitude And Orientation



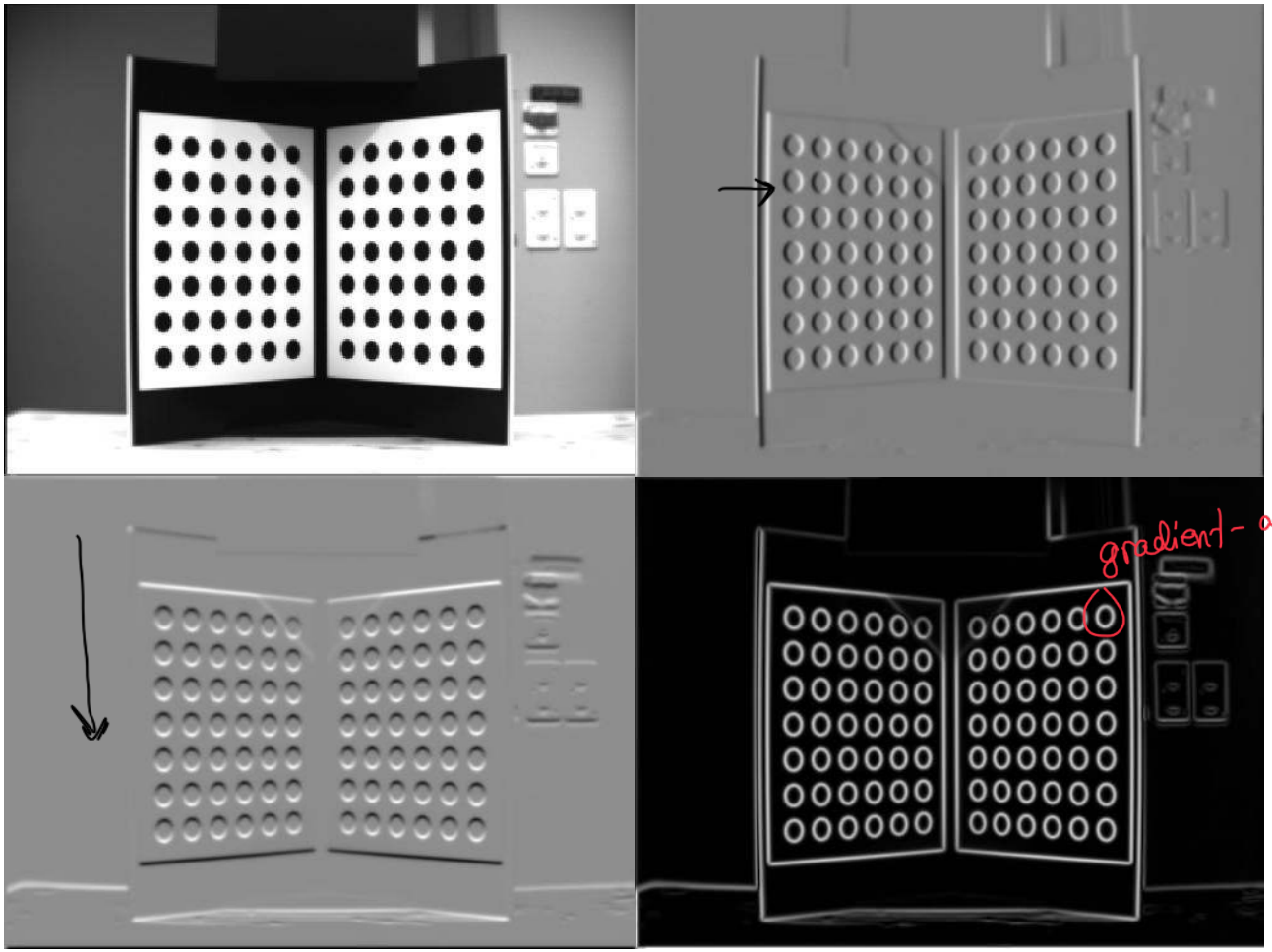
$$\text{Measure of contrast}: \; G = \sqrt{\frac{\partial I}{\partial x}^2 + \frac{\partial I}{\partial y}^2}$$

of gradient vector

$$\text{Edge orientation}: \; \theta = \arctan\left(\frac{\partial I}{\partial y}, \frac{\partial I}{\partial x}\right)$$

Gradient Images

I



$\frac{\partial I}{\partial y}$

black → (−)
gray → 0
white → (+)

$\frac{\partial I}{\partial x}$

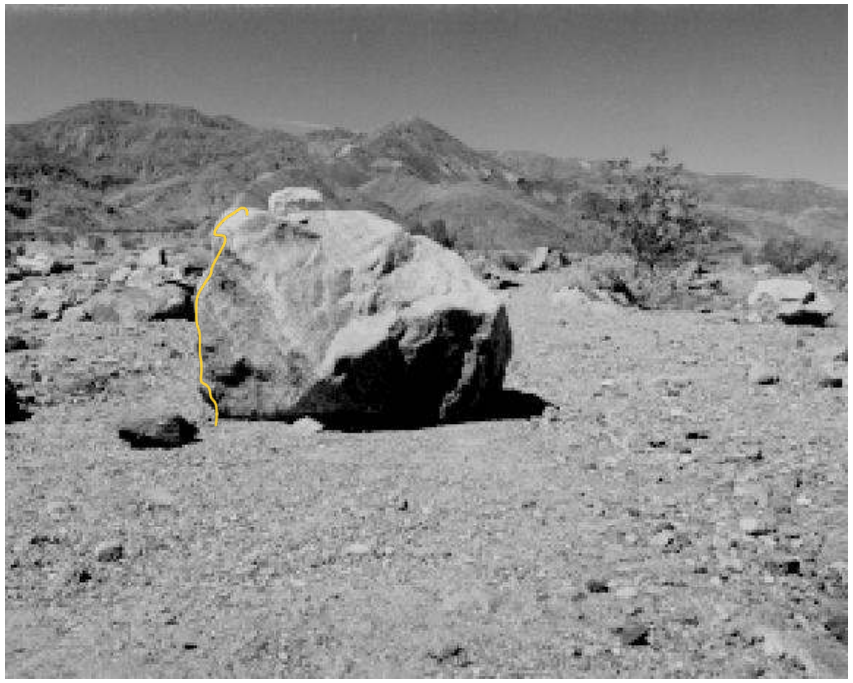
gradient - almost constant

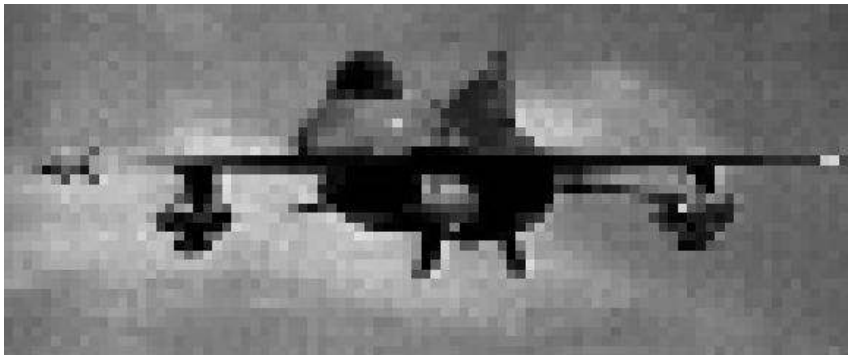
$$\sqrt{\frac{\partial I}{\partial x}^2 + \frac{\partial I}{\partial y}^2}$$

The gradient magnitude is unaffected by orientation ....

invariant

Real Images





… but not directly usable in most real-world images.

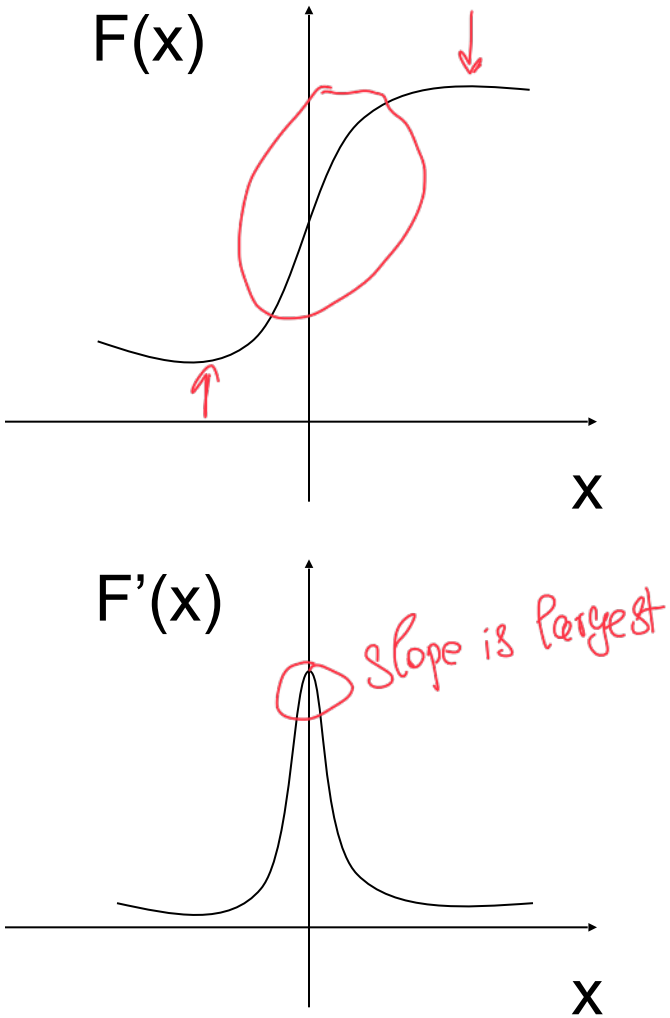
Edge Operators

How do we compute gradients?

- Difference Operators
- Convolution Operators
- Trained Detectors
- Deep Nets

explicit evaluations

Gradient Methods



gray-level goes from low level to high level

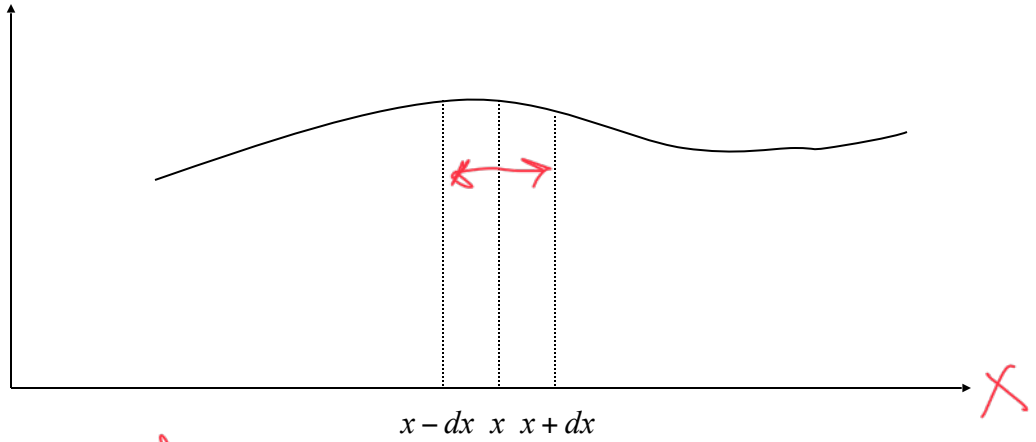
Edge = Sharp variation

Large first derivative

1D Finite Differences

Digital images → 1D

In one dimension:



dx→0

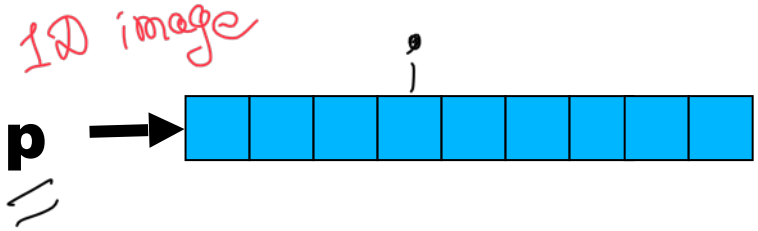
$$\frac{df}{dx} \approx \frac{f(x+dx)-f(x)}{dx} \approx \frac{f(x+dx)-f(x-dx)}{2dx}$$

Approximate

$$\frac{d^2 f}{dx^2} \approx \frac{f(x+dx)-2f(x)+f(x-dx)}{dx^2}$$

Coding 1D Finite Differences

Line stored as an array:

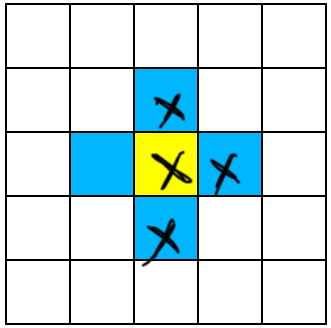


- for i in range(n-1):
  q[i]=(p[i+1]-p[i]) derivative

- for i in range(1,n-1):
  q[i]=(p[i+1]-p[i-1])/2 other version of derivative
  dx=1

- q=(p[2:]-p[:-2])/2 much faster, no loops

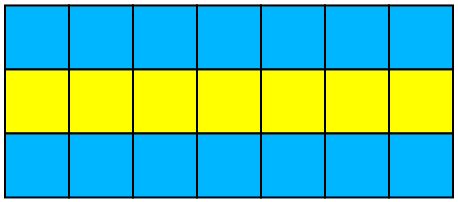
2D Finite Differences



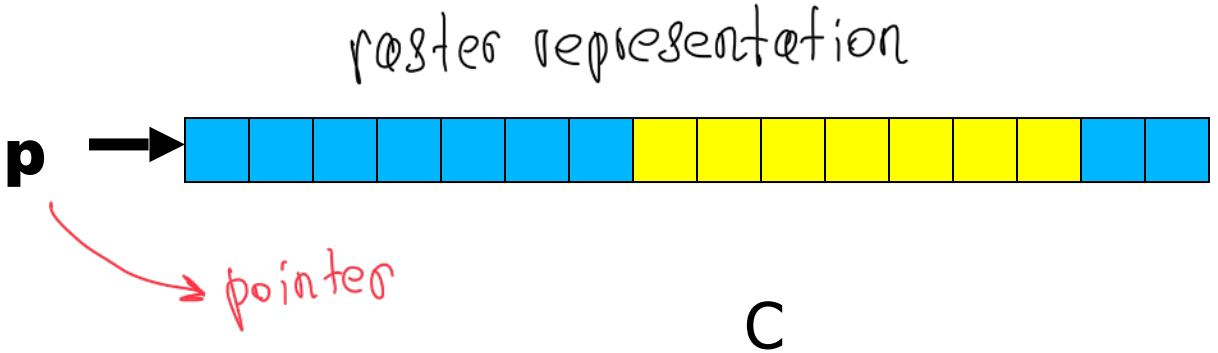
$$\frac{\partial f}{\partial x} \approx \frac{f(x+dx,y)-f(x,y)}{dx} \approx \frac{f(x+dx,y)-f(x-dx,y)}{2dx}$$

$$\frac{\partial f}{\partial y} \approx \frac{f(x,y+dy)-f(x,y)}{dy} \approx \frac{f(x,y+dy)-f(x,y-dy)}{2dy}$$

Coding 2D Finite Differences



Python



C

Image stored as a 2D array:

- dx = p[1:,:]-p[:-1,:]
  dy = p[:,1:]-p[:,:-1]

difference between immediate neighbours

x+dx, x-dx

- dx = (p[2:,:]-p[:-2,:])/2
  dy = (p[:,2:]-p[:,:-2])/2

Image stored in raster format:

```
{
  int i;
  for(i=0;i<xdim;i++){
    dx[i] = p[i+1]     -p[i];
    dy[i] = p[i+xdim]-p[i];
  }
}
```

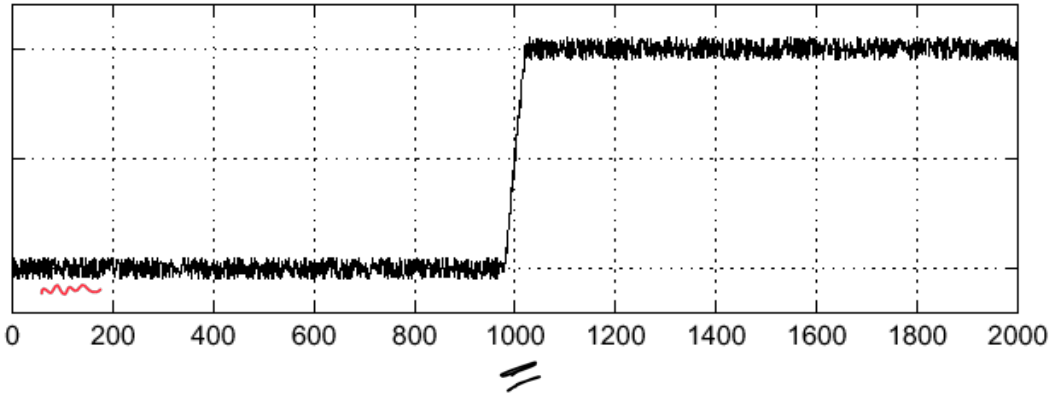
pixel immediately below i

- Only 1D array accesses
- No multiplications

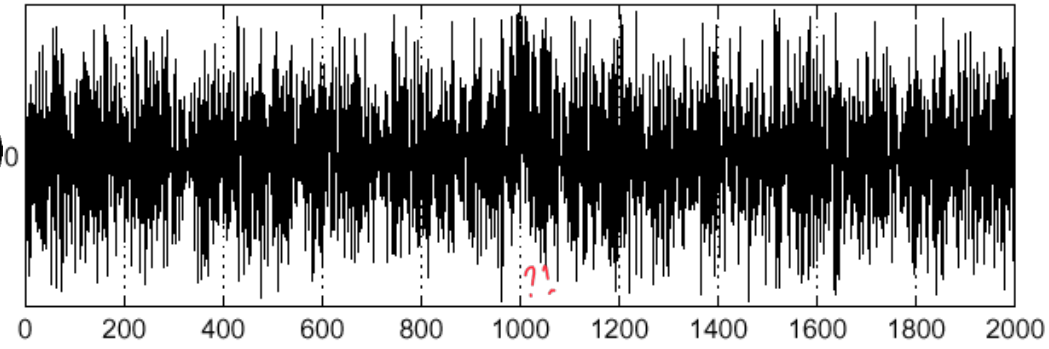
—> Can be exploited to increase speed.

Noise in 1D

Consider a single row or column of the image:



$\frac{d}{dx} f(x)$



noisy derivative

noise magnified

Fourier Interpretation

$f \longrightarrow F$: fourier transform

Function	Fourier Transform
$\frac{df}{dx}(x)$	$uF(u)$
$\frac{\delta f}{\delta x}(x,y)$	$uF(u,v)$
$\frac{\delta f}{\delta y}(x,y)$	$vF(u,v)$

↑u ⇒ ↑freq

frequency domain

noise = bunch of freq's

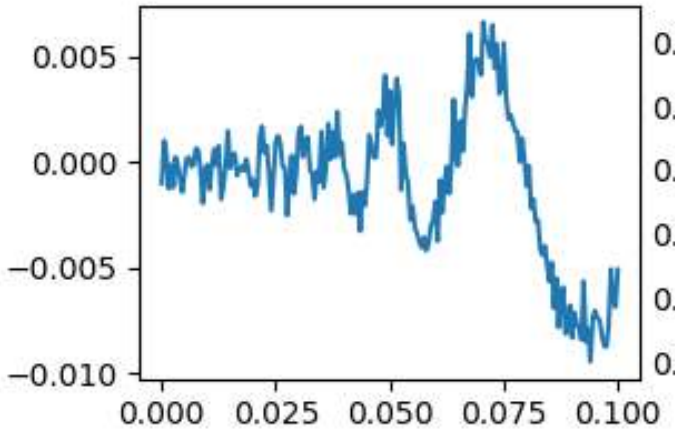
→ Differentiating emphasizes high frequencies and therefore noise!

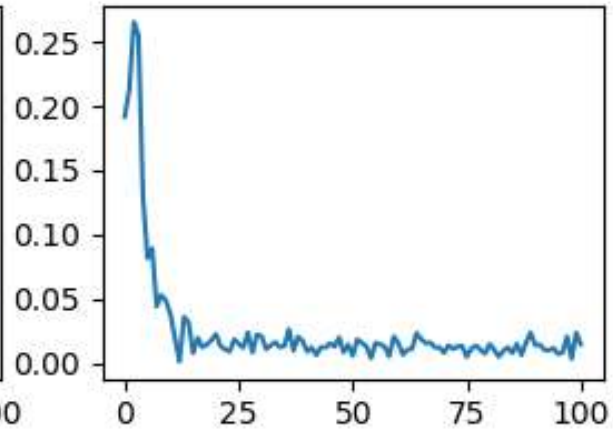
large u, v ⇒ ↑ frequency

$f(x) = x^2 \sin(1/x)$

→ continuous
→ not diff'ble at zero

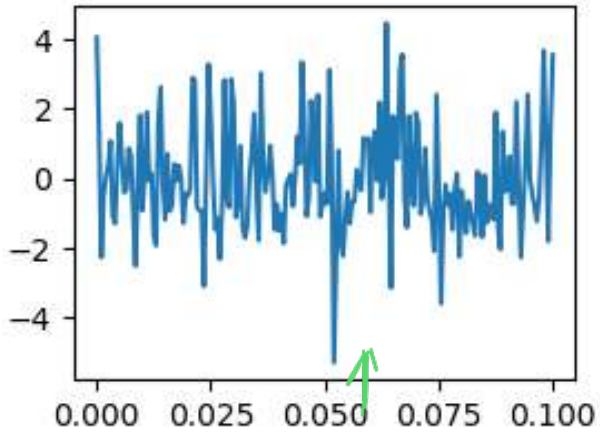
f

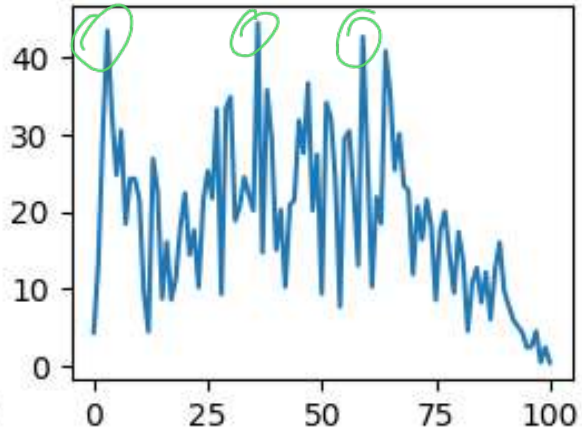




F

$\frac{df}{dx}$





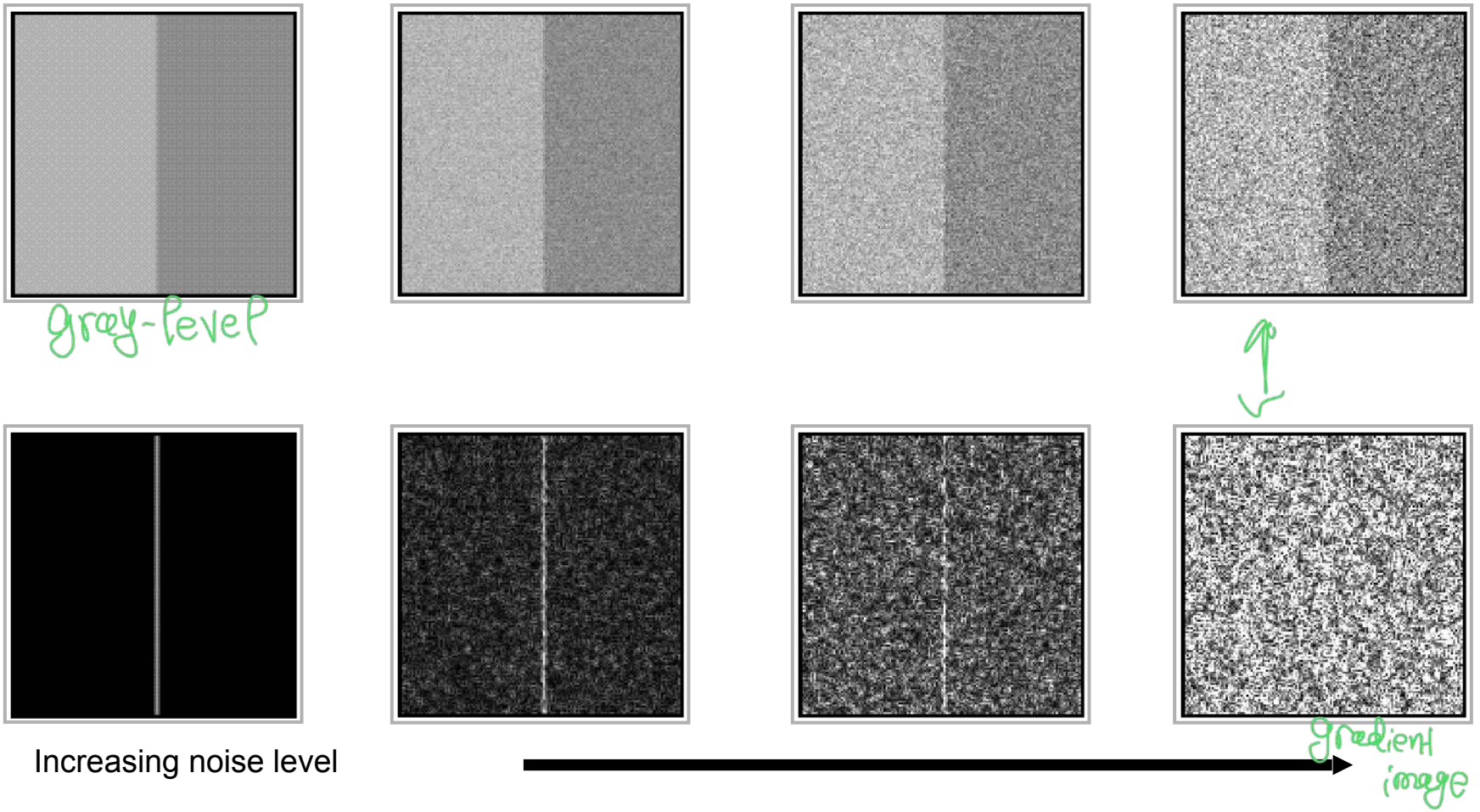
adding noise to function uF

Original function
+
Noise

Fourier transform

adding noise makes the derivative unrecognizable

Noise in 2D



As the amount of noise increases, the derivatives stop being meaningful.

Removing Noise

Problem:

magnifies high freq's

- High frequencies and differentiation do not mix well.

Solution:

- Suppress high frequencies by
  - using the Discrete Fourier Transform.

Discrete Fourier Transform

$$F(\mu,\nu) = \frac{1}{\sqrt{M*N}} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) e^{-2i\pi(\mu x/M + \nu y/N)}$$

Sum over all pixels in 2D image

$$f(x,y) = \frac{1}{\sqrt{M*N}} \sum_{\mu=0}^{M-1} \sum_{\nu=0}^{N-1} F(\mu,\nu) e^{+2i\pi(\mu x/M + \nu y/N)}$$

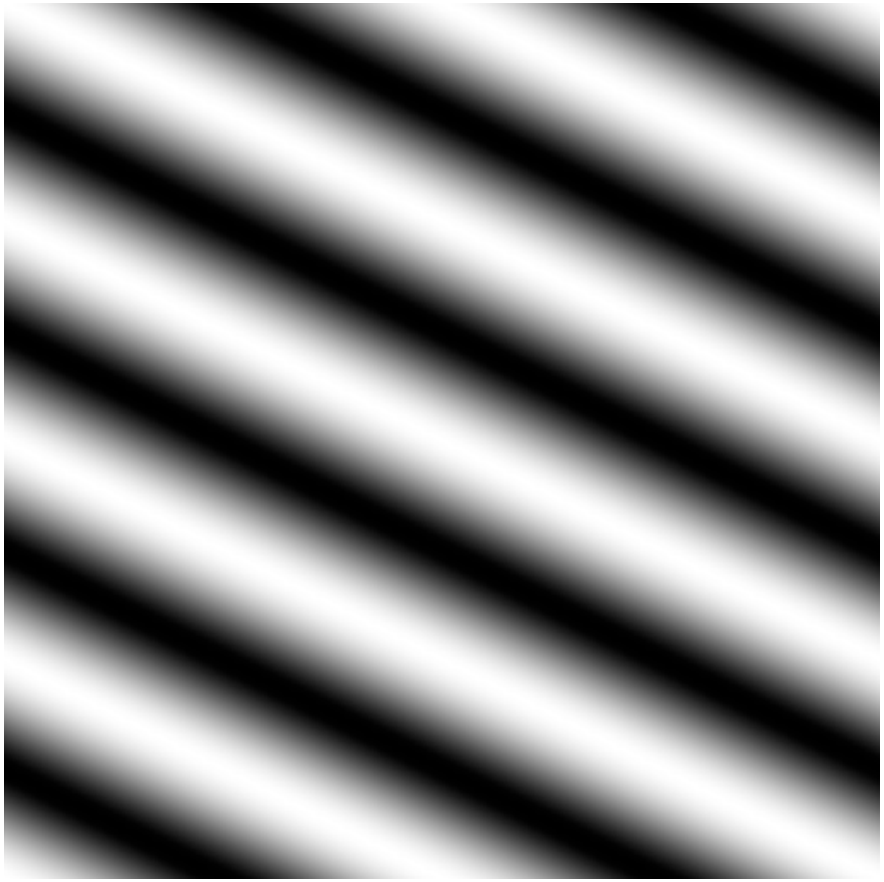
Inverse Fourier Transform

The DFT is the discrete equivalent of the 2D Fourier transform:

- The 2D function f is written as a sum of sinusoids.
- The DFT of f convolved with g is the product of their DFTs.

$$DFT\{f * g\} = DFT(f) \cdot DFT(g)$$

Fourier Basis Element



Sinusoidal waves

$e^{ix} = \cos(x) + i\sin(x)$

Real part of

$$e^{+2i\pi(ux+vy)}$$

where

- $\sqrt{u^2 + v^2}$ represents the frequency,
- $\text{atan}(v, u)$ represents the orientation.

of sinusoidal waves

Fourier Basis Element

Real part of

$$e^{+2i\pi(ux+vy)}$$

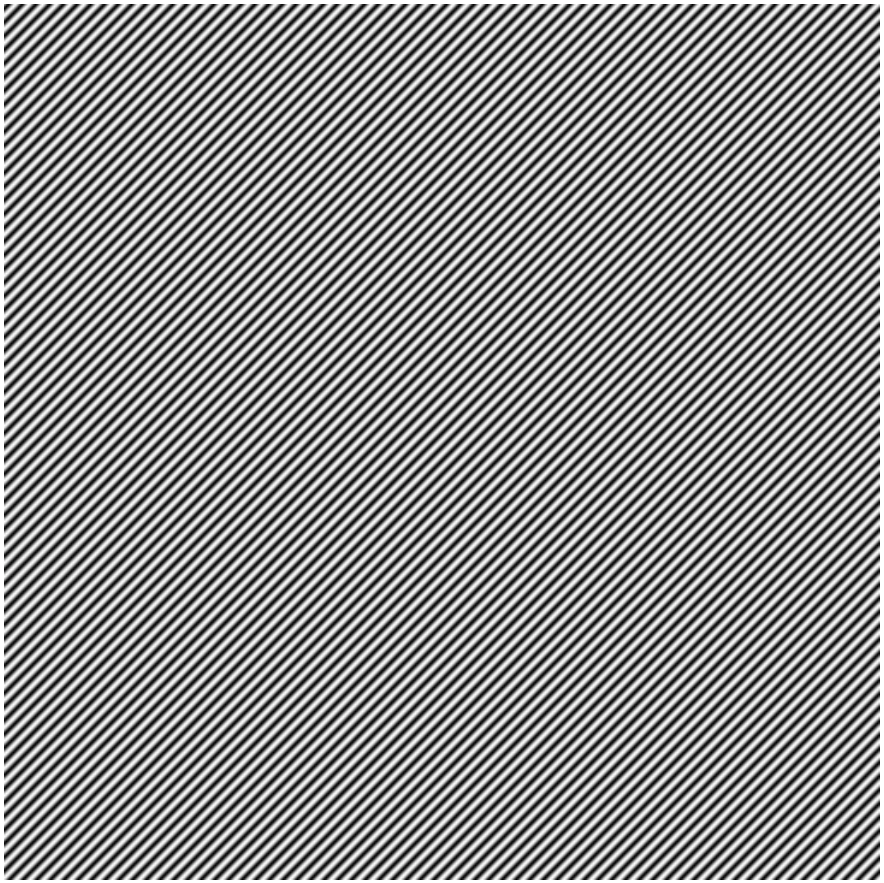
where

- $\sqrt{u^2+v^2}$ is larger than before.

frequency ↑

orientation changed

Fourier Basis Element



Real part of

$$e^{+2i\pi(ux+vy)}$$

where

- $\sqrt{u^2+v^2}$ is larger still.

Truncated Inverse DFT

$$F(\mu,\nu) = \frac{1}{\sqrt{M*N}} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) e^{-2i\pi(\mu x/M + \nu y/N)}$$

$$f(x,y) = \frac{1}{\sqrt{M*N}} \sum_{\mu=0}^{M-1} \sum_{\nu=0}^{N-1} F(\mu,\nu) e^{+2i\pi(\mu x/M + \nu y/N)}$$

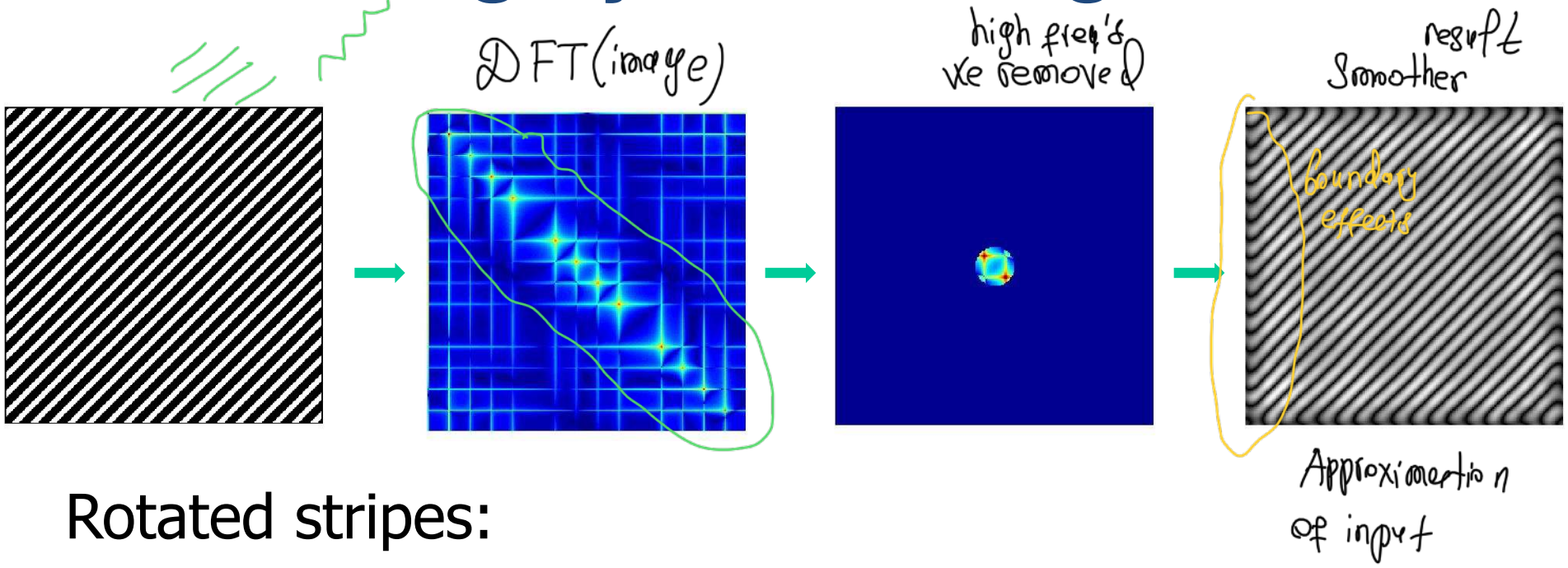
getting rid of high freq's

$$f(x,y) = \frac{1}{\sqrt{M*N}} \sum_{\mu^2+\nu^2<T} F(\mu,\nu) e^{+2i\pi(\mu x/M + \nu y/N)}$$

T is a hand-specified threshold.

- The sinusoids corresponding to $\mu^2 + \nu^2 \geq T$ depict high frequencies.
- Removing them amounts to removing high-frequencies.

Smoothing by Truncating the IDFT



Rotated stripes:

- Dominant diagonal structures
- Discretization produces additional harmonics

—> Removing higher frequencies and reconstructing yields a smoothed image.

Removing Noise

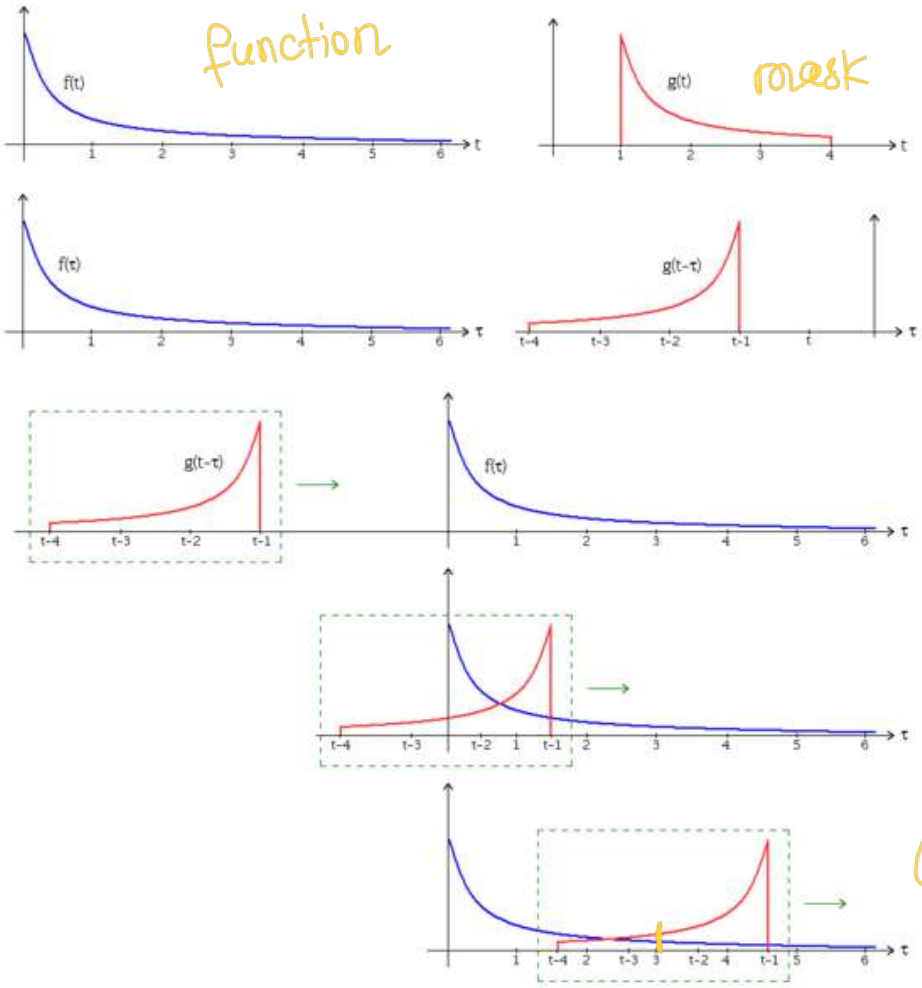
Problem:

- High frequencies and differentiation do not mix well.

Solution:

- Suppress high frequencies by
  - using the Discrete Fourier Transform,
  - convolving with a low-pass filter.

1D Convolution



$$g * f(t) = \int_{\tau} g(t-\tau) f(\tau) d\tau$$

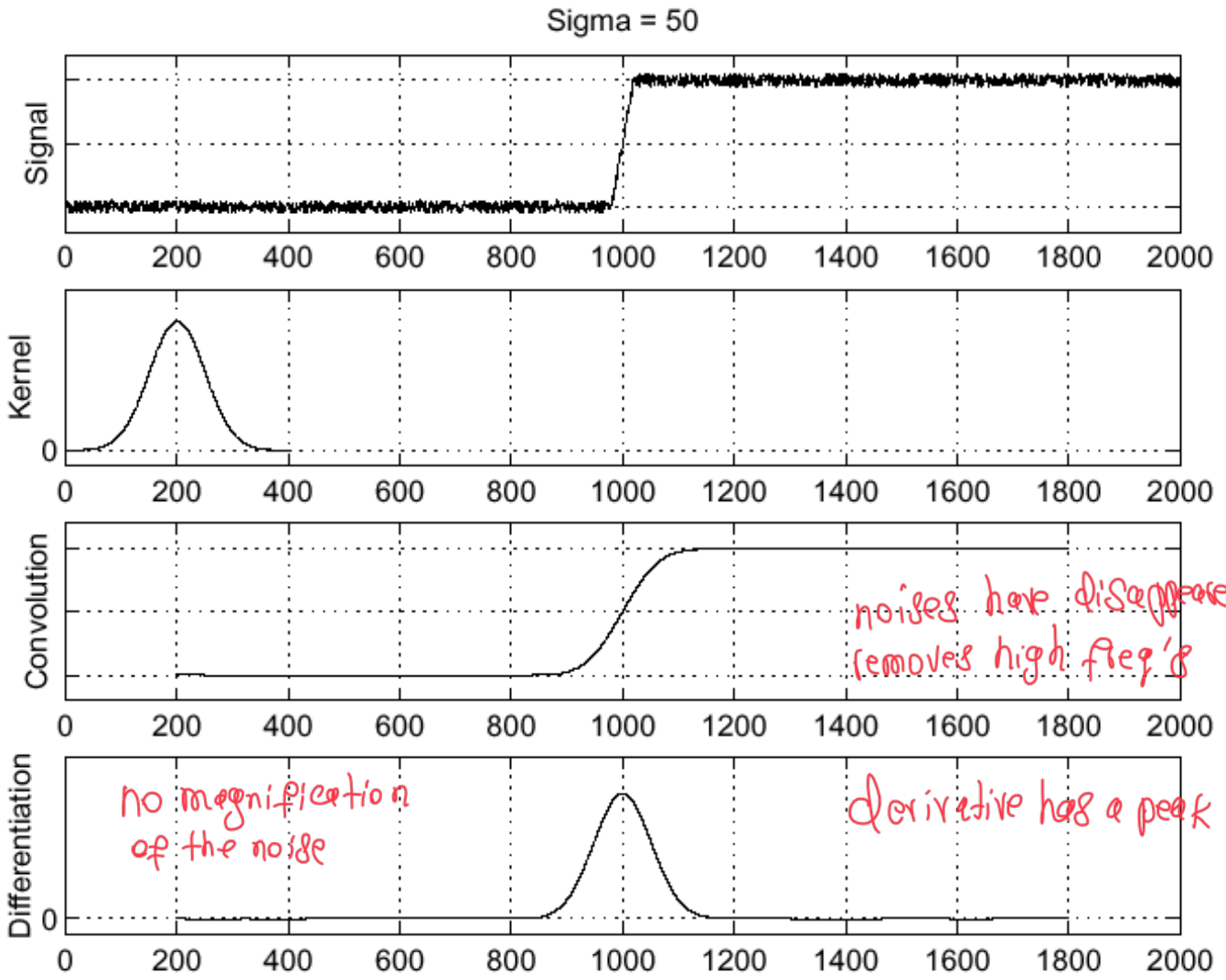
Smooth Before Differentiating

f

g

$g * f$

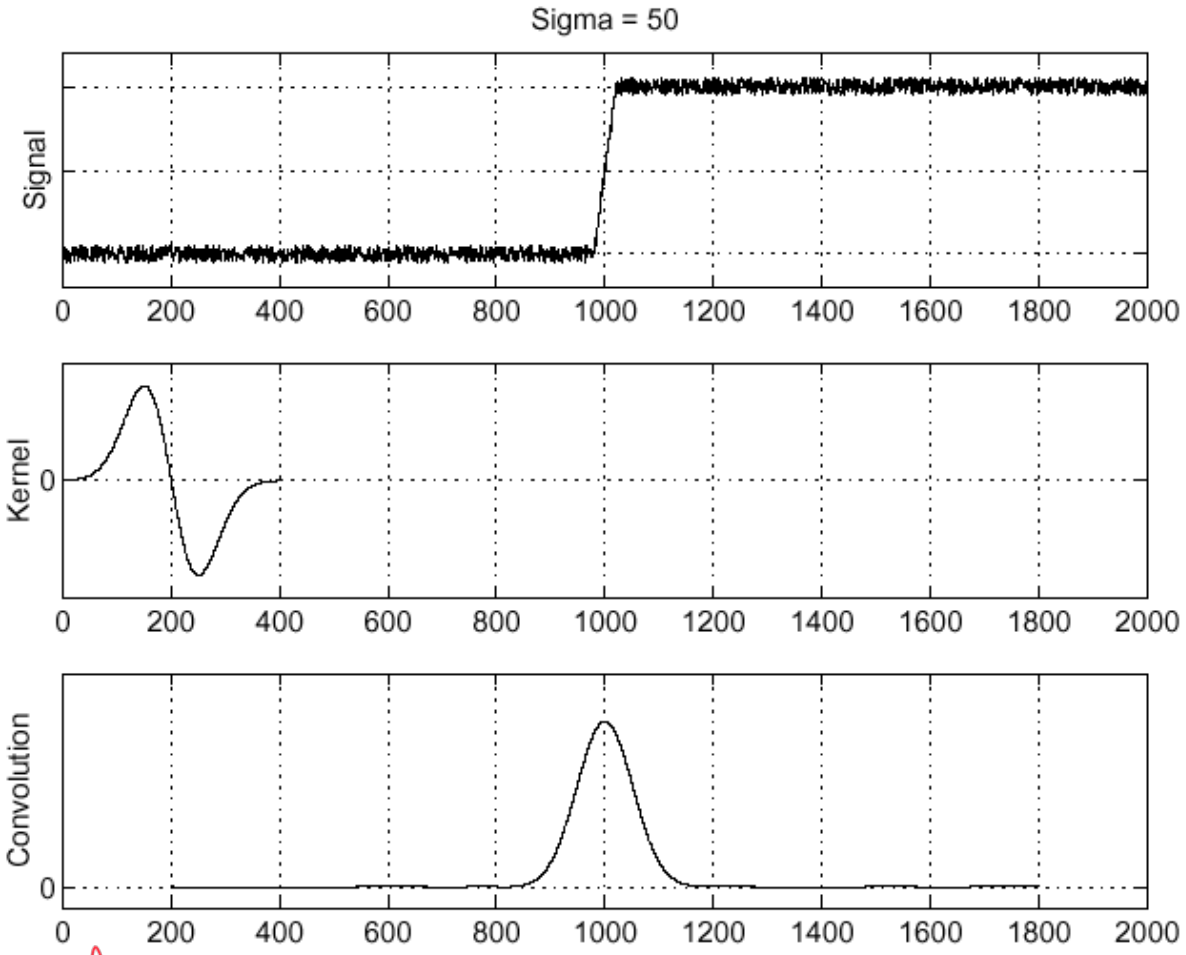
$$\frac{\partial}{\partial x}(g * f)$$



Simultaneously Smooth and Differentiate

$$\frac{\partial}{\partial x}(g * f) = \frac{\partial g}{\partial x} * f$$

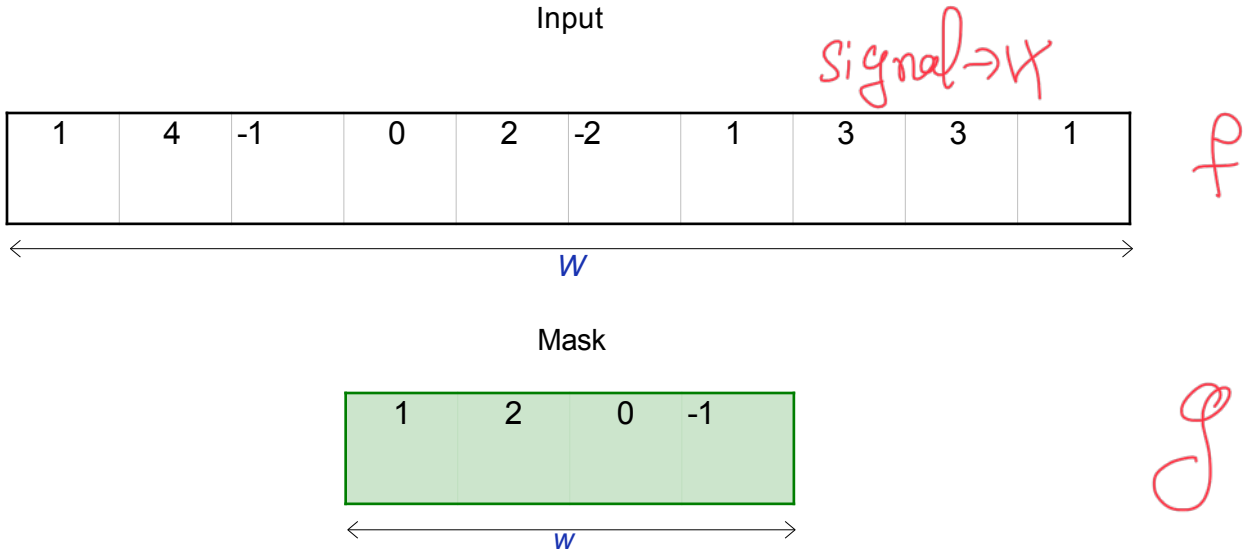
$$\frac{d}{dx}(f * g) = \frac{df}{dx} * g$$



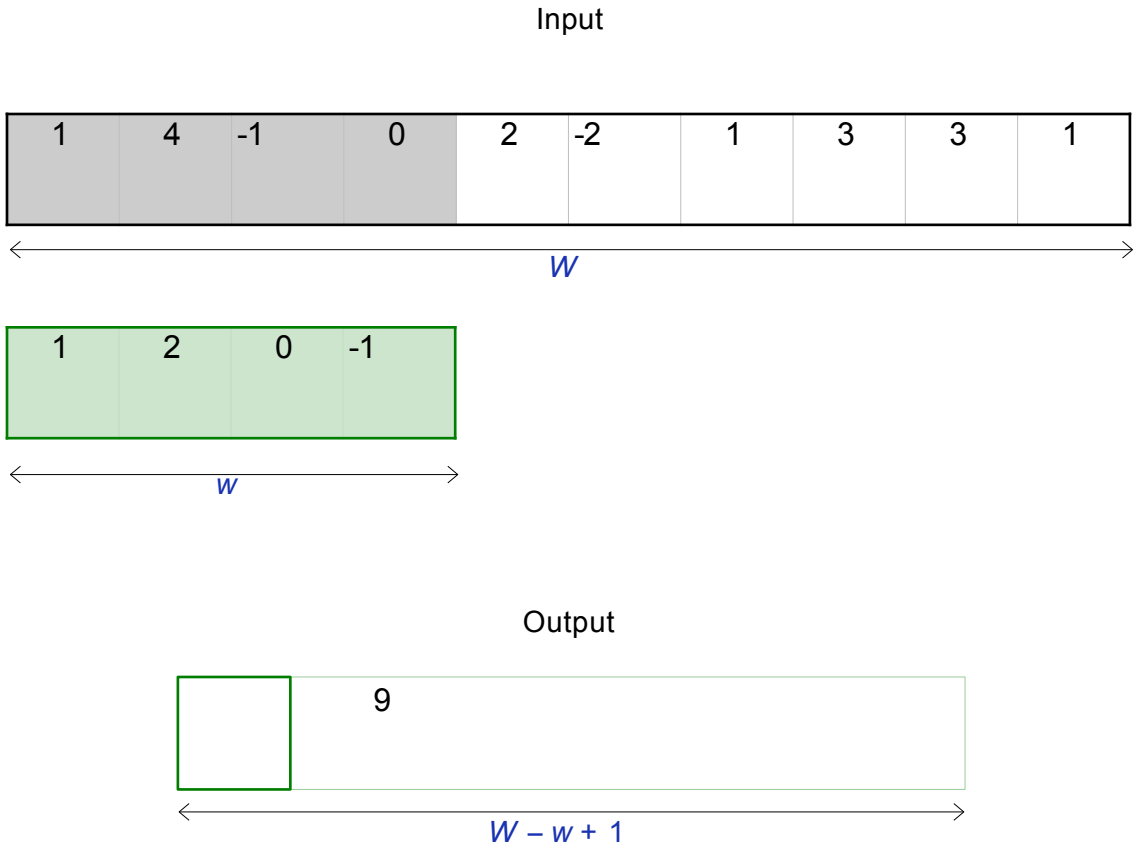
differentiation and smoothing at the same time

--> Faster because dg/dx can be precomputed.

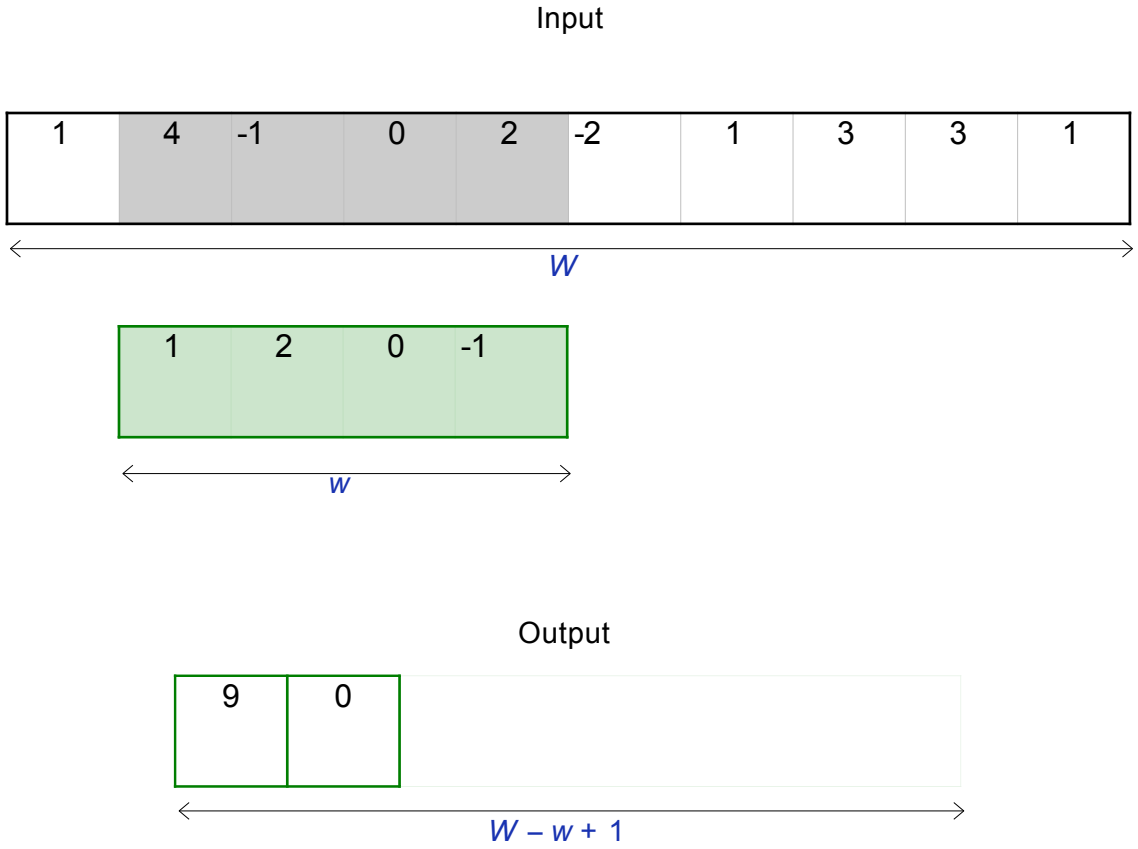
Discrete 1D Convolution



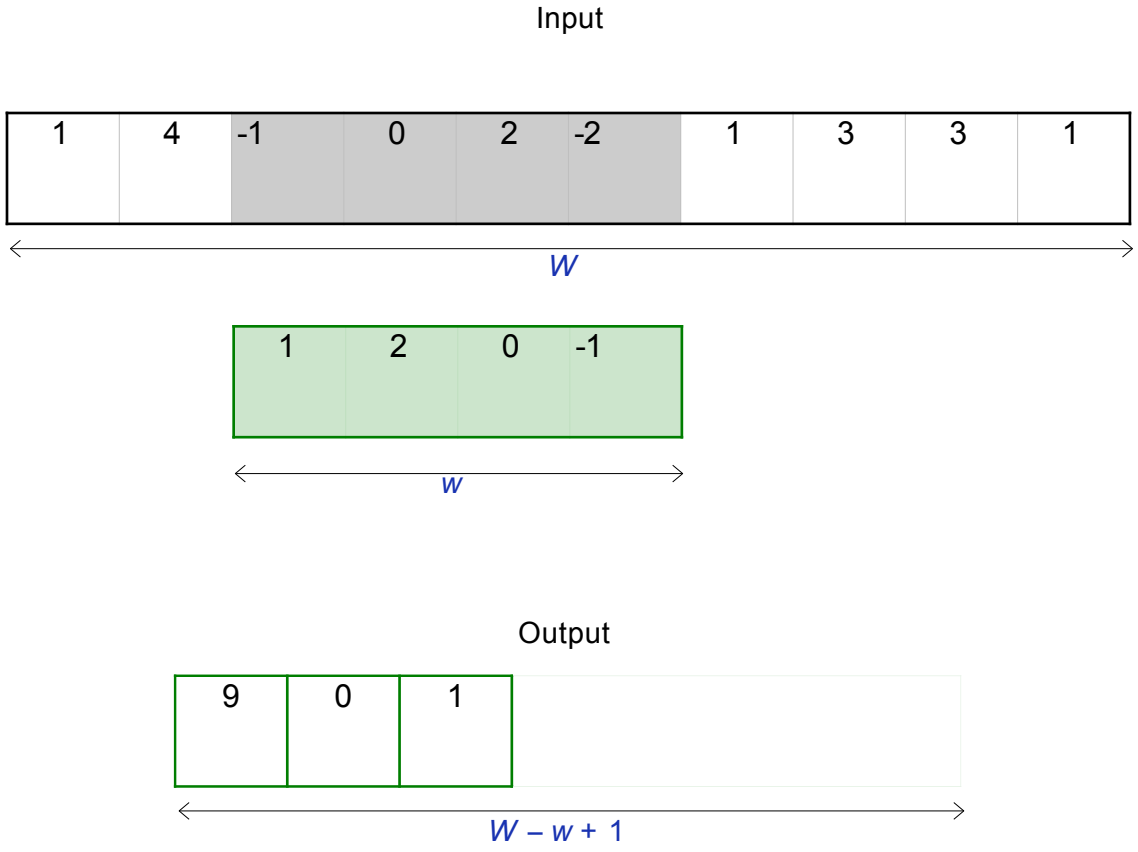
Discrete 1D Convolution



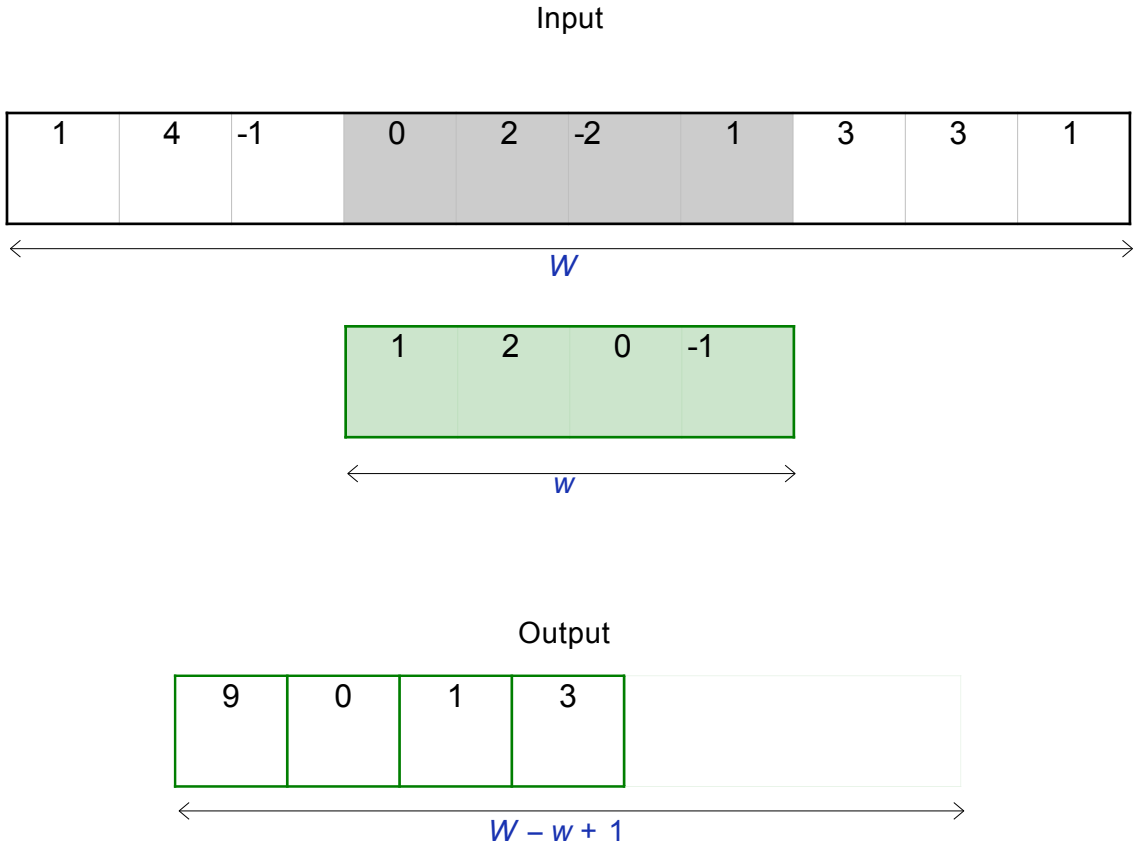
Discrete 1D Convolution



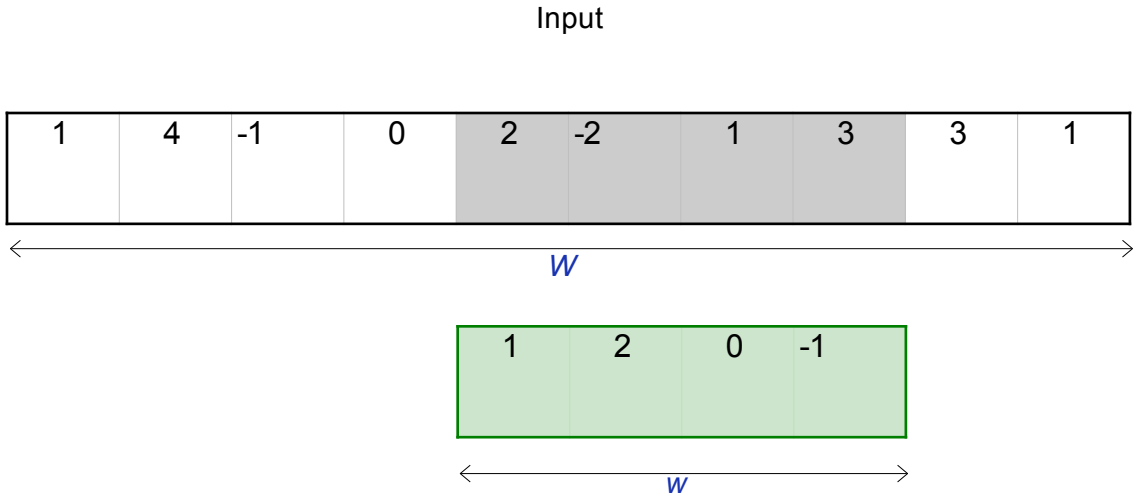
Discrete 1D Convolution



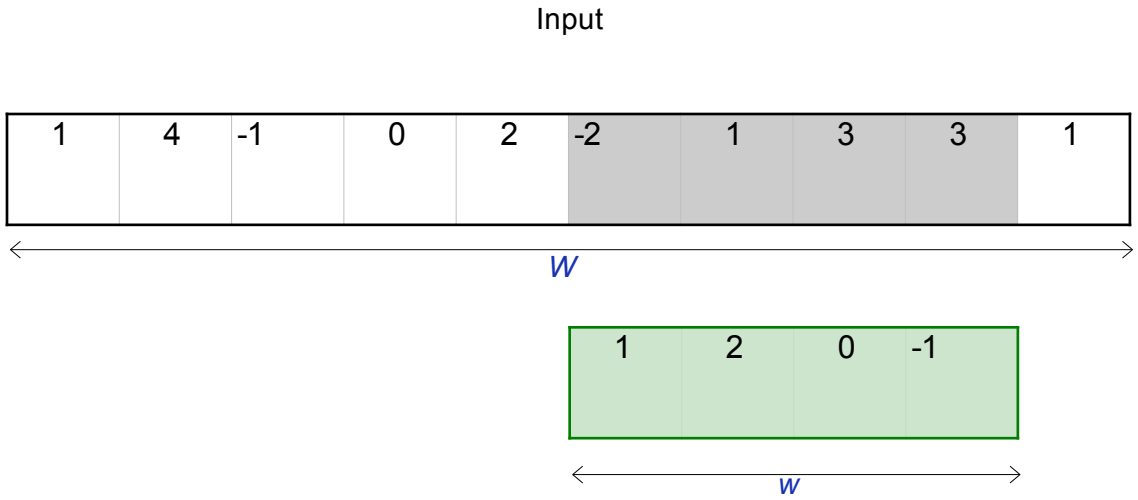
Discrete 1D Convolution



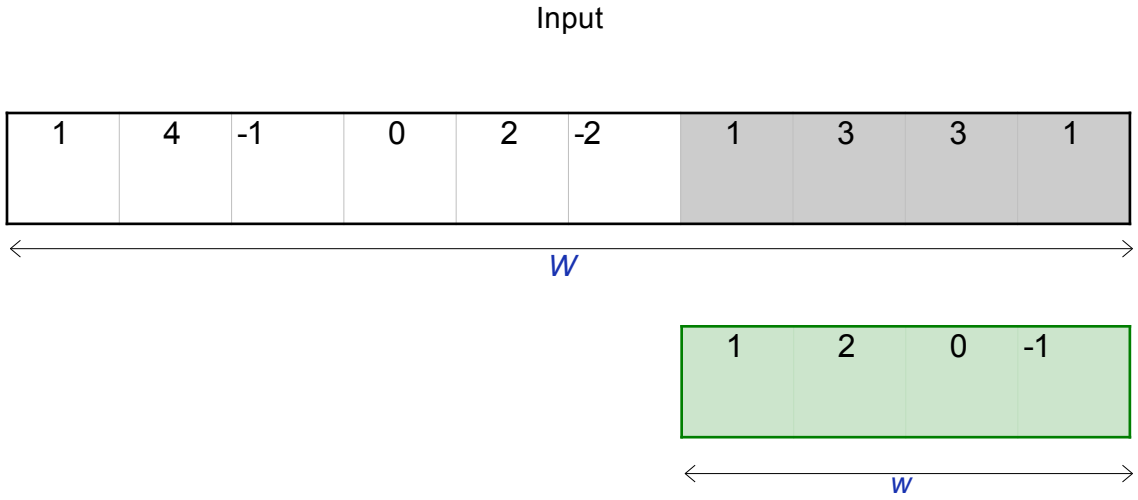
Discrete 1D Convolution



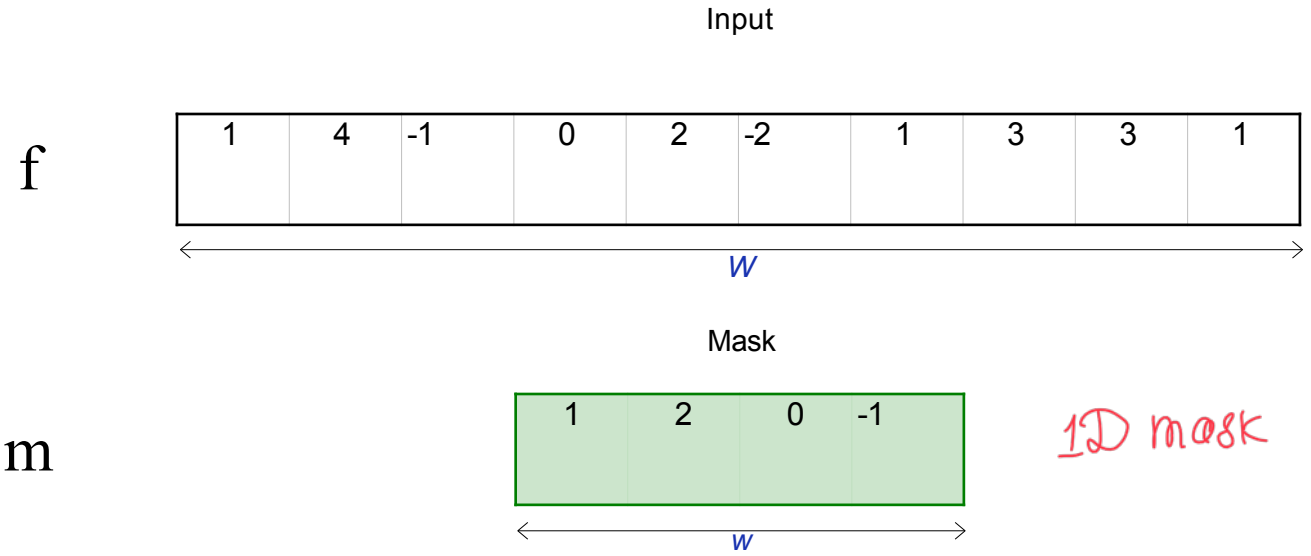
Discrete 1D Convolution



Discrete 1D Convolution



Discrete 1D Convolution



1D signal

1D mask

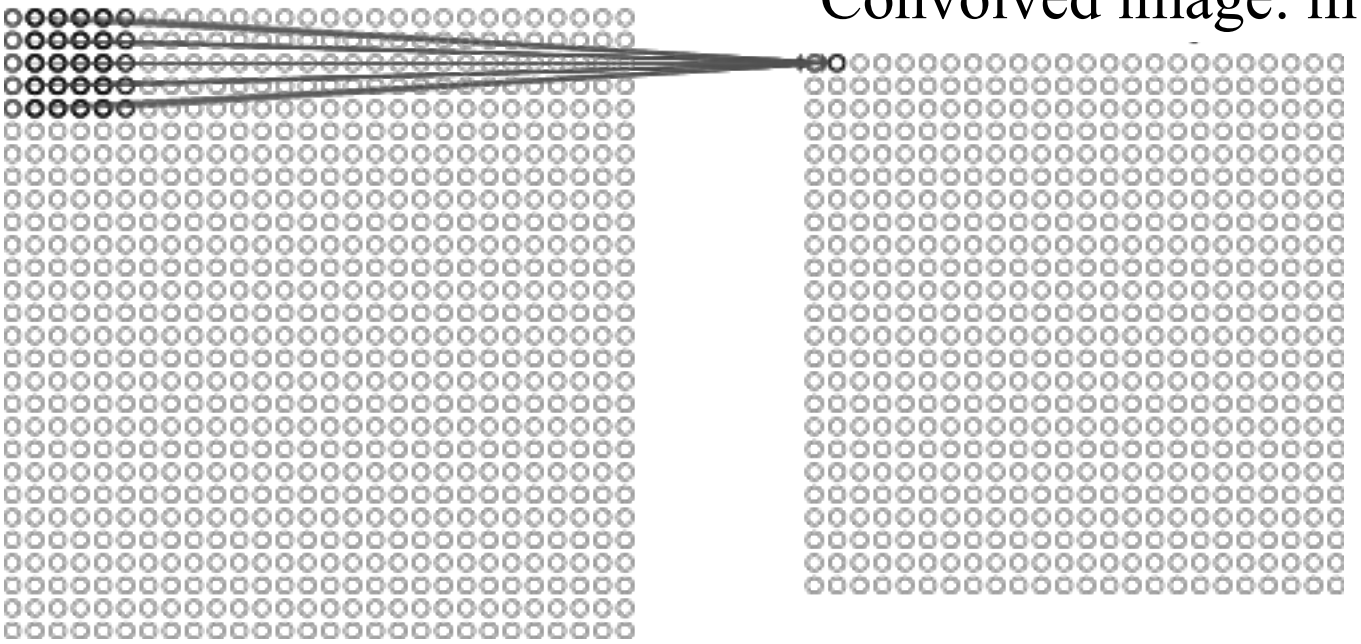
$$m * f(x) = \sum_{i=0}^{w} m(i) f(x - i)$$

instead of integral, we've sum

Discrete 2D Convolution

Input image: f

Convolved image: m**f



Convolution mask m, also known as a *kernel*.

$$\begin{bmatrix} m_{11} & \dots & m_{1w} \\ \dots & \dots & \dots \\ m_{w1} & \dots & m_{ww} \end{bmatrix}$$

W x W size

$$m**f(x,y) = \sum_{i=0}^{w}\sum_{j=0}^{w} m(i,j)f(x-i,y-j)$$

Replace (x,y) pixel → sum of weighted values of mask and f

Differentiation As Convolution

Convolving with 1D mask

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

$$[-0.5,0,0.5] \rightarrow \frac{\partial f}{\partial x} \approx \frac{f(x+dx,y)-f(x-dx,y)}{2dx}$$

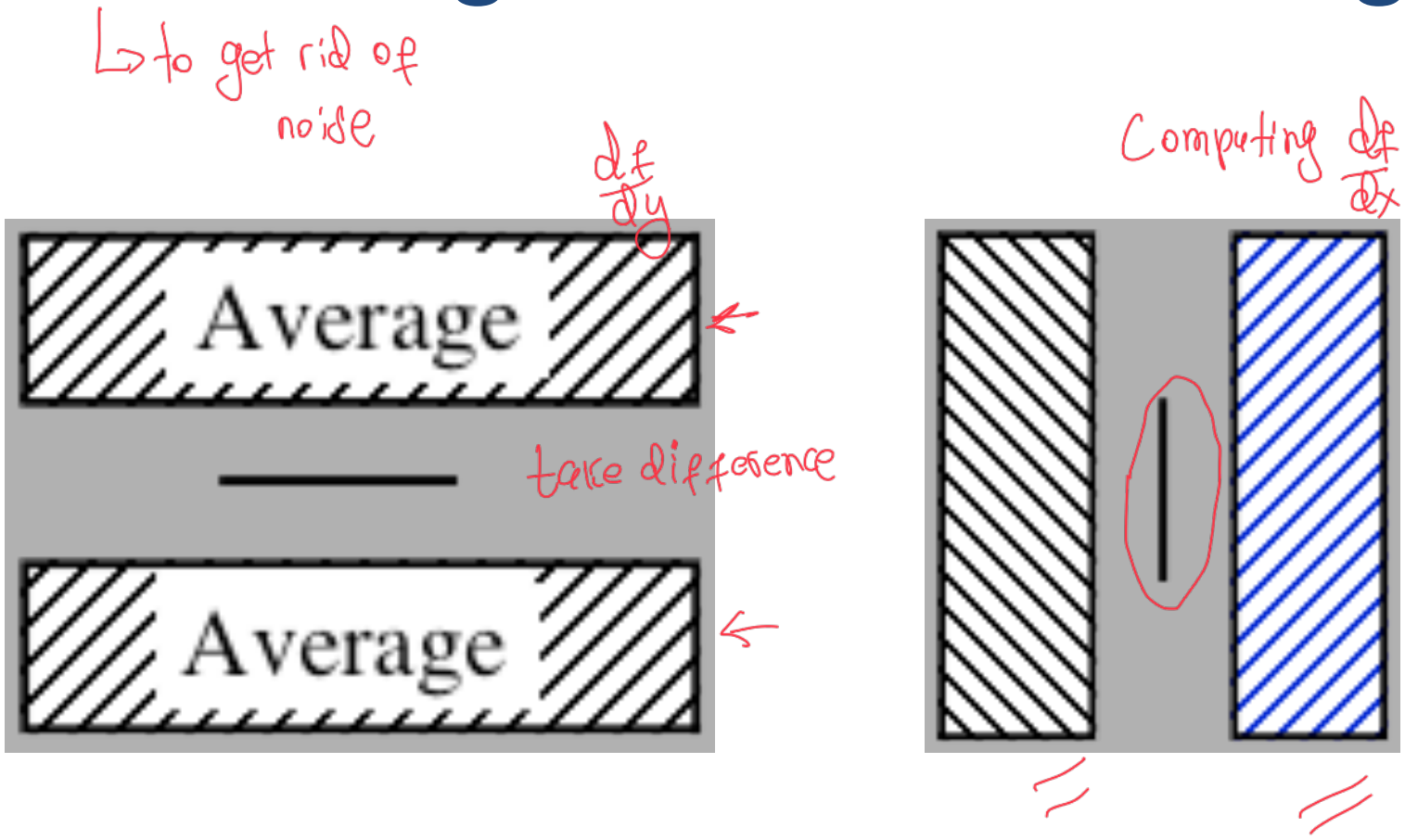
$$\begin{bmatrix}-1\\1\end{bmatrix} \rightarrow \frac{\partial f}{\partial y} \approx \frac{f(x,y+dy)-f(x,y)}{dy}$$

$$\begin{bmatrix}-0.5\\0\\0.5\end{bmatrix} \rightarrow \frac{\partial f}{\partial y} \approx \frac{f(x,y+dy)-f(x,y-dy)}{2dy}$$

to kill higher freq's

→ Use wider masks to add some smoothing

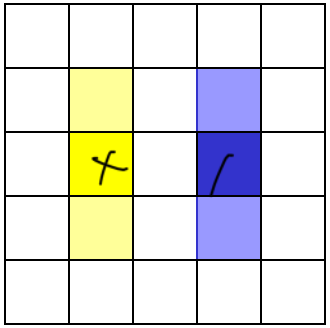
Smoothing and Differentiating



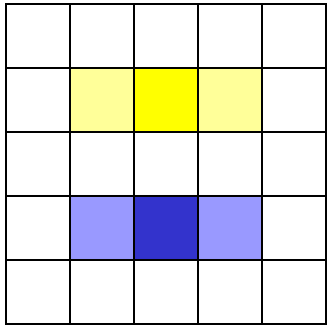
Compute the difference of averages on either side of the central pixel.

3X3 Masks

x derivative



y derivative



$$\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \text{ and } \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

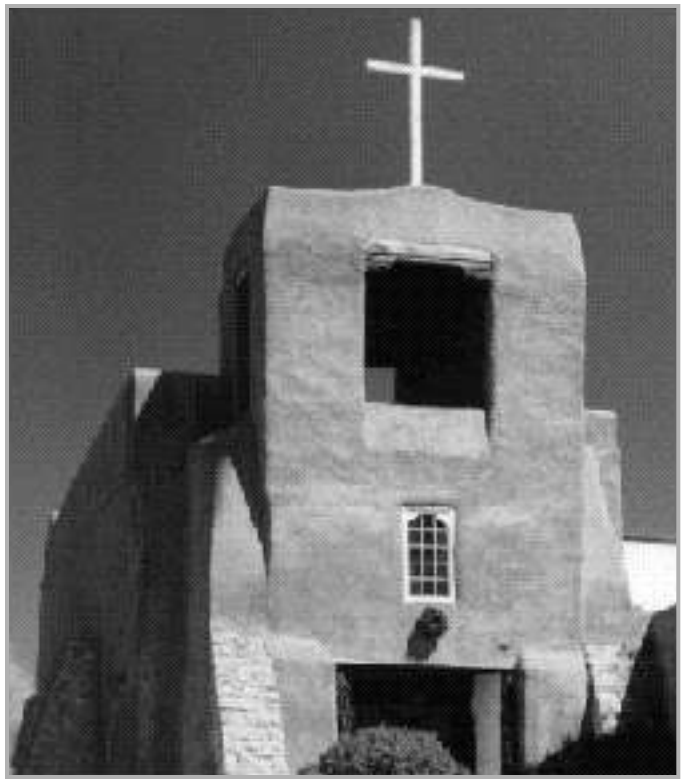
Prewitt operator

more weight

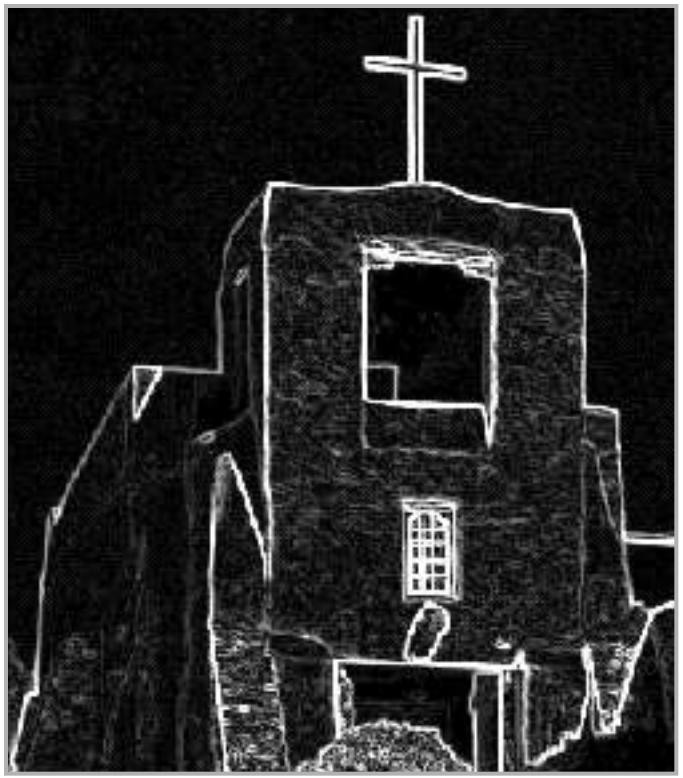
$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \text{ and } \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

Sobel operator

Prewitt Example



Santa Fe Mission



Gradient Image

Sobel Example

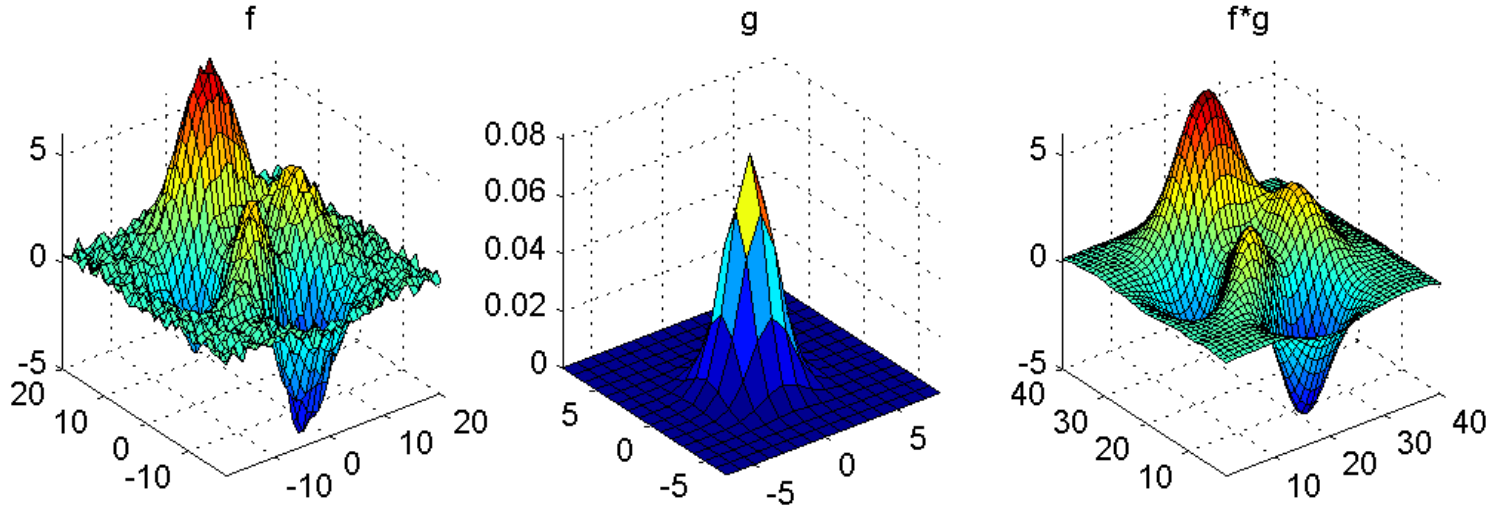
think as a surface

Compute gradients

not so interesting

edges in trees: gray-levels constant, → we get them

Gaussian Smoothing



- More principled way to eliminate high frequency noise.
- Is fast because the kernel is
  - small,
  - separable.

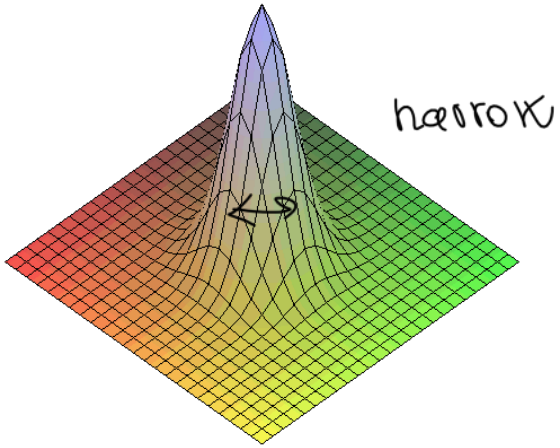
More justifable

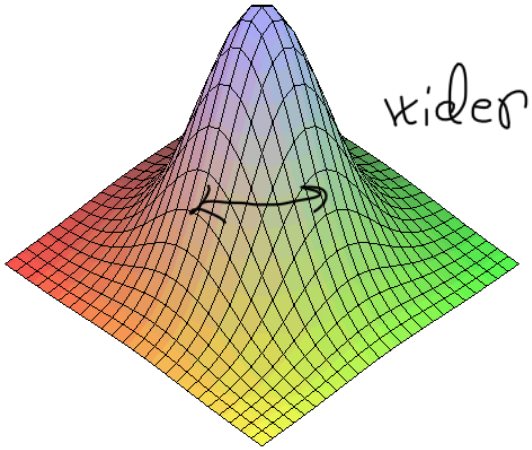
"smoother" mask

← fast computation

DFT → IDFT but

Gaussian Functions





$\sigma = 1$ $\sigma = 2$

↳ variance (width)

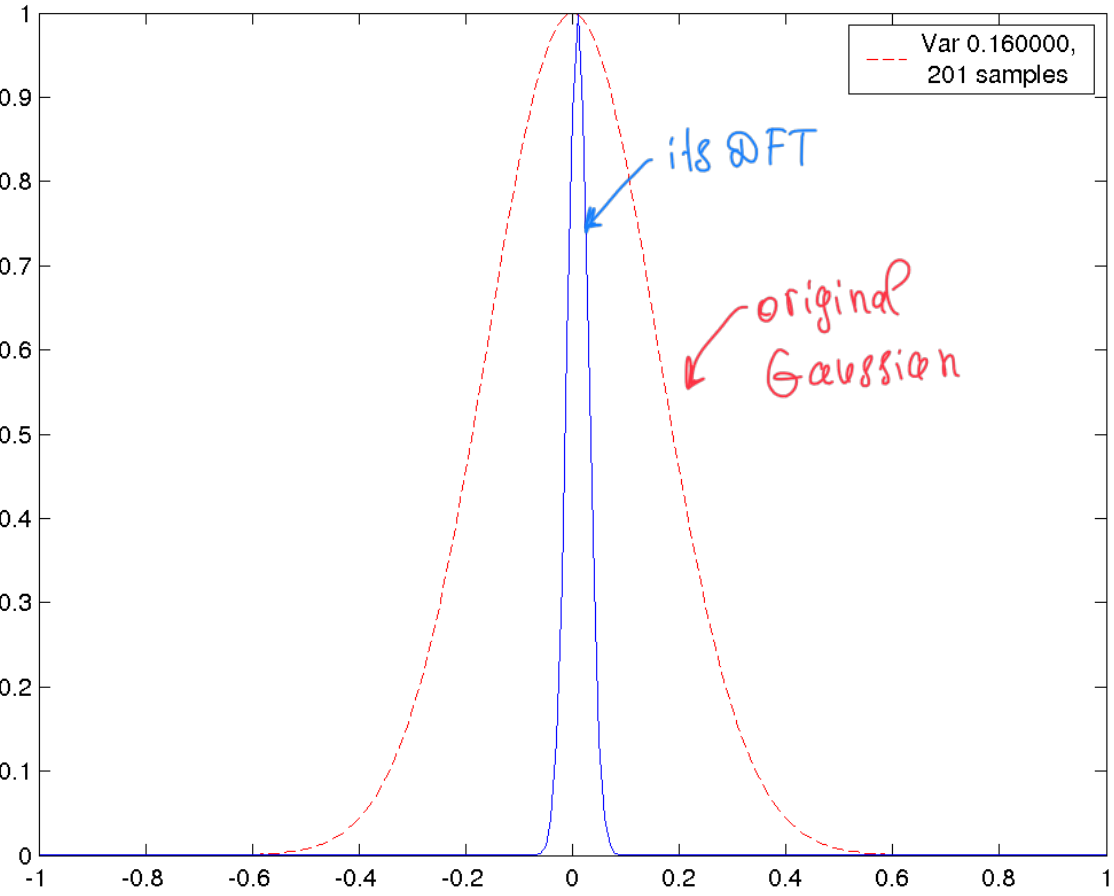
$$g_2(x, y) = \frac{1}{2\pi\sigma^2} \exp(-(x^2 + y^2)/2\sigma^2)$$

$x^2+y^2 > \sigma^2 \Rightarrow \exp(\dots)$ becomes super small

we ignore it

- The integral is always 1.0
- The larger σ, the broader the Gaussian is.
- As σ approaches 0, the Gaussian approximates a Dirac function.

DFT of a Gaussian



- The DFT of a Gaussian is a Gaussian.
- It has finite support.
- Its width is inversely proportional to that of the original Gaussian.

(OG) wider width → (DFT) narrower width

Gaussians as Low-Pass Filters

Support of a function → set of pts where func is not zero, or closure of that set

- The Fourier transform of a convolution is the product of their Fourier transforms: $\mathcal{F}(g * f) = \mathcal{F}(g)\mathcal{F}(f)$.
- If g is a Gaussian, so is $\mathcal{F}(g)$.
- Furthermore if g is broad, the support of $\mathcal{F}(g)$ is small.
- So is the support of $\mathcal{F}(g * f)$.
- There are no more high-frequencies in $g * f$.

and gaussian → not so

g-broad (small) ⇒ $\mathcal{F}(g*f)$ - small (broad)

much efficient

—> Convolving with a Gaussian suppresses the high frequencies.

You can reconstruct g*f without high freq's

Gaussian Smoothed Images

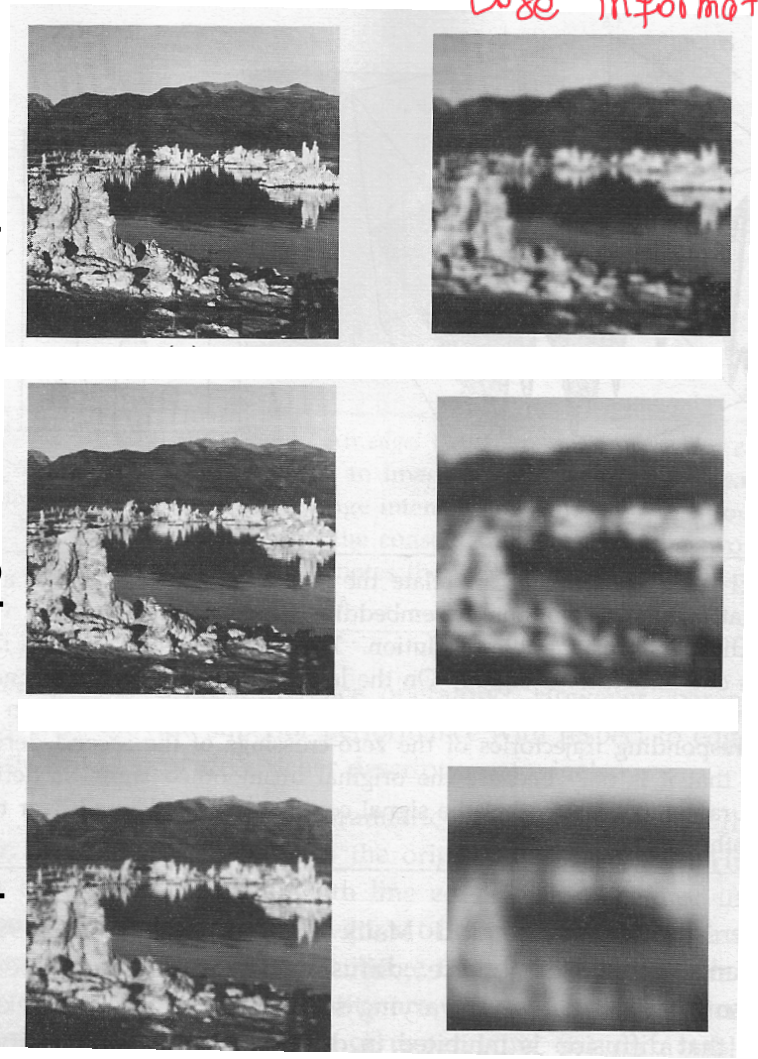
Convolving with gaussian g

(high freq's)
Lose information

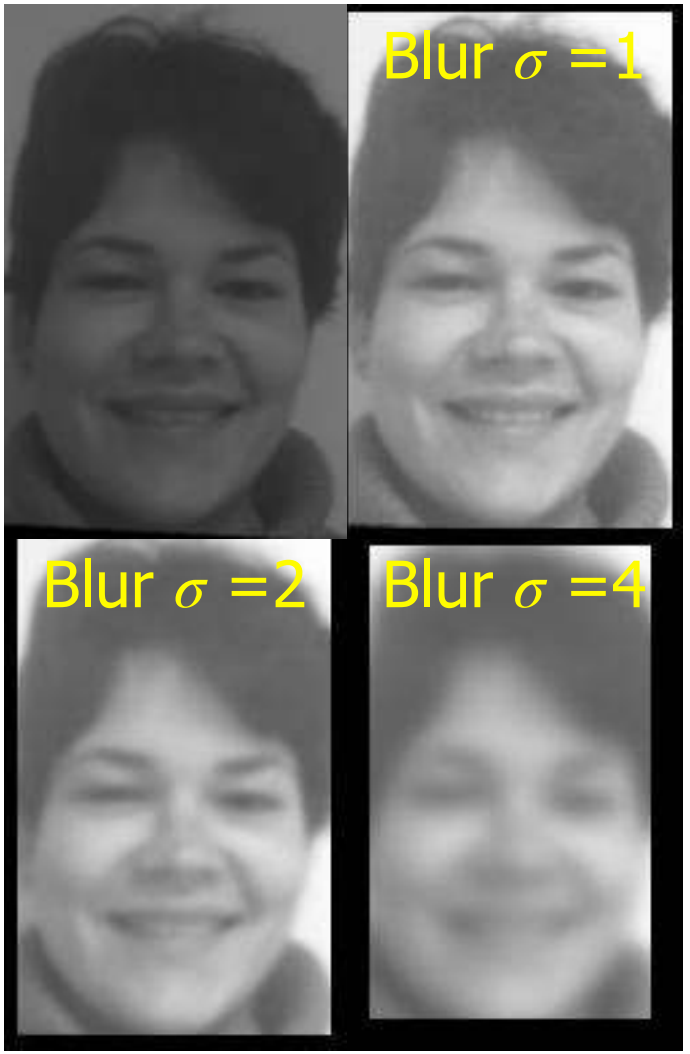
$\sigma = 1$

$\sigma = 2$

$\sigma = 4$



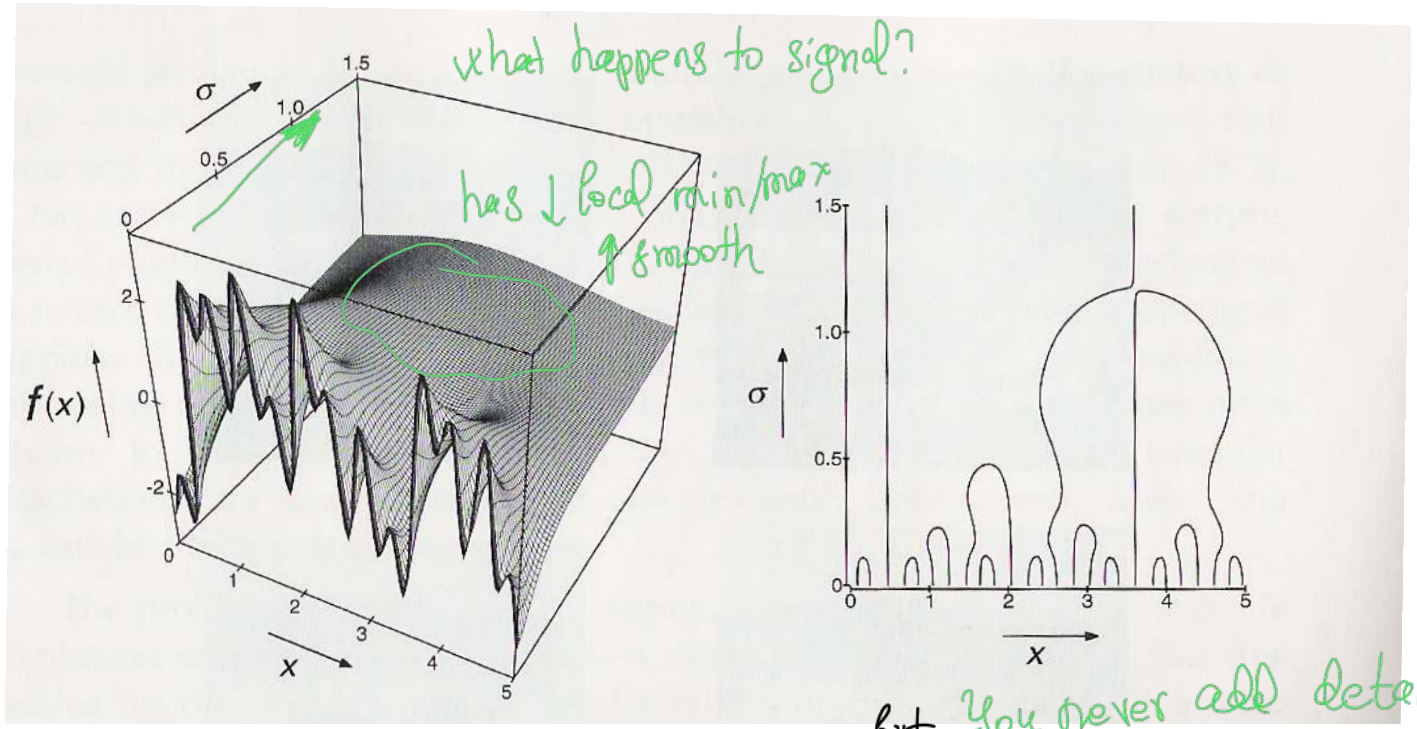
Original image Blurred image



No right value for $\sigma \rightsquigarrow 1$ details, resolution, ...

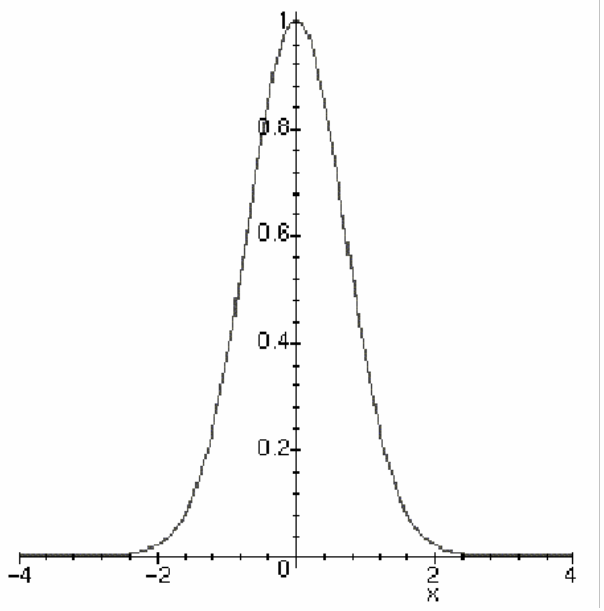
Lose more details

Scale Space



Increasing scale (σ) removes high frequencies (details) but never adds artifacts.

Separability



1D-gaussian

$$g_1(x) = \frac{1}{\sqrt{\pi}\sigma} \exp(-x^2/\sigma^2)$$

$$g_2(x,y) = g_1(x)g_1(y)$$

2D-gaussian

No need for explicit 2D-Gaussian convolutions

$$\int_u\int_v g_2(u,v) f(x-u, y-v) du dv = \int_u g_1(u)\left(\int_v g_1(v) f(x-u, y-v) dv\right) du$$

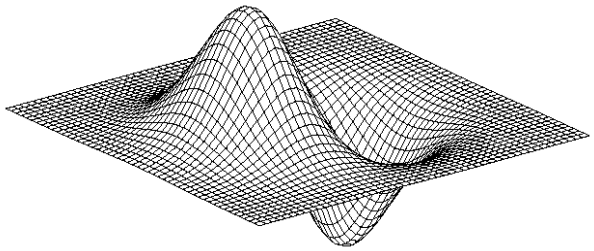
$$= \int_v g_1(v)\left(\int_u g_1(u) f(x-u, y-v) du\right) dv$$

2D-convolutions by Gaussian = ~ 2 × 1D convolution by Gaussian

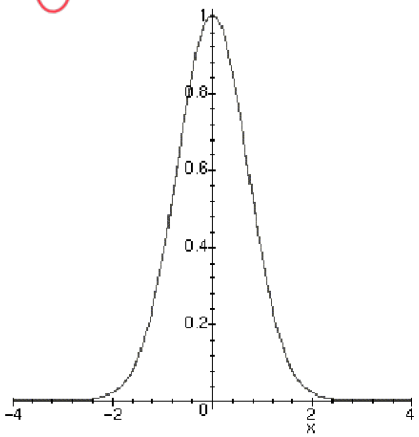
—> 2D convolutions are never required. Smoothing can be achieved by successive 1D convolutions, which is faster.

Continuous Gaussian Derivatives

$g_2(x,y) = g_1(x)\, g_1(y)$



=



x

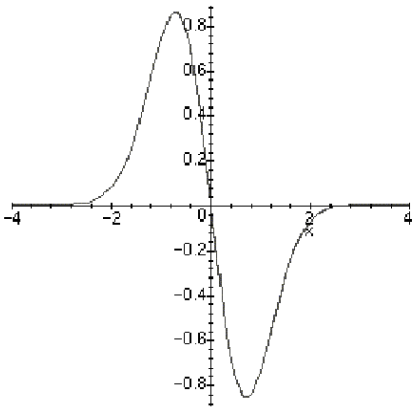


Image derivatives computed by convolving with the derivative of a Gaussian:

smoothed version

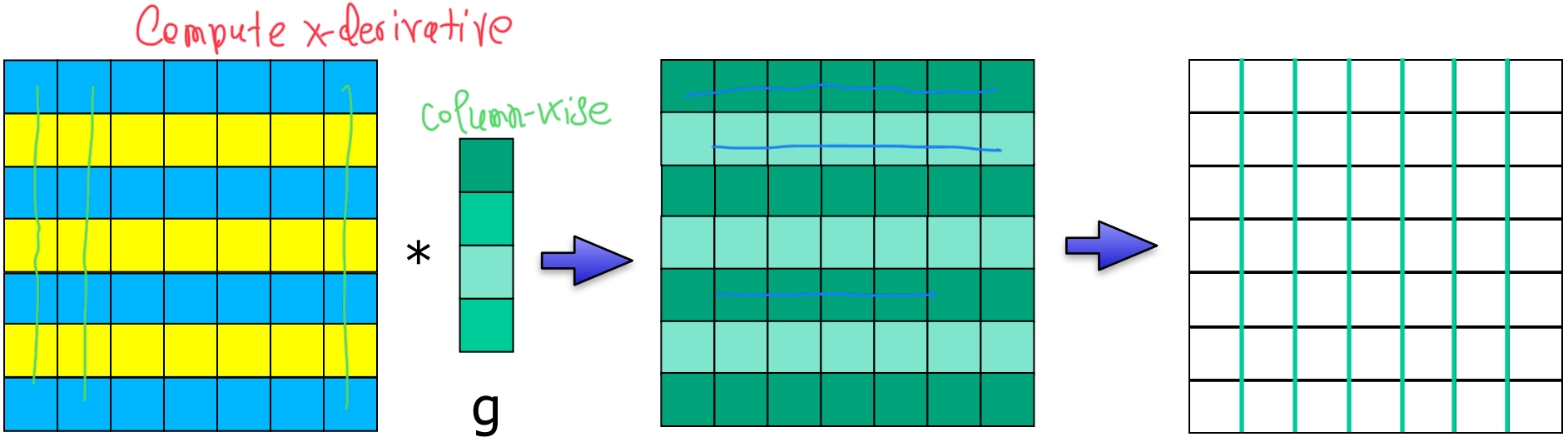
with Gaussian

with its derivative

$$\frac{\partial}{\partial x}\int_u\int_v g_2(u,v)\, f(x-u,y-v)\,du\,dv = \int_u g_1'(u)\left(\int_v g_1(v)\, f(x-u,y-v)\,dv\right)du$$

$$\frac{\partial}{\partial y}\int\int g_2(u,v)\, f(x-u,y-v)\,du\,dv = \int_v g_1'(v)\left(\int_u g_1(u)\, f(x-u,y-v)\,du\right)dv$$

Discrete Gaussian Derivatives



Sigma=1:

pre-computed

g : 0.000070 0.010332 0.207532 0.564131 0.207532 0.010332 0.000070 → centered mask

g': 0.000418 0.041330 0.415065 0.000000 -0.415065 -0.041330 -0.000418

Pen=6 ✓

Sigma=2:

↑support, ↑Pen of mask g

fast, efficient ★

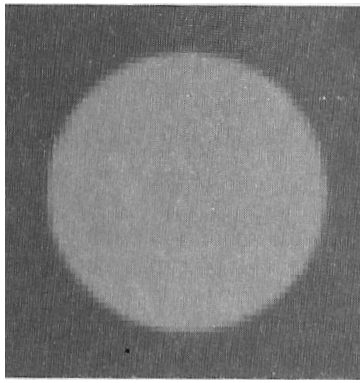
g : 0.005167 0.029735 0.103784 0.219712 0.282115 0.219712 0.103784 0.029735 0.005167

g': 0.010334 0.044602 0.103784 0.109856 0.000000 -0.109856 -0.103784 -0.044602 -0.010334

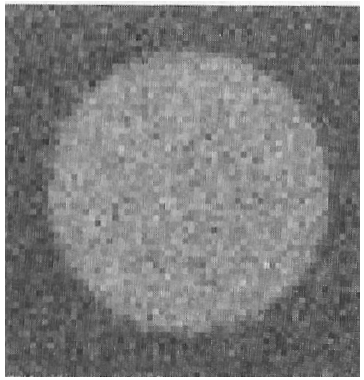
—> Only requires 1D convolutions with relatively small masks.

Increasing Sigma

Input Images



No Noise



Noise Added

No noise: As you ↑σ, your mask pen ↑ with more convolution op's ⇒ flatten contours

Localization → not good

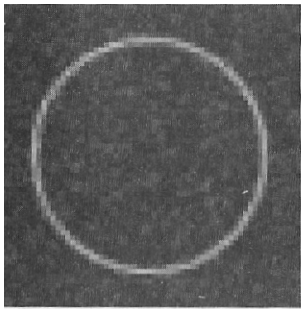
σ=1

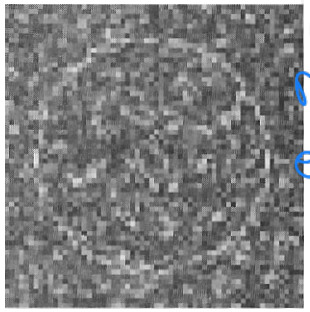
σ=2

Robustness to noise vs Precision of the detection

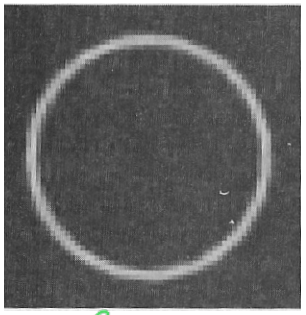
σ=4

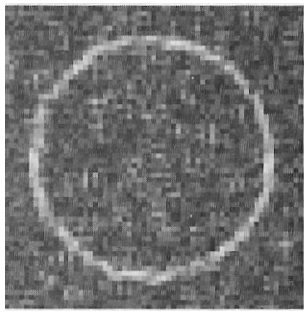
Gradient Images



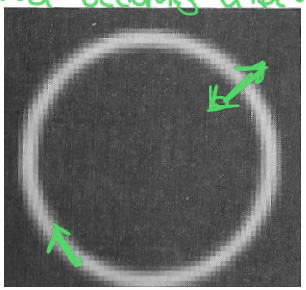


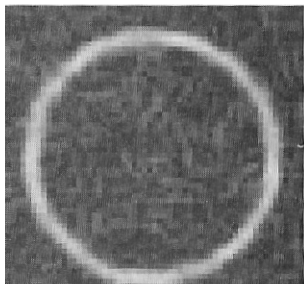
high responses everywhere





response becomes wider





sparse (few) where edges response is

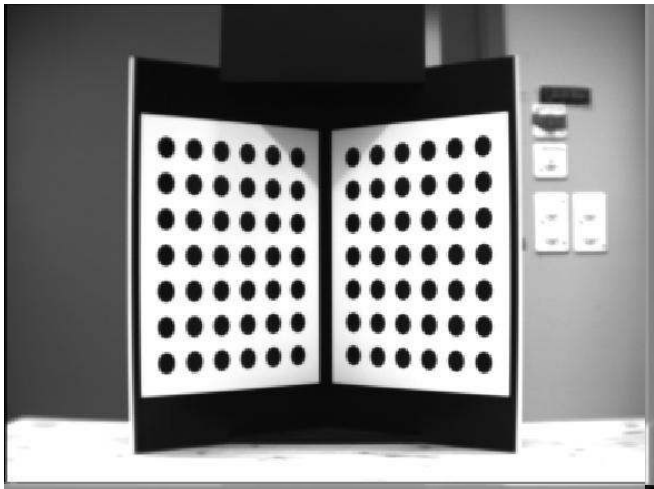
No Noise

Noise Added

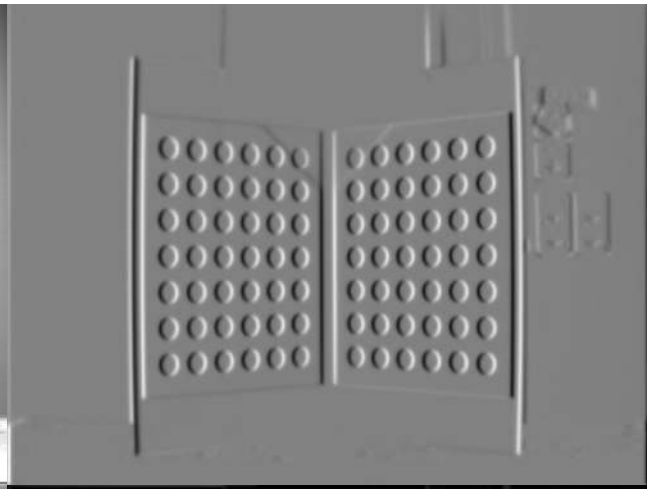
—> Larger sigma values improve robustness but degrade precision.

Derivative Images

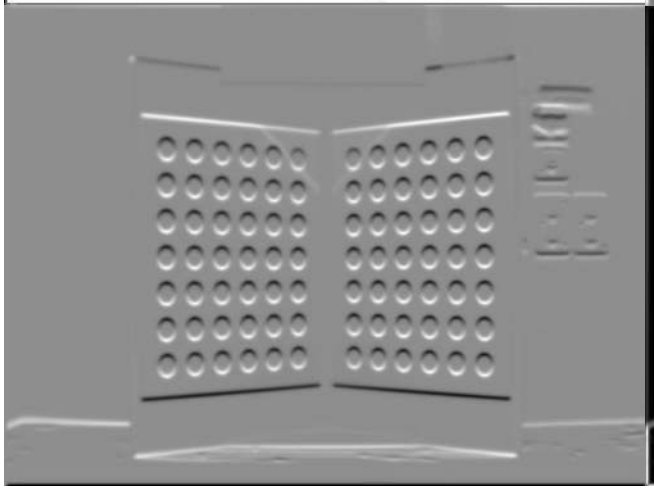
I



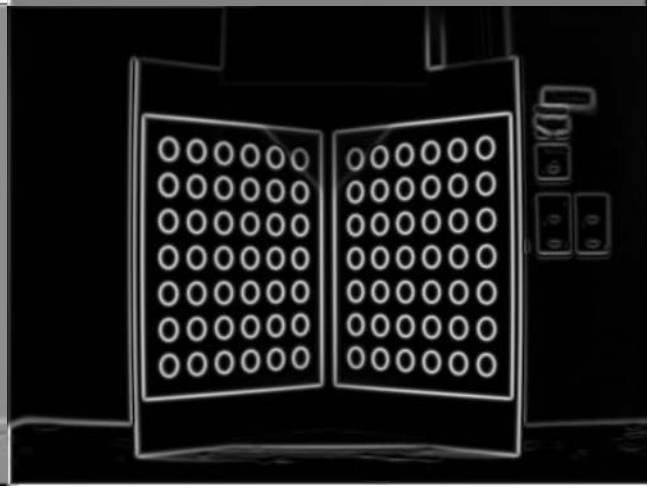
$\frac{\partial I}{\partial x}$



$\frac{\partial I}{\partial y}$



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

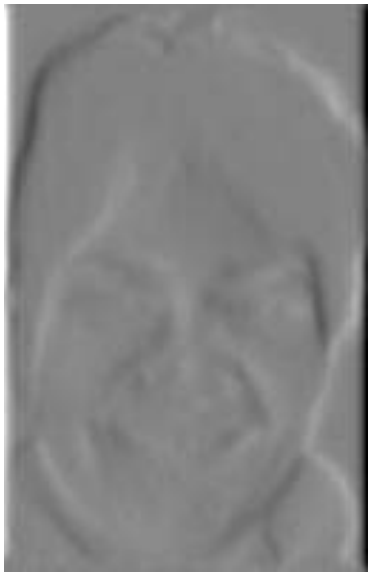


Derivative Images

Strong responses ✓
But also unnecessary places ✓

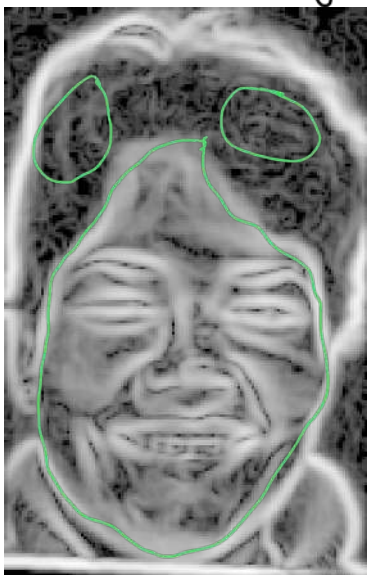
gradient image

I



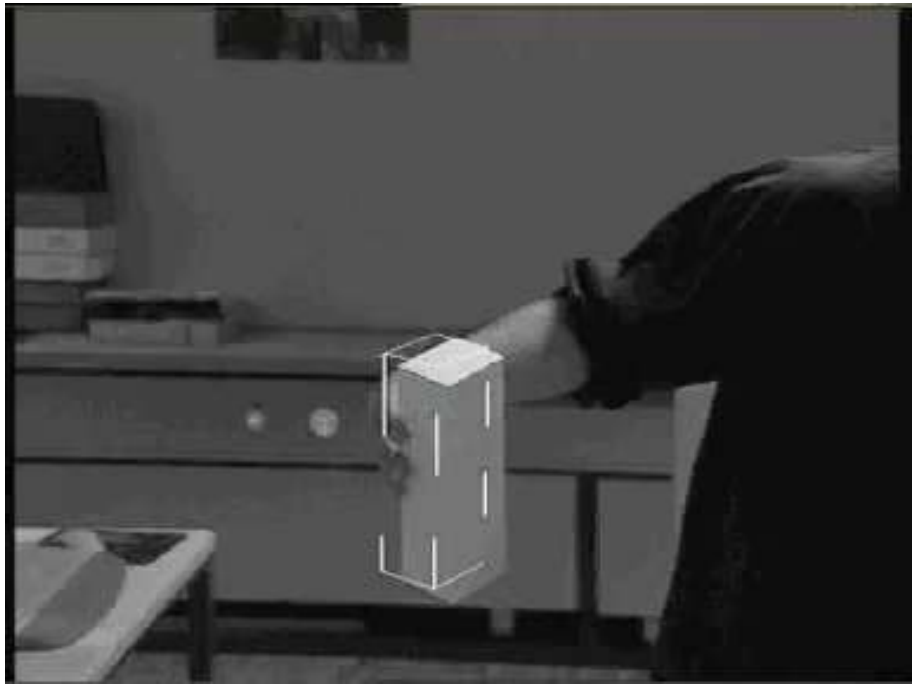
$\frac{\partial I}{\partial x}$

$\frac{\partial I}{\partial y}$



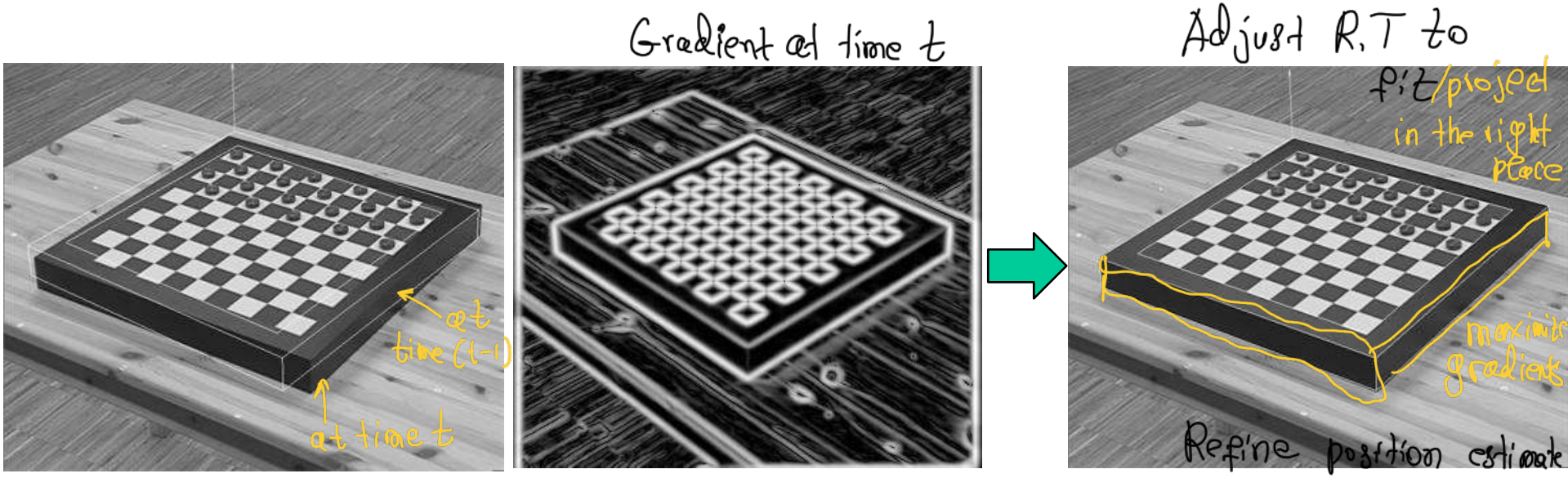
$\sqrt{\frac{\partial I}{\partial x}^2 + \frac{\partial I}{\partial y}^2}$

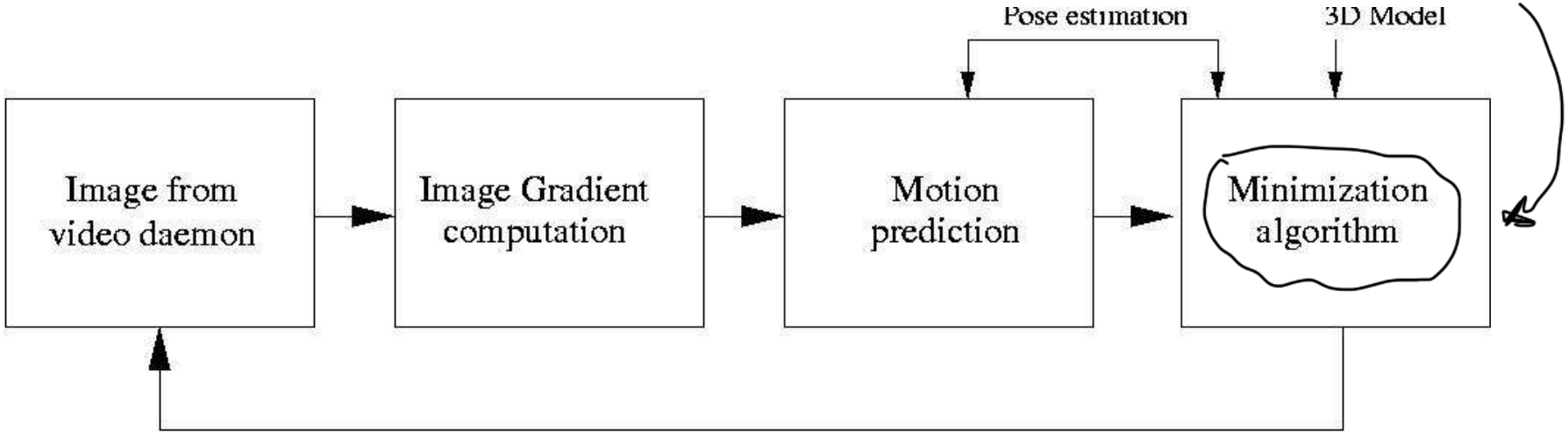
Gradient-Based Tracking



Maximize edge-strength along projection of the 3—D wireframe.

Gradient Maximization





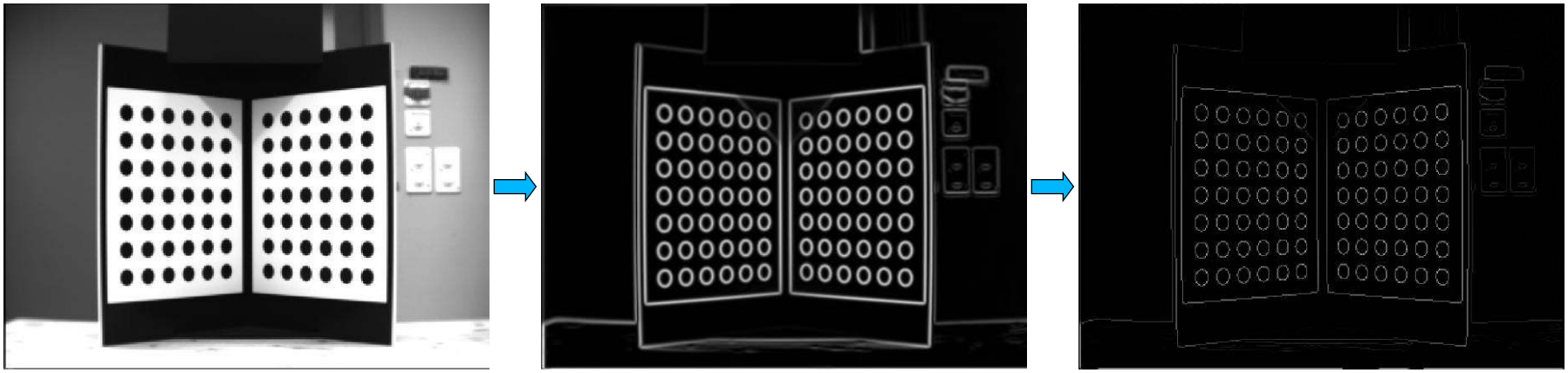
Real-Time Tracking

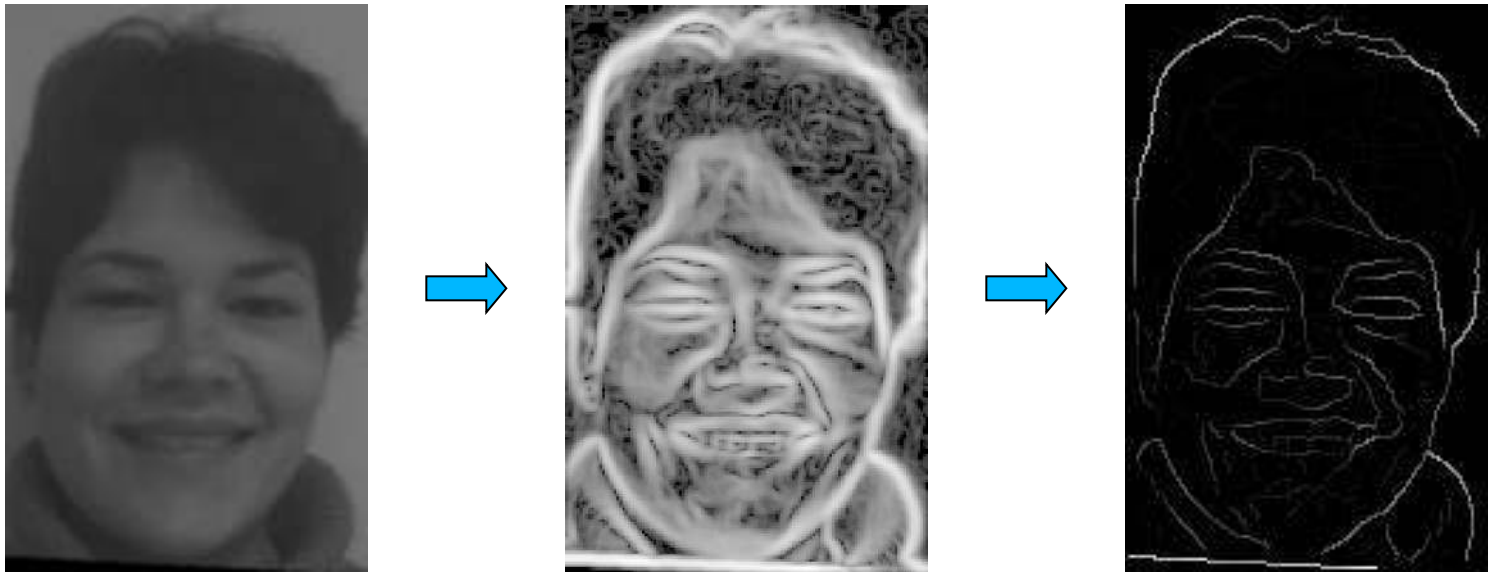
Results from Gradient-computation are not perfect, but they're already exploitable

Canny Edge Detector

I $\sqrt{\frac{\partial I^2}{\partial x} + \frac{\partial I^2}{\partial y}}$ Thinned gradient image

Thick





1-pixel Right

Canny Edge Detector

$\left[\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}\right]$ at $\forall$ point (x, y)

Gradient image

Convolution $\sqrt{\left(\frac{\partial I}{\partial x}\right)^2 + \left(\frac{\partial I}{\partial y}\right)^2}$

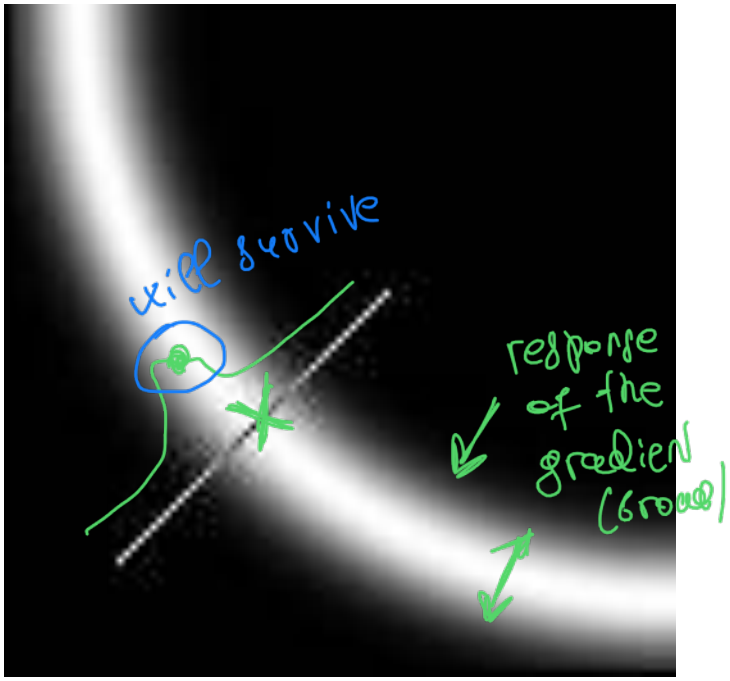
- Gradient strength
- Gradient direction arctan

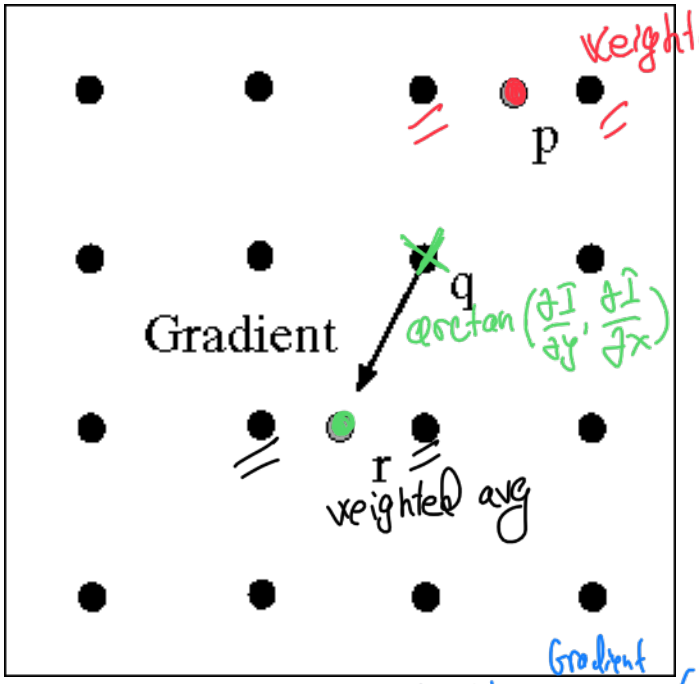
Thresholding

Non Maxima Suppression keep specific pixels instead of wide # of pixels

Hysteresis Thresholding

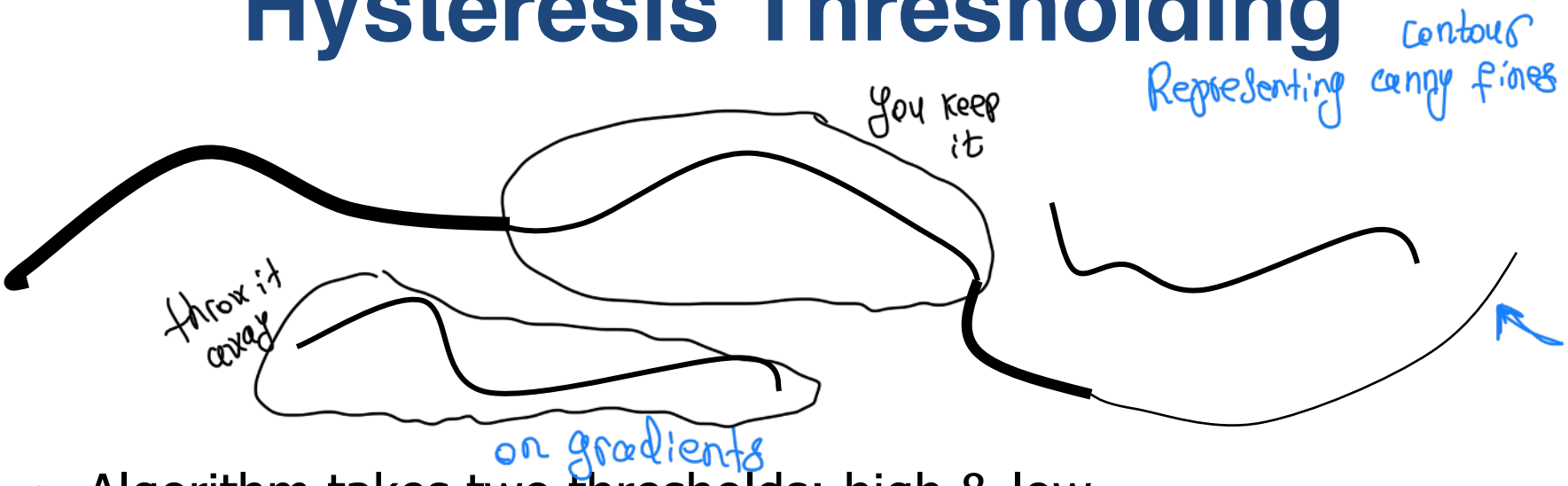
Non-Maxima Suppression





Check if pixel is local maximum along gradient direction, which requires checking interpolated pixels p and r.

Hysteresis Thresholding



- Algorithm takes two thresholds: high & low
  - A pixel with edge strength above high threshold is an edge.
  - Any pixel with edge strength below low threshold is not.
  - Any pixel above the low threshold and next to an edge is an edge.

- Iteratively label edges
  - Edges grow out from 'strong edges'
  - Iterate until no change in image.

Canny Results

Edge detector

good contours ✓

σ=1, T2=255, T1=1

high low

↝ it can not close the "T"

'Y' or 'T' junction problem with Canny operator

Canny Results

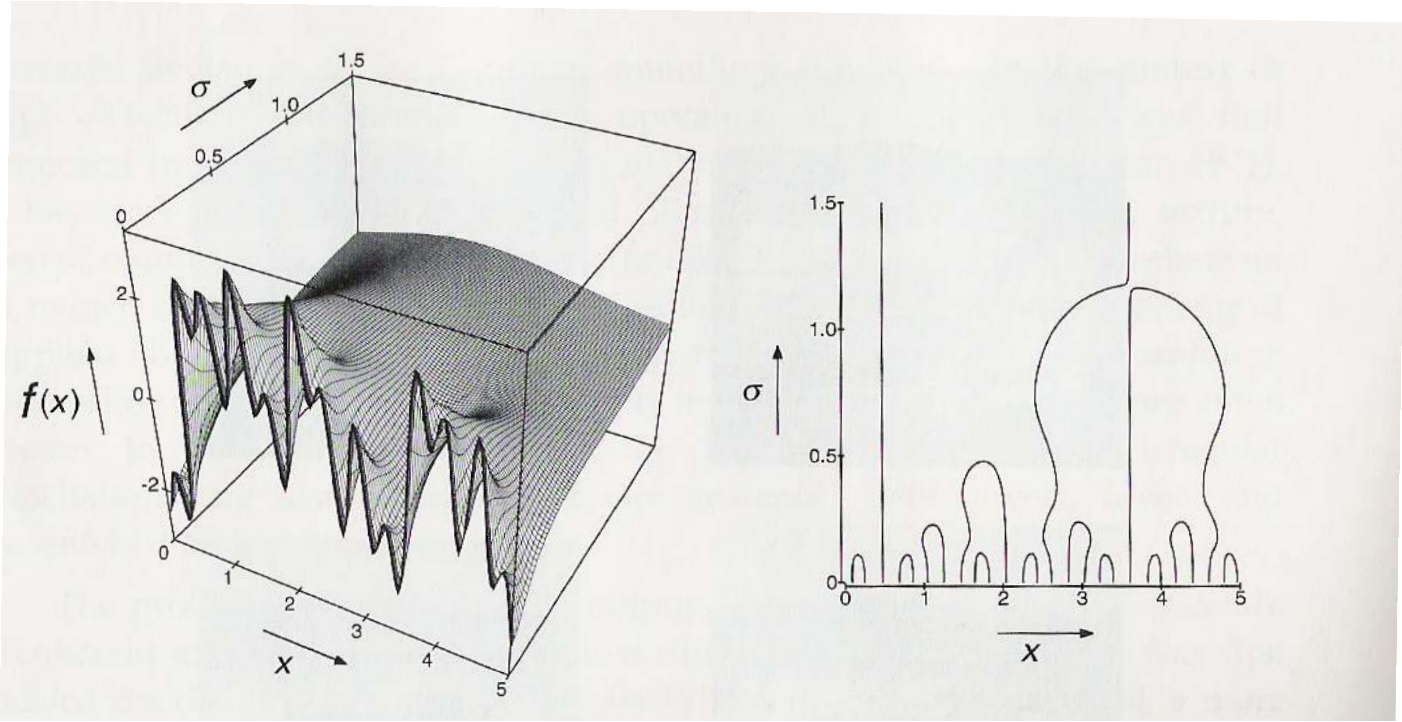
σ=1, T2=255, T1=220

σ=1, T2=128, T1=1

σ=2, T2=128, T1=1

↑6 ⇒ gets rid of contours
high freq's
never adds new contours

Scale Space Revisited



Increasing scale (σ) removes details but never adds new ones:

- Edge position may shift.
- Two edges may merge.
- An edge may **not** split into two.

Multiple Scales

$\sigma = 1$

$\sigma = 2$

$\sigma = 4$

→Choosing the right scale is a difficult semantic problem.

Scale vs Threshold

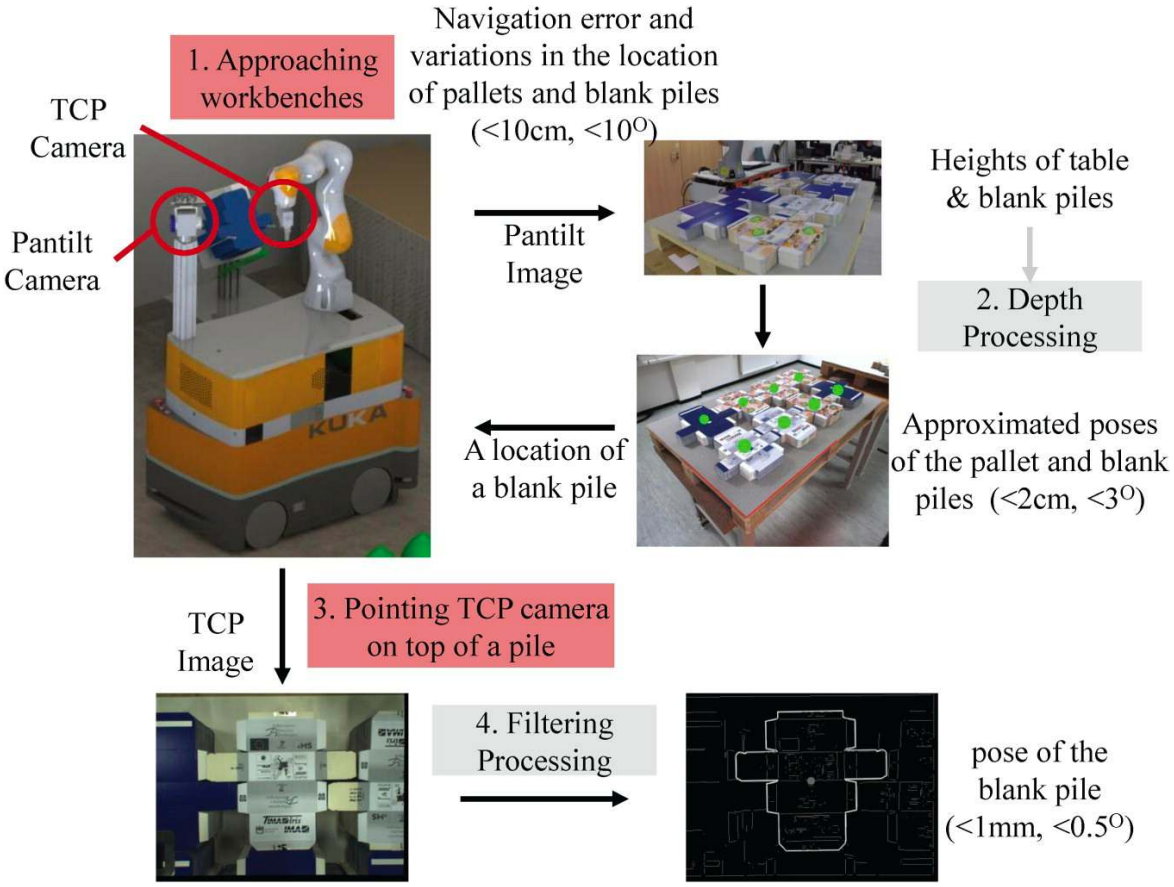
difference this makes huge

Fine scale
High threshold

Coarse scale
High threshold

Coarse scale
Low threshold

Industrial Application



In industrial environments where the Canny parameters can be properly adjusted:

You can control lighting

- It is fast.
- Does not require training data.

Lim et al. , Advanced Robotics 2019.

Visual Servoing

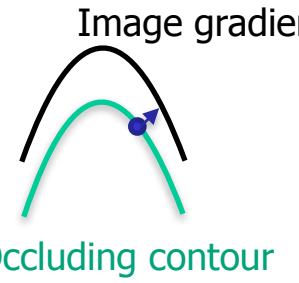
—> A useful tool in our toolbox.

Tracking a Rocket

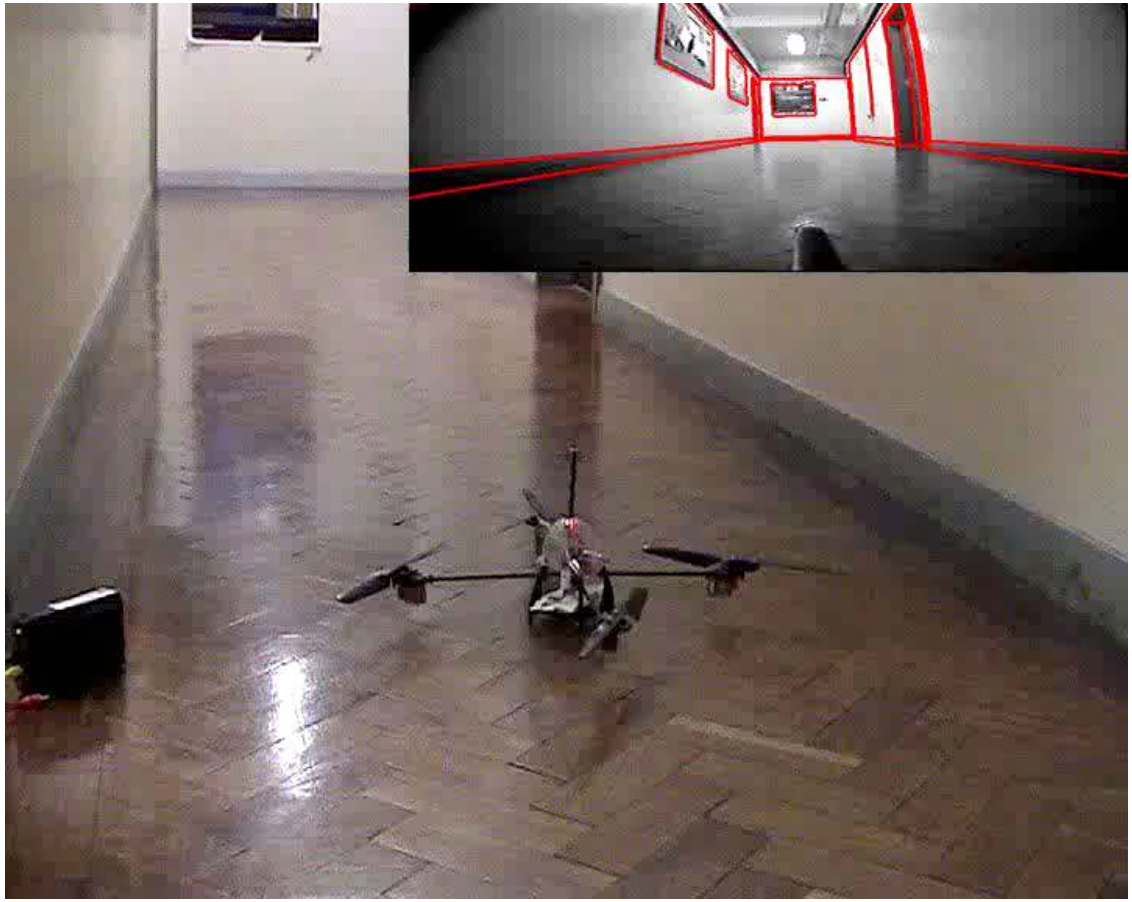



Given an initial pose estimate:

- Find the occluding contours. using canny
- Find closest edge points in the normal direction.
- Re-estimate pose (object) to minimize sum of square distances.
- Iterate until convergence.



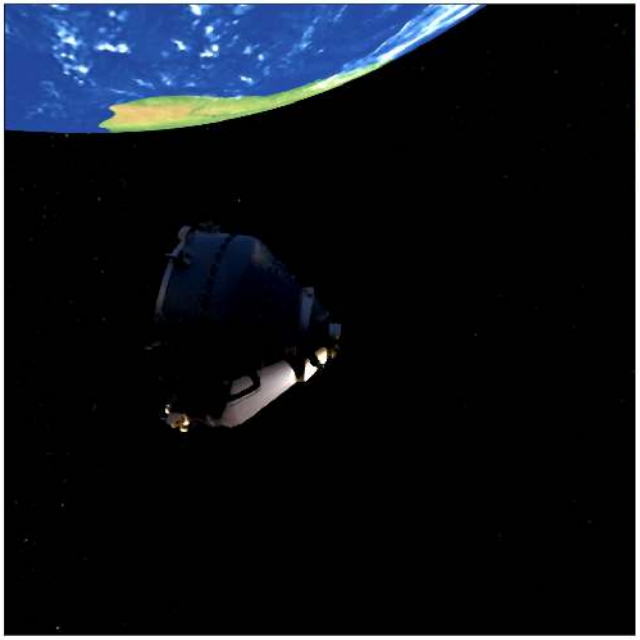
Visual Servoing

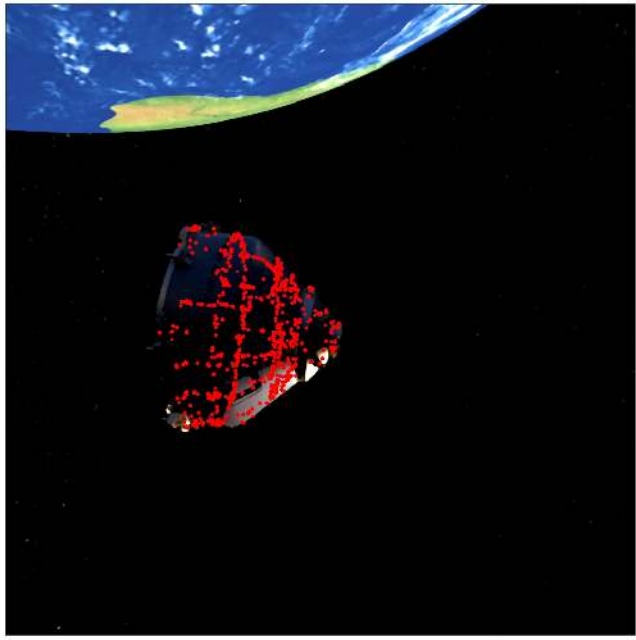


You can set parameters T_1, T_2

Space Cleaning

Capturing and deorbiting a dead satellite.





- A more sophisticated version of this old algorithm will blast off in 2025!
- ESA does not yet trust neural nets for such a mission.

Limitations of the Canny Algorithm

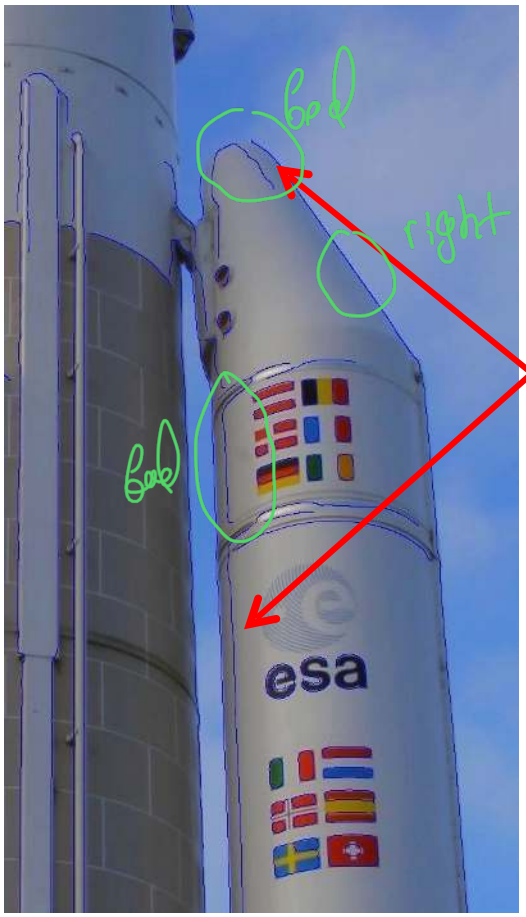
Find contours of the road

increasing σ

There is no ideal value of σ!

If you want to find edges of the road ⇒ no particular σ works

Steep Smooth Shading

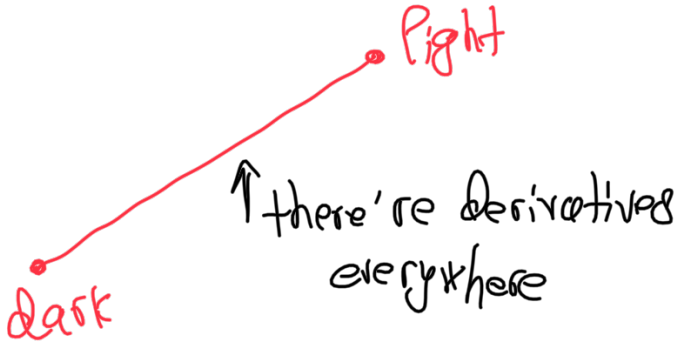


Canny → finding step edges to find where there're a lot of gradients

→ creates large gradients

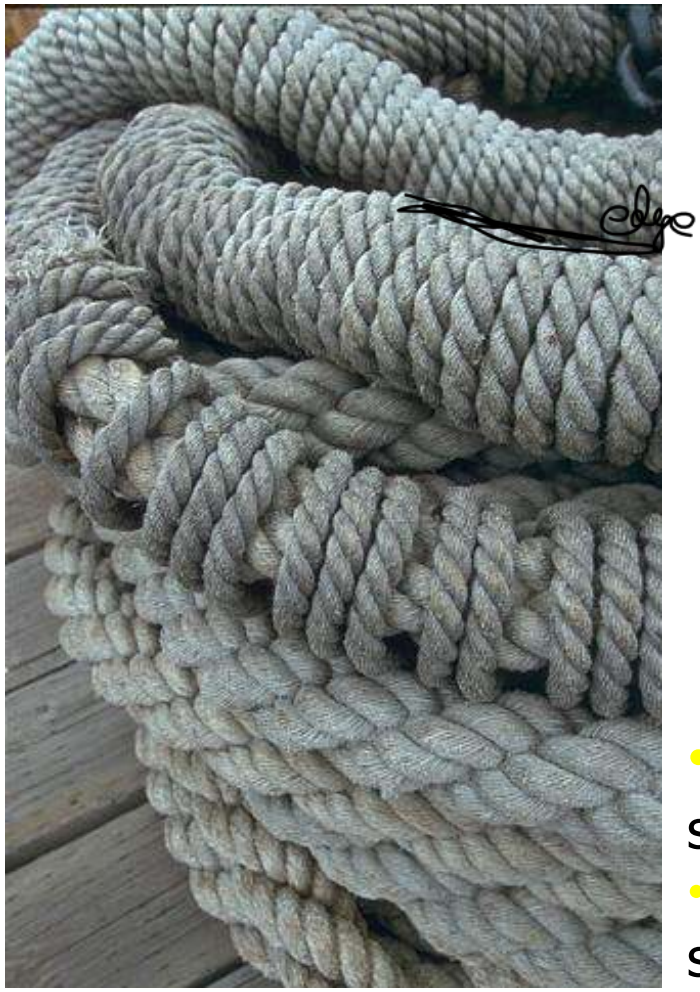
Transition light → dark continuously

- Rapidly varying gray levels.
- Large gradients.



→ Shading can produce spurious edges.

Texture Boundaries



- Not all image contours are characterized by strong contrast.
- Sometimes, textural changes are just as significant.

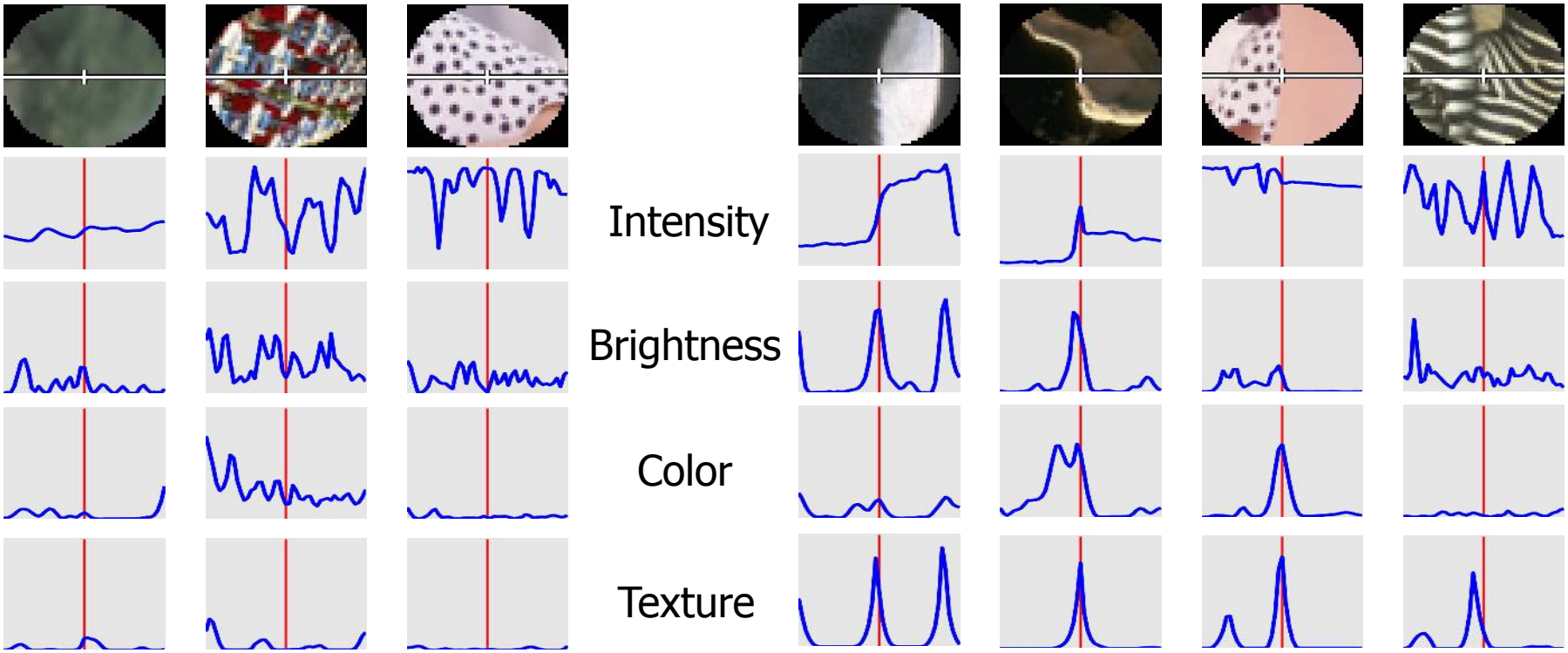
Different Boundary Types

No contours

Non-boundaries

There are

Boundaries



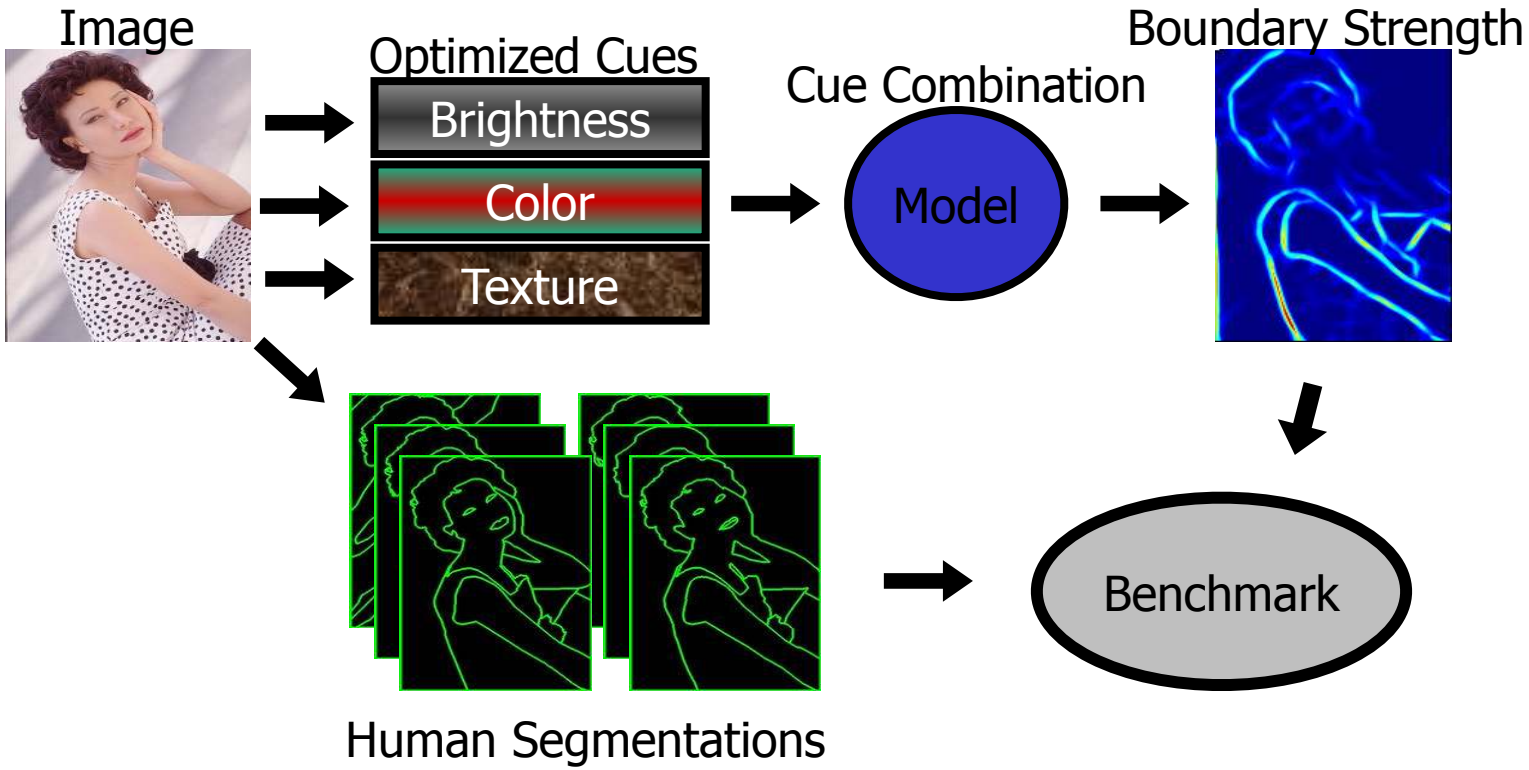
Teach ML model to detect boundaries

Martin et al. , PAMI'04

Training Database

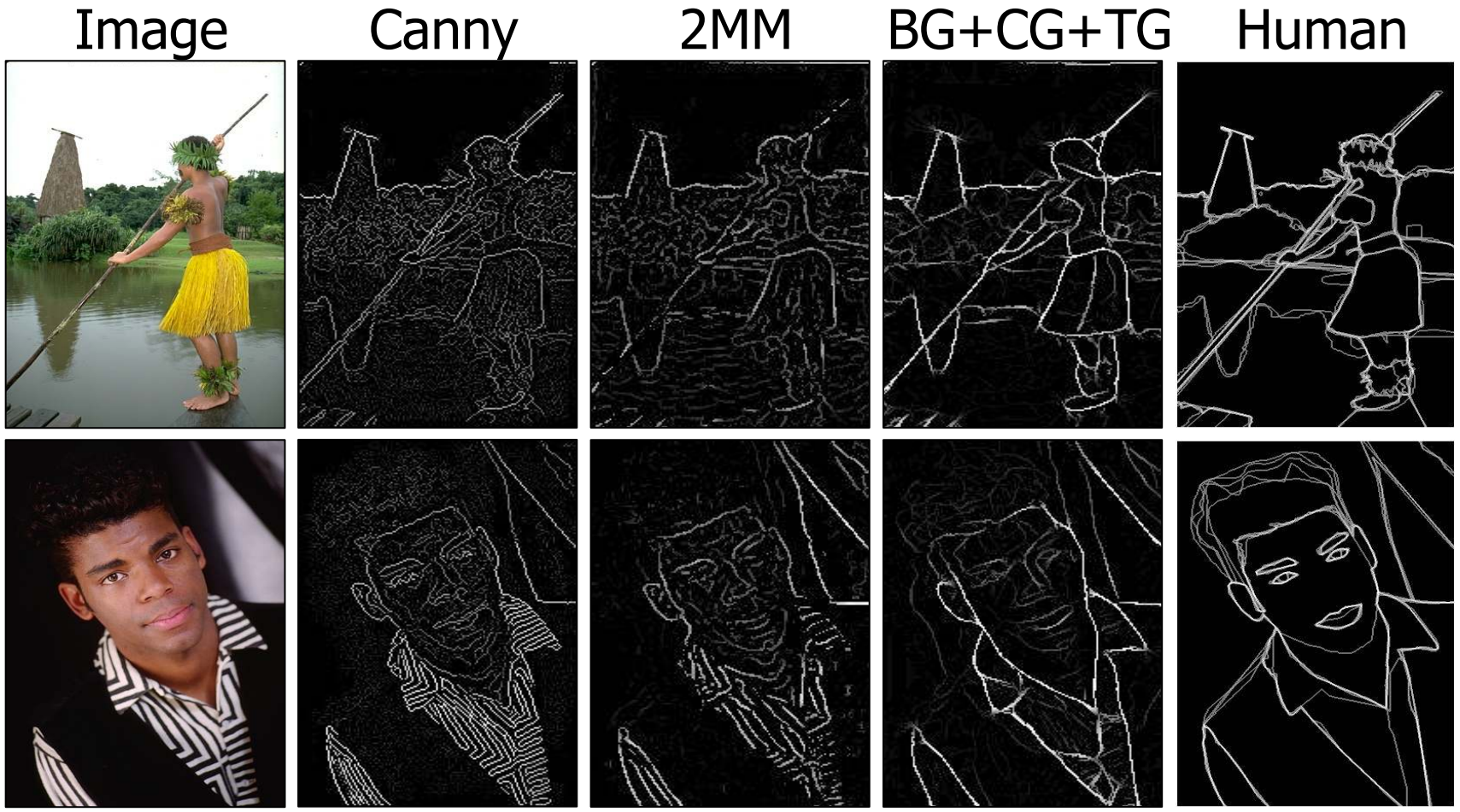
1000 images with 5 to 10 segmentations each.

Machine Learning



Learn the probability of being a boundary pixel on the basis of a set of features.

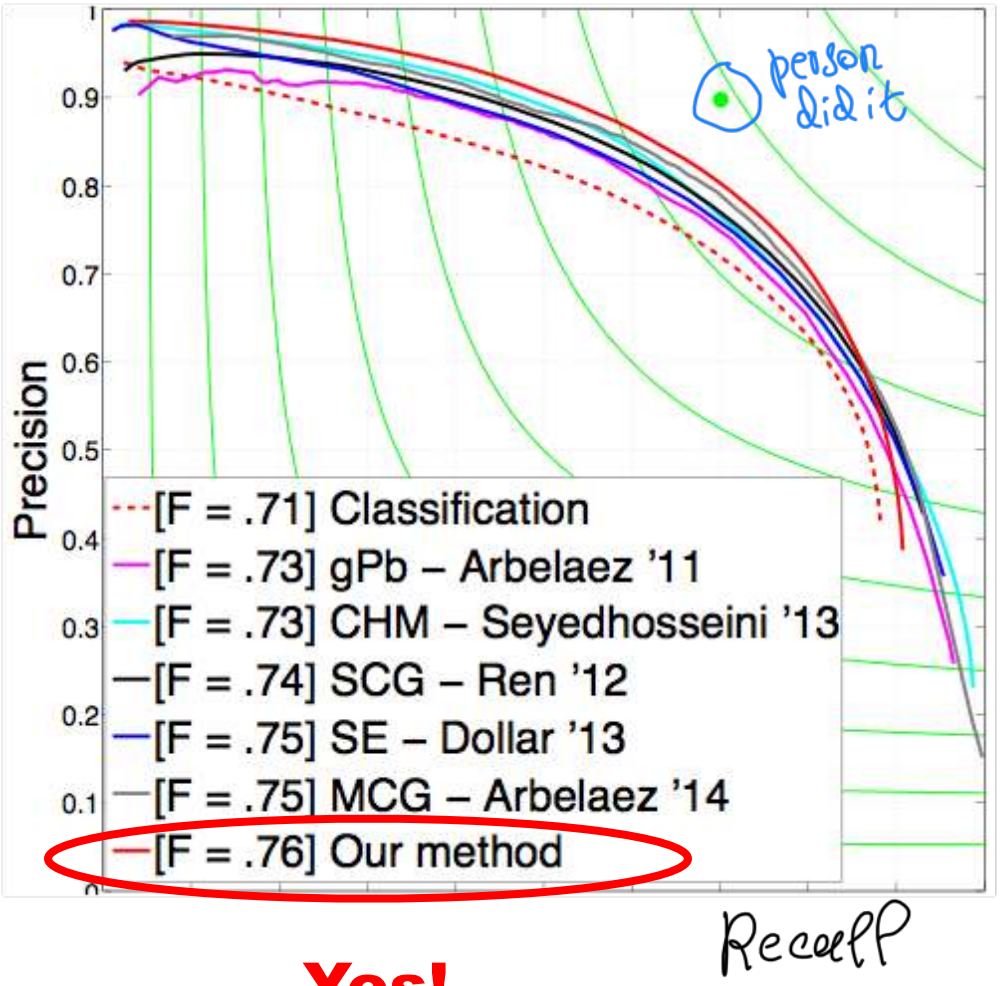
Comparative Results

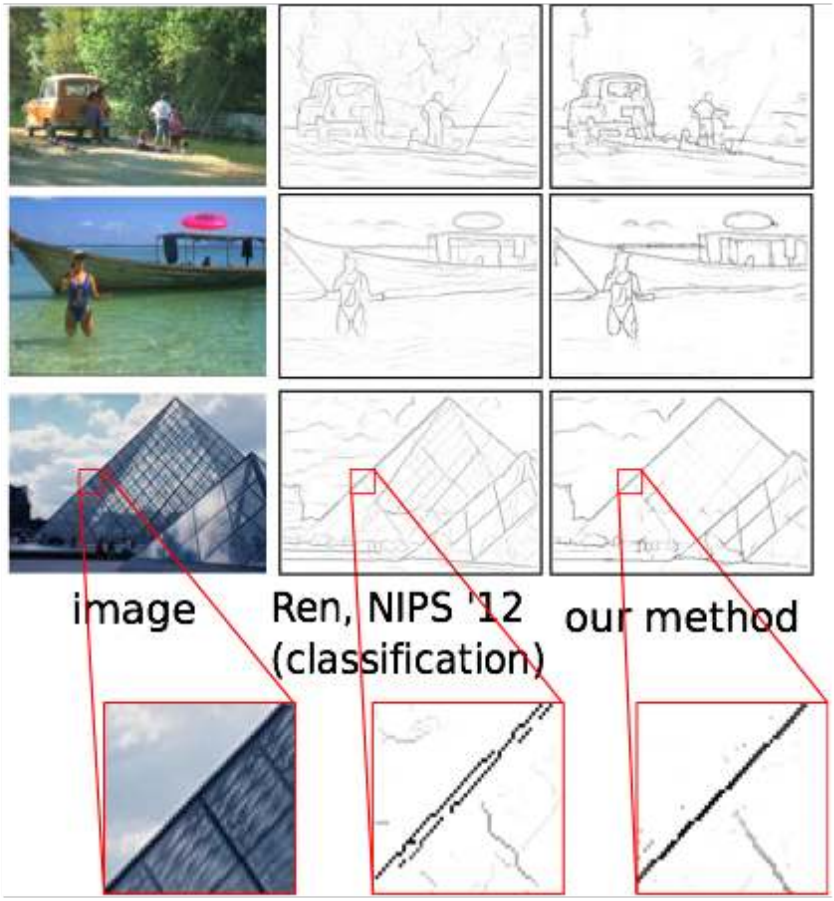


Find contours which person would find

Martin et al. , PAMI'04

Classification vs Regression





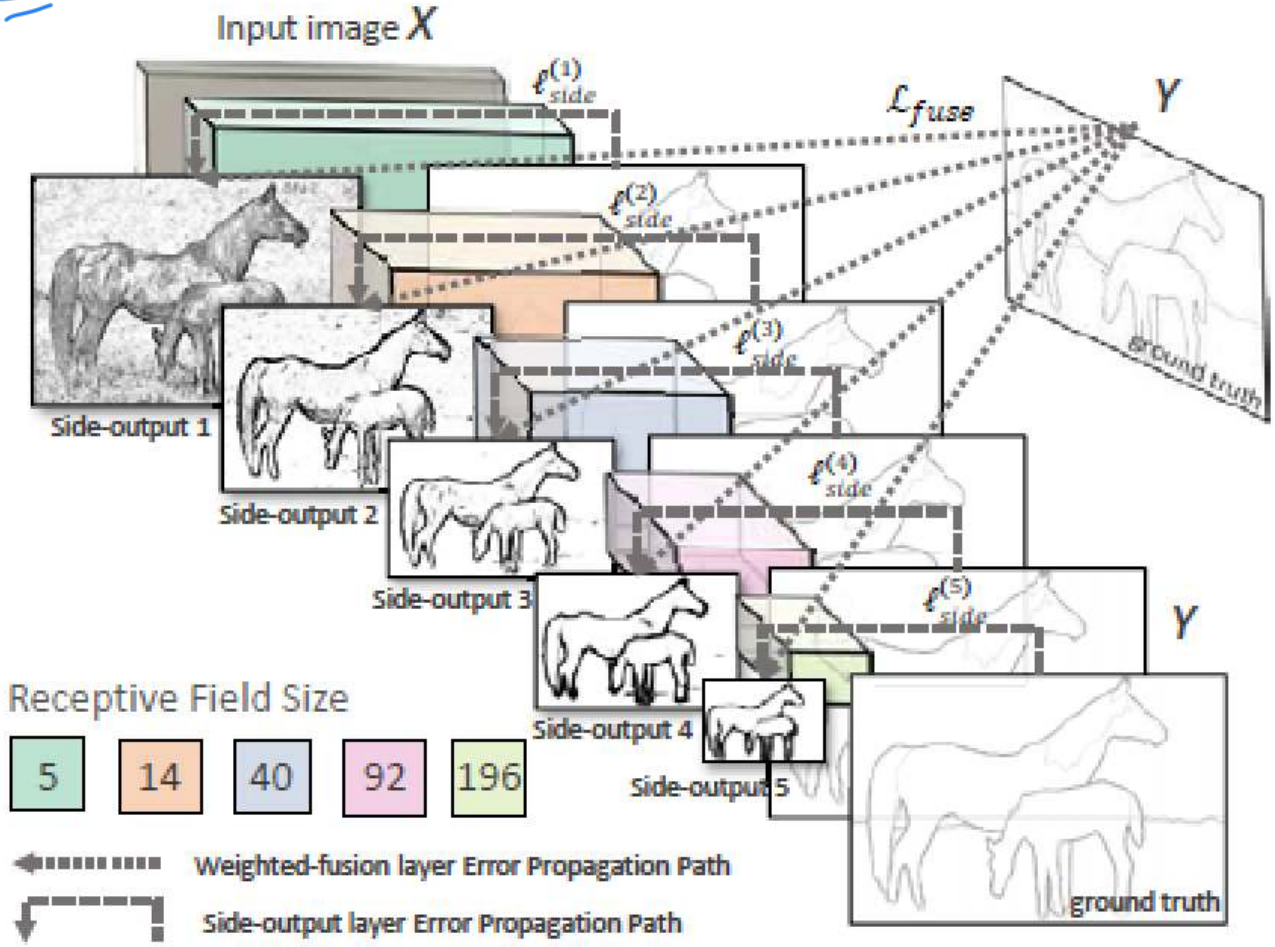
Yes!

human performance → not perfect

Sironi, et al. PAMI'15

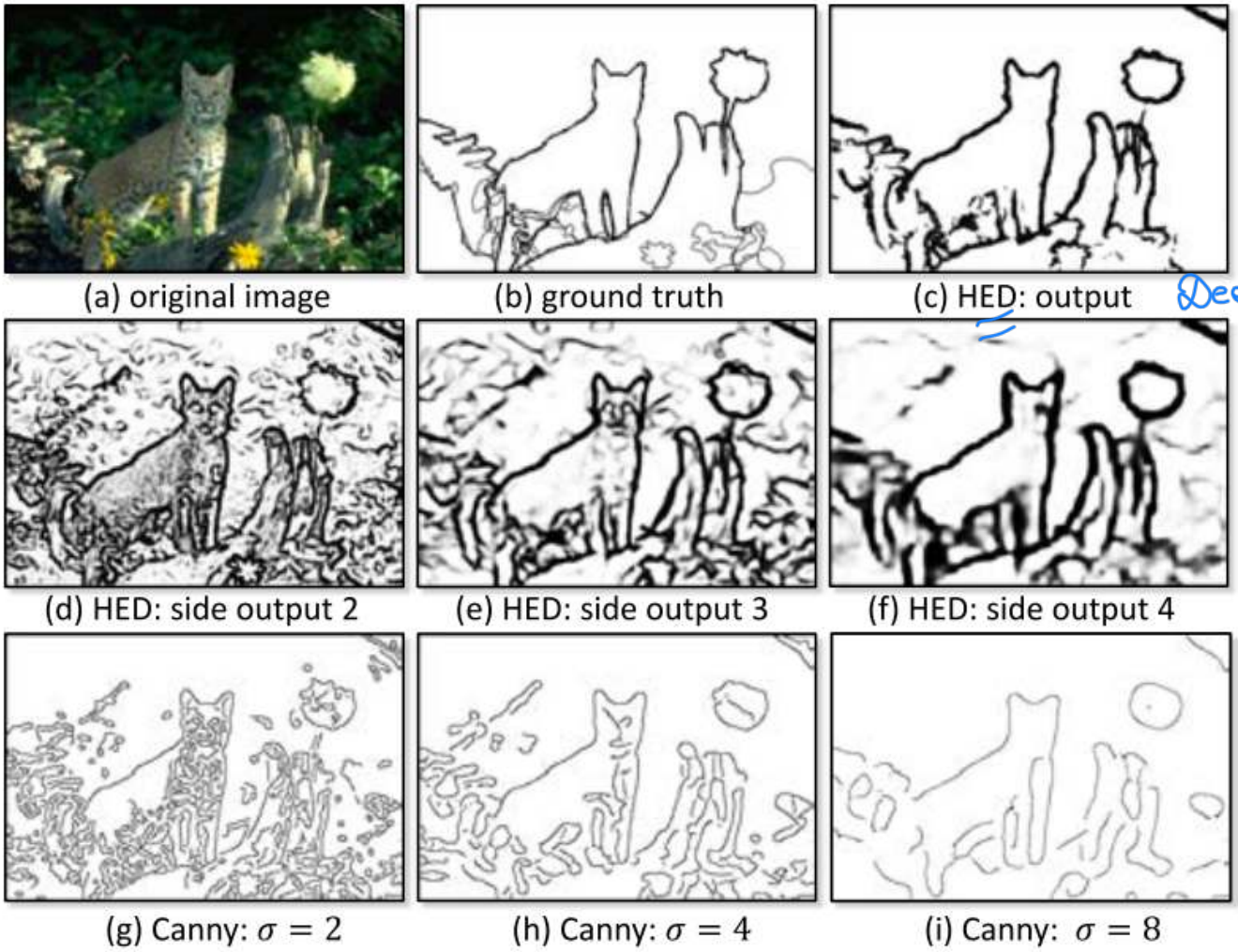
Deep Learning

ML on contours ⇒ DL these days



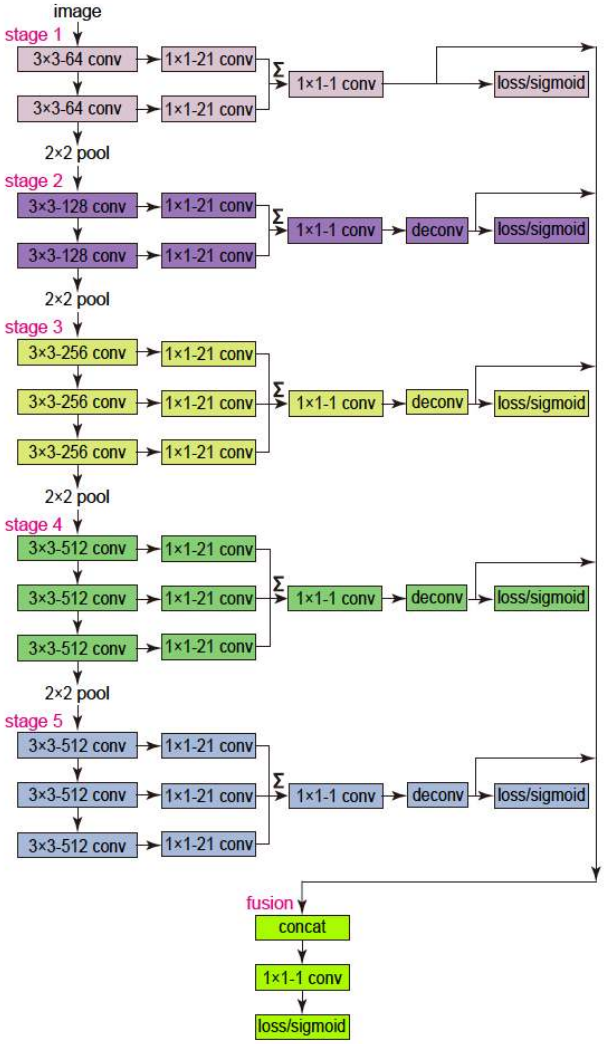
Deep Learning Vs Canny

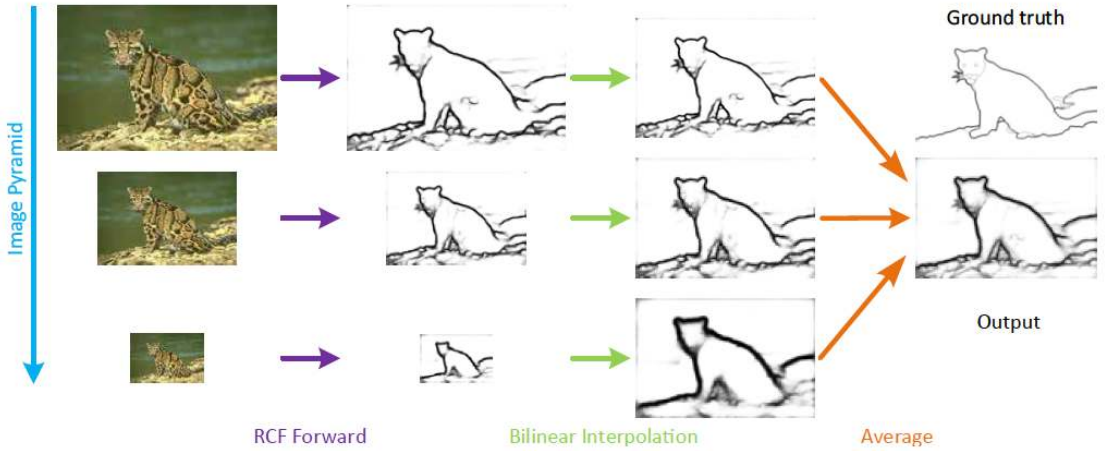
finding relevant contours



finding significant contours → hard

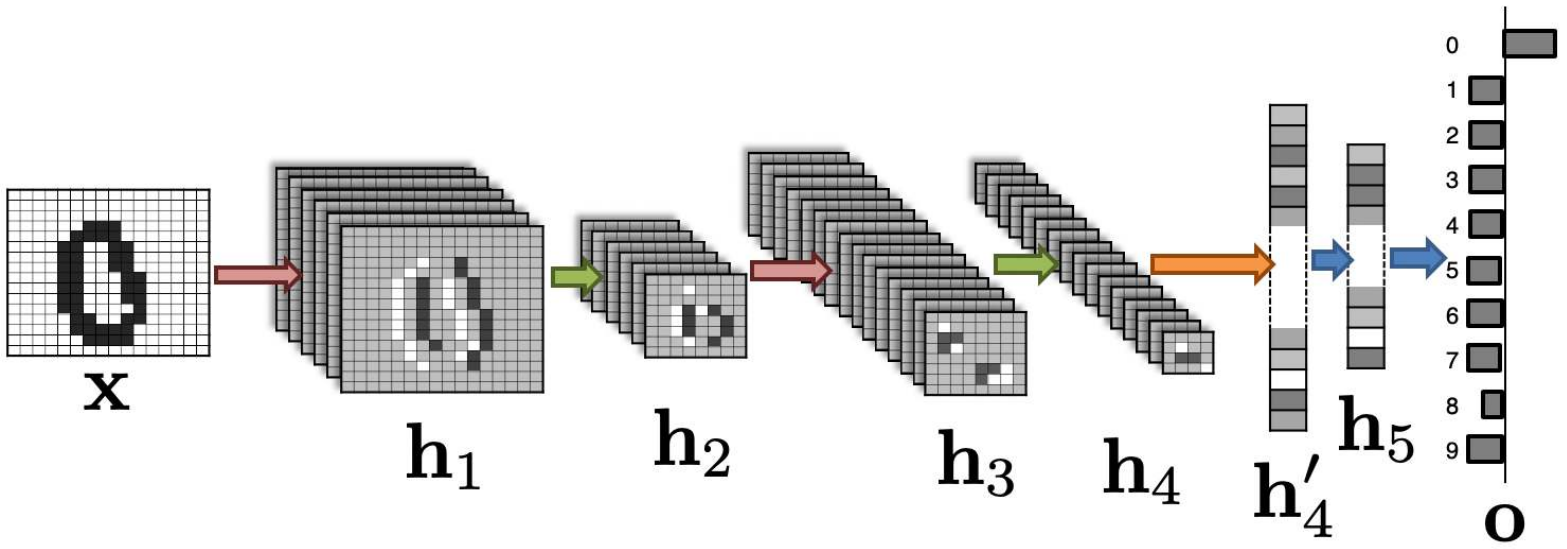
Deeper Learning





Liu et al. , CVPR'17

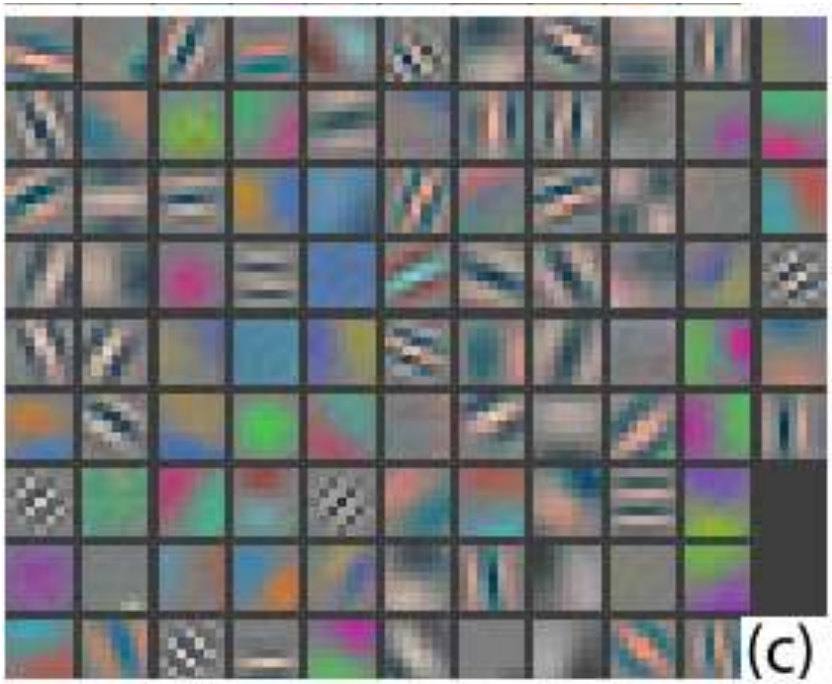
Convolutional Neural Network

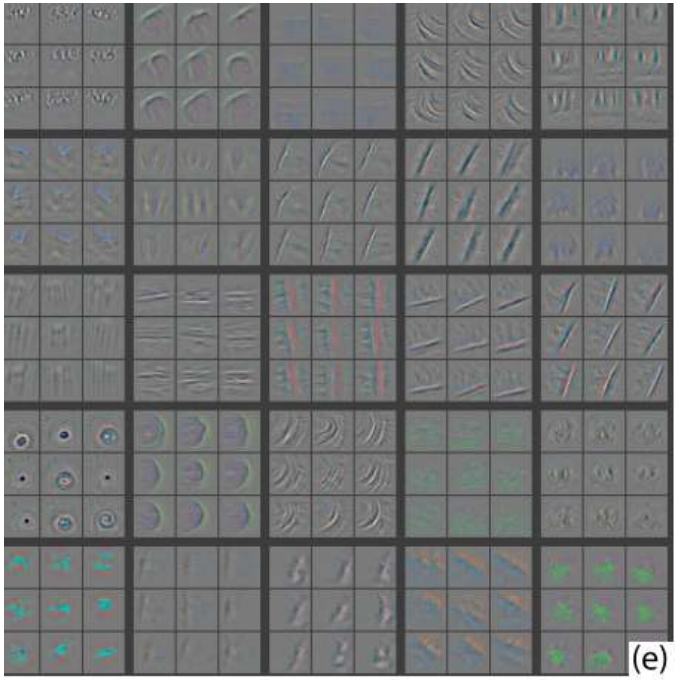


- Succession of convolutional and pooling layers.
- Fully connected layers at the end.

—> Will be discussed in more detail in the next lecture.

A Partial Explanation?



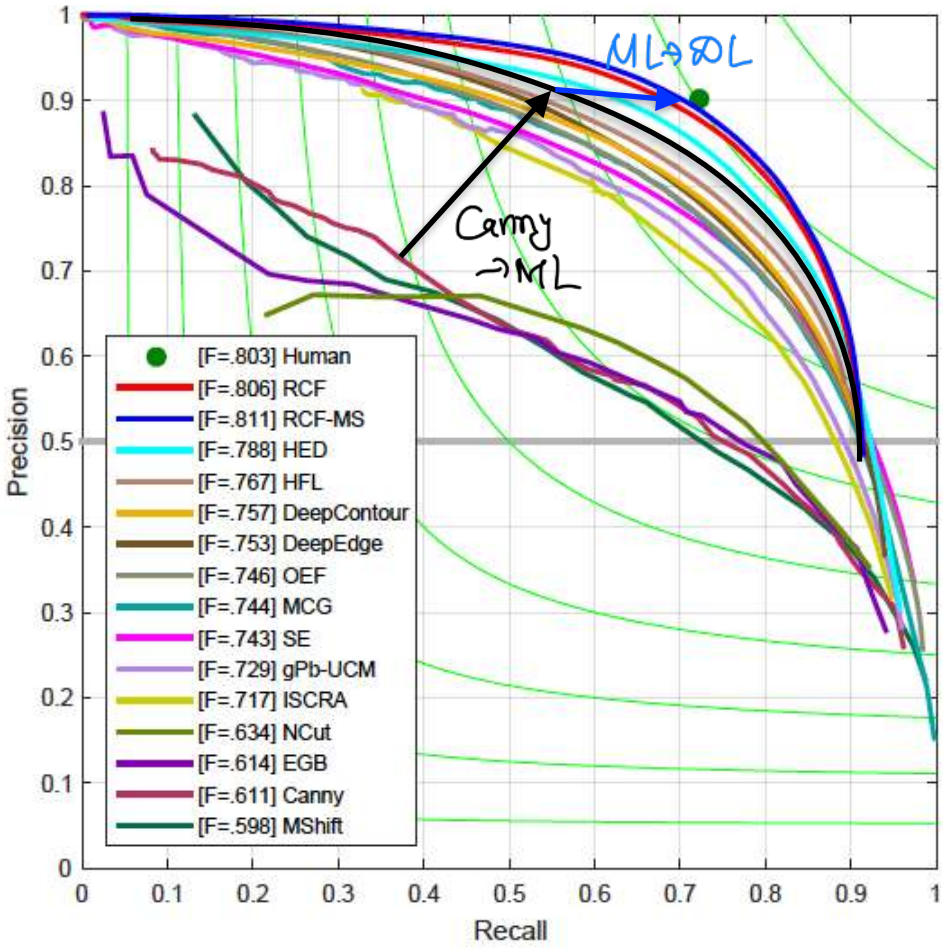


Derivatives of Gaussians ??

First and second layer features of a Convolutional Neural Net:

- They can be understood as performing multiscale filtering.
- The weights and thresholds are chosen by the optimization procedure.

50 Years Of Edge Detection



In rockets, gradual transition dark white image derivatives are computed so they get fired everywhere

- Convolution operators respond to steep smooth shading.
- Parametric matchers tend to reject non ideal edges. Too noisy
- Arbitrary thresholds and scale sizes are required. σ, T_1, T_2
- Learning-based methods need exhaustive databases. for ML

  Canny survived → no training database needed
- There still is work to go from contours to objects.

Canny, PAMI'86 —> Sironi et al. PAMI'15

Sironi et al. PAMI'15 —> Liu et al. , CVPR'17

Let us talk about deep networks.