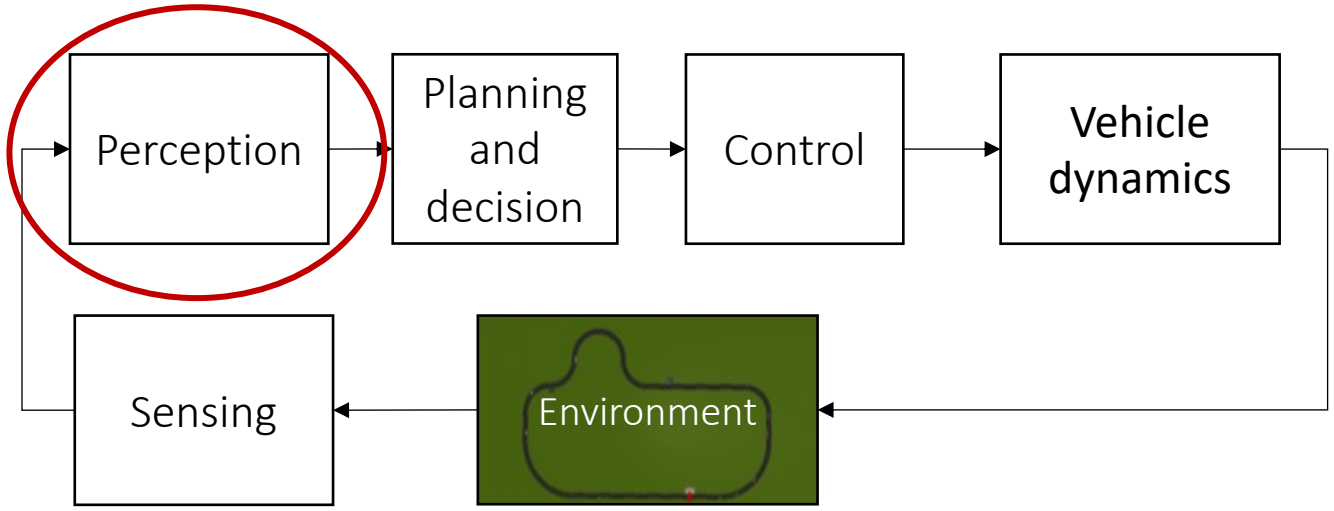
Illinois Code a Car Summer Camp

Day 2

August 1-5 2022

Anatomy of a robotaxi

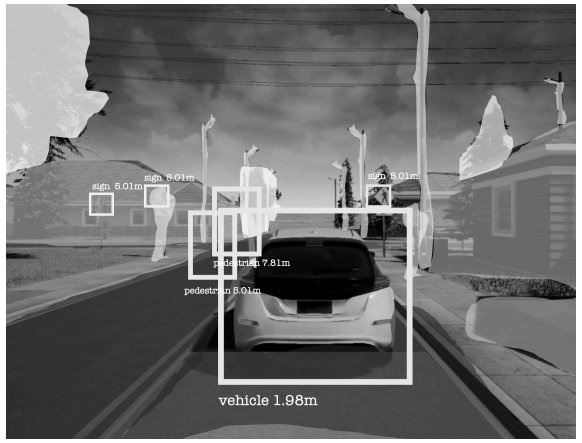
Sensing	Perception	Decisions and planning	Control
Physics-based models of cameras, LIDAR, radar, GPS, and so on.	Programs for object tracking, scene understanding, and so on.	Programs and multi-agent models of pedestrians, cars, and so on.	Dynamical models of vehicle engine, powertrain, steering, tires, and so on.



env: Lidar / vision

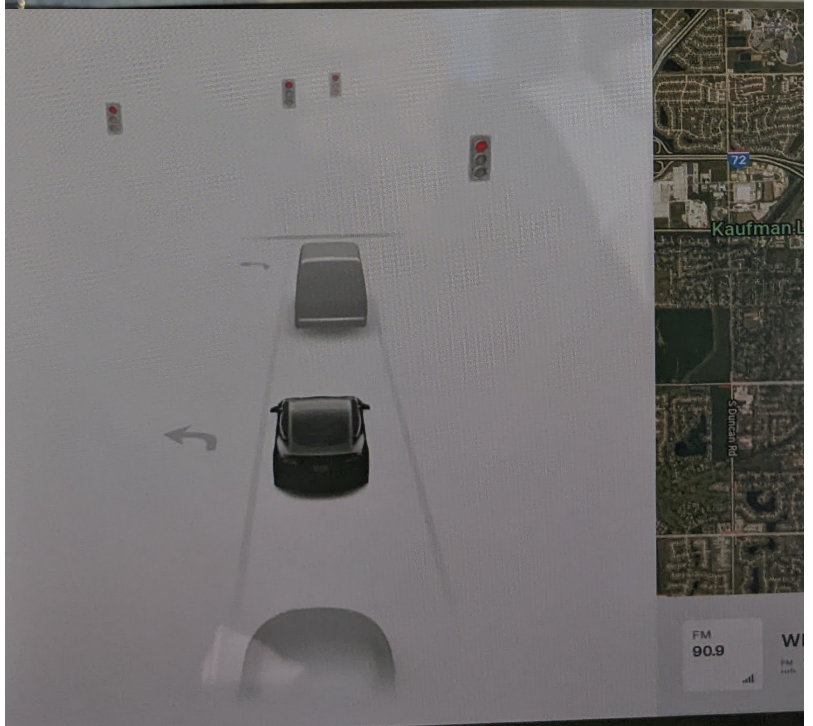
Perception Subsystem

- Converts signals from the environment into meaningful bits
- Bits can be used to make decisions and plan vehicle path



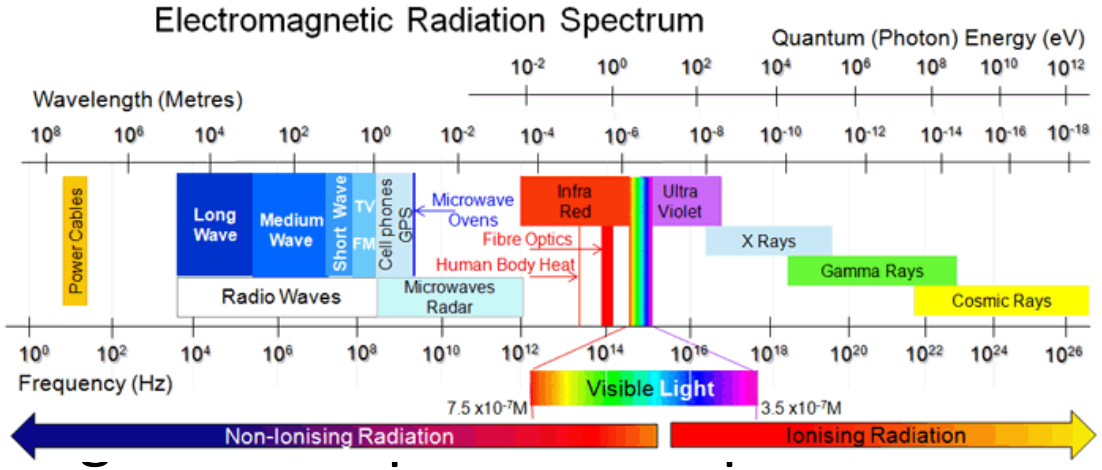
Perception

Programs for object detection, lane tracking, scene understanding, etc.

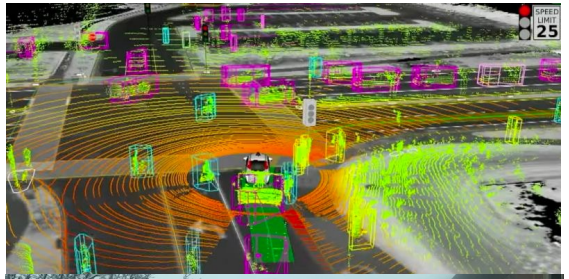


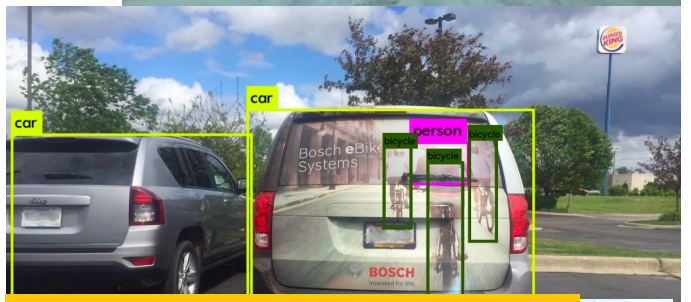
Perception: EM to Objects

Problem: Process electromagnetic radiation from the environment to construct a *model* of the world, so that the constructed model is close to the real world



- Chall ions of years of human evolution
  - Moravec's paradox
- Problem with definitions: How can we define what the meaning of a "car," "bicycle," "lane," etc. is?





Is this a bike?

Is this a car?

Computer Vision and Lane Detection

How can we code cars to perceive lines?

- Linear Filtering → remove “noise” from images
- Edge detection → find important “areas” of images

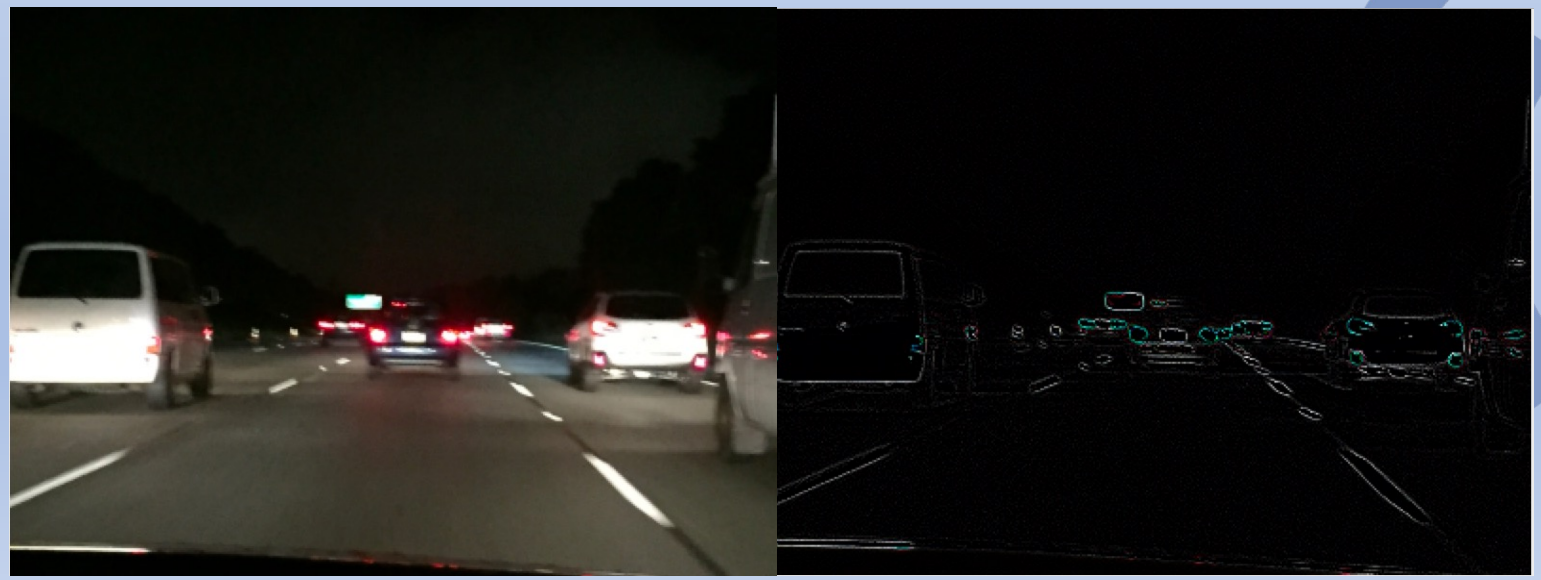
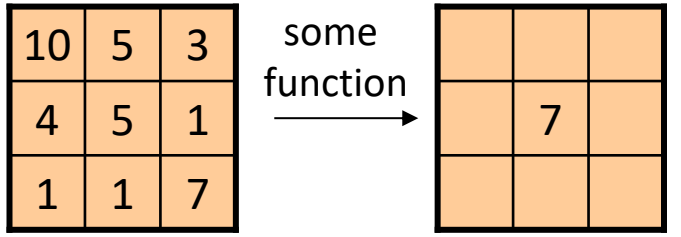


Image Representation

Jupyter notebook

What is filtering?

- Modify pixels in an image based on some function of a local neighborhood of pixels
- Simplest: linear filtering
  - Modify pixels in an image based on some function of a local neighborhood



Bright(img,k): for all i,j

img'[i][j] = k*img[i][j]

Shifting right by s Shift(img,s):

img'[k] = img[k-s]; img'[0]…img'[s-1] is undefined

Moving Average

- Let's replace each pixel with a *weighted* average of its neighborhood
- These weights are called the *filter kernel*
- What are the weights for the average of a 3x3 neighborhood?

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

"box filter"

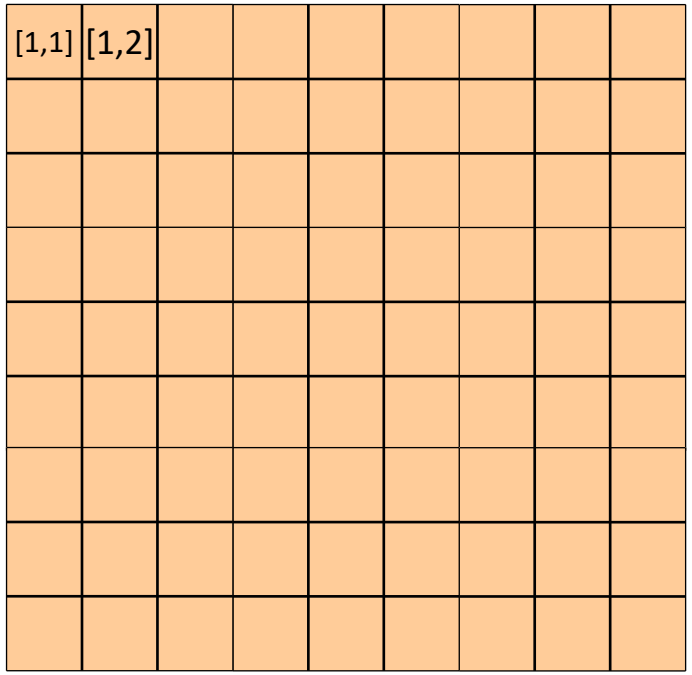
Convolution

- … describe more

convolution mask g[,]

1,1	1,2	1,3
2,1	2,2	2,3
3,1	3,2	3,3

image[i,j]

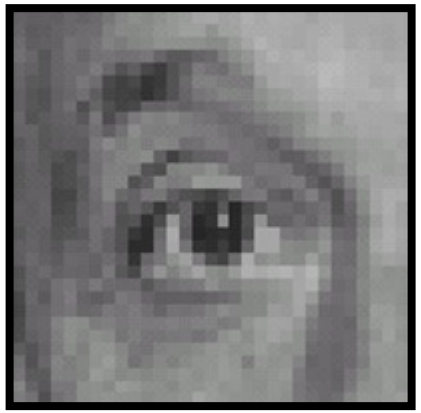


Output or convolved image

f = g * img

f[i,j] = g[1,1] img[i-1,j-1] + g[1,2] img[i-1,j] + g[1,3] img[i-1,j+1]
+ g[2,1] img[i,j-1] + g[2,2] img[i,j] + g[2,3] img[i,j+1]
+ g[3,1] img[i+1,j-1] + g[3,2] img[i+1,j] + g[3,3] img[i+1,j+1]

Practice with Linear Filters

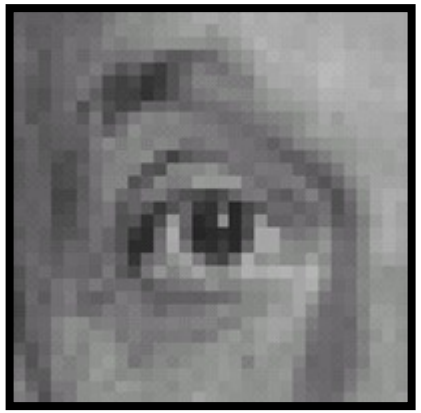


Original

0	0	0
0	1	0
0	0	0

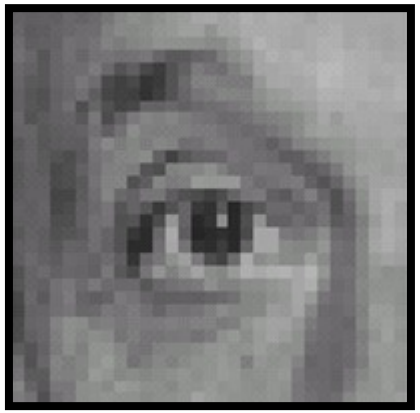
?

Practice with Linear Filters



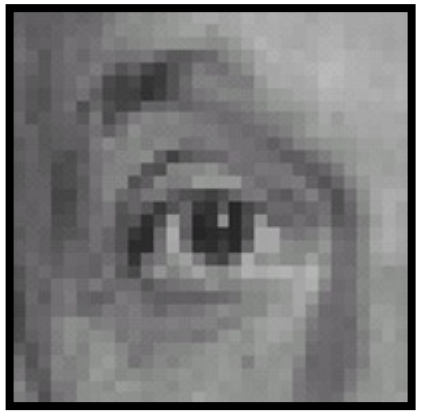
Original

0	0	0
0	1	0
0	0	0



Filtered
(no change)

Practice with Linear Filters

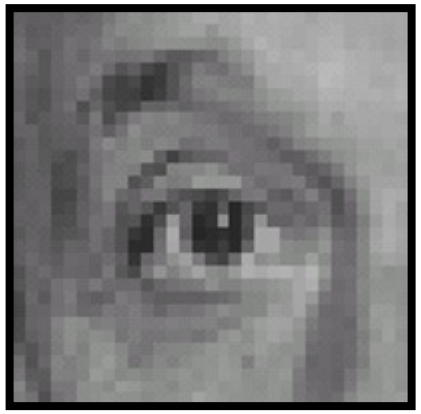


Original

0	0	0
0	0	1
0	0	0

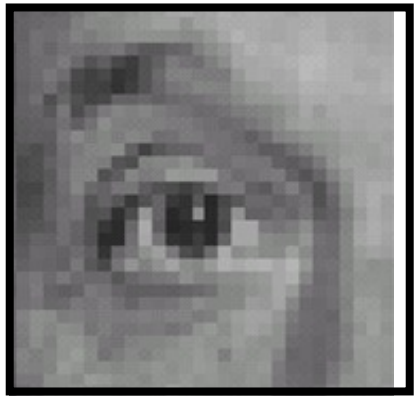
?

Practice with Linear Filters



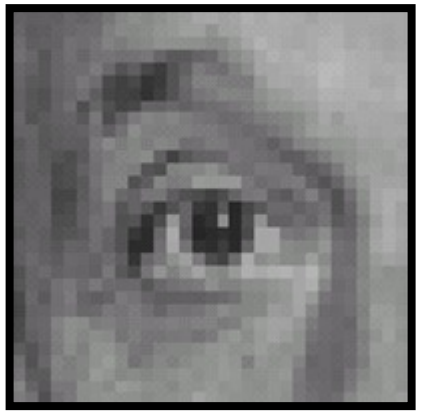
Original

0	0	0
0	0	1
0	0	0



Shifted *left*
by 1 Pixel

Practice with Linear Filters

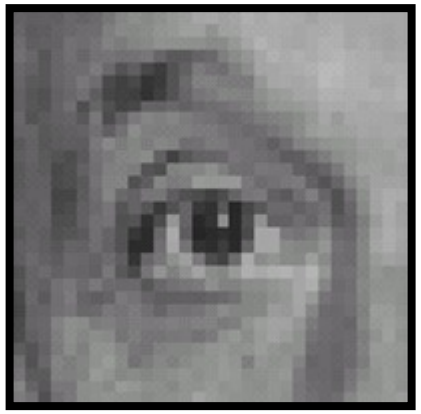


Original

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

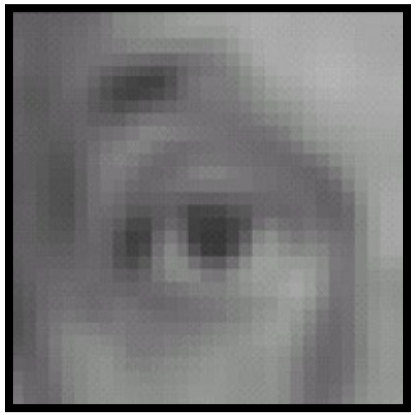
?

Practice with Linear Filters



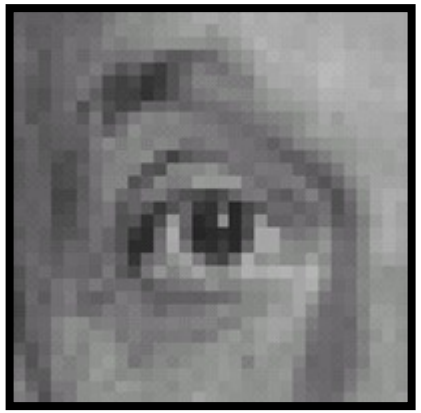
Original

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



Blur (with a box filter)

Practice with Linear Filters

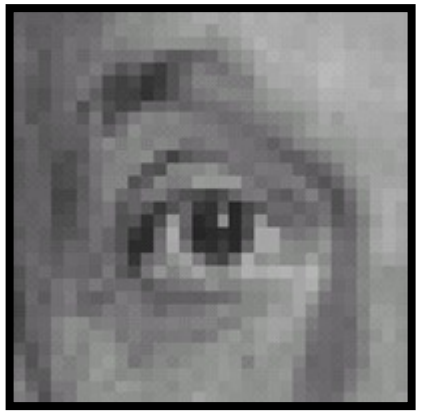


Original

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad ?$$

(Note that filter sums to 1)

Practice with Linear Filters



Original

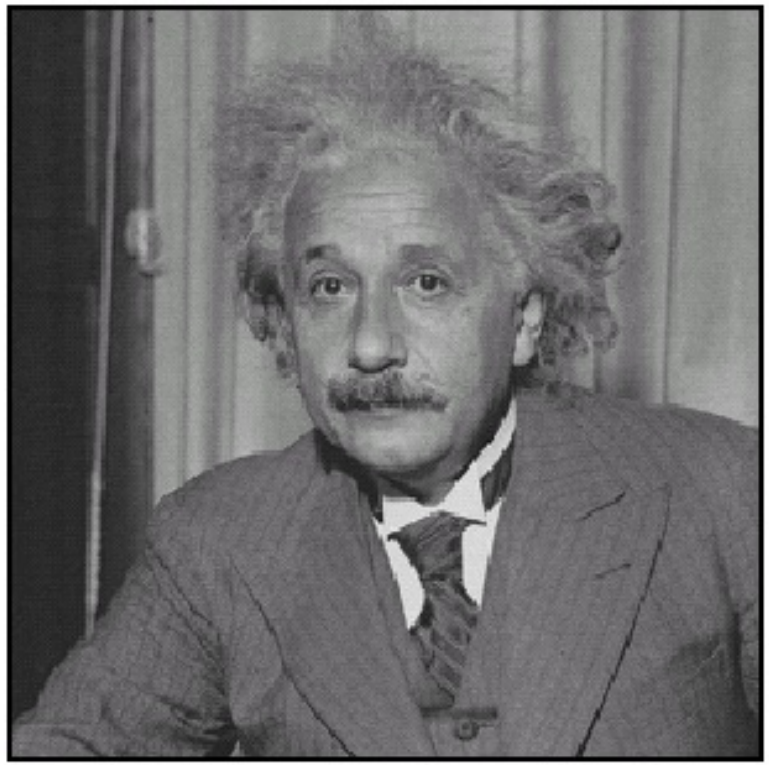
$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

(Note that filter sums to 1)

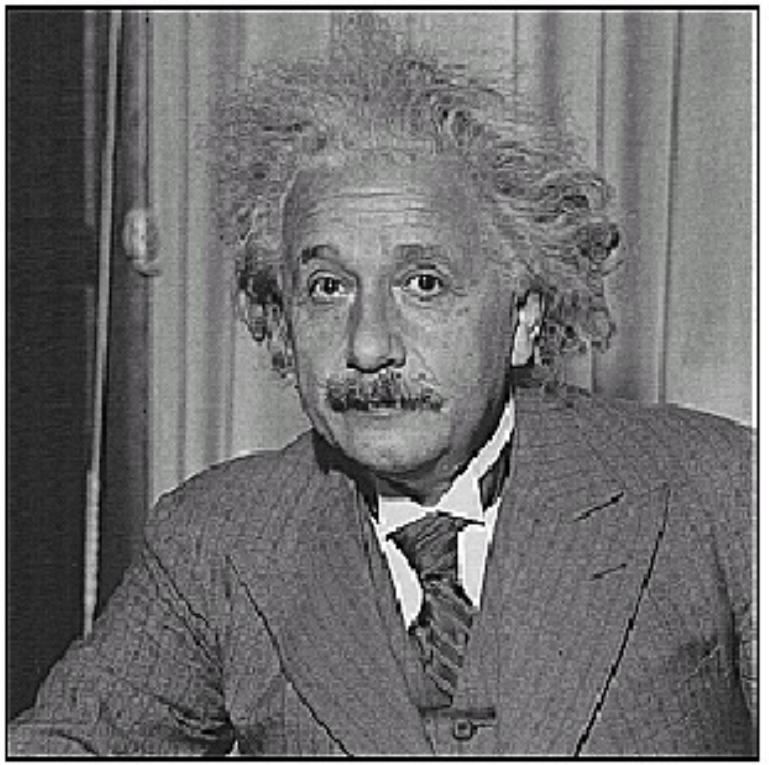
Sharpening filter

- Accentuates differences with local average

Sharpening



before



after

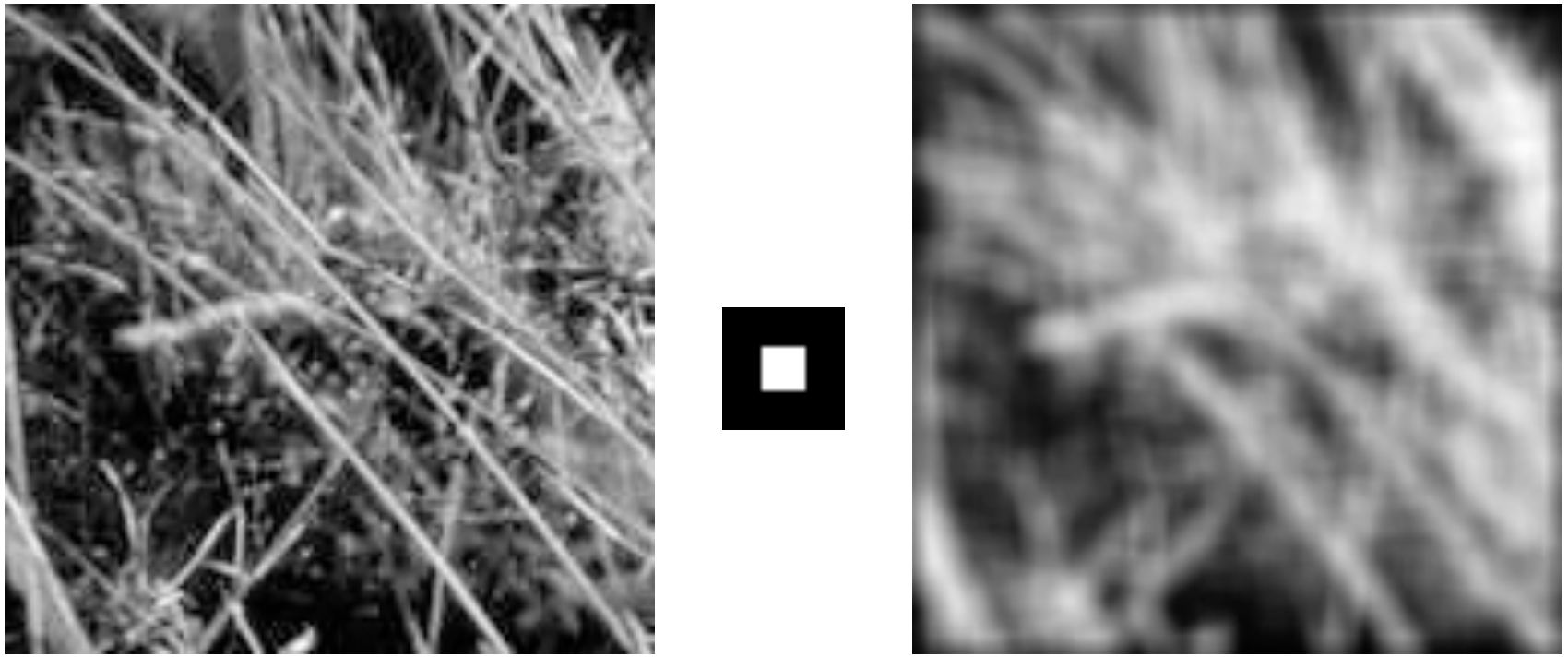
Sharpening

What does the blurring take away?

Let's add it back:

Smoothing with box filter revisited

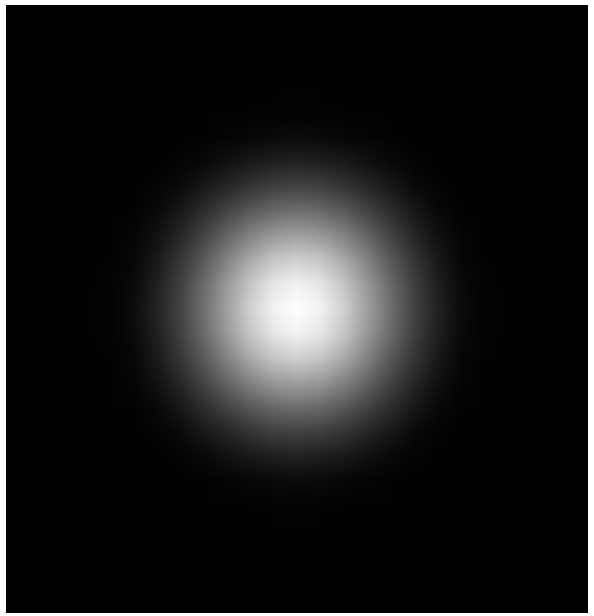
- What's wrong with the picture?



Source: D. Forsyth

Smoothing with box filter revisited

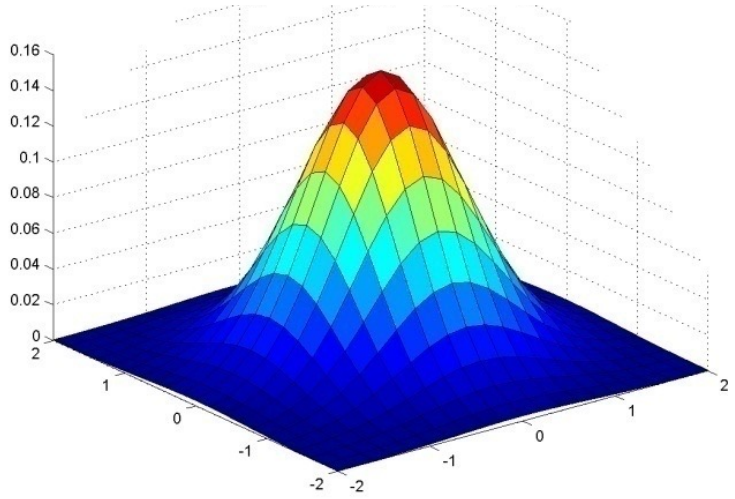
- Solution: To eliminate edge effects, weight contribution of neighborhood pixels according to their closeness to the center

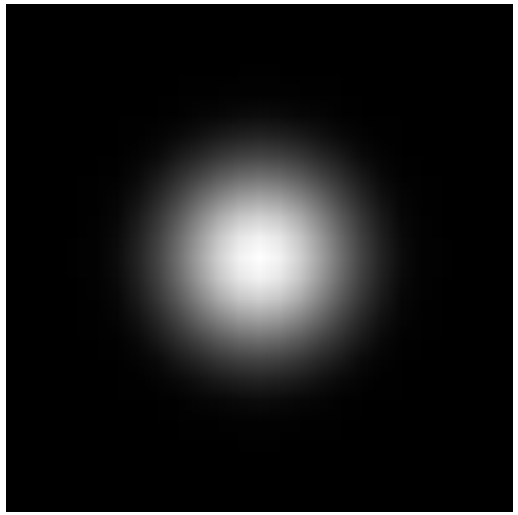


"fuzzy blob"

Gaussian Kernel

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





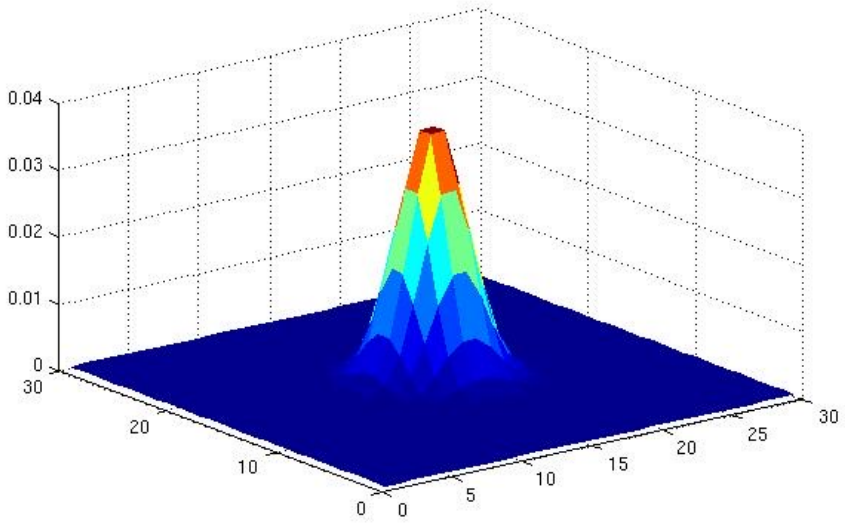
0.003	0.013	0.022	0.013	0.003
0.013	0.059	0.097	0.059	0.013
0.022	0.097	0.159	0.097	0.022
0.013	0.059	0.097	0.059	0.013
0.003	0.013	0.022	0.013	0.003

5 x 5, $\sigma = 1$

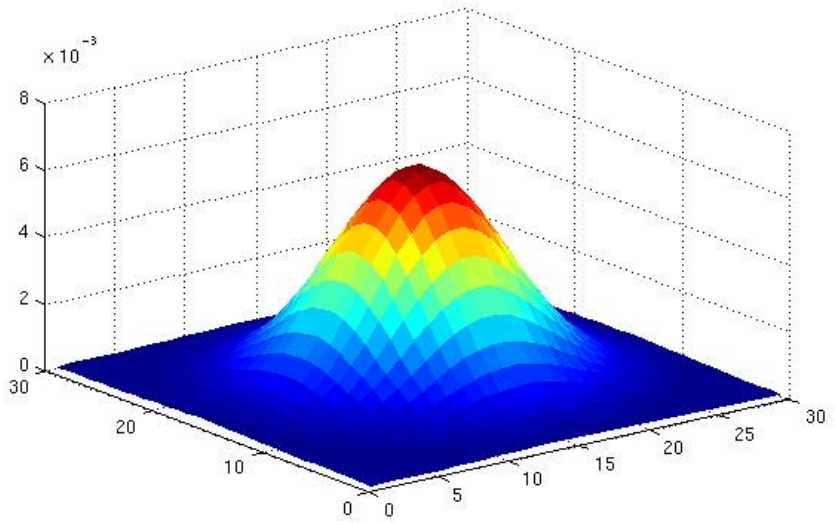
Gaussian Kernel

- Standard Deviation σ: determines extent of smoothing

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



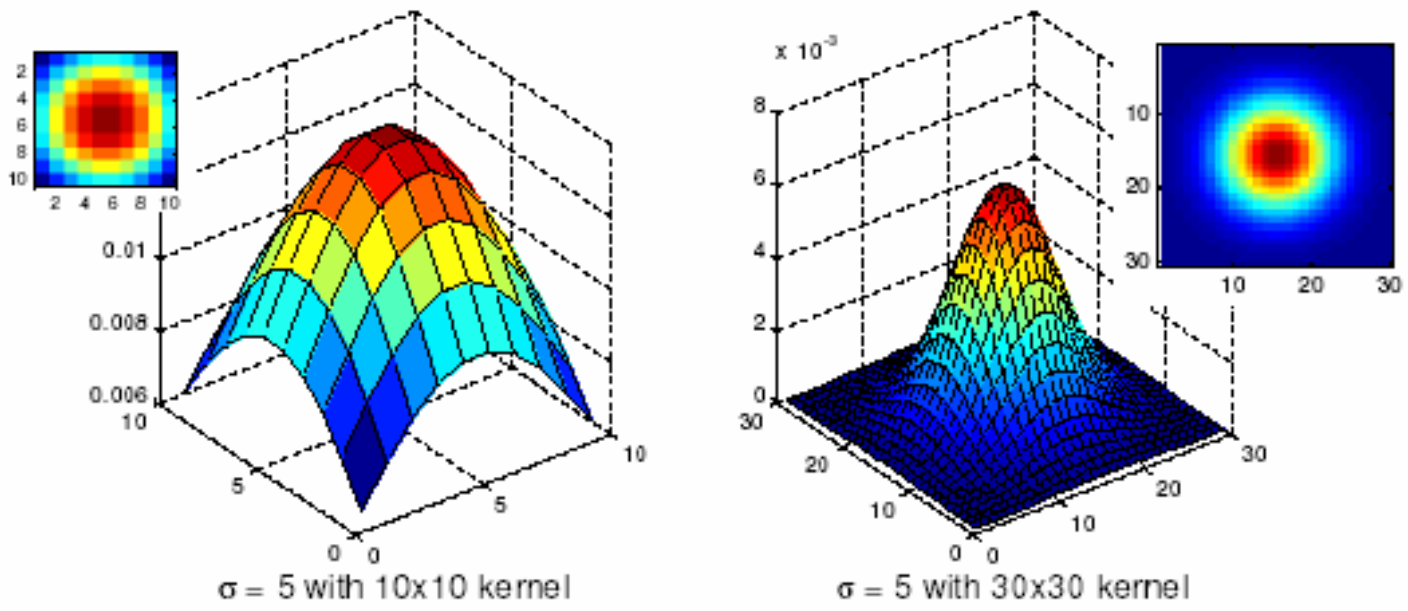
σ = 2 with 30 x 30 kernel

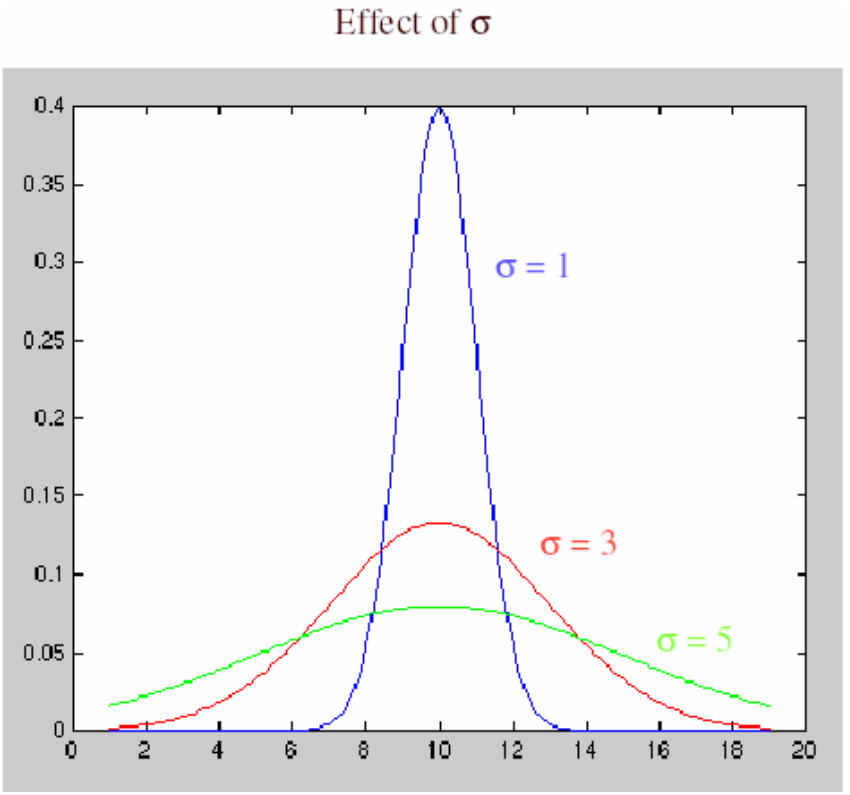


σ = 5 with 30 x 30 kernel

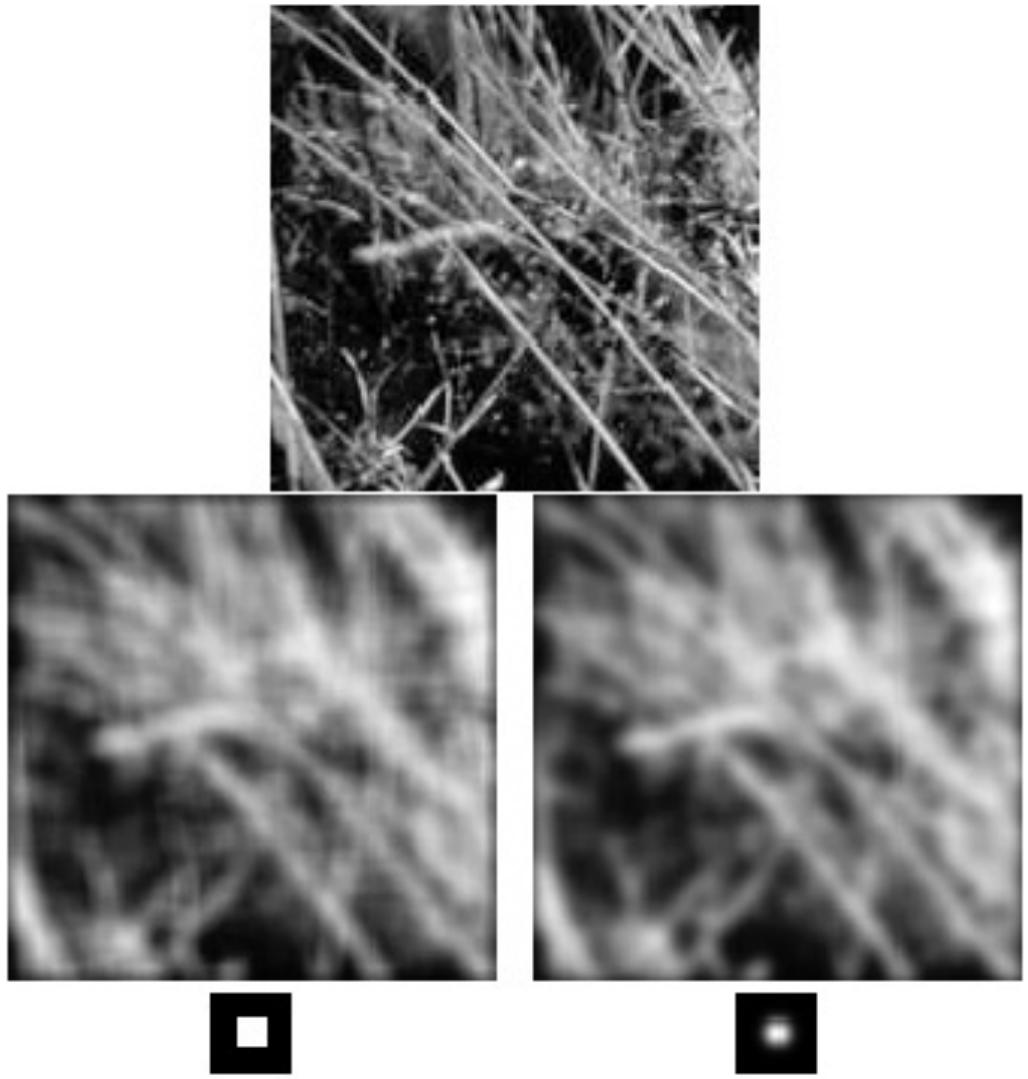
Choosing Gaussian Kernel Width

- The Gaussian function has infinite support, but discrete filters use finite kernels
- Rule of thumb: set filter half-width to about **3σ**





Gaussian vs. box filtering



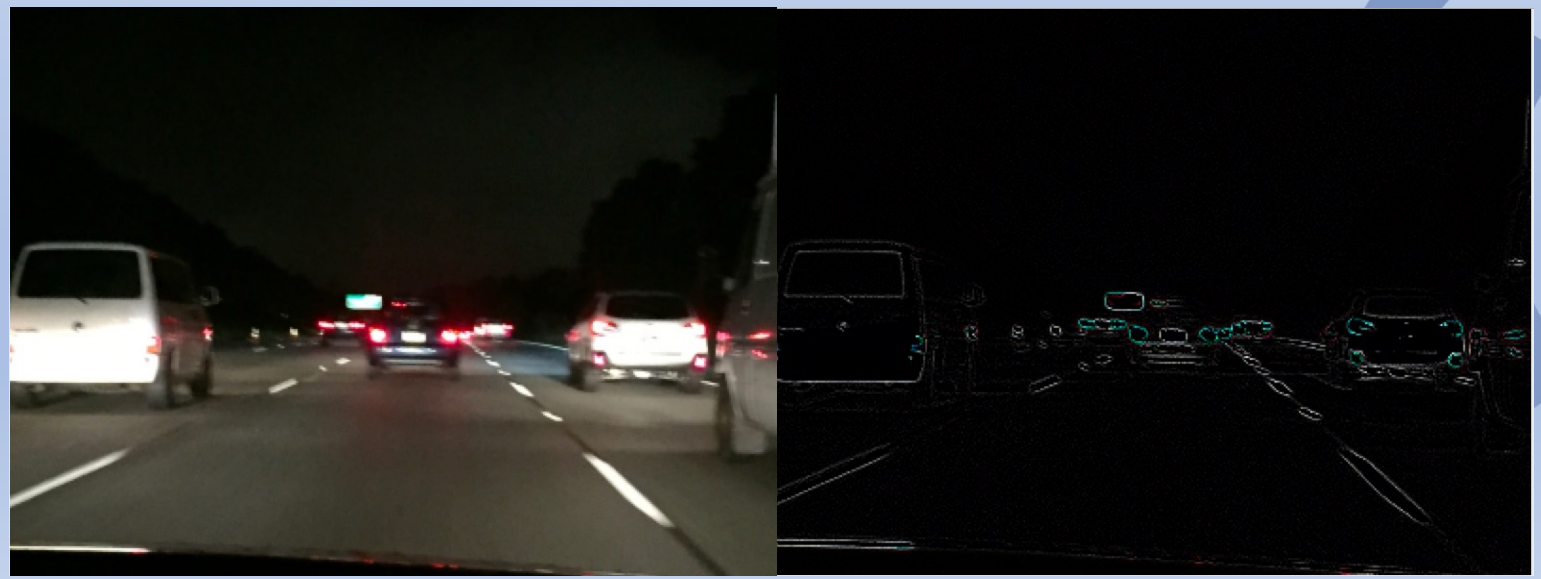
Gaussian filtering in OpenCV

Jupyter notebook

Computer Vision and Lane Detection

How can we code cars to perceive lines?

- Linear Filtering → remove “noise” from images
- Edge detection → find important “areas” of images

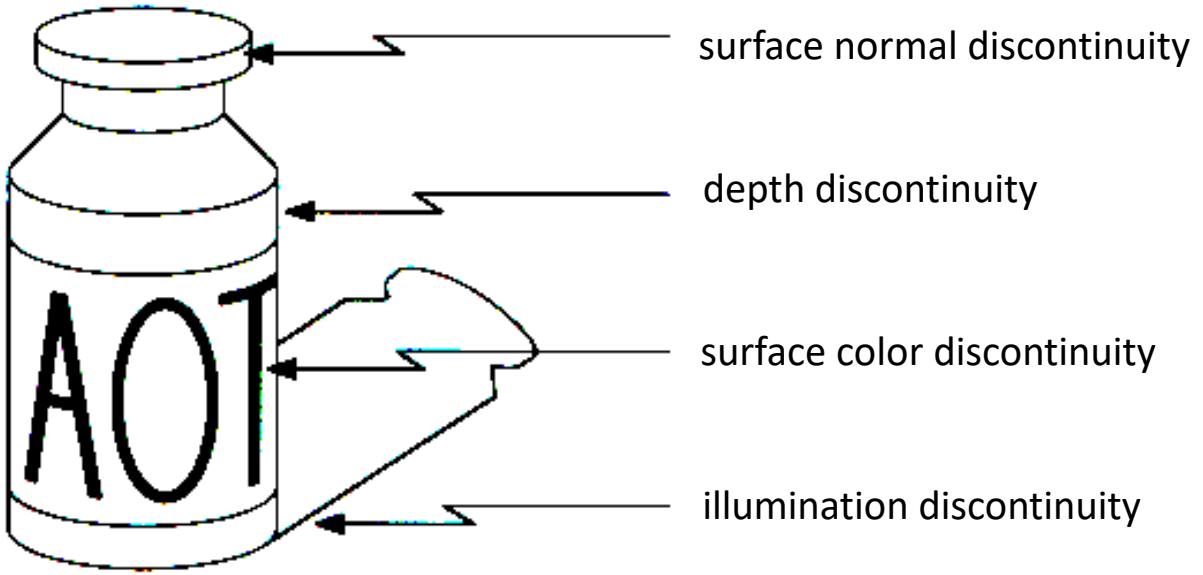


Edge detection

Winter in Kraków photographed by Marcin Ryczek

Edge detection

- **Goal:** Identify sudden changes (discontinuities) in an image
- Intuitively, edges carry most of the semantic and shape information from the image
  - E.g., Lanes, traffic signs, cars

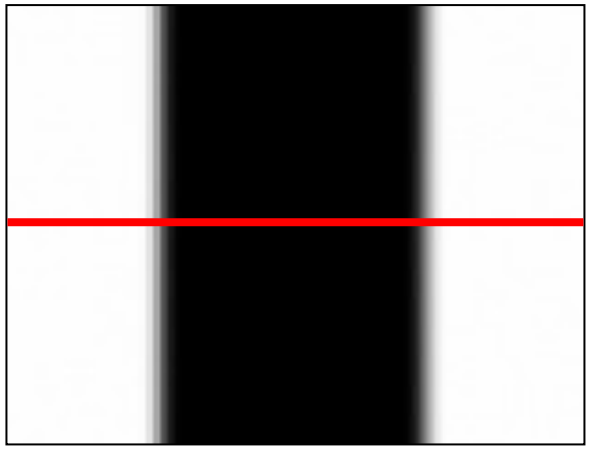


Sources: D. Lowe and S. Seitz

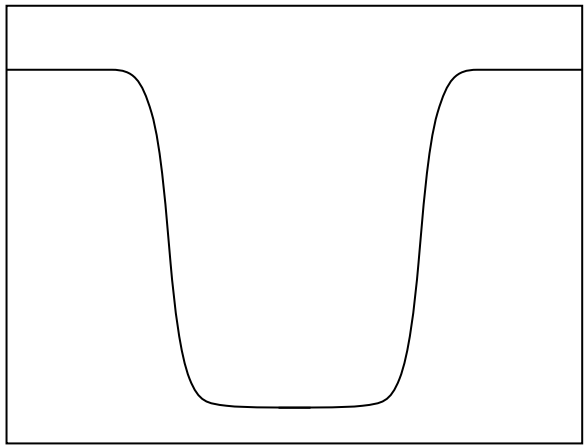
Edge detection

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

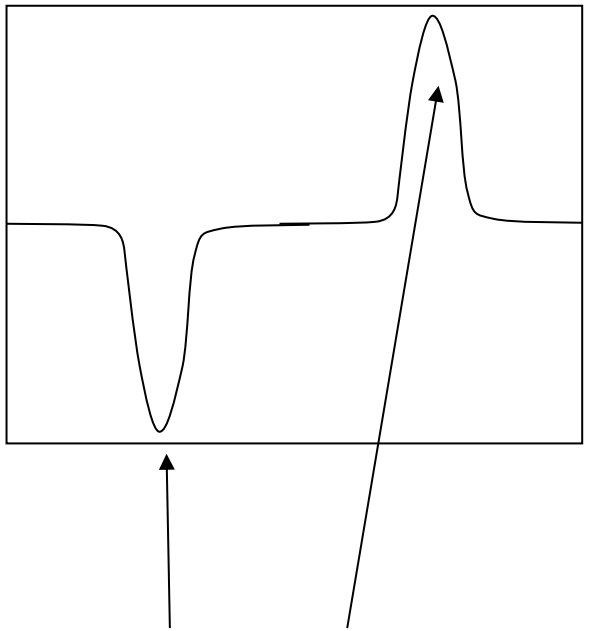
image



intensity function
(along horizontal scanline)



first derivative



edges correspond to
extrema (max or min) of derivative (slope of a curve)

Derivatives with convolution

- For a 2d function, f(x,y), the partial derivative w.r.t *x* is:

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

- For discrete data, we can approximate using finite differences:

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

- To implement the above as convolution, what would be the associated filter?

Source: K. Grauman

Convolution

Output or convolved image

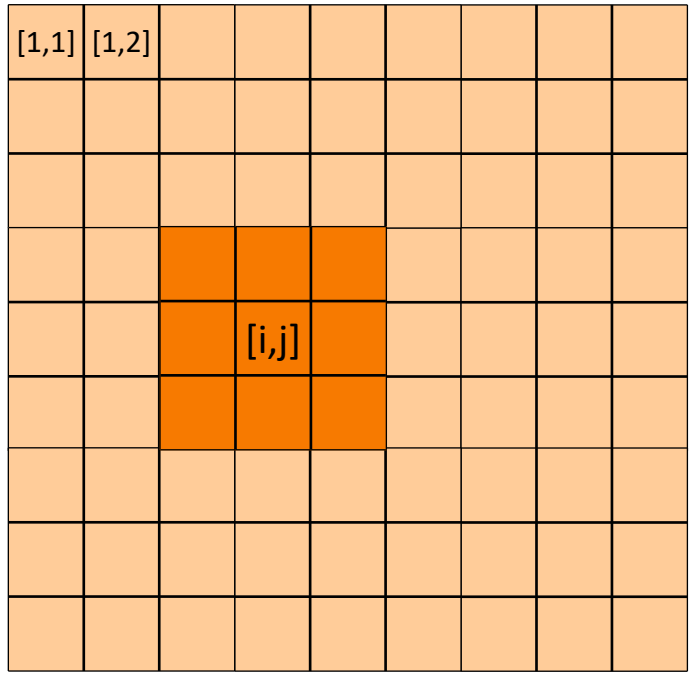
f = g * img

f[i,j] = -1.img[i,j-1] + 1. img[i,j]

convolution mask g[,]

-1	1

image[i,j]



Partial Derivatives of an image

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

-1	1

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

-1
1

1
-1

- Which shows change with respect to x?

Finite Difference filters

- Other approximations of derivative filters exist:

Prewitt: $M_x =$

-1	0	1
-1	0	1
-1	0	1

; $M_y =$

1	1	1
0	0	0
-1	-1	-1

Sobel: $M_x =$

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

; $M_y =$

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

Roberts: $M_x =$

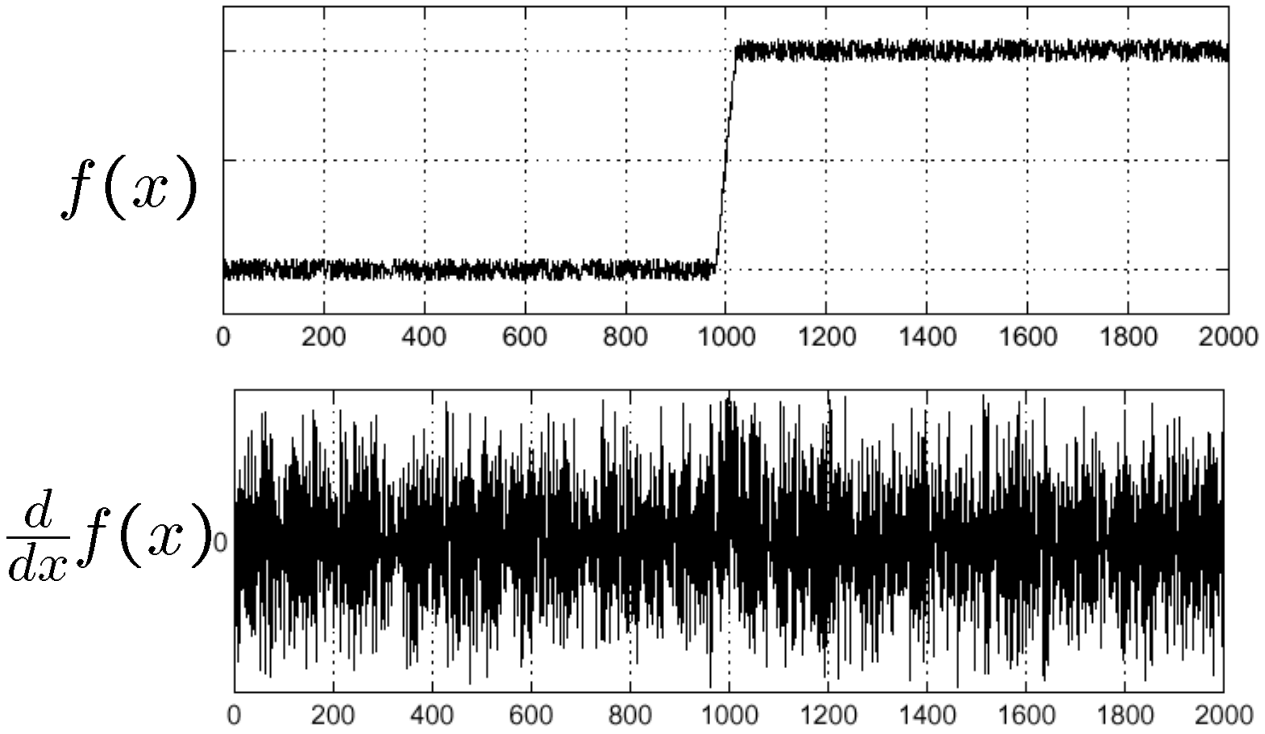
0	1
-1	0

; $M_y =$

1	0
0	-1

Effects of Noise

- Consider a single row or column of the image
- With noise, finding an edge if difficult



Solution: smooth first

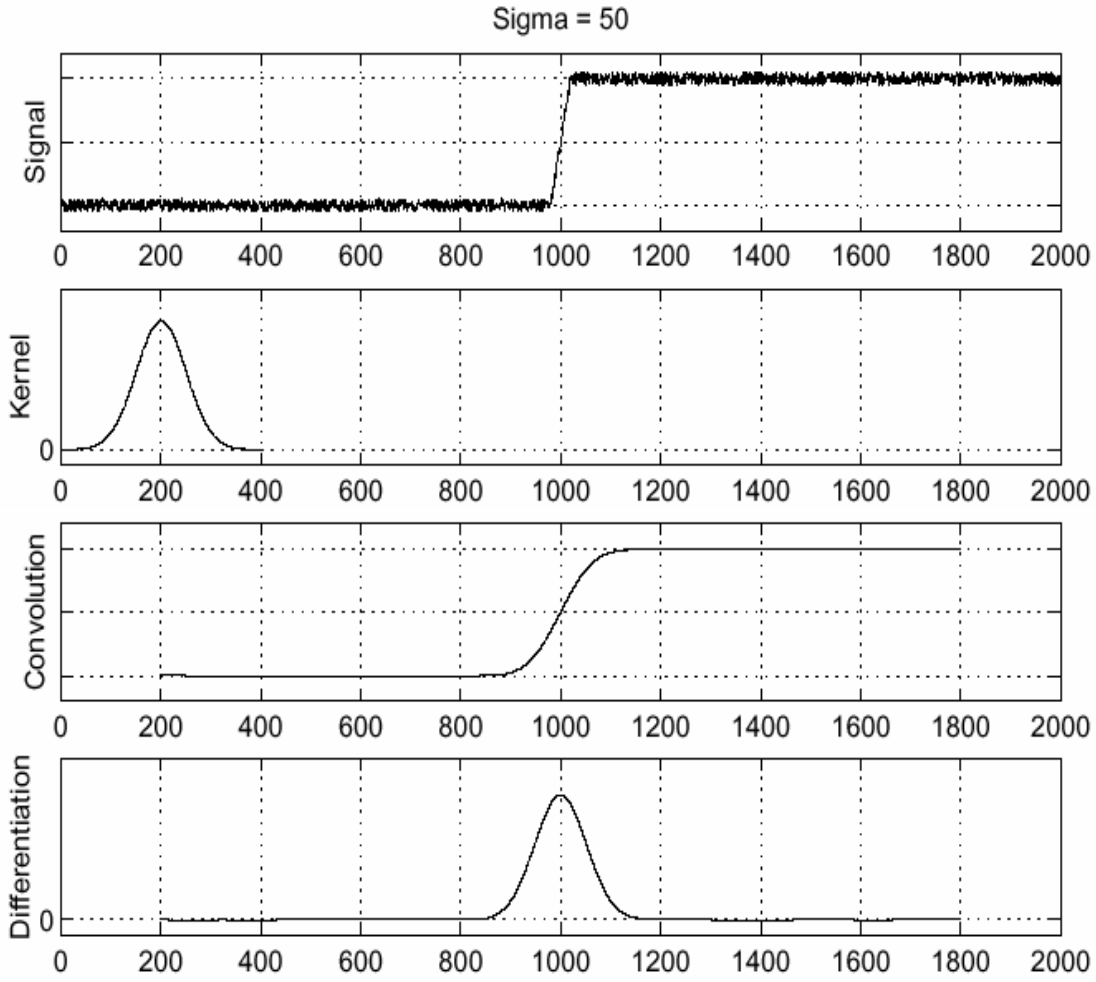
- Look for peaks in $\frac{d}{dx}(f * g)$

f

$Gaussian\ Filter \leftarrow g$

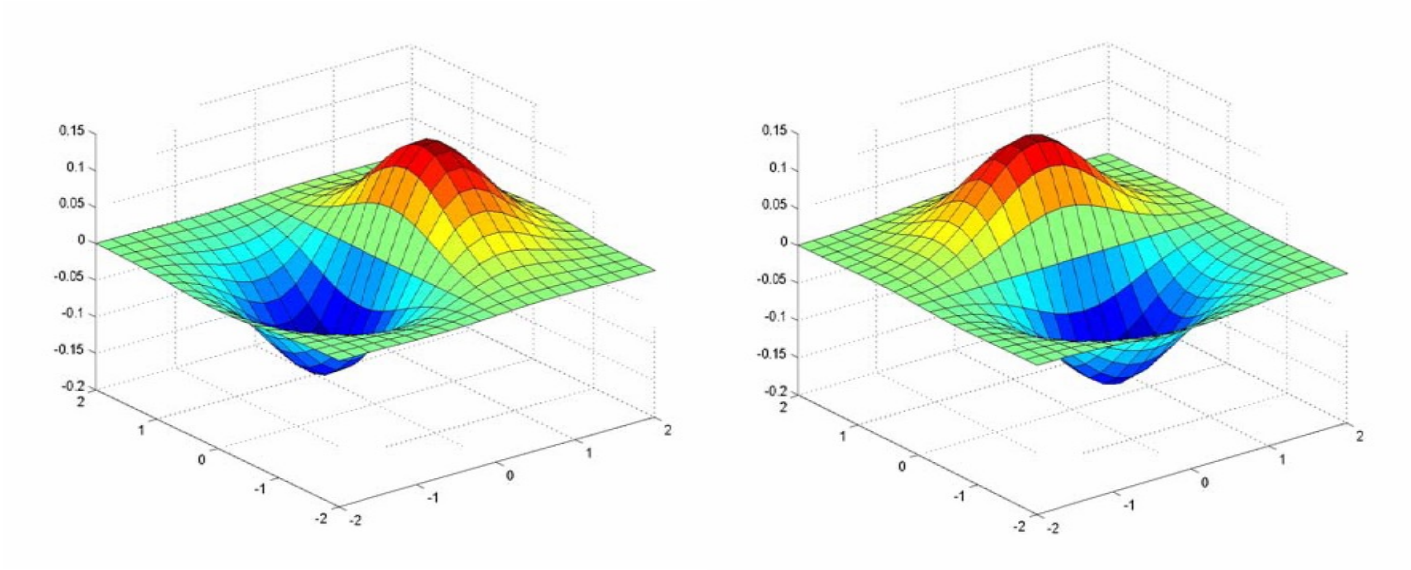
$After\ Gaussian\ Filter \leftarrow f * g$

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

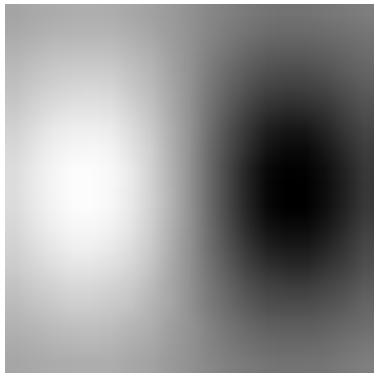


Derivative of Gaussian filters

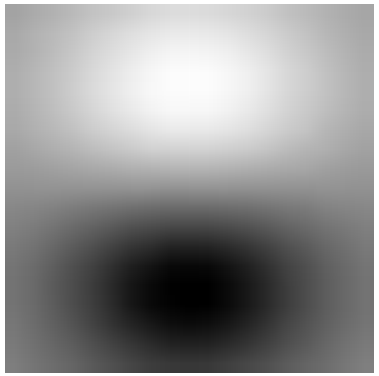
- Which one finds horizontal/vertical edges

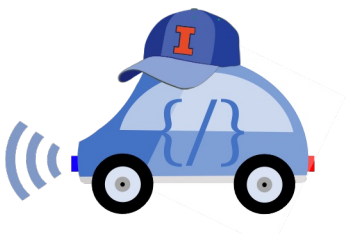


x-direction



y-direction

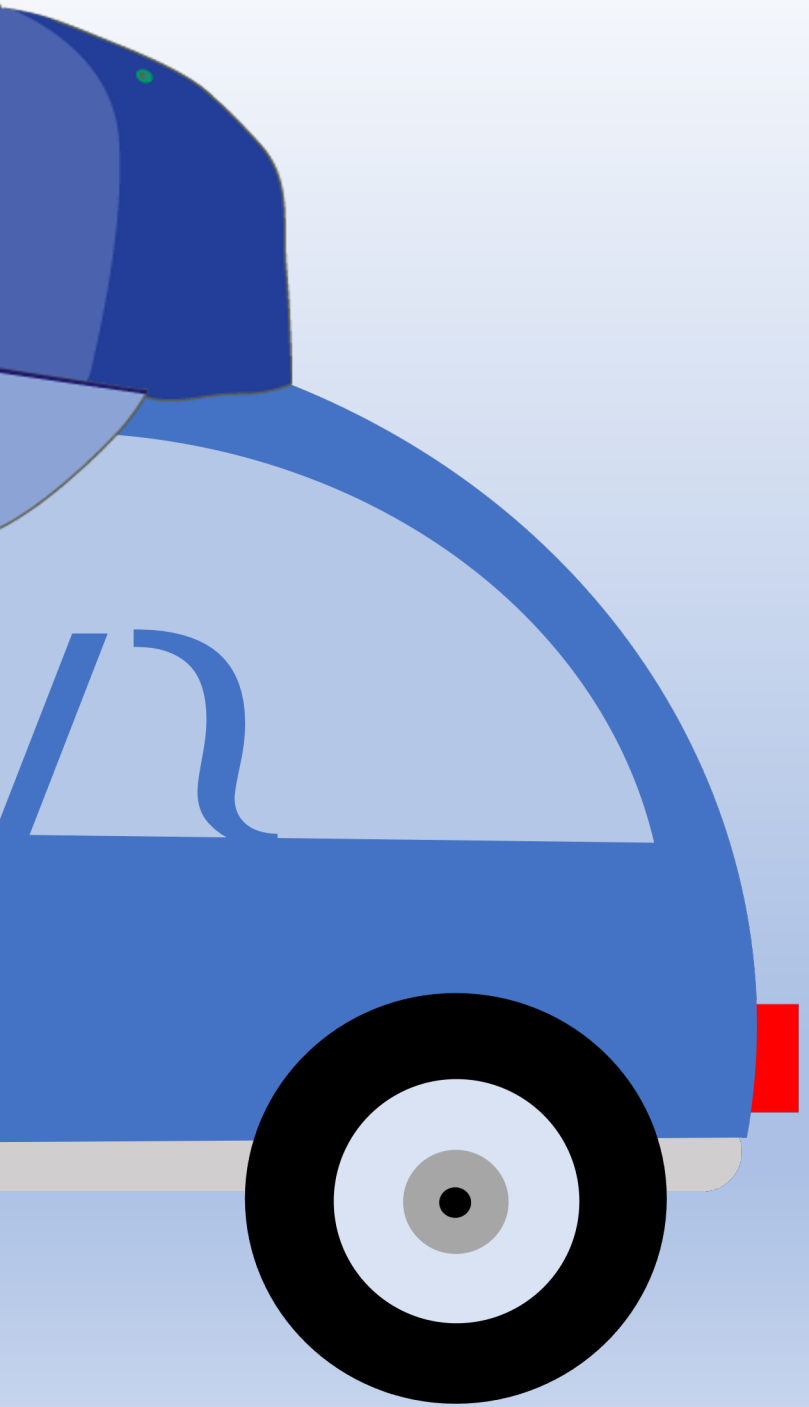




Sobel operator in OpenCV

Jupyter notebook

Summary (fix)



- ❑ Convert EM into information computers understand
- ❑ Convolution
- ❑ Filtering
- ❑ Sobel operator