CS195: Computer Vision

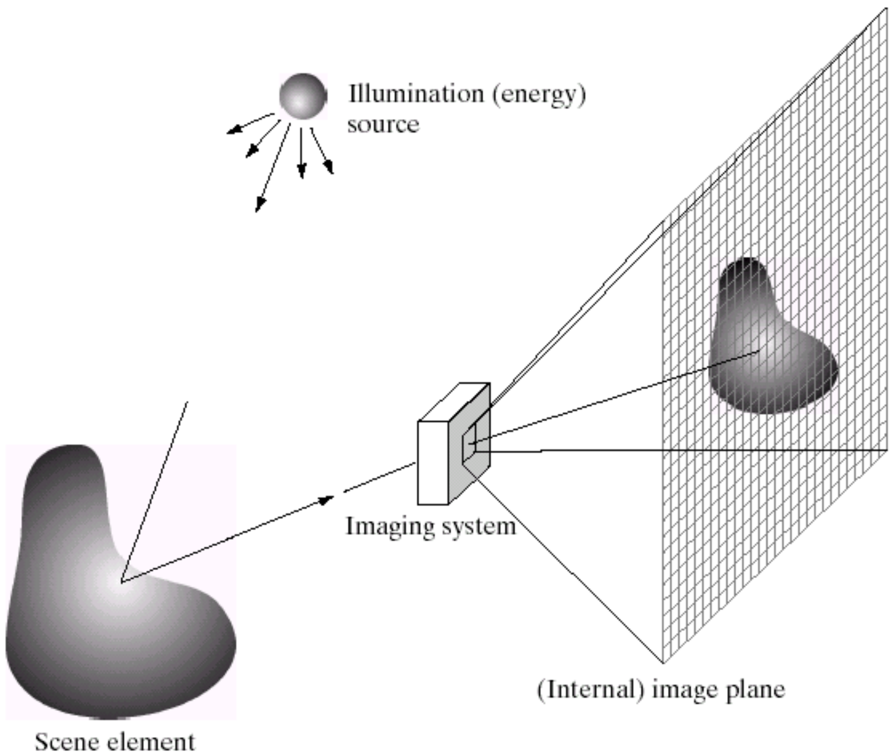
Image Filtering

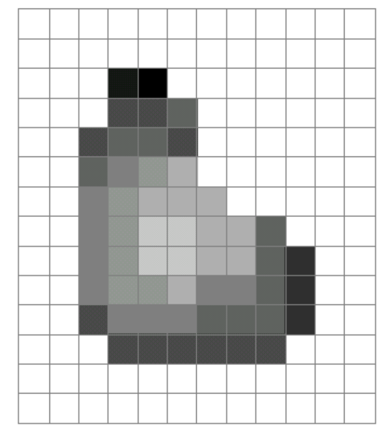
Cross-correlation

August 28, 2024

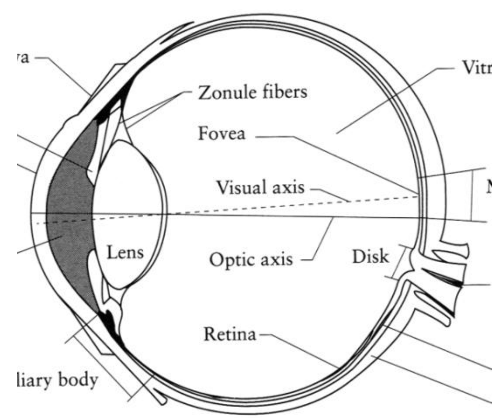
Md Alimoor Reza
Assistant Professor of Computer Science

What is an image?





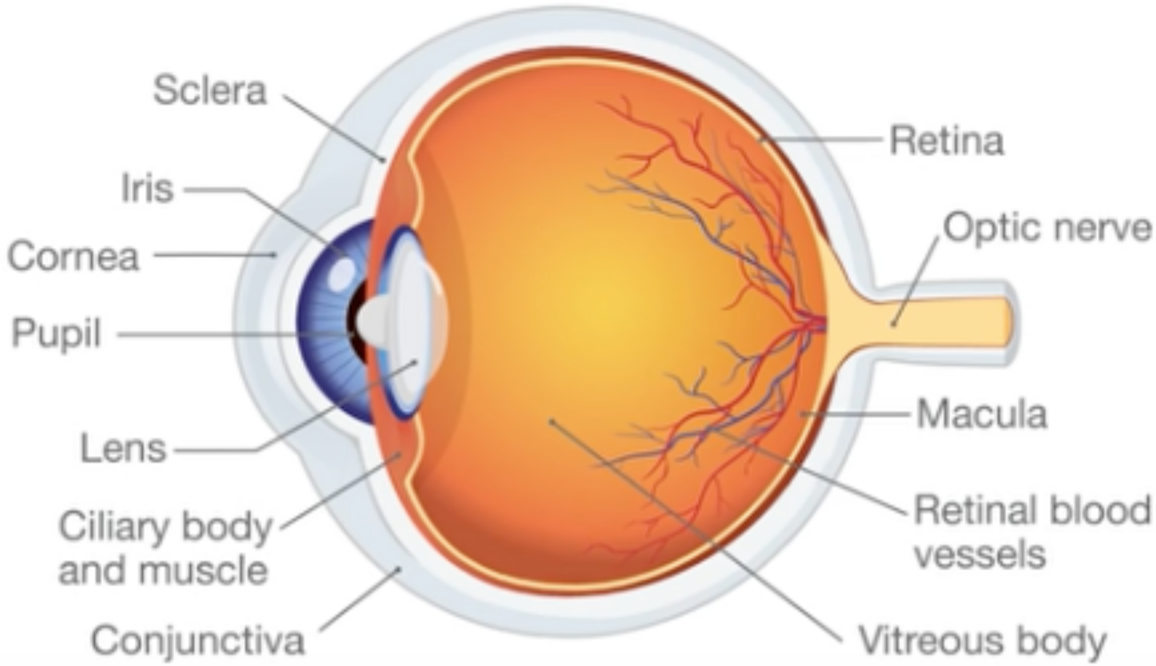
Digital Camera



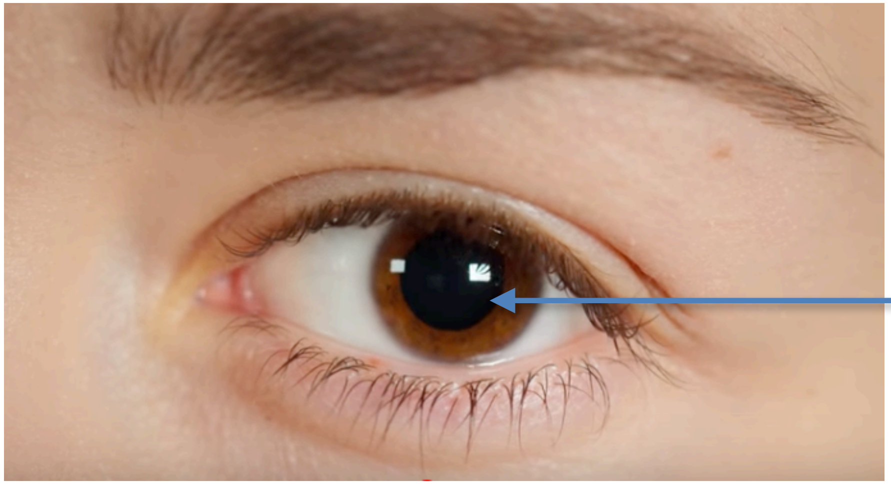
The Eye

Source: A. Efros

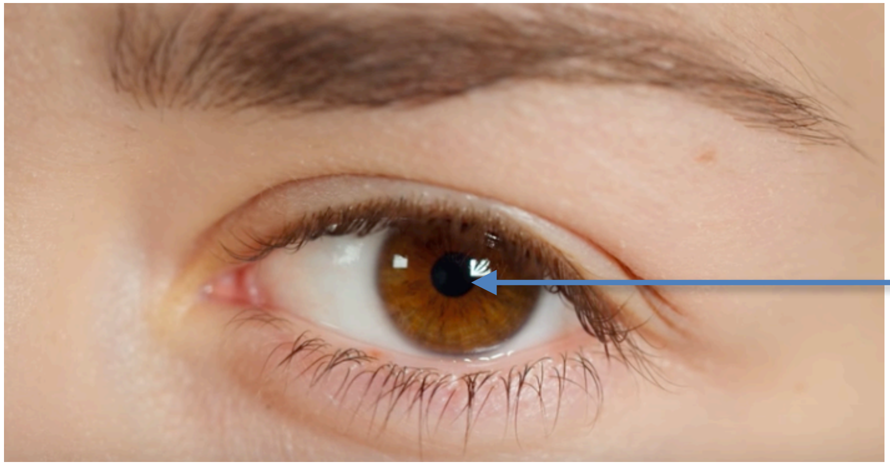
Human vision



Human vision

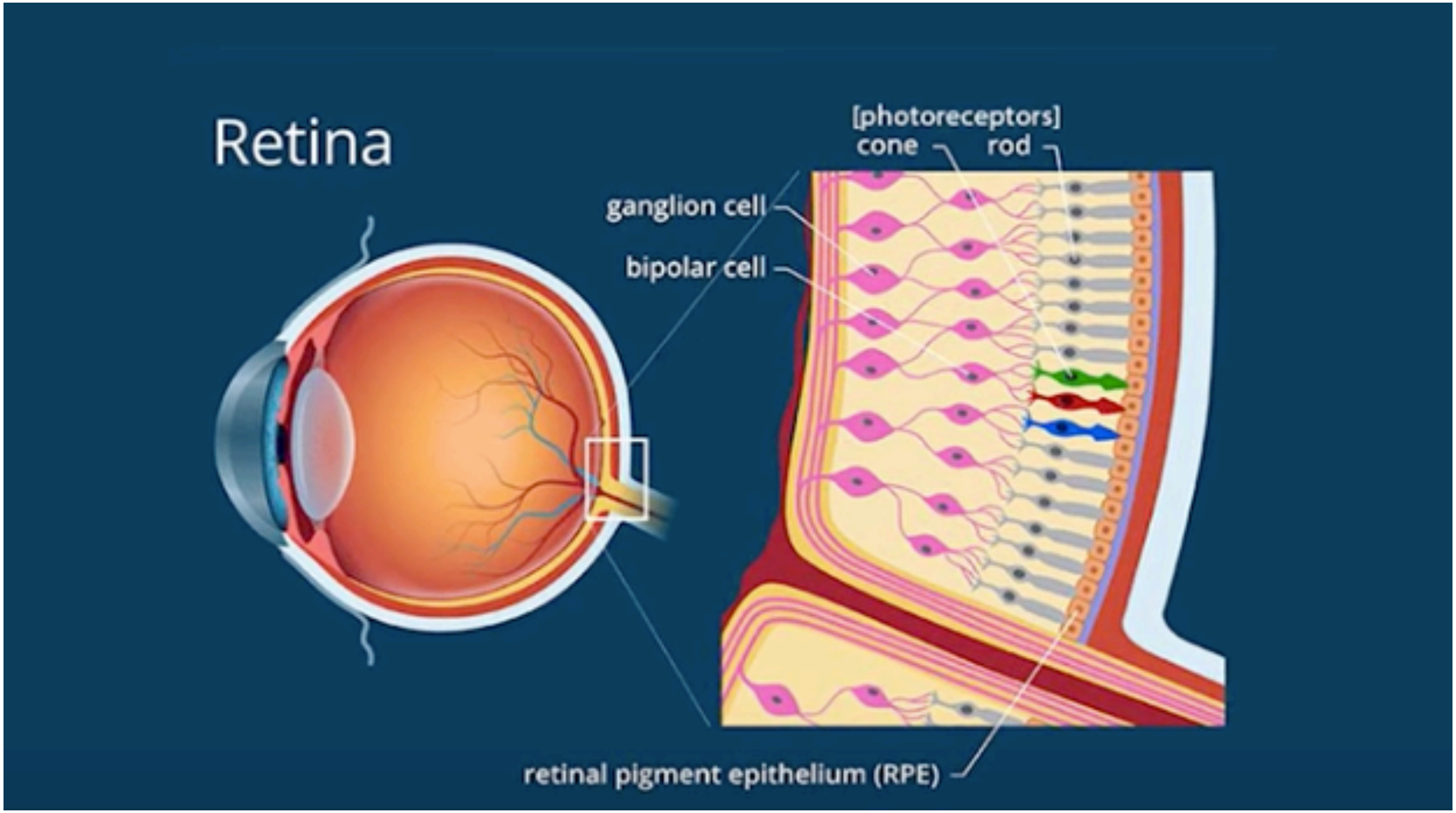


In the dark

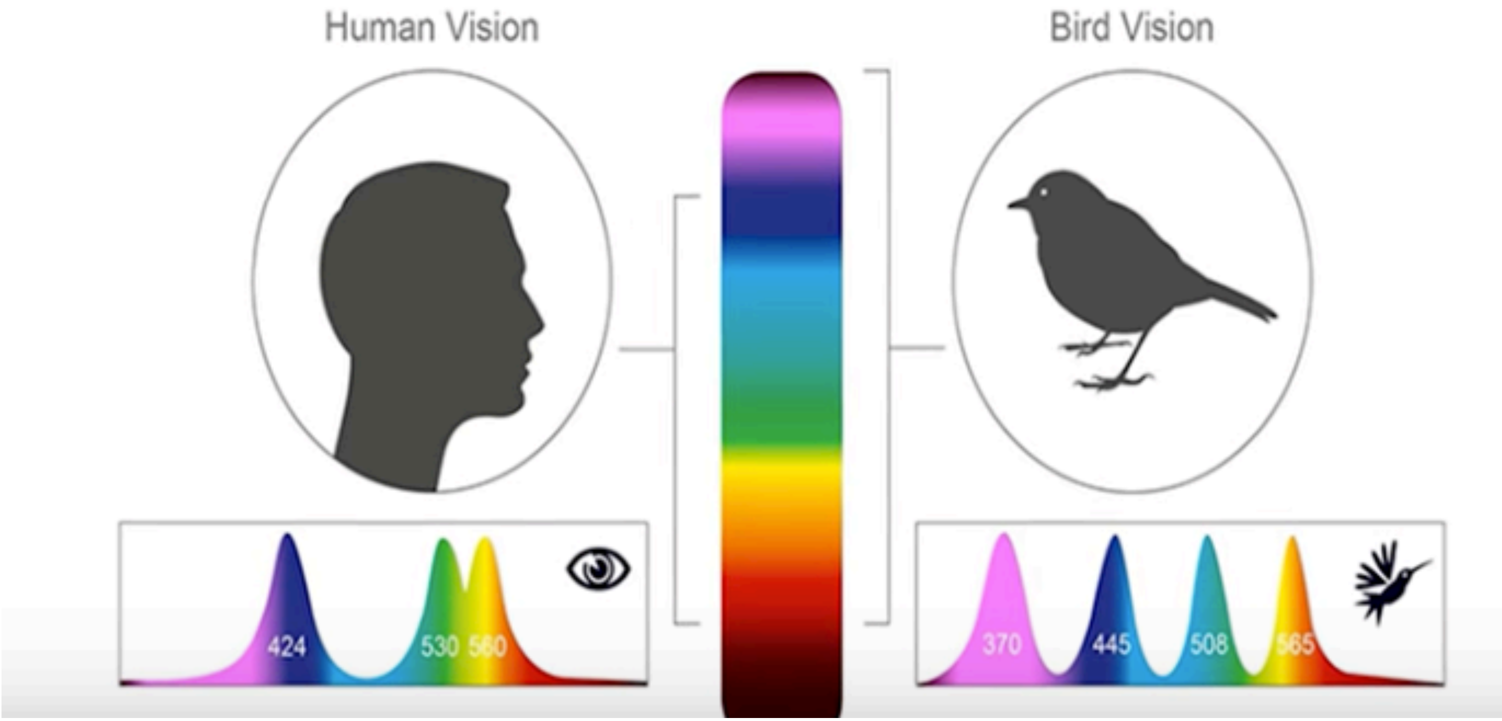


In bright light

Human vision

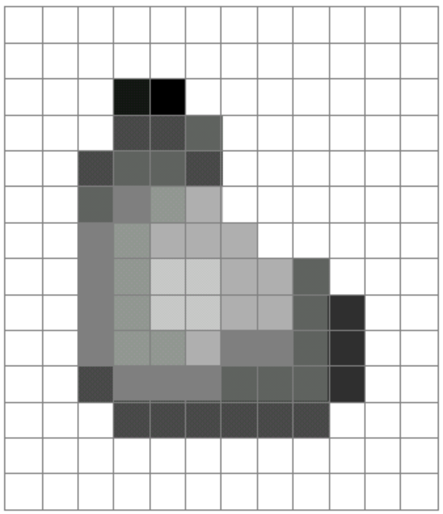


Human vision



What is an image?

- A grid of intensity values



=

255	255	255	255	255	255	255	255	255	255	255	255
255	255	255	255	255	255	255	255	255	255	255	255
255	255	255	20	0	255	255	255	255	255	255	255
255	255	255	75	75	75	255	255	255	255	255	255
255	255	75	95	95	75	255	255	255	255	255	255
255	255	96	127	145	175	255	255	255	255	255	255
255	255	127	145	175	175	175	255	255	255	255	255
255	255	127	145	200	200	175	175	95	255	255	255
255	255	127	145	200	200	175	175	95	47	255	255
255	255	127	145	145	175	127	127	95	47	255	255
255	255	74	127	127	127	95	95	95	47	255	255
255	255	255	74	74	74	74	74	74	255	255	255
255	255	255	255	255	255	255	255	255	255	255	255
255	255	255	255	255	255	255	255	255	255	255	255

(common to use one byte per value: 0 = black, 255 = white)

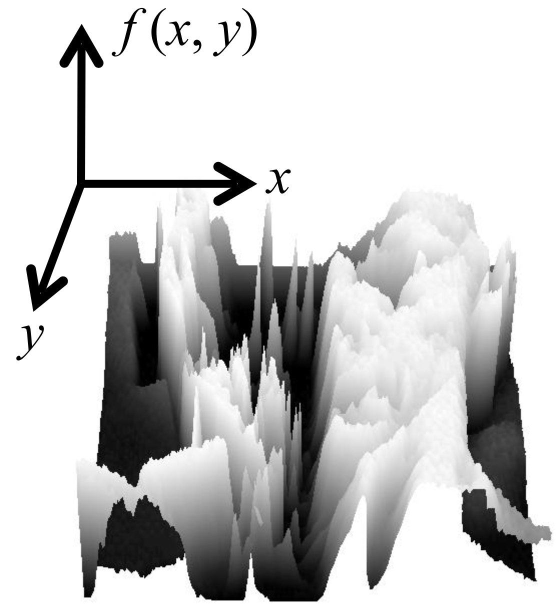
Red

Green

Blue

What is a (grayscale) image?

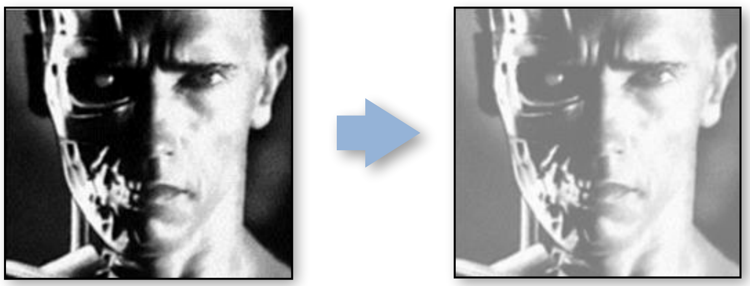
- A **2D function** f:$\mathrm{R}^2 \rightarrow \mathrm{R}$
  - $f(x,y)$ gives the **grayscale intensity** at position (x,y)



  - A **digital** image is a discrete (**domain is sampled**, **range is quantized**) version of this function

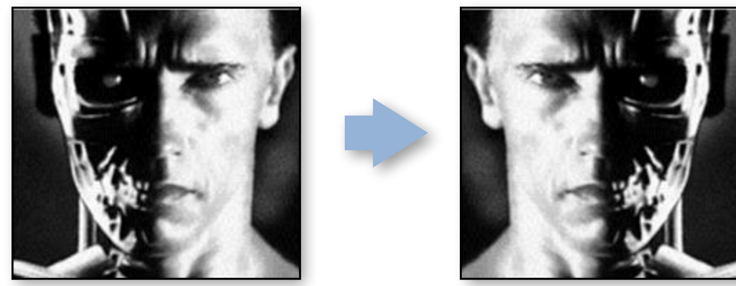
Image transformations

- As with any function, we can apply operators to an image



$$g(x,y) = f(x,y) + 20$$

Becomes brighter than the original



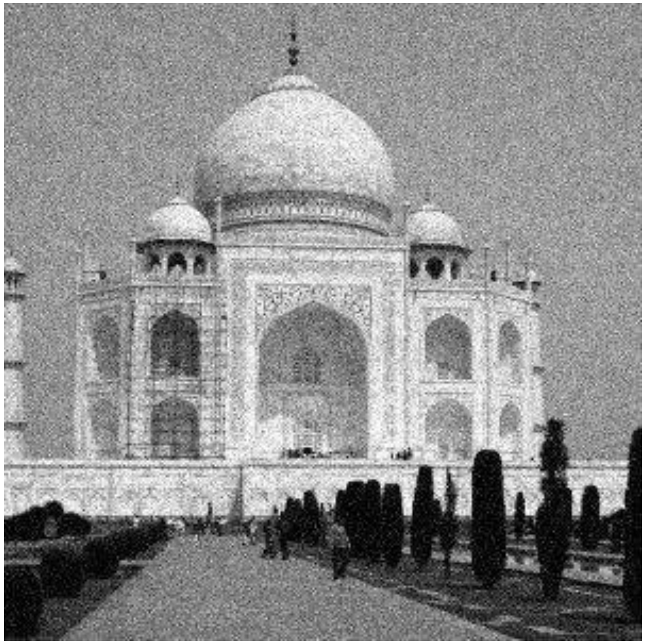
$$g(x,y) = f(-x,y)$$

Create mirror reflection of the original

Drake UNIVERSITY

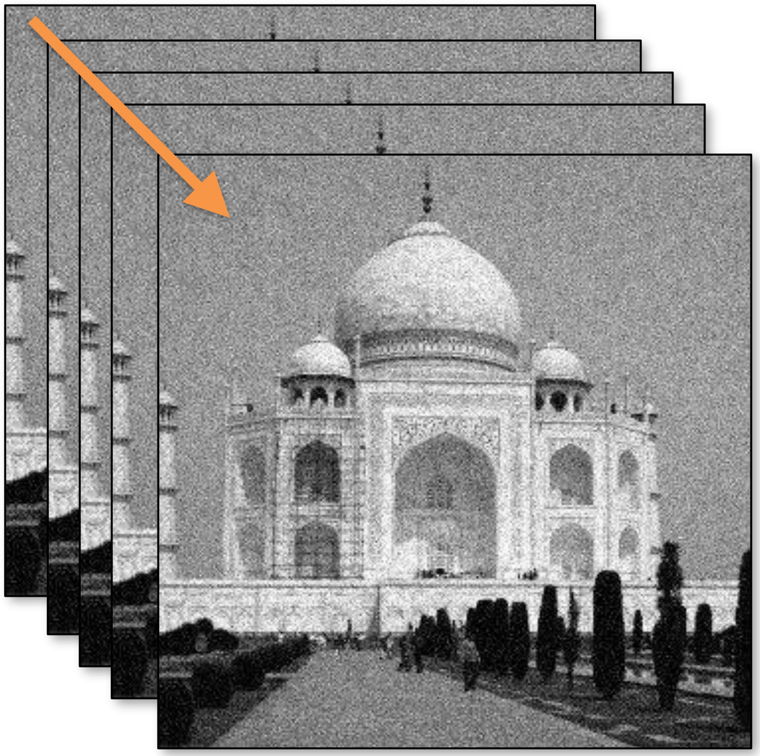
Image noise

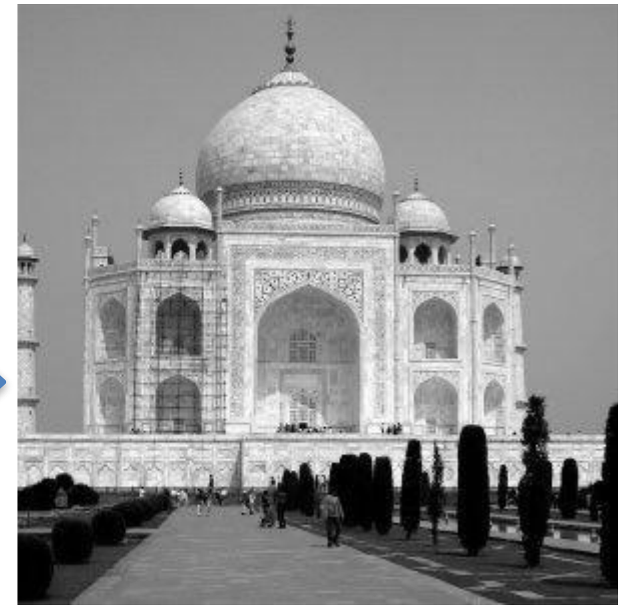
- Imaging systems (sensor, lens, compression algorithm, etc.) add unwanted noise



Question: Noise reduction

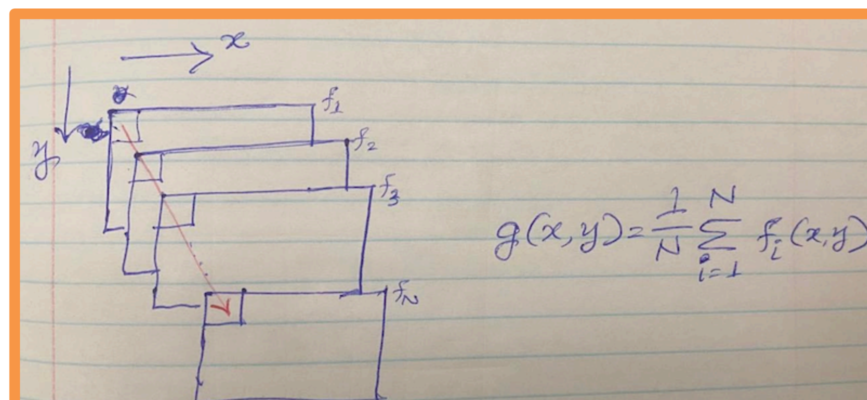
- Given a camera and a still scene, how can you reduce noise?





Take lots of images and average them!

What's the next best thing?



Source: S. Seitz

Question: What assumptions need to be correct for proposed averaging method?

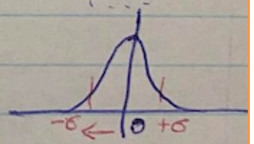
- Given a camera and a still scene, we assume a particular noise model
  - Additive Gaussian Noise

$$f_{noise}(x,y) = f_{actual}(x,y) + \epsilon$$

where, $\epsilon = \mathcal{N}(0, \sigma^2)$

zero mean

sigma standard deviation.



$$\mathcal{N}(0,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Image filtering

- Modify the pixels in an image based on some function of a local neighborhood of each pixel

10	5	3
4	5	1
1	1	7

Local image data

Some function

	7	

Modified image data

Source: L. Zhang

Image filtering

- Modify the pixels in an image based on some function of a local neighborhood of each pixel

10	5	3
4	5	1
1	1	7

Local image data

Some function

	7	

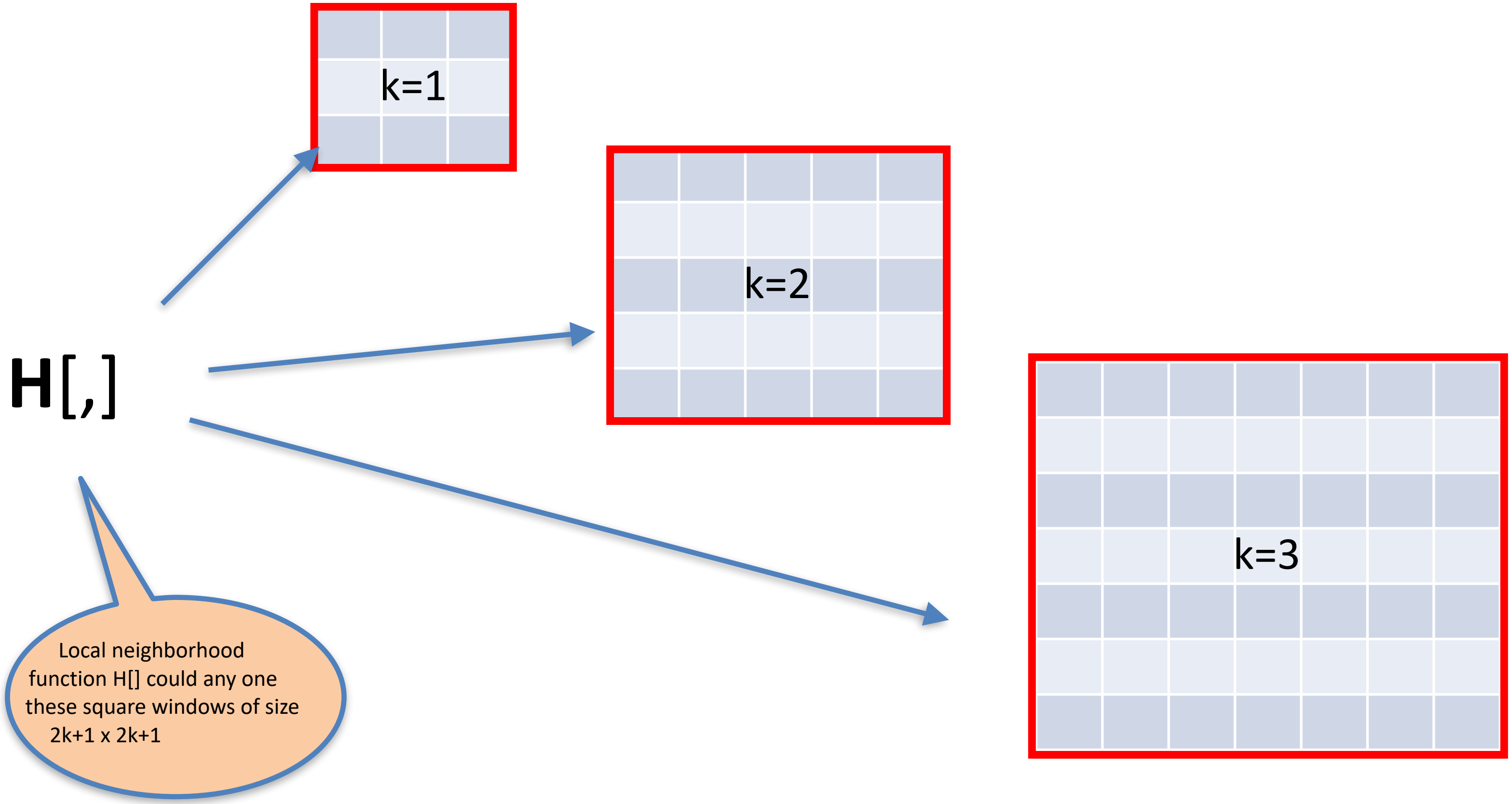
Modified image data

$F[\cdot,\cdot]$ $\quad$ $\mathbf{H}[,]$ $\quad$ $G[\cdot,\cdot]$

Source: L. Zhang

Image filtering

- Modify the pixels in an image based on some function of a local neighborhood of each pixel

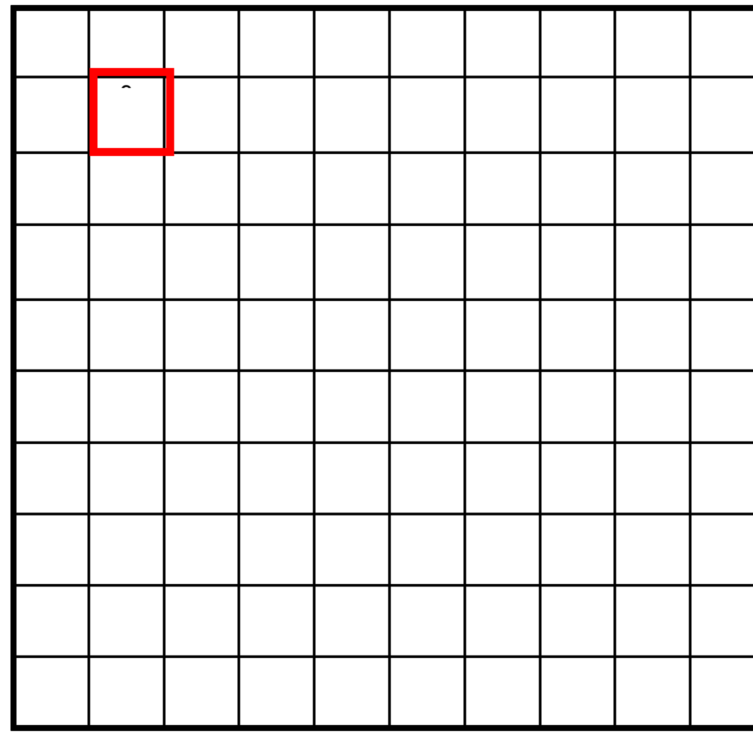


Mean filtering

$F[.,.]$

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

$G[.,.]$



$$G(i,j) = \frac{1}{(2k+1)^2} \sum_{u=-k}^{k} \sum_{v=-k}^{k} F(u+i, v+j)$$

Credit: S. Seitz

Mean filtering

$F[.,.]$

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

$G[.,.]$

	0	10							

$$G(i,j) = \frac{1}{(2k+1)^2} \sum_{u=-k}^{k} \sum_{v=-k}^{k} F(u+i, v+j)$$

Credit: S. Seitz

Mean filtering

$F[.,.]$

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

$G[.,.]$

	0	10	20						

$$G(i,j) = \frac{1}{(2k+1)^2} \sum_{u=-k}^{k} \sum_{v=-k}^{k} F(u+i, v+j)$$

Credit: S. Seitz

Mean filtering

$F[.,.]$

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

$G[.,.]$

	0	10	20	30					

$$G(i,j) = \frac{1}{(2k+1)^2} \sum_{u=-k}^{k} \sum_{v=-k}^{k} F(u+i, v+j)$$

Credit: S. Seitz

Mean filtering

H

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

F

0	0	0
0	0	0
90	90	90

$F[\cdot,\cdot]$

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

$G[\cdot,\cdot]$

	0	10	20	30	30				

$$G(i,j) = \frac{1}{(2k+1)^2} \sum_{u=-k}^{k} \sum_{v=-k}^{k} F(u+i, v+j)$$

Credit: S. Seitz

Mean filtering

$F[.,.]$

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

$G[.,.]$

	0	10	20	30	30				
				?					

$$G(i,j) = \frac{1}{(2k+1)^2} \sum_{u=-k}^{k} \sum_{v=-k}^{k} F(u+i, v+j)$$

Credit: S. Seitz

Mean filtering

$F[.,.]$

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

$G[.,.]$

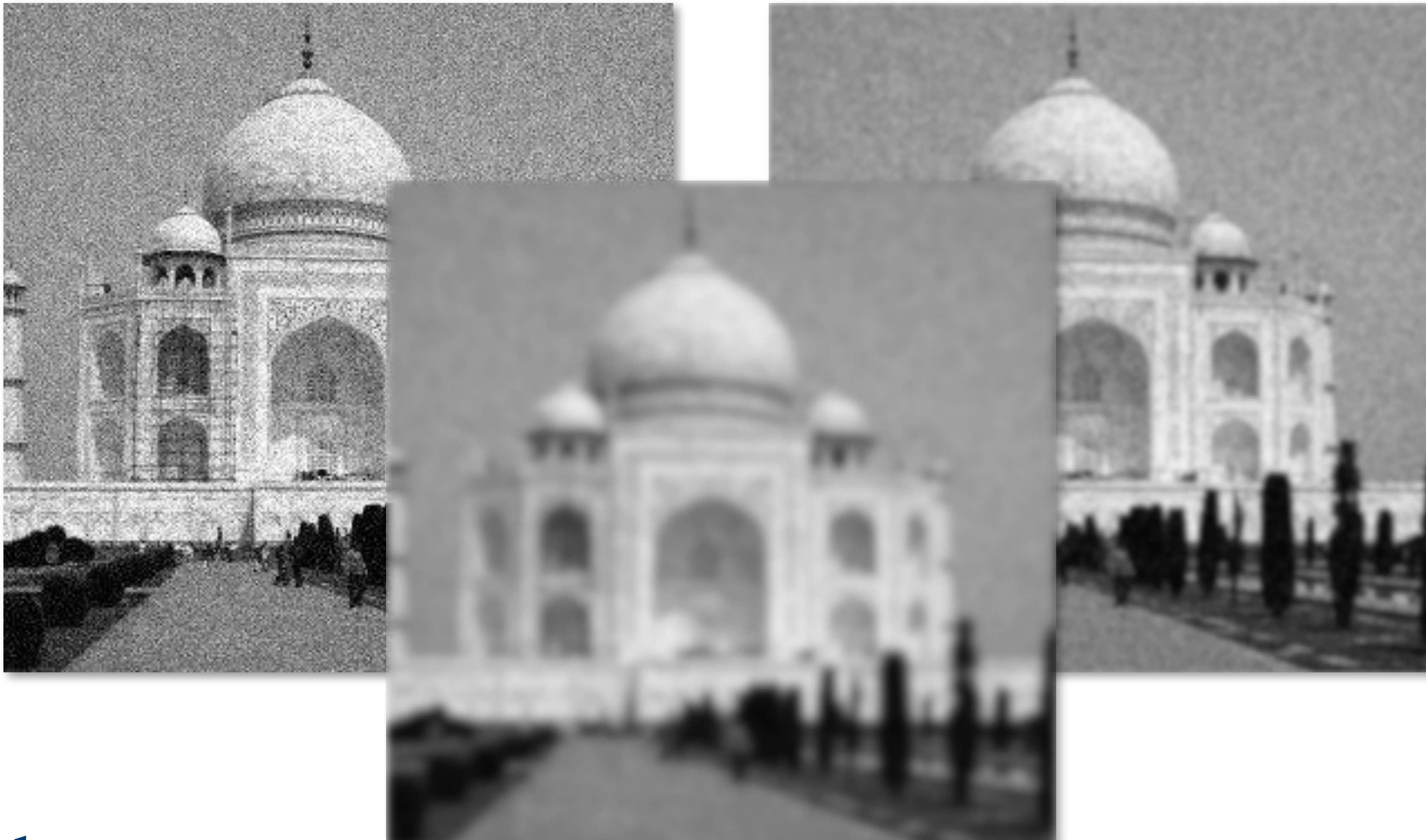
	0	10	20	30	30				
						?			
				50					

$$G(i,j) = \frac{1}{(2k+1)^2} \sum_{u=-k}^{k} \sum_{v=-k}^{k} F(u+i, v+j)$$

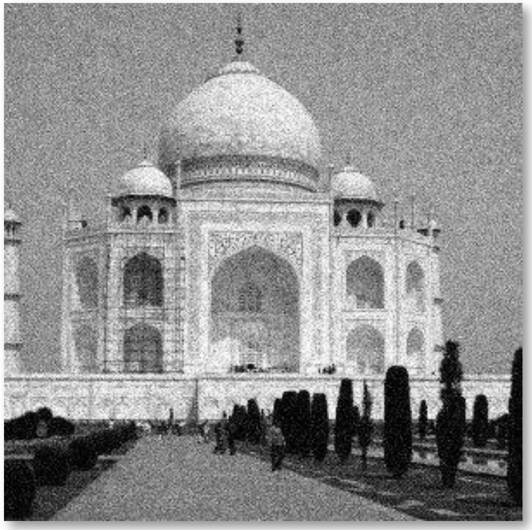
Credit: S. Seitz

Mean filtering

- Larger k => more blurring

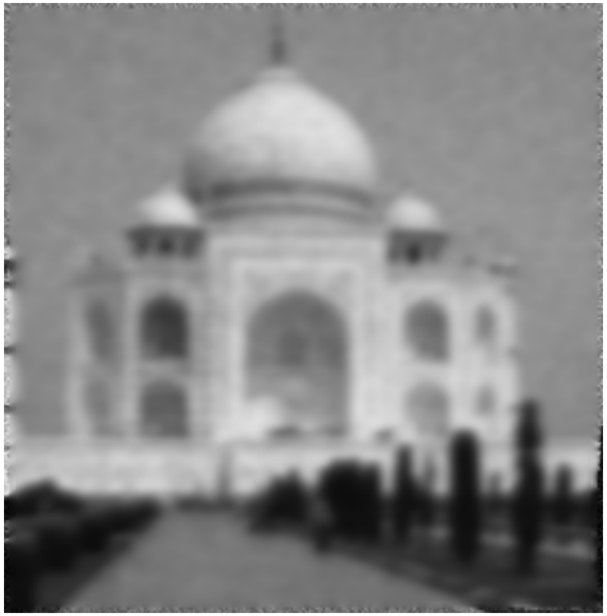


Mean filtering



- Larger k => more blurring

Mean filtering k=1



Mean filtering k=3

Mean filtering k=5

Cross-correlation

The mean filter is just a specific case of a *cross-correlation*, a very general operation

Let *F* an image, *H* be a kernel (of size 2k+1 x 2k+1), then the cross-correlation of F with H is:

$$G = H \otimes F$$

H

1/9	1/9	1/9
0	0	0
0	0	0

F

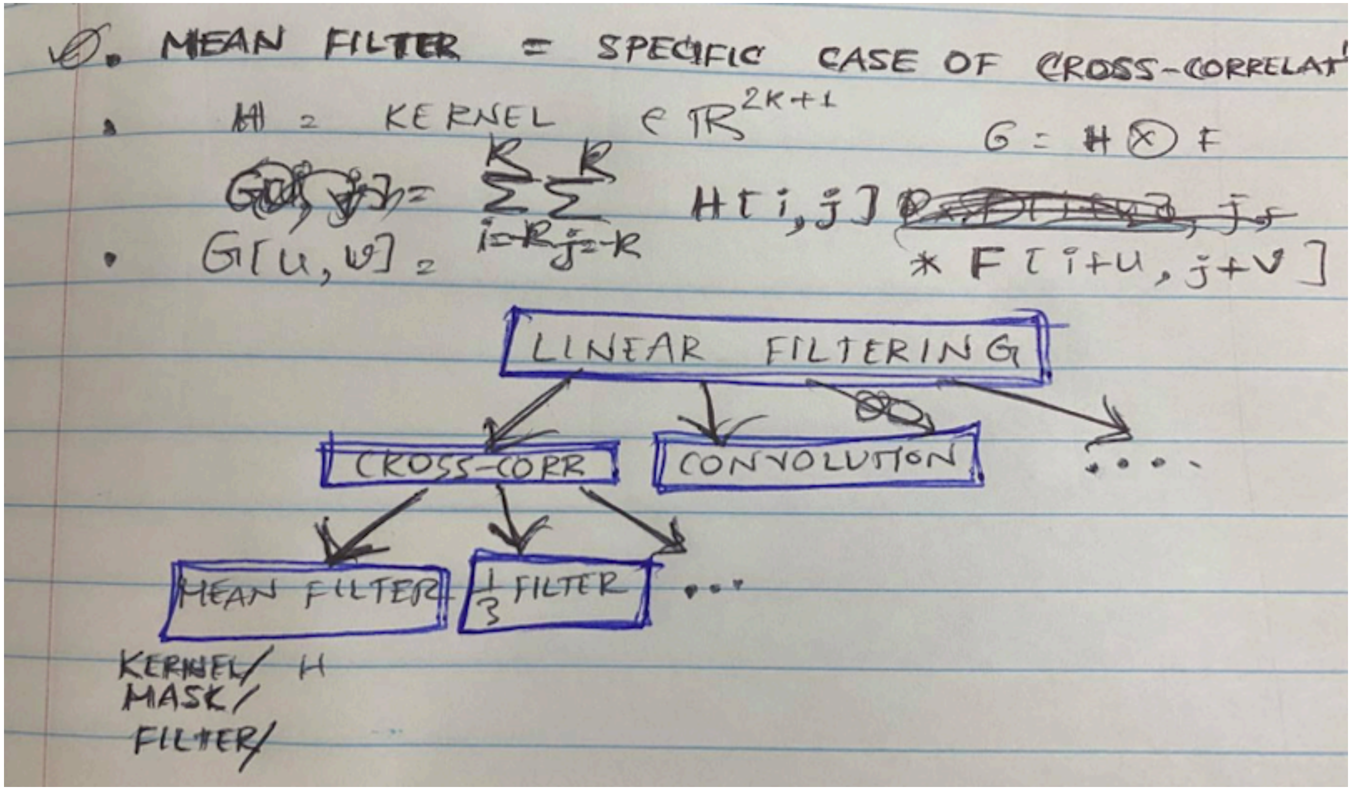
0	0	0
0	0	0
90	90	90

$$G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H[u,v] F[i+u, j+v]$$

0+0+0+0+0+0+(1/9*90)+(1/9*90)+(1/9*90)
= 30

Linear filtering

- More general version: linear filtering (cross-correlation, convolution)
  - Replace each pixel by a *linear combination* of its neighbors



Linear filtering

- More general version: linear filtering (cross-correlation, convolution)
  - Replace each pixel by a *linear combination* of its neighbors
- The prescription for the linear combination is called the “kernel” (or “mask”, “filter”)

10	5	3
4	6	1
1	1	8

Local image data

0	0	0
0	0.5	0
0	1	0.5

kernel

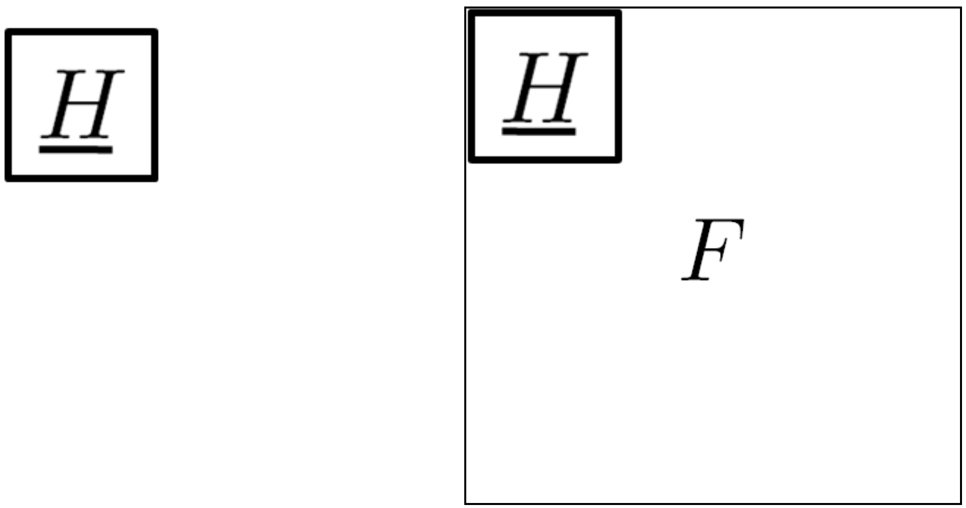
	8	

Modified image data

0.5*6 + 1*1 + 0.5*8 = 3 + 1 + 4 = 8

Source: L. Zhang

Cross correlation



Adapted from F. Durand

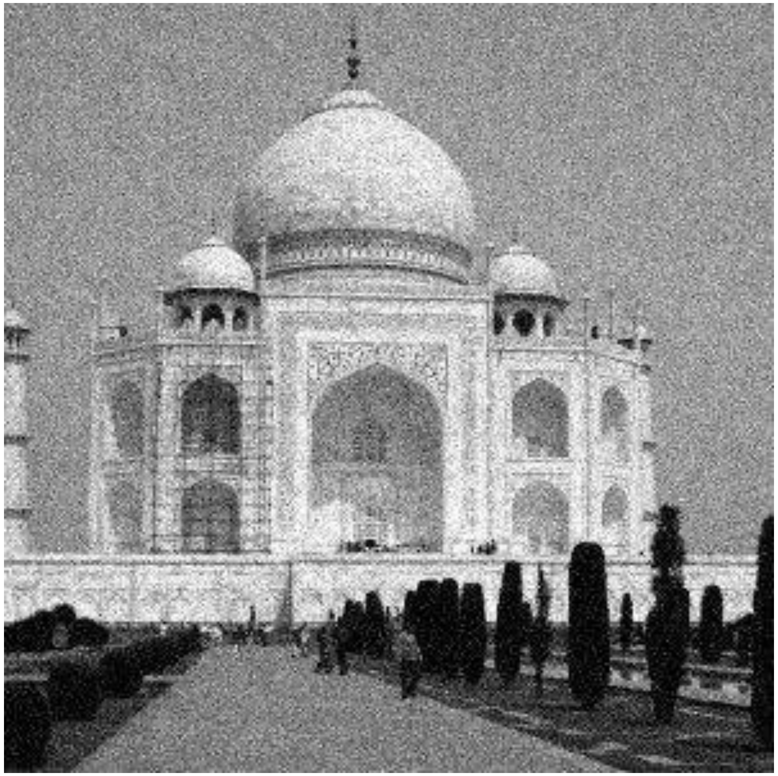
Cross-correlation examples

$$\underset{H}{\begin{array}{|c|c|c|}\hline 0 & 0 & 0 \\ \hline 0 & 1 & 0 \\ \hline 0 & 0 & 0 \\ \hline\end{array}} \otimes \underset{F}{\begin{array}{|c|c|c|c|c|}\hline 0 & 0 & 0 & 0 & 0 \\ \hline 0 & a & b & c & 0 \\ \hline 0 & d & e & f & 0 \\ \hline 0 & g & h & i & 0 \\ \hline 0 & 0 & 0 & 0 & 0 \\ \hline\end{array}} =$$

$$\underset{H}{\begin{array}{|c|c|c|}\hline a & b & c \\ \hline d & e & f \\ \hline g & h & i \\ \hline\end{array}} \otimes \underset{F}{\begin{array}{|c|c|c|c|c|}\hline 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 1 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 \\ \hline\end{array}} =$$

Coding activity: cross-correlation

- Reduce noise using cross-correlation from the given still scene ‘Tajmahal’ and others.



Coding activity: simple image manipulation tasks using PIL library

Coding activity: cross-correlation

```
  print('')
'''

# compute cross-correlation
F       = img_pil
new_img = img_pil

for y in range(kernel_size, rows-kernel_size):
  for x in range(kernel_size, cols-kernel_size):

    # compute cross-correlation centered at pixel location (x, y)
    old_pixel_value = F.getpixel((x, y)) # we don't need it anymore

    new_pixel_value = 0.0
    for v_row in range(-k,k+1):
      for u_col in range(-k,k+1):
        cur_kernel_value = H[v_row + k, u_col + k] # small trick to adjust the indexing
        cur_pixel_value  = F.getpixel((x + u_col, y + v_row))
        # MODIFICATION 1: calculate the updated value of pixel
        # your code ...

    # update the value at location (x,y) with newly computed pixel value
    # MODIFICATION 1: change the pixel value in the 'new_img' with the calculate new value
    # your code ...
```

Cross-correlation

The mean filter is just a specific case of a *cross-correlation*, a very general operation

Let *F* an image, *H* be a kernel (of size 2k+1 x 2k+1), then the cross-correlation of F with H is:

$$G = H \otimes F$$

H

1/9	1/9	1/9
0	0	0
0	0	0

F

0	0	0
0	0	0
90	90	90

$$G[i,j] = \sum_{u=-k}^{k} \sum_{v=-k}^{k} H[u,v]F[i+u,j+v]$$

0+0+0+0+0+0+(1/9*90)+(1/9*90)+(1/9*90)
= 30

More activities

- <u>In class activities:</u> Implement linear filtering with the following Kernels.

.33	0	0
.33	0	0
.33	0	0

Kernel 1

0	0	.33
0	.33	0
.33	0	0

Kernel 2

.33	.33	.33
0	0	0
0	0	0

Kernel 3

.33	0	0
0	.33	0
0	0	.33

Kernel 4