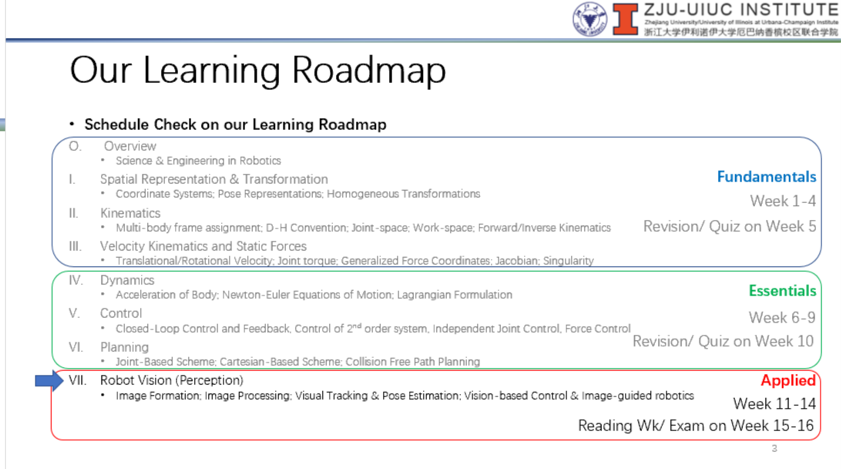
ECE 470: Introduction to Robotics
Lecture 23

Liangjing Yang
Assistant Professor, ZJU-UIUC Institute
liangjingyang@intl.zju.edu.cn

Wechat ID: Liangjing_Yang



Overview of Robot Vision

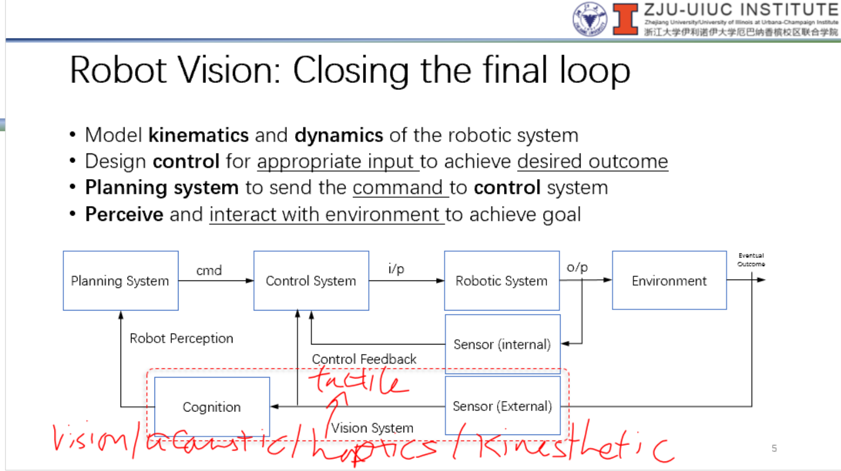
O. Introduction to Robot Vision
- What is Robot Vision?

I. Image Formation
- The science behind machine vision (+ represent as a form of signal)

II. Image Processing
- Common techniques to manipulate, enhance & analyse images

III. Robot Vision Applications
- 3D Vision; Photogrammetry; Vision-based techniques in robotics- visual servo, pose estimation, localization, mapping, navigation

Overview of Robot Vision



O. Introduction to Robot Vision

- **What is Robot Vision?**

I. Image Formation

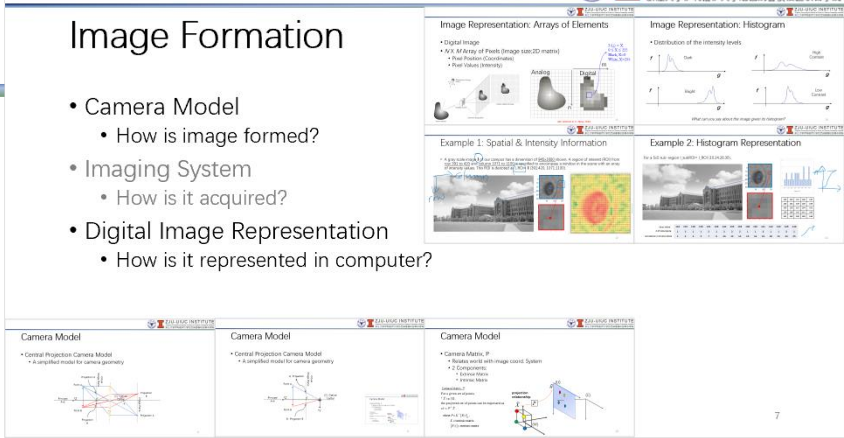
- The science behind machine vision (+ represent as a form of signal)

II. Image Processing

- Common techniques to manipulate, enhance & analyse images

III. Robot Vision Applications

- 3D Vision; Photogrammetry; Vision-based techniques in robotics- visual servo, pose estimation, localization, mapping, navigation



Overview of Robot Vision

O. Introduction to Robot Vision

- What is Robot Vision?

I. Image Formation

- **The science behind machine vision (+ represent as a form of signal)**

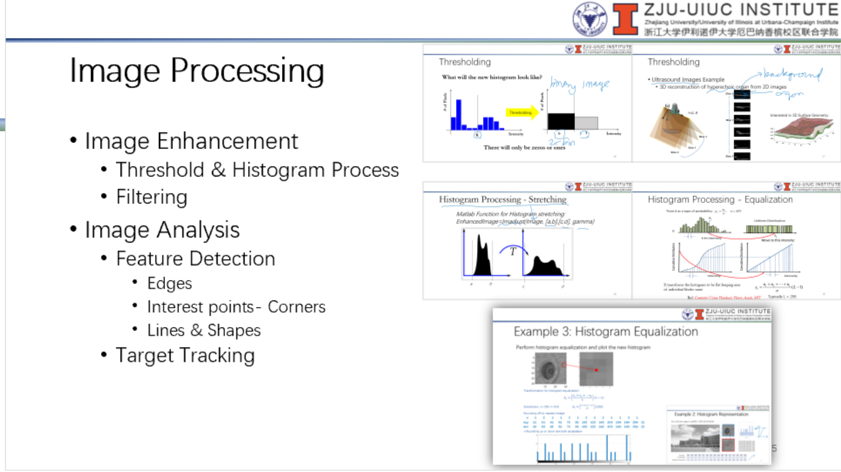
II. Image Processing

- Common techniques to manipulate, enhance & analyse images

III. Robot Vision Applications

- 3D Vision; Photogrammetry; Vision-based techniques in robotics- visual servo, pose estimation, localization, mapping, navigation

Overview of Robot Vision



O. Introduction to Robot Vision
- What is Robot Vision?

I. Image Formation
- The science behind machine vision (+ represent as a form of signal)

II. Image Processing
- **Common techniques to manipulate, enhance & analyse images**

III. Robot Vision Applications
- 3D Vision; Photogrammetry; Vision-based techniques in robotics- visual servo, pose estimation, localization, mapping, navigation

Image Processing

- Image Enhancement
  - Thresholding & Histogram Processing
  - Filtering
- Image Analysis
  - Feature Detection
    - Edges
    - Interest points- Corners
    - Lines & Shapes
    - Target Tracking

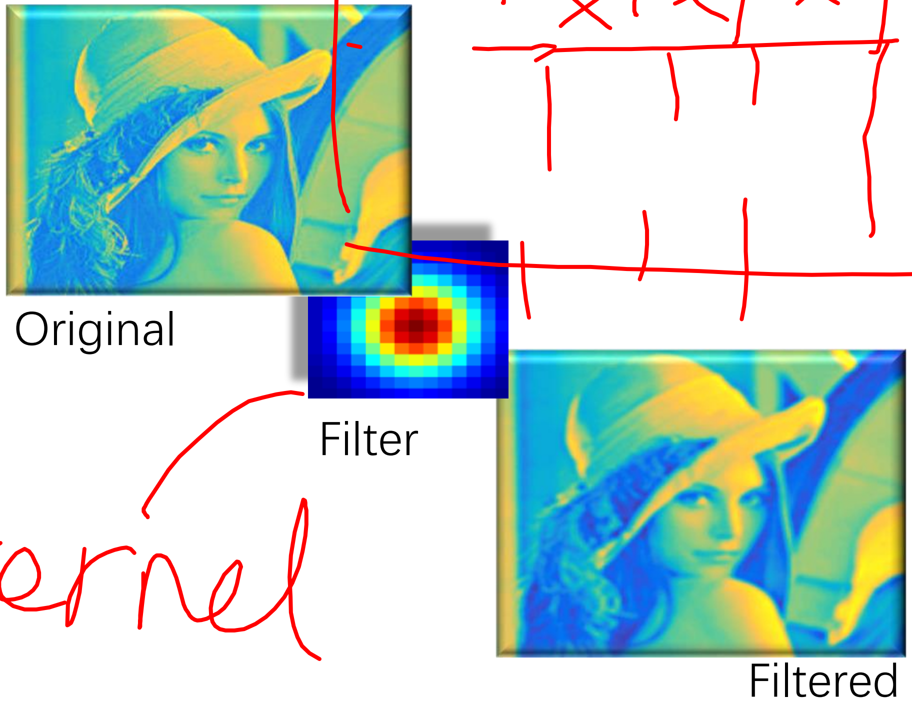
Image Processing

- Image Enhancement
  - Thresholding & Histogram Processing
  - Filtering
- Image Analysis
  - Feature Detection
    - Edges
    - Interest points- Corners
    - Lines & Shapes
    - Target Tracking

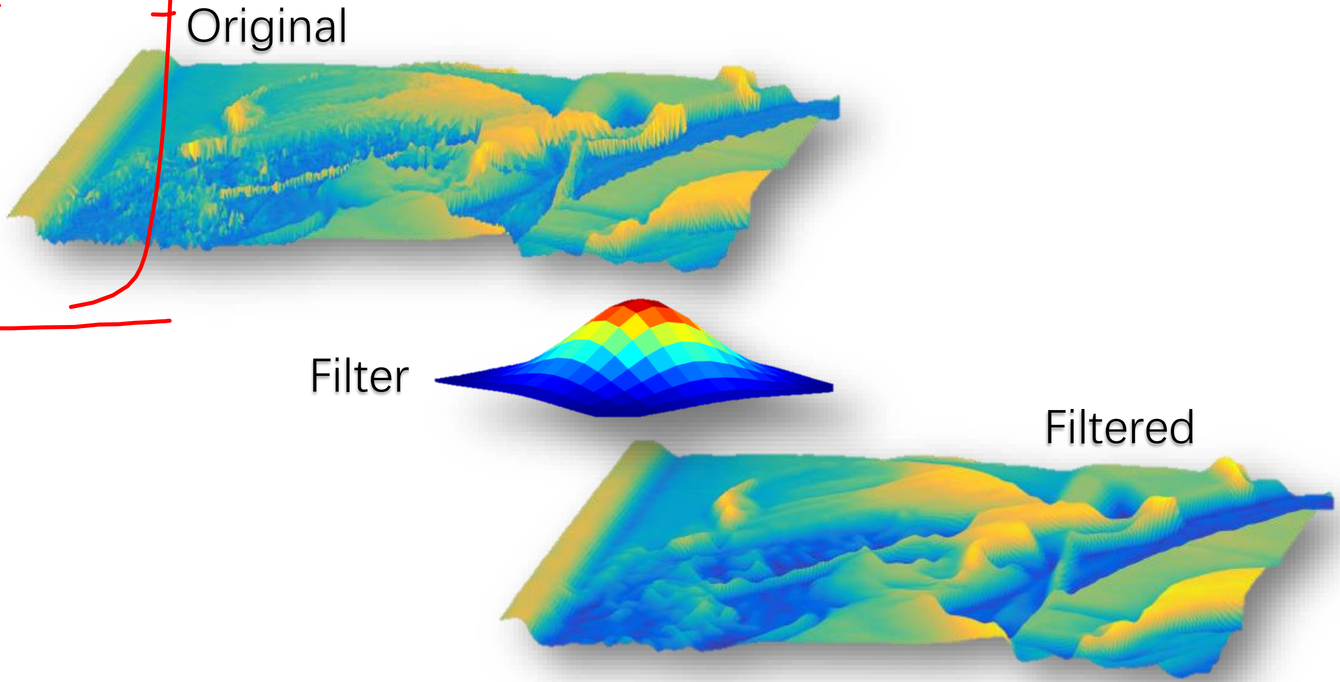
Filtering

- Operation that modify pixels based on their neighbourhood values

to be transform

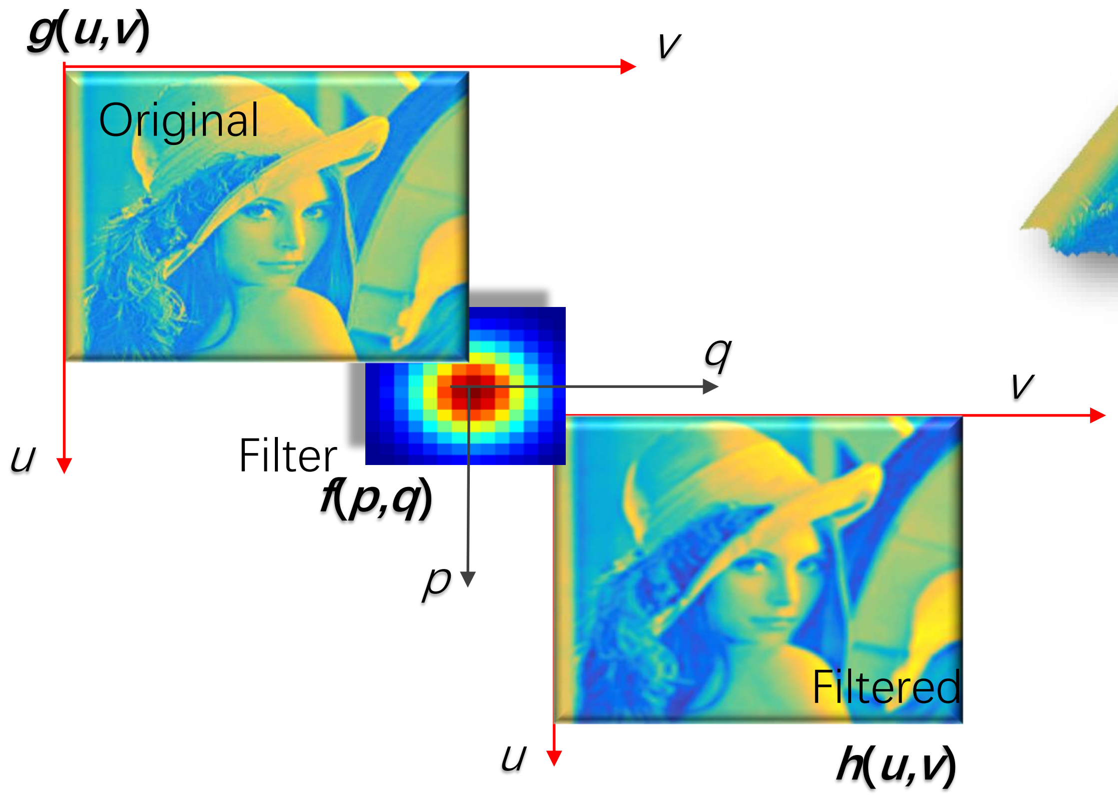


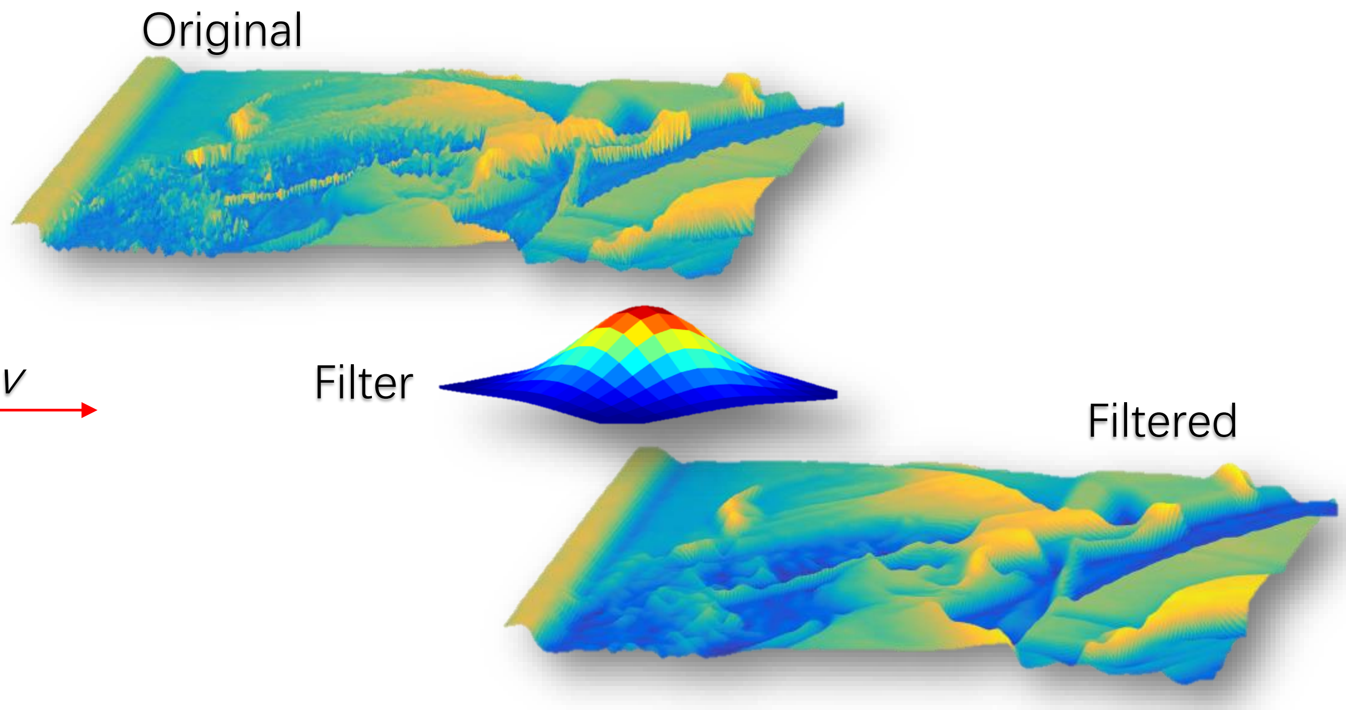
kernel



Filtering

- Operation that modify pixels based on their neighbourhood values
- Filters/kernels can be designed to operate on pixels by convolution
  - mean, weighted sum etc.
  - Non-linear operator: Median Filter





Filtering

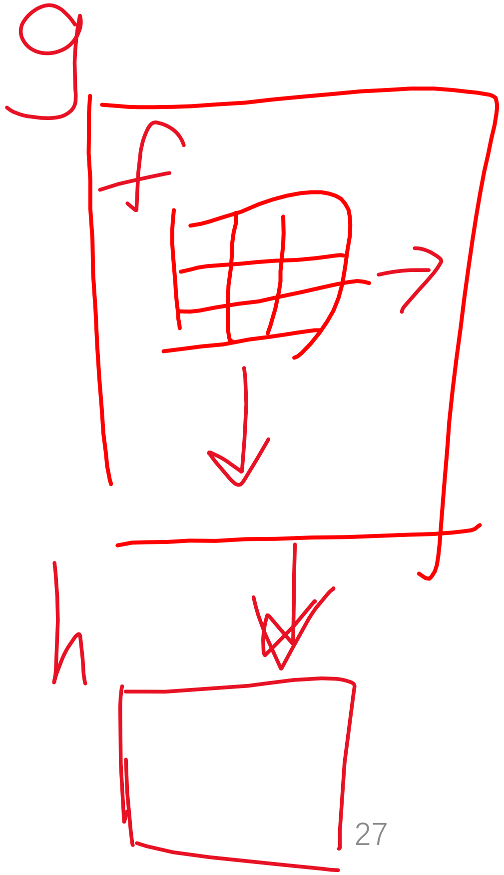
- Operation that modify pixels based on their neighbourhood values
- Filters/kernels can be designed to operate on pixels by convolution
  - mean, weighted sum etc.
  - Non-linear operator: Median Filter
- Enhancement effects
  - smoothing, sharpening, and edge enhancement.

Filtering

- Operation that modify pixels based on their neighbourhood values
- Filters/kernels can be designed to operate on pixels by convolution
- Enhancement effects

Operation	Kernel ω	Image result g(x,y)
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Edge detection	$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$	
	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	
	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	

Operation	Kernel ω	Image result g(x,y)
Sharpen	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Box blur (normalized)	$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	
Gaussian blur 3 × 3 (approximation)	$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$	
Gaussian blur 5 × 5 (approximation)	$\frac{1}{256}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$	
Unsharp masking 5 × 5 Based on Gaussian blur with amount as 1 and threshold as 0 (with no image mask)	$\frac{-1}{256}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & -476 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$	



https://en.wikipedia.org/wiki/Kernel_(image_processing)

Filtering

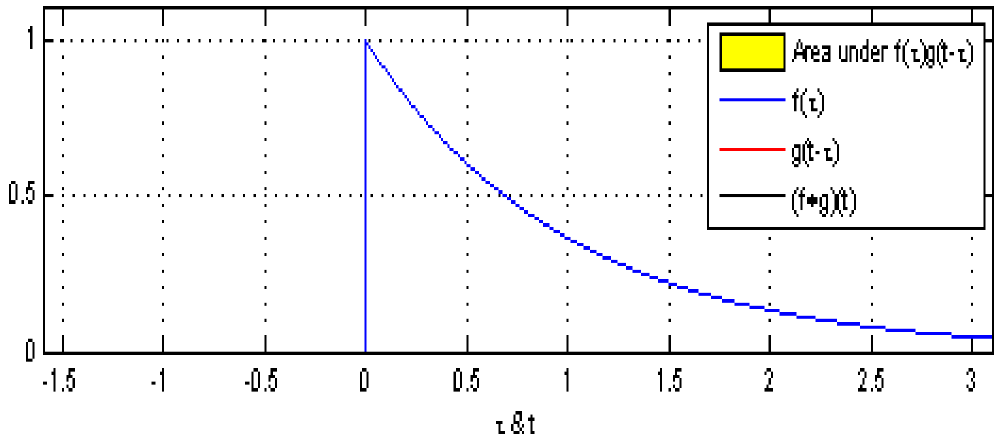
- Convolution Operation
  - pass an image ***g*** size *M* x*N* through a filter ***f*** size *P*x*Q*
  - obtain an output image ***h***

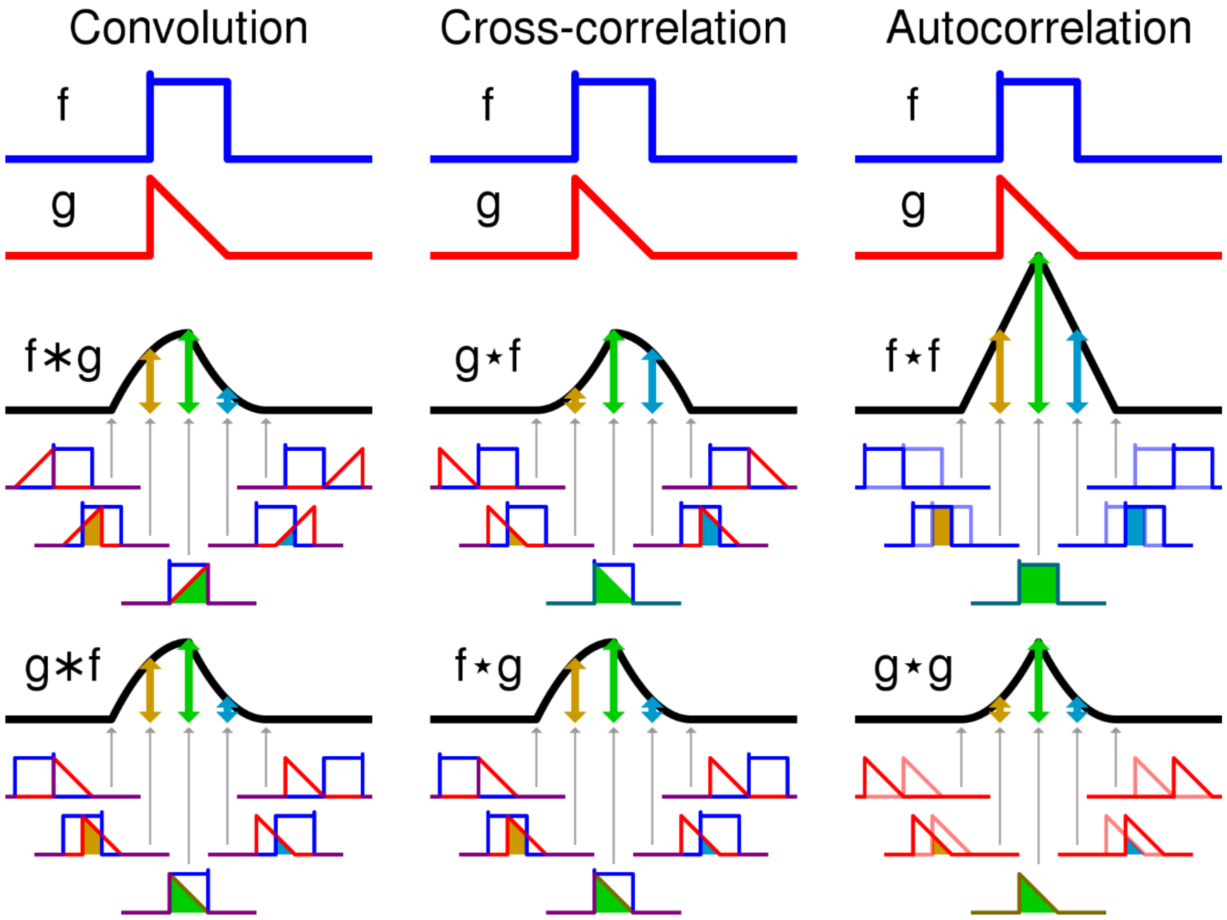
$$h = f * g$$

$$h(u,v) = \sum_{p}^{P} \sum_{q}^{Q} f(p,q) g(u-p, v-q)$$

Filtering

- Convolution Operation
- Related Operations





By Convolution_of_spiky_function_with_box.gif: Brian Ambergderivative work: Tinos (talk) - Convolution_of_spiky_function_with_box.gif, https://commons.wikimedia.org/w/index.php?curid=11003944

Filtering

- Mean filter

$$f(i,j)=\frac{1}{PQ}$$

$$F=\frac{1}{PQ}\begin{array}{cc} & Q \\ P & \begin{bmatrix} 1 & \cdots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \cdots & 1 \end{bmatrix} \end{array}$$

Filtering

- How would you implement a Mean filter?

j \ i								
1	0	0	0	0	0	0	0	0
2	0	0	0	0	**1**	0	0	0
3	0	0	**1**	0	0	0	0	0
4	0	0	0	0	45	45	45	0
5	0	0	45	45	45	45	45	45
6	0	0	45	**60**	45	45	45	0
7	0	0	45	45	45	**0**	45	0
8	0	0	0	0	45	45	45	0

Filtering

- How would you implement a Mean filter?

convolution

*

j \ i								
1	0	0	0	0	0	0	0	0
2	0	0	0	0	**1**	0	0	0
3	0	0	**1**	0	0	0	0	0
4	0	0	0	0	45	45	45	0
5	0	0	45	45	45	45	45	45
6	0	0	45	**60**	45	45	45	0
7	0	0	45	45	45	**0**	45	0
8	0	0	0	0	45	45	45	0

Filtering

- Example: Mean filter

$h(3,3) = [f(3,3) \times g(1,1)] \cdots + [f(p,q) \times g(i,j)] + \cdots$

$= (0 \times 1/9) + (0 \times 1/9) + (0 \times 1/9) + \cdots$
$(0 \times 1/9) + (1 \times 1/9) + (0 \times 1/9) + \cdots$
$(0 \times 1/9) + (0 \times 1/9) + (0 \times 1/9)$

$h(4,6) = [f(4,6) \times g(1,1)] \cdots + [f(p,q) \times g(i,j)] + \cdots$

$= (45 \times 1/9) + (45 \times 1/9) + (45 \times 1/9) + \cdots$
$(45 \times 1/9) + (60 \times 1/9) + (45 \times 1/9) + \cdots$
$(45 \times 1/9) + (45 \times 1/9) + (45 \times 1/9)$

$h(4,7) = [f(4,7) \times g(1,1)] \cdots + [f(p,q) \times g(i,j)] + \cdots$

$= (0 \times 1/9) + (0 \times 1/9) + (0 \times 1/9) + \cdots$
$(45 \times 1/9) + (45 \times 1/9) + (0 \times 1/9) + \cdots$
$(45 \times 1/9) + (45 \times 1/9) + (45 \times 1/9)$

convolution

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

Noise Removal 0

Edge Blurring 25

j \ i	1	2	3	4	5	6	7	8
1	0	0	0	0	0	0	0	0
2	0	0	0	0	1	0	0	0
3	0	0	1	0	0	0	0	0
4	0	0	0	0	45	45	45	0
5	0	0	45	45	45	45	45	45
6	0	0	45	60	45	45	45	0
7	0	0	45	45	45	0	45	0
8	0	0	0	0	45	45	45	0

Feature Smoothing 47 40

Filtering

- Gaussian filter

$$f(p,q)=\frac{1}{2\pi\sigma^2}e^{\frac{-\left(p^2+q^2\right)}{2\sigma^2}}$$

e.g. for σ= 3, f(0,0)=e/(18π)

for a 5 x 5 filter,

$$F=\begin{pmatrix} f(-2,-2) & f(-2,-1) & f(-2,0) & f(-2,1) & f(-2,2) \\ f(-1,-2) & f(-1,-1) & f(-1,0) & f(-1,1) & f(-1,2) \\ f(0,-2) & f(-0,-1) & \boxed{f(0,0)} & f(0,1) & f(0,2) \\ f(1,-2) & f(1,-1) & f(1,0) & f(1,1) & f(1,2) \\ f(2,-2) & f(2,-1) & f(2,0) & f(2,1) & f(2,2) \end{pmatrix}$$

Gaussian distribut^n

Filtering

- Gaussian filter

$$f(p,q)=\frac{1}{2\pi\sigma^2}e^{\frac{-(p^2+q^2)}{2\sigma^2}}$$

for scale σ =3,

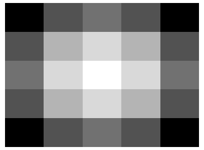
$$F = \begin{pmatrix} 0.0318 & 0.0375 & 0.0397 & 0.0375 & 0.0318 \\ 0.0375 & 0.0443 & 0.0469 & 0.0443 & 0.0375 \\ 0.0397 & 0.0469 & 0.0495 & 0.0469 & 0.0397 \\ 0.0375 & 0.0443 & 0.0469 & 0.0443 & 0.0375 \\ 0.0318 & 0.0375 & 0.0397 & 0.0375 & 0.0318 \end{pmatrix}$$

Filtering

- Gaussian filter

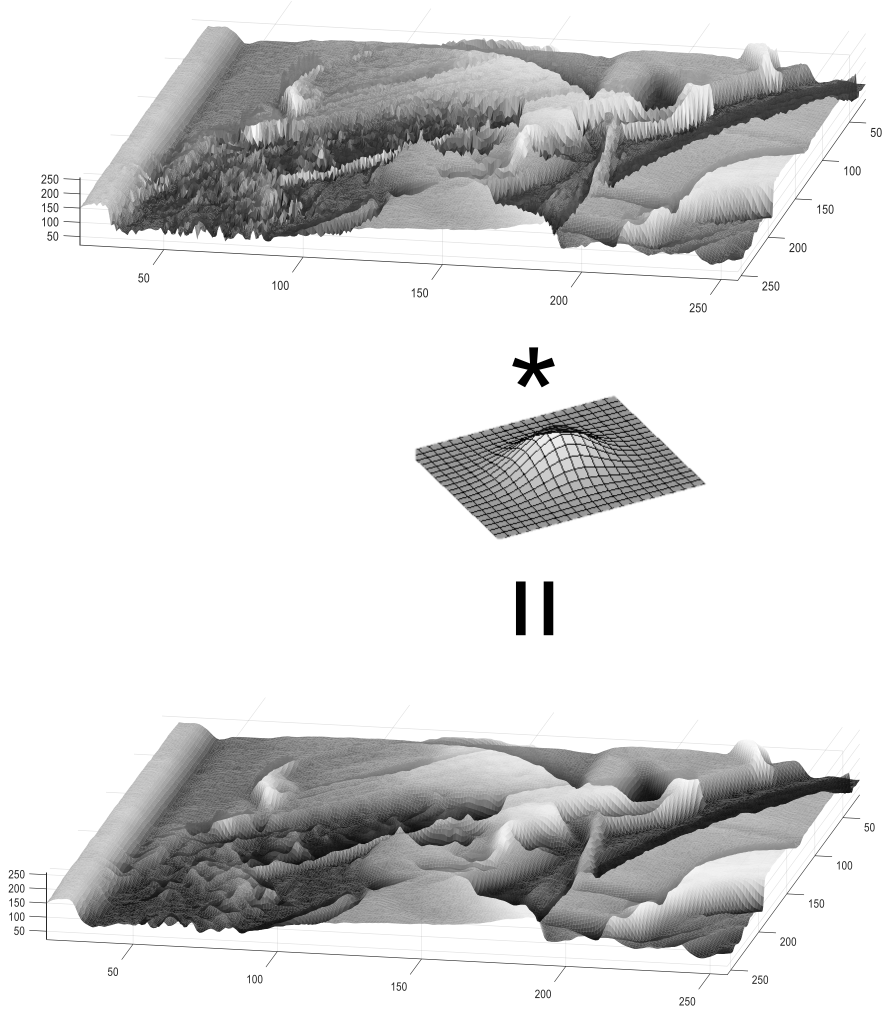
Original image

*



=

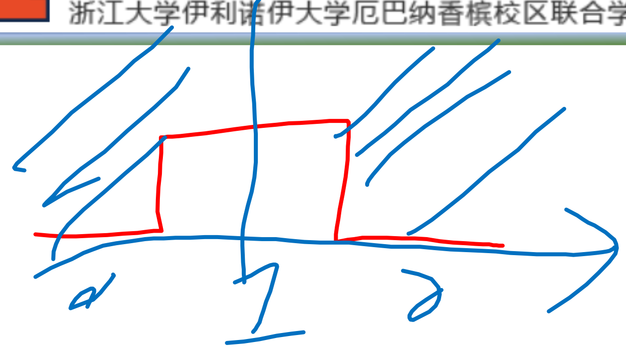
Smoothed image, σ = 3



What about filtering in the frequency domain?

- Recall your signal processing courses; probably very used to this

Filtering in the frequency domain



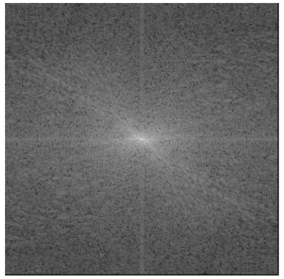
$$H(u,v) = F(u,v)G(u,v)$$

Fourier Transform → Filter Function → Inverse Fourier Transform

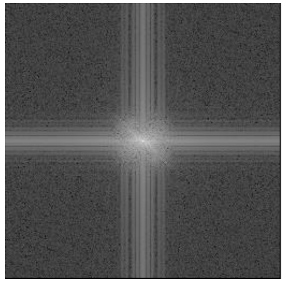
Pre-processing

Post-processing

G(x,y)



F(u,v) G(x,y)



g(x,y)

Original Image

Filtered Image

h(x,y)

Noise Filtering using FFT

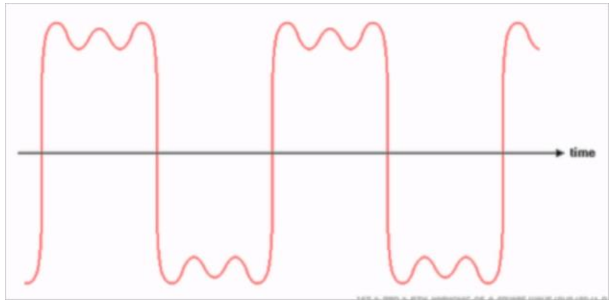
Fourier Transform

- Transforms the time domain signal g(t) to the frequency domain signal G(f)
- Each signal with same frequency becomes a spike in the frequency domain – easy to separate
- Desired signal h(t) can be obtained by removing noise in the frequency domain

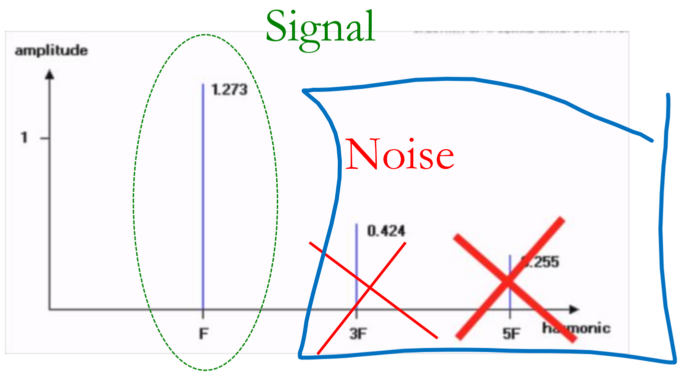
$$F(g(t)) = G(f) = \int_{-\infty}^{\infty} g(t) e^{-i2\pi ft} \, dt$$

$$g(t) = F^{-1}(G(f)) = \int_{-\infty}^{\infty} G(f) e^{i2\pi ft} \, df$$

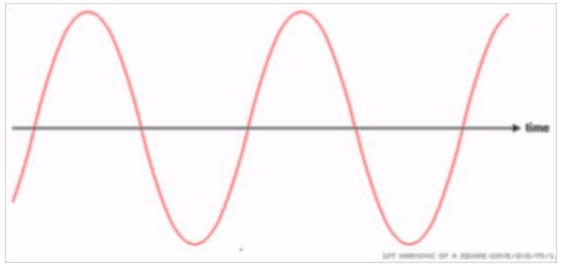
g(t)



G(f) =F(g(t))



h(t)



Filtering in the frequency domain

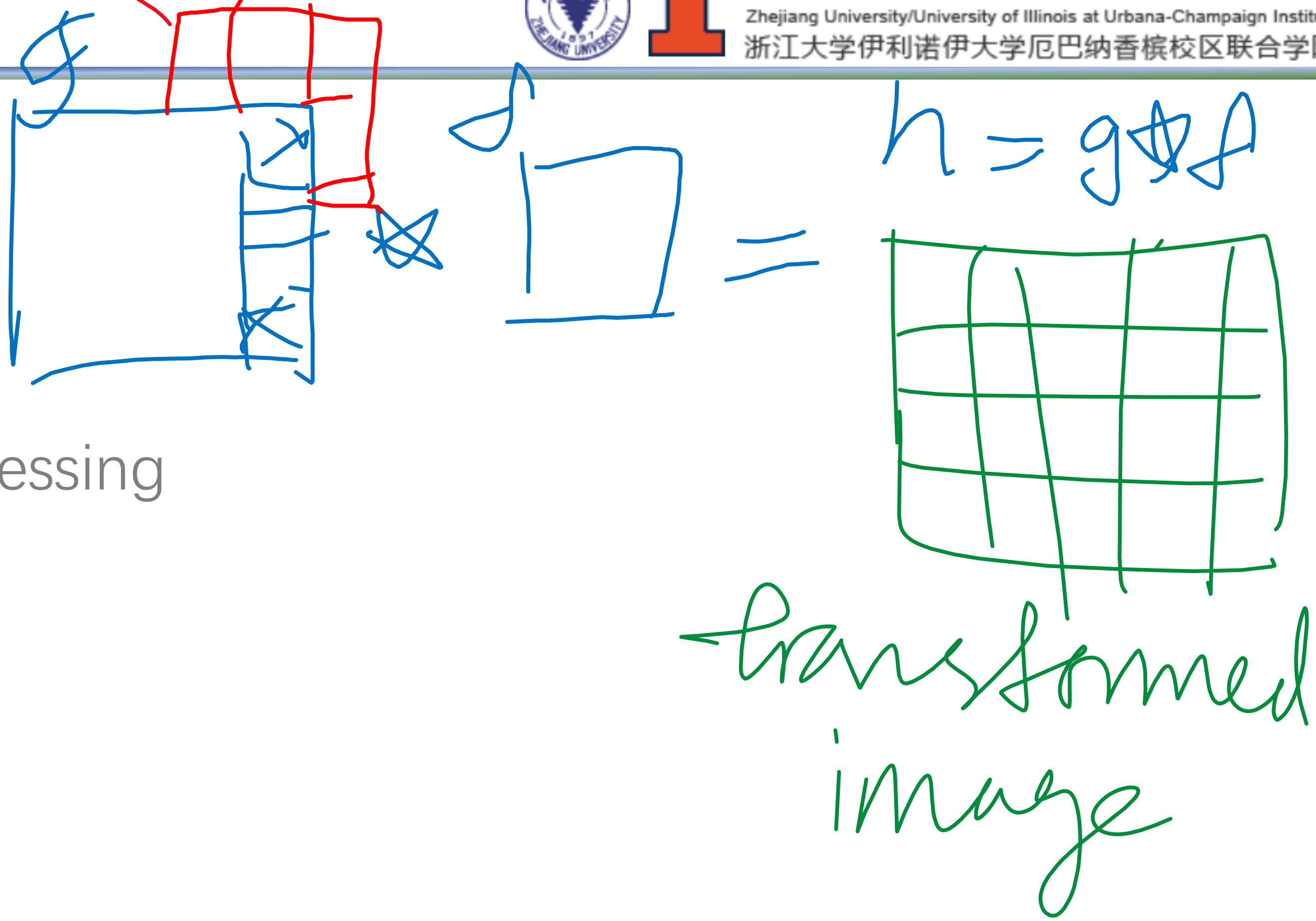
- We can use low-pass filter for noise removal since noise are associated with high frequency
- How would high-pass filter be useful?

Image Processing

- Image Enhancement
  - Thresholding & Histogram Processing
  - Filtering
- Image Analysis
  - Feature Detection
    - Edges
    - Interest points- Corners
    - Lines & Shapes
    - Target Tracking

Ways of handling edges

Image Processing



- Image Enhancement
  - Thresholding & Histogram Processing
  - Filtering
- Image Analysis
  - <u>Feature Detection</u>
    - Edges
    - Lines & Shapes
    - Interest points- Corners
    - Target Tracking

Image Analysis

- Extraction of relevant information from images (by means of image processing technique)
- Relevant information can include
  - Contours: Edges
  - Geometries: Lines & Shapes
  - Interest Points: Corners, blobs etc.
  - Object Motions: Target Tracking

Point and line detection using your intuition

- Given this image, identify the location of the point

10	9	10	20
10	252	9	10
10	9	10	10
9	10	10	9

- Given the following images, identify the lines

10	250	10	9
10	252	9	10
10	251	10	10
9	252	10	9

250	9	10	9
10	252	9	10
10	9	251	10
9	10	10	252

250	9	10	9
249	252	9	10
10	250	251	10
9	10	250	252

Edge Detection

- Before we can identify the lines and contours, we need to perform edge detection
- Types of edges:
  - Step edge:
    - The image intensity abruptly changes from one value to a different value

  - Ramp edge:
    - Intensity change is not instantaneous, but occurs over a finite distance

Edge Detection

- Edges are locations with high image gradient or derivative
- A simple edge detection:
    - Compute gradient magnitude at each pixel
    - If the gradient magnitude exceeds a threshold, report a edge point
- The derivative of each pixel can be estimated using finite difference method:
    - $\frac{\partial I}{\partial x} = \frac{I(x+1,y)-I(x-1,y)}{2}$ → Δ in the x-direction
    - $\frac{\partial I}{\partial y} = \frac{I(x,y+1)-I(x,y-1)}{2}$ → Δ in the y-direction

Gradient Vector

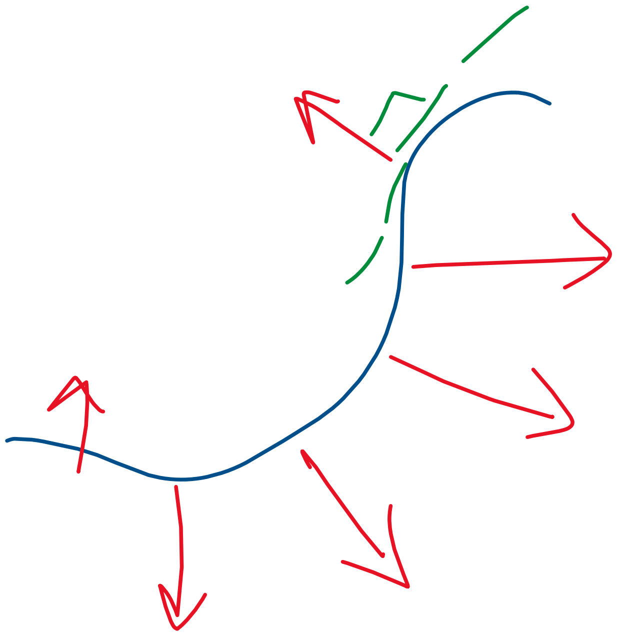
- $\frac{\partial I(x.y)}{\partial x} = \frac{I(x+1,y)-I(x-1,y)}{2}$
- $\frac{\partial I(x,y)}{\partial y} = \frac{I(x,y+1)-I(x,y-1)}{2}$
- Gradient Vector: $\nabla I(x,y) = \left[\frac{\partial I(x,y)}{\partial x}, \frac{\partial I(x,y)}{\partial y}\right]^T$
  - $|\nabla I(x,y)| = \sqrt{\left(\frac{\partial I(x,y)}{\partial x}\right)^2 + \left(\frac{\partial I(x,y)}{\partial y}\right)^2}$
  - $\theta(x,y) = \tan^{-1}\left(\frac{\partial I(x,y)}{\partial y} / \frac{\partial I(x,y)}{\partial x}\right)$

250	250	250	250
250	250	250	10
250	250	10	10
250	10	10	10

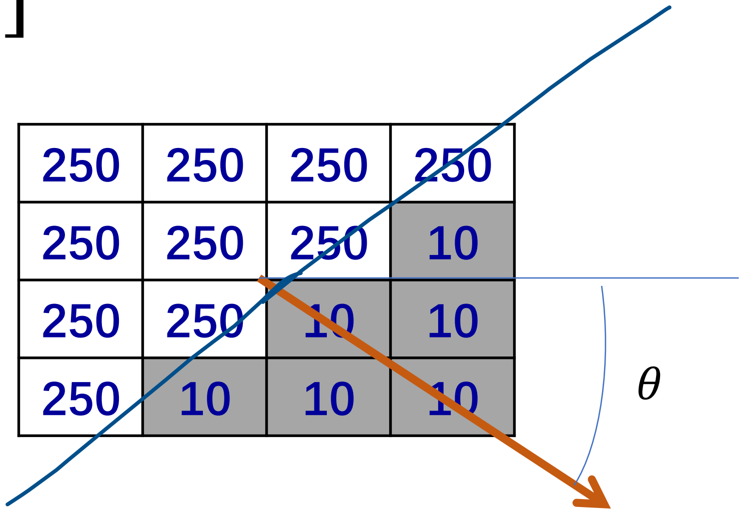
θ

Edge normal

Gradient Vector



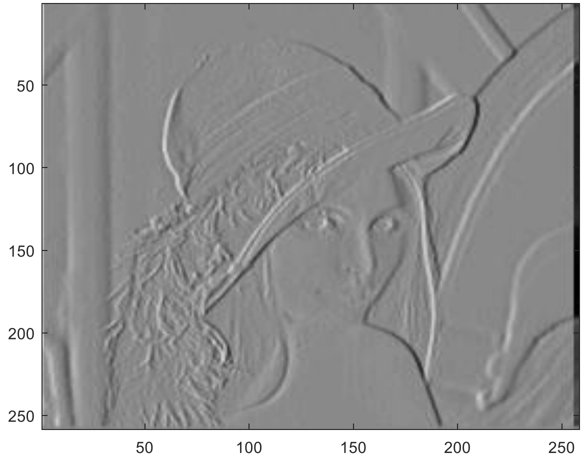
- $\frac{\partial I(x.y)}{\partial x} = \frac{I(x+1,y)-I(x-1,y)}{2}$
- $\frac{\partial I(x,y)}{\partial y} = \frac{I(x,y+1)-I(x,y-1)}{2}$
- Gradient Vector: $\nabla I(x,y) = \left[\frac{\partial I(x,y)}{\partial x}, \frac{\partial I(x,y)}{\partial y}\right]^T$
  - $|\nabla I(x,y)| = \sqrt{\left(\frac{\partial I(x,y)}{\partial x}\right)^2 + \left(\frac{\partial I(x,y)}{\partial y}\right)^2}$
  - $\theta(x,y) = \tan^{-1}\left(\frac{\partial I(x,y)}{\partial y} / \frac{\partial I(x,y)}{\partial x}\right)$



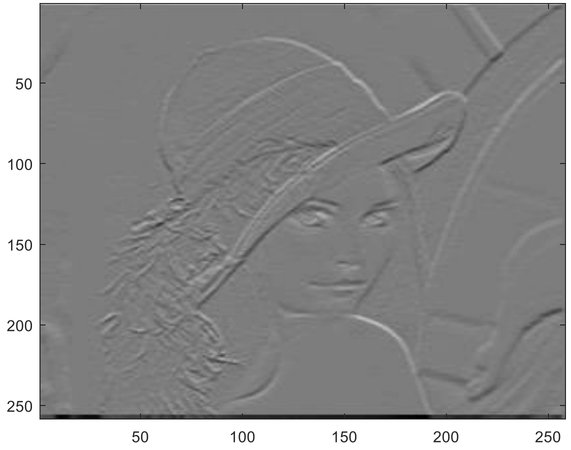
250	250	250	250
250	250	250	10
250	250	10	10
250	10	10	10

Sobel operator

- $\frac{\partial I(x.y)}{\partial x} = \frac{I(x+1,y)-I(x-1,y)}{2}$
- $\frac{\partial I(x,y)}{\partial y} = \frac{I(x,y+1)-I(x,y-1)}{2}$



- Sobel Operator:
  - $\frac{\partial I(x.y)}{\partial x} = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} * \mathbf{A}$
  - $\frac{\partial I(x.y)}{\partial y} = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} * \mathbf{A}$



  - where A is the source image and $*$ denotes the 2-dimensional convolution operation

Canny Edge Detection (Next Lecture……)

- Canny edge detection is probably the most used and taught edge detection algorithm
- Involves 5 steps:
  1. Apply Gaussian filter to smoothen the image in order to remove the noise
  2. Find the intensity gradients of the image
  3. Apply non-maximum suppression to get rid of spurious response to edge detection
  4. Apply edge detection using two threshold value
  5. Finalize edge detection by hysteresis
- J. Canny, "A Computational Approach to Edge Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, no. 6, 1986.