

# SCC.366: Media Coding & Processing

2022-2023

Week 01 – Lecture 1

## Introduction to Media Coding and Processing

Dr. Hossein Rahmani

Senior Lecturer in Data Science

Email and Team: [h.rahmani@lancaster.ac.uk](mailto:h.rahmani@lancaster.ac.uk)

# How this Course is Taught

- Lectures
  - Monday:  
09:00-10:00 (Furness LT 2)



- Tuesday:  
17:00-18:00 (Mngt School LT 08)

- Labs (Weeks: 2, 4, 6, 8, 10)

- Group#1-Thursday: 09:00-11:00 (Science & Technology B070)
- Group#2-Thursday: 16:00-18:00 (Science & Technology B080)



# Learning Objectives

- To gain an understanding of
  - **Media coding:** how media (e.g., **images**, **audio** and **videos**) are encoded within the digital domain
  - **Media processing:** how additional information can be generated through processing images and videos
  - Emerging technologies in media processing
- Module overview
  - Image processing (in spatial and frequency domain)
    - Spatial filters, histograms, edges, segmentation, matching, watermarking, etc.
  - Image coding
    - Lossless compression
    - Lossy compression, e.g., JPEG compression
  - Video coding and analysis, e.g., motion estimation, change detection

# How this course is assessed

---

- **Coursework (40%)**

| Number | Title           | Deadline                            | Weight in the final mark (%) | Return Date |
|--------|-----------------|-------------------------------------|------------------------------|-------------|
| 1      | Media Coding I  | Friday Week 6 (16:00, 18 Nov 2022)  | 30%                          | Week 7-9    |
| 2      | Media Coding II | Friday Week 10 (16:00, 16 Dec 2022) | 70%                          | Week 11     |

- **Final Exam: (60%)**

# Feedback

---

## How is feedback provided?

**Detailed/personal:** by the academics and TAs.

**Overall/class:** by the academics in lectures (and/or via the class announcements). We will give updates on how the class is performing overall, and any common mistakes/areas to improve.

# What is Plagiarism?

---

- **Submitting someone else's work as your own, including:**
  - Submitting (e.g.) code that someone else wrote
  - Paying for someone else to do it for you
  - Working on a piece of non-group work together as a group, and submitting it as individual work
  - Sharing of code that you then possibly adapt



# What You Can Expect from Us

---

- to make all our lecture notes/ video captures available on moodle
- to give you references to follow up
- to personally check the labs are running smoothly, and the TAs are offering support
- to arrange extra support if you've already tried the normal routes (book, web, forum, TAs)
- to offer feedback on formative coursework promptly
- to respond to email (ideally as a last resort! - *note: we get more email than we can handle*, and have a lot of teaching/research/admin commitments, so are often *not* in our offices!)

# What We Expect from You

---

Integrity (no plagiarism, no faking results) and effort (active learning):

- i.e. come to lectures, even early ones (it helps!)
- go to labs (these are compulsory!)
- get the textbook and use our/the world's resources responsibly
- take notes
- read around the subject/try things for yourself
- ask us questions in lectures/labs
- take notes (again, because the slides are not enough when you try to revise, really...!)
- plan your time and coursework carefully

# How do I get help?

---

- Use the labs to ask for help
- Use the course forum on Moodle
- My office is on D floor at InfoLab (D17):
  - Pop in my office or alternatively send me an email to arrange a meeting
  - We expect you to have a) tried, b) used the teaching resources, c) looked for solutions yourself, d) asked the TAs
- Any other issues:
  - [scc-teaching-office@lancaster.ac.uk](mailto:scc-teaching-office@lancaster.ac.uk)

## Further Readings (optional)

---

[Gon] Gonzalez and Woods,  
*"Digital Image Processing, 3<sup>rd</sup> Ed."*,  
Prentice Hall 2008.

[Cov] Cover and Thomas,  
*"Elements of Information Theory"*,  
Wiley-Blackwell 1991.

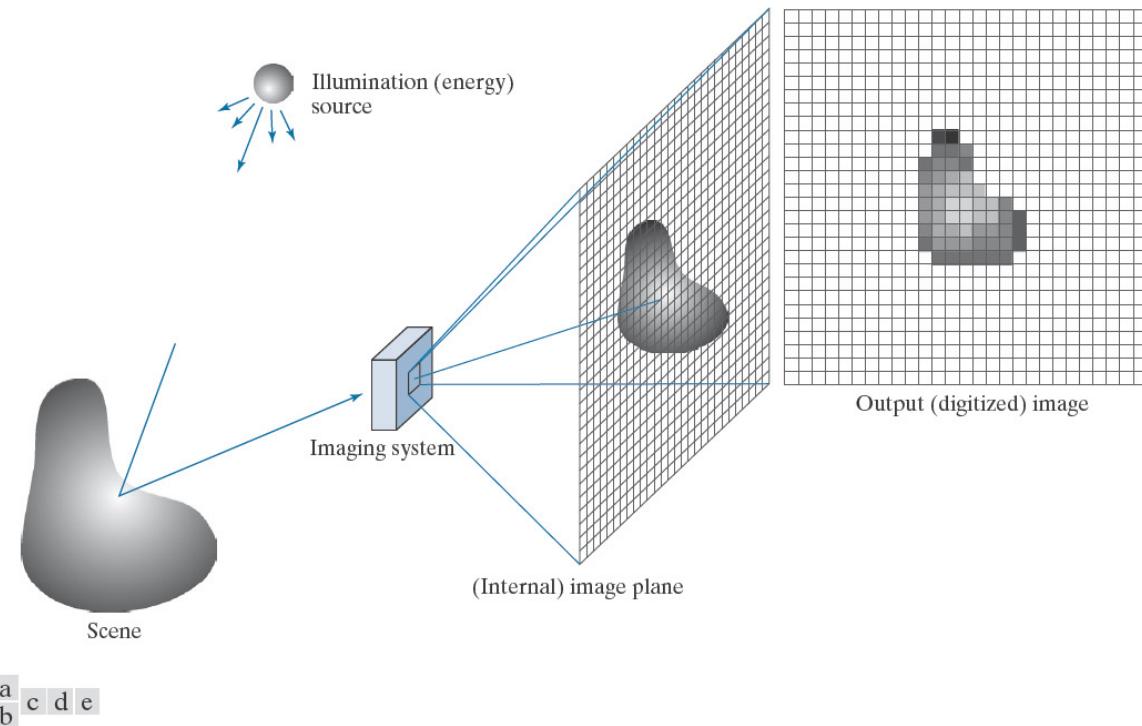
# Outline

---

- **What is a digital image? (Lecture 1)**
- Different types of digital images (**Lecture 1**)
- What is digital image processing? (**Lecture 2**)
  - Image/video compression
  - Image enhancement
  - Image understanding (computer vision)
- Summary of the main points

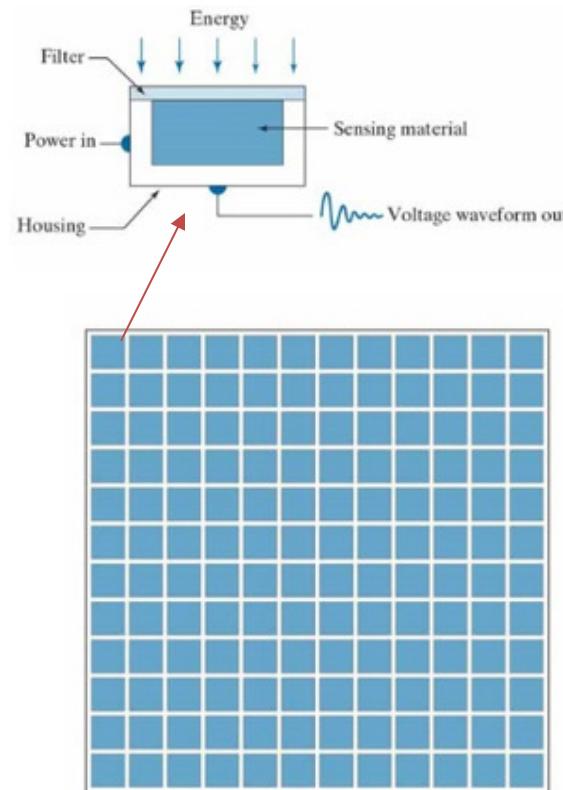
# Simple Image Formation Model

- (a) Illumination (energy) source. (b) A scene. (c) Imaging system. (d) Projection of the scene onto the image plane. (e) Digitized image.



# Simple Image Formation Model

- Array of sensors

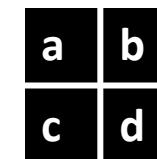
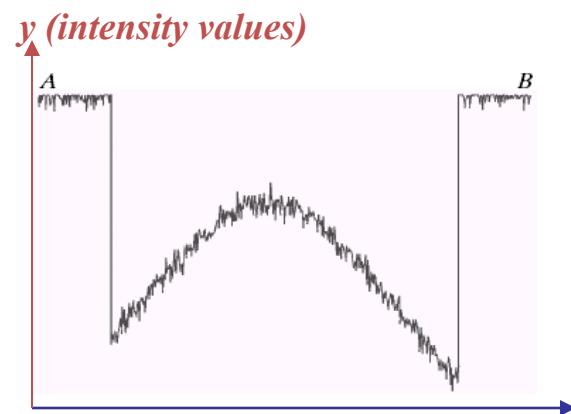
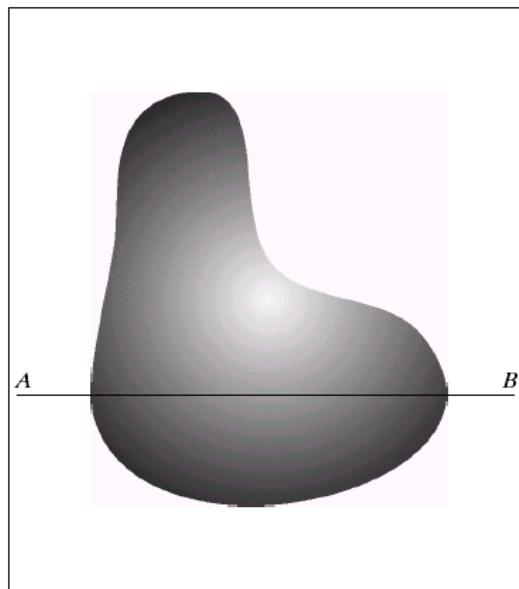


# Image Sampling and Quantization

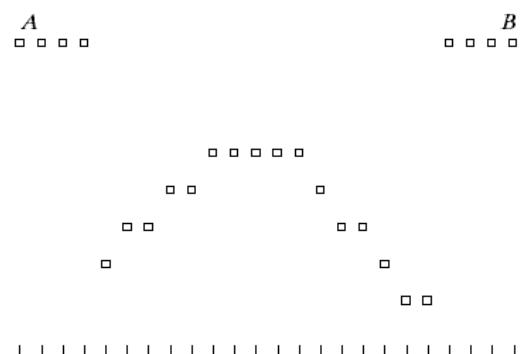
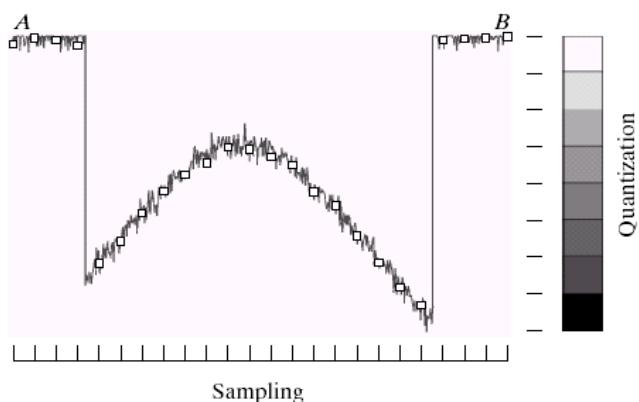
---

- To create a digital image, we need to convert the continuous sensed data into digital form. This process includes 2 processes:
  - **Sampling:** Digitizing the co-ordinate value is called sampling.
  - **Quantization:** Digitizing the amplitude value is called quantization.

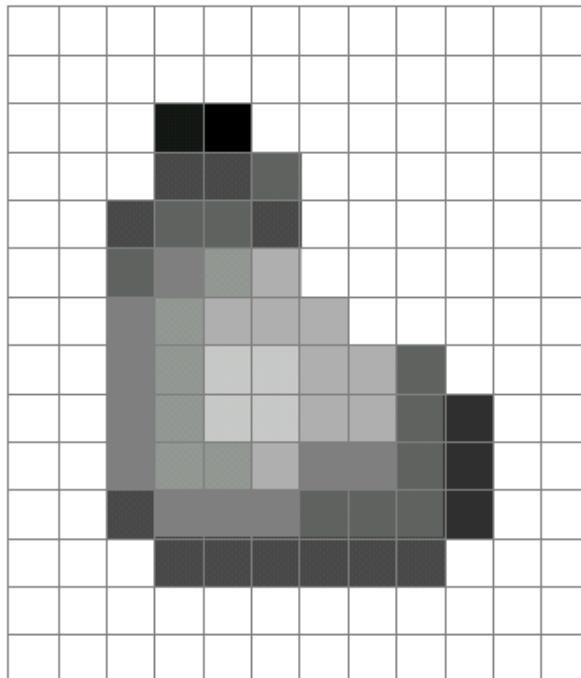
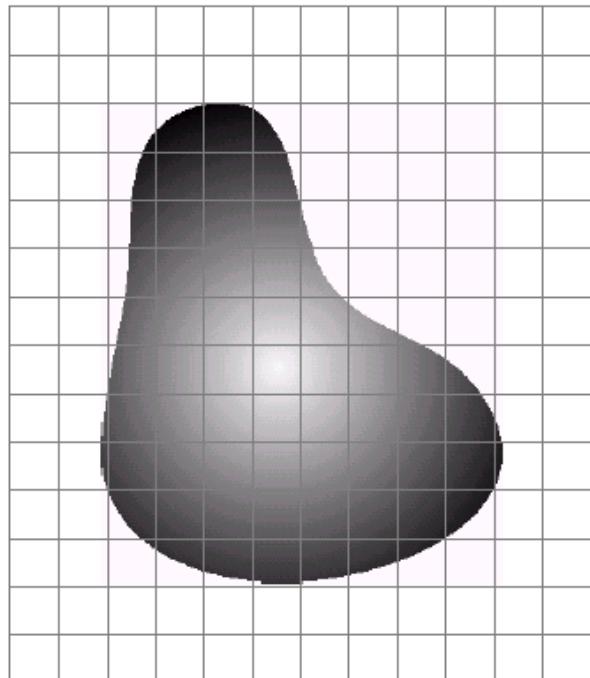
# Sampling and Quantization



Generating a digital image. (a) Continuous image. (b) A scaling line from A to B in the continuous image, used to illustrate the concepts of sampling and quantization. (c) sampling and quantization. (d) Digital scan line.

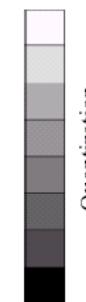


# Sampling and Quantization

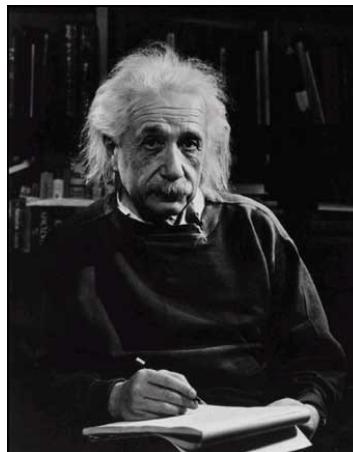


a | b

- (a) Continuous image projected onto a sensor array.  
(b) Result of image sampling and quantization.



# The Digital Image



What we see

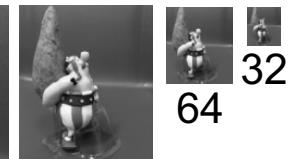
|   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|
| 0 | 3 | 2 | 5 | 4 | 7 | 6 | 9 | 8 |
| 3 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 2 | 1 | 0 | 3 | 2 | 5 | 4 | 7 | 6 |
| 5 | 2 | 3 | 0 | 1 | 2 | 3 | 4 | 5 |
| 4 | 3 | 2 | 1 | 0 | 3 | 2 | 5 | 4 |
| 7 | 4 | 5 | 2 | 3 | 0 | 1 | 2 | 3 |
| 6 | 5 | 4 | 3 | 2 | 1 | 0 | 3 | 2 |
| 9 | 6 | 7 | 4 | 5 | 2 | 3 | 0 | 1 |
| 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |

What a computer sees

**Digital image:** defined as a grid, as a 2D function  $f(x,y)$ ;  
 $x,y$  - spatial coordinates of a pixel;  $f$ : amplitude of any pairs  $(x,y)$  of coordinates and called **intensity** or grey level at that point.



# Image Sampling





# Image Quantization



8-bit



7-bit



6-bit



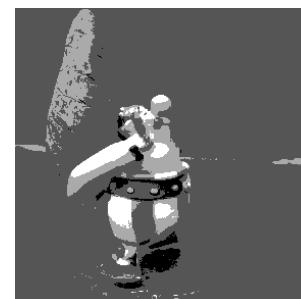
5-bit



4-bit



3-bit



2-bit



1-bit

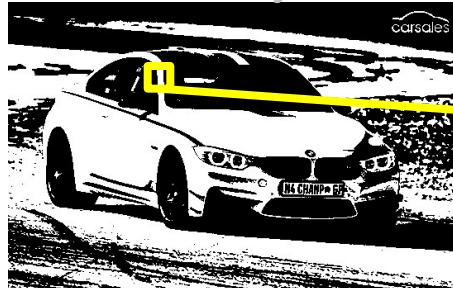
# Outline

---

- What is a digital image? **(Lecture 1)**
- **Different types of digital images (Lecture 1)**
- What is digital image processing? **(Lecture 2)**
  - Image/video compression
  - Image enhancement
  - Image understanding (computer vision)
- Summary of the main points

# Image Modalities (2D images)

Binary



|   |   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |

Gray



e.g., intensity values are 0-255

|     |     |     |     |     |     |     |     |     |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 125 | 14  | 124 | 123 | 220 | 111 | 234 | 185 | 231 |
| 84  | 110 | 151 | 245 | 255 | 0   | 151 | 15  | 156 |
| 151 | 121 | 180 | 95  | 74  | 122 | 152 | 151 | 150 |
| 151 | 156 | 30  | 35  | 80  | 95  | 135 | 140 | 189 |
| 125 | 14  | 124 | 123 | 220 | 111 | 234 | 185 | 231 |
| 151 | 121 | 180 | 95  | 74  | 122 | 152 | 151 | 150 |
| 84  | 110 | 151 | 245 | 255 | 0   | 151 | 15  | 156 |
| 151 | 121 | 180 | 95  | 74  | 122 | 152 | 151 | 150 |
| 125 | 14  | 124 | 123 | 220 | 111 | 234 | 185 | 231 |

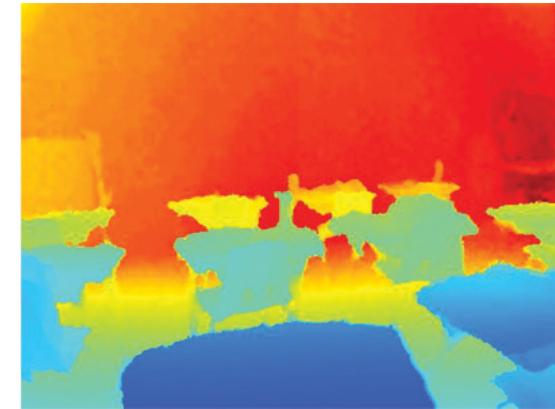
Color



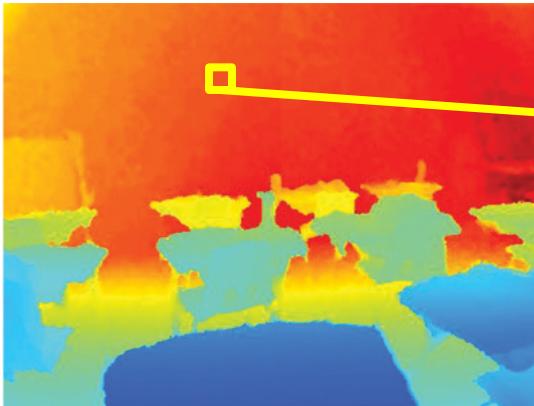
e.g., RGB color space

|     |     |     |     |     |     |     |     |     |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 125 | 14  | 124 | 123 | 220 | 111 | 234 | 185 | 231 |
| 84  | 110 | 151 | 245 | 255 | 0   | 151 | 15  | 156 |
| 151 | 121 | 180 | 95  | 74  | 122 | 152 | 151 | 150 |
| 151 | 156 | 30  | 35  | 80  | 95  | 135 | 140 | 189 |
| 125 | 14  | 124 | 123 | 220 | 111 | 234 | 185 | 231 |
| 151 | 121 | 180 | 95  | 74  | 122 | 152 | 151 | 150 |
| 84  | 110 | 151 | 245 | 255 | 0   | 151 | 15  | 156 |
| 151 | 121 | 180 | 95  | 74  | 122 | 152 | 151 | 150 |
| 125 | 14  | 124 | 123 | 220 | 111 | 234 | 185 | 231 |
| 84  | 110 | 151 | 245 | 255 | 0   | 151 | 15  | 156 |
| 151 | 121 | 180 | 95  | 74  | 122 | 152 | 151 | 150 |
| 125 | 14  | 124 | 123 | 220 | 111 | 234 | 185 | 231 |

# Image Modalities (Depth/2.5D images)



2.5D image (Depth map)

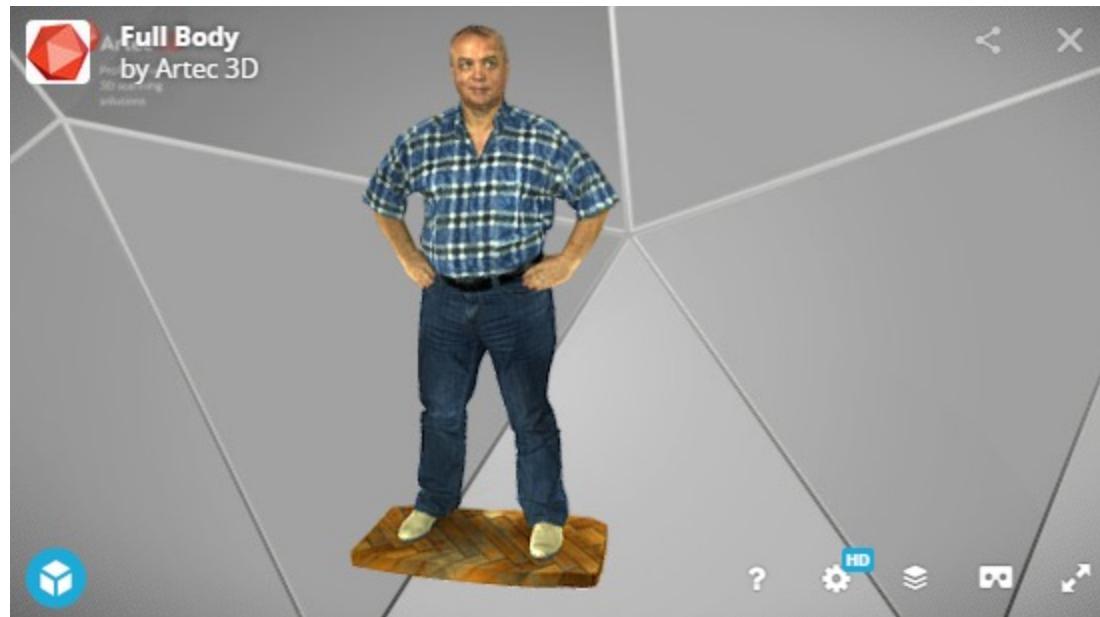


|     |     |     |     |     |     |     |     |     |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 125 | 14  | 124 | 123 | 220 | 111 | 234 | 185 | 231 |
| 84  | 110 | 151 | 245 | 255 | 0   | 151 | 15  | 156 |
| 151 | 121 | 180 | 95  | 74  | 122 | 152 | 151 | 150 |
| 151 | 156 | 30  | 35  | 80  | 95  | 135 | 140 | 189 |
| 125 | 14  | 124 | 123 | 220 | 111 | 234 | 185 | 231 |
| 151 | 121 | 180 | 95  | 74  | 122 | 152 | 151 | 150 |
| 84  | 110 | 151 | 245 | 255 | 0   | 151 | 15  | 156 |
| 151 | 121 | 180 | 95  | 74  | 122 | 152 | 151 | 150 |
| 125 | 14  | 124 | 123 | 220 | 111 | 234 | 185 | 231 |

Each pixel shows  
distance to the camera



# Image Modalities (3D images)



# SCC.366: Media Coding & Processing

2022-2023

Week 01 – Lecture 2

## Introduction to Media Coding and Processing

Dr. Hossein Rahmani

Senior Lecturer in Data Science

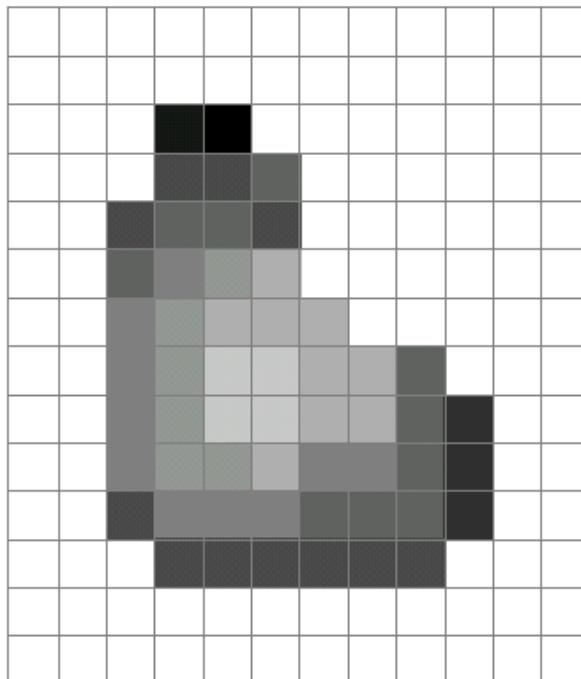
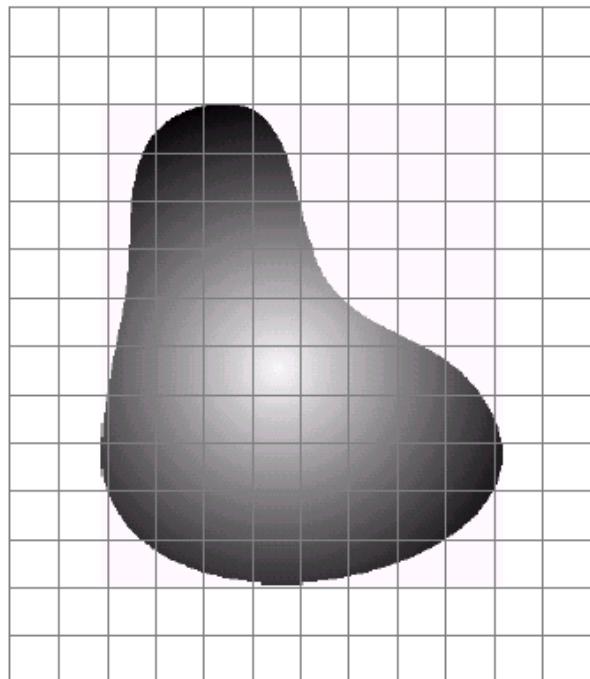
Email and Team: [h.rahmani@lancaster.ac.uk](mailto:h.rahmani@lancaster.ac.uk)

# Outline

---

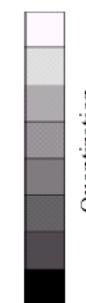
- **What is a digital image? (Lecture 1)**
- **Different types of digital images (Lecture 1)**
- What is digital image processing? (Lecture 2)
  - Image/video compression
  - Image enhancement
  - Image understanding (computer vision)
- Summary of the main points

# Review - Sampling and Quantization

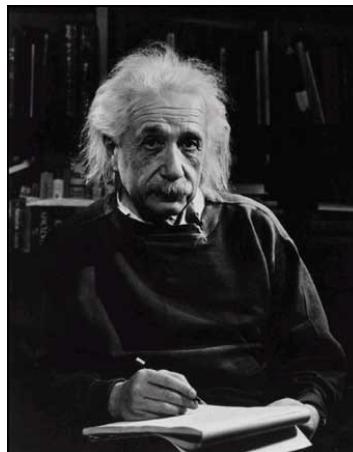


a | b

- (a) Continuous image projected onto a sensor array.  
(b) Result of image sampling and quantization.



# Review - The Digital Image



What we see

|   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|
| 0 | 3 | 2 | 5 | 4 | 7 | 6 | 9 | 8 |
| 3 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 2 | 1 | 0 | 3 | 2 | 5 | 4 | 7 | 6 |
| 5 | 2 | 3 | 0 | 1 | 2 | 3 | 4 | 5 |
| 4 | 3 | 2 | 1 | 0 | 3 | 2 | 5 | 4 |
| 7 | 4 | 5 | 2 | 3 | 0 | 1 | 2 | 3 |
| 6 | 5 | 4 | 3 | 2 | 1 | 0 | 3 | 2 |
| 9 | 6 | 7 | 4 | 5 | 2 | 3 | 0 | 1 |
| 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |

What a computer sees

**Digital image:** defined as a grid, as a 2D function  $f(x,y)$ ;  
 $x,y$  - spatial coordinates of a pixel;  $f$ : amplitude of any pairs  $(x,y)$  of coordinates and called **intensity** or grey level at that point.

# Outline

---

- What is a digital image? (Lecture 1)
- Different types of digital images (Lecture 1)
- **What is digital image processing? (Lecture 2)**
  - **Image/video compression**
  - Image enhancement
  - Image understanding (computer vision)
- Summary of the main points

# What is Image Processing?

---

**Image processing:** subclass of signal processing specifically concerned with images

## **Image processing areas:**

**Image compression:** concerned with how to decrease the *redundancy* of the image data in order to *store* or *transmit* data in an efficient form (lossy or lossless).

# What is image coding/compression?

- Image Coding (known as Image Compression) is the art/science of representing images with the least information (no. of bits) consistent with achieving an acceptable image quality.



$$\text{Compression ratio} = \frac{\text{Uncompressed size}}{\text{Compressed size}} = \frac{0.5\text{MByte}}{50\text{KByte}}$$

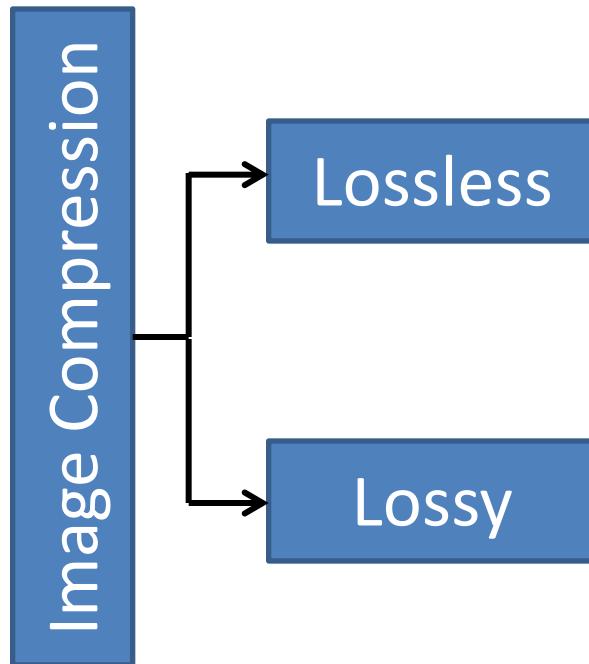


0.5 MBytes  
(Bitmap)

50 KBytes  
(JPEG)

# Lossless v.s. Lossy Compression

- The objective of image compression is to reduce redundant image data in order to be able to store or transmit data in an efficient form.



**Lossless** image compression is a compression algorithm that allows the original image to be **perfectly reconstructed** from the original data.

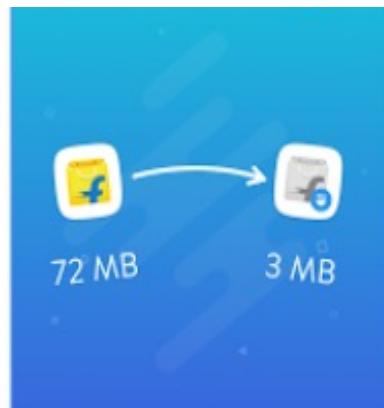
**Lossy** image compression is a type of compression where **a certain amount of information is discarded** which means that some data are lost and hence the image cannot be decompressed with 100% originality.

# Lossy & lossless coding

---

- **Lossless coding**
  - Decoded data are the exact copies of the originals
  - Low-compression ratio
  - Common for texts, executable programs, source codes, spreadsheets, etc.
- **Lossy coding**
  - Decoded data are approximations of the originals
  - High-compression ratio
  - Common for image, video, audio, etc.
  - Users' perception is the main objective

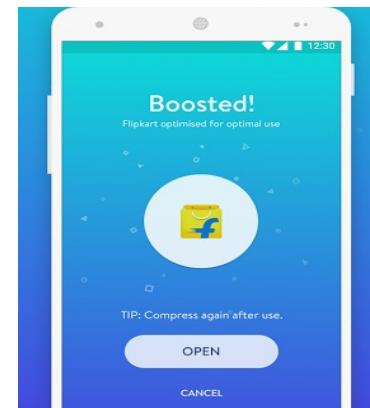
# Why do we code images?



More Files



More Applications



Less Network Usage

Providing **the same** service:

Less Required bandwidth/connection speed, less required memory, less consumed power, smaller equipment size, etc

## Why do we code images? (cnt'd)

---

Enabling technology for image **storage** and distribution

- Digital broadcasting
- Video conferencing
- Medical imaging
- DVD, YouTube, etc.

# Image coding standards

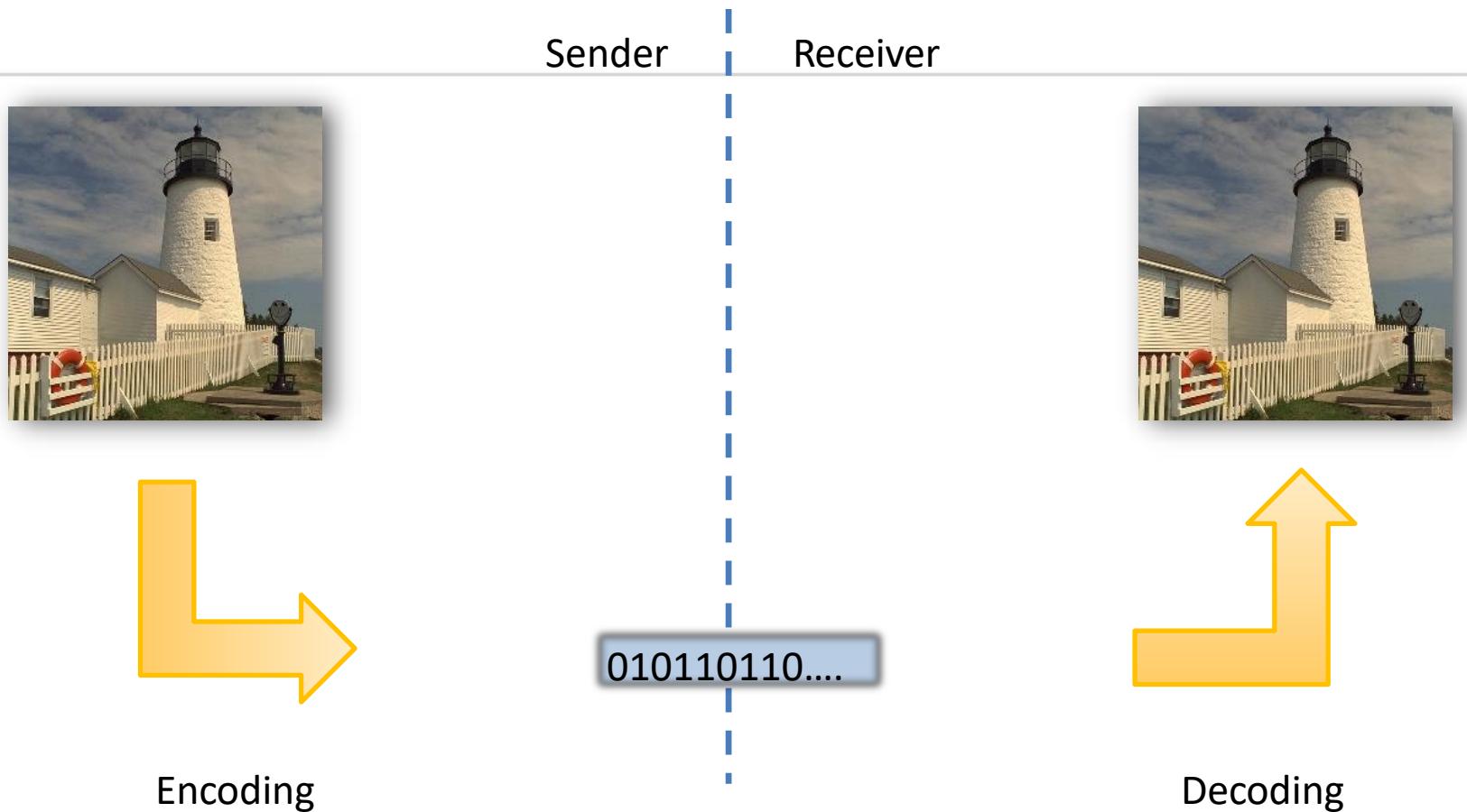


Image coding is **standardized**: the sender and the receiver agree on data format.

# Image coding standards

---

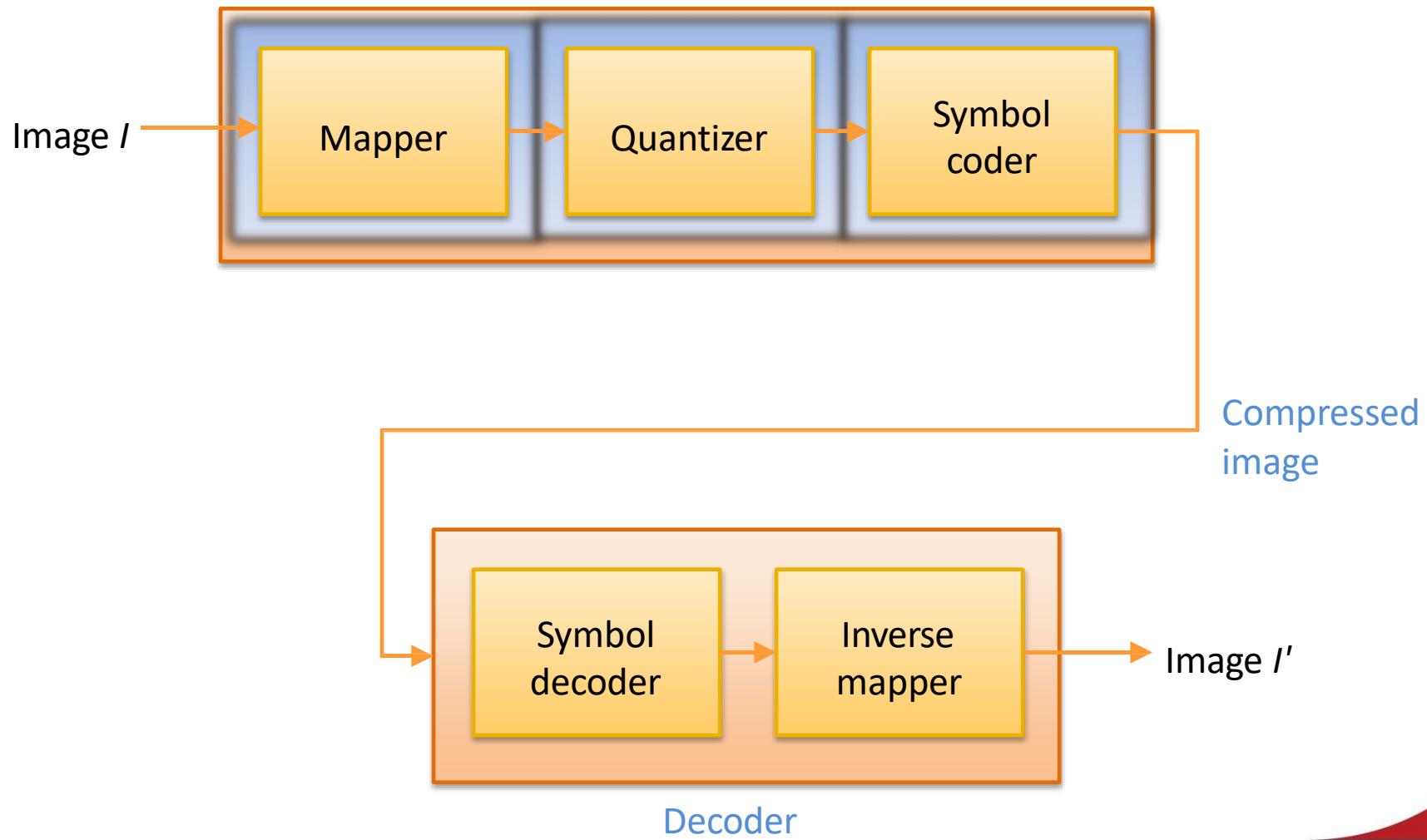
- JPEG (Joint Photographic Expert Group): lossy
- JPEG 2000: lossy
- GIF (Graphics Interchange Format): lossless
- PNG (Portable Network Graphics): lossless
- etc.

## Image coding criteria: 'good' coding scheme

---

- ✓ Compression ratio
- ✓ Compression & decompression complexity (speed)
- ✓ Implementation easiness
- ✓ Progressive decoding
- ✓ Scalability
- ✓ ...

# Image coding (and decoding) System



# What you will see in the coming weeks?

---

- In this module, we will discuss the following topics:
  - Different coding algorithms such as Huffman coding, Run-Length coding,
  - How JPEG compression works?
  - How this concept can be extended to video compression?
  - Data hiding and watermarking in both spatial and frequency domains

# Outline

---

- What is a digital image? (Lecture 1)
- Different types of digital images (Lecture 1)
- What is digital image processing? (**Lecture 2**)
  - Image/video compression
  - **Image enhancement**
  - Image understanding (computer vision)
- Summary of the main points

# What is Image Processing?

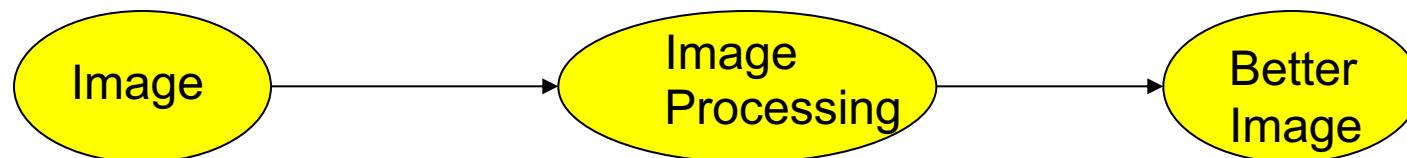
**Image processing:** subclass of signal processing specifically concerned with pictures

## Image processing areas:

**Image compression:** concerned with how to decrease the **redundancy** of the image data in order to **store** or **transmit** data in an efficient form (lossy or lossless).

**Image enhancement** aims to:

1. **improve human perception** of information in images, or
2. **provide 'better' input** for other **automated** image processing techniques
  - reduce noise** in an image and reveal features in the image.



# Image Enhancement (Intensity Corrections)



Very poor contrast



Improved contrast

# Image Enhancement (Improving sharpness)



Original image



Improved sharpness and  
balanced tonality



# Image Enhancement (Image restoration)



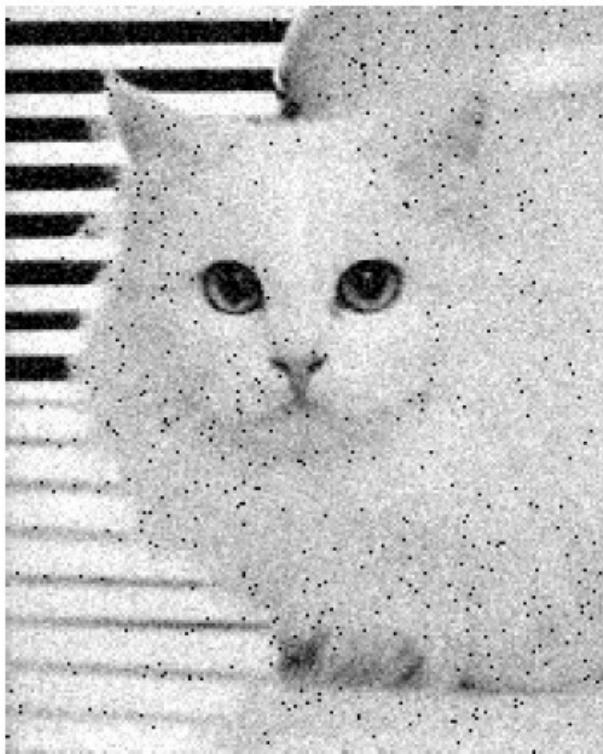
Blurred image



Restored image by inverse filtering

# Image Enhancement (Noise removal)

---



Noisy image



Noise removal filtering

# What you will see in the coming weeks?

---

- In this module, we will discuss the following topics:
  - Image enhancement in both spatial and frequency domains, e.g., intensity transformation, histogram matching and equalization, spatial filtering, Discrete Cosine Transformation etc.
  - Extracting useful information, such as edges from images that can be used for understanding image semantic and image enhancement
  - Image Pyramids
  - Noise removal
  - Etc.

# Outline

---

- What is a digital image? (Lecture 1)
- Different types of digital images (Lecture 1)
- What is digital image processing? (**Lecture 2**)
  - Image/video compression
  - Image enhancement
  - **Image understanding (computer vision)**
- Summary of the main points

# What is Image Processing?

---

**Image processing:** subclass of signal processing specifically concerned with pictures

## Image processing areas:

**Image compression:** concerned with how to decrease the **redundancy** of the image data in order to **store** or **transmit** data in an efficient form (lossy or lossless).

**Image enhancement** aims to:

1. **improve human perception** of information in images, or
2. **provide 'better' input** for other **automated** image processing techniques

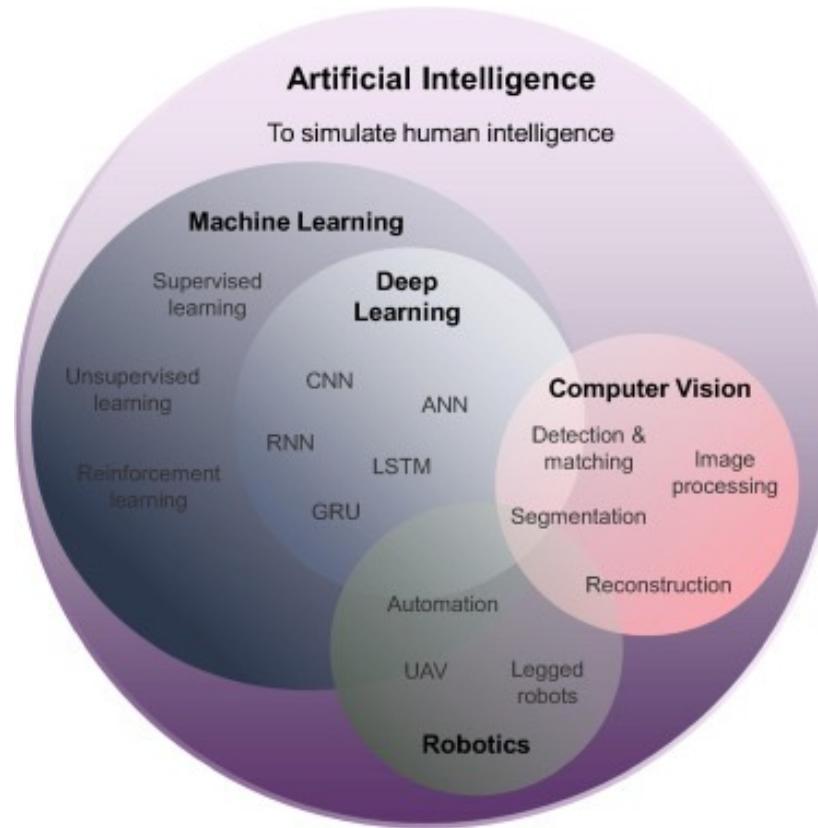
▪**reduce noise** in an image and reveal features in the image.

**Image understanding (Computer Vision):** the process of **interpreting** regions, objects in the image, to figure out what actually happens in the image.

Understand the content of an image, extract **meaningful information** from images

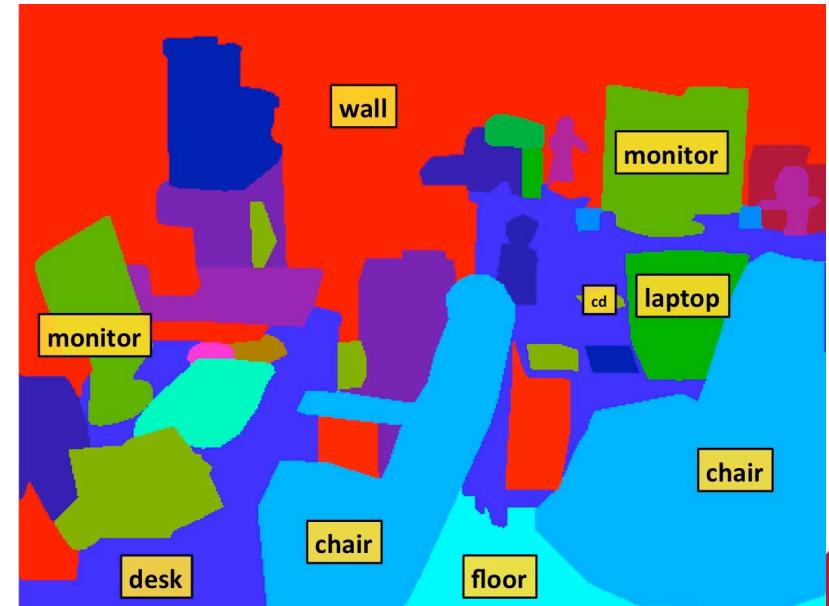
# What is Computer Vision?

**Computer Vision** works on enabling computers to “see” and “understand” images in the same way that human vision does, and then provide appropriate output.



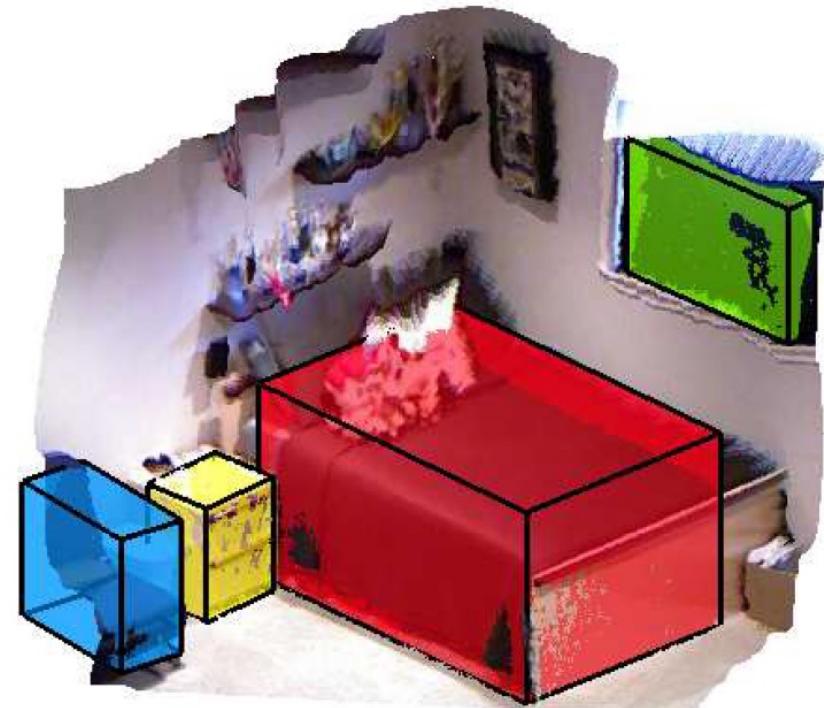
# What does it mean to “see” and “understand”?

- To know what is where by looking



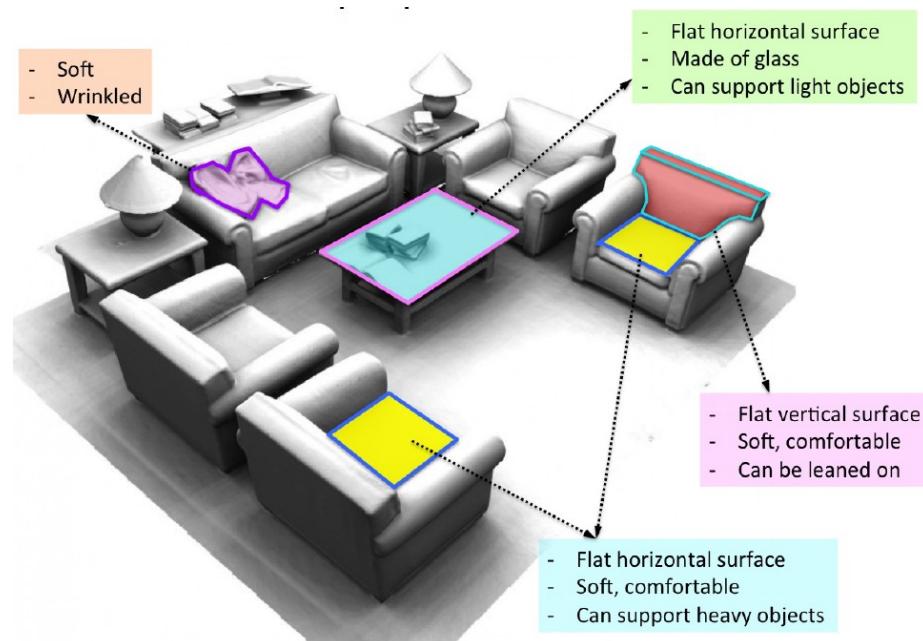
# What does it mean to “see” and “understand”?

- To know what is where by looking
- Understand where things are in the world



# What does it mean to “see” and “understand”?

- To know what is where by looking
- Understand where things are in the world
- What are their 3D properties



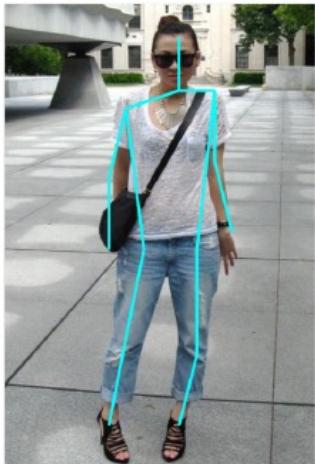
# What does it mean to “see” and “understand”?

- To know what is where by looking
- Understand where things are in the world
- What are their 3D properties
- What actions are taking place?



# Applications of Computer Vision

- Figure out what people are wearing



**Paper Doll Parsing**

Upload a JPG file or type in a JPG image URL to try our clothing parser.

No file chosen

Or, you may try one of the following images.



[About this project](#)

<http://clothingparsing.com>



# Applications of Computer Vision

- DeepFake: a form of artificial intelligence called deep learning to make images of fake events





# Applications of Computer Vision

- Make-A-Video (just released): A state-of-the-art system that generates videos from text



<https://makeavideo.studio/>

# Applications of Computer Vision

- Generate image captions automatically

## Tags

- feminist
- sleepwear
- pregnancy
- mommies
- lesbianism

## Nearest Caption in the Training Dataset

a little girl carries a younger girl on the sidewalk .

## Generated Captions

- a girl is standing outside with her hand on the ground .
- a little boy and young girl sit on the street .
- a girl in a blue shirt holds a small child on the sidewalk .
- a young girl is holding a cup on the ground .
- two girls in the street holding her hands .



# Applications of Computer Vision

- You can make yourself look better (and competitors worse)

↓ attractive   original   ↑ attractive   ↓ friendly   original   ↑ friendly



↓ age

original

↑ age



# What you will see in the coming weeks?

---

- In this module, we will discuss the following topics:
  - Image matching
  - Global and local feature extraction
  - Image segmentation
  - Video segmentation
  - Motion analysis such as optical flow
  - Etc.

# Summary

---

- **Image compression:** concerned with how we can reduce the redundancy of the image data in order to store or transmit data in an efficient form (lossy or lossless).
- **Image enhancement:** aims to improve human perception of information in images, or to provide 'better' image for automated image processing systems. Reduce noise in an image and reveal features in the image. Enhancement alters an image to makes it clearer to human observers.
- **Image understanding:** the process of interpreting the regions, objects in the image, to figure out what actually happens in the image. Understanding usually attempts to mimic the human visual system in extracting meaning from an image

# SCC.366: Media Coding & Processing

2022-2023

Week 02 – Lecture 1

## Image Transformation - Intensity Transformation

Dr. Hossein Rahmani

Senior Lecturer in Data Science

Email and Team: [h.rahmani@lancaster.ac.uk](mailto:h.rahmani@lancaster.ac.uk)

# Outline

---

- Image types (Binary, Grey scale, color)
- Processing images
  - Spatial vs. Frequency domain
  - Intensity transformations vs. Spatial filtering
- Basic intensity transformations
  - Image negatives,
  - Log transformations,
  - Power-law or Gamma transformations
  - Piecewise linear transformations
  - Bit-plane slicing
- Image histogram
  - Histogram equalization,
  - Histogram matching (or histogram specification)

# Review - What is Image Processing?

---

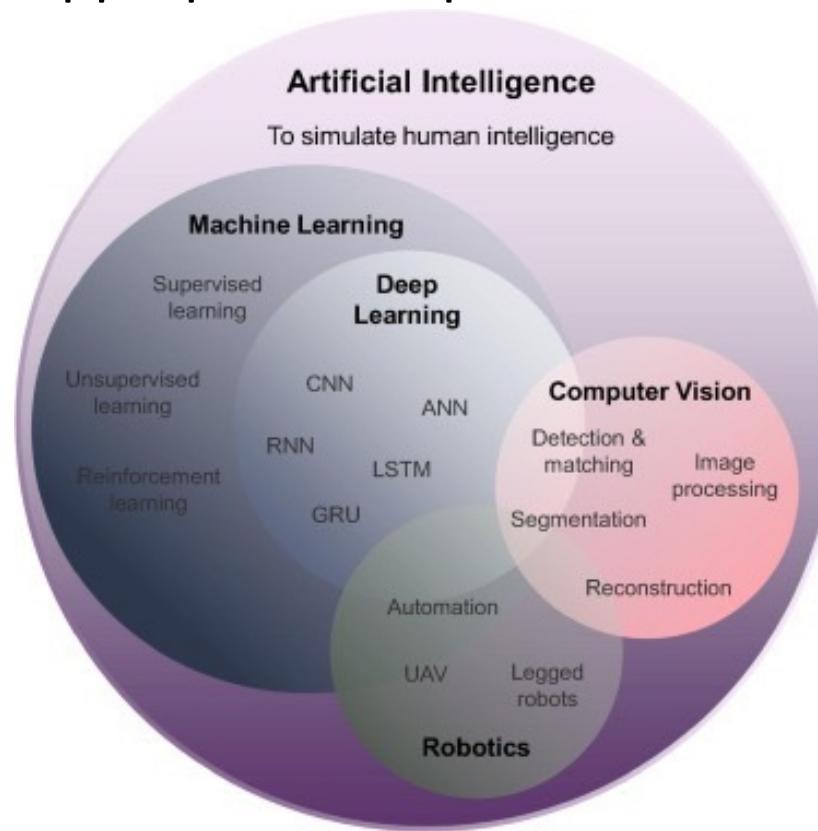
**Image processing:** subclass of signal processing specifically concerned with pictures

## Image processing areas:

- **Image compression:** concerned with how to decrease the *redundancy* of the image data in order to *store* or *transmit* data in an efficient form (lossy or lossless).
- **Image enhancement** aims to:
  1. improve human perception of information in images, or
  2. provide 'better' input for other automated image processing techniques
- **Image understanding (Computer Vision):** the process of interpreting regions, objects in the image, to figure out what actually happens in the image.

# Review - What is Computer Vision?

**Computer Vision** works on enabling computers to “see” and “understand” images in the same way that human vision does, and then provide appropriate output.



# Why is Vision Hard?

---

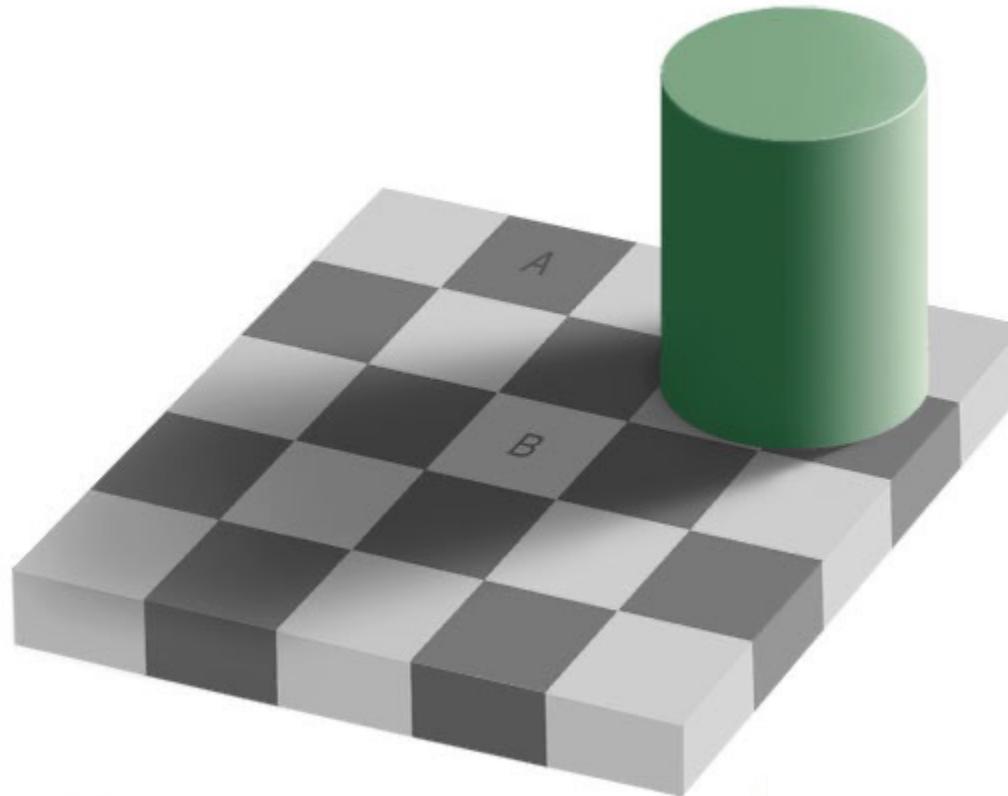
- Human vision seems to work quite well.
- How well does it really work?
- **Let's play some games!**



# Why is Vision Hard?

---

- Which square is lighter, A or B?

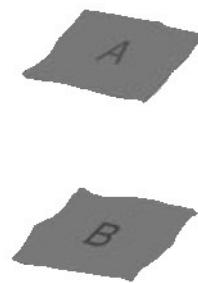




# Why is Vision Hard?

---

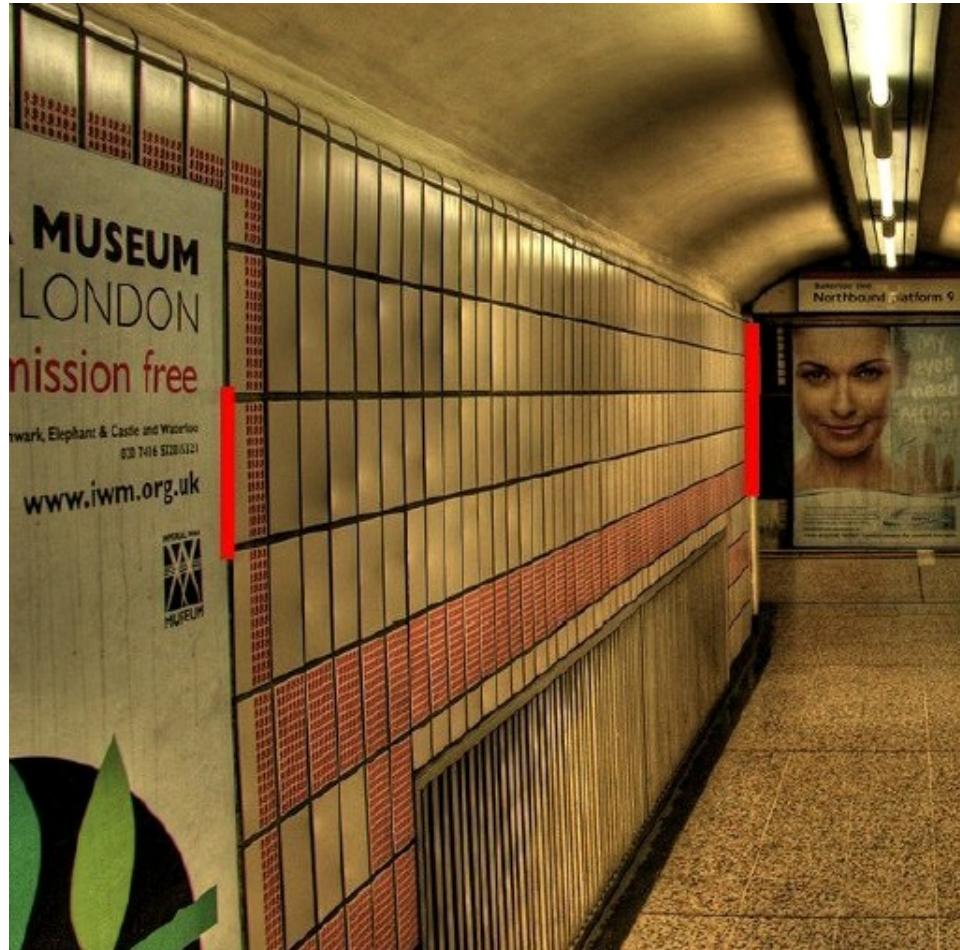
- Which square is lighter, A or B?





# Why is Vision Hard?

- Which red line is longer?





# Why is Vision Hard?

---

- Which red line is longer?



# Outline

---

- **Image types (Binary, Grey scale, color)**
- Processing images
  - Spatial vs. Frequency domain
  - Intensity transformations vs. Spatial filtering
- Basic intensity transformations
  - Image negatives,
  - Log transformations,
  - Power-law or Gamma transformations
  - Piecewise linear transformations
  - Bit-plane slicing
- Image histogram
  - Histogram equalization,
  - Histogram matching (or histogram specification)

# Fundamentals of Digital Images



- An image is represented by a grid of **picture elements** called **pixels**.
  - **Pixels are arranged in columns and rows**
  - **Spatial coordinate:**  $(x,y)$  for 2D case such as photograph,  
 $(x,y,z)$  for 3D case such as Computer Tomography scan images
- 11  
**How to present movies? (x,y,t)**

# White and Black Images

---

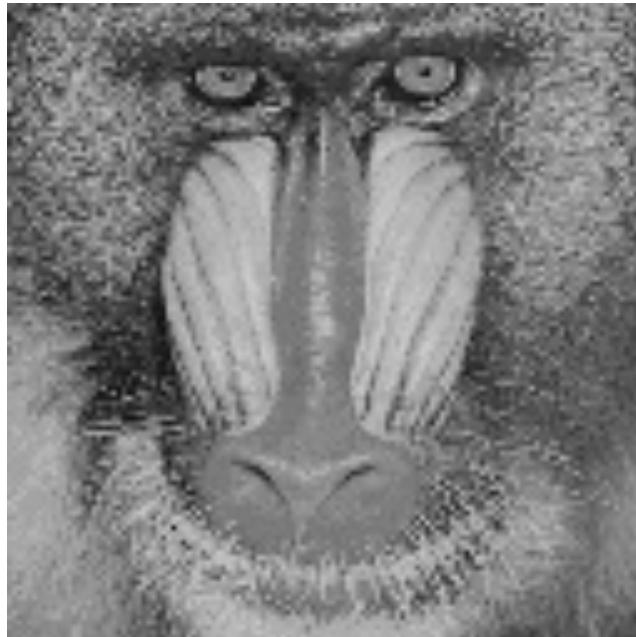


- White and black (binary) image is a matrix of 0s and 1s
- A pixel color is represented by 1 bit only

**Value 0 = Black**

**Value 1 = White**

# Grey Scale (Intensity) Images



- Grey scale image
  - Pixels are scalars
  - Typically  $2^8 = 256$  grey tones are used to make up the image, i.e. 8 bits in the range [0,255]

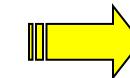
If a pixel color is represented by **2 bits**:

- Value 00 = Black
- Value 01 = Grey
- Value 10 = Light Grey
- Value 11 = White

# Color (RGB) Images



**Color RGB image:**  
each pixel contains a vector  
representing red, green and  
blue components.



RGB components

|    |    |    |    |    |    |
|----|----|----|----|----|----|
| 10 | 10 | 16 | 28 |    |    |
| 9  | 65 | 70 | 56 | 43 |    |
| 15 | 32 | 99 | 70 | 56 | 78 |
| 32 | 21 | 60 | 90 | 96 | 67 |
|    | 54 | 85 | 85 | 43 | 92 |
|    |    | 32 | 65 | 87 | 99 |
|    |    |    |    | 14 |    |

# Color (RGB) Images

---

- **Bit depth of an RGB image:** # of bits used to represent a pixel in the RGB space
- Each R, G, & B image = 8-bit image
  - depth =  $3 \times 8 = 24$  bits => “full-color” image
- What is the total number of colors?  
**Hint:** The total number of shades of red is :  $2^8 = 256$ 
  - total # of colors =  $(2^8)^3 = 16777216$

# Color Models

- RGB Model
  - Colour value of a pixel = sum of the intensities of the colour components **red**, **green** and **blue**
    - Maximum intensity of all three components results in a white pixel
- $Y C_b C_r$  model
  - $Y$  represents the value of the luminance
  - $C_b$  and  $C_r$  are two orthogonal colour vectors.
- HSV (hue, saturation, value) model
  - or also known as HSB (hue, saturation, brightness)
- The colour value of a pixel can be easily converted from model to model

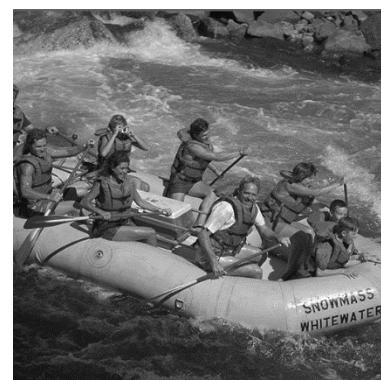
# Color Model ( $Y \ C_b \ C_r$ )

- $Y \ C_b \ C_r$  model
  - $Y$  represents the value of the luminance
  - $C_b$  and  $C_r$  are two orthogonal colour vectors.
- Advantage of  $Y \ C_b \ C_r$  Model in image processing
  - Value of the luminance is directly available
    - grey-scale version of the image can be created very easily

$$Y = 0.299R + 0.587G + 0.114B$$

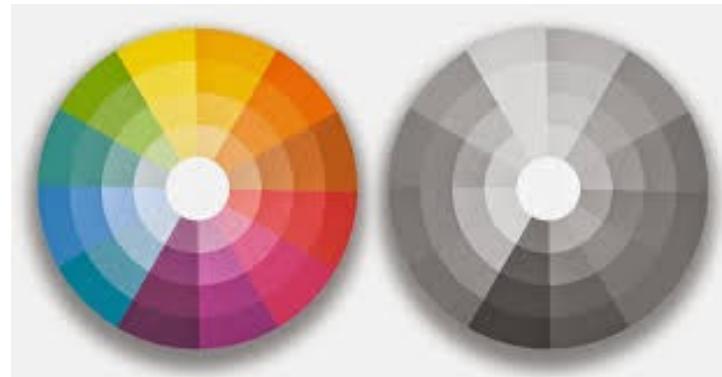
$$C_b = -0.168R - 0.331G + 0.500B + 128$$

$$C_r = 0.500R - 0.419G - 0.081B + 128$$



# Color to Grayscale

- Colour values R, G and B contribute differently to grayscale impression
  - Green contributes most, then red, then blue
  - Conversion
    - R, G and B can be converted to one pixel grayscale value through:  
$$\text{gray-value} = 0.299 \text{ R} + 0.587 \text{ G} + 0.114 \text{ B}$$





# Color to Binary

- The simplest binarization technique is called thresholding
  - Convert RGB image to grayscale
  - Set pixels above/below a given threshold to either black or white



# Outline

---

- Image types (Binary, Grey scale, color)
- Processing images
  - **Spatial vs. Frequency domain**
  - Intensity transformations vs. Spatial filtering
- Basic intensity transformations
  - Image negatives,
  - Log transformations,
  - Power-law or Gamma transformations
  - Piecewise linear transformations
  - Bit-plane slicing
- Image histogram
  - Histogram equalization,
  - Histogram matching (or histogram specification)

# Some Definitions

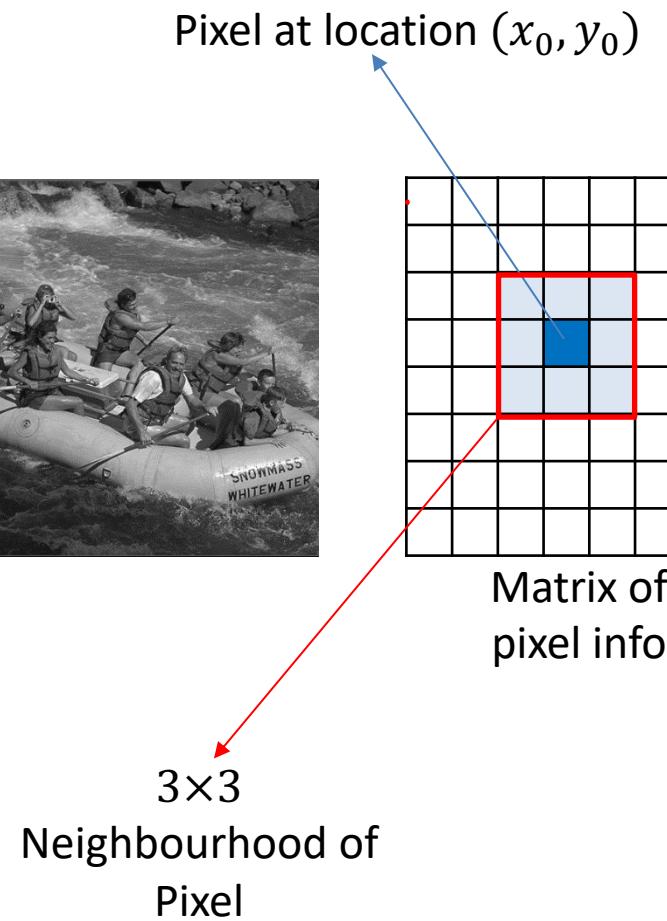
---

- **Spatial domain**
  - The image plane itself
  - Processing in the spatial domain deals with direct manipulation of the image pixels
- **Frequency domain**
  - Consider the image as a 2D signal and apply Fourier or DCT transform
  - Process the image in the transformed domain
  - Apply the inverse transform to return into the spatial domain.

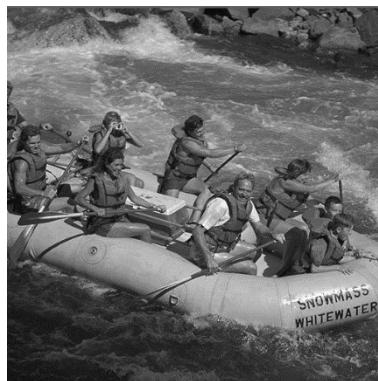
# Spatial Domain

$$g(x, y) = T[f(x, y)]$$

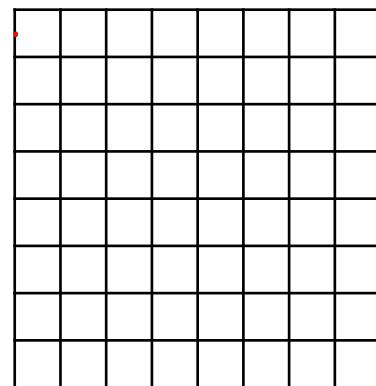
- In the discrete case
  - $f$  and  $g$  are 2D matrices of size  $W \times H$
  - $T$  is a matrix of size  $w \times h$  which is much smaller than  $W \times H$
  - The value of each pixel of  $g$  is the result of applying the operator  $T$  at the corresponding location in  $f$ .



# Frequency Domain (DCT/DFT)

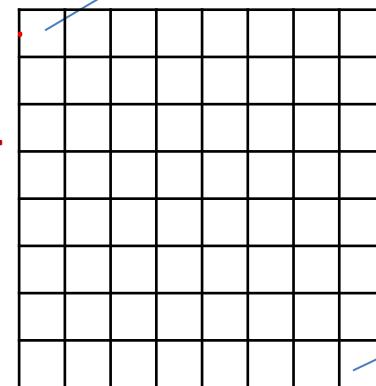


All components in the spatial matrix are equally important



Matrix of  
pixel info

DCT  
➡



Matrix of  
frequency info

Lowest frequency

Highest frequency

- DCT/DFT transforms the spatial matrix to a new space where components have different meaning
- The image processing takes place in the frequency domain
- Its inverse is called “inverse DCT” or “iDCT”

# Outline

---

- Image types (Binary, Grey scale, color)
- Processing images
  - Spatial vs. Frequency domains
  - **Intensity transformations vs. Spatial filtering**
- Basic intensity transformations
  - Image negatives,
  - Log transformations,
  - Power-law or Gamma transformations
  - Piecewise linear transformations
  - Bit-plane slicing
- Image histogram
  - Histogram equalization,
  - Histogram matching (or histogram specification)

# Some Definitions

---

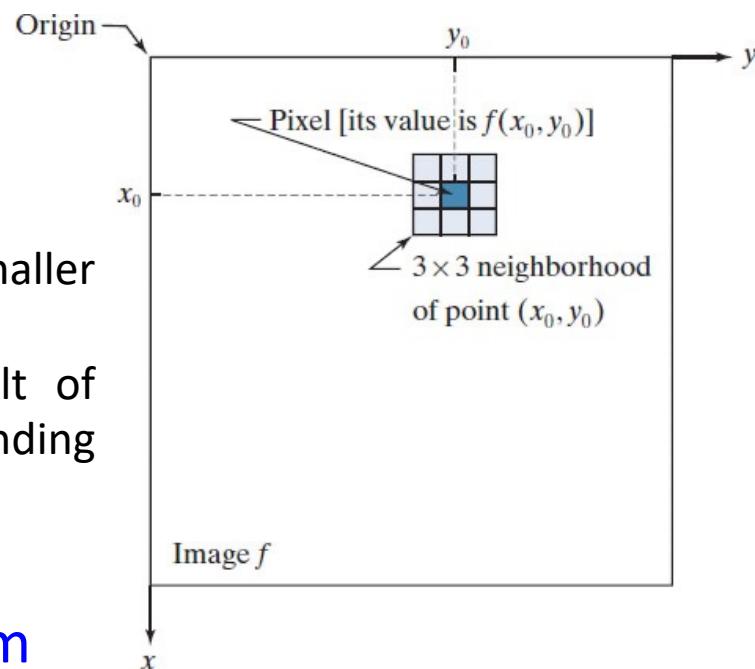
- **Spatial domain:**
  - The image plane itself
  - Processing in the spatial domain deals with direct manipulation of the image pixels
- Types of transformations that can be applied to an image
  - **Intensity transformations** (Operate on single pixels) **Week 2**
  - **Spatial filtering** (Working on a neighbourhood of every pixel) **Week 3**

# Basics

- **Intensity transformation and spatial filtering**

$$g(x, y) = T[f(x, y)]$$

- In the discrete case
  - $f$  and  $g$  are 2D matrices of size  $W \times H$
  - $T$  is a matrix of size  $w \times h$  which is much smaller than  $W \times H$
  - The value of each pixel of  $g$  is the result of applying the operator  $T$  at the corresponding location in  $f$ .
- If  $w \times h = 1 \times 1$  then intensity transform
- Otherwise, it's spatial filtering



# Outline

---

- Image types (Binary, Grey scale, color)
- Processing images
  - Spatial vs. Frequency domain
  - Intensity transformations vs. Spatial filtering
- **Basic intensity transformations**
  - Image negatives,
  - Log transformations,
  - Power-law or Gamma transformations
  - Piecewise linear transformations
  - Bit-plane slicing
- Image histogram
  - Histogram equalization,
  - Histogram matching (or histogram specification)

# Basic Intensity Transforms

---

- Definition
  - $g(x, y) = T[f(x, y)]$
- Examples
  - Image negatives
  - Log transformations
  - Power-law (Gamma) transformations
  - Piecewise linear transformations
  - Bit-plane slicing
- In the following
  - Each image is represented with a 2D matrix of L intensity levels. That is each pixel takes a value between 0 and L-1.
  - For example, L=256 for a standard grayscale image

# Basic intensity transforms

- **Image negatives**

- Reversing the intensity levels:  $g(x, y) = L - 1 - f(x, y)$
- It produces the equivalent of a photographic negatives
- What is the application of this transformation?



Input image

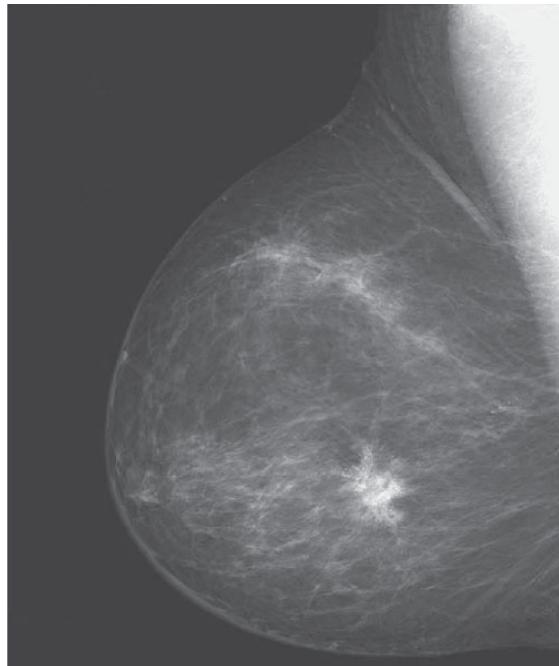


Its negative



# Basic intensity transforms

- **Image negatives**
  - Can be used to enhance white or gray details embedded in dark regions especially when the black areas are dominant in size



Input image



Its negative

- What we have seen
  - Limitations of Human Visual System
  - Image processing can be done in spatial domain or frequency domain
  - Spatial methods can be divided into two categories:
    - Intensity Transformation
    - Spatial Filtering
  - Intensity transformations applied to individual pixels
    - E.g., image negatives

# SCC.366: Media Coding & Processing

2022-2023

Week 02 – Lecture 2

## Intensity Transformation Cont.

Dr. Hossein Rahmani

Senior Lecturer in Data Science

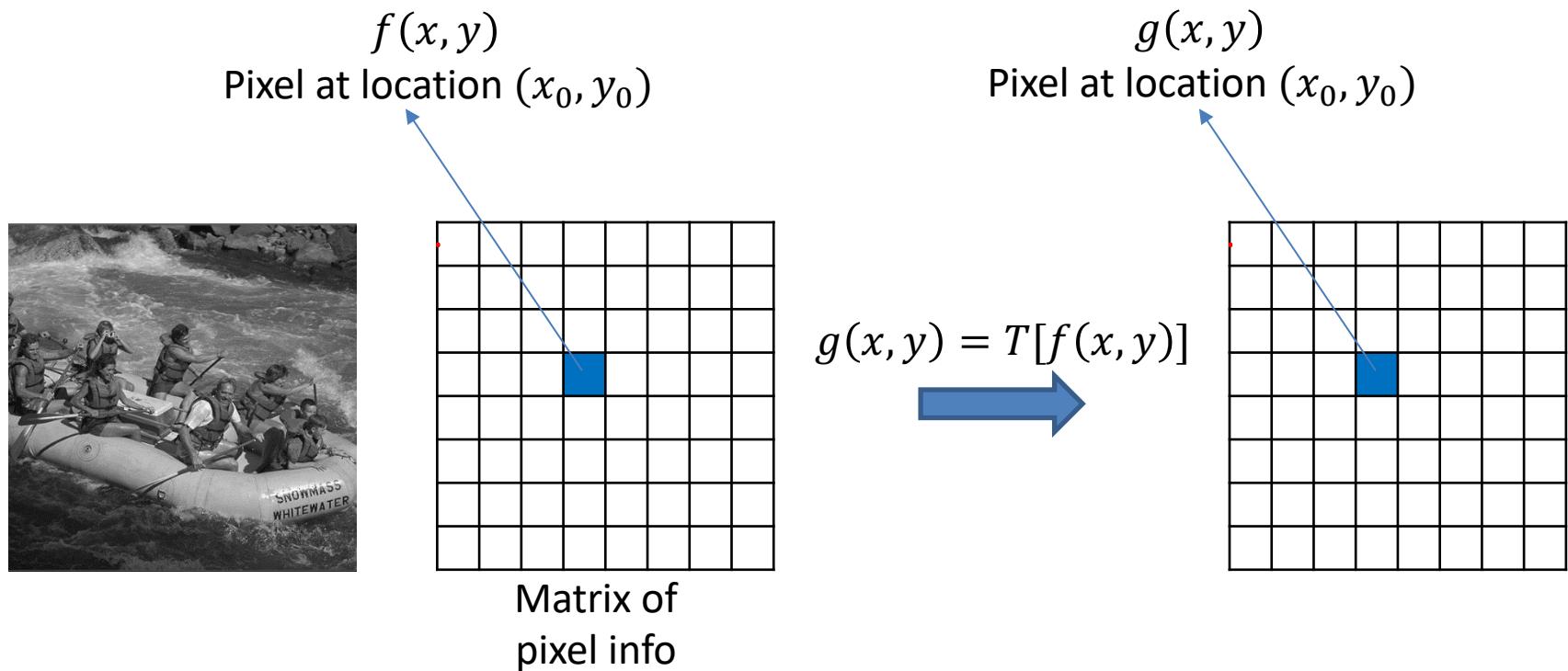
Email and Team: [h.rahmani@lancaster.ac.uk](mailto:h.rahmani@lancaster.ac.uk)

# Outline

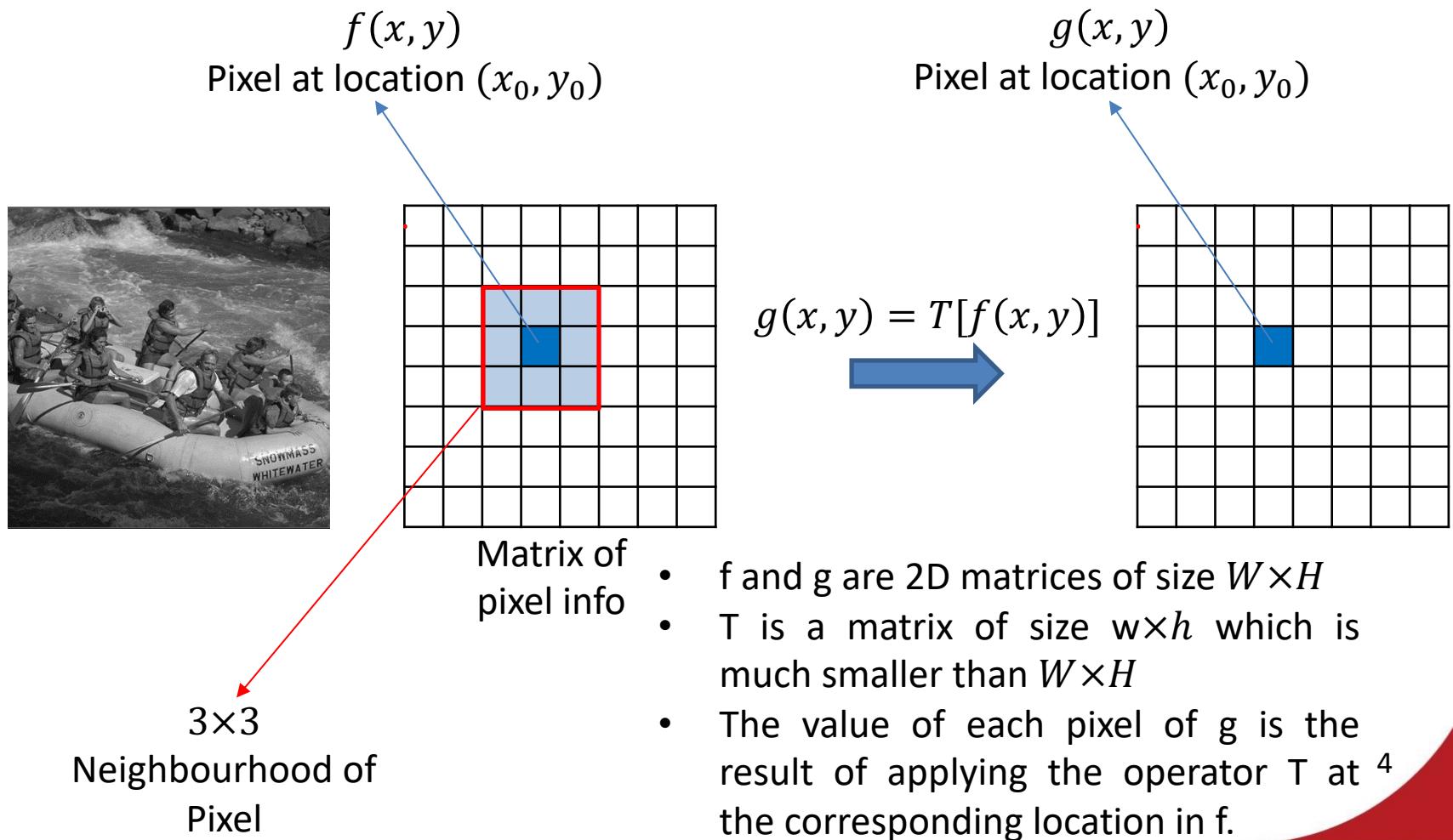
---

- Image types (Binary, Grey scale, color)
- Processing images
  - Spatial vs. Frequency domain
  - Intensity transformations vs. Spatial filtering
- **Basic intensity transformations**
  - Image negatives,
  - Log transformations,
  - Power-law or Gamma transformations
  - Piecewise linear transformations
  - Bit-plane slicing
- Image histogram
  - Histogram equalization,
  - Histogram matching (or histogram specification)

# Review - Spatial Domain (Intensity Transformation)



# Review - Spatial Domain (Spatial Filtering)



# Basic Intensity Transforms

---

- Definition
  - $g(x, y) = T[f(x, y)]$
- Examples
  - Image negatives
  - Log transformations
  - Power-law (Gamma) transformations
  - Piecewise linear transformations
  - Bit-plane slicing
- In the following
  - Each image is represented with a 2D matrix of L intensity levels. That is each pixel takes a value between 0 and L-1.
  - For example, L=256 for a standard grayscale image

# Basic intensity transforms

- **Image negatives**

- Reversing the intensity levels:  $g(x, y) = L - 1 - f(x, y)$
- It produces the equivalent of a photographic negatives
- What is the application of this transformation?

Maximum Intensity value  
e.g., 255 for 8-bit gray image



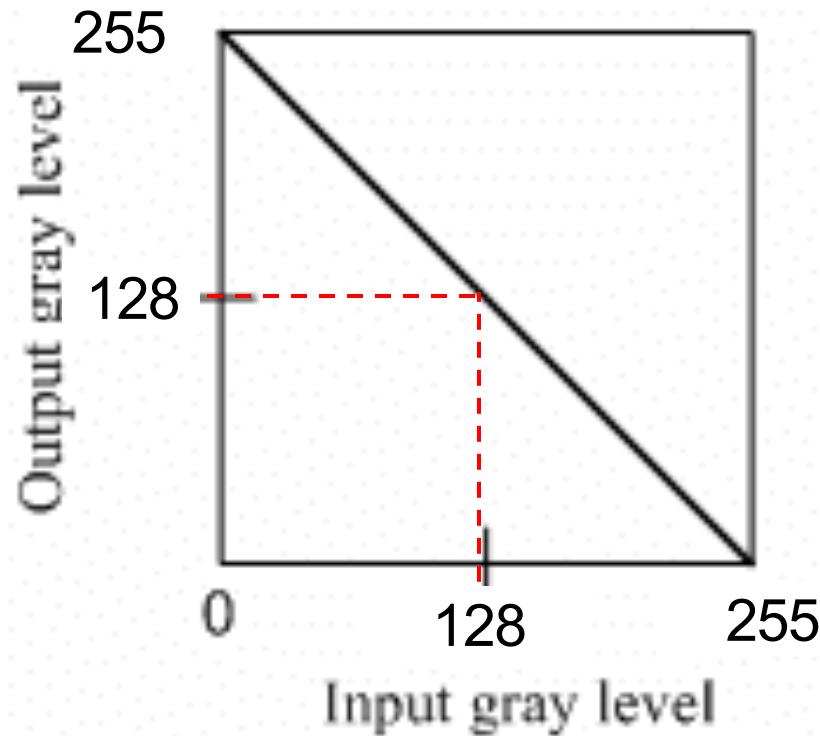
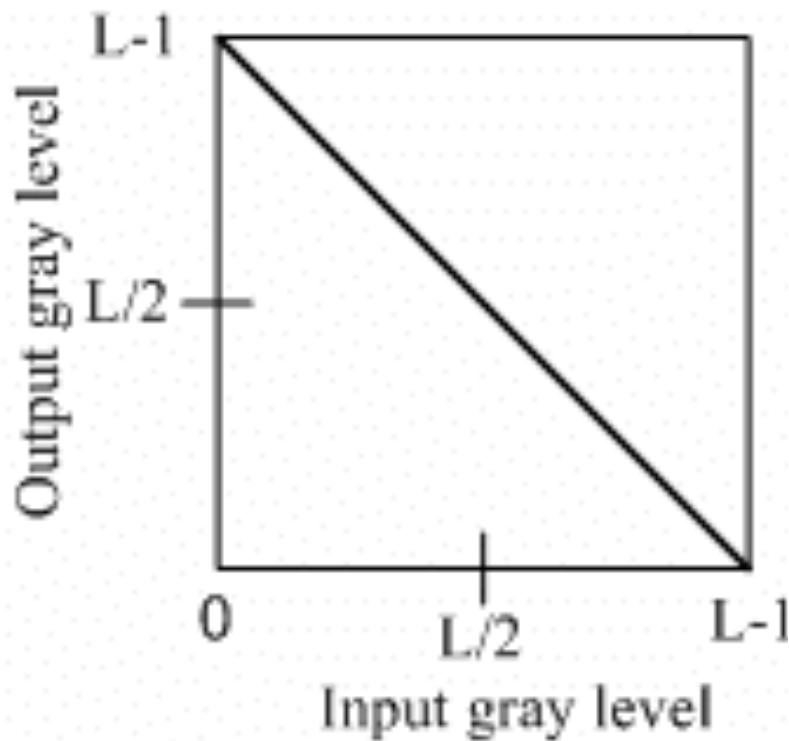
Input image



Its negative

# Basic intensity transforms

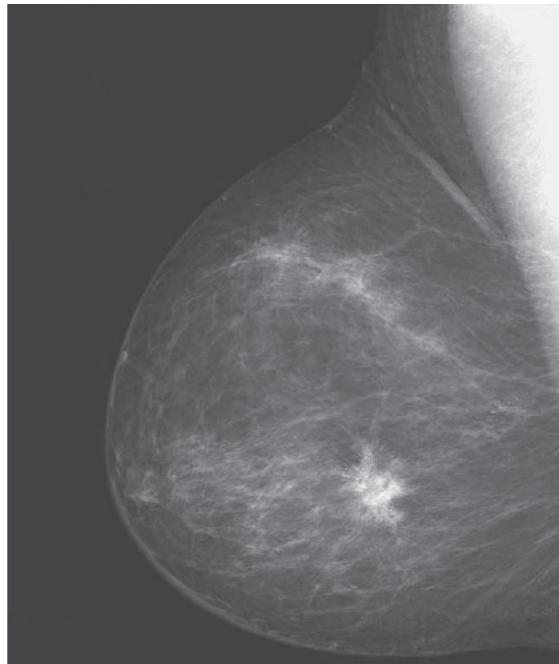
- **Image negatives**
  - Negative transformation function





# Basic intensity transforms

- **Image negatives**
  - Can be used to enhance white or gray details embedded in dark regions especially when the black areas are dominant in size



Input image

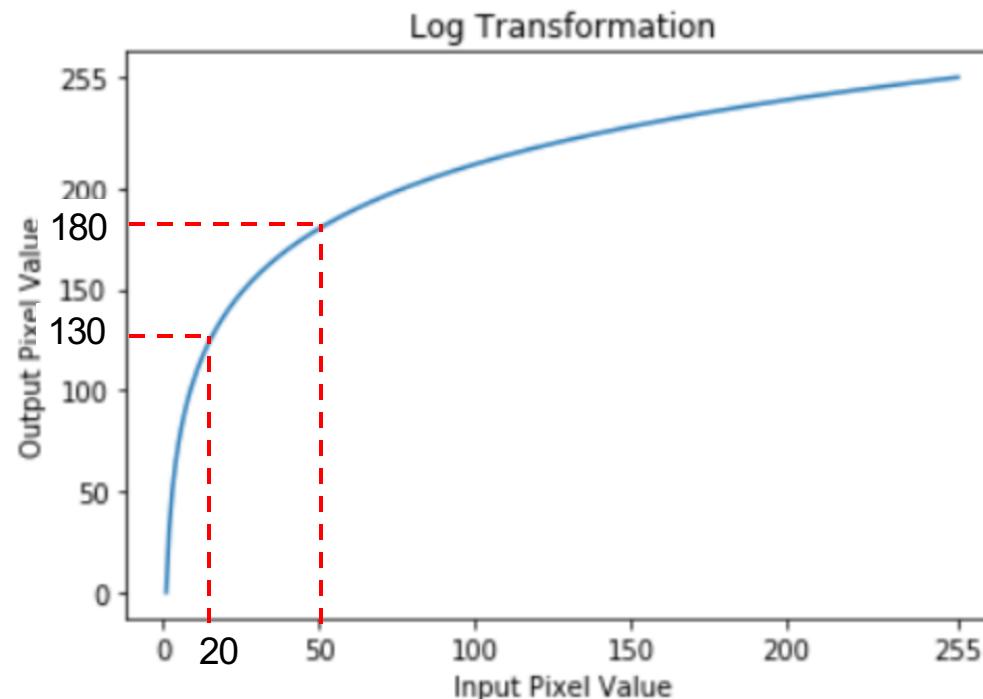


Its negative

# Basic intensity transforms

- **Log transformation**

- Expands the dark pixels in the image while compressing the brighter pixels
- Increases the **dynamic range** of low intensities
- **Dynamic range** refers to the ratio of max and min intensity values



# Basic intensity transforms

- **Log transformation**

- When the dynamic range of the image (e.g., values are between 0 and 10,000,000) is greater than that of displaying device (e.g., 0-255), the lower intensity values are suppressed.

|     |     |     |     |           |     |     |     |     |
|-----|-----|-----|-----|-----------|-----|-----|-----|-----|
| 20  | 403 | 120 | 67  | 23        | 6   | 79  | 403 | 0   |
| 132 | 999 | 5   | 245 | 80        | 34  | 222 | 10  | 8   |
| 255 | 0   | 43  | 5   | 0         | 89  | 5   | 0   | 67  |
| 0   | 34  | 0   | 8   | 1,000,000 | 212 | 9   | 34  | 245 |
| 7   | 89  | 89  | 12  | 329       | 6   | 19  | 89  | 5   |
| 302 | 212 | 3   | 100 | 1         | 5   | 18  | 212 | 9   |
| 98  | 6   | 187 | 320 | 100       | 67  | 134 | 6   | 3   |
| 43  | 5   | 45  | 410 | 76        | 245 | 7   | 5   | 187 |

# Basic intensity transforms

- **Log transformation**

- When the dynamic range of the image (e.g., values are between 0 and 10,000,000) is greater than that of displaying device (e.g., 0-255), the lower intensity values are suppressed.

|   |   |   |   |     |   |   |   |   |
|---|---|---|---|-----|---|---|---|---|
| 0 | 0 | 0 | 0 | 0   | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0   | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0   | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 255 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0   | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0   | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0   | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0   | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0   | 0 | 0 | 0 | 0 |

# Basic intensity transforms

---

- **Log transformation**

- When the dynamic range of the image (e.g., values are between 0 and 10,000,000) is greater than that of displaying device (e.g., 0-255), the lower intensity values are suppressed.
- For such images the limited dynamic range of the display will generally just display the most intense pixels and the remainder of the display will essentially be black (0 level)
- To overcome this issue, **we can use log transform to enhance lower intensity values**, and thus the image shows **significantly more details**.

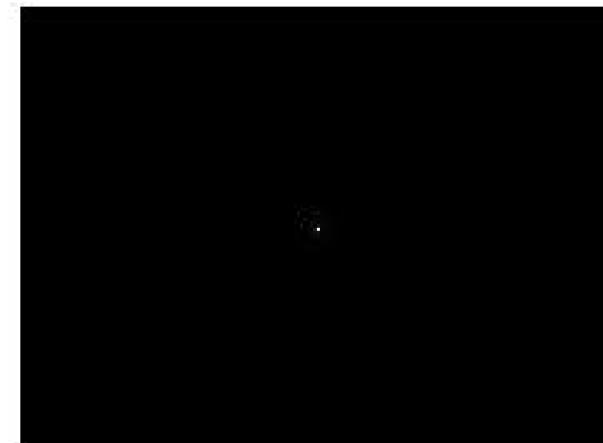
# Basic intensity transforms

- **Log transformations**

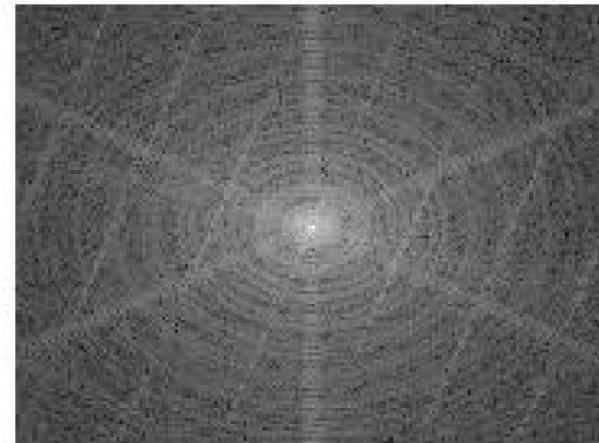
- $$g(x, y) = c \log(1 + f(x, y)).$$

Original intensity value

$c$  is constant and  $f(x, y) \geq 0$



Input image  
Pixel values range from 0 to  $10^6$



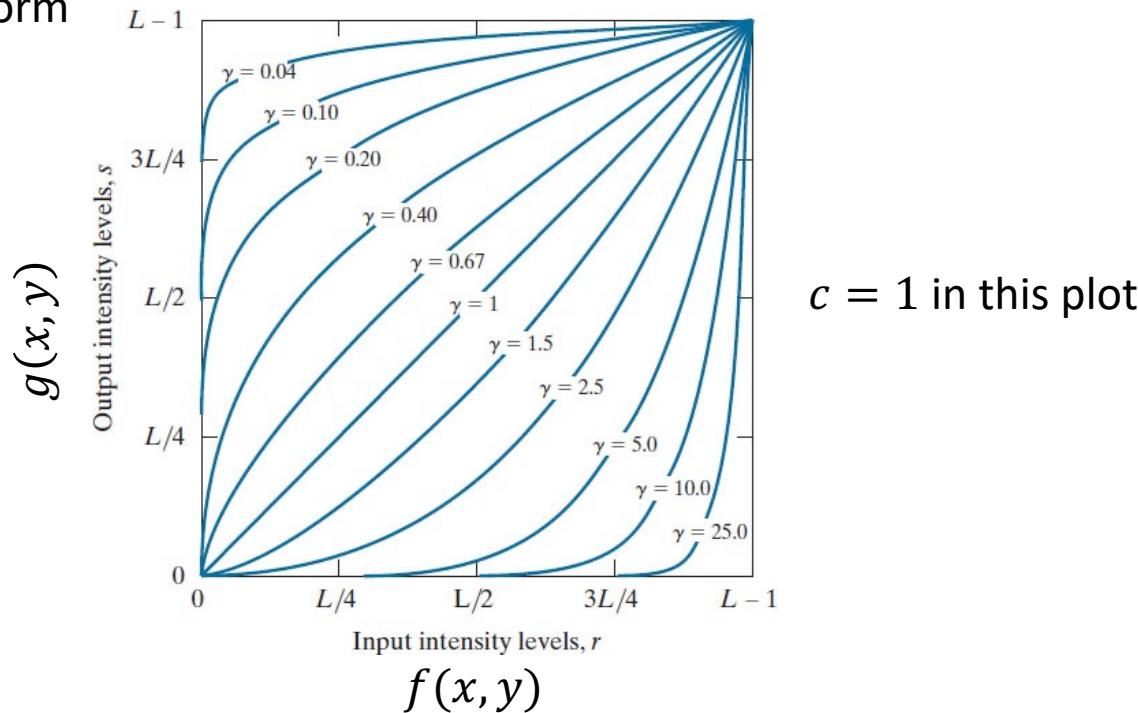
Its log transform with  $c = 1$   
Values range from 0 to 6.2

# Basic intensity transforms

- **Power-law (Gamma) transformation**

$$\text{New intensity value} \leftarrow g(x, y) = cf(x, y)^\gamma$$

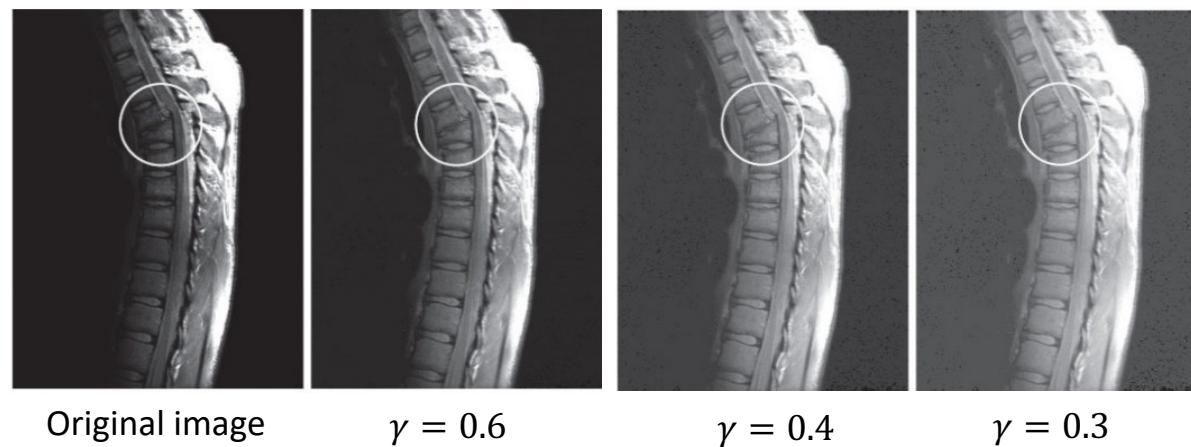
- $c$  and  $\gamma$  are positive constants
- A family of different transformation curves obtained by varying  $\gamma$ , for example log transform



# Basic intensity transforms

- **Power-law (Gamma) transformation**

- A family of different transformation curves obtained by varying  $\gamma$ , for example log transform
- Original image and results of applying Gamma transformations with  $c = 1$  and  $\gamma = 0.6, 0.4, 0.3$  (increasing the dynamic range of low intensities)



# Basic intensity transforms

- **Power-law (Gamma) transformation**

- A family of different transformation curves obtained by varying  $\gamma$ , for example **inverse log** transform
- Original image and results of applying Gamma transformations with  $c = 1$  and  $\gamma = 3, 4, 5$  (**increasing the dynamic range of high intensities**)



Original image



$\gamma = 3$



$\gamma = 4$

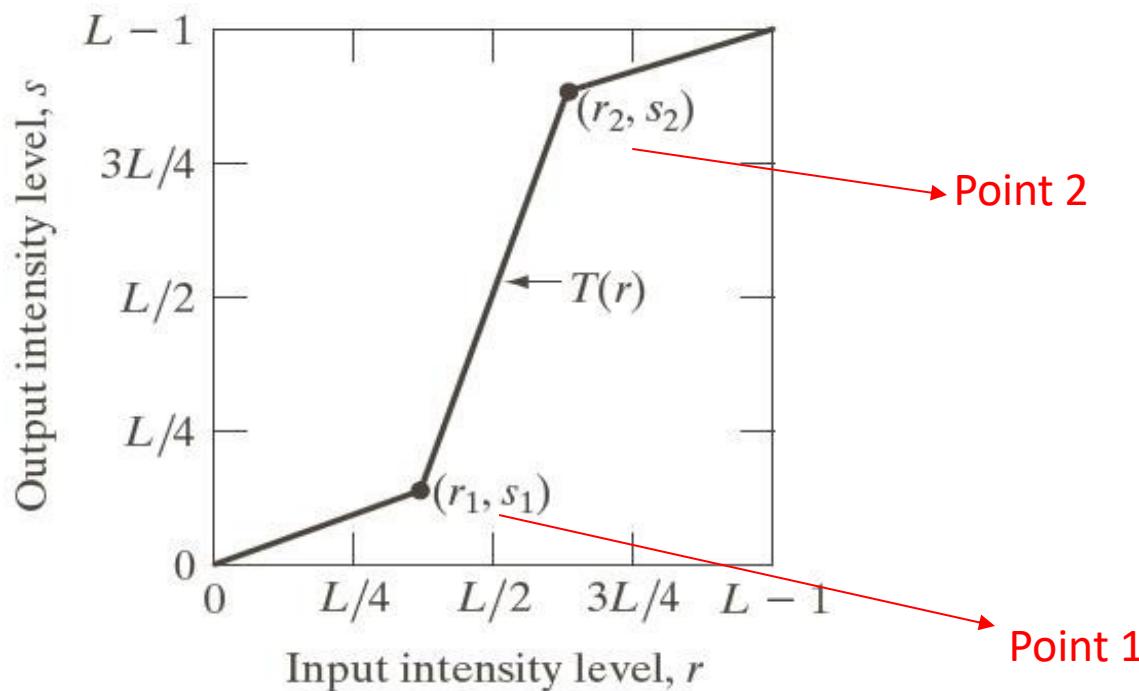


$\gamma = 5$

# Basic intensity transforms (CW1-Q1)

- **Piecewise linear transformation**

- Piecewise linear functions can be arbitrary complex
- Thus, more flexibility in the design of the transformation





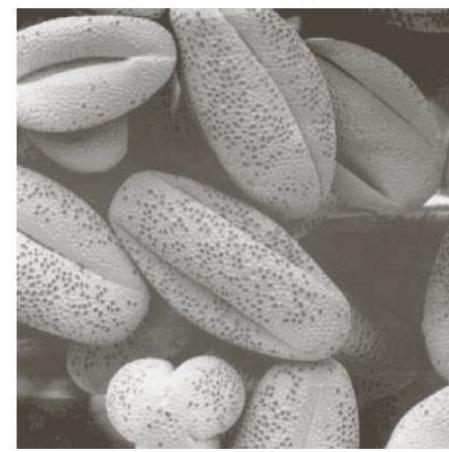
# Basic intensity transforms

- **Piecewise linear transformations**

- Contrast stretching
  - Low contrast images can result from poor illumination



Low contrast image



Contrast stretching

**Contrast:** the **range** of intensity values effectively used within a given image.

# Basic intensity transforms

- **Piecewise linear transformations**

- Contrast stretching
  - Low contrast images can result from poor illumination



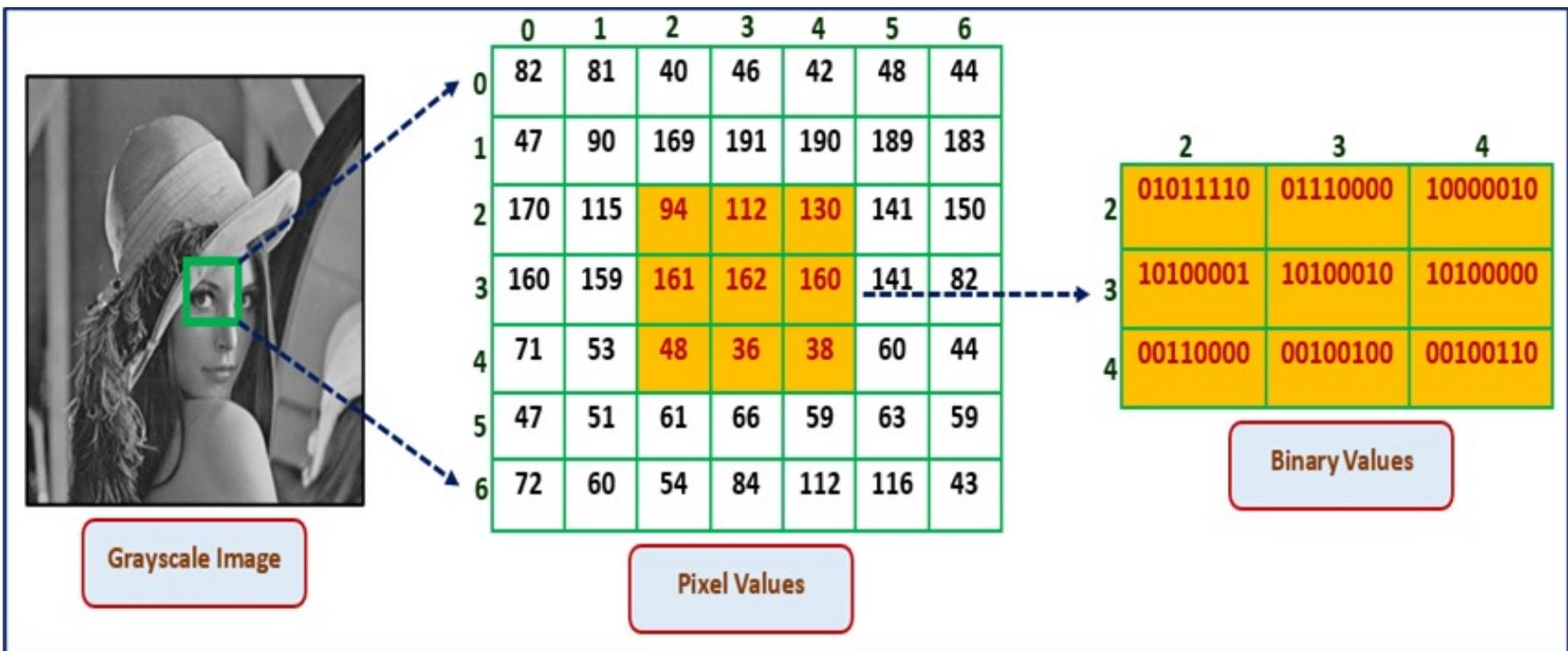
Low contrast image



Contrast stretching

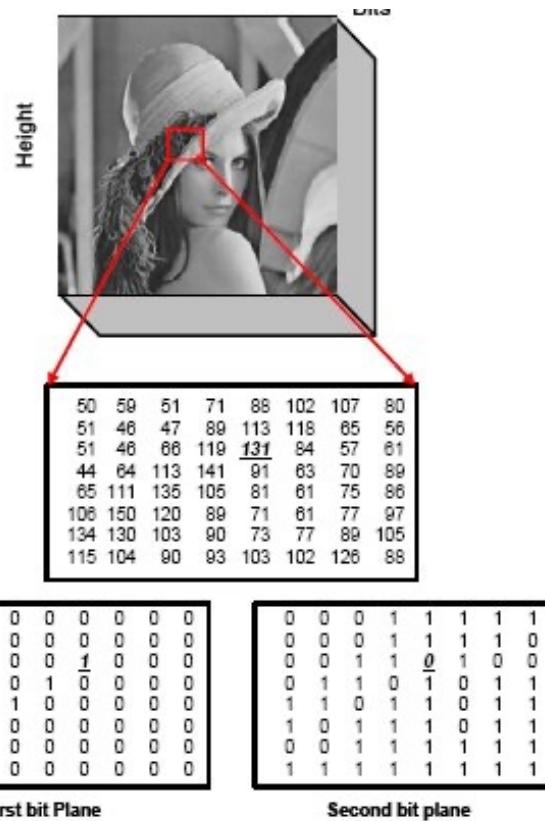
# Basic intensity transforms

- **Bit-plane slicing**
  - Each pixel in a 256-level grayscale image is composed of 8 bits



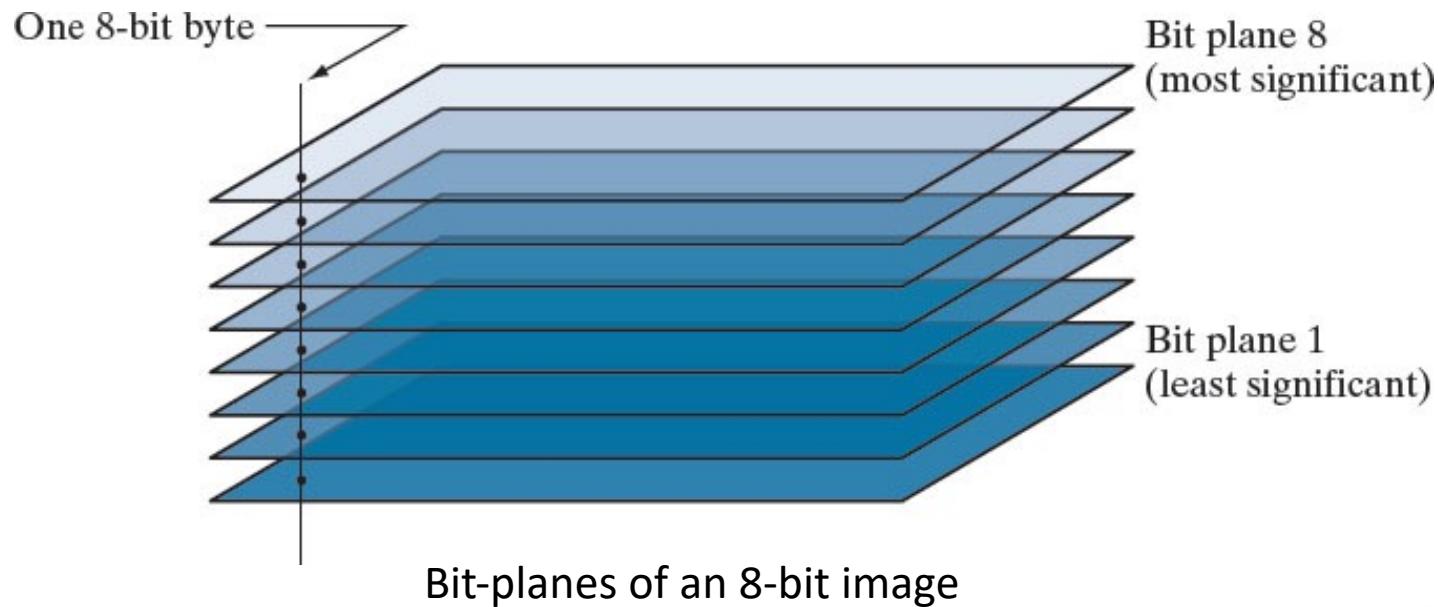
# Basic intensity transforms

- **Bit-plane slicing**
  - Each pixel in a 256-level grayscale image is composed of 8 bits



# Basic intensity transforms

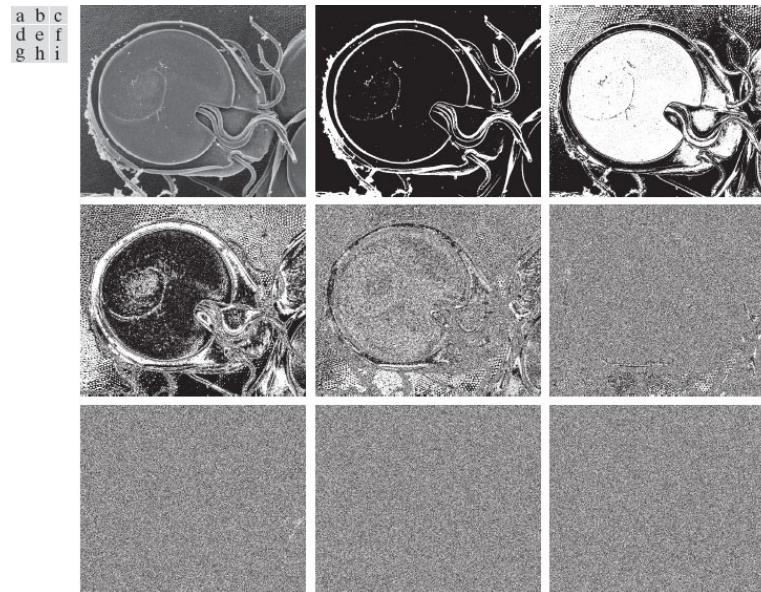
- **Bit-plane slicing**
  - Each pixel in a 256-level grayscale image is composed of 8 bits





# Basic intensity transforms

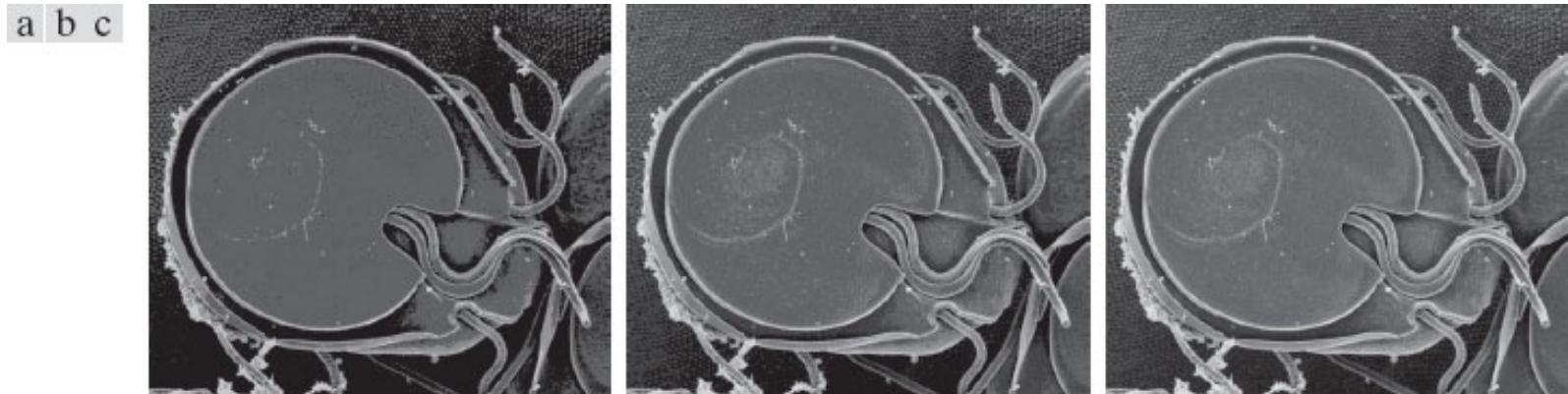
- **Bit-plane slicing (Example)**
- (a) An 8-bit grayscale image of size 837\*988 pixels. (b) through (i) Bit planes 8 through 1, respectively, where plane 1 contains the least significant bit. Each bit plane is a binary image.





# Basic intensity transforms

- **Bit-plane slicing (Example)**
- Image reconstructed from bit planes: (a) 8 and 7; (b) 8, 7, and 6; (c) 8, 7, 6, and 5.
- Can be used for image compression and data hiding???



- What we have seen
  - The difference between intensity transformations and spatial filtering
  - Intensity transformations are applied to individual pixels
  - Different types of intensity transformation functions including:
    - Image negatives
    - Log transformations
    - Power-law (Gamma) transformations
    - Piecewise linear transformations
    - Bit-plane slicing
  - How bit-plane slicing can be used for data hiding

# SCC.366: Media Coding & Processing

2022-2023

Week 03 – Lecture 1

## Image Histogram and its Applications

Dr. Hossein Rahmani

Senior Lecturer in Data Science

Email and Team: h.rahmani@lancaster.ac.uk

# Outline

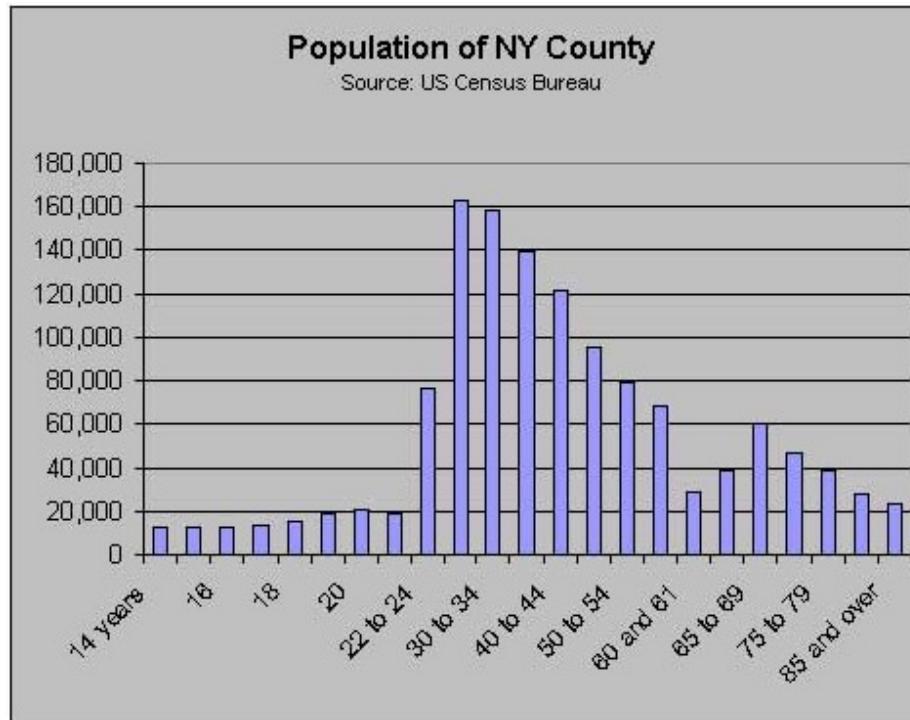
---

- Image types (Binary, Grey scale, color)
- Processing images
  - Spatial vs. Frequency domain
  - Intensity transformations vs. Spatial filtering
- Basic intensity transformations
  - Image negatives,
  - Log transformations,
  - Power-law or Gamma transformations
  - Piecewise linear transformations
  - Bit-plane slicing
- **Image histogram**
  - Histogram equalization,
  - Histogram matching (or histogram specification)



# Histograms

- A Histogram is a Graphical Way to Represent Data
- The height of each bar is the number of data points whose value falls into the range for the bar, called **bin**.



# Image Histogram (Definitions)

---

- **Histogram**
  - The histogram function is defined over all possible intensity levels.
  - For each intensity level, its value is equal to the number of the pixels with that intensity.
  - Very popular tool for real-time image processing



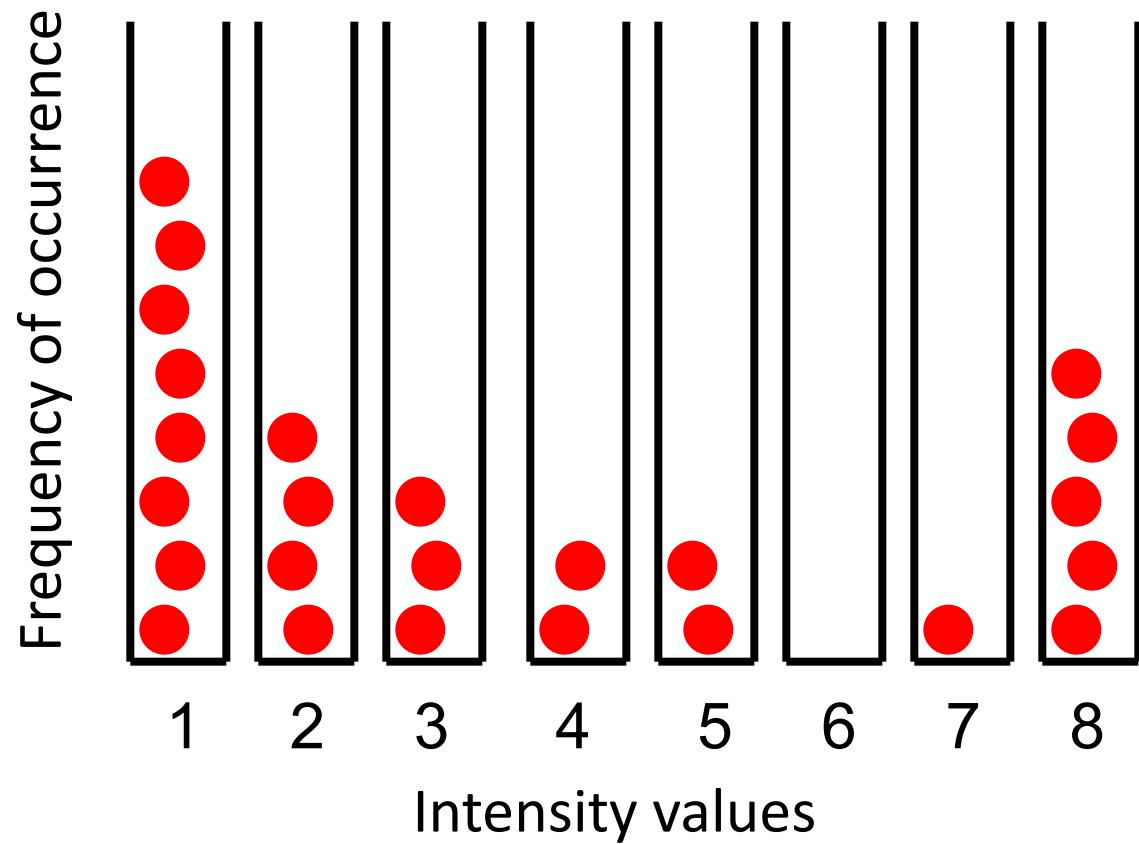
## Image Histogram (Example 1)

- Consider a 5x5 image with integer intensities in the range between 1 and 8:

|   |   |   |   |   |
|---|---|---|---|---|
| 1 | 8 | 4 | 3 | 4 |
| 1 | 1 | 1 | 7 | 8 |
| 8 | 8 | 3 | 3 | 1 |
| 2 | 2 | 1 | 5 | 2 |
| 1 | 1 | 8 | 5 | 2 |

# Image Histogram (Example 1)

|   |   |   |   |   |
|---|---|---|---|---|
| 1 | 8 | 4 | 3 | 4 |
| 1 | 1 | 1 | 7 | 8 |
| 8 | 8 | 3 | 3 | 1 |
| 2 | 2 | 1 | 5 | 2 |
| 1 | 1 | 8 | 5 | 2 |



# Histogram Function (Example 1)

$$h(r_1) = 8$$

$$h(r_2) = 4$$

$$h(r_3) = 3$$

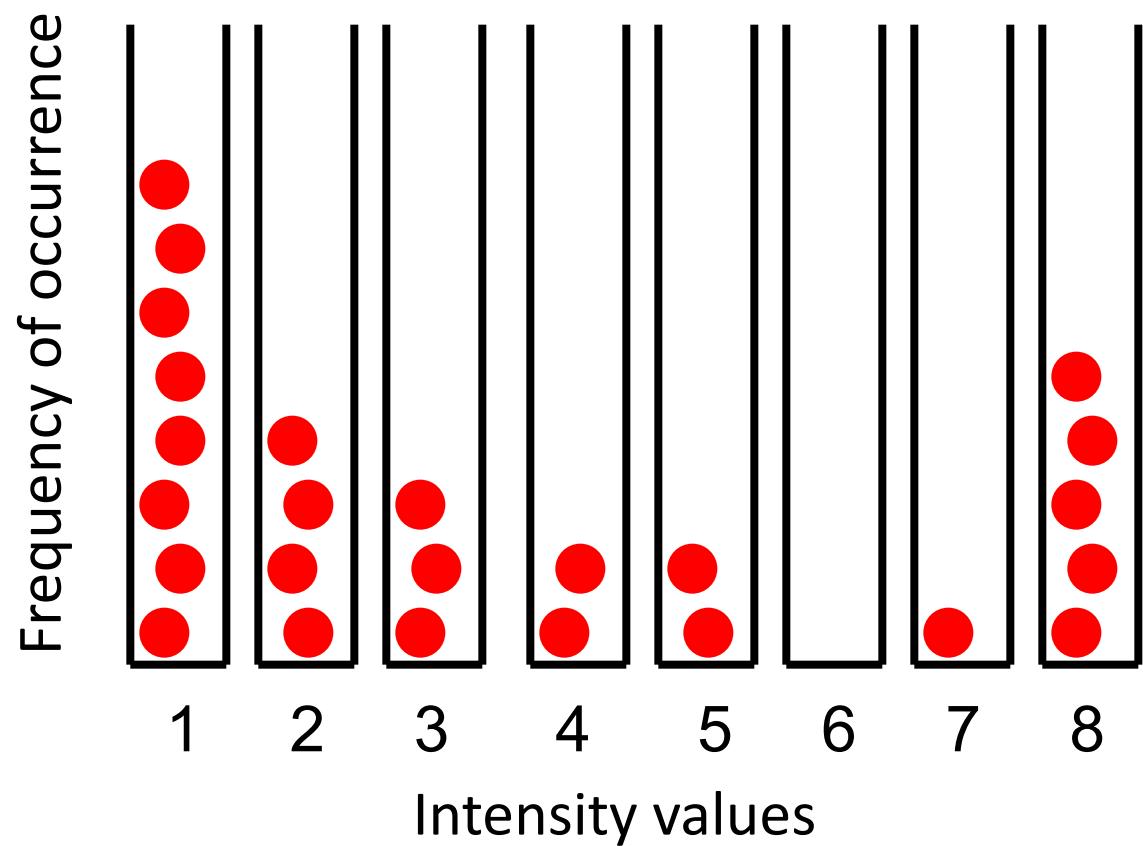
$$h(r_4) = 2$$

$$h(r_5) = 2$$

$$h(r_6) = 0$$

$$h(r_7) = 1$$

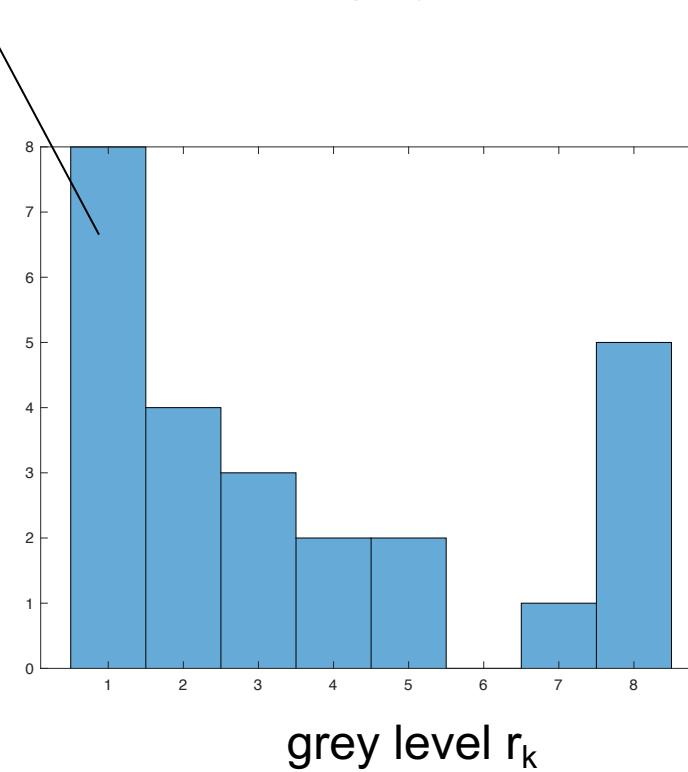
$$h(r_8) = 5$$



# Image Histogram (Example 1)

|   |   |   |   |   |
|---|---|---|---|---|
| 1 | 8 | 4 | 3 | 4 |
| 1 | 1 | 1 | 7 | 8 |
| 8 | 8 | 3 | 3 | 1 |
| 2 | 2 | 1 | 5 | 2 |
| 1 | 1 | 8 | 5 | 2 |

$h(r_k) = \text{no. of pixels with grey level } r_k$



# Normalised Image Histogram

---

- The **normalised histogram function** is the histogram function divided by the **total number of the pixels ( $n$ )** of the image:

$$p(r_k) = \frac{h(r_k)}{n} = \frac{n_k}{n}$$

- It gives a measure of how likely is for a pixel to have a certain intensity (i.e., probability).
- The sum of the normalised histogram function over the range of all intensities is 1.

# Normalised Histogram Function (Example)

$$h(r_1) = 8$$

$$h(r_2) = 4$$

$$h(r_3) = 3$$

$$h(r_4) = 2$$

$$h(r_5) = 2$$

$$h(r_6) = 0$$

$$h(r_7) = 1$$

$$h(r_8) = 5$$



$$p(r_1) = 8/25 = 0.32$$

$$p(r_2) = 4/25 = 0.16$$

$$p(r_3) = 3/25 = 0.12$$

$$p(r_4) = 3/25 = 0.08$$

$$p(r_5) = 2/25 = 0.08$$

$$p(r_6) = 0/25 = 0.00$$

$$p(r_7) = 1/25 = 0.04$$

$$p(r_8) = 5/25 = 0.20$$

1 8 4 3 4

1 1 1 7 8

8 8 3 3 1

2 2 1 5 2

1 1 8 5 2

25 pixels

# Applications of Histograms

---

## □ Histograms help detect image acquisition issues

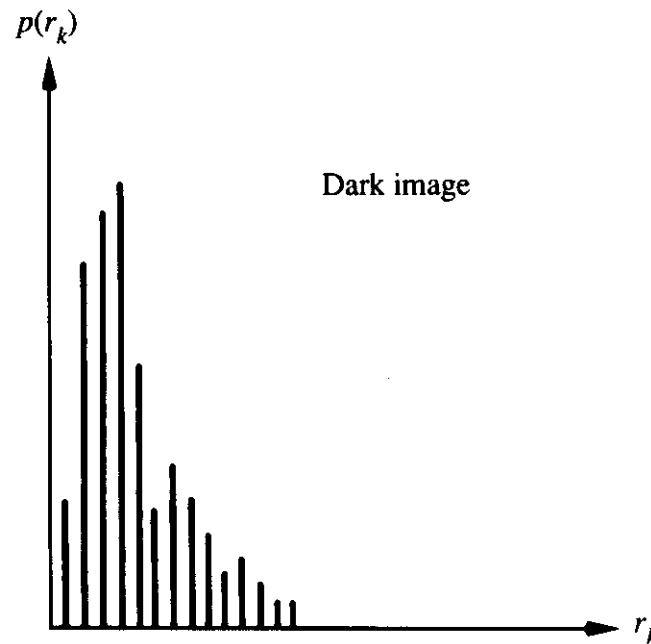
- Problems with image can be identified on histogram
  - Brightness
  - Over and under exposure
  - Contrast

## □ Point operations can be used to alter histogram

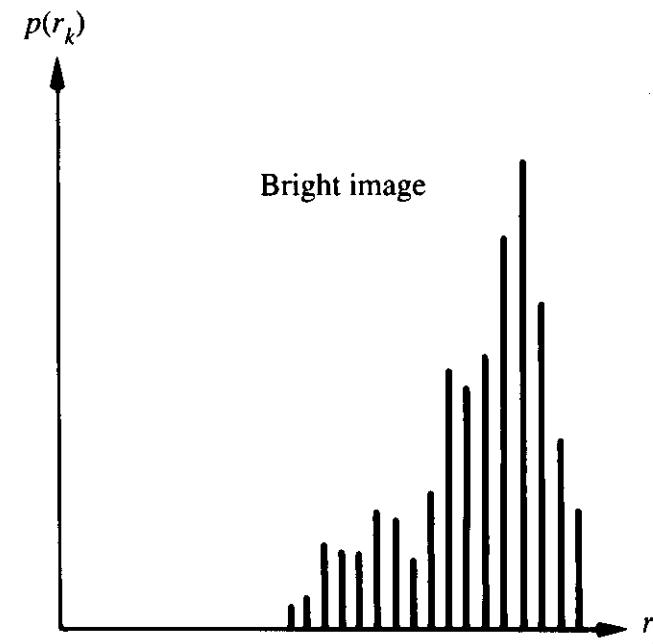
- Addition
- Multiplication
- Exp and Log
- Histogram Equalization
- Histogram Matching

# The Image Histogram & Brightness

$p(r_k)$ : Number of times the corresponding grey level occurs in the image



Intensity levels



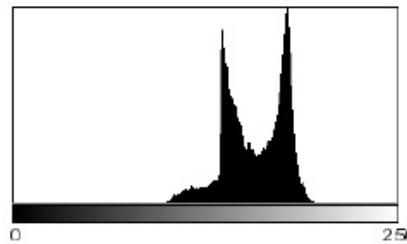
Intensity levels

# The Image Histogram & Exposure

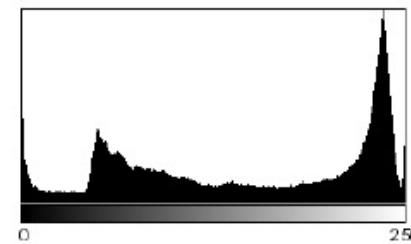


An image that is brighter than it should be can be considered **over-exposed**.

# The Image Histogram & Contrast



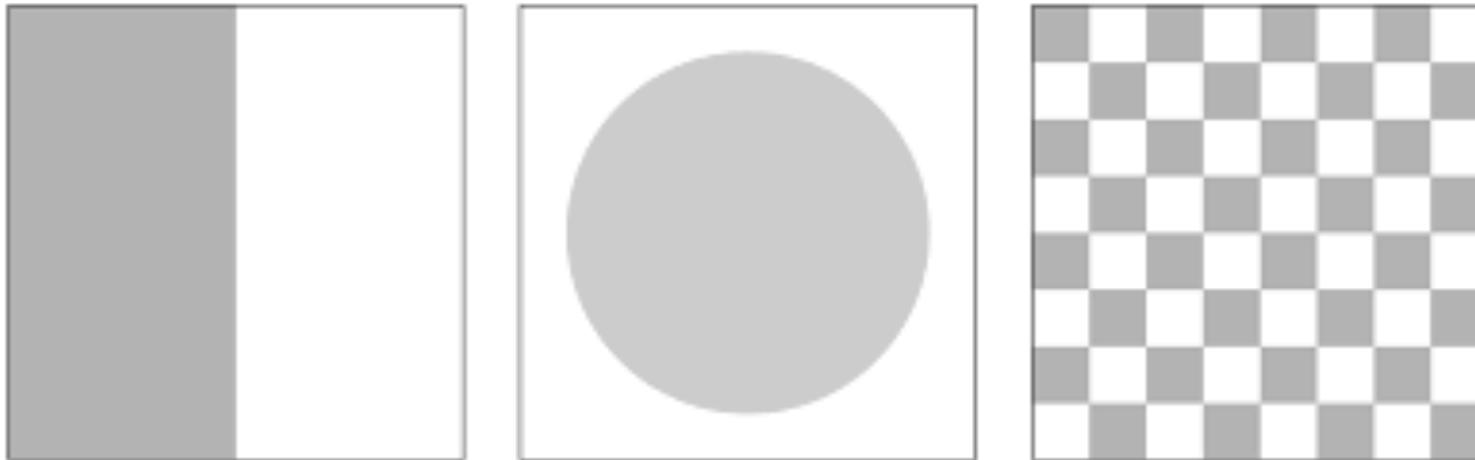
Low contrast



High contrast

**Contrast:** the **range** of intensity values effectively used within a given image

# Can we reconstruct image from histogram?



- **Question:** Draw histograms of the above 3 images
  - Half of pixels are gray, half are white
  - Different images can have same histogram
  - The ***histogram*** conveys no information on the spatial distribution of ***pixels*** with a certain ***grey level***.
  - Can we reconstruct image from histogram? **No!**

# Outline

---

- Image types (Binary, Grey scale, color)
- Processing images
  - Spatial vs. Frequency domain
  - Intensity transformations vs. Spatial filtering
- Basic intensity transformations
  - Image negatives,
  - Log transformations,
  - Power-law or Gamma transformations
  - Piecewise linear transformations
  - Bit-plane slicing
- Image histogram
  - **Histogram equalization**
  - Histogram matching (or histogram specification)

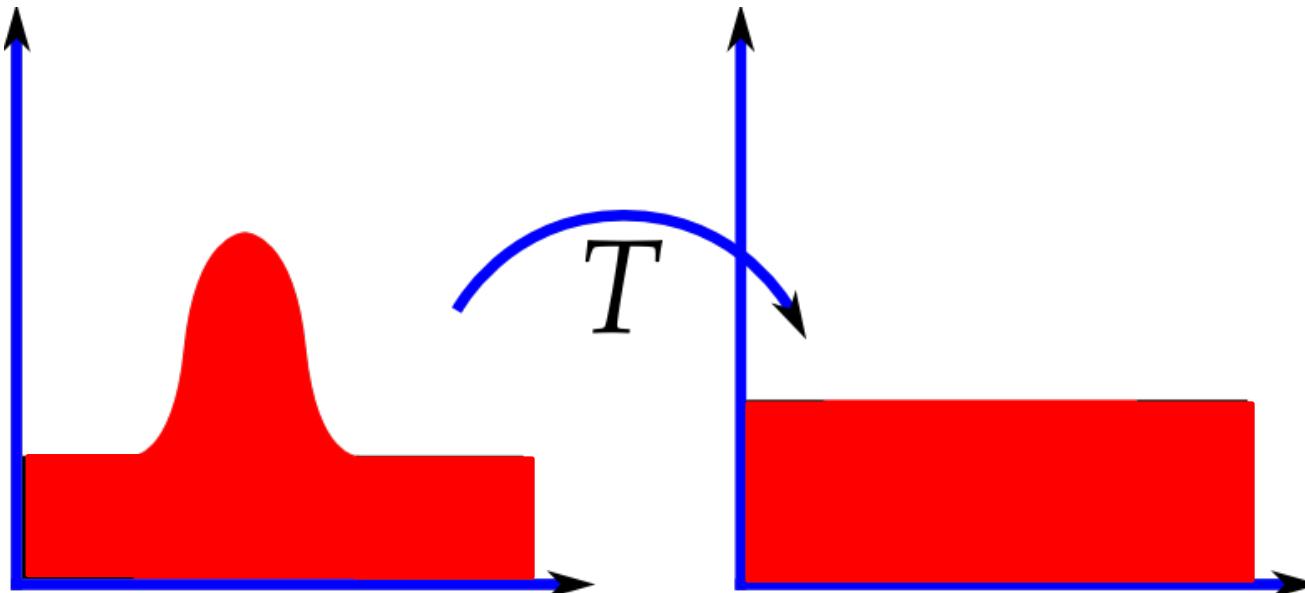
# Histogram Equalization

---

- Histogram Equalization is a computer image processing technique used to improve contrast in images.
- It accomplishes this by effectively spreading out the most frequent intensity values, i.e. stretching out the intensity range of the image.
- Apply a point operation that changes histogram of modified image into **uniform distribution**

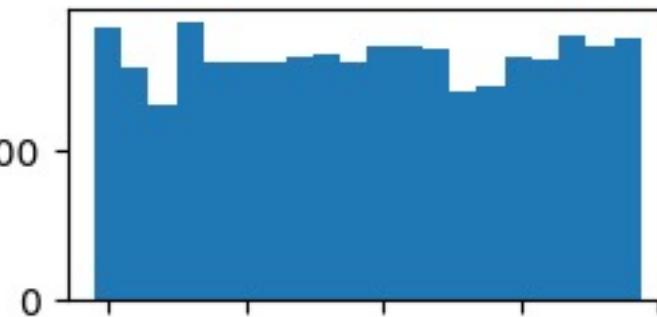
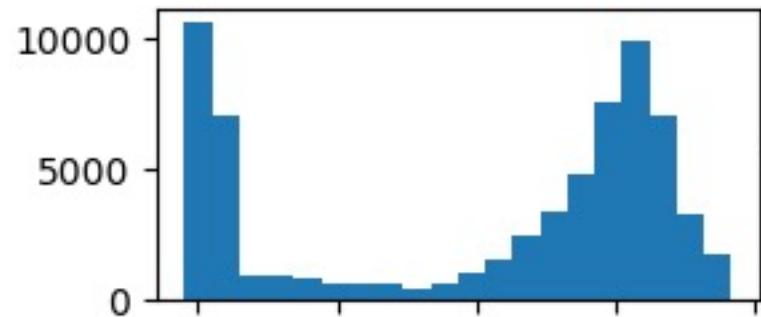
# Histogram Equalization

- Apply a point operation that changes histogram of modified image into uniform distribution
- Spreading out the frequencies in an image (or equalizing the image) is a simple way to improve dark or washed out images



# Histogram Equalization

- Apply a point operation that changes histogram of modified image into uniform distribution

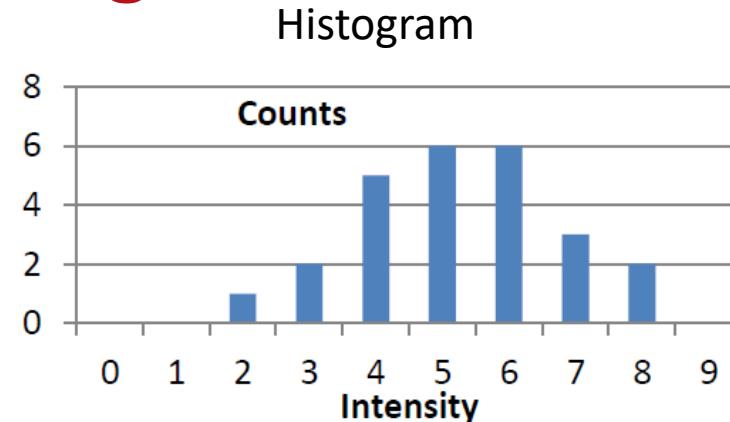
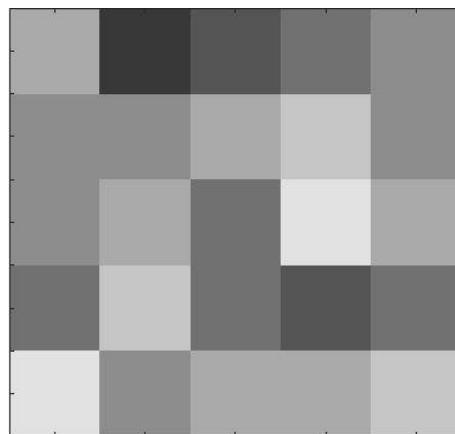


# Histogram Equalization Algorithm

**First step: Create image histogram**

Original Image

|   |   |   |   |   |
|---|---|---|---|---|
| 6 | 2 | 3 | 4 | 5 |
| 5 | 5 | 6 | 7 | 5 |
| 5 | 6 | 4 | 8 | 6 |
| 4 | 7 | 4 | 3 | 4 |
| 8 | 5 | 6 | 6 | 7 |



| Intensity | Frequency |
|-----------|-----------|
| 0         | 0         |
| 1         | 0         |
| 2         | 1         |
| 3         | 2         |
| 4         | 5         |
| 5         | 6         |
| 6         | 6         |
| 7         | 3         |
| 8         | 2         |
| 9         | 0         |

# Histogram Equalization Algorithm

**Second step: Create normalized image histogram**

$$P_0 = \frac{0}{25} = 0 \quad P_1 = \frac{0}{25} = 0 \quad P_2 = \frac{1}{25} = 0.04 \quad P_3 = \frac{2}{25} = 0.08 \quad P_4 = \frac{5}{25} = 0.2$$

$$P_5 = \frac{6}{25} = 0.24 \quad P_6 = \frac{6}{25} = 0.24 \quad P_7 = \frac{3}{25} = 0.12 \quad P_8 = \frac{2}{25} = 0.08 \quad P_9 = \frac{0}{25} = 0$$

| Intensity | Frequency | Normalized |
|-----------|-----------|------------|
| 0         | 0         | 0          |
| 1         | 0         | 0          |
| 2         | 1         | 0.04       |
| 3         | 2         | 0.08       |
| 4         | 5         | 0.2        |
| 5         | 6         | 0.24       |
| 6         | 6         | 0.24       |
| 7         | 3         | 0.12       |
| 8         | 2         | 0.08       |
| 9         | 0         | 0          |

# Histogram Equalization Algorithm

Third step: Calculating cumulative distribution function

| Intensity | Frequency | Normalized | cumulative |
|-----------|-----------|------------|------------|
| 0         | 0         | 0          | 0          |
| 1         | 0         | 0          | 0          |
| 2         | 1         | 0.04       | 0.04       |
| 3         | 2         | 0.08       | 0.12       |
| 4         | 5         | 0.2        | 0.32       |
| 5         | 6         | 0.24       | 0.56       |
| 6         | 6         | 0.24       | 0.8        |
| 7         | 3         | 0.12       | 0.92       |
| 8         | 2         | 0.08       | 1          |
| 9         | 0         | 0          | 1          |

# Histogram Equalization Algorithm

Forth step: Applying transformation

*New intensity = floor(maximum intensity × cumulative)*

“*floor(.)*” function maps a **real number** to the **largest previous integer**

| Intensity | Frequency | Normalized | cumulative | transformation           |
|-----------|-----------|------------|------------|--------------------------|
| 0         | 0         | 0          | 0          | $\text{floor}(9*0)=0$    |
| 1         | 0         | 0          | 0          | $\text{floor}(9*0)=0$    |
| 2         | 1         | 0.04       | 0.04       | $\text{floor}(9*0.04)=0$ |
| 3         | 2         | 0.08       | 0.12       | $\text{floor}(9*0.12)=1$ |
| 4         | 5         | 0.2        | 0.32       | $\text{floor}(9*0.32)=2$ |
| 5         | 6         | 0.24       | 0.56       | $\text{floor}(9*0.56)=5$ |
| 6         | 6         | 0.24       | 0.8        | $\text{floor}(9*0.8)=7$  |
| 7         | 3         | 0.12       | 0.92       | $\text{floor}(9*0.92)=8$ |
| 8         | 2         | 0.08       | 1          | $\text{floor}(9*1)=9$    |
| 9         | 0         | 0          | 1          | $\text{floor}(9*1)=9$    |

# Histogram Equalization Algorithm

Forth step: Applying transformation

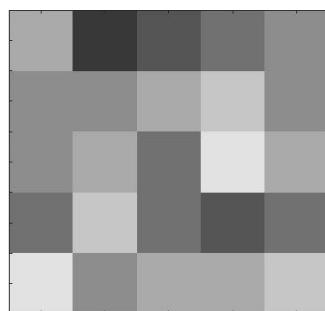
$$s = T(i) = \text{floor}((L - 1) * \sum_{k=0}^i p_k) = \text{floor}(9 * \sum_{k=0}^i p_k)$$

“*floor(.)*” function maps a **real number** to the **largest previous integer**

| Intensity | Frequency | Normalized | cumulative | transformation |
|-----------|-----------|------------|------------|----------------|
| 0         | 0         | 0          | 0          | 0              |
| 1         | 0         | 0          | 0          | 0              |
| 2         | 1         | 0.04       | 0.04       | 0              |
| 3         | 2         | 0.08       | 0.12       | 1              |
| 4         | 5         | 0.2        | 0.32       | 2              |
| 5         | 6         | 0.24       | 0.56       | 5              |
| 6         | 6         | 0.24       | 0.8        | 7              |
| 7         | 3         | 0.12       | 0.92       | 8              |
| 8         | 2         | 0.08       | 1          | 9              |
| 9         | 0         | 0          | 1          | 9              |

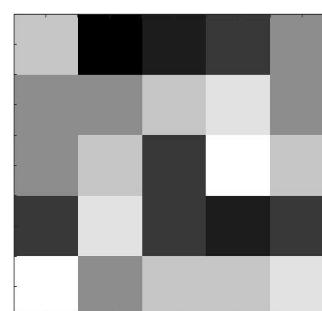
# Histogram Equalization Algorithm

|   |   |   |   |   |
|---|---|---|---|---|
| 6 | 2 | 3 | 4 | 5 |
| 5 | 5 | 6 | 7 | 5 |
| 5 | 6 | 4 | 8 | 6 |
| 4 | 7 | 4 | 3 | 4 |
| 8 | 5 | 6 | 6 | 7 |



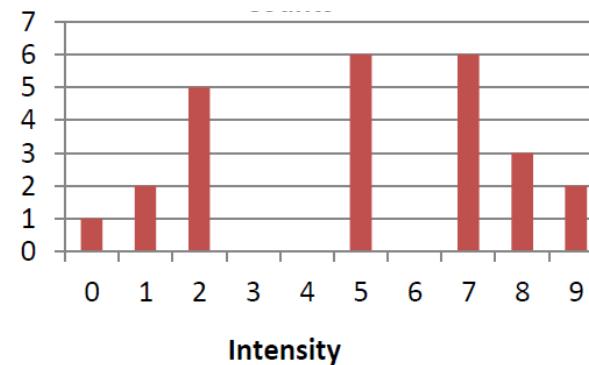
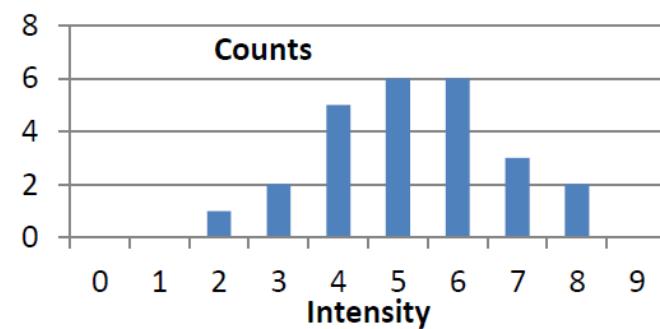
Original Image

|   |   |   |   |   |
|---|---|---|---|---|
| 7 | 0 | 1 | 2 | 5 |
| 5 | 5 | 7 | 8 | 5 |
| 5 | 7 | 2 | 9 | 7 |
| 2 | 8 | 2 | 1 | 2 |
| 9 | 5 | 7 | 7 | 8 |



Equalized Image

| Intensity | transformation |
|-----------|----------------|
| 0         | 0              |
| 1         | 0              |
| 2         | 0              |
| 3         | 1              |
| 4         | 2              |
| 5         | 5              |
| 6         | 7              |
| 7         | 8              |
| 8         | 9              |
| 9         | 9              |



# Histogram Equalization (Example)



Original image  
very poor contra



Image after histogram equalization  
improved contrast



# Histogram Equalization (Example)



Original image  
very poor contrast



Image after histogram equalization  
improved contrast

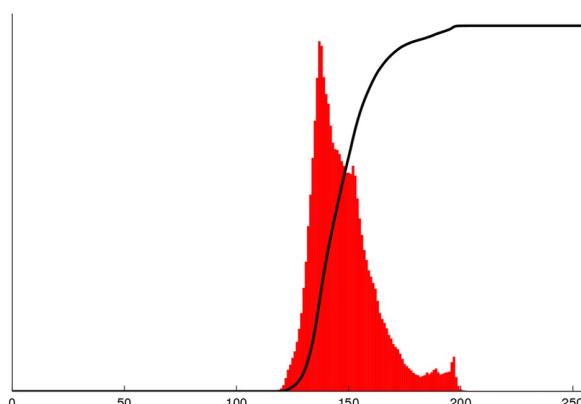
# Histogram Equalization (Example)



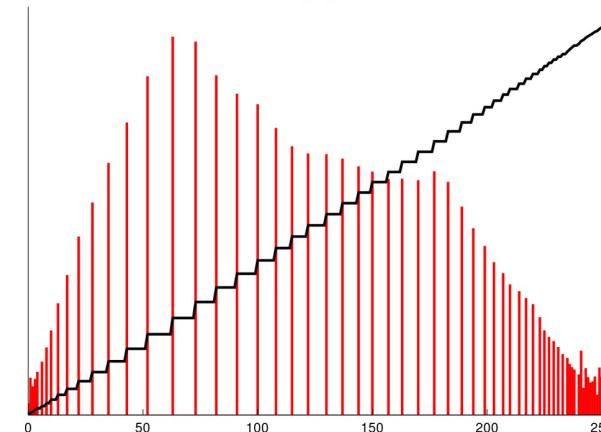
(a)



(b)



(c)



(d)

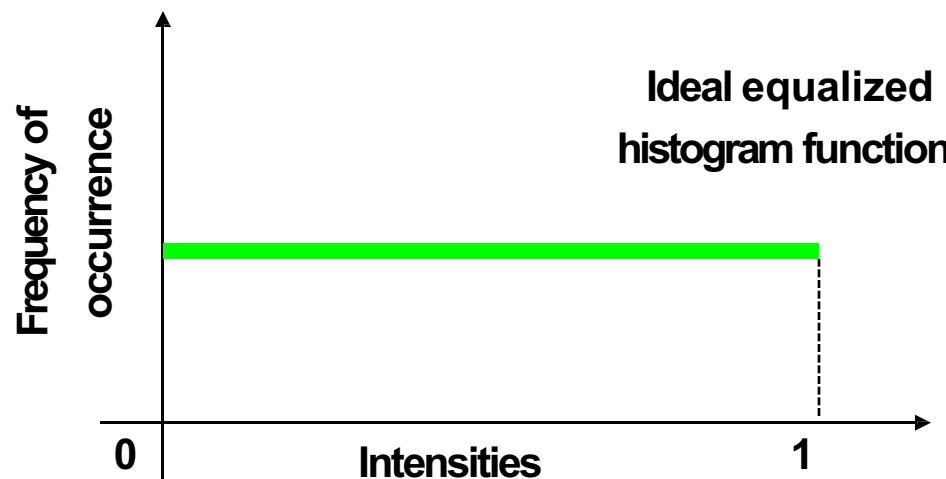
# Histogram Equalization (Summary)

---

- Histogram equalization tries to make the intensity distribution uniform.
  - This can enhance detail.
  - The full grey level dynamic range is exploited.

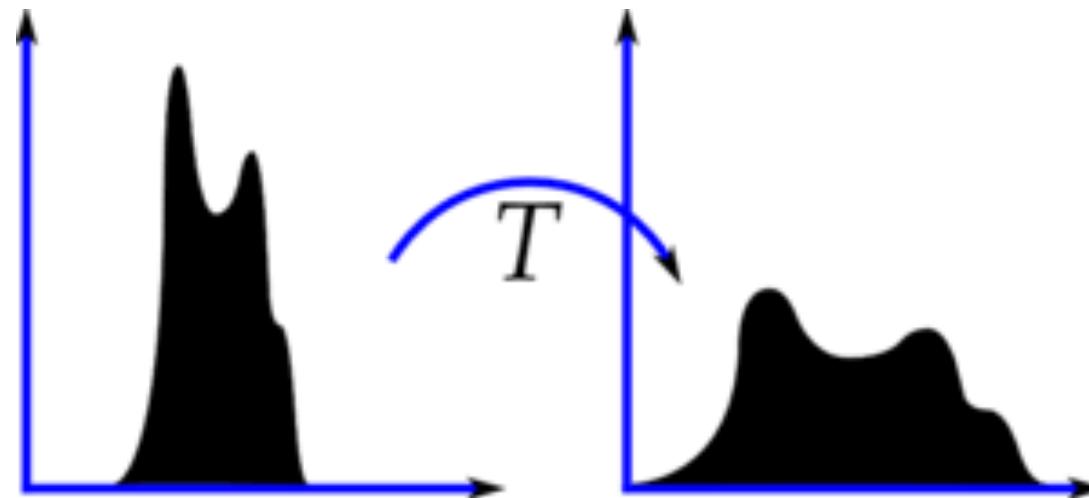
# Is histogram equalization always good?

- Histogram equalization tries to make the intensity distribution uniform.
  - Because of the irregular initial distribution, usually this is not possible.
  - Sometimes histogram equalization introduces artefacts



# Is histogram equalization always good?

- Histogram equalization tries to make the intensity distribution uniform.
  - Because of the irregular initial distribution, usually this is not possible.
  - Sometimes histogram equalization introduces artefacts



# Outline

---

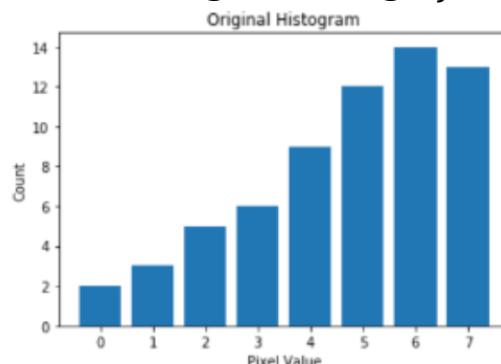
- Image types (Binary, Grey scale, color)
- Processing images
  - Spatial vs. Frequency domain
  - Intensity transformations vs. Spatial filtering
- Basic intensity transformations
  - Image negatives,
  - Log transformations,
  - Power-law or Gamma transformations
  - Piecewise linear transformations
  - Bit-plane slicing
- Image histogram
  - Histogram equalization,
  - **Histogram matching (or histogram specification)**

# Histogram Matching

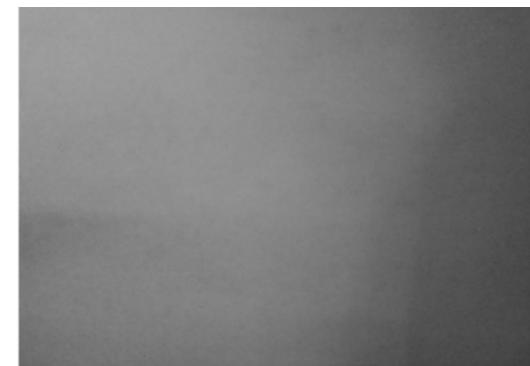
- Instead
  - We can specify the function we want for the histogram



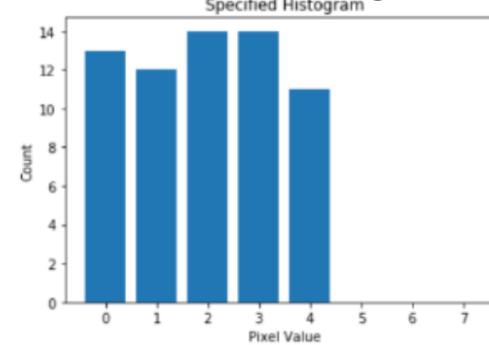
Original image  $f$



$h_f$  Initial (source) Histogram



Specified image  $g$



$h_g$  Specified (target) Histogram

# Histogram Matching

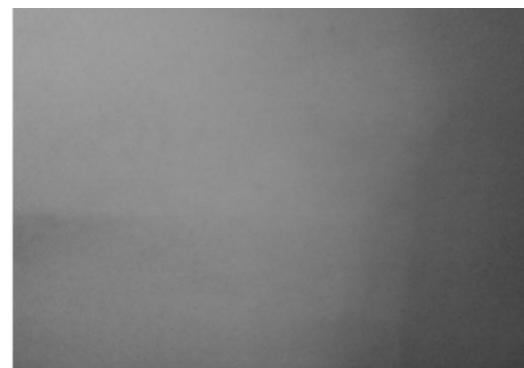
- Histogram Matching is also called Histogram Specification
- Goal
  - Given an image  $f$  with a histogram  $h_f$
  - Given a specified image  $g$  with a histogram  $h_g$
  - Find a transformation  $T$  such that  $T(h_f)$  matches the distribution of  $h_g$ .
- Remarks
  - If the distribution of  $h_g$  is uniform, then  $T$  is the same as the histogram equalization.

# Histogram Matching

- Goal
  - Given an image  $f$  with a histogram  $h_f$
  - Given a specified image  $g$  with a histogram  $h_g$
  - Find a transformation  $T$  such that  $T(h_f)$  matches the distribution of  $h_g$ .



Original image  $f$   
with histogram  $h_f$



Specified image  $g$   
with histogram  $h_g$

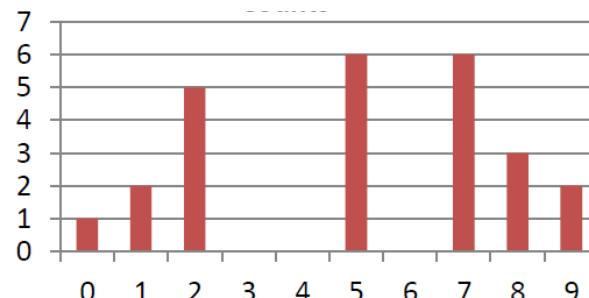
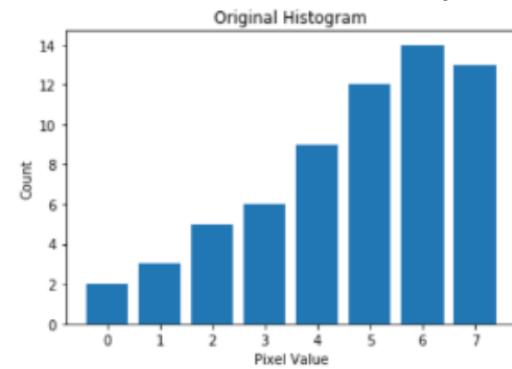


New image  
with histogram similar to  
 $h_g$

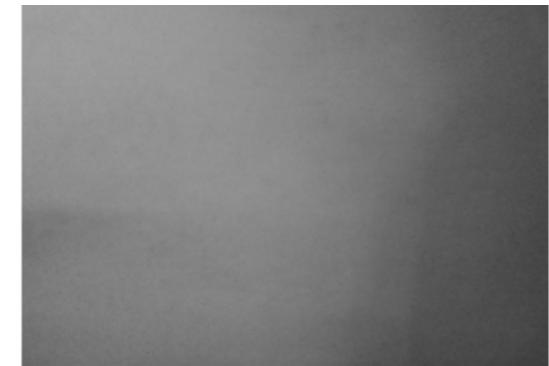
# Histogram Matching Algorithm (Overall idea)



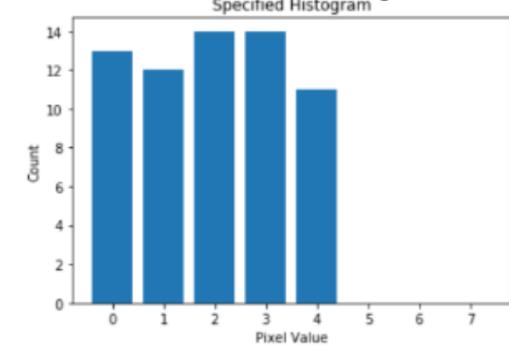
Original image  $f$



Uniform histogram



Specified image  $g$

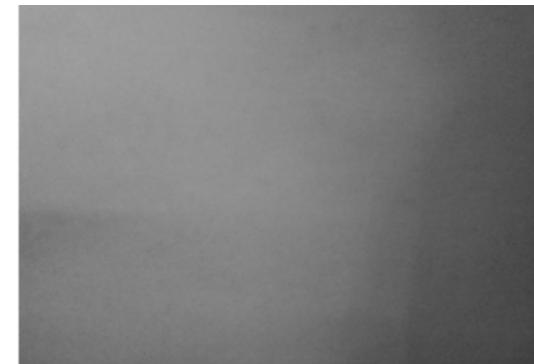
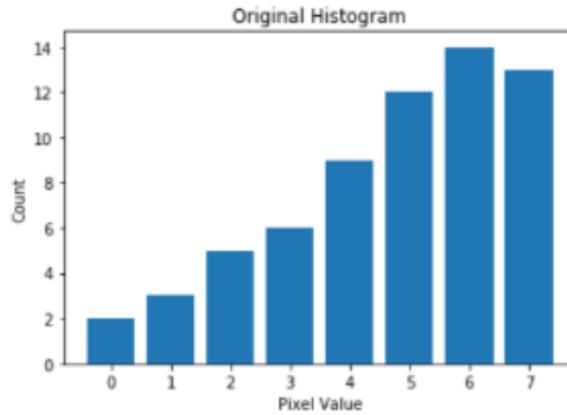


# Histogram Matching Algorithm (Step 1)

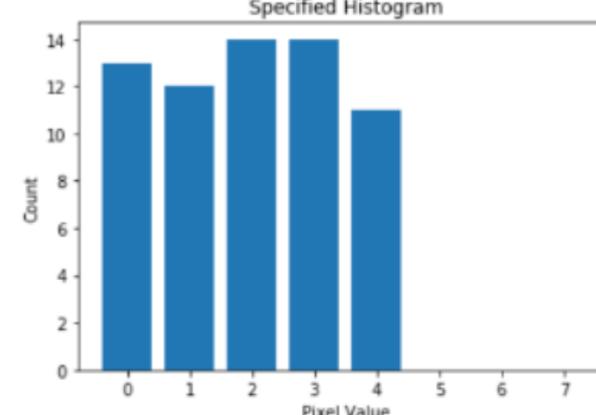
- **Step 1:** Given two images, calculate histogram equalization for both original and the specified images



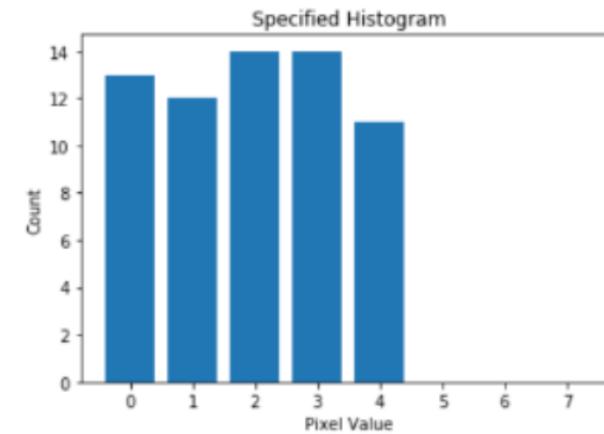
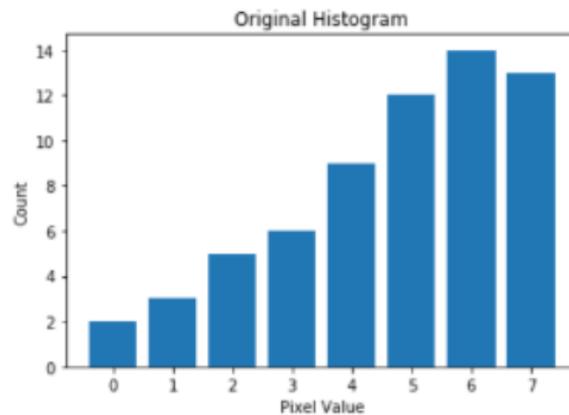
Original image  $f$



Specified image  $g$



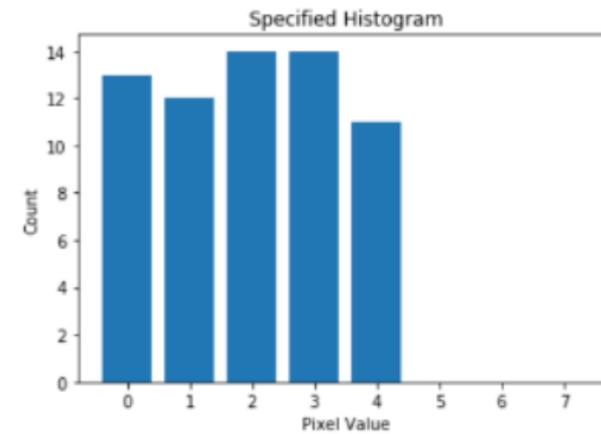
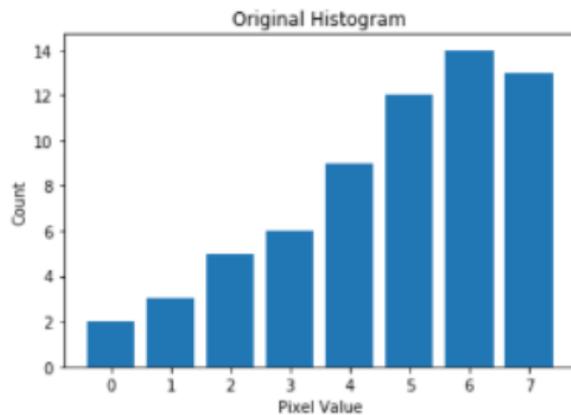
# Histogram Matching Algorithm (Step 1)



| Intensity | Frequency | transformation |
|-----------|-----------|----------------|
| 0         | 2         | 0              |
| 1         | 3         | 1              |
| 2         | 5         | 1              |
| 3         | 6         | 2              |
| 4         | 9         | 3              |
| 5         | 12        | 4              |
| 6         | 14        | 6              |
| 7         | 13        | 7              |

| Intensity | Frequency | transformation |
|-----------|-----------|----------------|
| 0         | 13        | 1              |
| 1         | 12        | 3              |
| 2         | 14        | 4              |
| 3         | 14        | 6              |
| 4         | 11        | 7              |
| 5         | 0         | 7              |
| 6         | 0         | 7              |
| 7         | 0         | 7              |

# Histogram Matching Algorithm (Step 1)

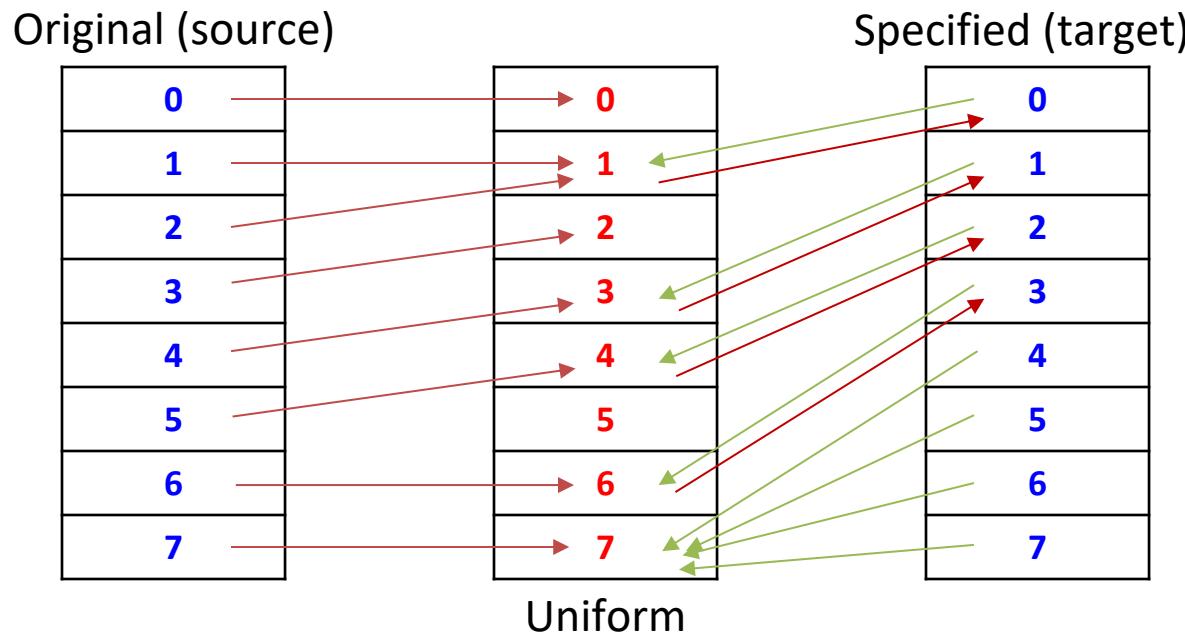


| Intensity | transformation |
|-----------|----------------|
| 0         | 0              |
| 1         | 1              |
| 2         | 1              |
| 3         | 2              |
| 4         | 3              |
| 5         | 4              |
| 6         | 6              |
| 7         | 7              |

| Intensity | transformation |
|-----------|----------------|
| 0         | 1              |
| 1         | 3              |
| 2         | 4              |
| 3         | 6              |
| 4         | 7              |
| 5         | 7              |
| 6         | 7              |
| 7         | 7              |

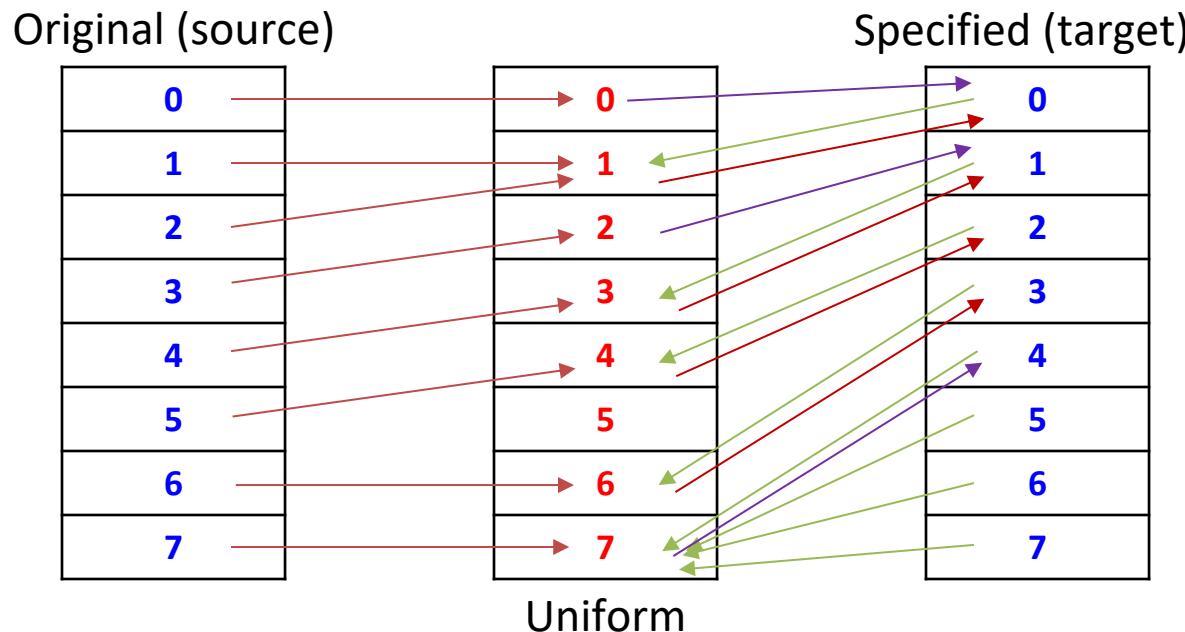
# Histogram Matching Algorithm (Intuitively)

- **Step 2:** Calculate the inverse of the specified



# Histogram Matching Algorithm (Intuitively)

- **Step 2:** Calculate the inverse of the specified



- **What we have seen**

- What is an image histogram and how to create image histograms for gray-scale images
- How histogram equalization works and how to apply it to color images
- How histogram matching works
  - It enables the matching of the histogram of one image to the histogram of another image

# SCC.366: Media Coding & Processing

2022-2023

Week 03 – Lecture 2

## Image Histogram and its Applications Cont.

Dr. Hossein Rahmani

Senior Lecturer in Data Science

Email and Team: h.rahmani@lancaster.ac.uk

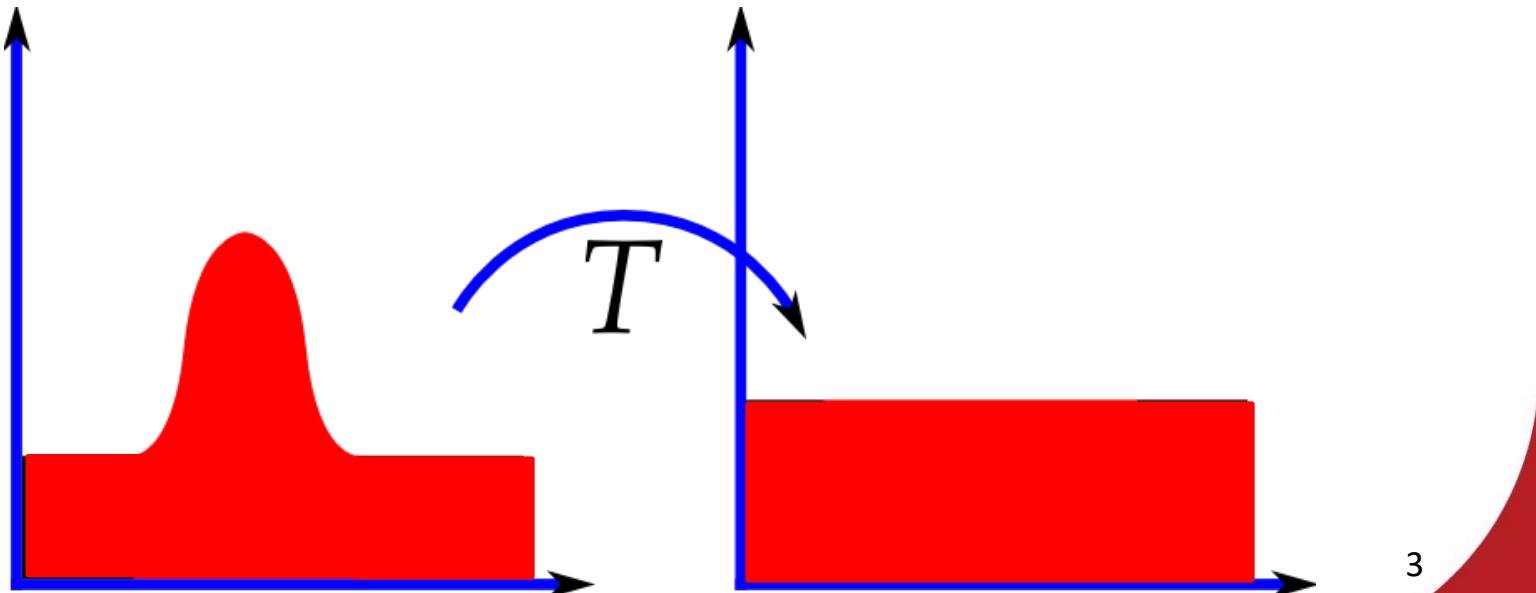
# Outline

---

- Image types (Binary, Grey scale, color)
- Processing images
  - Spatial vs. Frequency domain
  - Intensity transformations vs. Spatial filtering
- Basic intensity transformations
  - Image negatives,
  - Log transformations,
  - Power-law or Gamma transformations
  - Piecewise linear transformations
  - Bit-plane slicing
- Image histogram
  - Histogram equalization,
  - **Histogram matching (or histogram specification)**

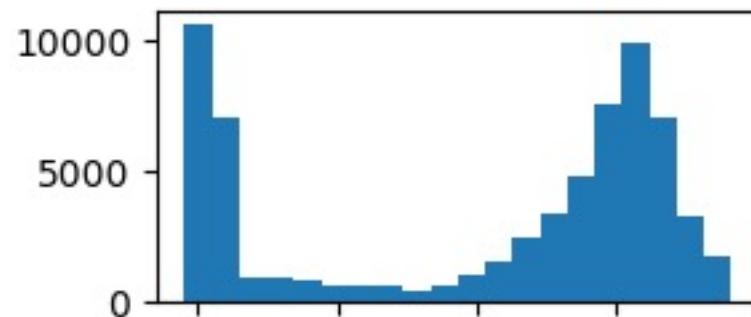
# Review - Histogram Equalization

- Apply a point operation that changes histogram of modified image into uniform distribution
- Spreading out the frequencies in an image (or equalizing the image) is a simple way to improve dark or washed out images



# Review - Histogram Equalization

- Apply a point operation that changes histogram of modified image into uniform distribution

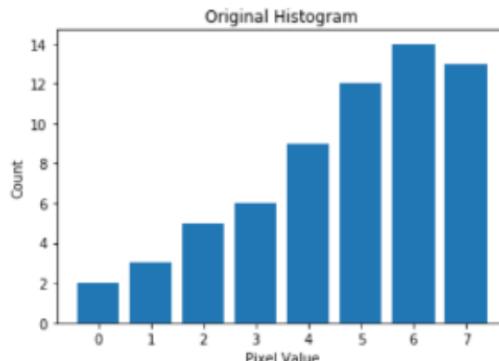


# Review - Histogram Matching

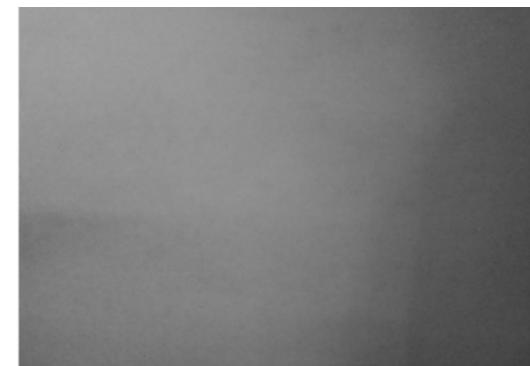
- Instead
  - We can specify the function we want for the histogram



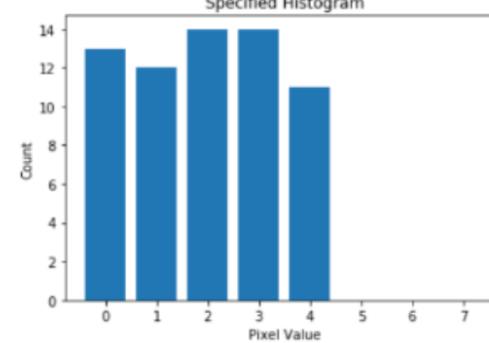
Original image  $f$



$h_f$  Initial (source) Histogram



Specified image  $g$

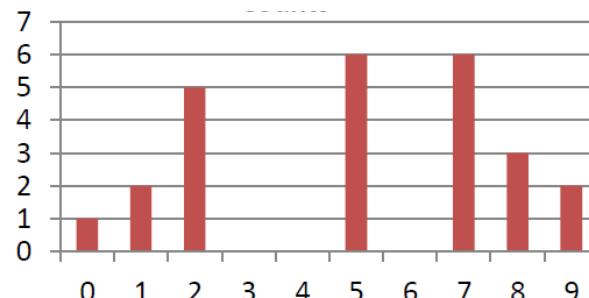
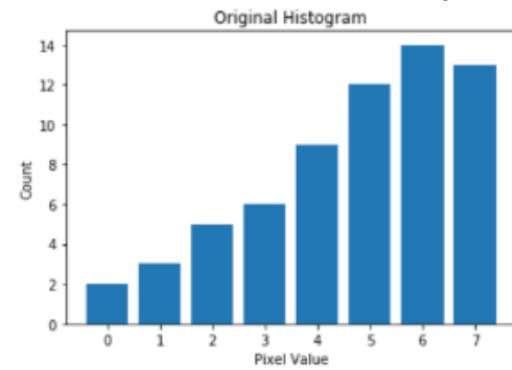


$h_g$  Specified (target) Histogram

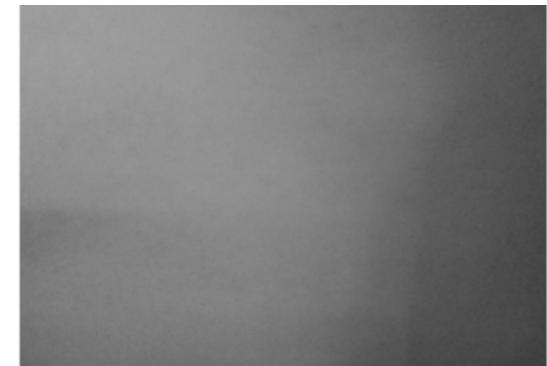
# Histogram Matching Algorithm (Overall idea)



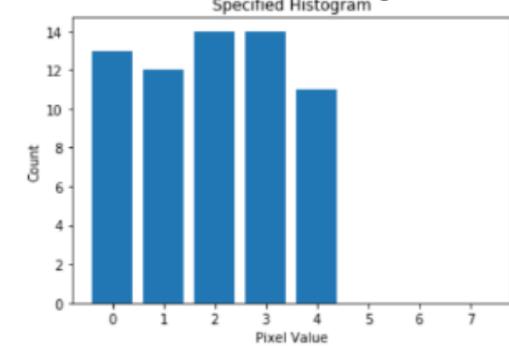
Original image  $f$



Uniform histogram



Specified image  $g$

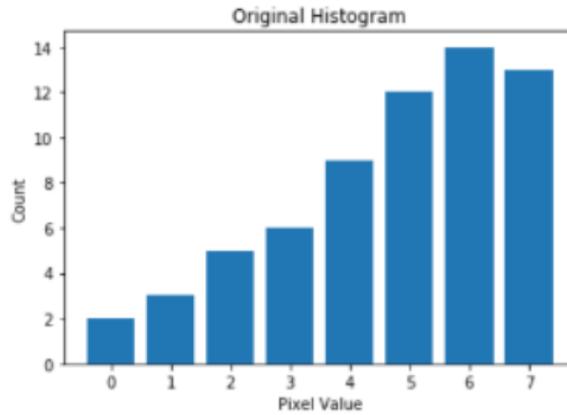


# Histogram Matching Algorithm (Step 1)

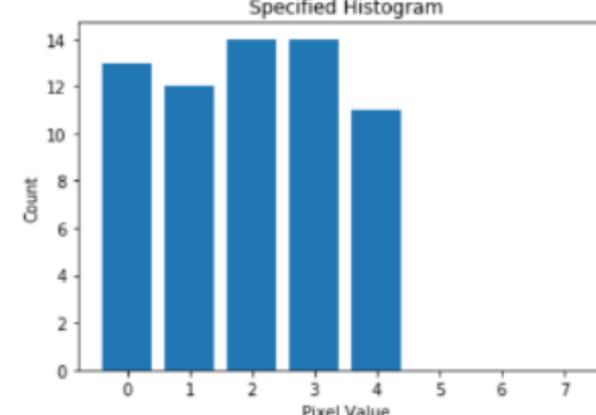
- **Step 1:** Given two images, calculate histogram equalization for both original and the specified images



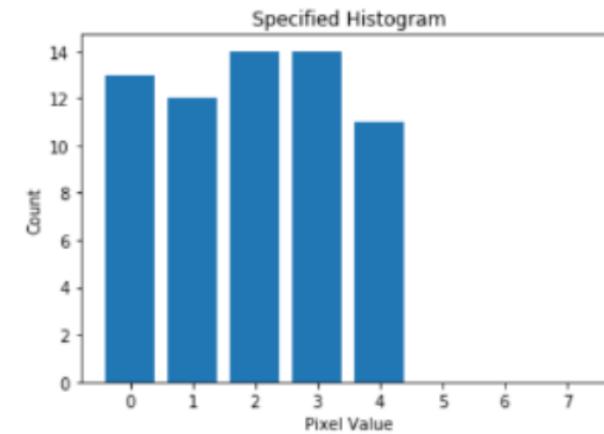
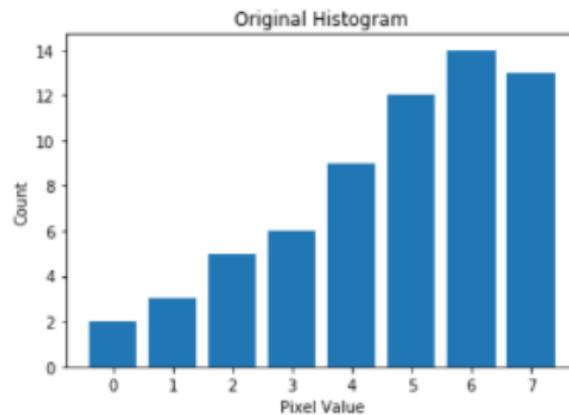
Original image  $f$



Specified image  $g$



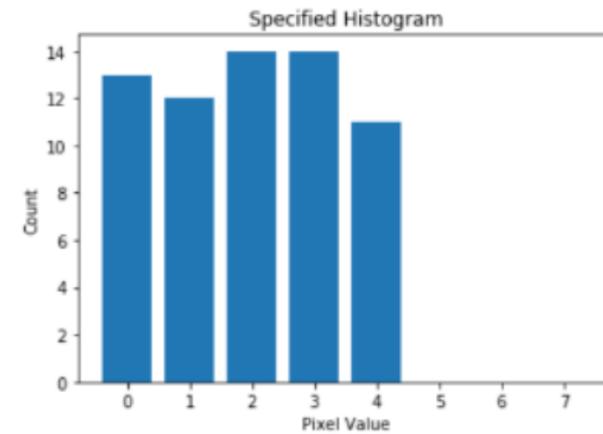
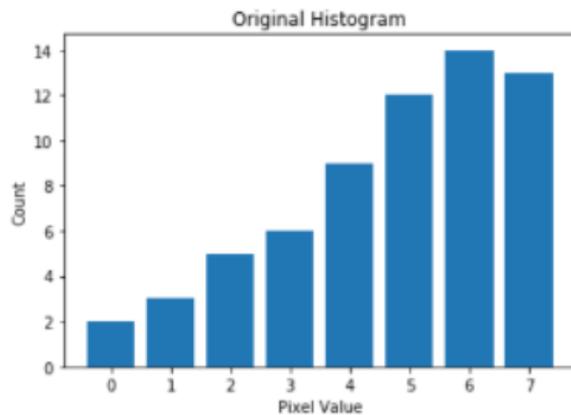
# Histogram Matching Algorithm (Step 1)



| Intensity | Frequency | transformation |
|-----------|-----------|----------------|
| 0         | 2         | 0              |
| 1         | 3         | 1              |
| 2         | 5         | 1              |
| 3         | 6         | 2              |
| 4         | 9         | 3              |
| 5         | 12        | 4              |
| 6         | 14        | 6              |
| 7         | 13        | 7              |

| Intensity | Frequency | transformation |
|-----------|-----------|----------------|
| 0         | 13        | 1              |
| 1         | 12        | 3              |
| 2         | 14        | 4              |
| 3         | 14        | 6              |
| 4         | 11        | 7              |
| 5         | 0         | 7              |
| 6         | 0         | 7              |
| 7         | 0         | 7              |

# Histogram Matching Algorithm (Step 1)

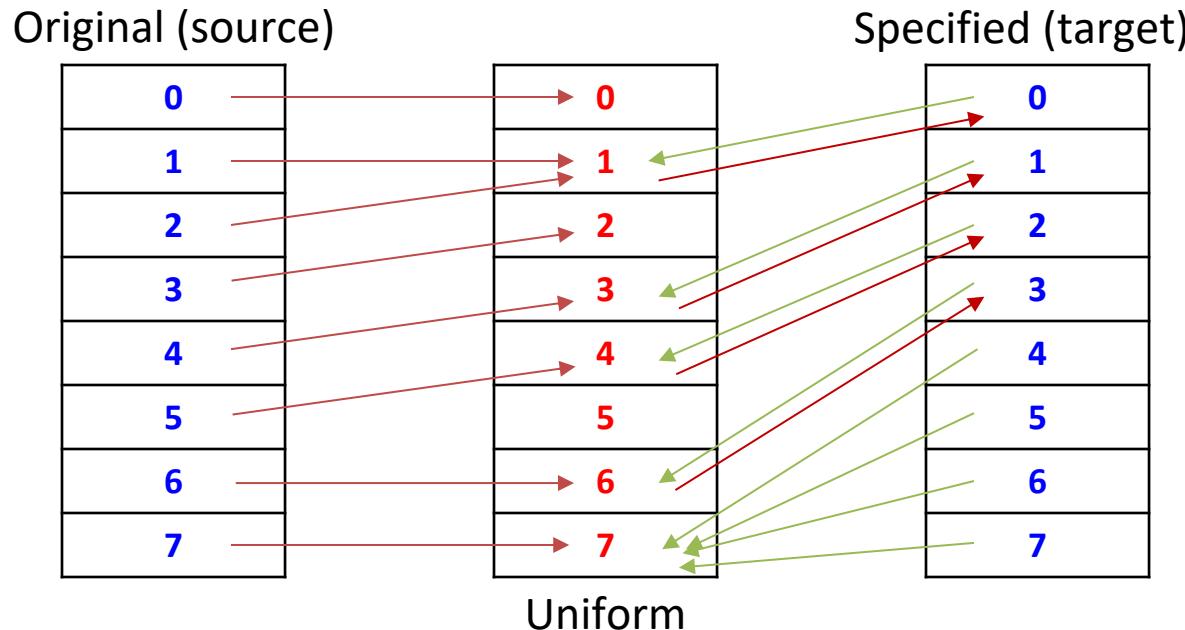


| Intensity | transformation |
|-----------|----------------|
| 0         | 0              |
| 1         | 1              |
| 2         | 1              |
| 3         | 2              |
| 4         | 3              |
| 5         | 4              |
| 6         | 6              |
| 7         | 7              |

| Intensity | transformation |
|-----------|----------------|
| 0         | 1              |
| 1         | 3              |
| 2         | 4              |
| 3         | 6              |
| 4         | 7              |
| 5         | 7              |
| 6         | 7              |
| 7         | 7              |

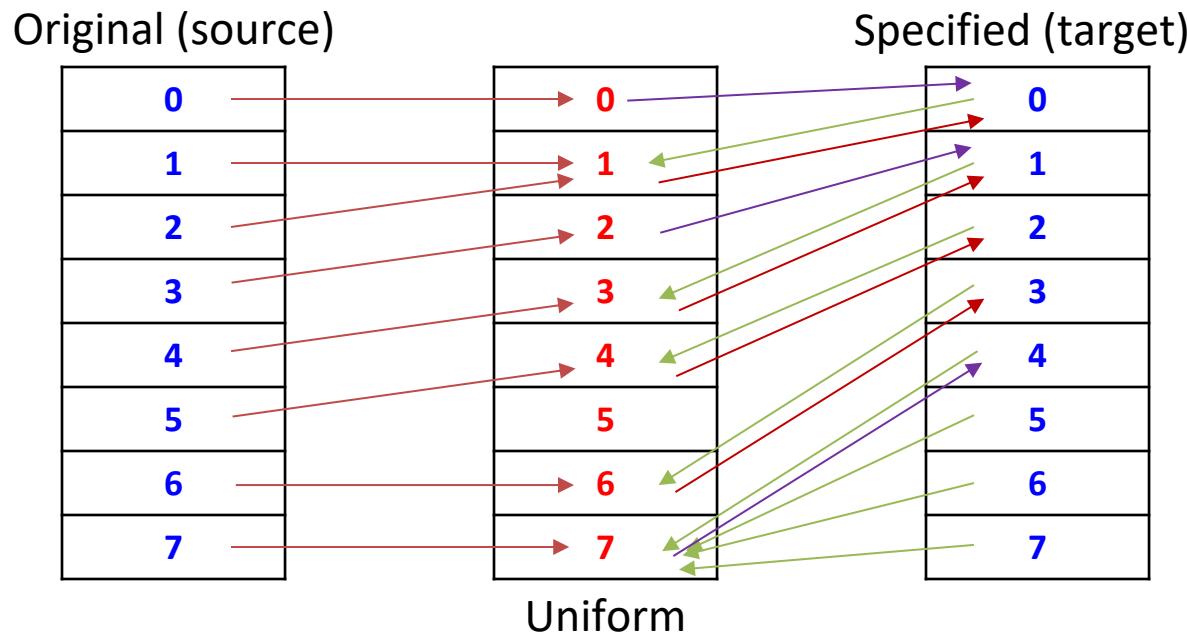
# Histogram Matching Algorithm (Intuitively)

- **Step 2:** Calculate the inverse of the specified

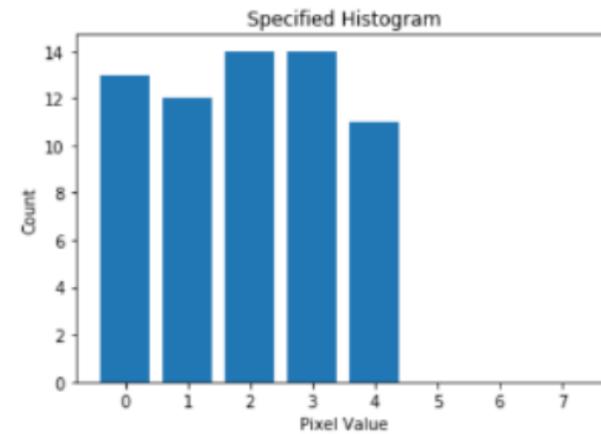
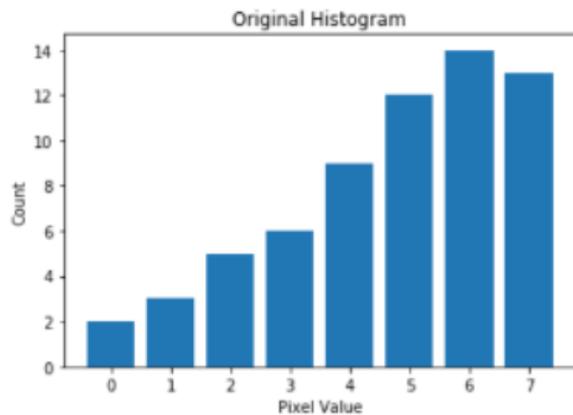


# Histogram Matching Algorithm (Intuitively)

- **Step 2:** Calculate the inverse of the specified



# Histogram Matching Algorithm (Algorithmic)



| Intensity | transformation |
|-----------|----------------|
| 0         | 0              |
| 1         | 1              |
| 2         | 1              |
| 3         | 2              |
| 4         | 3              |
| 5         | 4              |
| 6         | 6              |
| 7         | 7              |

| Intensity | transformation |
|-----------|----------------|
| 0         | 1              |
| 1         | 3              |
| 2         | 4              |
| 3         | 6              |
| 4         | 7              |
| 5         | 7              |
| 6         | 7              |
| 7         | 7              |

# Histogram Matching Algorithm (Algorithmic)

- **Step 2:** Calculate the inverse of the specified
  - First add the “Final Map” column

| Intensity | Transformation (original) | Transformation (specified) | Final Map |
|-----------|---------------------------|----------------------------|-----------|
| 0         | 0                         | 1                          |           |
| 1         | 1                         | 3                          |           |
| 2         | 1                         | 4                          |           |
| 3         | 2                         | 6                          |           |
| 4         | 3                         | 7                          |           |
| 5         | 4                         | 7                          |           |
| 6         | 6                         | 7                          |           |
| 7         | 7                         | 7                          |           |

# Histogram Matching Algorithm (Algorithmic)

- **Step 2:** Calculate the inverse of the specified
  - Pick one by one the values from “Transformation (original)” column and find it in the “Transformation (specified)” column and note down the index, i.e., the value in the “Intensity” column

| Intensity | Transformation (original) | Transformation (specified) | Final Map |
|-----------|---------------------------|----------------------------|-----------|
| 0         | 0                         | 1                          |           |
| 1         | 1                         | 3                          |           |
| 2         | 1                         | 4                          |           |
| 3         | 2                         | 6                          |           |
| 4         | 3                         | 7                          |           |
| 5         | 4                         | 7                          |           |
| 6         | 6                         | 7                          |           |
| 7         | 7                         | 7                          |           |

# Histogram Matching Algorithm (Algorithmic)

- **Step 2:** Calculate the inverse of the specified
  - Pick one by one the values from “Transformation (original)” column and find it in the “Transformation (specified)” column and note down the index, i.e., the value in the “Intensity” column

| Intensity | Transformation (original) | Transformation (specified) | Final Map |
|-----------|---------------------------|----------------------------|-----------|
| 0         | 0                         | 1                          |           |
| 1         | 1                         | 3                          | 0         |
| 2         | 1                         | 4                          |           |
| 3         | 2                         | 6                          |           |
| 4         | 3                         | 7                          |           |
| 5         | 4                         | 7                          |           |
| 6         | 6                         | 7                          |           |
| 7         | 7                         | 7                          |           |

# Histogram Matching Algorithm (Algorithmic)

- **Step 2:** Calculate the inverse of the specified
  - Pick one by one the values from “Transformation (original)” column and find it in the “Transformation (specified)” column and note down the index, i.e., the value in the “Intensity” column

| Intensity | Transformation (original) | Transformation (specified) | Final Map |
|-----------|---------------------------|----------------------------|-----------|
| 0         | 0                         | 1                          |           |
| 1         | 1                         | 3                          | 0         |
| 2         | 1                         | 4                          |           |
| 3         | 2                         | 6                          |           |
| 4         | 3                         | 7                          |           |
| 5         | 4                         | 7                          |           |
| 6         | 6                         | 7                          |           |
| 7         | 7                         | 7                          |           |

# Histogram Matching Algorithm (Algorithmic)

- **Step 2:** Calculate the inverse of the specified
  - Pick one by one the values from “Transformation (original)” column and find it in the “Transformation (specified)” column and note down the index, i.e., the value in the “Intensity” column

| Intensity | Transformation (original) | Transformation (specified) | Final Map |
|-----------|---------------------------|----------------------------|-----------|
| 0         | 0                         | 1                          |           |
| 1         | 1                         | 3                          | 0         |
| 2         | 1                         | 4                          | 0         |
| 3         | 2                         | 6                          |           |
| 4         | 3                         | 7                          |           |
| 5         | 4                         | 7                          |           |
| 6         | 6                         | 7                          |           |
| 7         | 7                         | 7                          |           |

# Histogram Matching Algorithm (Algorithmic)

- **Step 2:** Calculate the inverse of the specified
  - Pick one by one the values from “Transformation (original)” column and find it in the “Transformation (specified)” column and note down the index, i.e., the value in the “Intensity” column

| Intensity | Transformation (original) | Transformation (specified) | Final Map |
|-----------|---------------------------|----------------------------|-----------|
| 0         | 0                         | 1                          |           |
| 1         | 1                         | 3                          | 0         |
| 2         | 1                         | 4                          | 0         |
| 3         | 2                         | 6                          |           |
| 4         | 3                         | 7                          |           |
| 5         | 4                         | 7                          |           |
| 6         | 6                         | 7                          |           |
| 7         | 7                         | 7                          |           |

# Histogram Matching Algorithm (Algorithmic)

- **Step 2:** Calculate the inverse of the specified
  - Pick one by one the values from “Transformation (original)” column and find it in the “Transformation (specified)” column and note down the index, i.e., the value in the “Intensity” column

| Intensity | Transformation (original) | Transformation (specified) | Final Map |
|-----------|---------------------------|----------------------------|-----------|
| 0         | 0                         | 1                          |           |
| 1         | 1                         | 3                          | 0         |
| 2         | 1                         | 4                          | 0         |
| 3         | 2                         | 6                          |           |
| 4         | 3                         | 7                          | 1         |
| 5         | 4                         | 7                          |           |
| 6         | 6                         | 7                          |           |
| 7         | 7                         | 7                          |           |

# Histogram Matching Algorithm (Algorithmic)

- **Step 2:** Calculate the inverse of the specified
  - Pick one by one the values from “Transformation (original)” column and find it in the “Transformation (specified)” column and note down the index, i.e., the value in the “Intensity” column

| Intensity | Transformation (original) | Transformation (specified) | Final Map |
|-----------|---------------------------|----------------------------|-----------|
| 0         | 0                         | 1                          |           |
| 1         | 1                         | 3                          | 0         |
| 2         | 1                         | 4                          | 0         |
| 3         | 2                         | 6                          |           |
| 4         | 3                         | 7                          | 1         |
| 5         | 4                         | 7                          |           |
| 6         | 6                         | 7                          |           |
| 7         | 7                         | 7                          |           |

# Histogram Matching Algorithm (Algorithmic)

- **Step 2:** Calculate the inverse of the specified
  - Pick one by one the values from “Transformation (original)” column and find it in the “Transformation (specified)” column and note down the index, i.e., the value in the “Intensity” column

| Intensity | Transformation (original) | Transformation (specified) | Final Map |
|-----------|---------------------------|----------------------------|-----------|
| 0         | 0                         | 1                          |           |
| 1         | 1                         | 3                          | 0         |
| 2         | 1                         | 4                          | 0         |
| 3         | 2                         | 6                          |           |
| 4         | 3                         | 7                          | 1         |
| 5         | 4                         | 7                          | 2         |
| 6         | 6                         | 7                          |           |
| 7         | 7                         | 7                          |           |

# Histogram Matching Algorithm (Algorithmic)

- **Step 2:** Calculate the inverse of the specified
  - Pick one by one the values from “Transformation (original)” column and find it in the “Transformation (specified)” column and note down the index, i.e., the value in the “Intensity” column

| Intensity | Transformation (original) | Transformation (specified) | Final Map |
|-----------|---------------------------|----------------------------|-----------|
| 0         | 0                         | 1                          |           |
| 1         | 1                         | 3                          | 0         |
| 2         | 1                         | 4                          | 0         |
| 3         | 2                         | 6                          |           |
| 4         | 3                         | 7                          | 1         |
| 5         | 4                         | 7                          | 2         |
| 6         | 6                         | 7                          |           |
| 7         | 7                         | 7                          |           |

# Histogram Matching Algorithm (Algorithmic)

- **Step 2:** Calculate the inverse of the specified
  - Pick one by one the values from “Transformation (original)” column and find it in the “Transformation (specified)” column and note down the index, i.e., the value in the “Intensity” column

| Intensity | Transformation (original) | Transformation (specified) | Final Map |
|-----------|---------------------------|----------------------------|-----------|
| 0         | 0                         | 1                          |           |
| 1         | 1                         | 3                          | 0         |
| 2         | 1                         | 4                          | 0         |
| 3         | 2                         | 6                          |           |
| 4         | 3                         | 7                          | 1         |
| 5         | 4                         | 7                          | 2         |
| 6         | 6                         | 7                          | 3         |
| 7         | 7                         | 7                          |           |

# Histogram Matching Algorithm (Algorithmic)

- **Step 2:** Calculate the inverse of the specified
  - What to do with other intensity values?

| Intensity | Transformation (original) | Transformation (specified) | Final Map |
|-----------|---------------------------|----------------------------|-----------|
| 0         | 0                         | 1                          |           |
| 1         | 1                         | 3                          | 0         |
| 2         | 1                         | 4                          | 0         |
| 3         | 2                         | 6                          |           |
| 4         | 3                         | 7                          | 1         |
| 5         | 4                         | 7                          | 2         |
| 6         | 6                         | 7                          | 3         |
| 7         | 7                         | 7                          |           |

# Histogram Matching Algorithm (Algorithmic)

- **Step 2:** Calculate the inverse of the specified
  - If the value doesn't exist in the "Transformation (specified)" column, then find the index of its nearest one

| Intensity | Transformation (original) | Transformation (specified) | Final Map |
|-----------|---------------------------|----------------------------|-----------|
| 0         | 0                         | 1                          | 0         |
| 1         | 1                         | 3                          | 0         |
| 2         | 1                         | 4                          | 0         |
| 3         | 2                         | 6                          |           |
| 4         | 3                         | 7                          | 1         |
| 5         | 4                         | 7                          | 2         |
| 6         | 6                         | 7                          | 3         |
| 7         | 7                         | 7                          |           |

# Histogram Matching Algorithm (Algorithmic)

- **Step 2:** Calculate the inverse of the specified
  - If multiple nearest values exist in the “Transformation (specified)” column, then pick the one which is greater

| Intensity | Transformation (original) | Transformation (specified) | Final Map |
|-----------|---------------------------|----------------------------|-----------|
| 0         | 0                         | 1                          | 0         |
| 1         | 1                         | 3                          | 0         |
| 2         | 1                         | 4                          | 0         |
| 3         | 2                         | 6                          | 1         |
| 4         | 3                         | 7                          | 1         |
| 5         | 4                         | 7                          | 2         |
| 6         | 6                         | 7                          | 3         |
| 7         | 7                         | 7                          |           |

# Histogram Matching Algorithm (Algorithmic)

- **Step 2:** Calculate the inverse of the specified
  - If multiple same values exist in the “Transformation (specified)” column, then pick the one which has lowest index (i.e., “Intensity”) value

| Intensity | Transformation (original) | Transformation (specified) | Final Map |
|-----------|---------------------------|----------------------------|-----------|
| 0         | 0                         | 1                          | 0         |
| 1         | 1                         | 3                          | 0         |
| 2         | 1                         | 4                          | 0         |
| 3         | 2                         | 6                          | 1         |
| 4         | 3                         | 7                          | 1         |
| 5         | 4                         | 7                          | 2         |
| 6         | 6                         | 7                          | 3         |
| 7         | 7                         | 7                          | 4         |



# Histogram matching - examples



Original

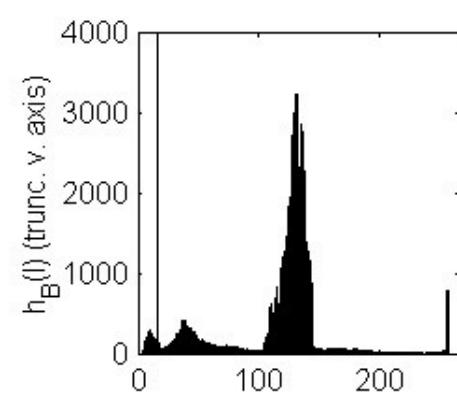
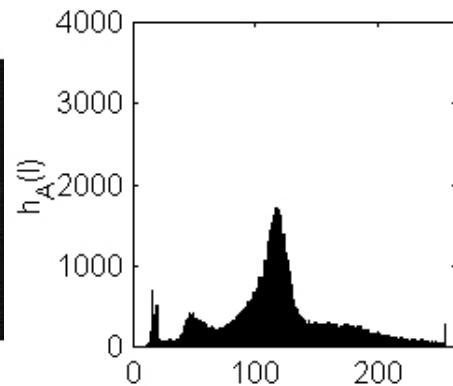
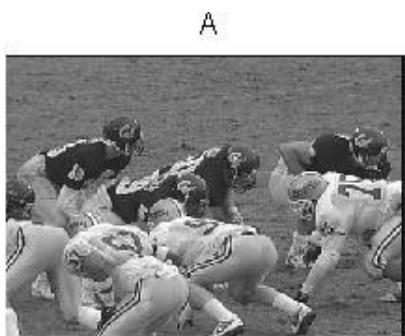


Specified

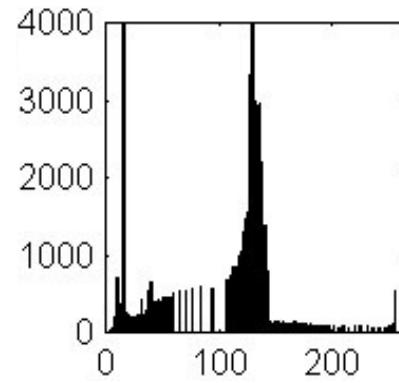


Matched

# Histogram matching - examples



C (hist. matched to B)



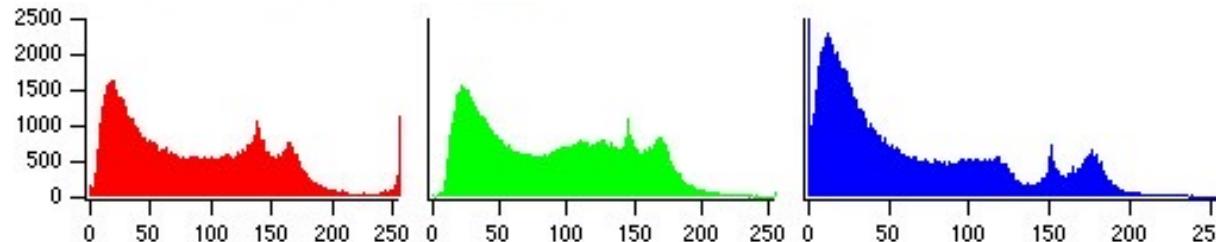
# Outline

---

- Image types (Binary, Grey scale, color)
- Processing images
  - Spatial vs. Frequency domain
  - Intensity transformations vs. Spatial filtering
- Basic intensity transformations
  - Image negatives,
  - Log transformations,
  - Power-law or Gamma transformations
  - Piecewise linear transformations
  - Bit-plane slicing
- Image histogram (**How to apply to color images?**)
  - Histogram equalization,
  - Histogram matching (or histogram specification)

# Understanding Colour Histogram

- **Colour Histogram** represents the distribution of colors in an image.
  - Left: no color (0% color saturation), right representing full color (100% color saturation).
  - For a colour image with RGB space, we have 3 separate curves, for each colour: R, G and B.



# Histogram Equalisation on RGB Images

---

- How to perform Histogram Equalisation on color images?
  - **Approach 1:** Channel splitting and equalizing each channel separately
- Is this approach appropriate? Why?
  - NO. Histogram equalization involves intensity values of the image, not the color components
- So for a simple RGB colour image, Histogram Equalisation should not be applied individually on each channel.
- Rather, it should be applied such that intensity values are equalized without disturbing the color balance of the image.

# Histogram Equalisation on RGB Images

---

- How to perform Histogram Equalisation on colour images?
  - **Approach 2:** Convert the color space of the image from RGB into one of the color spaces which separate intensity values from color components. Then, perform histogram equalisation of the intensity plane. Finally, convert the image back to RGB.
- Some of these colour spaces are:
  - HSV
  - YUV
  - YCbCr
- YCbCr is preferred as it is designed for digital images.

## RGB to YCbCr

---

- YCbCr colour space separates brightness Y from chromaticity CbCr.

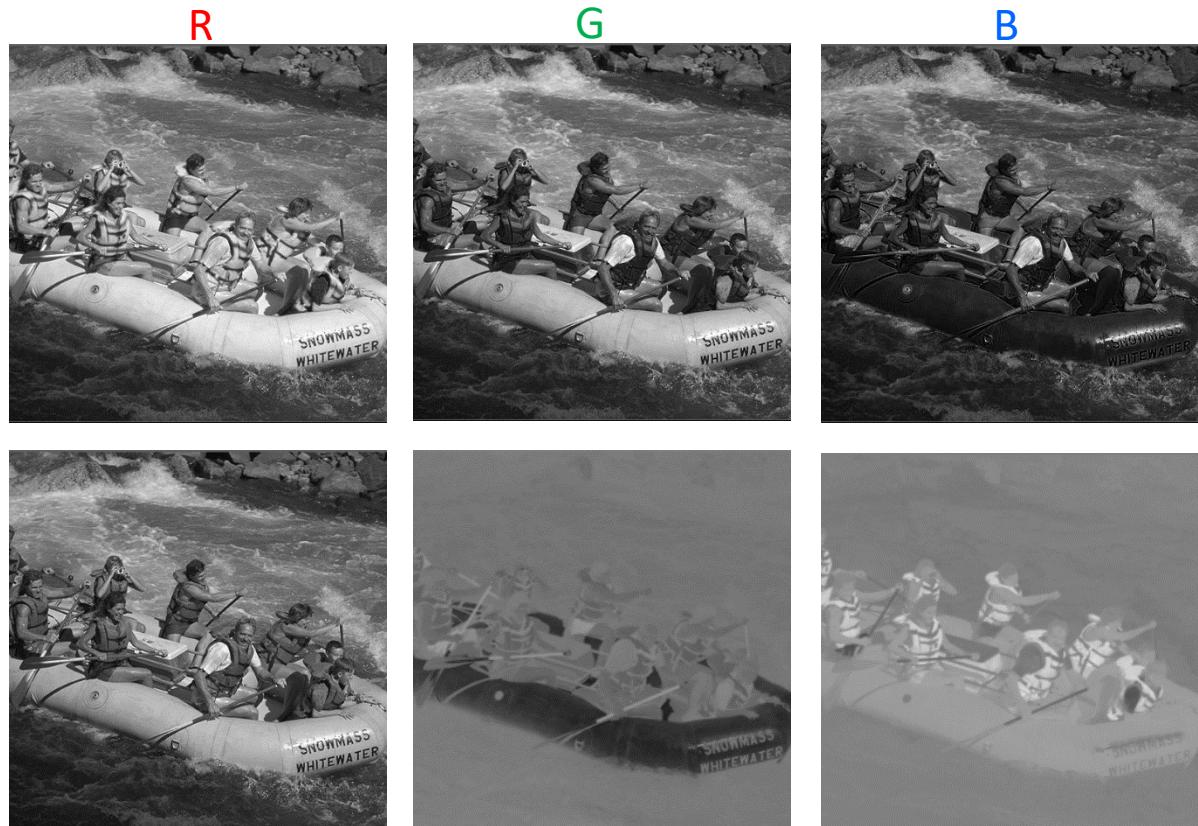
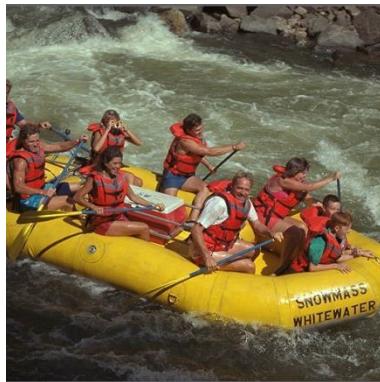
$$C_b = -0.168R - 0.331G + 0.500B + 128$$

$$C_r = 0.500R - 0.419G - 0.081B + 128$$

$$Y = 0.299R + 0.587G + 0.114B$$

# RGB vs. YCbCr

- YCbCr colour space separates brightness Y from chromaticity CbCr.



Y

C<sub>b</sub>

C<sub>r</sub>

# Histogram Equalisation on RGB Images

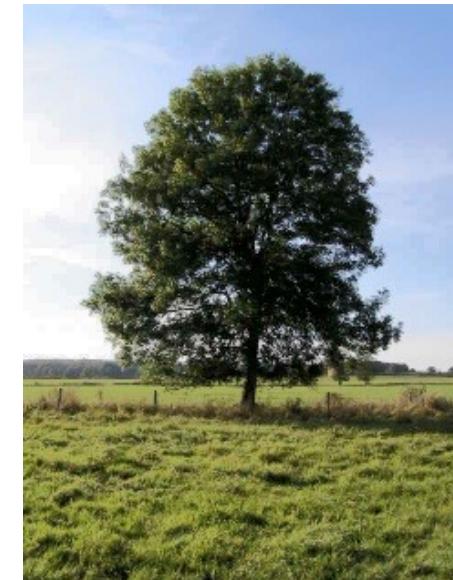
- How to perform Histogram Equalisation on colour images?
  - **Approach 1:** Channel splitting and equalizing each channel separately
  - **Approach 2:** Convert the colour space of the image from RGB into one of the colour spaces which separate intensity values from colour components. Then, perform histogram equalisation of the intensity plane. Finally, convert the image back to RGB.



Original Image



Approach 1



Approach 2

- **What we have seen**

- What is an image histogram and how to create image histograms for gray-scale and color images
- How histogram equalization works and how to apply it to color images
- How histogram matching works
  - It enables the matching of the histogram of one image to the histogram of another image

# SCC.366: Media Coding & Processing

2022-2023

Week 04 – Lecture 1

## Image Transformation - Spatial Filtering

Dr. Hossein Rahmani

Senior Lecturer in Data Science

Email and Team: [h.rahmani@lancaster.ac.uk](mailto:h.rahmani@lancaster.ac.uk)

# Outline

---

- Kernel/Filter
- Smoothing vs. Sharpening
- Smoothing Spatial Filters
  - Linear Filters (Mean filters)
    - Averaging filter
    - Weighted averaging filter (e.g., Gaussian filter)
  - Order Statistics (Non-linear) filters
    - Minimum filter
    - Maximum filter
    - Median filter
- Sharpening Spatial Filters
  - Unsharp masking
  - First order derivative
  - Second order derivative (Laplacian)

# Review - Some Definitions

---

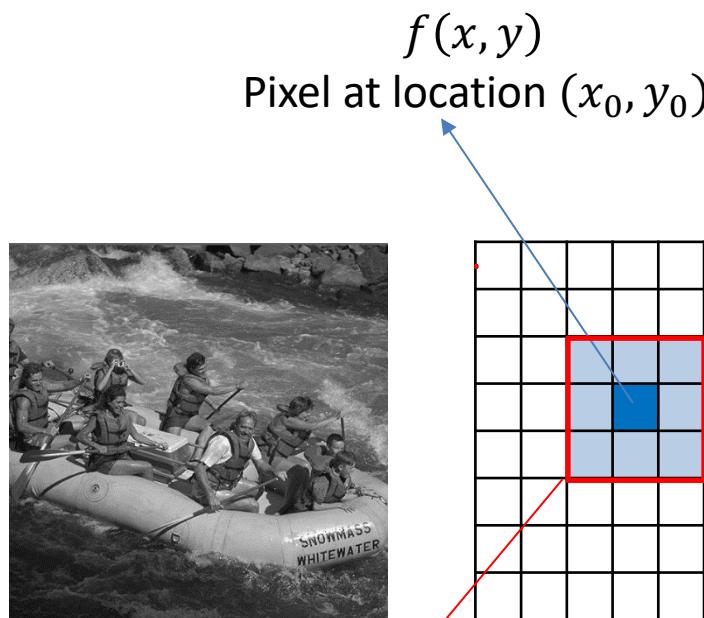
- **Spatial domain:**
  - The image plane itself
  - Processing in the spatial domain deals with direct manipulation of the image pixels
- Types of transformations that can be applied to an image
  - **Intensity transformations** (Operate on single pixels)
  - **Spatial filtering** (Working on a neighbourhood of every pixel)

# Outline

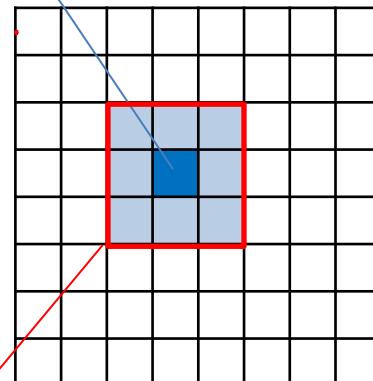
---

- **Kernel/Filter**
- Smoothing vs. Sharpening
- Smoothing Spatial Filters
  - Linear Filters (Mean filters)
    - Averaging filter
    - Weighted averaging filter (e.g., Gaussian filter)
  - Order Statistics (Non-linear) filters
    - Minimum filter
    - Maximum filter
    - Median filter
- Sharpening Spatial Filters
  - Unsharp masking
  - First order derivative
  - Second order derivative (Laplacian)

# Spatial Domain (Spatial Filtering)

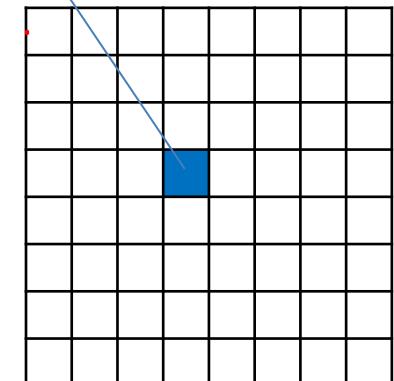


3×3  
Neighbourhood of  
Pixel



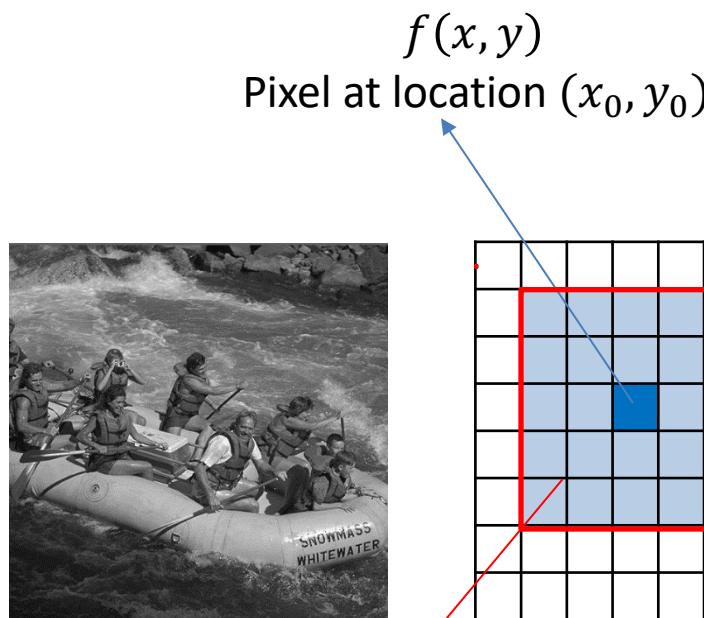
$g(x, y)$   
Pixel at location  $(x_0, y_0)$

$$g(x, y) = T[f(x, y)]$$

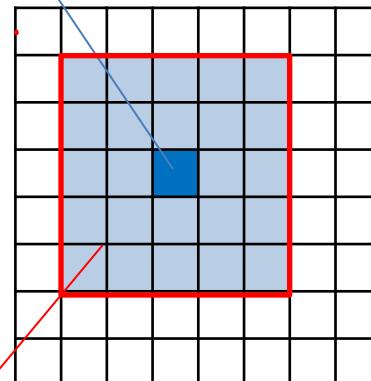


- $f$  and  $g$  are 2D matrices of size  $W \times H$
- $T$  is a matrix of size  $w \times h$  which is much smaller than  $W \times H$
- The value of each pixel of  $g$  is the result of applying the operator  $T$  at 5 the corresponding location in  $f$ .

# Spatial Domain (Spatial Filtering)

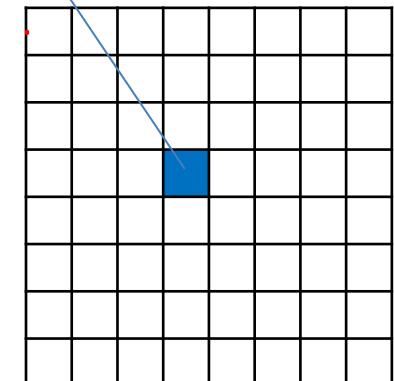


5×5  
Neighbourhood of  
Pixel



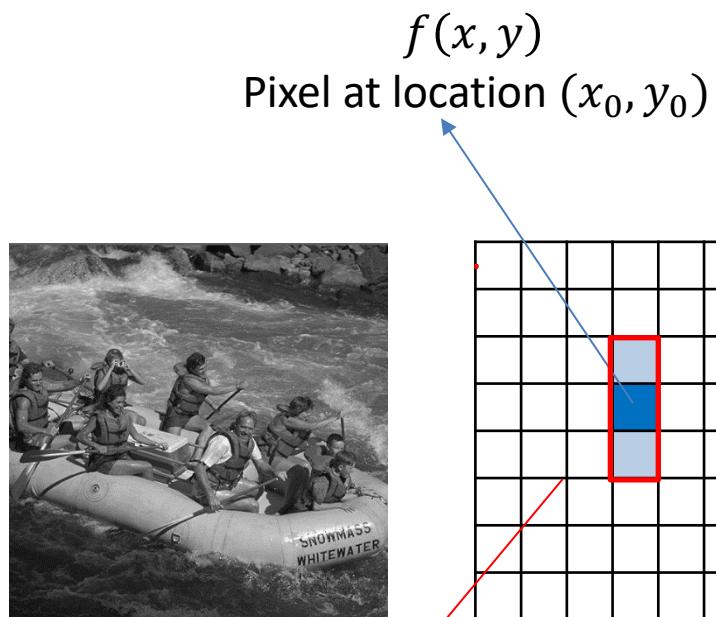
$g(x, y)$   
Pixel at location  $(x_0, y_0)$

$$g(x, y) = T[f(x, y)]$$

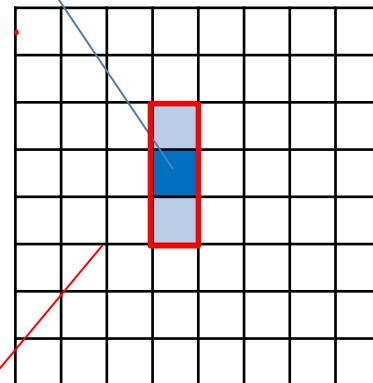


- $f$  and  $g$  are 2D matrices of size  $W \times H$
- $T$  is a matrix of size  $w \times h$  which is much smaller than  $W \times H$
- The value of each pixel of  $g$  is the result of applying the operator  $T$  at <sup>6</sup> the corresponding location in  $f$ .

# Spatial Domain (Spatial Filtering)

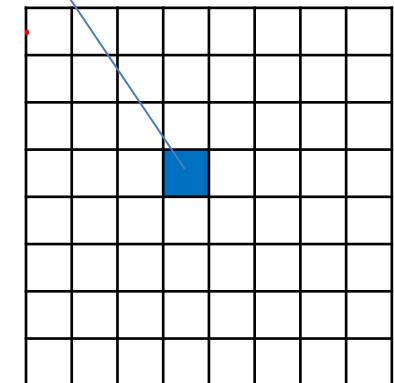


3x1  
Neighbourhood of  
Pixel



$g(x, y)$   
Pixel at location  $(x_0, y_0)$

$$g(x, y) = T[f(x, y)]$$

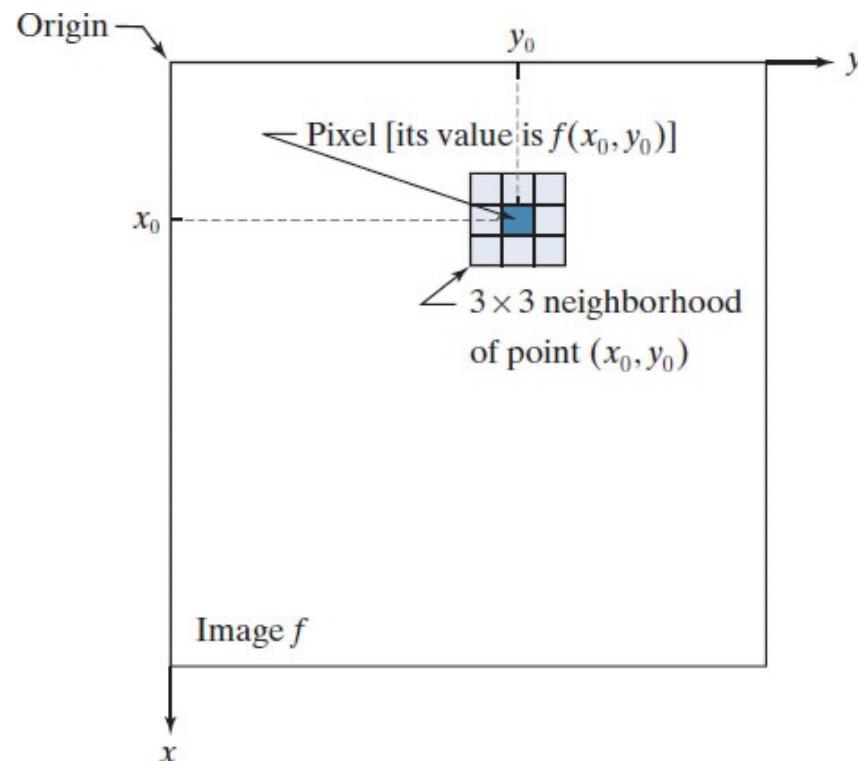


- $f$  and  $g$  are 2D matrices of size  $W \times H$
- $T$  is a matrix of size  $w \times h$  which is much smaller than  $W \times H$
- The value of each pixel of  $g$  is the result of applying the operator  $T$  at <sup>7</sup> the corresponding location in  $f$ .

# Spatial Filters

Need to define:

- ✓ A neighborhood
- ✓ A Mask/Filter/Kernel



- Typically, the neighborhood is rectangular and its size is much smaller than that of image  $f(x,y)$ - e.g., 3x3 or 5x5
- Can we have a filter of size 2x2 or 4x4 etc.?

# Outline

---

- Kernel/Filter
- **Smoothing vs. Sharpening**
- Smoothing Spatial Filters
  - Linear Filters (Mean filters)
    - Averaging filter
    - Weighted averaging filter (e.g., Gaussian filter)
  - Order Statistics (Non-linear) filters
    - Minimum filter
    - Maximum filter
    - Median filter
- Sharpening Spatial Filters
  - Unsharp masking
  - First order derivative
  - Second order derivative (Laplacian)

# Types of Spatial Filters

---

- **Smoothing Spatial Filters** are used for blurring and noise reduction in the image.
- Blurring is pre-processing steps for **removal of small details**
- **Noise Reduction** is accomplished by blurring.
  - Linear Filters (Mean filters)
    - Averaging filter
    - Weighted averaging filter (e.g., Gaussian filter)
  - Order Statistics (Non-linear) filters
    - Minimum filter
    - Maximum filter
    - Median filter

# Types of Spatial Filters

---

- **Sharpening Spatial Filter** is also known as derivative filter.
- The purpose of the sharpening spatial filter is just the **opposite** of the smoothing spatial filter.
- Its main focus is on the removal of blurring and **highlight the edges**.
- Types of sharpening filters:
  - Unsharp masking
  - First order derivative
  - Second order derivative

# Outline

---

- Kernel/Filter
- Smoothing vs. Sharpening
- Smoothing Spatial Filters
  - **Linear Filters (Mean filters)**
    - Averaging filter
    - Weighted averaging filter (e.g., Gaussian filter)
  - Order Statistics (Non-linear) filters
    - Minimum filter
    - Maximum filter
    - Median filter
- Sharpening Spatial Filters
  - Unsharp masking
  - First order derivative
  - Second order derivative (Laplacian)

# Linear Filters

- **Linear Filters (Mean Filter)**

- **Averaging filter:** It is used in reduction of the detail in image. All coefficients are equal.



Image:  $F$

$$\frac{1}{9} \times \begin{matrix} & \text{Filter (3,3)} \\ \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array} & \text{OR} \\ \begin{array}{|c|c|c|} \hline \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \hline \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \hline \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \hline \end{array} & \begin{array}{|c|c|c|} \hline \end{array} \end{matrix}$$

Filter/Kernel:  $H$

# Linear Filters

---

- **Linear Filters (Mean Filter)**

- **Averaging filter:** It is used in reduction of the detail in image. All coefficients are equal.

|   |   |   |   |   |
|---|---|---|---|---|
| 3 | 3 | 3 | 1 | 0 |
| 2 | 1 | 2 | 2 | 0 |
| 2 | 0 | 2 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 2 | 1 |

Image:  $F$

$$\frac{1}{9} \times \begin{matrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{matrix}$$

Filter (3,3)

Filter/Kernel:  $H$

# Linear Filters

- **Linear Filters (Mean Filter)**

- **Averaging filter:** It is used in reduction of the detail in image. All coefficients are equal.

|   |   |   |   |   |
|---|---|---|---|---|
| 3 | 3 | 3 | 1 | 0 |
| 2 | 1 | 2 | 2 | 0 |
| 2 | 0 | 2 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 2 | 1 |

Image:  $F$

$$\frac{1}{9} \times \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$

Filter (3,3)

$$\begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$

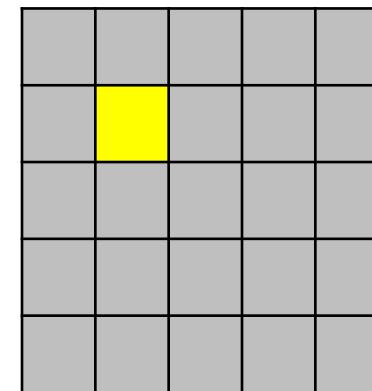
Filter/Kernel:  $H$

# Linear Filter

- **Linear Filter (Mean Filter)**

- **Averaging filter:** It is used in reduction of the detail in image. All coefficients are equal.

$$\frac{1}{9} \times \begin{array}{|c|c|c|c|c|} \hline 3 & 1 & 3 & 1 & 3 & 1 \\ \hline 2 & 1 & 1 & 1 & 2 & 1 \\ \hline 2 & 1 & 0 & 1 & 2 & 1 \\ \hline 1 & 0 & 0 & 0 & 0 & 1 \\ \hline 1 & 1 & 0 & 2 & 1 & 1 \\ \hline \end{array}$$



$$\frac{1}{9} \times \begin{array}{|c|c|c|} \hline 3 \times 1 & 3 \times 1 & 3 \times 1 \\ \hline 2 \times 1 & 1 \times 1 & 2 \times 1 \\ \hline 2 \times 1 & 0 \times 1 & 2 \times 1 \\ \hline \end{array}$$

Output Image:  $G$

Sum

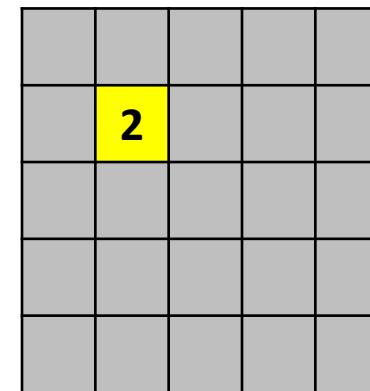
$$\frac{1}{9} \times (3 + 3 + 3 + 2 + 1 + 2 + 2 + 0 + 2) = 2$$

# Linear Filter

- **Linear Filter (Mean Filter)**

- **Averaging filter:** It is used in reduction of the detail in image. All coefficients are equal.

$$\frac{1}{9} \times \begin{array}{|c|c|c|c|c|} \hline 3 & 1 & 3 & 1 & 3 & 1 \\ \hline 2 & 1 & 1 & 1 & 2 & 1 \\ \hline 2 & 1 & 0 & 1 & 2 & 1 \\ \hline 1 & 0 & 0 & 0 & 0 & 1 \\ \hline 1 & 1 & 0 & 2 & 1 & 1 \\ \hline \end{array}$$



$$\frac{1}{9} \times \begin{array}{|c|c|c|} \hline 3 \times 1 & 3 \times 1 & 3 \times 1 \\ \hline 2 \times 1 & 1 \times 1 & 2 \times 1 \\ \hline 2 \times 1 & 0 \times 1 & 2 \times 1 \\ \hline \end{array}$$

Output Image:  $G$

Sum

$$\frac{1}{9} \times (3 + 3 + 3 + 2 + 1 + 2 + 2 + 0 + 2) = 2$$

# Linear Filters

- Handling pixels close to boundaries
  - Solution: Pad with zeroes

|   |   |   |   |   |
|---|---|---|---|---|
| 3 | 3 | 3 | 1 | 0 |
| 2 | 1 | 2 | 2 | 0 |
| 2 | 0 | 2 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 2 | 1 |

Image:  $F$

Filter (3,3)

$$\frac{1}{9} \times \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$

Filter/Kernel:  $H$

# Linear Filters

- Handling pixels close to boundaries
  - Solution: Pad with zeroes

|   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 3 | 3 | 3 | 1 | 0 | 0 | 0 |
| 0 | 2 | 1 | 2 | 2 | 0 | 0 | 0 |
| 0 | 2 | 0 | 2 | 0 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 | 2 | 1 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Image:  $F$

Filter (3,3)

$$\frac{1}{9} \times$$

|   |   |   |
|---|---|---|
| 1 | 1 | 1 |
| 1 | 1 | 1 |
| 1 | 1 | 1 |

Filter/Kernel:  $H$

# Linear Filters

- Handling pixels close to boundaries
  - Solution: Pad with zeroes

|   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 |
| 0 | 3 | 1 | 3 | 1 | 3 | 1 | 1 | 0 |
| 0 | 2 | 1 | 1 | 1 | 2 | 1 | 2 | 0 |
| 0 | 2 | 0 | 0 | 2 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 | 2 | 1 | 1 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

|     |     |     |
|-----|-----|-----|
| 0×1 | 0×1 | 0×1 |
| 3×1 | 3×1 | 3×1 |
| 2×1 | 1×1 | 2×1 |

$$\frac{1}{9} \times \begin{matrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{matrix}$$

Filter (3,3)

$$\frac{1}{9} \times (0 + 0 + 0 + 3 + 3 + 3 + 2 + 1 + 2) = \dots$$

# Convolution

---

- This process is called a **convolution** operation

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

- And is written as:

$$G = H * F$$

- Where,  $H$  is a filter/kernel and  $F$  is an image

# Convolution (Example)

---

$$\begin{array}{c}
 \begin{matrix} & & \\ & & \\ & & \end{matrix} \\
 H
 \end{array}
 \quad *
 \quad
 \begin{matrix}
 \begin{matrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 90 & 90 & 90 & 90 & 90 & 0 \\ 0 & 0 & 90 & 90 & 90 & 90 & 90 & 0 \\ 0 & 0 & 90 & 90 & 90 & 90 & 90 & 0 \\ 0 & 0 & 90 & 0 & 90 & 90 & 90 & 0 \\ 0 & 0 & 90 & 90 & 90 & 90 & 90 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 90 & 0 & 0 & 0 & 0 & 0 & 0 \end{matrix} \\
 F
 \end{matrix}
 =
 \begin{matrix}
 \begin{matrix} 0 & 10 & 20 & 30 & 30 & 30 & 20 & 10 \\ 0 & 20 & 40 & 60 & 60 & 60 & 40 & 20 \\ 0 & 30 & 60 & 90 & 90 & 90 & 60 & 30 \\ 0 & 30 & 50 & 80 & 80 & 90 & 60 & 30 \\ 0 & 30 & 50 & 80 & 80 & 90 & 60 & 30 \\ 0 & 20 & 30 & 50 & 50 & 60 & 40 & 20 \\ 10 & 20 & 30 & 30 & 30 & 30 & 20 & 10 \\ 10 & 10 & 10 & 0 & 0 & 0 & 0 & 0 \end{matrix} \\
 G
 \end{matrix}$$

# Convolution (Example)

---

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

|   |    |    |    |    |    |    |    |   |
|---|----|----|----|----|----|----|----|---|
| 0 | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0 |
| 0 | 0  | 90 | 90 | 90 | 90 | 90 | 90 | 0 |
| 0 | 0  | 90 | 90 | 90 | 90 | 90 | 90 | 0 |
| 0 | 0  | 90 | 90 | 90 | 90 | 90 | 90 | 0 |
| 0 | 0  | 90 | 90 | 0  | 90 | 90 | 90 | 0 |
| 0 | 0  | 90 | 90 | 90 | 90 | 90 | 90 | 0 |
| 0 | 0  | 0  | 0  | 0  | 0  | 0  | 0  | 0 |
| 0 | 90 | 0  | 0  | 0  | 0  | 0  | 0  | 0 |

=

|    |    |    |    |    |    |    |    |
|----|----|----|----|----|----|----|----|
| 0  | 10 | 20 | 30 | 30 | 30 | 20 | 10 |
| 0  | 20 | 40 | 60 | 60 | 60 | 40 | 20 |
| 0  | 30 | 60 | 90 | 90 | 90 | 60 | 30 |
| 0  | 30 | 50 | 80 | 80 | 90 | 60 | 30 |
| 0  | 30 | 50 | 80 | 80 | 90 | 60 | 30 |
| 0  | 20 | 30 | 50 | 50 | 60 | 40 | 20 |
| 10 | 20 | 30 | 30 | 30 | 30 | 20 | 10 |
| 10 | 10 | 10 | 0  | 0  | 0  | 0  | 0  |

*F*

*G*

# Convolution (Example)

- Pad with zeroes

$$\frac{1}{9} \times \begin{matrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{matrix} * \begin{matrix} H \\ F \end{matrix}$$

|   |   |   |    |    |    |    |    |   |   |
|---|---|---|----|----|----|----|----|---|---|
| 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 | 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 | 0 | 0  | 0  | 0  | 0  | 0  | 0 | 0 |

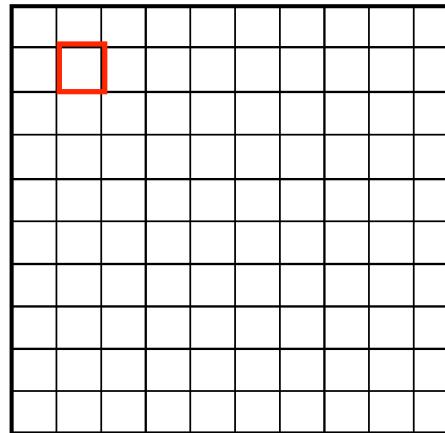
- Question:** If size of the kernel/filter is  $5 \times 5$ , how many rows/columns of zeros should be added?

# Convolution (Example)

$F[x, y]$

|   |   |    |  |    |    |    |    |    |    |   |   |
|---|---|----|--|----|----|----|----|----|----|---|---|
| 0 | 0 | 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 | 90 | 0 | 0 |
| 0 | 0 | 0  |  | 90 | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0  |  | 90 | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0  |  | 90 | 0  | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0  |  | 90 | 90 | 90 | 90 | 90 | 90 | 0 | 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  | 0 | 0 |

$G[x, y]$





# Convolution (Example)

$F[x, y]$

|   |   |    |    |    |    |    |    |    |   |   |   |
|---|---|----|----|----|----|----|----|----|---|---|---|
| 0 | 0 | 0  | 0  | 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 | 90 | 0 | 0 | 0 |
| 0 | 0 | 0  | 90 | 90 | 90 | 90 | 90 | 90 | 0 | 0 | 0 |
| 0 | 0 | 0  | 90 | 90 | 90 | 90 | 90 | 90 | 0 | 0 | 0 |
| 0 | 0 | 0  | 90 | 0  | 90 | 90 | 90 | 90 | 0 | 0 | 0 |
| 0 | 0 | 0  | 90 | 90 | 90 | 90 | 90 | 90 | 0 | 0 | 0 |
| 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  | 0  | 0 | 0 | 0 |

$G[x, y]$

|  |  |  |   |    |  |  |  |  |  |  |  |
|--|--|--|---|----|--|--|--|--|--|--|--|
|  |  |  |   |    |  |  |  |  |  |  |  |
|  |  |  | 0 | 10 |  |  |  |  |  |  |  |
|  |  |  |   |    |  |  |  |  |  |  |  |
|  |  |  |   |    |  |  |  |  |  |  |  |
|  |  |  |   |    |  |  |  |  |  |  |  |
|  |  |  |   |    |  |  |  |  |  |  |  |
|  |  |  |   |    |  |  |  |  |  |  |  |
|  |  |  |   |    |  |  |  |  |  |  |  |
|  |  |  |   |    |  |  |  |  |  |  |  |
|  |  |  |   |    |  |  |  |  |  |  |  |

# Convolution (Example)

---

$F[x, y]$

|   |   |    |   |    |    |    |    |    |    |   |   |
|---|---|----|---|----|----|----|----|----|----|---|---|
| 0 | 0 | 0  | 0 | 0  | 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 | 90 | 0 | 0 |
| 0 | 0 | 0  | 0 | 90 | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0  | 0 | 90 | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0  | 0 | 90 | 0  | 90 | 90 | 90 | 90 | 0 | 0 |
| 0 | 0 | 0  | 0 | 90 | 90 | 90 | 90 | 90 | 90 | 0 | 0 |
| 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  | 0  | 0  | 0 | 0 |

$G[x, y]$

|  |   |    |    |  |  |  |  |  |  |  |  |
|--|---|----|----|--|--|--|--|--|--|--|--|
|  |   |    |    |  |  |  |  |  |  |  |  |
|  | 0 | 10 | 20 |  |  |  |  |  |  |  |  |
|  |   |    |    |  |  |  |  |  |  |  |  |
|  |   |    |    |  |  |  |  |  |  |  |  |
|  |   |    |    |  |  |  |  |  |  |  |  |
|  |   |    |    |  |  |  |  |  |  |  |  |
|  |   |    |    |  |  |  |  |  |  |  |  |
|  |   |    |    |  |  |  |  |  |  |  |  |
|  |   |    |    |  |  |  |  |  |  |  |  |
|  |   |    |    |  |  |  |  |  |  |  |  |

# Convolution (Example)

---

$F[x, y]$

|   |   |    |    |    |    |    |    |    |   |   |   |
|---|---|----|----|----|----|----|----|----|---|---|---|
| 0 | 0 | 0  | 0  | 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 | 90 | 0 | 0 | 0 |
| 0 | 0 | 0  | 90 | 90 | 90 | 90 | 90 | 90 | 0 | 0 | 0 |
| 0 | 0 | 0  | 90 | 90 | 90 | 90 | 90 | 90 | 0 | 0 | 0 |
| 0 | 0 | 0  | 90 | 0  | 90 | 90 | 90 | 90 | 0 | 0 | 0 |
| 0 | 0 | 0  | 90 | 90 | 90 | 90 | 90 | 90 | 0 | 0 | 0 |
| 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  | 0  | 0 | 0 | 0 |

$G[x, y]$

|  |   |    |    |    |  |  |  |  |  |  |  |
|--|---|----|----|----|--|--|--|--|--|--|--|
|  |   |    |    |    |  |  |  |  |  |  |  |
|  | 0 | 10 | 20 | 30 |  |  |  |  |  |  |  |
|  |   |    |    |    |  |  |  |  |  |  |  |
|  |   |    |    |    |  |  |  |  |  |  |  |
|  |   |    |    |    |  |  |  |  |  |  |  |
|  |   |    |    |    |  |  |  |  |  |  |  |
|  |   |    |    |    |  |  |  |  |  |  |  |
|  |   |    |    |    |  |  |  |  |  |  |  |
|  |   |    |    |    |  |  |  |  |  |  |  |
|  |   |    |    |    |  |  |  |  |  |  |  |
|  |   |    |    |    |  |  |  |  |  |  |  |

# Convolution (Example)

---

$F[x, y]$

|   |   |    |    |    |    |    |    |    |   |   |   |
|---|---|----|----|----|----|----|----|----|---|---|---|
| 0 | 0 | 0  | 0  | 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 | 90 | 0 | 0 | 0 |
| 0 | 0 | 0  | 90 | 90 | 90 | 90 | 90 | 90 | 0 | 0 | 0 |
| 0 | 0 | 0  | 90 | 90 | 90 | 90 | 90 | 90 | 0 | 0 | 0 |
| 0 | 0 | 0  | 90 | 0  | 90 | 90 | 90 | 90 | 0 | 0 | 0 |
| 0 | 0 | 0  | 90 | 90 | 90 | 90 | 90 | 90 | 0 | 0 | 0 |
| 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  | 0  | 0 | 0 | 0 |

$G[x, y]$

|  |   |    |    |    |    |
|--|---|----|----|----|----|
|  |   |    |    |    |    |
|  | 0 | 10 | 20 | 30 | 30 |
|  |   |    |    |    |    |
|  |   |    |    |    |    |
|  |   |    |    |    |    |
|  |   |    |    |    |    |
|  |   |    |    |    |    |
|  |   |    |    |    |    |
|  |   |    |    |    |    |
|  |   |    |    |    |    |

# Convolution (Example)

---

$F[x, y]$

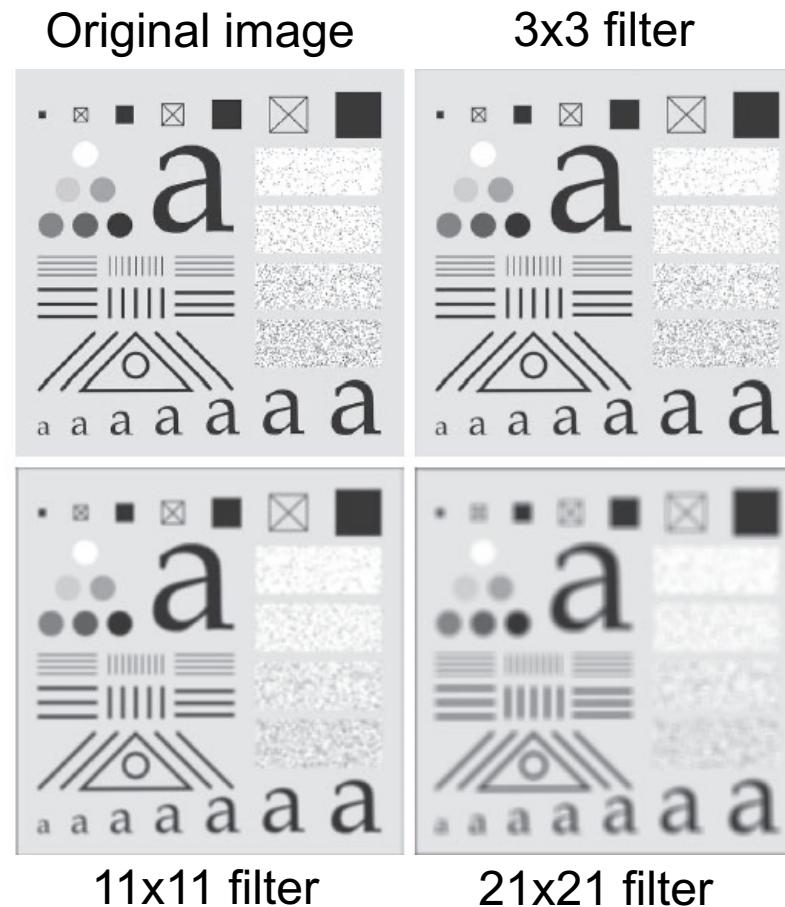
|   |   |    |    |    |    |    |    |   |   |   |   |
|---|---|----|----|----|----|----|----|---|---|---|---|
| 0 | 0 | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0 | 0 | 0 |
| 0 | 0 | 0  | 0  | 0  | 0  | 0  | 0  | 0 | 0 | 0 | 0 |
| 0 | 0 | 0  | 90 | 90 | 90 | 90 | 90 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0  | 90 | 90 | 90 | 90 | 90 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0  | 90 | 90 | 90 | 90 | 90 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0  | 90 | 0  | 90 | 90 | 90 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0  | 90 | 90 | 90 | 90 | 90 | 0 | 0 | 0 | 0 |
| 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  | 0 | 0 | 0 | 0 |

$G[x, y]$

|    |    |    |    |    |    |    |    |    |  |  |  |
|----|----|----|----|----|----|----|----|----|--|--|--|
|    |    |    |    |    |    |    |    |    |  |  |  |
|    | 0  | 10 | 20 | 30 | 30 | 30 | 20 | 10 |  |  |  |
|    | 0  | 20 | 40 | 60 | 60 | 60 | 40 | 20 |  |  |  |
|    | 0  | 30 | 60 | 90 | 90 | 90 | 60 | 30 |  |  |  |
|    | 0  | 30 | 50 | 80 | 80 | 90 | 60 | 30 |  |  |  |
|    | 0  | 30 | 50 | 80 | 80 | 90 | 60 | 30 |  |  |  |
|    | 0  | 20 | 30 | 50 | 50 | 60 | 40 | 20 |  |  |  |
| 10 | 20 | 30 | 30 | 30 | 30 | 20 | 10 |    |  |  |  |
| 10 | 10 | 10 | 0  | 0  | 0  | 0  | 0  | 0  |  |  |  |
|    |    |    |    |    |    |    |    |    |  |  |  |

# Linear Filters (averaging filter) - Example

- Mask/Filter size determines the degree of smoothing (loss of detail)



# Linear Filters (averaging filter) - Example



Original Image

$$\frac{1}{N^2} \begin{bmatrix} 1 & 1 & \cdots & 1 \\ 1 & 1 & \cdots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \cdots & 1 \end{bmatrix}_{N \times N}$$

Avg. Mask



$N = 3$



$N = 5$



$N = 7$



$N = 11$



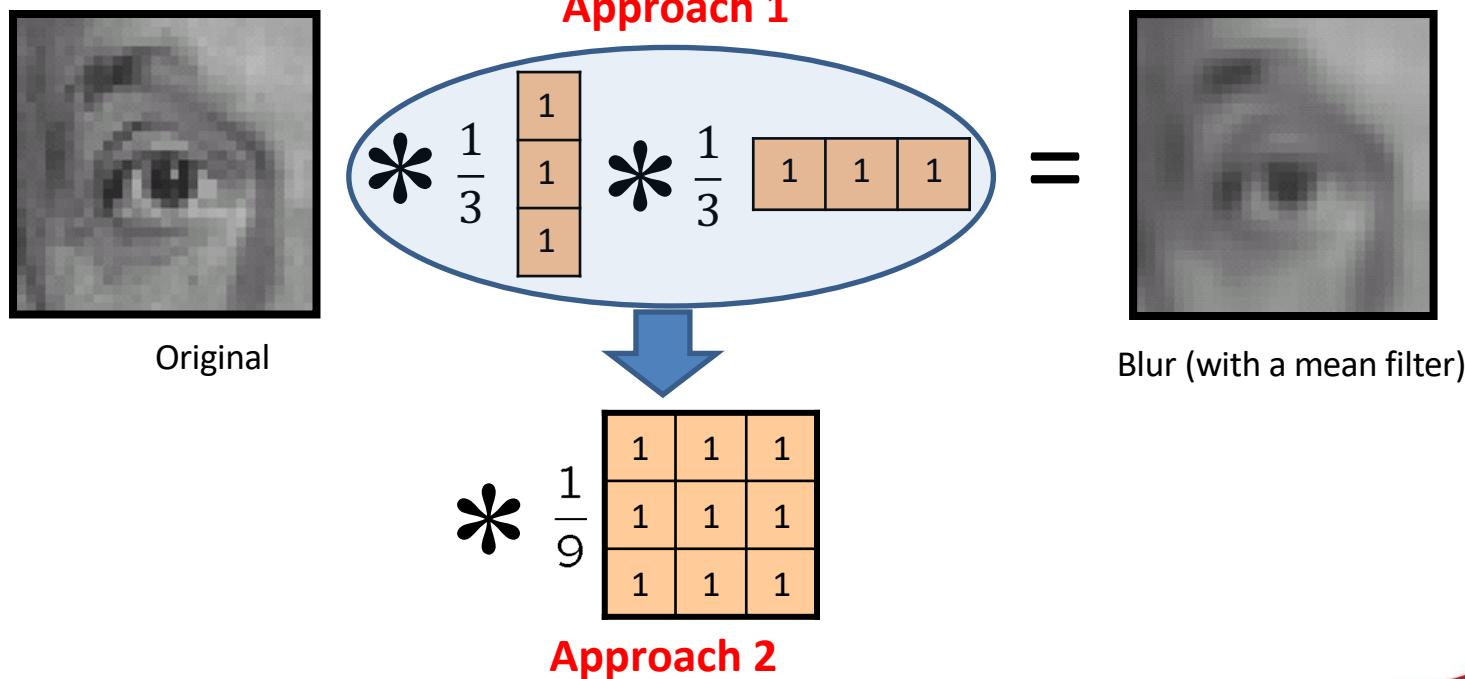
$N = 15$



$N = 21$

# Linear filters: Question#1

- Perform convolution as a two-step process
  - Convolve each column of the input image with the vertical projection of the kernel to create an intermediate image.
  - Convolve each row of the intermediate image with the horizontal projection of the kernel.
- **Question:** Consider an example image and show the below is correct



# Linear filters: Question#1

- **Solution (Approach 2):**

| Input image |          |          |          |          |
|-------------|----------|----------|----------|----------|
| $x_{24}$    | $x_{25}$ | $x_{21}$ | $x_{22}$ | $x_{23}$ |
| $x_{19}$    | $x_{20}$ | $x_{16}$ | $x_{17}$ | $x_{18}$ |
| $x_4$       | $x_5$    | $x_1$    | $x_2$    | $x_3$    |
| $x_9$       | $x_{10}$ | $x_6$    | $x_7$    | $x_8$    |
| $x_{14}$    | $x_{15}$ | $x_{11}$ | $x_{12}$ | $x_{13}$ |

 $*$ 

|               |     |     |     |
|---------------|-----|-----|-----|
| $\frac{1}{9}$ | $1$ | $1$ | $1$ |
|               | $1$ | $1$ | $1$ |
|               | $1$ | $1$ | $1$ |

 $=$ 

| Output image |     |        |     |     |
|--------------|-----|--------|-----|-----|
| ...          | ... | ...    | ... | ... |
| ...          | ... | ...    | ... | ... |
| ...          | ... | $x'_1$ | ... | ... |
| ...          | ... | ...    | ... | ... |
| ...          | ... | ...    | ... | ... |

# Linear filters: Question#1

- Solution (Approach 2):

Input image

|          |          |          |          |          |
|----------|----------|----------|----------|----------|
| $x_{24}$ | $x_{25}$ | $x_{21}$ | $x_{22}$ | $x_{23}$ |
| $x_{19}$ | $x_{20}$ | $x_{16}$ | $x_{17}$ | $x_{18}$ |
| $x_4$    | $x_5$    | $x_1$    | $x_2$    | $x_3$    |
| $x_9$    | $x_{10}$ | $x_6$    | $x_7$    | $x_8$    |
| $x_{14}$ | $x_{15}$ | $x_{11}$ | $x_{12}$ | $x_{13}$ |

$\ast \frac{1}{9} \begin{matrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{matrix} =$

Output image

|     |     |        |     |     |
|-----|-----|--------|-----|-----|
| ... | ... | ...    | ... | ... |
| ... | ... | ...    | ... | ... |
| ... | ... | $x'_1$ | ... | ... |
| ... | ... | ...    | ... | ... |
| ... | ... | ...    | ... | ... |

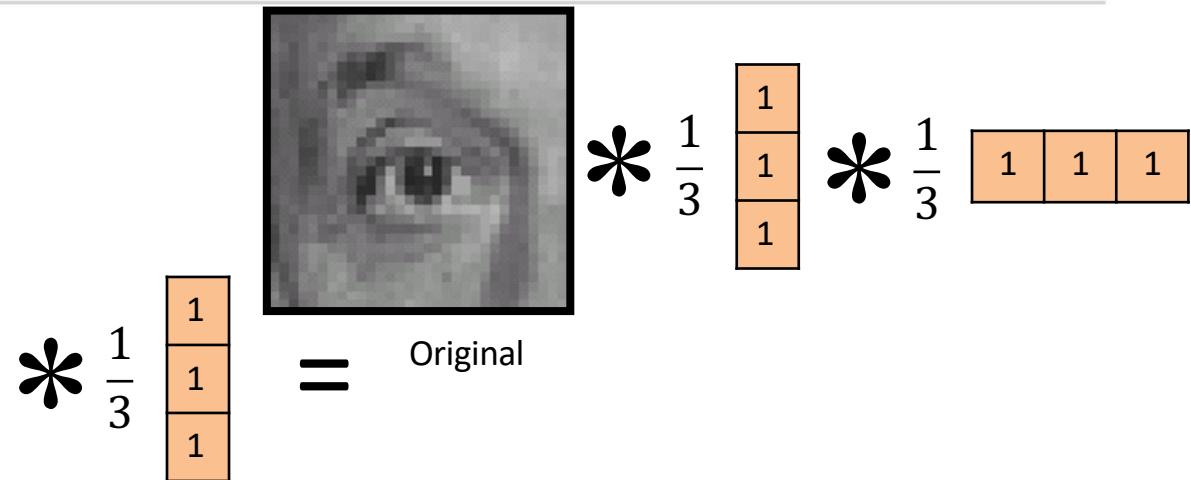
$$x'_1 = \frac{x_1 + x_2 + x_5 + x_6 + x_7 + x_{10} + x_{16} + x_{17} + x_{20}}{9}$$

# Linear filters: Question#1

- Solution (Approach 1):

Input image

|          |          |          |          |          |
|----------|----------|----------|----------|----------|
| $x_{24}$ | $x_{25}$ | $x_{21}$ | $x_{22}$ | $x_{23}$ |
| $x_{19}$ | $x_{20}$ | $x_{16}$ | $x_{17}$ | $x_{18}$ |
| $x_4$    | $x_5$    | $x_1$    | $x_2$    | $x_3$    |
| $x_9$    | $x_{10}$ | $x_6$    | $x_7$    | $x_8$    |
| $x_{14}$ | $x_{15}$ | $x_{11}$ | $x_{12}$ | $x_{13}$ |



|     |                                   |                                |                                |     |
|-----|-----------------------------------|--------------------------------|--------------------------------|-----|
| ... | ...                               | ...                            | ...                            | ... |
| ... | ...                               | ...                            | ...                            | ... |
| ... | $\frac{x_5 + x_{10} + x_{20}}{3}$ | $\frac{x_1 + x_{16} + x_6}{3}$ | $\frac{x_2 + x_7 + x_{17}}{3}$ | ... |
| ... | ...                               | ...                            | ...                            | ... |
| ... | ...                               | ...                            | ...                            | ... |

Output image

# Linear filters: Question#1

- Solution (Approach 1)::

|     |                                   |                                |                                |     |
|-----|-----------------------------------|--------------------------------|--------------------------------|-----|
| ... | ...                               | ...                            | ...                            | ... |
| ... | ...                               | ...                            | ...                            | ... |
| ... | $\frac{x_5 + x_{10} + x_{20}}{3}$ | $\frac{x_1 + x_{16} + x_6}{3}$ | $\frac{x_2 + x_7 + x_{17}}{3}$ | ... |
| ... | ...                               | ...                            | ...                            | ... |
| ... | ...                               | ...                            | ...                            | ... |

$\ast \frac{1}{3} \begin{matrix} 1 & 1 & 1 \end{matrix}$

$$x'_1 = \frac{\frac{x_1 + x_{16} + x_6}{3} + \frac{x_2 + x_7 + x_{17}}{3} + \frac{x_5 + x_{10} + x_{20}}{3}}{3}$$

# Linear filters: Question#1

- Solution (Approach 1)::

|     |                                   |                                |                                |     |
|-----|-----------------------------------|--------------------------------|--------------------------------|-----|
| ... | ...                               | ...                            | ...                            | ... |
| ... | ...                               | ...                            | ...                            | ... |
| ... | $\frac{x_5 + x_{10} + x_{20}}{3}$ | $\frac{x_1 + x_{16} + x_6}{3}$ | $\frac{x_2 + x_7 + x_{17}}{3}$ | ... |
| ... | ...                               | ...                            | ...                            | ... |
| ... | ...                               | ...                            | ...                            | ... |

$\ast \frac{1}{3} \begin{matrix} 1 & 1 & 1 \end{matrix}$

$$x'_1 = \frac{\underline{x_1 + x_2 + x_5 + x_6 + x_7 + x_{10} + x_{16} + x_{17} + x_{20}}}{\frac{3}{3}}$$

# Linear filters: Question#1

- Solution (Approach 1)::

|     |                                   |                                |                                |     |
|-----|-----------------------------------|--------------------------------|--------------------------------|-----|
| ... | ...                               | ...                            | ...                            | ... |
| ... | ...                               | ...                            | ...                            | ... |
| ... | $\frac{x_5 + x_{10} + x_{20}}{3}$ | $\frac{x_1 + x_{16} + x_6}{3}$ | $\frac{x_2 + x_7 + x_{17}}{3}$ | ... |
| ... | ...                               | ...                            | ...                            | ... |
| ... | ...                               | ...                            | ...                            | ... |

$\ast \frac{1}{3} \begin{matrix} 1 & 1 & 1 \end{matrix}$

$$x'_1 = \frac{x_1 + x_2 + x_5 + x_6 + x_7 + x_{10} + x_{16} + x_{17} + x_{20}}{9}$$

# Linear filters: Question#1 - Solution

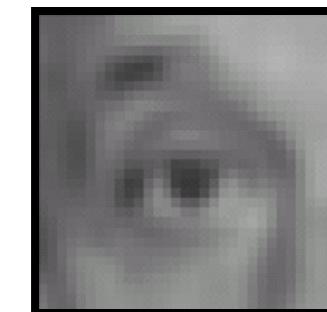
- **Question:** Consider an example image and show the below is correct
- **Approach 1** produces the same image as **Approach 2**



Original

$$\text{Original} * \frac{1}{3} \begin{array}{c} 1 \\ 1 \\ 1 \\ 1 \end{array} * \frac{1}{3} \begin{array}{c} 1 & 1 & 1 \end{array} = \text{Blur (with a mean filter)}$$

**Approach 1**



$$* \frac{1}{9} \begin{array}{ccc} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{array}$$

**Approach 2**

# Linear Filters (Weighted Averaging Filter)

- **Linear Filter (Mean Filter)**

- **Averaging filter:** It is used in reduction of the detail in image. All coefficients are equal.
  - By changing filter size, we control the reduction of the detail in image
- **Weighted averaging filter:** In this, pixels are multiplied by different coefficients. Centre pixel is multiplied by a higher value than average filter. Thus, giving more importance to some pixels.
  - By changing filter size and filter weights, we control the reduction of the detail in image, so more flexible than averaging filter

|   |   |   |   |   |
|---|---|---|---|---|
| 3 | 3 | 3 | 1 | 0 |
| 2 | 1 | 2 | 2 | 0 |
| 2 | 0 | 2 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 2 | 1 |

Filter (3,3)

$$\frac{1}{16} \times$$

|   |   |   |
|---|---|---|
| 1 | 2 | 1 |
| 2 | 4 | 2 |
| 1 | 2 | 1 |

# Linear Filters (Weighted Averaging Filter)

---

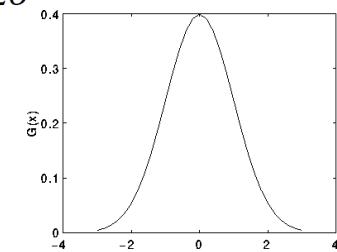
- **Linear Filter (Mean Filter)**

- **Averaging filter:** It is used in reduction of the detail in image. All coefficients are equal.
  - By changing filter size, we control the reduction of the detail in image
- **Weighted averaging filter:** In this, pixels are multiplied by different coefficients. Centre pixel is multiplied by a higher value than average filter. Thus, giving more importance to some pixels.
  - By changing filter size and filter weights, we control the reduction of the detail in image, so more flexible than averaging filter
  - **Gaussian filter** is a type of weighted averaging filter

# Linear Filters (Gaussian filter)

- **Gaussian Filter**

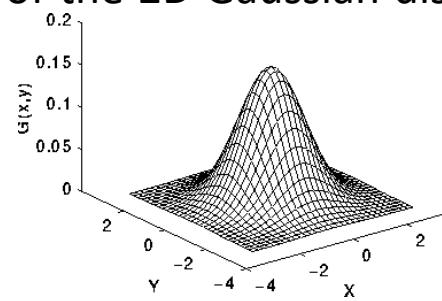
- In one dimension, the Gaussian function is:  $G(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}}$
- The familiar bell shaped Gaussian distribution.



- When working with images we need to use the two dimensional Gaussian function.
- This is simply the product of two 1D Gaussian functions (one for each direction) and is given by:

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

- A graphical representation of the 2D Gaussian distribution with mean (0,0) and  $\sigma = 1$  is shown below.

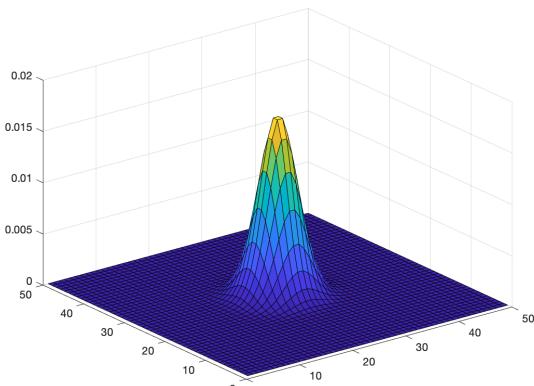


# Linear Filters (Gaussian filter)

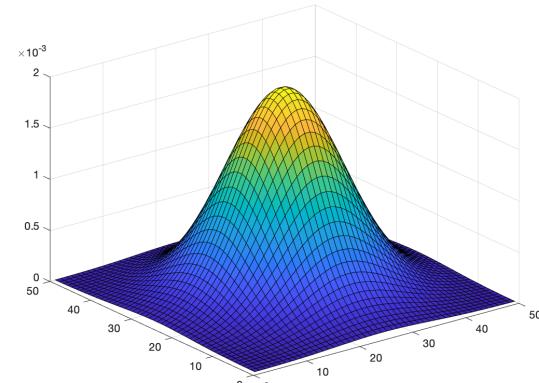
- **Gaussian Filter**

- Graphical representations of the 2D Gaussian distribution with mean (0,0) and different  $\sigma$  are shown below.

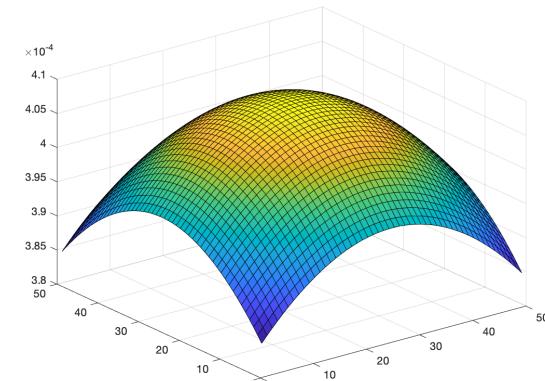
$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$



$\sigma=3$



$\sigma=9$



$\sigma=100$

# Linear Filters (Gaussian filter)

- **Gaussian Filter**

- We need to produce a discrete approximation to the Gaussian function.
- The kernel coefficients diminish with increasing distance from the kernel's centre.
- Gaussian kernel coefficients depend on the value of  $\sigma$

Filter (3,3)

$$\frac{1}{16} \times \begin{array}{|c|c|c|} \hline 1 & 2 & 1 \\ \hline 2 & 4 & 2 \\ \hline 1 & 2 & 1 \\ \hline \end{array}$$

Filter (5,5)

$$\frac{1}{273} \times \begin{array}{|c|c|c|c|c|} \hline 1 & 4 & 7 & 4 & 1 \\ \hline 4 & 16 & 26 & 16 & 4 \\ \hline 7 & 26 & 41 & 26 & 7 \\ \hline 4 & 16 & 26 & 16 & 4 \\ \hline 1 & 4 & 7 & 4 & 1 \\ \hline \end{array}$$

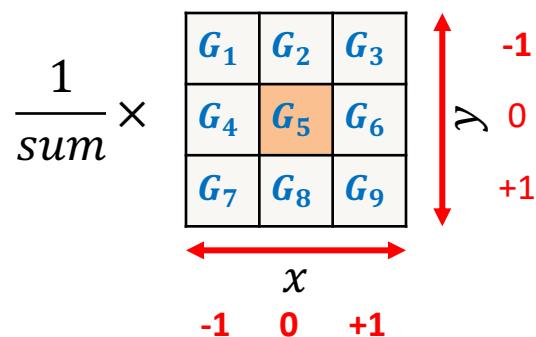
# Linear Filters (Gaussian filter)

- **Gaussian Filter**

- How to produce a discrete approximation to the Gaussian function? For example, create a 3\*3 Gaussian filter.

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

Filter (3,3)



$$G_5 = G(0,0) = \frac{1}{2\pi\sigma^2} e^{\frac{0^2+0^2}{2\sigma^2}}$$

$$sum = G_1 + G_2 + \dots + G_9$$

# Summary

---

- Kernel/Filter
- Smoothing vs. Sharpening
- Smoothing Spatial Filters
  - Linear Filters (Mean filters) also called Lowpass filters
    - Averaging filter
    - Weighted averaging filter (e.g., Gaussian filter)

# SCC.366: Media Coding & Processing

2022-2023

Week 04 – Lecture 2

## Image Transformation - Spatial Filtering Cont.

Dr. Hossein Rahmani

Senior Lecturer in Data Science

Email and Team: [h.rahmani@lancaster.ac.uk](mailto:h.rahmani@lancaster.ac.uk)

# Outline

---

- Kernel/Filter
- Smoothing vs. Sharpening
- Smoothing Spatial Filters
  - Linear Filters (Mean filters)
    - Averaging filter
    - Weighted averaging filter (e.g., Gaussian filter)
  - Order Statistics (Non-linear) filters
    - Minimum filter
    - Maximum filter
    - Median filter
- Sharpening Spatial Filters
  - Unsharp masking
  - First order derivative
  - Second order derivative (Laplacian)

# Review - Linear Filters

---

- **Linear Filter (Mean Filter)**

- **Averaging filter:** It is used in reduction of the detail in image. All coefficients are equal.
  - By changing filter size, we control the reduction of the detail in image
- **Weighted averaging filter:** In this, pixels are multiplied by different coefficients. Centre pixel is multiplied by a higher value than average filter. Thus, giving more importance to some pixels.
  - By changing filter size and filter weights, we control the reduction of the detail in image, so more flexible than averaging filter
  - **Gaussian filter** is a type of weighted averaging filter

# Review - Averaging Filter

- **Linear Filters (Mean Filter)**

- **Averaging filter:** It is used in reduction of the detail in image.

|   |   |   |   |   |
|---|---|---|---|---|
| 3 | 3 | 3 | 1 | 0 |
| 2 | 1 | 2 | 2 | 0 |
| 2 | 0 | 2 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 2 | 1 |

Filter (3,3)

$$\frac{1}{9} \times \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$

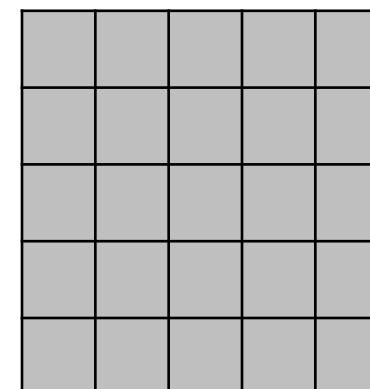


Image:  $F$

Filter/Kernel:  $H$

Output Image:  $G$

- Why we need  $\frac{1}{9} \times$  Filter?
  - To normalize the output values, e.g., if input values are pixel intensities between 0 and 255, then outputs also need to be between 0 and 255

# Review - Averaging Filter

- **Linear Filters (Mean Filter)**

- **Averaging filter:** It is used in reduction of the detail in image.

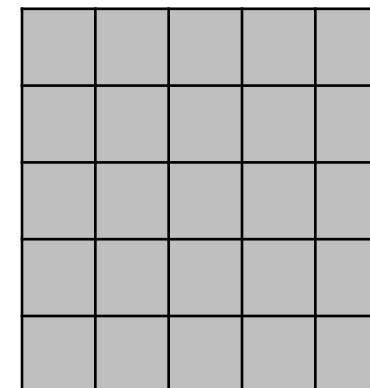
|   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 3 | 3 | 3 | 1 | 0 | 0 |
| 0 | 2 | 1 | 2 | 2 | 0 | 0 |
| 0 | 2 | 0 | 2 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 2 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Image:  $F$

Filter (3,3)

$$\frac{1}{9} \times \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$

Filter/Kernel:  $H$



Output Image:  $G$

# Review - Averaging Filter

- **Linear Filters (Mean Filter)**

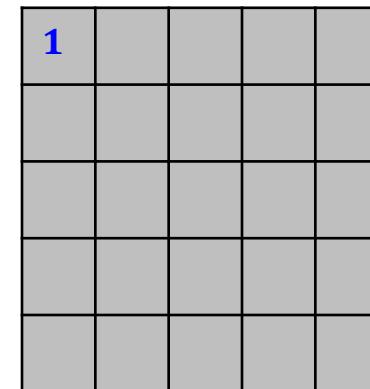
- **Averaging filter:** It is used in reduction of the detail in image.

|   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 3 | 1 | 3 | 1 | 0 | 0 | 0 |
| 0 | 1 | 2 | 1 | 1 | 2 | 2 | 0 | 0 |
| 0 | 2 | 0 | 2 | 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 | 2 | 1 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Image:  $F$

Filter (3,3)

$$\frac{1}{9} \times \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$



Output Image:  $G$

$$\frac{1}{9} \times \begin{array}{|c|c|c|} \hline 0 \times 1 & 0 \times 1 & 0 \times 1 \\ \hline 0 \times 1 & 3 \times 1 & 3 \times 1 \\ \hline 0 \times 1 & 2 \times 1 & 1 \times 1 \\ \hline \end{array}$$

→  $\frac{1}{9} \times (0 + 0 + 0 + 0 + 3 + 3 + 0 + 2 + 1) = 1$

# Review - Averaging Filter

- **Linear Filters (Mean Filter)**

- **Averaging filter:** It is used in reduction of the detail in image.

|   |                |                |                |   |   |   |
|---|----------------|----------------|----------------|---|---|---|
| 0 | 0 <sup>1</sup> | 0 <sup>1</sup> | 0 <sup>1</sup> | 0 | 0 | 0 |
| 0 | 3 <sup>1</sup> | 3 <sup>1</sup> | 3 <sup>1</sup> | 1 | 0 | 0 |
| 0 | 2 <sup>1</sup> | 1 <sup>1</sup> | 2 <sup>1</sup> | 2 | 0 | 0 |
| 0 | 2              | 0              | 2              | 0 | 1 | 0 |
| 0 | 1              | 0              | 0              | 0 | 1 | 0 |
| 0 | 1              | 1              | 0              | 2 | 1 | 0 |
| 0 | 0              | 0              | 0              | 0 | 0 | 0 |

Image:  $F$

Filter (3,3)

$$\frac{1}{9} \times \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$

|   |     |  |  |  |
|---|-----|--|--|--|
| 1 | 1.6 |  |  |  |
|   |     |  |  |  |
|   |     |  |  |  |
|   |     |  |  |  |
|   |     |  |  |  |

Output Image:  $G$

$$\frac{1}{9} \times \begin{array}{|c|c|c|} \hline 0 \times 1 & 0 \times 1 & 0 \times 1 \\ \hline 3 \times 1 & 3 \times 1 & 3 \times 1 \\ \hline 2 \times 1 & 1 \times 1 & 2 \times 1 \\ \hline \end{array}$$

→  $\frac{1}{9} \times (0 + 0 + 0 + 3 + 3 + 3 + 2 + 1 + 2) = \frac{14}{9} = 1.6$

# Review - Averaging Filter

- **Linear Filters (Mean Filter)**

- **Averaging filter:** It is used in reduction of the detail in image.

|   |   |                |                |                |   |   |
|---|---|----------------|----------------|----------------|---|---|
| 0 | 0 | 0 <sup>1</sup> | 0 <sup>1</sup> | 0 <sup>1</sup> | 0 | 0 |
| 0 | 3 | 3 <sup>1</sup> | 3 <sup>1</sup> | 1 <sup>1</sup> | 0 | 0 |
| 0 | 2 | 1 <sup>1</sup> | 2 <sup>1</sup> | 2 <sup>1</sup> | 0 | 0 |
| 0 | 2 | 0              | 2              | 0              | 1 | 0 |
| 0 | 1 | 0              | 0              | 0              | 1 | 0 |
| 0 | 1 | 1              | 0              | 2              | 1 | 0 |
| 0 | 0 | 0              | 0              | 0              | 0 | 0 |

Image:  $F$

Filter (3,3)

$$\frac{1}{9} \times \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$

|   |     |     |  |  |
|---|-----|-----|--|--|
| 1 | 1.6 | 1.3 |  |  |
|   |     |     |  |  |
|   |     |     |  |  |
|   |     |     |  |  |
|   |     |     |  |  |

Output Image:  $G$

$$\frac{1}{9} \times \begin{array}{|c|c|c|} \hline 0 \times 1 & 0 \times 1 & 0 \times 1 \\ \hline 3 \times 1 & 3 \times 1 & 1 \times 1 \\ \hline 1 \times 1 & 2 \times 1 & 2 \times 1 \\ \hline \end{array}$$

→  $\frac{1}{9} \times (0 + 0 + 0 + 3 + 3 + 1 + 1 + 2 + 2) = \frac{12}{9} = 1.3$

# Review - Averaging Filter

- **Linear Filters (Mean Filter)**

- **Averaging filter:** It is used in reduction of the detail in image.

|   |   |   |                |                |                |   |
|---|---|---|----------------|----------------|----------------|---|
| 0 | 0 | 0 | 0 <sup>1</sup> | 0 <sup>1</sup> | 0 <sup>1</sup> | 0 |
| 0 | 3 | 3 | 3 <sup>1</sup> | 1 <sup>1</sup> | 0 <sup>1</sup> | 0 |
| 0 | 2 | 1 | 2 <sup>1</sup> | 2 <sup>1</sup> | 0 <sup>1</sup> | 0 |
| 0 | 2 | 0 | 2              | 0              | 1              | 0 |
| 0 | 1 | 0 | 0              | 0              | 1              | 0 |
| 0 | 1 | 1 | 0              | 2              | 1              | 0 |
| 0 | 0 | 0 | 0              | 0              | 0              | 0 |

Image:  $F$

Filter (3,3)

$$\frac{1}{9} \times \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$

|   |     |     |     |  |
|---|-----|-----|-----|--|
| 1 | 1.6 | 1.3 | 0.9 |  |
|   |     |     |     |  |
|   |     |     |     |  |
|   |     |     |     |  |
|   |     |     |     |  |

Output Image:  $G$

|     |     |     |
|-----|-----|-----|
| 0×1 | 0×1 | 0×1 |
| 3×1 | 1×1 | 0×1 |
| 2×1 | 2×1 | 0×1 |

$$\rightarrow \frac{1}{9} \times (0 + 0 + 0 + 3 + 1 + 0 + 2 + 2 + 0) = \frac{8}{9} = 0.9$$

# Review - Averaging Filter

- **Linear Filters (Mean Filter)**

- **Averaging filter:** It is used in reduction of the detail in image.

|   |   |   |   |                |                |                |
|---|---|---|---|----------------|----------------|----------------|
| 0 | 0 | 0 | 0 | 0 <sup>1</sup> | 0 <sup>1</sup> | 0 <sup>1</sup> |
| 0 | 3 | 3 | 3 | 1 <sup>1</sup> | 0 <sup>1</sup> | 0 <sup>1</sup> |
| 0 | 2 | 1 | 2 | 2 <sup>1</sup> | 0 <sup>1</sup> | 0 <sup>1</sup> |
| 0 | 2 | 0 | 2 | 0              | 1              | 0              |
| 0 | 1 | 0 | 0 | 0              | 1              | 0              |
| 0 | 1 | 1 | 0 | 2              | 1              | 0              |
| 0 | 0 | 0 | 0 | 0              | 0              | 0              |

Image:  $F$

Filter (3,3)

$$\frac{1}{9} \times \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$

|   |     |     |     |     |
|---|-----|-----|-----|-----|
| 1 | 1.6 | 1.3 | 0.9 | 0.3 |
|   |     |     |     |     |
|   |     |     |     |     |
|   |     |     |     |     |
|   |     |     |     |     |

Output Image:  $G$

|     |     |     |
|-----|-----|-----|
| 0×1 | 0×1 | 0×1 |
| 1×1 | 0×1 | 0×1 |
| 2×1 | 0×1 | 0×1 |

Filter/Kernel:  $H$

$$\rightarrow \frac{1}{9} \times (0 + 0 + 0 + 1 + 0 + 0 + 2 + 0 + 0) = \frac{3}{9} = 0.3$$

# Review - Averaging Filter

- **Linear Filters (Mean Filter)**

- **Averaging filter:** It is used in reduction of the detail in image.

|                |                |                |   |   |   |   |
|----------------|----------------|----------------|---|---|---|---|
| 0              | 0              | 0              | 0 | 0 | 0 | 0 |
| 0 <sup>1</sup> | 3 <sup>1</sup> | 3 <sup>1</sup> | 3 | 1 | 0 | 0 |
| 0 <sup>1</sup> | 2 <sup>1</sup> | 1 <sup>1</sup> | 2 | 2 | 0 | 0 |
| 0 <sup>1</sup> | 2 <sup>1</sup> | 0 <sup>1</sup> | 2 | 0 | 1 | 0 |
| 0              | 1              | 0              | 0 | 0 | 1 | 0 |
| 0              | 1              | 1              | 0 | 2 | 1 | 0 |
| 0              | 0              | 0              | 0 | 0 | 0 | 0 |

Image:  $F$

Filter (3,3)

$$\frac{1}{9} \times \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$

|     |     |     |     |     |
|-----|-----|-----|-----|-----|
| 1   | 1.6 | 1.3 | 0.9 | 0.3 |
| 1.2 |     |     |     |     |
|     |     |     |     |     |
|     |     |     |     |     |
|     |     |     |     |     |

Output Image:  $G$

|     |     |     |
|-----|-----|-----|
| 0×1 | 3×1 | 3×1 |
| 0×1 | 2×1 | 1×1 |
| 0×1 | 2×1 | 0×1 |



$$\frac{1}{9} \times (0 + 3 + 3 + 0 + 2 + 1 + 0 + 2 + 0) = \frac{11}{9} = 1.2$$

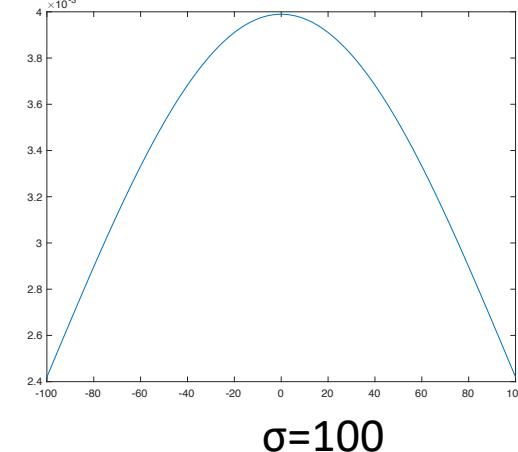
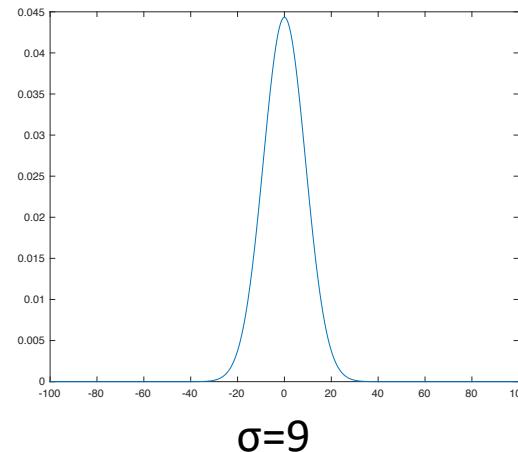
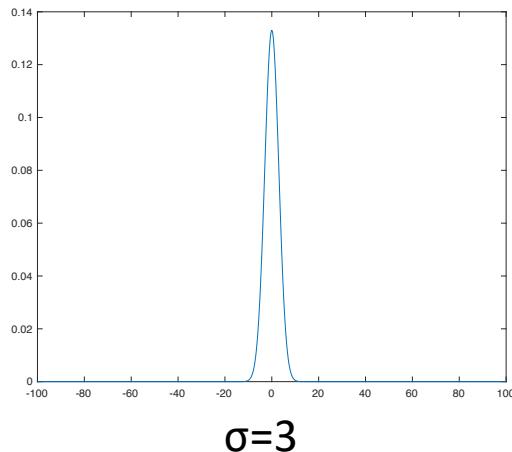
# Review - Gaussian filter

- **Gaussian Filter (1D)**



- Gaussian filter is a type of weighted averaging filter.
- Graphical representations of the 1D Gaussian distribution with mean 0 and different  $\sigma$  are shown below.

$$G(x, y) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}}$$

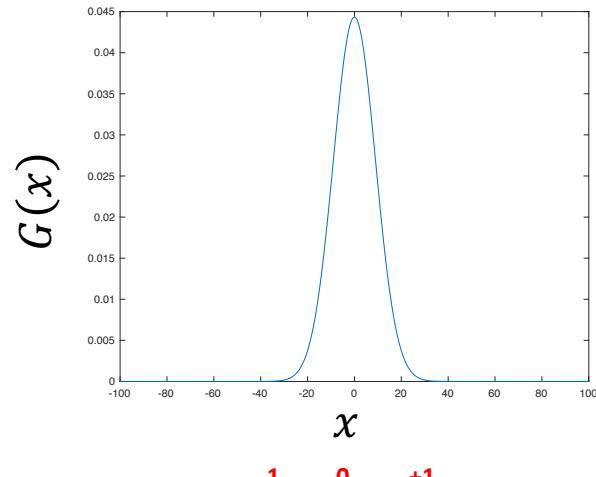


# Review - Gaussian filter

- **Gaussian Filter (1D)**

- Gaussian filter is a type of weighted averaging filter.
- Graphical representations of the 1D Gaussian distribution with mean 0 and different  $\sigma$  are shown below.

$$G(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}}$$



$$G(0) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{0}{2\sigma^2}}$$

$$G(-1) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{(-1)^2}{2\sigma^2}}$$

$$G(+1) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{(1)^2}{2\sigma^2}}$$

|        |        |        |
|--------|--------|--------|
| 0.0965 | 0.1592 | 0.0965 |
|--------|--------|--------|

# Review - Gaussian filter

---

- **Gaussian Filter (1D)**

- Gaussian filter is a type of weighted averaging filter.
- Graphical representations of the 1D Gaussian distribution with mean 0 and different  $\sigma$  are shown below.
- We need  $\frac{1}{sum}$  to normalise values

$$\frac{1}{sum} \times \begin{array}{|c|c|c|} \hline 0.0965 & 0.1592 & 0.0965 \\ \hline \end{array}$$

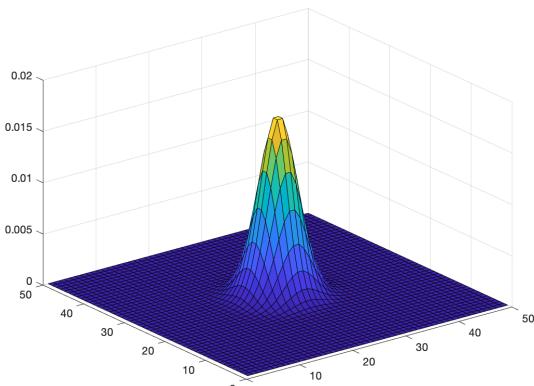
$$sum = 0.0965 + 0.1592 + 0.0965$$

# Review - Gaussian filter

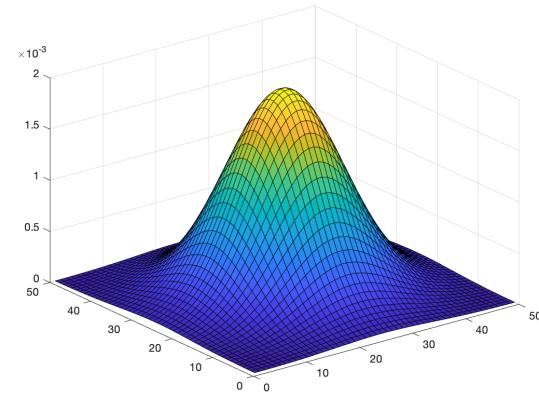
- **Gaussian Filter (2D)**

- Gaussian filter is a type of weighted averaging filter.
- Graphical representations of the 2D Gaussian distribution with mean (0,0) and different  $\sigma$  are shown below.

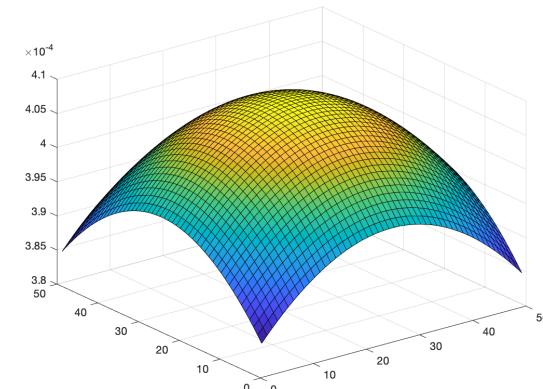
$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$



$\sigma=3$



$\sigma=9$



$\sigma=100$

# Review - Gaussian filter

---

- **Gaussian Filter (2D)**

- Gaussian filter is a type of weighted averaging filter.
- We need to produce a discrete approximation to the Gaussian function.
- The kernel coefficients diminish with increasing distance from the kernel's centre.
- Gaussian kernel coefficients depend on the value of  $\sigma$

Filter (3,3)

$$\frac{1}{16} \times \begin{array}{|c|c|c|} \hline 1 & 2 & 1 \\ \hline 2 & 4 & 2 \\ \hline 1 & 2 & 1 \\ \hline \end{array}$$

Filter (5,5)

$$\frac{1}{273} \times \begin{array}{|c|c|c|c|c|} \hline 1 & 4 & 7 & 4 & 1 \\ \hline 4 & 16 & 26 & 16 & 4 \\ \hline 7 & 26 & 41 & 26 & 7 \\ \hline 4 & 16 & 26 & 16 & 4 \\ \hline 1 & 4 & 7 & 4 & 1 \\ \hline \end{array}$$

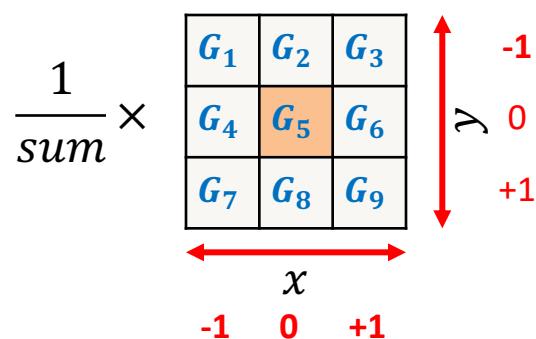
# Review - Gaussian filter

- **Gaussian Filter (2D)**

- How to produce a discrete approximation to the Gaussian function? For example, create a 3\*3 Gaussian filter.

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

Filter (3,3)



$$G_5 = G(0,0) = \frac{1}{2\pi\sigma^2} e^{-\frac{0^2+0^2}{2\sigma^2}}$$

$$sum = G_1 + G_2 + \dots + G_9$$

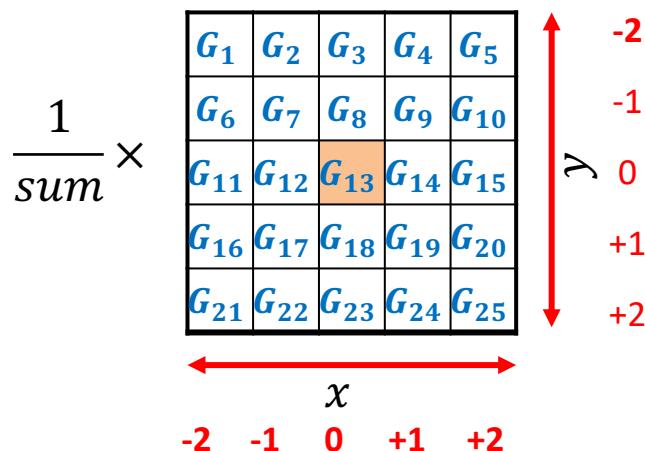
# Review - Gaussian filter

- **Gaussian Filter (2D)**

- How to produce a discrete approximation to the Gaussian function? For example, create a 3\*3 Gaussian filter.

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

Filter (5,5)

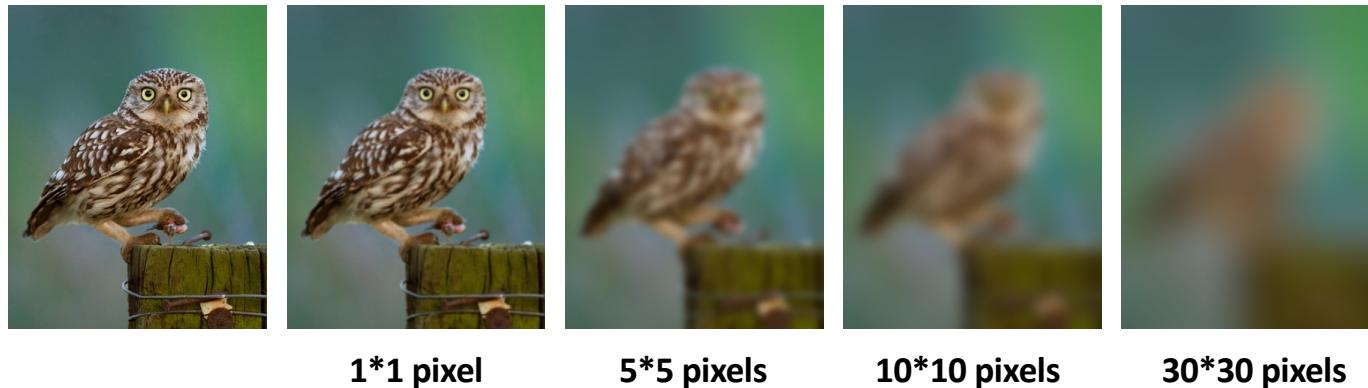


$$G_{13} = G(0,0) = \frac{1}{2\pi\sigma^2} e^{-\frac{0^2+0^2}{2\sigma^2}}$$

$$sum = G_1 + G_2 + \dots + G_{25}$$

# Linear Filters (Gaussian filter)

- Impact of different kernel sizes



# Gaussian vs. Averaging filters

- Both remove high-frequency components from the image (low-pass filter)



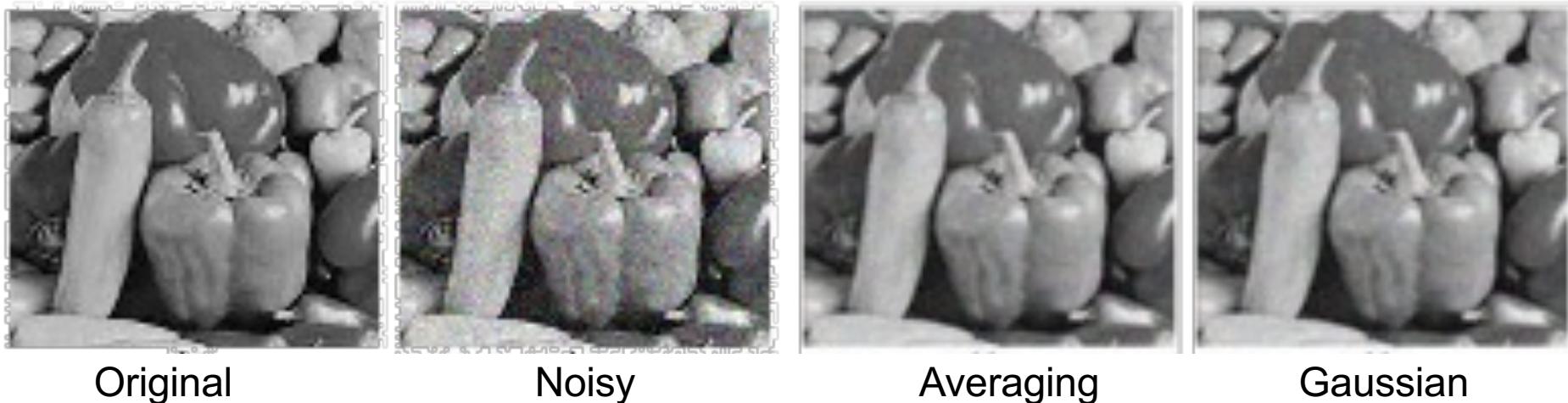
Averaging



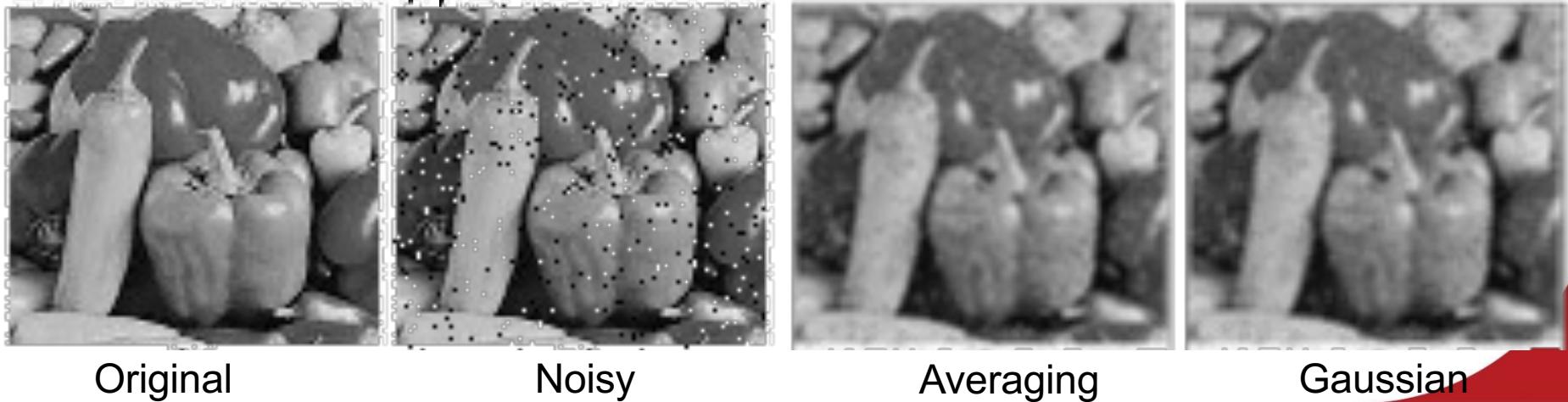
Gaussian

# Noise removal using mean filters

- Gaussian noise

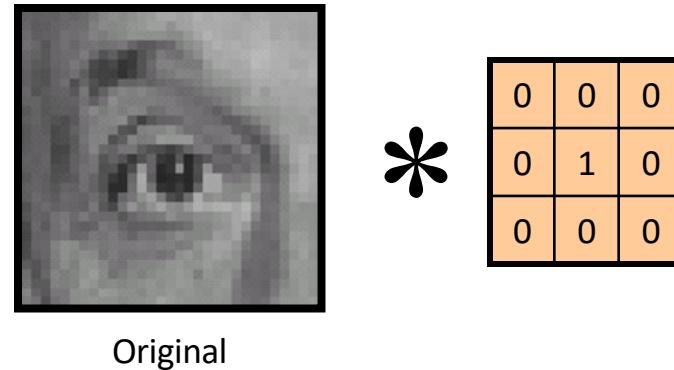


- Salt and Pepper noise

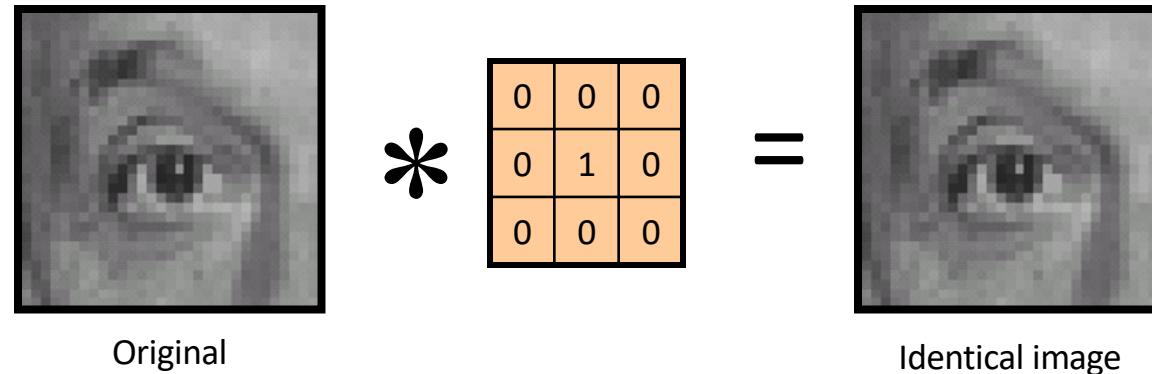


## Linear filters: Question#2

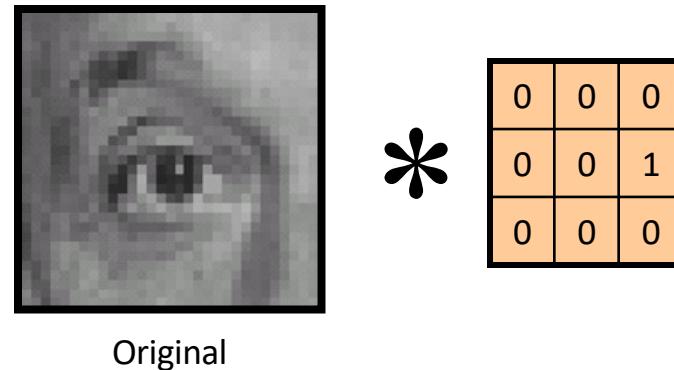
---



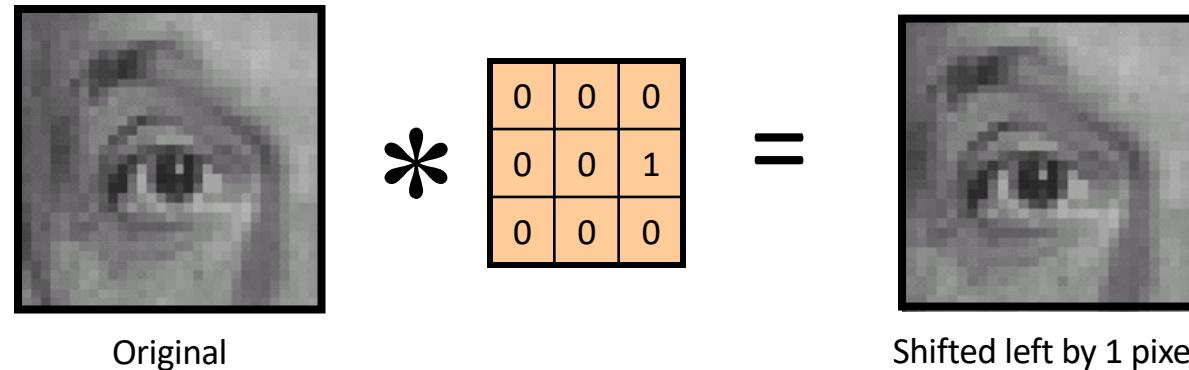
- **Question#2:**
  - What image operation does filtering with this kernel perform?



## Linear filters: Question#3



- **Question#3:**
  - What image operation does filtering with this kernel perform?



# Linear filters: Question#4 (Homework)

---

- **Question#4:**

- Apply the given Gaussian filter to the image
- Borders: keep border values as they are

|    |    |    |    |    |    |
|----|----|----|----|----|----|
| 15 | 20 | 25 | 25 | 15 | 10 |
| 20 | 15 | 50 | 30 | 20 | 15 |
| 20 | 50 | 55 | 60 | 30 | 20 |
| 20 | 15 | 65 | 30 | 15 | 30 |
| 15 | 20 | 30 | 20 | 25 | 30 |
| 20 | 25 | 15 | 20 | 10 | 15 |

\*     $\frac{1}{16} \times$ 
Filter (3,3)

|   |   |   |
|---|---|---|
| 1 | 2 | 1 |
| 2 | 4 | 2 |
| 1 | 2 | 1 |

# Linear filters: Question#5 (Homework)

- **Question#5:**

- Apply 1-D Gaussian filters sequentially to the image
  - First apply the below filter while keeping the 1<sup>st</sup> and last columns as they are.

$$\frac{1}{4} \times \begin{array}{|c|c|c|} \hline 1 & 2 & 1 \\ \hline \end{array}$$

- Let's assume the output image after applying the first filter is  $O_1$
- Then apply the second filter as shown below on  $O_1$  while keeping the 1<sup>st</sup> and last rows as they are.

$$\frac{1}{4} \times \begin{array}{|c|c|c|} \hline 1 \\ \hline 2 \\ \hline 1 \\ \hline \end{array}$$

|    |    |    |    |    |    |
|----|----|----|----|----|----|
| 15 | 20 | 25 | 25 | 15 | 10 |
| 20 | 15 | 50 | 30 | 20 | 15 |
| 20 | 50 | 55 | 60 | 30 | 20 |
| 20 | 15 | 65 | 30 | 15 | 30 |
| 15 | 20 | 30 | 20 | 25 | 30 |
| 20 | 25 | 15 | 20 | 10 | 15 |

$$* \frac{1}{4} \times \begin{array}{|c|c|c|} \hline 1 & 2 & 1 \\ \hline \end{array}$$

$$* \frac{1}{4} \times \begin{array}{|c|c|c|} \hline 1 \\ \hline 2 \\ \hline 1 \\ \hline \end{array}$$

# Linear filters: Question#6 (Homework)

---

- **Question#6:**
  - Compare the output images from Question#1 and Question#2
  - Are they the same?

# Outline

---

- Kernel/Filter
- Smoothing vs. Sharpening
- Smoothing Spatial Filters
  - Linear Filters (Mean filters)
    - Averaging filter
    - Weighted averaging filter (e.g., Gaussian filter)
  - **Order Statistics (Non-linear) filters**
    - Minimum filter
    - Maximum filter
    - Median filter
- Sharpening Spatial Filters
  - Unsharp masking
  - First order derivative
  - Second order derivative (Laplacian)

# Non-Linear Filters (Minimum filter)

- **Linear Filter (Mean Filter)**

- **Averaging filter:** It is used in reduction of the detail in image. All coefficients are equal.
- **Weighted averaging filter:** In this, pixels are multiplied by different coefficients. Center pixel is multiplied by a higher value than average filter.

- **Order Statistics (Non-linear) Filter**

- **Minimum filter:** The value of the center is replaced by the smallest value in the window.

|   |   |   |   |   |
|---|---|---|---|---|
| 3 | 3 | 3 | 1 | 0 |
| 2 | 1 | 2 | 2 | 0 |
| 2 | 0 | 2 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 2 | 1 |

Filter (3,3)

Input Image

# Non-Linear Filters (Minimum filter)

- **Linear Filter (Mean Filter)**

- **Averaging filter:** It is used in reduction of the detail in image. All coefficients are equal.
- **Weighted averaging filter:** In this, pixels are multiplied by different coefficients. Center pixel is multiplied by a higher value than average filter.

- **Order Statistics (Non-linear) Filter**

- **Minimum filter:** The value of the center is replaced by the smallest value in the window.

|   |   |   |   |   |
|---|---|---|---|---|
| 3 | 3 | 3 | 1 | 0 |
| 2 | 1 | 2 | 2 | 0 |
| 2 | 0 | 2 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 2 | 1 |

Input Image

Given (3,3) filter, the minimum value is 0. So, the centre pixel is changed to 0 in the output image

# Non-Linear Filters (Maximum filter)

- **Linear Filter (Mean Filter)**

- **Averaging filter:** It is used in reduction of the detail in image. All coefficients are equal.
- **Weighted averaging filter:** In this, pixels are multiplied by different coefficients. Center pixel is multiplied by a higher value than average filter.

- **Order Statistics (Non-linear) filter**

- **Minimum filter:** The value of the center is replaced by the smallest value in the window.
- **Maximum filter:** The value of the center is replaced by the largest value in the window.

|   |   |   |   |   |
|---|---|---|---|---|
| 3 | 3 | 3 | 1 | 0 |
| 2 | 1 | 2 | 2 | 0 |
| 2 | 0 | 2 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 2 | 1 |

Filter (3,3)

Input Image

# Non-Linear Filters (Maximum filter)

- **Linear Filter (Mean Filter)**

- **Averaging filter:** It is used in reduction of the detail in image. All coefficients are equal.
- **Weighted averaging filter:** In this, pixels are multiplied by different coefficients. Center pixel is multiplied by a higher value than average filter.

- **Order Statistics (Non-linear) filter**

- **Minimum filter:** The value of the center is replaced by the smallest value in the window.
- **Maximum filter:** The value of the center is replaced by the largest value in the window.

|   |   |   |   |   |
|---|---|---|---|---|
| 3 | 3 | 3 | 1 | 0 |
| 2 | 1 | 2 | 2 | 0 |
| 2 | 0 | 2 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 2 | 1 |

Input Image

Given (3,3) filter, the maximum value is 3. So, the centre pixel is changed to 3 in the output image.

# Non-Linear Filters (Median filter)

- **Linear Filter (Mean Filter)**

- **Averaging filter:** It is used in reduction of the detail in image. All coefficients are equal.
- **Weighted averaging filter:** In this, pixels are multiplied by different coefficients. Center pixel is multiplied by a higher value than average filter.

- **Order Statistics (Non-linear) filter**

- **Minimum filter:** The value of the center is replaced by the smallest value in the window.
- **Maximum filter:** The value of the center is replaced by the largest value in the window.
- **Median filter:** First neighboring pixels are sorted and original values of the pixel is replaced by the median of the list.

Sorted pixel values in the (3,3) window

|             |   |   |   |   |   |   |   |   |
|-------------|---|---|---|---|---|---|---|---|
| 0           | 1 | 2 | 2 | 2 | 2 | 3 | 3 | 3 |
| ↑<br>Median |   |   |   |   |   |   |   |   |

Given (3,3) filter, the median is 2. So, the centre pixel is changed to 2 in the output image.

# Noise Removal using Median Filter

- Very effective for removing “salt and pepper” noise (i.e., random occurrences of black and white pixels).

Original Image



Image with Noise



Averaging

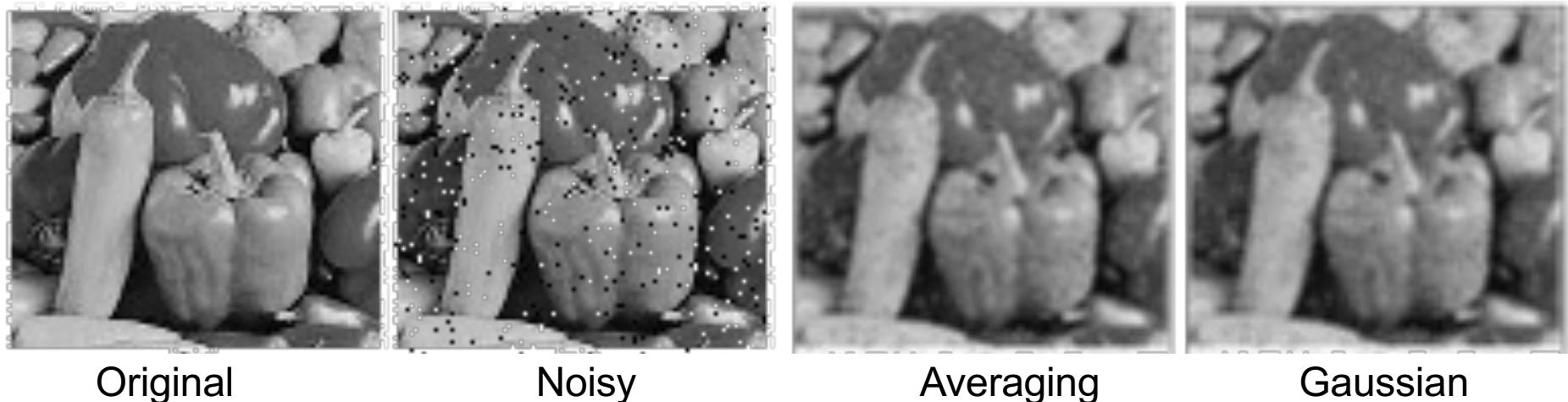


Median Filtering



# Noise removal using Median Filter

- Very effective for removing “salt and pepper” noise



Original

Noisy

Averaging

Gaussian



Median filter

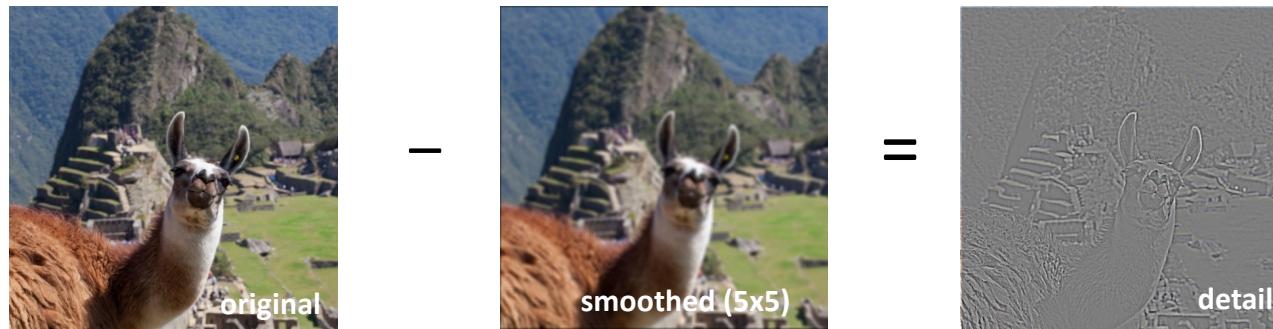
# Outline

---

- Kernel/Filter
- Smoothing vs. Sharpening
- Smoothing Spatial Filters
  - Linear Filters (Mean filters)
    - Averaging filter
    - Weighted averaging filter (e.g., Gaussian filter)
  - Order Statistics (Non-linear) filters
    - Minimum filter
    - Maximum filter
    - Median filter
- **Sharpening Spatial Filters**
  - Unsharp masking
  - First order derivative
  - Second order derivative (Laplacian)

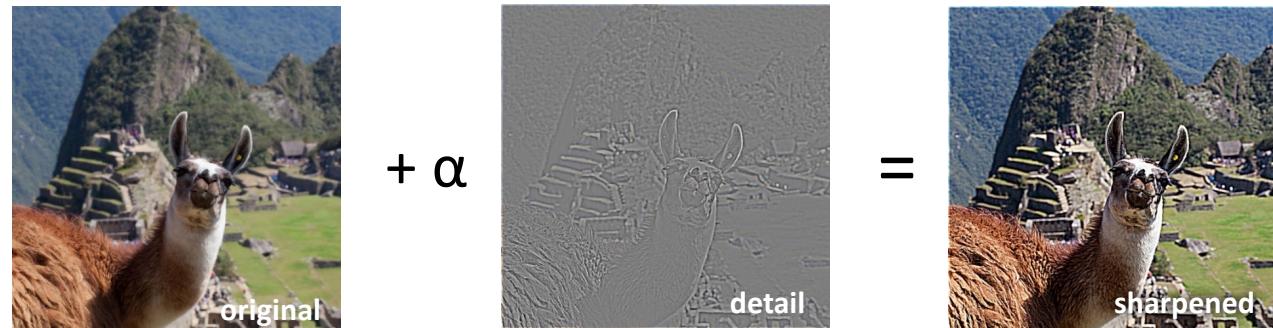
# Linear Filters (Sharpening Filter)

- What does blurring take away?
  - Let's assume  $\alpha=1$  for simplicity



(This “detail extraction” operation is also called a **high-pass filter**)

Let's add it back:



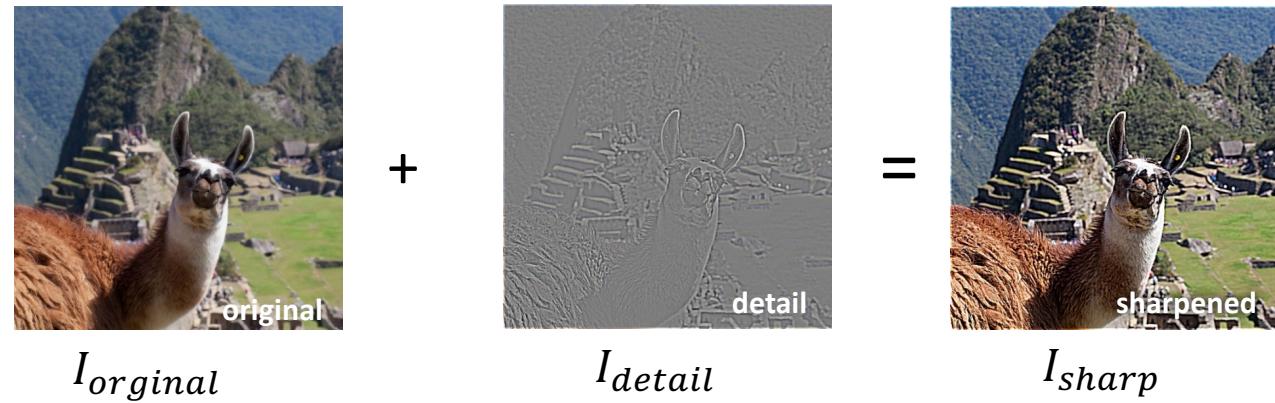
# Linear Filters (Sharpening Filter)

- What does blurring take away?



(This “detail extraction” operation is also called a **high-pass filter**)

Let's add it back:



# Linear Filters (Sharpening Filter)

- What does blurring take away?

$$I_{original} - I_{blur} = I_{detail}$$

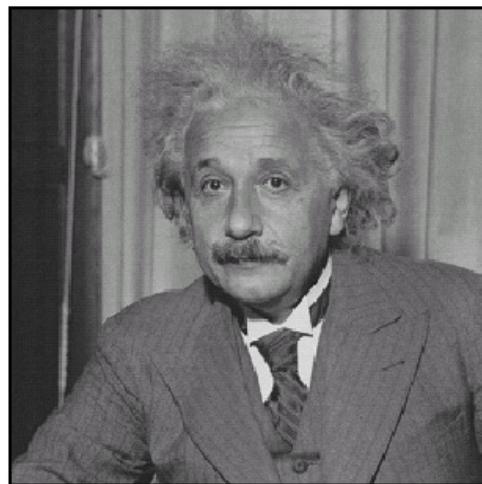
$$I_{original} + I_{detail} = I_{sharp}$$

$$I_{sharp} = I_{original} + (I_{original} - I_{blur})$$

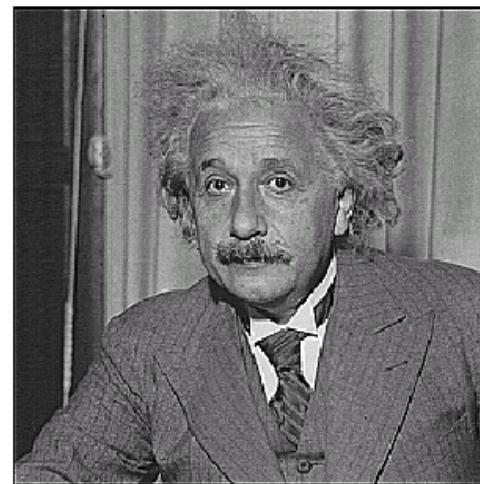
$$I_{sharp} = 2I_{original} - I_{blur}$$

# Linear Filters (Sharpening Filter) - Example

- **Warning:** the results of sharpening might contain negative values (i.e., re-map them to [0, 255])



before



after

# Linear Filters (Sharpening Filter) - Example

- **Warning:** the results of sharpening might contain negative values (i.e., re-map them to [0, 255])

Input Image



Sharpened Image



# Summary

---

- Kernel/Filter
- Smoothing vs. Sharpening
- Smoothing Spatial Filters
  - Linear Filters (Mean filters) also called Lowpass filters
    - Averaging filter
    - Weighted averaging filter (e.g., Gaussian filter)
  - Order Statistics (Non-linear) filters
    - Minimum filter
    - Maximum filter
    - Median filter (can be used for noise removal)
- Sharpening Spatial Filters also called Highpass filters
  - Unsharp masking
  - First order derivative (Edge detection – Next lecture)
  - Second order derivative (Laplacian) (Edge detection – Next lecture)

# SCC.366: Media Coding & Processing

2022-2023

Week 05 – Lecture 1

## Edge Detection

Dr. Hossein Rahmani

Senior Lecturer in Data Science

Email and Team: [h.rahmani@lancaster.ac.uk](mailto:h.rahmani@lancaster.ac.uk)

# Outline

---

- What is an Edge?
- Why do we care about edges?
- Origin of edges
- First order edge detectors
- Edge detection using Laplacian
- Canny edge detector
- Summary of the main points

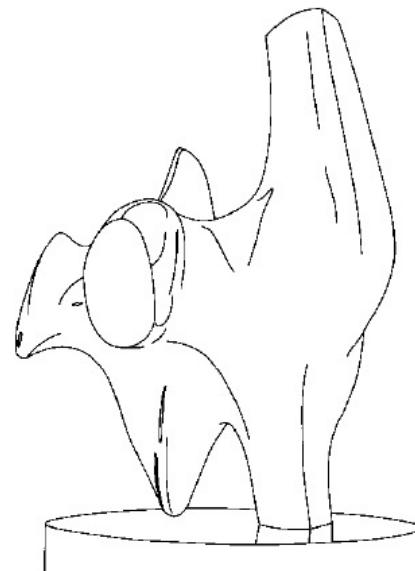
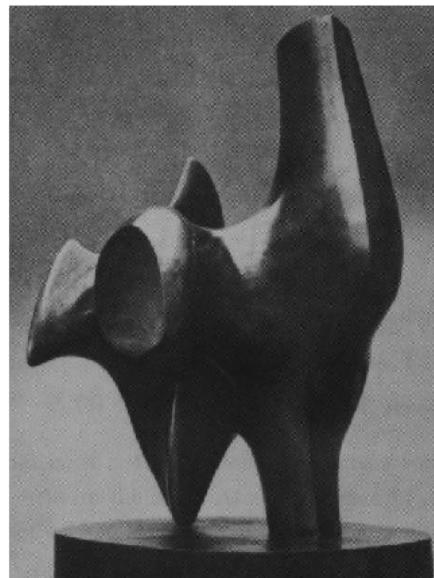
# Edge: Definition

- An edge is a significant local change in the image intensity.
- It is the boundary between two regions with distinct intensity level properties.



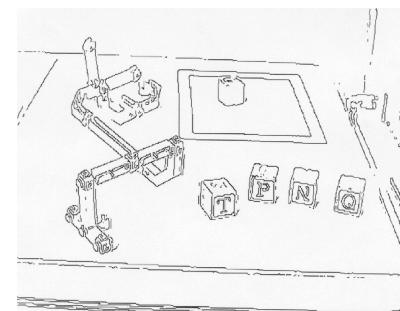
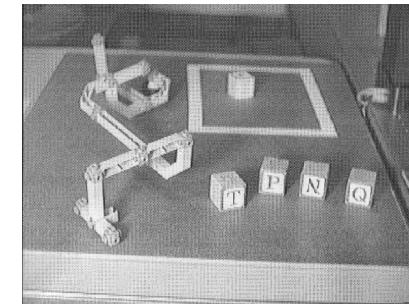
# Edge: Definition

- **Goal of edge detection:** Identify sudden changes (discontinuities) in an image
- Convert a 2D image into a set of points where image intensity changes rapidly.



# Why do we care about edges?

- Edges are more compact than pixels
  - Reduce the amount of data to be processed





# Why do we care about edges?

- Most semantic and shape information from the image can be encoded in the edges
  - Can be used for object detection and recognition





# Why do we care about edges?

- Most semantic and shape information from the image can be encoded in the edges
  - Can be used for object detection and recognition





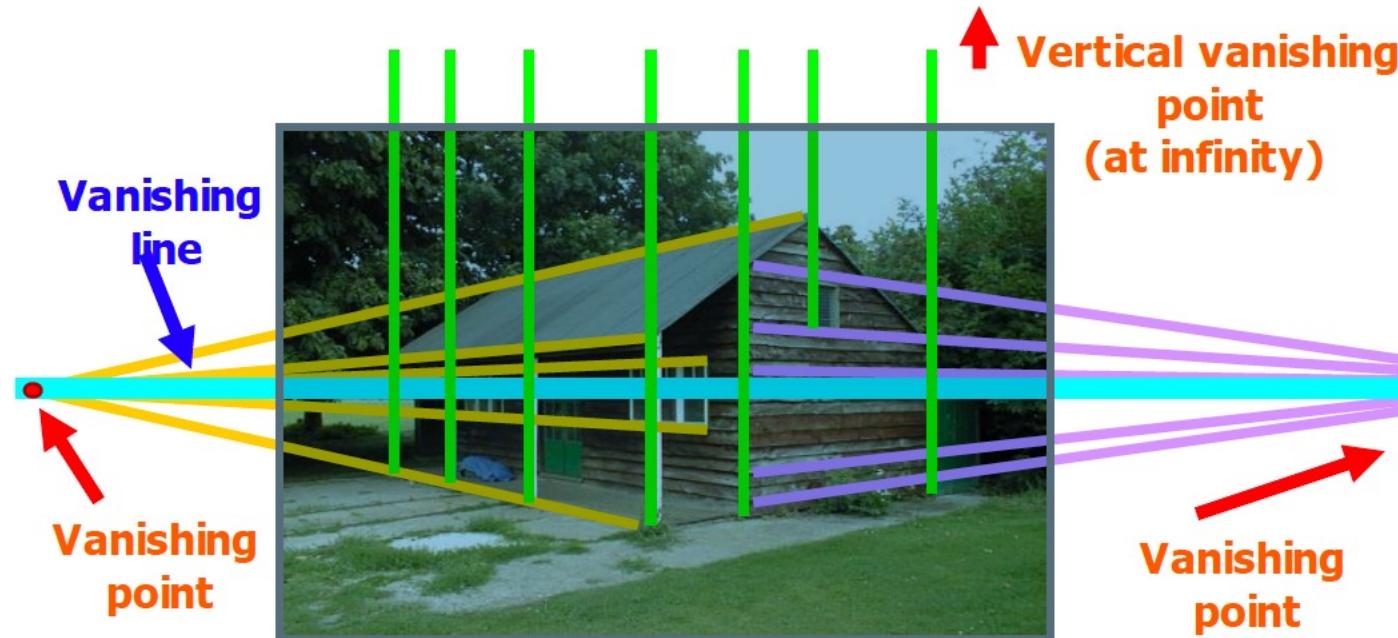
# Why do we care about edges?

- Most semantic and shape information from the image can be encoded in the edges
  - Can be used for object detection and recognition



# Why do we care about edges?

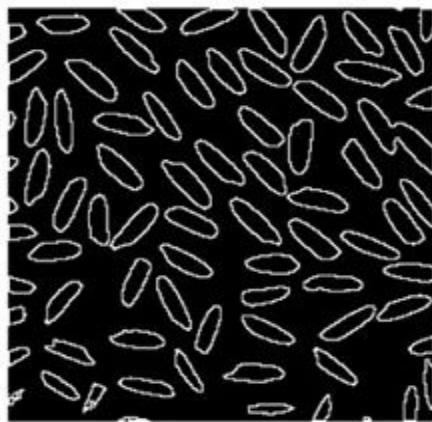
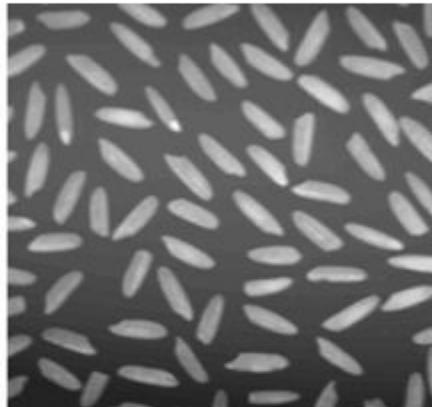
- Recover geometry and viewpoint





# Why do we care about edges?

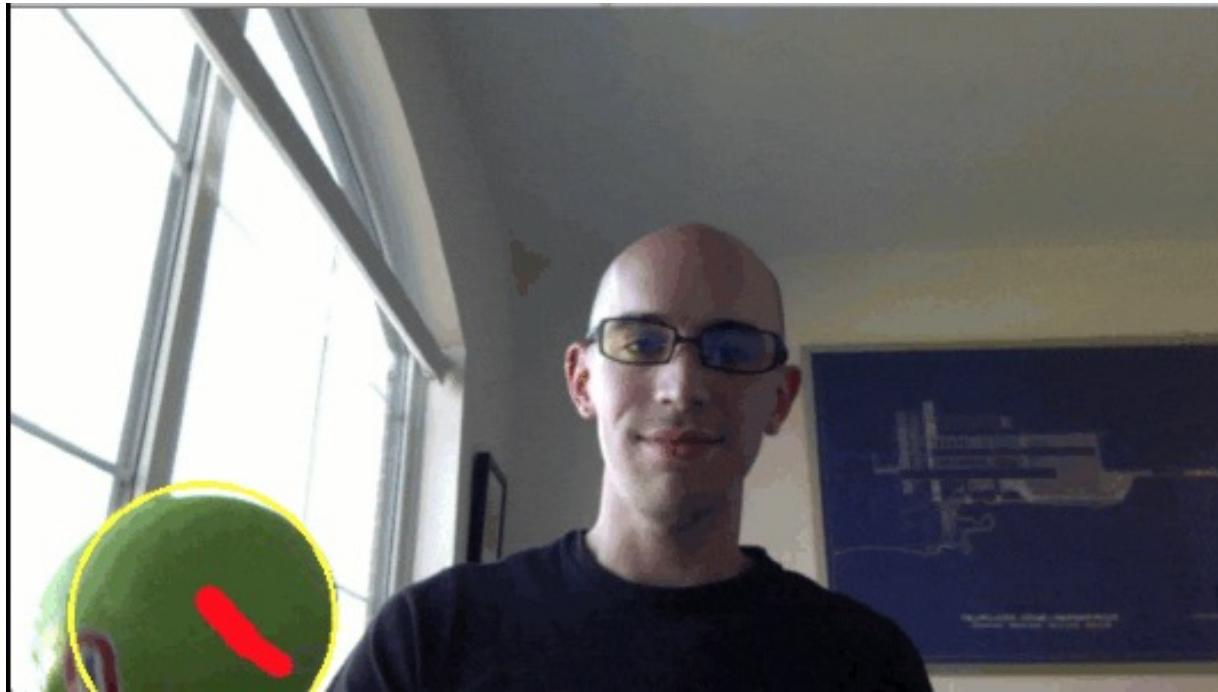
- Image segmentation
  - Separating objects from the background





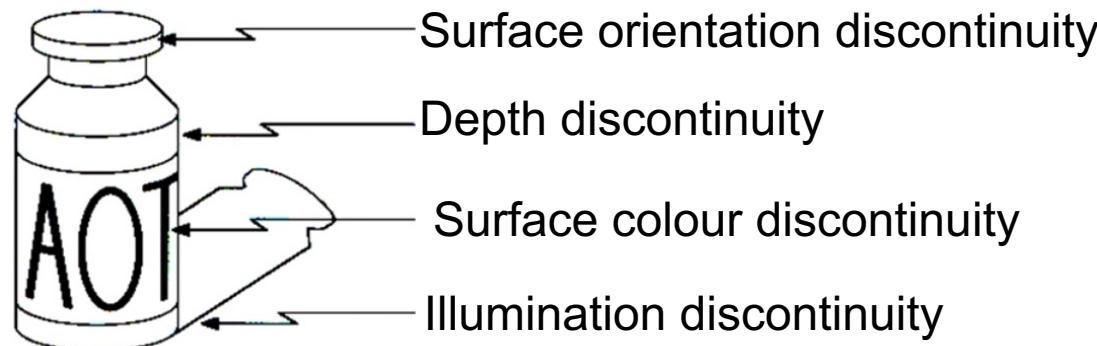
# Why do we care about edges?

- Motion analysis
  - Reliable object tracking



# Origin of Edges

- Geometric events
  - discontinuity in surface orientation
  - discontinuity in depth
  - discontinuity in surface colour/texture
- Non-Geometric events
  - Discontinuity in illumination (e.g. shadows from other objects or from the same object)





# Origin of Edges – Example

- Discontinuity in surface orientation
- Discontinuity in depth
- Discontinuity in surface colour/texture
- Discontinuity in illumination



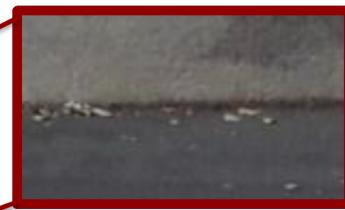


# Closeup of edges

- Which one?
  - Discontinuity in surface orientation
  - Discontinuity in depth
  - Discontinuity in surface colour/texture
  - Discontinuity in illumination



Discontinuity in surface orientation



# Closeup of edges

- Which one?
  - Discontinuity in surface orientation
  - Discontinuity in depth
  - Discontinuity in surface colour/texture
  - Discontinuity in illumination



Discontinuity in depth



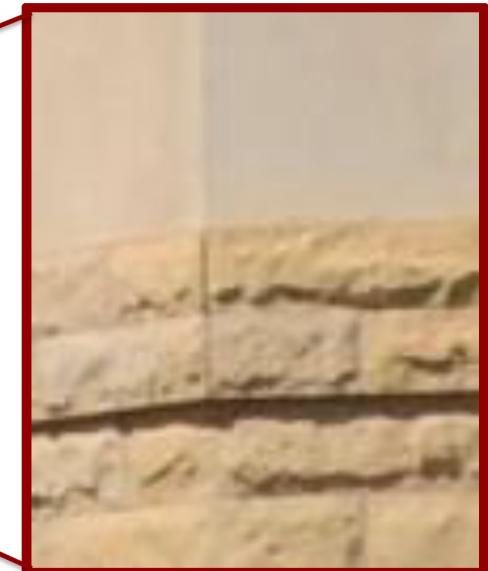


# Closeup of edges

- Which one?
  - Discontinuity in surface orientation
  - Discontinuity in depth
  - Discontinuity in surface colour/texture
  - Discontinuity in illumination



Discontinuity in surface colour





# Closeup of edges

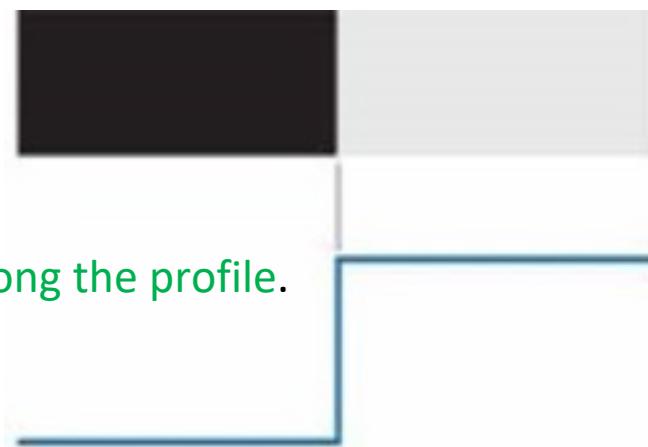
- Which one?
  - Discontinuity in surface orientation
  - Discontinuity in depth
  - Discontinuity in surface colour/texture
  - Discontinuity in illumination



Discontinuity in illumination

# Types of Edges

- **Step edge:** the image intensity **sharply** changes from one value to a different value.
- Step edges occur in images generated by a computer
- These clean, ideal edges can occur **over the distance of one pixel**, provided that no additional processing is used to make them look real.

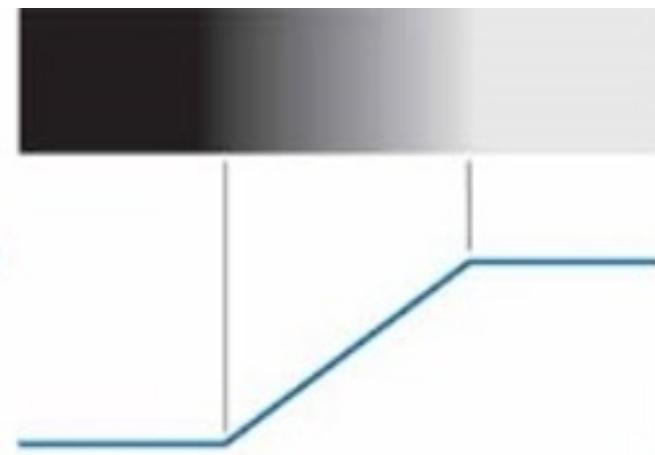


- A single “edge point” along the profile.

A step edge and its corresponding intensity profile

# Types of Edges

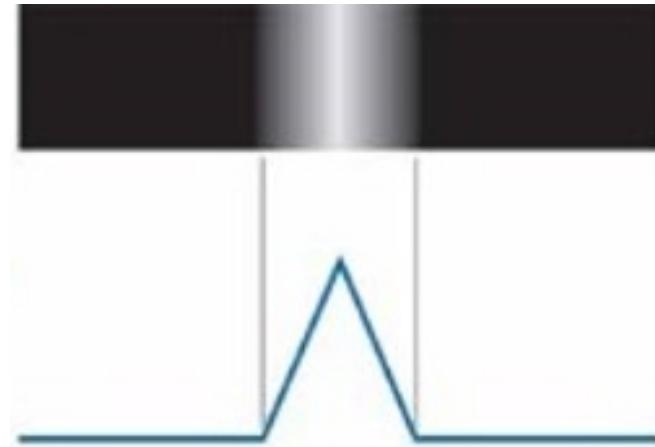
- **Ramp edge:** a step edge where the intensity change is not immediate but occurs over a finite distance
- Digital images have edges that are **blurred** and **noisy**,
  - Due to limitations in **the focusing mechanism** (e.g., lenses), and **the electronic components of the imaging system**.
- An edge point now is any point contained in the ramp



A ramp edge and its corresponding intensity profile

# Types of Edges

- **Roof edge:** the intensity changes from one value to another value over a finite distance and then returns to the starting value over a finite distance (generated usually by the intersection of surfaces).
- For example, in the digitization of **line drawings** and also in **satellite images**, where thin features, such as roads



A roof edge and its corresponding intensity profile

# Edge Detection & Derivative

---

- **Goal of edge detection:** Identify sudden changes (discontinuities) in an image
- **Steps of an edge detection algorithm:**
  - Calculate the rate of intensity change at each pixel in the image
    - How? Use the magnitude of gradient at each pixel (**derivatives!!!**)
  - Make a decision according to a threshold whether the pixel intensity changes abruptly.
    - How? Define a Threshold value

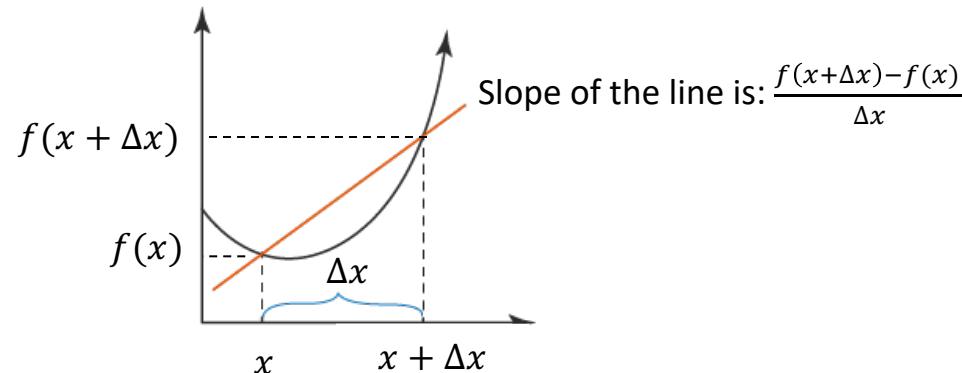
# Derivative in Mathematics

---

- Derivative is the rate of change of a **function** with respect to a **variable**.
- It represents the instantaneous rate of change at a **point on the function**.
- The derivative of a function  $f(x)$  is usually represented by  $\frac{df}{dx}$

# Derivative in Mathematics

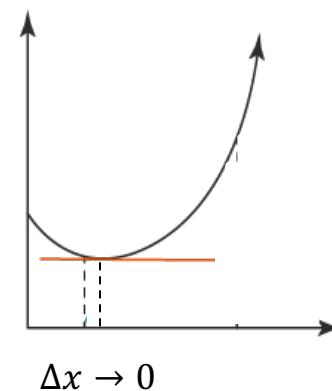
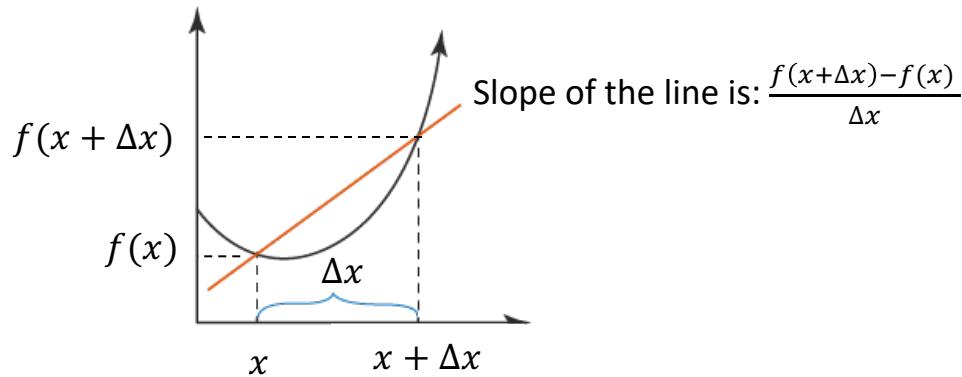
- Consider a curve of function  $f(x)$  and let two points on it be  $(x, f(x))$  and  $((x + \Delta x), f(x + \Delta x))$ .
- Then the slope of the secant line through these points is given by  $\frac{f(x+\Delta x)-f(x)}{\Delta x}$
- The **slope of the secant line** is also referred to as **the average rate of change** of function  $f(x)$  over the interval  $[x, x + \Delta x]$



# Derivative in Mathematics

- If we make the interval  $\Delta x$  smaller and smaller so that  $\Delta x \rightarrow 0$ , then the second point overlaps the original point and the secant line becomes the tangent line
- The slope of the tangent line is referred to as the **derivative of the function**. i.e.,

$$\frac{df}{dx} = f'(x) = \lim_{\Delta x \rightarrow 0} \frac{f(x + \Delta x) - f(x)}{\Delta x}$$



# Derivative in Mathematics

- SO, for a general function  $f(x)$ , the derivative  $f'(x)$  represents the **instantaneous rate of change** of  $f$  at  $x$ , i.e. the rate at which  $f$  changes at the “instant”  $x$ , and is described

$$\frac{df}{dx} = f'(x) = \lim_{\Delta x \rightarrow 0} \frac{f(x + \Delta x) - f(x)}{\Delta x}$$

OR

$$\frac{df}{dx} = f'(x) = \lim_{\Delta x \rightarrow 0} \frac{f(x) - f(x - \Delta x)}{\Delta x}$$

# Derivative in Mathematics

- **Example:** Find derivative of function  $f(x) = x$  using below derivative equation.

$$\frac{df}{dx} = f'(x) = \lim_{\Delta x \rightarrow 0} \frac{f(x + \Delta x) - f(x)}{\Delta x}$$

$f(x + \Delta x) = x + \Delta x$ 
 $f(x) = x$

→
←

$$\frac{df}{dx} = f'(x) = \lim_{\Delta x \rightarrow 0} \frac{(x + \Delta x) - x}{\Delta x} = \lim_{\Delta x \rightarrow 0} \frac{\Delta x}{\Delta x} = 1$$

# Derivative in Mathematics

- **Example:** Find derivative of function  $f(x) = x$  using below derivative equation.

$$\frac{df}{dx} = f'(x) = \lim_{\Delta x \rightarrow 0} \frac{f(x) - f(x - \Delta x)}{\Delta x}$$

$f(x) = x$ 
 $f(x - \Delta x) = x - \Delta x$

$$\frac{df}{dx} = f'(x) = \lim_{\Delta x \rightarrow 0} \frac{x - (x - \Delta x)}{\Delta x} = \lim_{\Delta x \rightarrow 0} \frac{\Delta x}{\Delta x} = 1$$

- Do you get the same result as in the previous slide?
  - YES

# Derivative in Mathematics

- **Question#1:** Find derivative of function  $f(x) = x^2$  using below derivative equations:

$$\frac{df}{dx} = f'(x) = \lim_{\Delta x \rightarrow 0} \frac{f(x + \Delta x) - f(x)}{\Delta x}$$

AND

$$\frac{df}{dx} = f'(x) = \lim_{\Delta x \rightarrow 0} \frac{f(x) - f(x - \Delta x)}{\Delta x}$$

# Image Derivative in 1D

- We can use first derivative to measure the instantaneous rate of change of function  $f(x)$  at each point  $x$

$$\frac{df}{dx} = f'(x) = \lim_{\Delta x \rightarrow 0} \frac{f(x + \Delta x) - f(x)}{\Delta x}$$

- Since the minimum distance between pixels is 1, the minimum value of  $\Delta x$  is  $\Delta x = 1$  and thus:

$$\frac{df}{dx} = f'(x) = \frac{f(x + 1) - f(x)}{1} = f(x + 1) - f(x)$$

**Forward difference**

# Image Derivative in 1D

- We can use first derivative to measure the instantaneous rate of change of function  $f(x)$  at each point  $x$

$$\frac{df}{dx} = f'(x) = \lim_{\Delta x \rightarrow 0} \frac{f(x) - f(x - \Delta x)}{\Delta x}$$

- Since the minimum distance between pixels is 1, the minimum value of  $\Delta x$  is  $\Delta x = 1$  and thus:

$$\frac{df}{dx} = f'(x) = \frac{f(x) - f(x - 1)}{1} = f(x) - f(x - 1)$$

**Backward difference**

# Image Derivative in 1D

- Using forward and backward difference equations:

**Forward difference**  $\frac{df}{dx} = f(x + 1) - f(x)$

**Backward difference**  $\frac{df}{dx} = f(x) - f(x - 1)$

- We can derive the central difference equation:

$$\frac{df}{dx} + \frac{df}{dx} = [f(x + 1) - f(x)] + [f(x) - f(x - 1)]$$

$$2 \frac{df}{dx} = f(x + 1) - f(x - 1)$$

**Central difference**  $\frac{df}{dx} = f'(x) = \frac{f(x + 1) - f(x - 1)}{2}$

# Image Derivative in 1D

- Forward, backward and central difference equations:

**Forward difference**

$$\frac{df}{dx} = f(x + 1) - f(x)$$

**Backward difference**

$$\frac{df}{dx} = f(x) - f(x - 1)$$

**Central difference**

$$\frac{df}{dx} = f(x + 1) - f(x - 1)$$

## Example: Derivative in 1D

- Calculate the derivative at each pixel using central difference

$$\frac{df}{dx} = f'(x) = f(x + 1) - f(x - 1)$$

$$f(x) = \begin{array}{cccccccccccc} 10 & 15 & 20 & 10 & 30 & 30 & 30 & 30 & 30 & 15 & 20 & 25 \end{array}$$

$$\frac{df}{dx} = \begin{array}{cccccccccccc} & & & & & & & & & & & \end{array}$$

## Example: Derivative in 1D

- Calculate the derivative at each pixel using central difference

$$\frac{df}{dx} = f'(x) = f(x + 1) - f(x - 1)$$

$$f(x) = \begin{array}{cccccccccccccc} 10 & 10 & 15 & 20 & 10 & 30 & 30 & 30 & 30 & 30 & 15 & 20 & 25 & 25 \end{array}$$

$$\frac{df}{dx} = \begin{array}{cccccccccccccc} & & & & & & & & & & & & & \end{array}$$

## Example: Derivative in 1D

- Calculate the derivative at each pixel using central difference

$$\frac{df}{dx} = f'(x) = f(x + 1) - f(x - 1)$$

$$f(x) = \begin{array}{cccccccccccccc} 10 & 10 & 15 & 20 & 10 & 30 & 30 & 30 & 30 & 30 & 15 & 20 & 25 & 25 \end{array}$$

$$\frac{df}{dx} = \begin{array}{cccccccccccccc} 5 & 10 & -5 & 10 & 20 & 0 & 0 & 0 & -15 & -10 & 10 & 5 \end{array}$$

# Convolution to Calculate Derivatives

---

- **Questions:**
  - Can we use convolution to calculate derivatives?
  - What is the filter corresponding to the central difference?
- Central difference:

$$\frac{df}{dx} = f(x + 1) - f(x - 1)$$

|    |   |   |
|----|---|---|
| -1 | 0 | 1 |
|----|---|---|

# Convolution to Calculate Derivatives

- Using central difference

$$\frac{df}{dx} = f'(x) = f(x + 1) - f(x - 1)$$

Filter

|    |   |   |
|----|---|---|
| -1 | 0 | 1 |
|----|---|---|

$f(x) =$

|    |    |    |    |    |    |    |    |    |    |    |    |
|----|----|----|----|----|----|----|----|----|----|----|----|
| 10 | 15 | 20 | 10 | 30 | 30 | 30 | 30 | 30 | 15 | 20 | 25 |
|----|----|----|----|----|----|----|----|----|----|----|----|

$$\frac{df}{dx} =$$

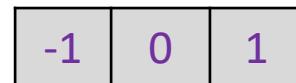
|  |  |  |  |  |  |  |  |  |  |  |  |
|--|--|--|--|--|--|--|--|--|--|--|--|
|  |  |  |  |  |  |  |  |  |  |  |  |
|--|--|--|--|--|--|--|--|--|--|--|--|

# Convolution to Calculate Derivatives

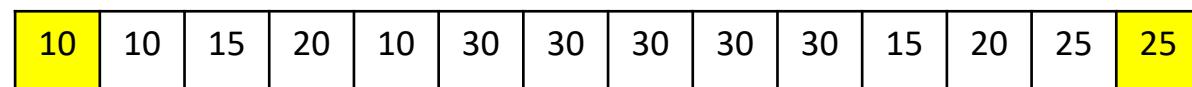
- **Replication** OR Zero padding

$$\frac{df}{dx} = f'(x) = f(x + 1) - f(x - 1)$$

Filter



$f(x) =$



$\frac{df}{dx} =$



# Convolution to Calculate Derivatives

- Convolution#1

$$\frac{df}{dx} = f'(x) = f(x + 1) - f(x - 1)$$

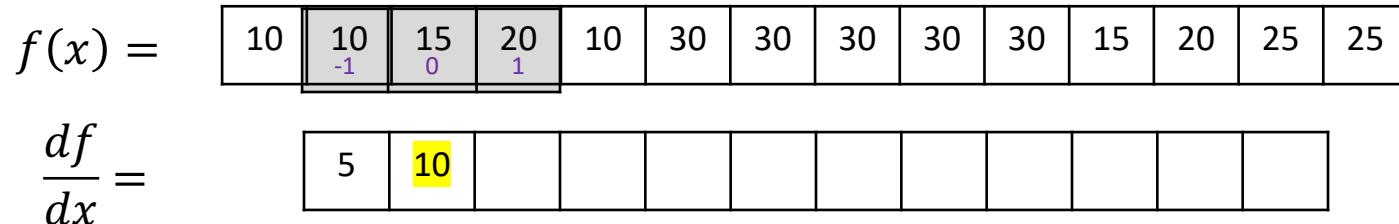
$$f(x) = \begin{array}{cccccccccccccc} 10 & 10 & 15 & 20 & 10 & 30 & 30 & 30 & 30 & 30 & 15 & 20 & 25 & 25 \\ \textcolor{purple}{-1} & \textcolor{purple}{0} & \textcolor{purple}{1} & & & & & & & & & & & & \end{array}$$

$$\frac{df}{dx} = \begin{array}{cccccccccccccc} 5 & & & & & & & & & & & & & & \end{array}$$

# Convolution to Calculate Derivatives

- Convolution#2

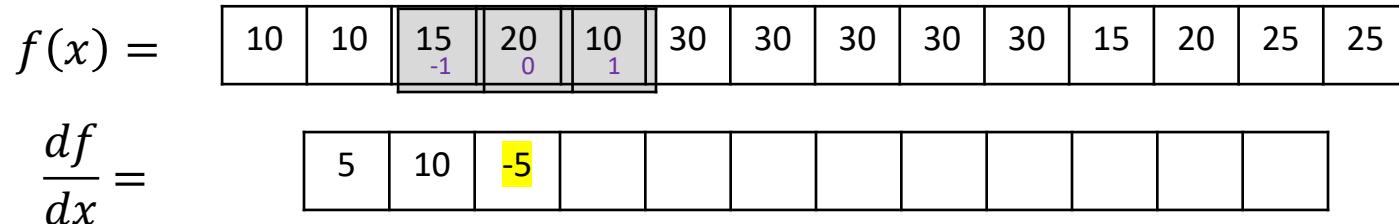
$$\frac{df}{dx} = f'(x) = f(x + 1) - f(x - 1)$$



# Convolution to Calculate Derivatives

- Convolution#3

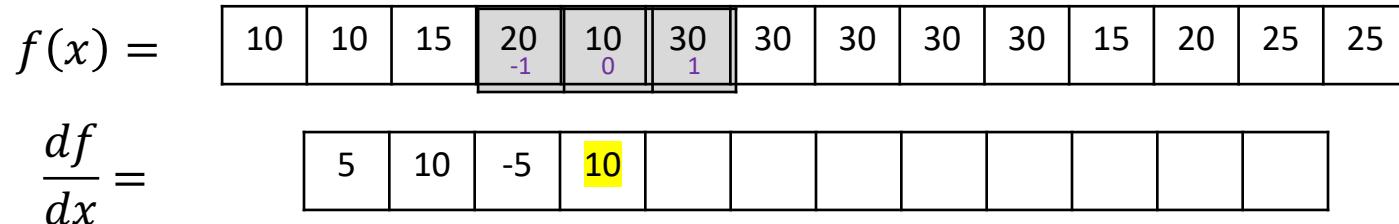
$$\frac{df}{dx} = f'(x) = f(x + 1) - f(x - 1)$$



# Convolution to Calculate Derivatives

- Convolution#4

$$\frac{df}{dx} = f'(x) = f(x + 1) - f(x - 1)$$





# Convolution to Calculate Derivatives

- Final result

$$\frac{df}{dx} = f'(x) = f(x + 1) - f(x - 1)$$

$$f(x) = \begin{array}{|c|c|c|c|c|c|c|c|c|c|c|c|c|c|c|} \hline 10 & 10 & 15 & 20 & 10 & 30 & 30 & 30 & 30 & 30 & 15 & 20 & 25 & 25 \\ \hline \end{array}$$

$$\frac{df}{dx} = \begin{array}{|c|c|c|c|c|c|c|c|c|c|c|c|c|c|c|} \hline 5 & 10 & -5 & 10 & 20 & 0 & 0 & 0 & -15 & -10 & 10 & 5 \\ \hline \end{array}$$

# 2D Image Derivatives

- An image is a function of two variables:  $f(x, y)$
- So the derivative of an image has two dimensions
- We can take the derivative in the  $x$  direction (row-wise) and in the  $y$  direction (column-wise)

$$f(8,2) = 17$$



|     |     |     |     |     |     |     |     |     |     |     |     |  |  |  |  |  |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|--|--|--|--|--|
| 157 | 153 | 174 | 168 | 150 | 152 | 129 | 151 | 172 | 161 | 155 | 156 |  |  |  |  |  |
| 155 | 182 | 163 | 74  | 75  | 62  | 93  | 17  | 110 | 210 | 180 | 154 |  |  |  |  |  |
| 180 | 180 | 50  | 14  | 34  | 6   | 10  | 53  | 48  | 106 | 159 | 181 |  |  |  |  |  |
| 206 | 109 | 5   | 124 | 131 | 111 | 120 | 204 | 166 | 15  | 56  | 180 |  |  |  |  |  |
| 194 | 68  | 137 | 261 | 237 | 239 | 239 | 228 | 227 | 87  | 71  | 201 |  |  |  |  |  |
| 172 | 105 | 207 | 233 | 233 | 214 | 220 | 239 | 228 | 98  | 74  | 206 |  |  |  |  |  |
| 188 | 68  | 179 | 269 | 185 | 215 | 211 | 158 | 139 | 75  | 20  | 169 |  |  |  |  |  |
| 189 | 97  | 165 | 84  | 10  | 168 | 134 | 11  | 31  | 62  | 22  | 148 |  |  |  |  |  |
| 199 | 168 | 191 | 193 | 158 | 227 | 178 | 143 | 182 | 106 | 36  | 190 |  |  |  |  |  |
| 206 | 174 | 166 | 262 | 236 | 231 | 149 | 178 | 228 | 43  | 95  | 294 |  |  |  |  |  |
| 190 | 216 | 116 | 149 | 236 | 187 | 85  | 150 | 79  | 38  | 218 | 241 |  |  |  |  |  |
| 190 | 234 | 147 | 108 | 227 | 210 | 127 | 102 | 36  | 101 | 255 | 224 |  |  |  |  |  |
| 190 | 214 | 173 | 46  | 103 | 143 | 95  | 50  | 2   | 109 | 249 | 215 |  |  |  |  |  |
| 187 | 196 | 236 | 75  | 1   | 81  | 47  | 0   | 6   | 217 | 255 | 211 |  |  |  |  |  |
| 183 | 202 | 237 | 145 | 0   | 0   | 12  | 108 | 200 | 138 | 243 | 236 |  |  |  |  |  |
| 195 | 206 | 123 | 207 | 177 | 121 | 123 | 200 | 175 | 12  | 96  | 218 |  |  |  |  |  |

|     |     |     |     |     |     |     |     |     |     |     |     |  |  |  |  |  |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|--|--|--|--|--|
| 157 | 153 | 174 | 168 | 150 | 152 | 129 | 151 | 172 | 161 | 155 | 156 |  |  |  |  |  |
| 155 | 182 | 163 | 74  | 75  | 62  | 93  | 17  | 110 | 210 | 180 | 154 |  |  |  |  |  |
| 180 | 180 | 50  | 14  | 34  | 6   | 10  | 53  | 48  | 106 | 159 | 181 |  |  |  |  |  |
| 206 | 109 | 5   | 124 | 131 | 111 | 120 | 204 | 166 | 15  | 56  | 180 |  |  |  |  |  |
| 194 | 68  | 137 | 261 | 237 | 239 | 239 | 228 | 227 | 87  | 71  | 201 |  |  |  |  |  |
| 172 | 105 | 207 | 233 | 233 | 214 | 220 | 239 | 228 | 98  | 74  | 206 |  |  |  |  |  |
| 188 | 68  | 179 | 269 | 185 | 215 | 211 | 158 | 139 | 75  | 20  | 169 |  |  |  |  |  |
| 189 | 97  | 165 | 84  | 10  | 168 | 134 | 11  | 31  | 62  | 22  | 148 |  |  |  |  |  |
| 199 | 168 | 191 | 193 | 158 | 227 | 178 | 143 | 182 | 106 | 36  | 190 |  |  |  |  |  |
| 206 | 174 | 166 | 262 | 236 | 231 | 149 | 178 | 228 | 43  | 95  | 294 |  |  |  |  |  |
| 190 | 216 | 116 | 149 | 236 | 187 | 85  | 150 | 79  | 38  | 218 | 241 |  |  |  |  |  |
| 190 | 234 | 147 | 108 | 227 | 210 | 127 | 102 | 36  | 101 | 255 | 224 |  |  |  |  |  |
| 190 | 214 | 173 | 46  | 103 | 143 | 95  | 50  | 2   | 109 | 249 | 215 |  |  |  |  |  |
| 187 | 196 | 236 | 75  | 1   | 81  | 47  | 0   | 6   | 217 | 255 | 211 |  |  |  |  |  |
| 183 | 202 | 237 | 145 | 0   | 0   | 12  | 108 | 200 | 138 | 243 | 236 |  |  |  |  |  |
| 195 | 206 | 123 | 207 | 177 | 121 | 123 | 200 | 175 | 12  | 96  | 218 |  |  |  |  |  |

y

x

# SCC.366: Media Coding & Processing

2022-2023

Week 05 – Lecture 2

## Edge Detection Cont.

Dr. Hossein Rahmani

Senior Lecturer in Data Science

Email and Team: [h.rahmani@lancaster.ac.uk](mailto:h.rahmani@lancaster.ac.uk)

# Outline

---

- What is an Edge?
- Why do we care about edges?
- Origin of edges
- First order edge detectors
- Edge detection using Laplacian
- Canny edge detector
- Summary of the main points

# Review - Image Derivative in 1D

- Forward, backward and central difference equations:

**Forward difference**

$$\frac{df}{dx} = f(x + 1) - f(x)$$

**Backward difference**

$$\frac{df}{dx} = f(x) - f(x - 1)$$

**Central difference**

$$\frac{df}{dx} = f(x + 1) - f(x - 1)$$

# Review – Derivatives as Convolution

---

- **Questions:**
  - Can we use convolution to calculate derivatives?
  - What is the filter corresponding to the central difference?
- Central difference:

$$\frac{df}{dx} = f(x + 1) - f(x - 1)$$

|    |   |   |
|----|---|---|
| -1 | 0 | 1 |
|----|---|---|

# 2D Image Derivatives

- An image is a function of two variables:  $f(x, y)$
- So the derivative of an image has two dimensions
- We can take the derivative in the  $x$  direction (row-wise) and in the  $y$  direction (column-wise)

$$f(8,2) = 17$$



|     |     |     |     |     |     |     |     |     |     |     |     |  |  |  |  |  |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|--|--|--|--|--|
| 157 | 153 | 174 | 168 | 150 | 152 | 129 | 151 | 172 | 161 | 155 | 156 |  |  |  |  |  |
| 155 | 182 | 163 | 74  | 75  | 62  | 93  | 17  | 110 | 210 | 180 | 154 |  |  |  |  |  |
| 180 | 180 | 50  | 14  | 34  | 6   | 10  | 53  | 48  | 106 | 159 | 181 |  |  |  |  |  |
| 206 | 109 | 5   | 124 | 131 | 111 | 120 | 204 | 166 | 15  | 56  | 180 |  |  |  |  |  |
| 194 | 68  | 137 | 261 | 237 | 239 | 239 | 228 | 227 | 87  | 71  | 201 |  |  |  |  |  |
| 172 | 105 | 207 | 233 | 233 | 214 | 220 | 239 | 228 | 98  | 74  | 206 |  |  |  |  |  |
| 188 | 68  | 179 | 269 | 185 | 215 | 211 | 158 | 139 | 75  | 20  | 169 |  |  |  |  |  |
| 189 | 97  | 165 | 84  | 10  | 168 | 134 | 11  | 31  | 62  | 22  | 148 |  |  |  |  |  |
| 199 | 168 | 191 | 193 | 158 | 227 | 178 | 143 | 182 | 106 | 36  | 190 |  |  |  |  |  |
| 206 | 174 | 166 | 262 | 236 | 231 | 149 | 178 | 228 | 43  | 95  | 294 |  |  |  |  |  |
| 190 | 216 | 116 | 149 | 236 | 187 | 85  | 150 | 79  | 38  | 218 | 241 |  |  |  |  |  |
| 190 | 234 | 147 | 108 | 227 | 210 | 127 | 102 | 36  | 101 | 255 | 224 |  |  |  |  |  |
| 190 | 214 | 173 | 46  | 103 | 143 | 95  | 50  | 2   | 109 | 249 | 215 |  |  |  |  |  |
| 187 | 196 | 236 | 75  | 1   | 81  | 47  | 0   | 6   | 217 | 255 | 211 |  |  |  |  |  |
| 183 | 202 | 237 | 145 | 0   | 0   | 12  | 108 | 200 | 138 | 243 | 236 |  |  |  |  |  |
| 195 | 206 | 123 | 207 | 177 | 121 | 123 | 200 | 175 | 12  | 96  | 218 |  |  |  |  |  |

|     |     |     |     |     |     |     |     |     |     |     |     |  |  |  |  |  |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|--|--|--|--|--|
| 157 | 153 | 174 | 168 | 150 | 152 | 129 | 151 | 172 | 161 | 155 | 156 |  |  |  |  |  |
| 155 | 182 | 163 | 74  | 75  | 62  | 93  | 17  | 110 | 210 | 180 | 154 |  |  |  |  |  |
| 180 | 180 | 50  | 14  | 34  | 6   | 10  | 53  | 48  | 106 | 159 | 181 |  |  |  |  |  |
| 206 | 109 | 5   | 124 | 131 | 111 | 120 | 204 | 166 | 15  | 56  | 180 |  |  |  |  |  |
| 194 | 68  | 137 | 261 | 237 | 239 | 239 | 228 | 227 | 87  | 71  | 201 |  |  |  |  |  |
| 172 | 105 | 207 | 233 | 233 | 214 | 220 | 239 | 228 | 98  | 74  | 206 |  |  |  |  |  |
| 188 | 68  | 179 | 269 | 185 | 215 | 211 | 158 | 139 | 75  | 20  | 169 |  |  |  |  |  |
| 189 | 97  | 165 | 84  | 10  | 168 | 134 | 11  | 31  | 62  | 22  | 148 |  |  |  |  |  |
| 199 | 168 | 191 | 193 | 158 | 227 | 178 | 143 | 182 | 106 | 36  | 190 |  |  |  |  |  |
| 206 | 174 | 166 | 262 | 236 | 231 | 149 | 178 | 228 | 43  | 95  | 294 |  |  |  |  |  |
| 190 | 216 | 116 | 149 | 236 | 187 | 85  | 150 | 79  | 38  | 218 | 241 |  |  |  |  |  |
| 190 | 234 | 147 | 108 | 227 | 210 | 127 | 102 | 36  | 101 | 255 | 224 |  |  |  |  |  |
| 190 | 214 | 173 | 46  | 103 | 143 | 95  | 50  | 2   | 109 | 249 | 215 |  |  |  |  |  |
| 187 | 196 | 236 | 75  | 1   | 81  | 47  | 0   | 6   | 217 | 255 | 211 |  |  |  |  |  |
| 183 | 202 | 237 | 145 | 0   | 0   | 12  | 108 | 200 | 138 | 243 | 236 |  |  |  |  |  |
| 195 | 206 | 123 | 207 | 177 | 121 | 123 | 200 | 175 | 12  | 96  | 218 |  |  |  |  |  |

y

x

# Image Derivatives (Example)

- An image is a function of two variables:  $f(x, y)$
- So the derivative of an image has two dimensions.
- We can take the derivative in the  $x$  direction (row-wise) and in the  $y$  direction (column-wise)

|  |  |                   |               |                   |  |  |  |  |  |
|--|--|-------------------|---------------|-------------------|--|--|--|--|--|
|  |  |                   |               |                   |  |  |  |  |  |
|  |  |                   |               |                   |  |  |  |  |  |
|  |  | $f(x - 1, y - 1)$ | $f(x, y - 1)$ | $f(x + 1, y - 1)$ |  |  |  |  |  |
|  |  | $f(x - 1, y)$     | $f(x, y)$     | $f(x + 1, y)$     |  |  |  |  |  |
|  |  | $f(x - 1, y + 1)$ | $f(x, y + 1)$ | $f(x + 1, y + 1)$ |  |  |  |  |  |
|  |  |                   |               |                   |  |  |  |  |  |
|  |  |                   |               |                   |  |  |  |  |  |
|  |  |                   |               |                   |  |  |  |  |  |
|  |  |                   |               |                   |  |  |  |  |  |
|  |  |                   |               |                   |  |  |  |  |  |

# Partial Derivatives

- We can use backward, forward or central difference
- Here, we use central difference:

$$\frac{df}{dx} = f(x + 1) - f(x - 1)$$

- In mathematics, a **partial derivative** of a function of several variables (e.g.,  $f(x, y)$ ) is its derivative with respect to one of those variables, with the others held constant, i.e.,

Partial derivative in the  $x$  direction

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

Partial derivative in the  $y$  direction

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

# Partial Derivative along $x$ axis (Example)

- Partial derivative in the  $x$  direction is:

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

- How would you implement this as a filter?

|    |   |   |
|----|---|---|
| -1 | 0 | 1 |
|----|---|---|

|     |  |                   |               |                   |  |  |  |  |  |
|-----|--|-------------------|---------------|-------------------|--|--|--|--|--|
|     |  |                   |               |                   |  |  |  |  |  |
|     |  |                   |               |                   |  |  |  |  |  |
|     |  | $f(x - 1, y - 1)$ | $f(x, y - 1)$ | $f(x + 1, y - 1)$ |  |  |  |  |  |
| $y$ |  | $f(x - 1, y)$     | $f(x, y)$     | $f(x + 1, y)$     |  |  |  |  |  |
|     |  | $f(x - 1, y + 1)$ | $f(x, y + 1)$ | $f(x + 1, y + 1)$ |  |  |  |  |  |
|     |  |                   |               |                   |  |  |  |  |  |
|     |  |                   |               |                   |  |  |  |  |  |
|     |  |                   |               |                   |  |  |  |  |  |
|     |  |                   |               |                   |  |  |  |  |  |
|     |  |                   |               |                   |  |  |  |  |  |

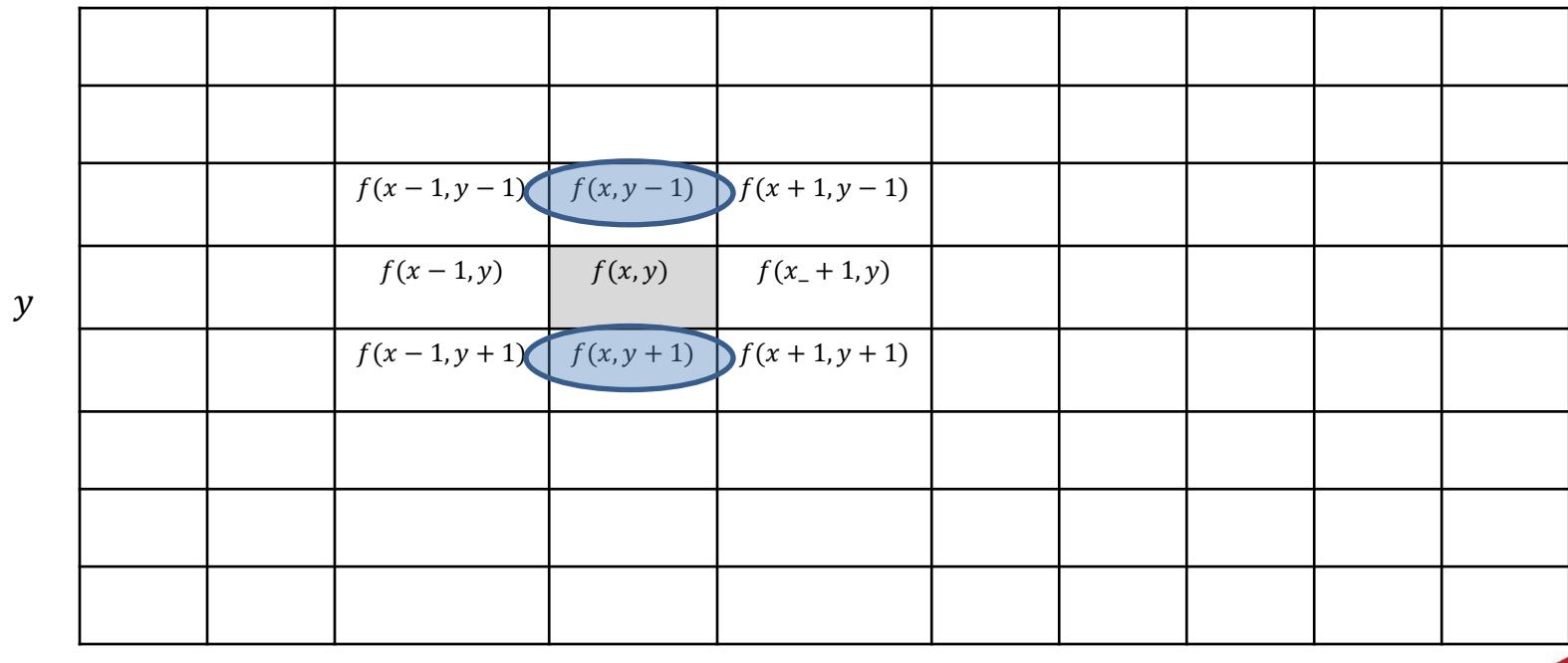
# Partial Derivative along $y$ axis (Example)

- Partial derivative in the  $y$  direction is:

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

- How would you implement this as a filter?

|    |
|----|
| -1 |
| 0  |
| 1  |

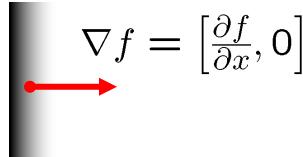


# Image Gradient

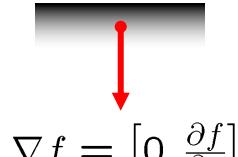
- The *gradient* of an image at each pixel is a vector:

$$\nabla f = \left[ \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right]$$

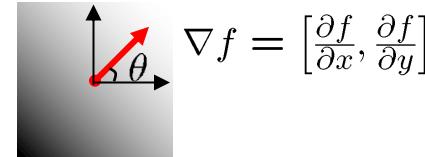
- The gradient points in the direction of most rapid increase in intensity



$$\nabla f = \left[ \frac{\partial f}{\partial x}, 0 \right]$$

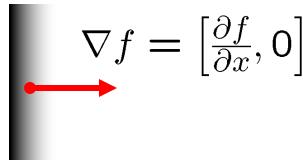


$$\nabla f = \left[ 0, \frac{\partial f}{\partial y} \right]$$

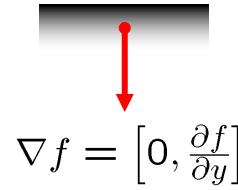


$$\nabla f = \left[ \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right]$$

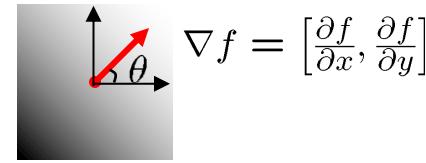
# Image Gradient



$$\nabla f = \left[ \frac{\partial f}{\partial x}, 0 \right]$$



$$\nabla f = \left[ 0, \frac{\partial f}{\partial y} \right]$$



$$\nabla f = \left[ \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right]$$

- The edge strength is given by the gradient magnitude:

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

$$||\nabla f|| = \left| \frac{\partial f}{\partial x} \right| + \left| \frac{\partial f}{\partial y} \right|$$

More efficient approximation  
of the gradient magnitude

- The gradient direction is given by:

$$\theta = \tan^{-1} \left( \frac{\partial f}{\partial y} / \frac{\partial f}{\partial x} \right)$$

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

# Image Gradient (Example)

- **Question:** Calculate the gradient of the below image at center pixel.

|         | Col x-1 | Col x | Col x+1 |
|---------|---------|-------|---------|
| Row y-1 | 28      | 27    | 26      |
| Row y   | 30      | 29    | 31      |
| Row y+1 | 15      | 17    | 16      |

# Image Gradient (Example)

- The gradient of an image at each pixel is a vector:

$$\nabla f = \left[ \frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right]$$

|         | Col x-1 | Col x | Col x+1 |
|---------|---------|-------|---------|
| Row y-1 | 28      | 27    | 26      |
| Row y   | 30      | 29    | 31      |
| Row y+1 | 15      | 17    | 16      |

# Image Gradient (Example)

- Partial derivative in the  $x$  direction is:

$$\frac{\partial f}{\partial x} = f(x + 1, y) - f(x - 1, y) = 31 - 30 = 1$$

|    |   |   |
|----|---|---|
| -1 | 0 | 1 |
|----|---|---|

$$\frac{\partial f}{\partial x} = -1 \times 30 + 0 \times 29 + 1 \times 31 = 1$$

|         | Col x-1 | Col x | Col x+1 |
|---------|---------|-------|---------|
| Row y-1 | 28      | 27    | 26      |
| Row y   | 30      | 29    | 31      |
| Row y+1 | 15      | 17    | 16      |

# Image Gradient (Example)

- Partial derivative in the  $y$  direction is:

$$\frac{\partial f}{\partial x} = f(x, y + 1) - f(x, y - 1) = 17 - 27 = -10$$

|    |
|----|
| -1 |
| 0  |
| 1  |

|         | Col x-1 | Col x | Col x+1 |
|---------|---------|-------|---------|
| Row y-1 | 28      | 27    | 26      |
| Row y   | 30      | 29    | 31      |
| Row y+1 | 15      | 17    | 16      |

$$\frac{\partial f}{\partial x} = -1 \times 27 + 0 \times 29 + 1 \times 17 = -10$$

## Image Gradient (Example)

- Partial derivative in the  $x$  direction is:  $\frac{\partial f}{\partial x} = 1$
- Partial derivative in the  $y$  direction is:  $\frac{\partial f}{\partial y} = -10$
- The gradient magnitude is:

$$||\nabla f|| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2} = \sqrt{(1)^2 + (-10)^2} = 10.05$$

## Image Gradient (Example)

- Partial derivative in the  $x$  direction is:  $\frac{\partial f}{\partial x} = 1$
- Partial derivative in the  $y$  direction is:  $\frac{\partial f}{\partial y} = -10$
- The gradient magnitude is:

$$||\nabla f|| = \left| \frac{\partial f}{\partial x} \right| + \left| \frac{\partial f}{\partial y} \right| = 1 + 10 = 11$$

More efficient approximation  
of the gradient magnitude

# Image Gradient (Example)

- Partial derivative in the  $x$  direction is:  $\frac{\partial f}{\partial x} = 1$
- Partial derivative in the  $y$  direction is:  $\frac{\partial f}{\partial y} = -10$
- The gradient magnitude is:

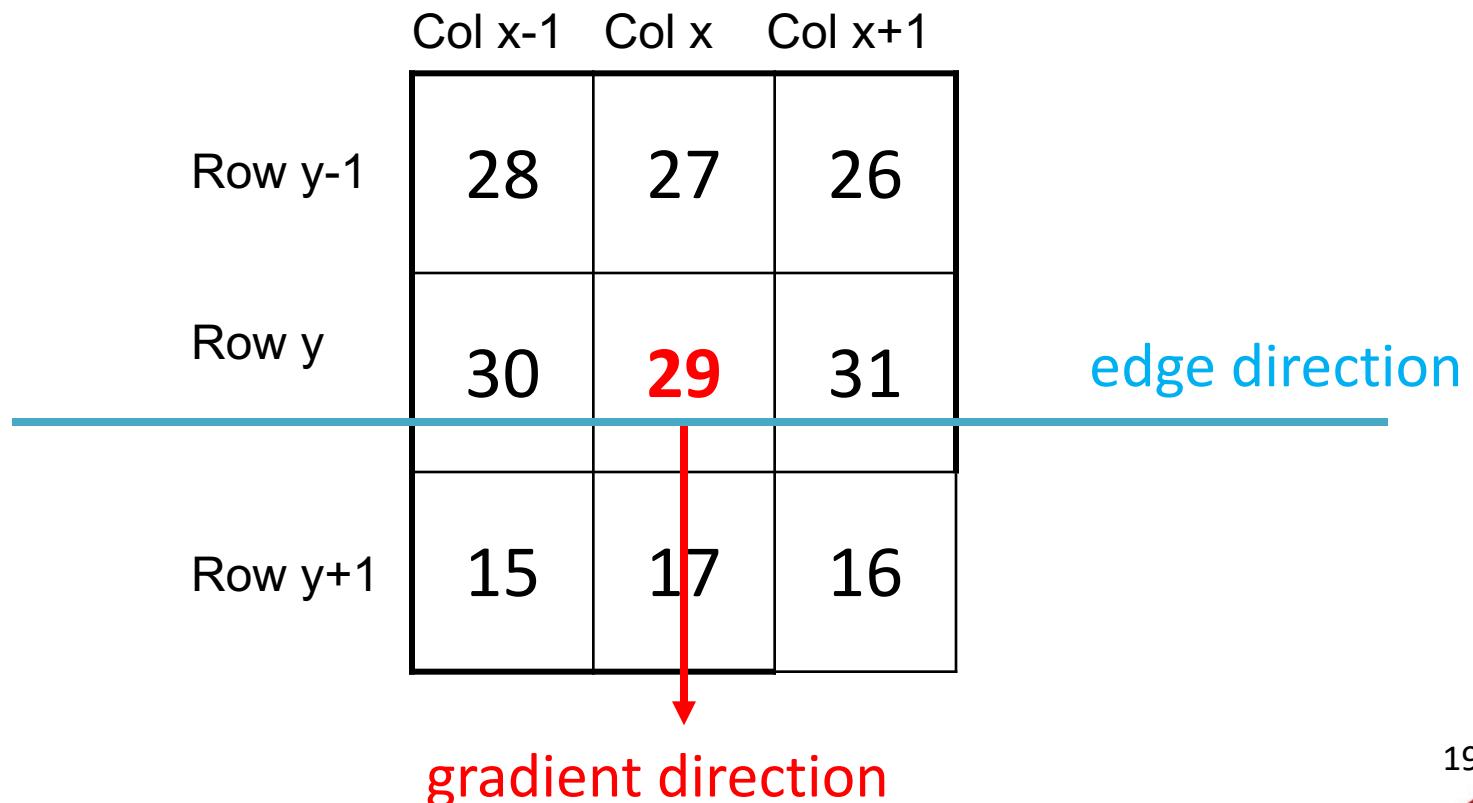
$$||\nabla f|| = \left| \frac{\partial f}{\partial x} \right| + \left| \frac{\partial f}{\partial y} \right| = 1 + 10 = 11$$

- The gradient direction is given by:

$$\theta = \tan^{-1} \left( \frac{\partial f}{\partial y} / \frac{\partial f}{\partial x} \right) = \tan^{-1}(-10) = -84.3^\circ$$

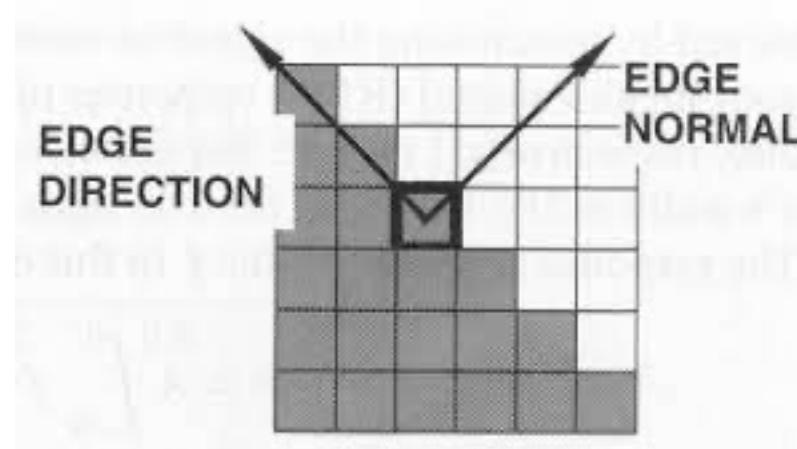
# Image Gradient (Example)

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



# Gradient Direction vs. Edge Direction

- How does gradient direction (also called edge normal) relate to the direction of the edge?



- The edge direction is **perpendicular** to the direction of the gradient vector at the point where the gradient is computed.

# Summary

---

- Calculate partial derivative for each pixel along the  $x$  direction is:

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

|    |   |   |
|----|---|---|
| -1 | 0 | 1 |
|----|---|---|

- Calculate partial derivative for each pixel along the  $y$  direction is:

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

|    |
|----|
| -1 |
| 0  |
| 1  |

- Thus, we have two output images corresponding to the above two filters

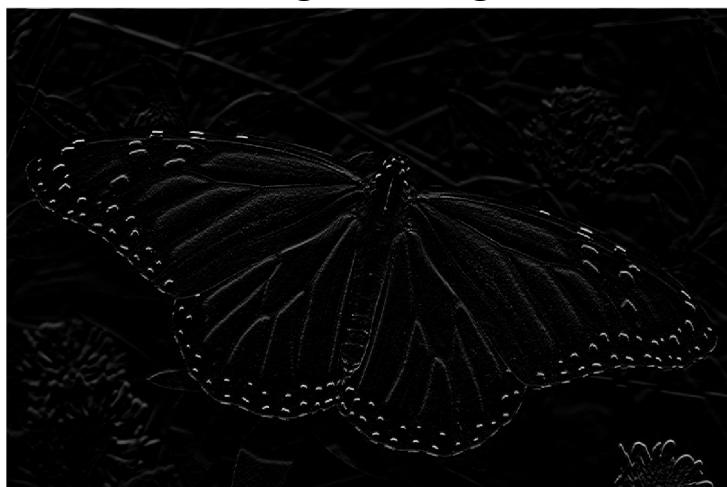
# Image Gradient (Visualisation)



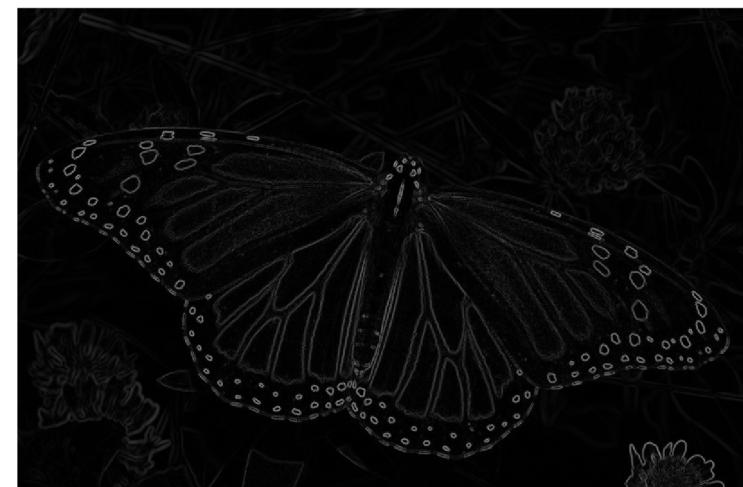
Original image



Partial derivative in the  $x$  direction



Partial derivative in the  $y$  direction



The gradient magnitude

# What are these filters?

- Sobel edge detector

$$\frac{\partial f}{\partial x}$$

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

$$\frac{\partial f}{\partial y}$$

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

- Prewitt edge detector

$$\frac{\partial f}{\partial x}$$

|    |   |   |
|----|---|---|
| -1 | 0 | 1 |
| -1 | 0 | 1 |
| -1 | 0 | 1 |

$$\frac{\partial f}{\partial y}$$

|    |    |    |
|----|----|----|
| -1 | -1 | -1 |
| 0  | 0  | 0  |
| 1  | 1  | 1  |

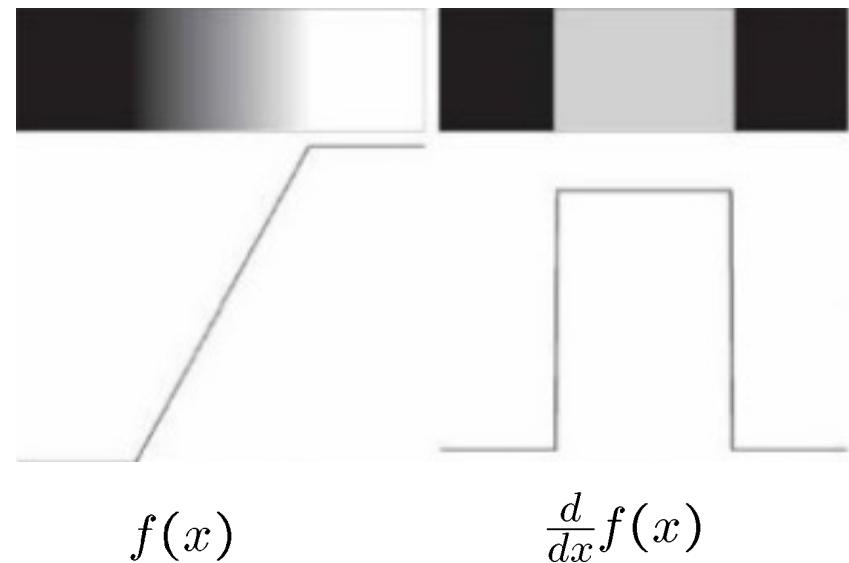
- Roberts edge detector

|   |    |
|---|----|
| 1 | 0  |
| 0 | -1 |

|    |   |
|----|---|
| 0  | 1 |
| -1 | 0 |

# Effects of Noise

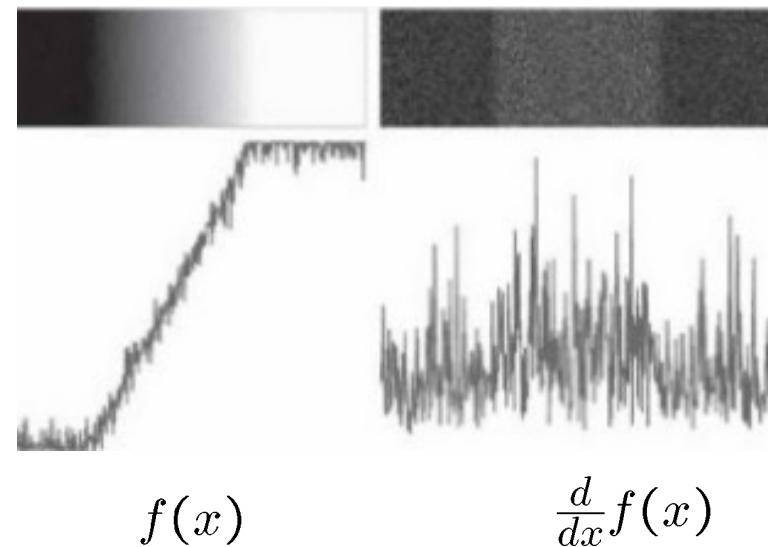
- Intensity profile of a ramp edge (**no noise**)





# Effects of Noise

- Intensity profile of a ramp edge **corrupted by noise**



# Smoothing Before Edge Detection

---

- To resolve this issue, we need to smooth the input
- How?
  - Apply a smoothing filter such as
    - Average filter → Prewitt Edge Detector
    - Gaussian filter → Sobel Edge Detector

# Prewitt Edge Detector

- Edge in  $x$  direction
  - First: Original image is convolved with an average filter such as: to get a new smooth image
  - Second: The smooth image is convolved with a derivative filter such as central difference  to get edges in the  $x$  direction i.e.,  $\frac{\partial f}{\partial x}$

# Prewitt Edge Detector - Example

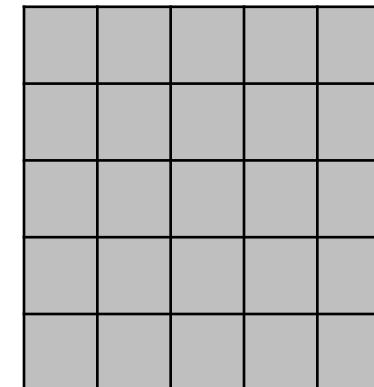
- Edge in  $x$  direction
  - First: Original image is convolved with an average filter to get a new smooth image

|   |   |   |   |   |
|---|---|---|---|---|
| 3 | 3 | 3 | 1 | 0 |
| 2 | 1 | 2 | 2 | 0 |
| 2 | 0 | 2 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 2 | 1 |

Image:  $F$

Filter (3,1)

$$\frac{1}{3} \times \begin{array}{|c|c|c|}\hline 1 \\ \hline 1 \\ \hline 1 \\ \hline \end{array}$$



Smooth Image

Filter/Kernel:  $H$

# Prewitt Edge Detector - Example

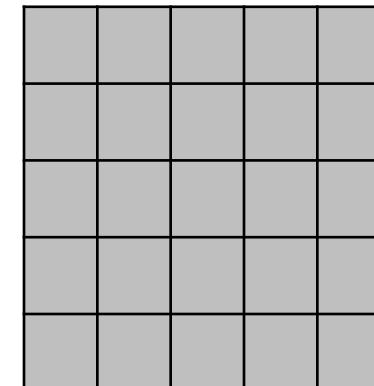
- Edge in  $x$  direction
  - First: Original image is convolved with an average filter to get a new smooth image (e.g., value padding)

| 3 | 3 | 3 | 1 | 0 |
|---|---|---|---|---|
| 3 | 3 | 3 | 1 | 0 |
| 2 | 1 | 2 | 2 | 0 |
| 2 | 0 | 2 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 2 | 1 |

Image:  $F$

Filter (3,1)

$$\frac{1}{3} \times \begin{array}{|c|} \hline 1 \\ \hline 1 \\ \hline 1 \\ \hline \end{array}$$



Smooth Image

Filter/Kernel:  $H$

# Prewitt Edge Detector - Example

- Edge in  $x$  direction
  - First: Original image is convolved with an average filter to get a new smooth image

|           |   |   |   |   |
|-----------|---|---|---|---|
| <b>31</b> | 3 | 3 | 1 | 0 |
| <b>31</b> | 3 | 3 | 1 | 0 |
| <b>21</b> | 1 | 2 | 2 | 0 |
| 2         | 0 | 2 | 0 | 1 |
| 1         | 0 | 0 | 0 | 1 |
| 1         | 1 | 0 | 2 | 1 |
| 1         | 1 | 0 | 2 | 1 |

Image:  $F$

Filter (3,1)

$$\frac{1}{3} \times \begin{array}{|c|} \hline 1 \\ \hline 1 \\ \hline 1 \\ \hline \end{array}$$

|     |  |  |  |  |
|-----|--|--|--|--|
| 2.7 |  |  |  |  |
|     |  |  |  |  |
|     |  |  |  |  |
|     |  |  |  |  |
|     |  |  |  |  |

Filter/Kernel:  $H$

Smooth Image

# Prewitt Edge Detector - Example

- Edge in  $x$  direction
  - First: Original image is convolved with an average filter to get a new smooth image

|   |   |   |   |   |   |
|---|---|---|---|---|---|
| 3 | 3 | 1 | 3 | 1 | 0 |
| 3 | 3 | 1 | 3 | 1 | 0 |
| 2 | 1 | 1 | 2 | 2 | 0 |
| 2 | 0 | 2 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 2 | 1 |   |

1    1    0    2    1

Image:  $F$

Filter (3,1)

$$\frac{1}{3} \times \begin{array}{|c|} \hline 1 \\ \hline 1 \\ \hline 1 \\ \hline \end{array}$$

|     |     |  |  |  |
|-----|-----|--|--|--|
| 2.7 | 2.3 |  |  |  |
|     |     |  |  |  |
|     |     |  |  |  |
|     |     |  |  |  |
|     |     |  |  |  |

Smooth Image

# Prewitt Edge Detector - Example

- Edge in  $x$  direction
  - First: Original image is convolved with an average filter to get a new smooth image

| 3 | 3 | 3 | 1 | 0  |
|---|---|---|---|----|
| 3 | 3 | 3 | 1 | 0  |
| 2 | 1 | 2 | 2 | 0  |
| 2 | 0 | 2 | 0 | 1  |
| 1 | 0 | 0 | 0 | 11 |
| 1 | 1 | 0 | 2 | 11 |
| 1 | 1 | 0 | 2 | 11 |

Image:  $F$

Filter (3,1)

$$\frac{1}{3} \times \begin{array}{|c|} \hline 1 \\ \hline 1 \\ \hline 1 \\ \hline \end{array}$$

|     |     |     |     |     |
|-----|-----|-----|-----|-----|
| 2.7 | 2.3 | 2.7 | 1.3 | 0   |
| 2.3 | 1.3 | 2.3 | 1   | 0.3 |
| 1.7 | 0.3 | 1.3 | 0.7 | 0.7 |
| 1.3 | 0.3 | 0.7 | 0.7 | 1   |
| 1   | 0.7 | 0   | 1.3 | 1   |

Filter/Kernel:  $H$

Smooth Image

# Prewitt Edge Detector - Example

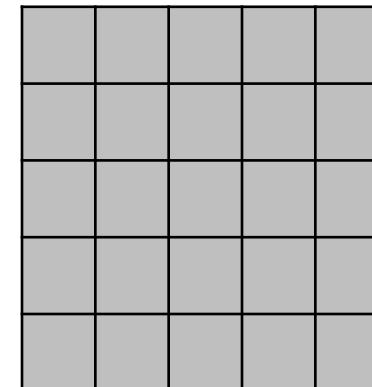
- Edge in  $x$  direction
  - **Second:** The smooth image is convolved with a derivative filter to get edges in the  $x$  direction i.e.,  $\frac{\partial f}{\partial x}$

|     |     |     |     |     |
|-----|-----|-----|-----|-----|
| 2.7 | 2.3 | 2.7 | 1.3 | 0   |
| 2.3 | 1.3 | 2.3 | 1   | 0.3 |
| 1.7 | 0.3 | 1.3 | 0.7 | 0.7 |
| 1.3 | 0.3 | 0.7 | 0.7 | 1   |
| 1   | 0.7 | 0   | 1.3 | 1   |

Smooth Image

Filter (1,3)

|    |   |   |
|----|---|---|
| -1 | 0 | 1 |
|----|---|---|



Filter/Kernel:  $H$

Edge Image:  $\frac{\partial f}{\partial x}$

# Prewitt Edge Detector - Example

- Edge in  $x$  direction
  - **Second:** The smooth image is convolved with a derivative filter to get edges in the  $x$  direction i.e.,  $\frac{\partial f}{\partial x}$

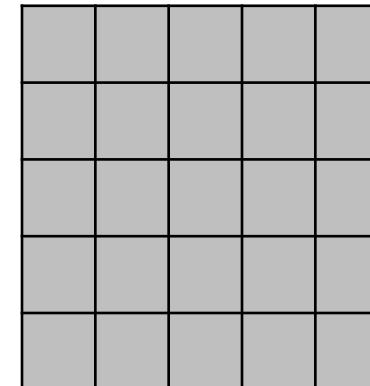
|     |     |     |     |     |     |
|-----|-----|-----|-----|-----|-----|
| 3.7 | 2.7 | 2.3 | 2.7 | 1.3 | 0   |
| 2.3 | 2.3 | 1.3 | 2.3 | 1   | 0.3 |
| 1.7 | 1.7 | 0.3 | 1.3 | 0.7 | 0.7 |
| 1.3 | 1.3 | 0.3 | 0.7 | 0.7 | 1   |
| 1   | 1   | 0.7 | 0   | 1.3 | 1   |

3.7  
2.3  
1.7  
1.3  
1

Filter (1,3)

|    |   |   |
|----|---|---|
| -1 | 0 | 1 |
|----|---|---|

0  
0.3  
0.7  
1  
1



Smooth Image

Filter/Kernel:  $H$

Edge Image:  $\frac{\partial f}{\partial x}$

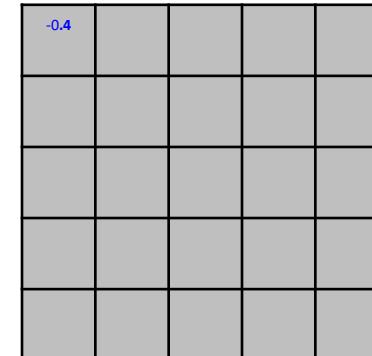
# Prewitt Edge Detector - Example

- Edge in  $x$  direction
  - **Second:** The smooth image is convolved with a derivative filter to get edges in the  $x$  direction i.e.,  $\frac{\partial f}{\partial x}$

|            |     |     |     |     |          |
|------------|-----|-----|-----|-----|----------|
| <b>2.7</b> | 2.7 | 2.3 | 2.7 | 1.3 | 0        |
| -1         | 0   | 1   |     |     |          |
| <b>2.3</b> | 2.3 | 1.3 | 2.3 | 1   | 0.3      |
| <b>1.7</b> | 1.7 | 0.3 | 1.3 | 0.7 | 0.7      |
| <b>1.3</b> | 1.3 | 0.3 | 0.7 | 0.7 | 1        |
| <b>1</b>   | 1   | 0.7 | 0   | 1.3 | <b>1</b> |

Filter (1,3)

0  
0.3  
0.7  
1  
1



Smooth Image

Filter/Kernel:  $H$

Edge Image:  $\frac{\partial f}{\partial x}$

# Prewitt Edge Detector - Example

- Edge in  $x$  direction
  - **Second:** The smooth image is convolved with a derivative filter to get edges in the  $x$  direction i.e.,  $\frac{\partial f}{\partial x}$

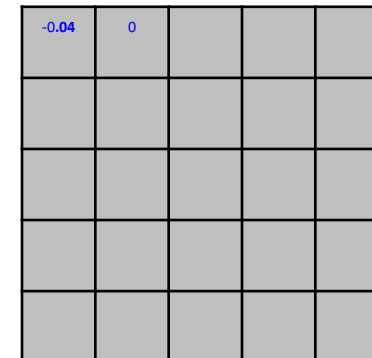
|     |           |          |          |     |     |
|-----|-----------|----------|----------|-----|-----|
| 2.7 | 2.7<br>-1 | 2.3<br>0 | 2.7<br>1 | 1.3 | 0   |
| 2.3 | 2.3       | 1.3      | 2.3      | 1   | 0.3 |
| 1.7 | 1.7       | 0.3      | 1.3      | 0.7 | 0.7 |
| 1.3 | 1.3       | 0.3      | 0.7      | 0.7 | 1   |
| 1   | 1         | 0.7      | 0        | 1.3 | 1   |

Smooth Image

Filter (1,3)

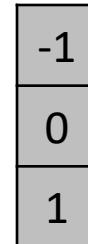
|    |   |   |
|----|---|---|
| -1 | 0 | 1 |
|----|---|---|

Filter/Kernel:  $H$



Edge Image:  $\frac{\partial f}{\partial x}$

# Prewitt Edge Detector

- Edges in  $y$  direction
  - First: Original image is convolved with an average filter such as:  get a new smooth image
  - Second: The smooth image is convolved with a derivative filter such as central difference to get edges in the  $y$  direction, i.e.,  $\frac{\partial f}{\partial y}$  

# Prewitt Edge Detector

- Edges in  $x$  direction

$$( f(x,y) * \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} ) * \begin{bmatrix} -1 & 0 & 1 \end{bmatrix} = f(x,y) * \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} = \frac{\partial f}{\partial x}$$

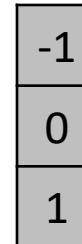
- Edges in  $y$  direction

$$( f(x,y) * \begin{bmatrix} 1 & 1 & 1 \end{bmatrix} ) * \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix} = f(x,y) * \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} = \frac{\partial f}{\partial y}$$

# Sobel Edge Detector

- Edge in  $x$  direction
  - First: Original image is convolved with a Gaussian filter such as: to get a new smooth image
  - Second: The smooth image is convolved with a derivative filter such as central difference  to get edges in the  $x$  direction i.e.,  $\frac{\partial f}{\partial x}$

# Sobel Edge Detector

- Edges in  $y$  direction
  - First: Original image is convolved with a Gaussian filter such as:  to get a new smooth image
  - Second: The smooth image is convolved with a derivative filter such as central difference to get edges in the  $y$  direction, i.e.,  $\frac{\partial f}{\partial y}$  

# Sobel Edge Detector

- Edges in  $x$  direction

$$( f(x,y) * \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} ) * \begin{bmatrix} -1 & 0 & 1 \end{bmatrix} = f(x,y) * \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} = \frac{\partial f}{\partial x}$$

- Edges in  $y$  direction

$$( f(x,y) * \begin{bmatrix} 1 & 2 & 1 \end{bmatrix} ) * \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix} = f(x,y) * \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} = \frac{\partial f}{\partial y}$$

# Edge Detection Operators

- Sobel edge detector

$$\frac{\partial f}{\partial x}$$

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

$$\frac{\partial f}{\partial y}$$

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

- Prewitt edge detector

$$\frac{\partial f}{\partial x}$$

|    |   |   |
|----|---|---|
| -1 | 0 | 1 |
| -1 | 0 | 1 |
| -1 | 0 | 1 |

$$\frac{\partial f}{\partial y}$$

|    |    |    |
|----|----|----|
| -1 | -1 | -1 |
| 0  | 0  | 0  |
| 1  | 1  | 1  |

- Roberts edge detector

|   |    |
|---|----|
| 1 | 0  |
| 0 | -1 |

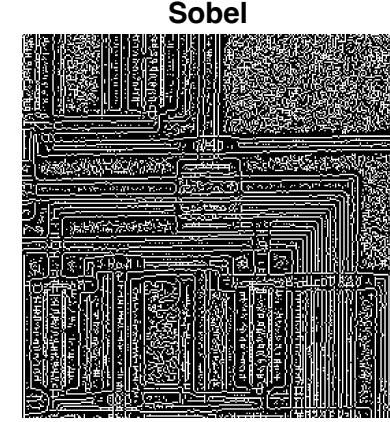
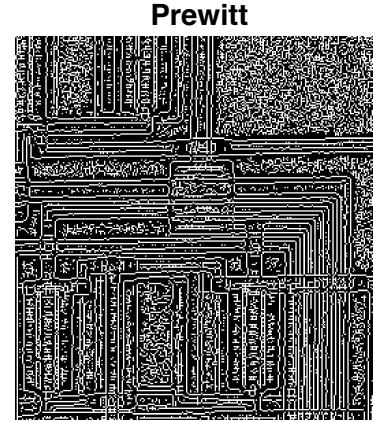
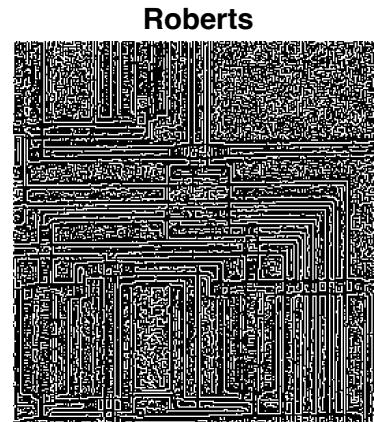
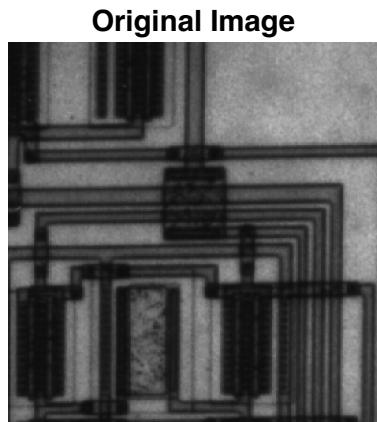
|    |   |
|----|---|
| 0  | 1 |
| -1 | 0 |

# Edge Detection (Without Thresholding)

- Generate binary image

$$B = \begin{cases} 0, & |\nabla f| = 0 \\ 1, & |\nabla f| > 0 \end{cases}$$

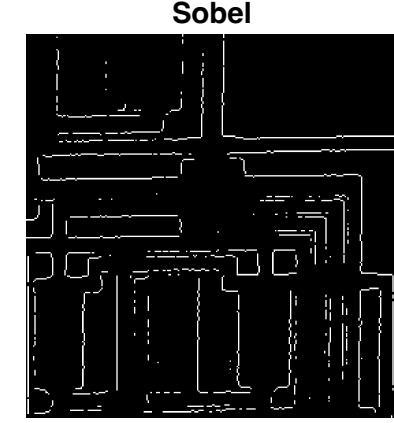
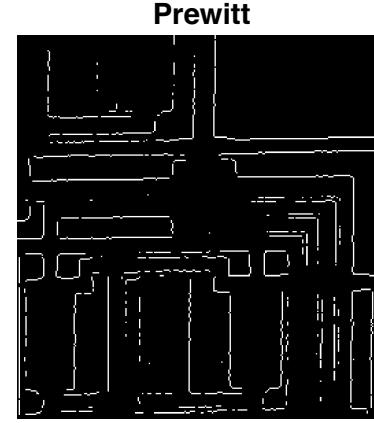
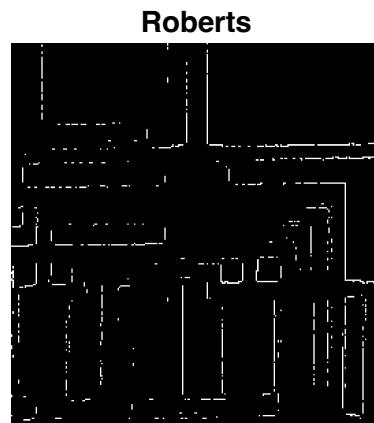
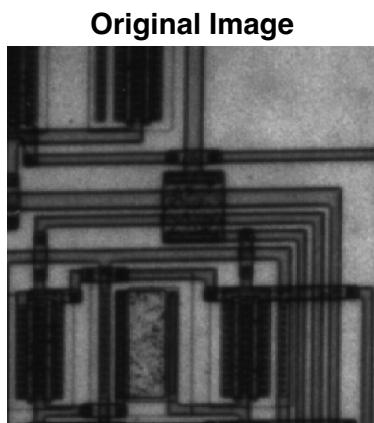
- There are many edges but how to pick the selective ones?



# Edge Detection (With Thresholding)

- **Thresholding:** Pixels with gradient magnitudes greater or equal to a threshold value are selected as edges
  - For example, threshold could be set to 33% of the maximum magnitude

$$B = \begin{cases} 0, & ||\nabla f|| < \text{threshold} \\ 1, & ||\nabla f|| \geq \text{threshold} \end{cases}$$



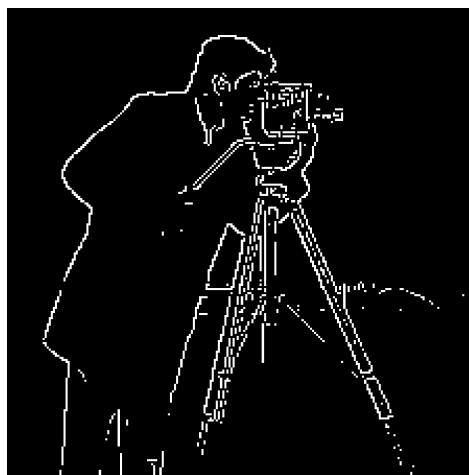
# Edge Detection (With Thresholding)

- **Thresholding:** Pixels with gradient magnitudes greater or equal to a threshold value are selected as edges

Original Image



Roberts



Prewitt



Sobel

