

Image and Video Processing

(EBU723U)

Introduction

Dr Miles Hansard

Dr Qianni Zhang

miles.hansard@qmul.ac.uk

Digital images ...

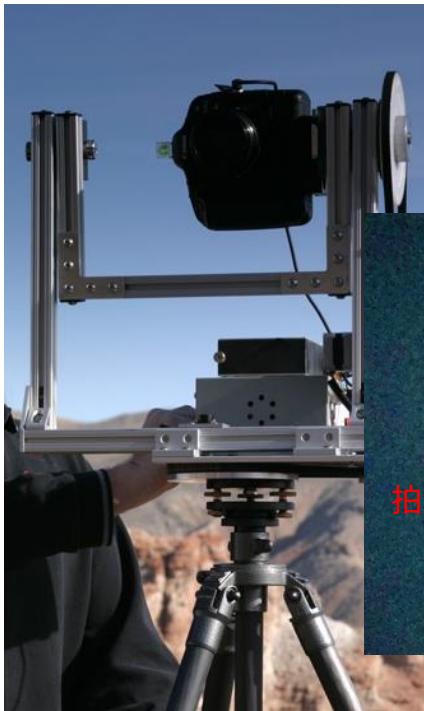


Comparison



- www.xrez.com

Panoramic imaging

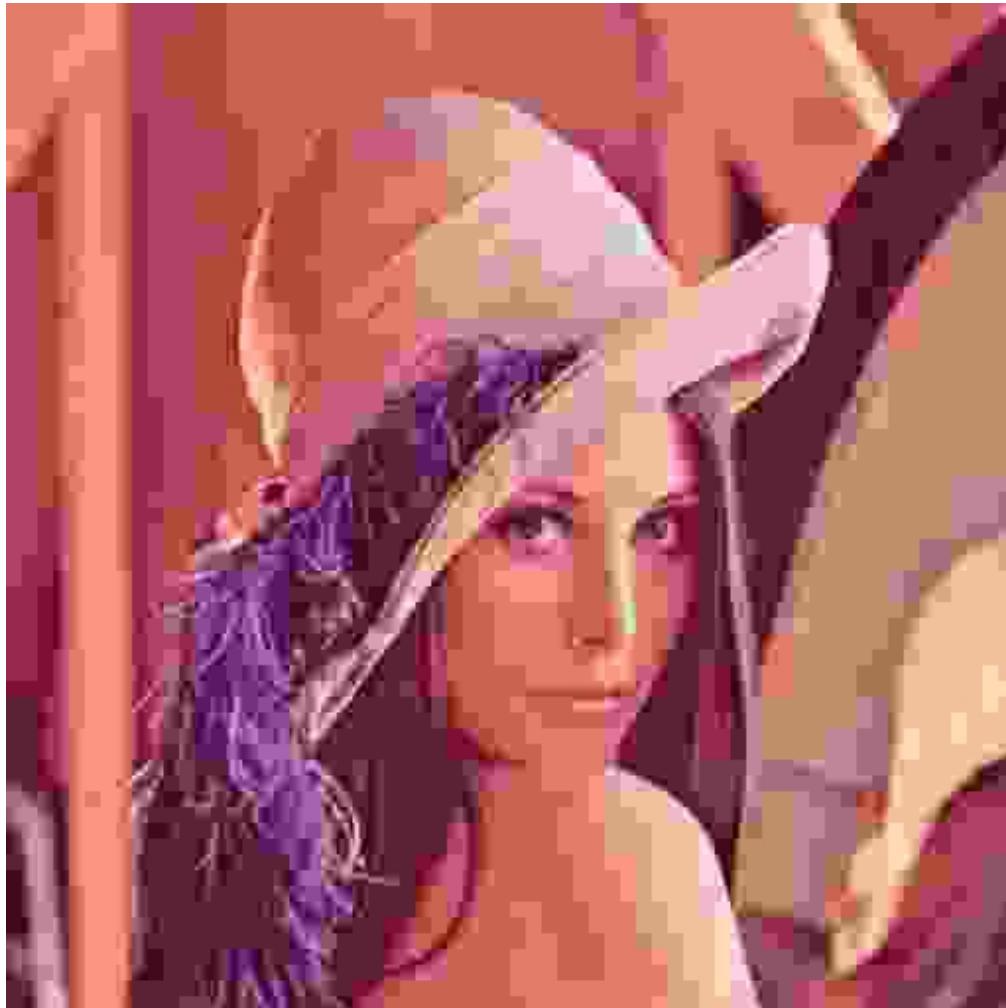


- www.xrez.com

Image file-sizes

- i-Phone 5: 3264×2448 camera
- $3264 \times 2448 \times 3 \times 8$ bits ≈ 24 Mb
- Size of the uncompressed RAW file
- Xrez Panorama:
- $30,000 \times 70,000 \times 3 \times 8$ bits ≈ 6 GB

What signal processing is there behind this image?



this version: 5K, original version: 450K

What signal processing is there behind this image?



JPEG 99% compression

Today's agenda

- Introduction to EBU723U
 - Aims
 - Objectives
 - Course organisation
 - Resources
- Examples

Today's agenda

- Introduction to EBU723U
 - Aims
 - Objectives
 - Course organisation
 - Resources
- Examples

Aims

- To provide an understanding of the most relevant **image and video processing tools**
- To discuss important topics related to the **creation and manipulation** of digital images
- To cover the fundamentals necessary to **design and develop** a wide range of imaging solutions
- To study basic **mathematical models** used to analyze still images

Objectives (1/2)

- To understand fundamental tasks involved in **creating and processing image and video information**
- To become familiar with the most important techniques to perform computer assisted **image interpretation**
- To become familiar with basic filters and **filter design** technologies for image enhancement, denoising, deblurring, etc.

Objectives (2/2)

- To have an appreciation of the demands of **imaging solutions** and challenges involved in their implementation
- To be able to describe the most relevant **compression** techniques and design requirements of image and video coding

Course structure

- Lectures
 - Four weeks (Hansard x 2 + Zhang x 2)
- Labs
 - To be announced
- In-class tests
 - In **second** and **fourth** teaching weeks

Assessment

- Final exam:
 - one written paper
 - Past paper will be put on QMPlus
- Coursework:
 - individual coursework
 - Two in-class tests

Resources

- Books
 - Digital Image Processing
R.C. Gonzales, R.E. Woods, Prentice Hall, ISBN 0130946508
 - Handbook of Image and Video Processing,
A Bovik, Academic Press, ISBN 0-12-119790-5
 - Video Coding: An Introduction to Standard Codecs
M. Ghanbari, IEE Publishing
 - JPEG2000: image compression fundamentals
D. Taubman, M. Marcellin, Kluwer
- Web

QMPlus – EBU723U

- News
- Module Information
- Lectures
- Coursework
- Lab sheets
- Exam
- Resources
- FAQs
- Message Board

Communication and advice

- EBU723U Messageboard is the preferred medium
- Problems with coursework, labs, etc.
 1. Contact the Assistant Tutor during lab hours
 2. Use EBU723U Message Board

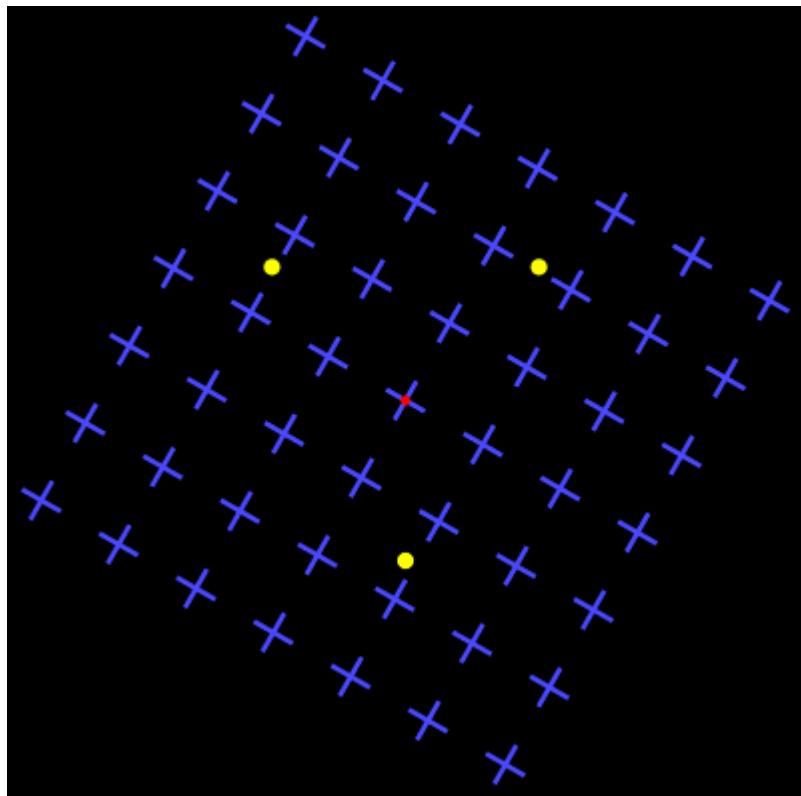
About me

- Mres/PhD in computer vision from UCL
- Worked in vision/robotics at INRIA in France
- Now in QMUL Vision Group
- www.eecs.qmul.ac.uk/~milesh/
- miles.hansard@qmul.ac.uk
- My interests:
 - 3D reconstruction
 - Multiview geometry and statistics
 - Biological vision

What you see is not what you get

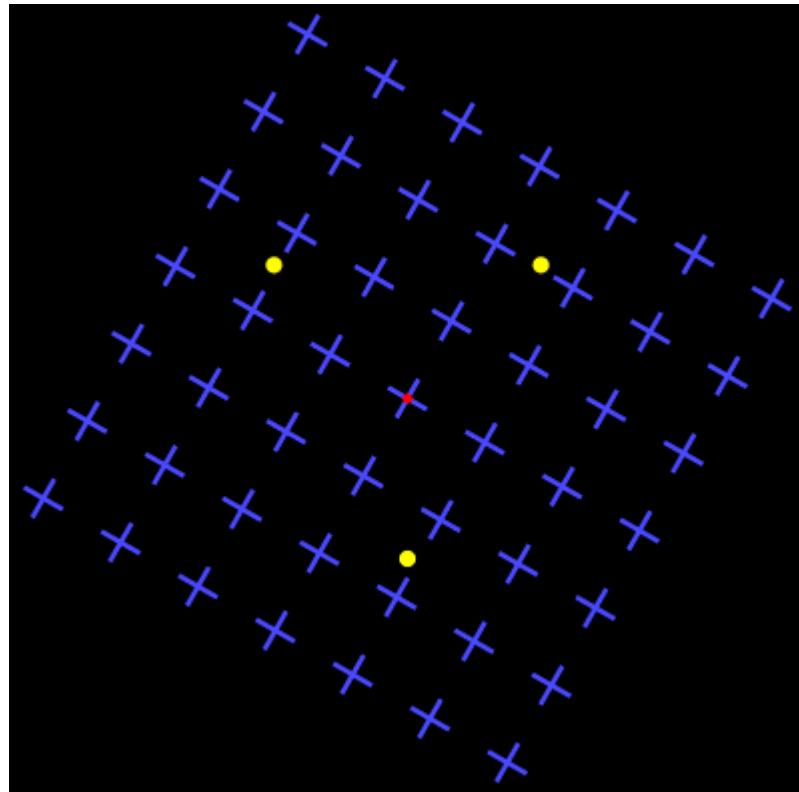
- Motion induced blindness
- Audio signal processing is based on an understanding of the ear
- Image signal processing is based on an understanding of the eye.
- Motion Induced Blindness

Experiment: MIB



- Watch the movie and focus on the central red dot.
- What happens to the yellow dots?

Experiment: Motion-induced blindness



Change blindness

- From O'Regan et al.



And again



Today's agenda

- Introduction to EBU723U
 - Aims
 - Objectives
 - Course organisation
 - Resources
- Examples

Image & video processing: applications

- Applications
 - Medical research
 - Image-based diagnosis
 - Environmental control
 - Forensics
 - Astronomy
 - Quality control
 - Biometrics (face/iris recognition, fingerprint...)
 - Handwriting recognition
 - Surveillance
 -

Compression



Original

450 K

JPEG Compression



Encoded Version

80% Quality

85 K

JPEG Compression

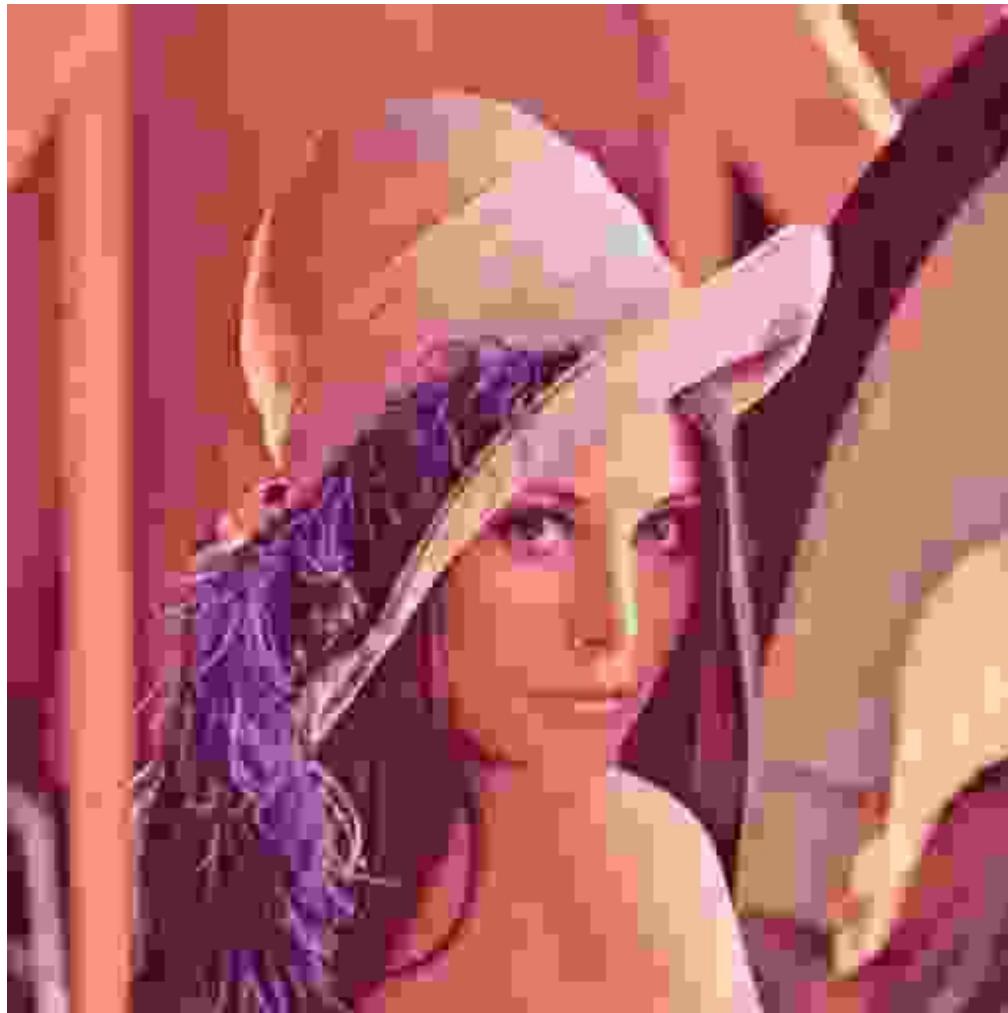


More compression...

10% Quality

15 K

JPEG Compression



More compression...

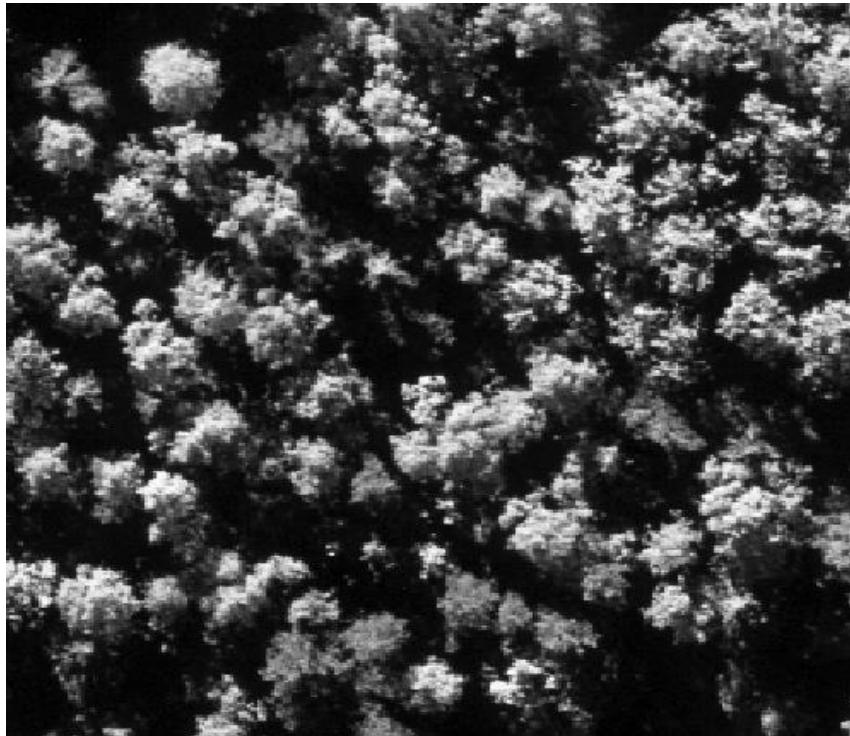
5% Quality

5 K

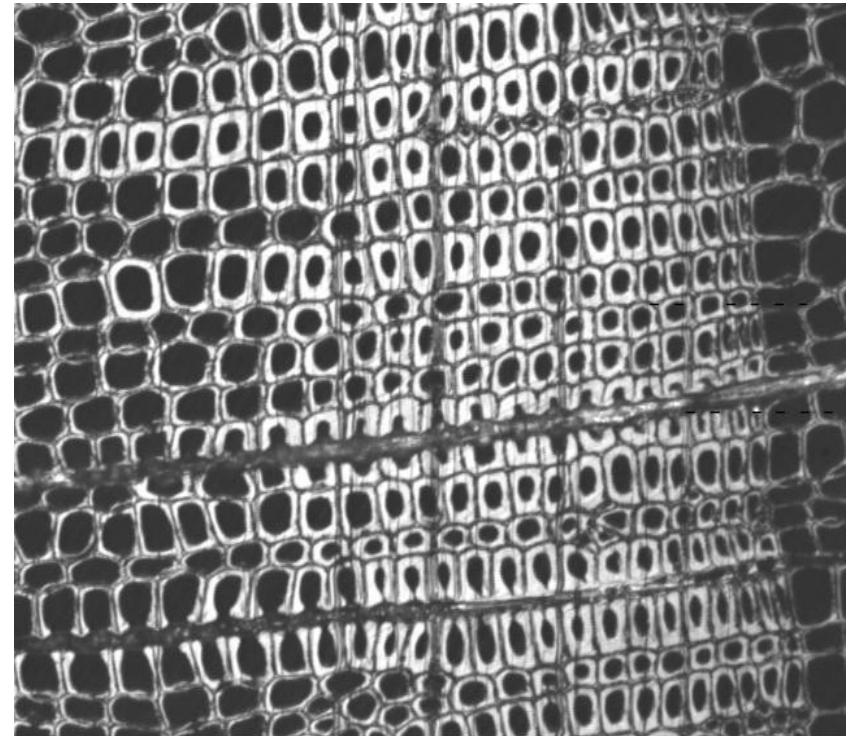
Scientific photography and videography

- Imaging systems
 - To gather quantitative information about physical systems and processes
 - individual cells
 - galaxies
 - Applications
 - microscopy
 - biomedical imaging
 - remote sensing
 - astronomy

Environmental control

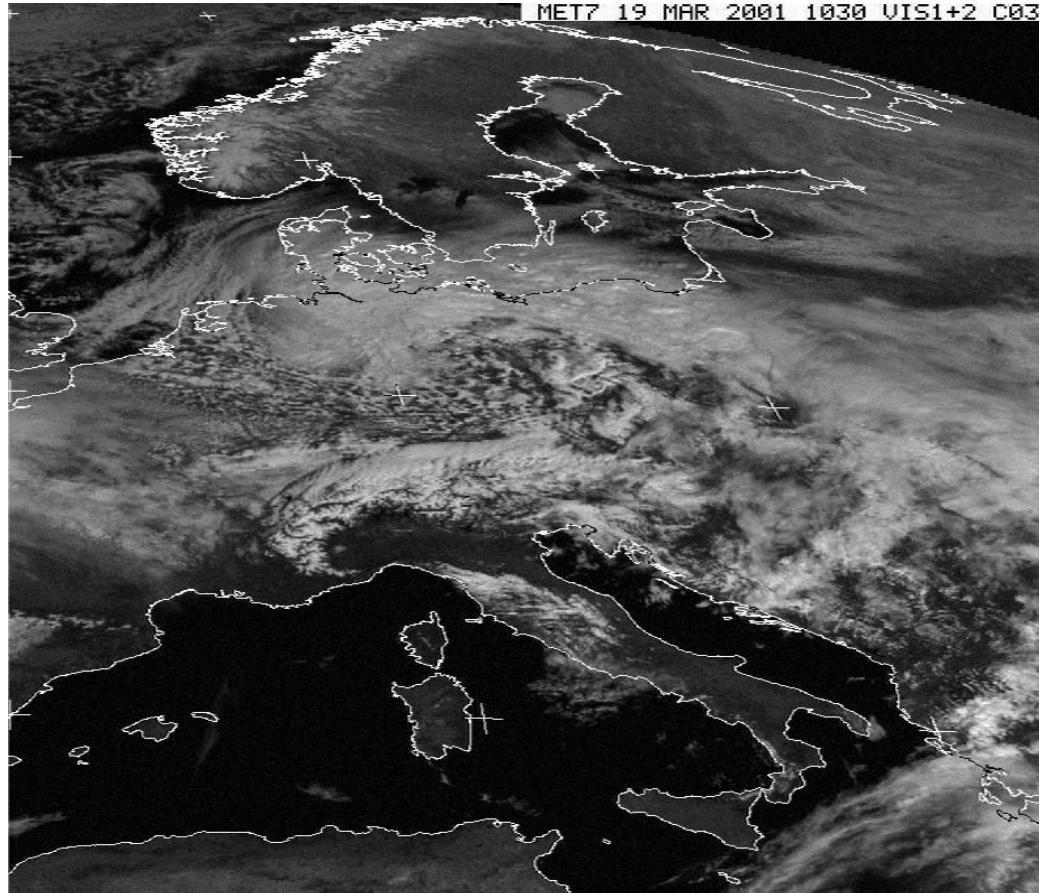


Forest



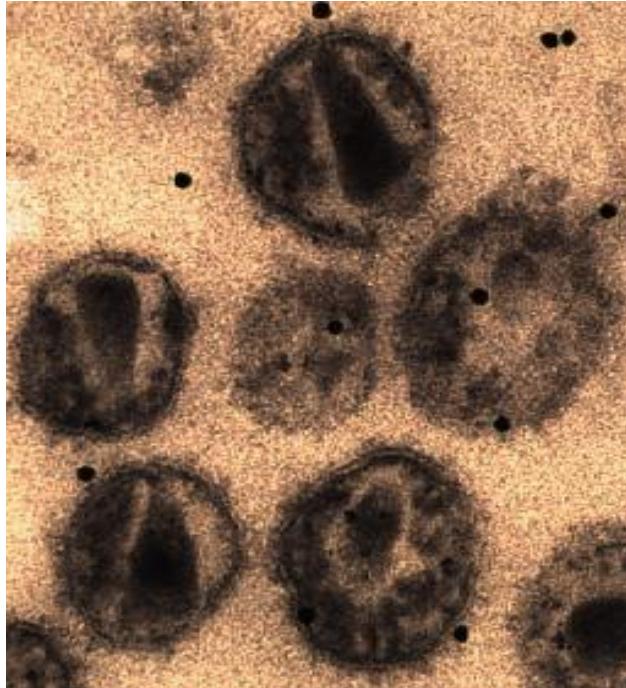
Wood Quality

Satellite imaging

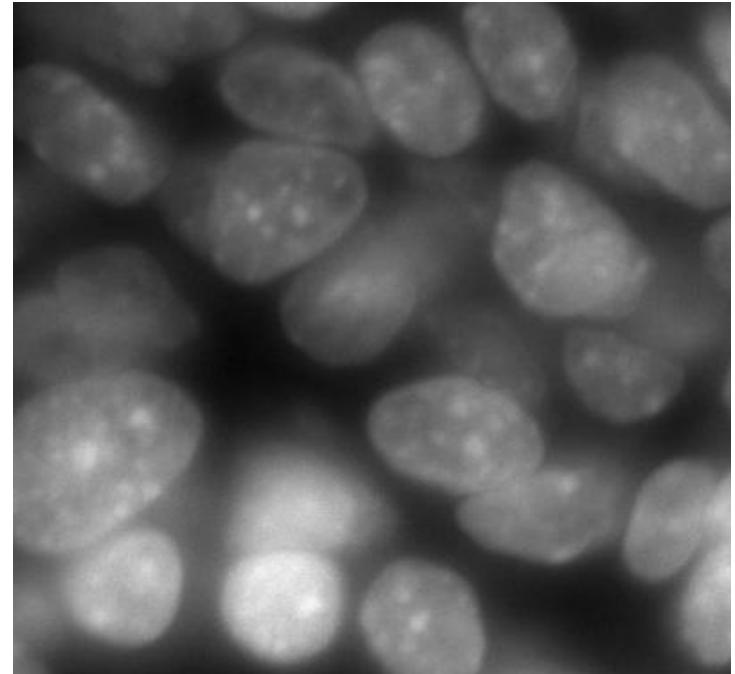


Weather Forecast

Medical research



AIDS-virus particles
(Electron microscopy)



Cancer tumor
(Fluorescence microscopy)

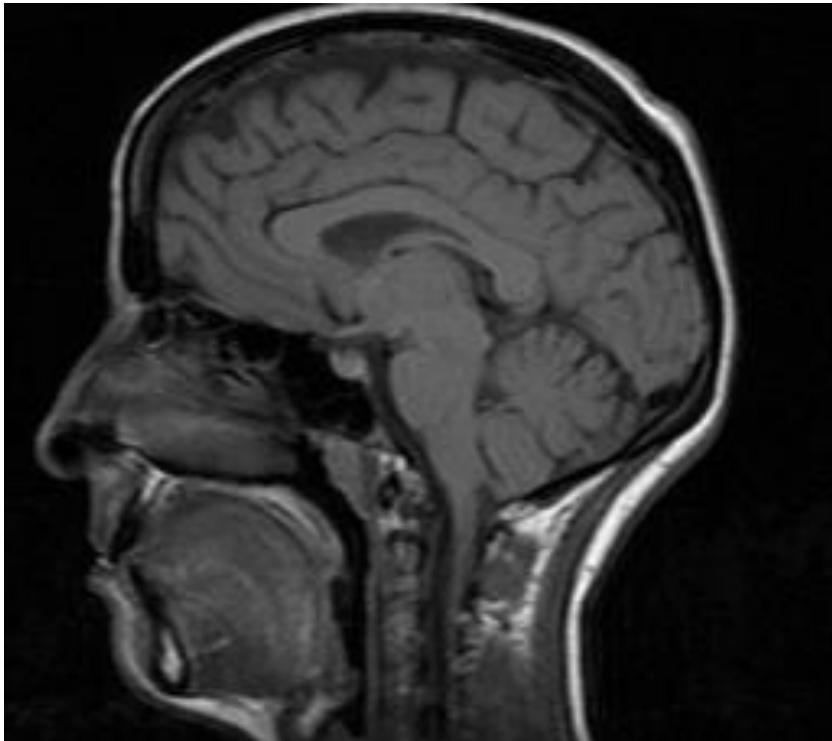
Advanced image and video processing

- Computational methods to produce new/enhanced images

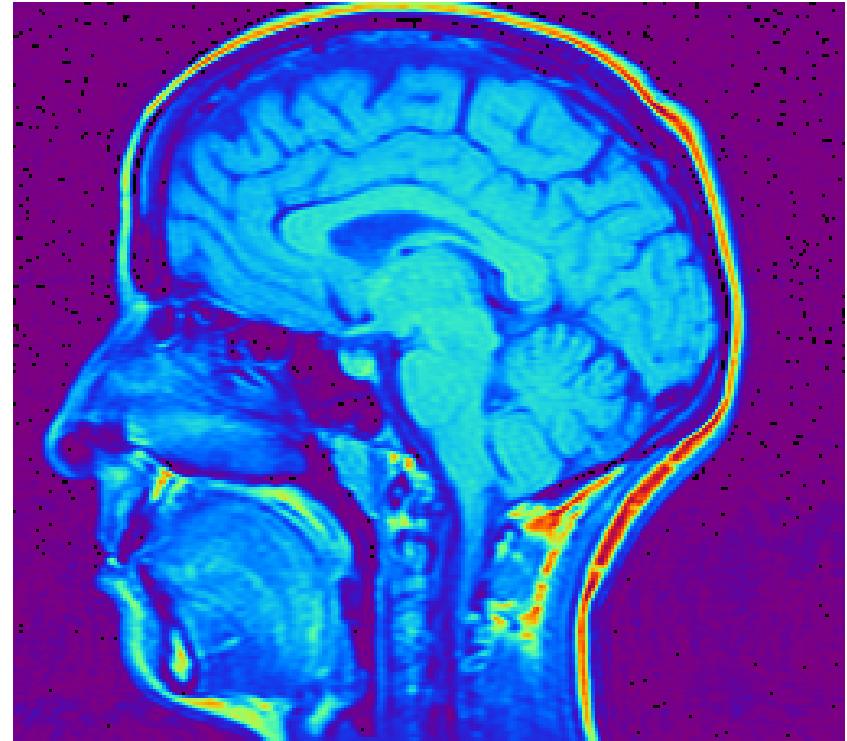
prior belief: e.g. We assume the background is yellow.

- use of image priors for enhancement
- image matting 天气预报的背景–抠图术
- image filling
- view interpolation

False colour images



original image



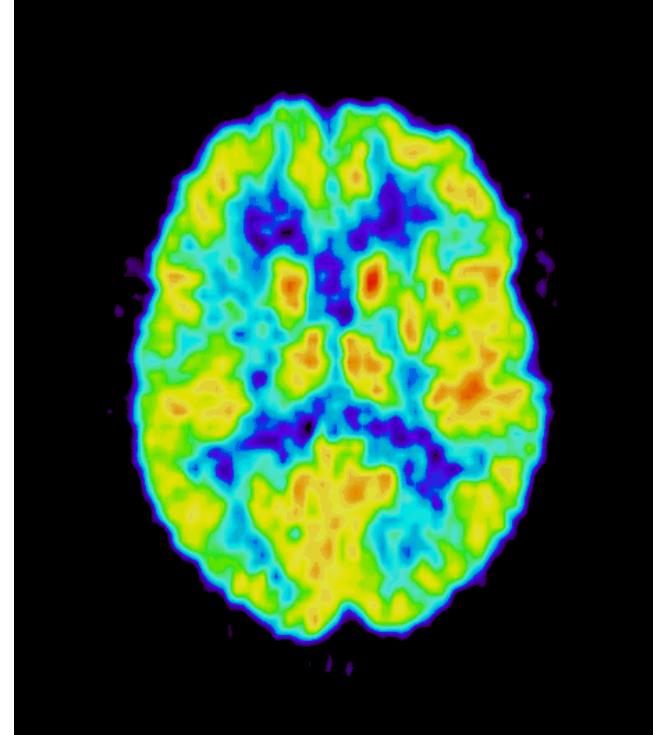
false colour image

Image-based diagnosis



X-ray image

(high resolution)

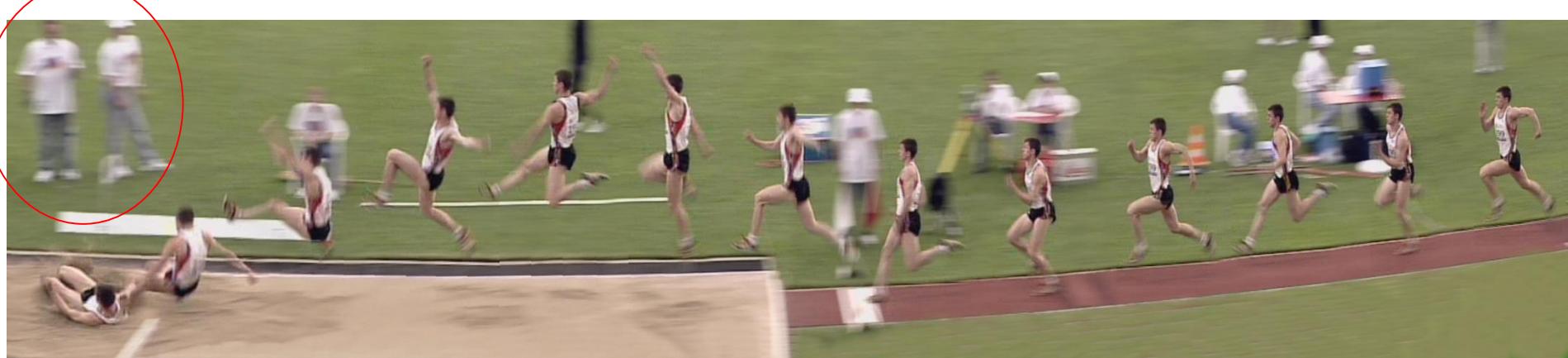


Positron Emission
Tomography (PET)

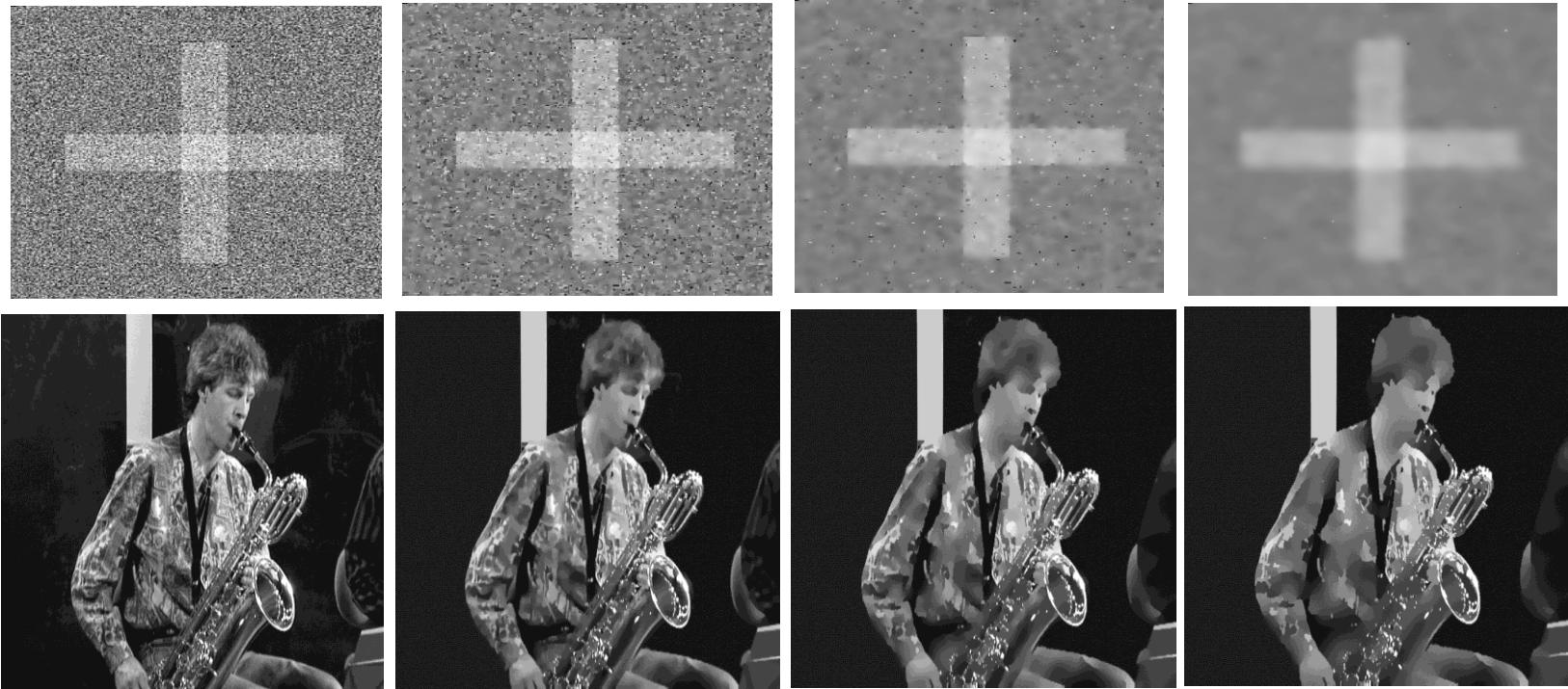
(Low resolution)

Video Production

artefact



Enhancement - denoising



low pass filter removes high frequency components (noise)

Spatial frequency

Noise & denoising



Denoising

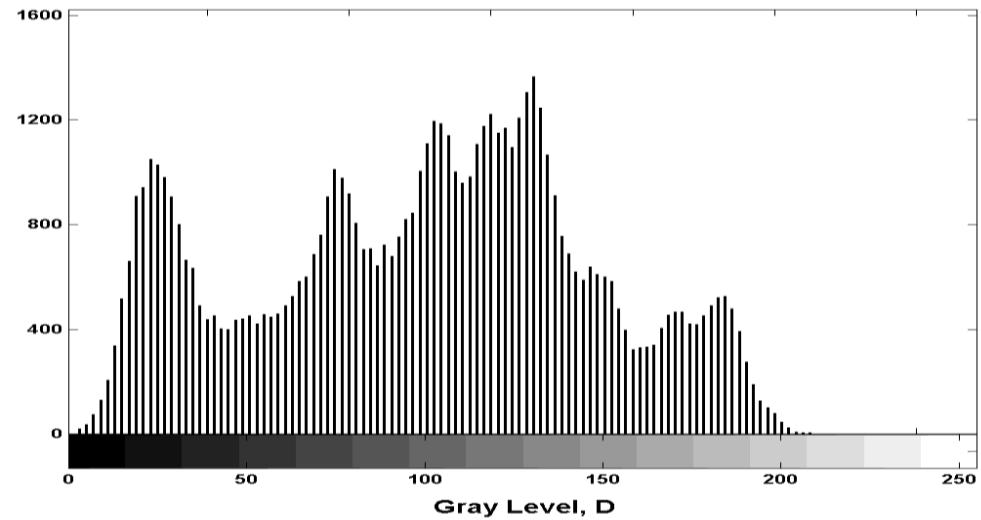
Histograms

H



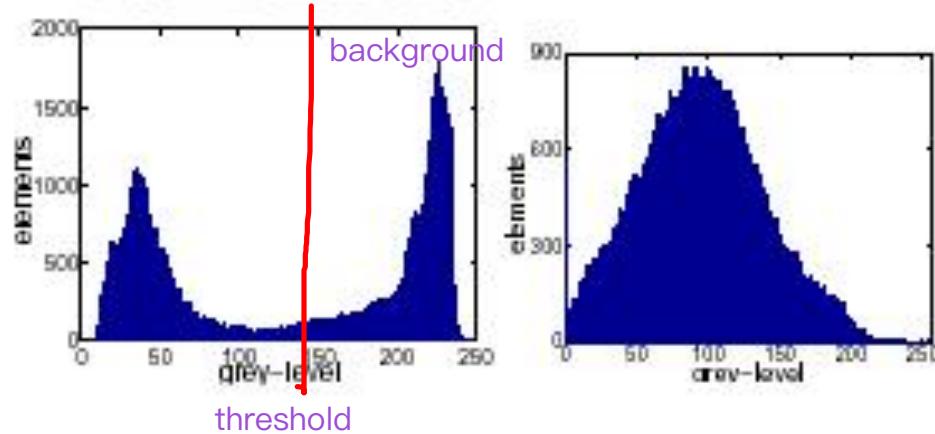
$\overline{R}^{H,W}$

No. Pixels, $H(D)$



\overline{R}^{255}

Segmentation



Boundary Issues

- Can you draw a line around the ‘edge’ of this galaxy?



Exploiting photo and video collections

- Intelligent browsing
 - Browsing and retrieving images from large collections.
 - Content based image retrieval - images can be grouped due to colour, texture, shape etc.
 - Metadata based image retrieval – images are grouped based on descriptions of the image.
 - Semantics = the study of meaning.
 - Semantic image analysis techniques aim to automatically detect high level semantic contents of images for annotation.

e.g. EXIF
–device
–time
–date
–gps
–focal length
–...

Metadata Based Image Retrieval

- Search for ‘moon corona’



Content Based Image Retrieval

- Drag-and-dropped an image of the London tube map into Google.



tubemap.gif (X) camera icon search icon
About 28,800 results (0.61 seconds)



Image size:
1256 × 836

Find other sizes of this image:
[All sizes](#) - [Small](#) - [Medium](#) - [Large](#)

Best guess for this image: [*london underground map*](#)

[PDF] [Tube map - Transport for London](#) search icon

www.tfl.gov.uk/assets/downloads/standard-tube-map.pdf

File Format: PDF/Adobe Acrobat - [Quick View](#)

London City Airport. Woolwich Arsenal Blackfriars

Underground station closed until late 2011 ...

Underground, Canary Wharf DLR and Heron Quays DLR ...



[**London Underground Map**](#) search icon

www.afn.org/~alplatt/tube.html - [Cached](#)

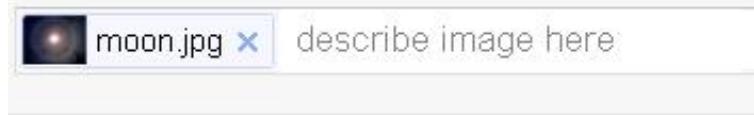
LONDON. 2 images

[Visually similar images](#) - [Report images](#)



Content Based Image Retrieval

- Dragged-and-dropped an image of the moon.
- Some images make sense.
- Others do not



Tip: Try entering a descriptive word in the search!



Image size:
255 × 201

No other sizes of this image found.

[Visually similar images](#) - Report images



Computer Vision

- Computer vision: machines that can see
- Hardware + image processing + intelligence
- Usually involves multiple images/videos and viewpoints
- Attempts to make an ‘object level’ understanding of the 3D world
- This is *not* a computer vision course, but image processing is *part* of computer vision

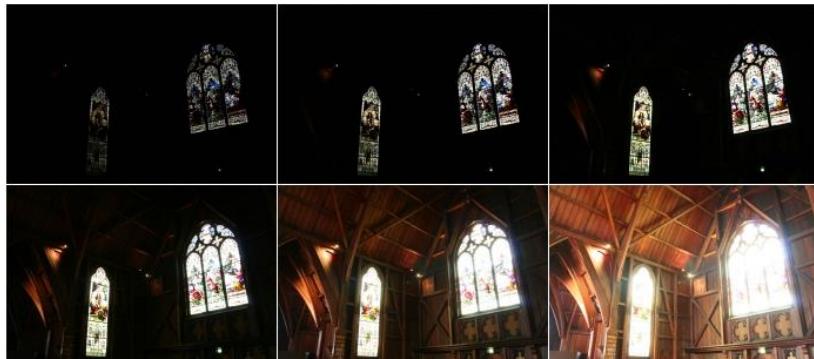
Exploiting photo and video collections

- Image collections to produce new/enhanced visual media
- Examples
 - intelligent compositing
 - Combining visual elements from different visual sources (blue screening)
 - object recognition/ classification based image manipulation
 - organizing and navigating large image collections

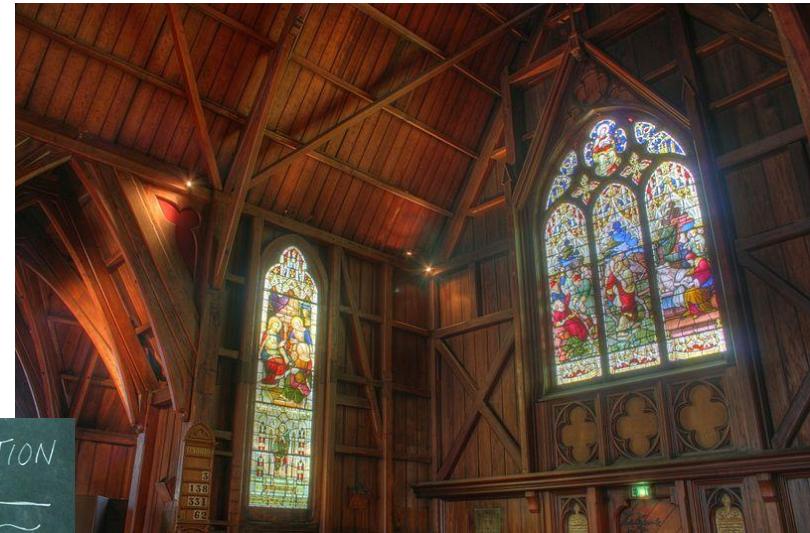
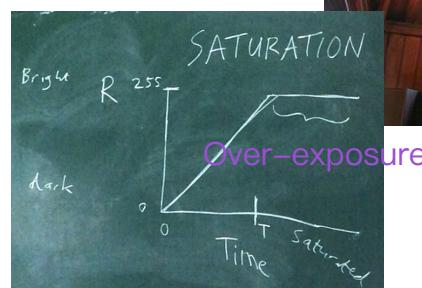
High Dynamic Range (HDR) images

Steps from black to white
> 256

- Combine multiple images from same viewpoint
- Get the best of several different exposures



Input images



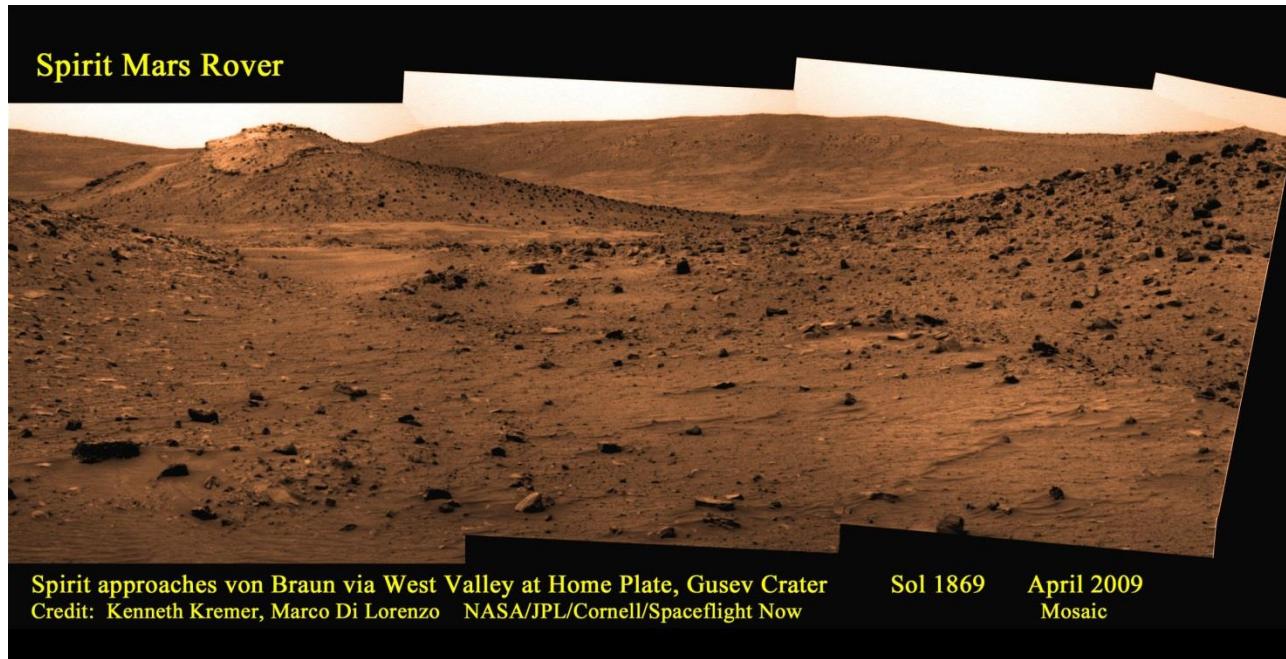
Outdoor HDR image



Problem:
Shadows might
seem false

Panoramic mosaics

- Available on recent smartphones



Example from Mars

Panoramic distortion

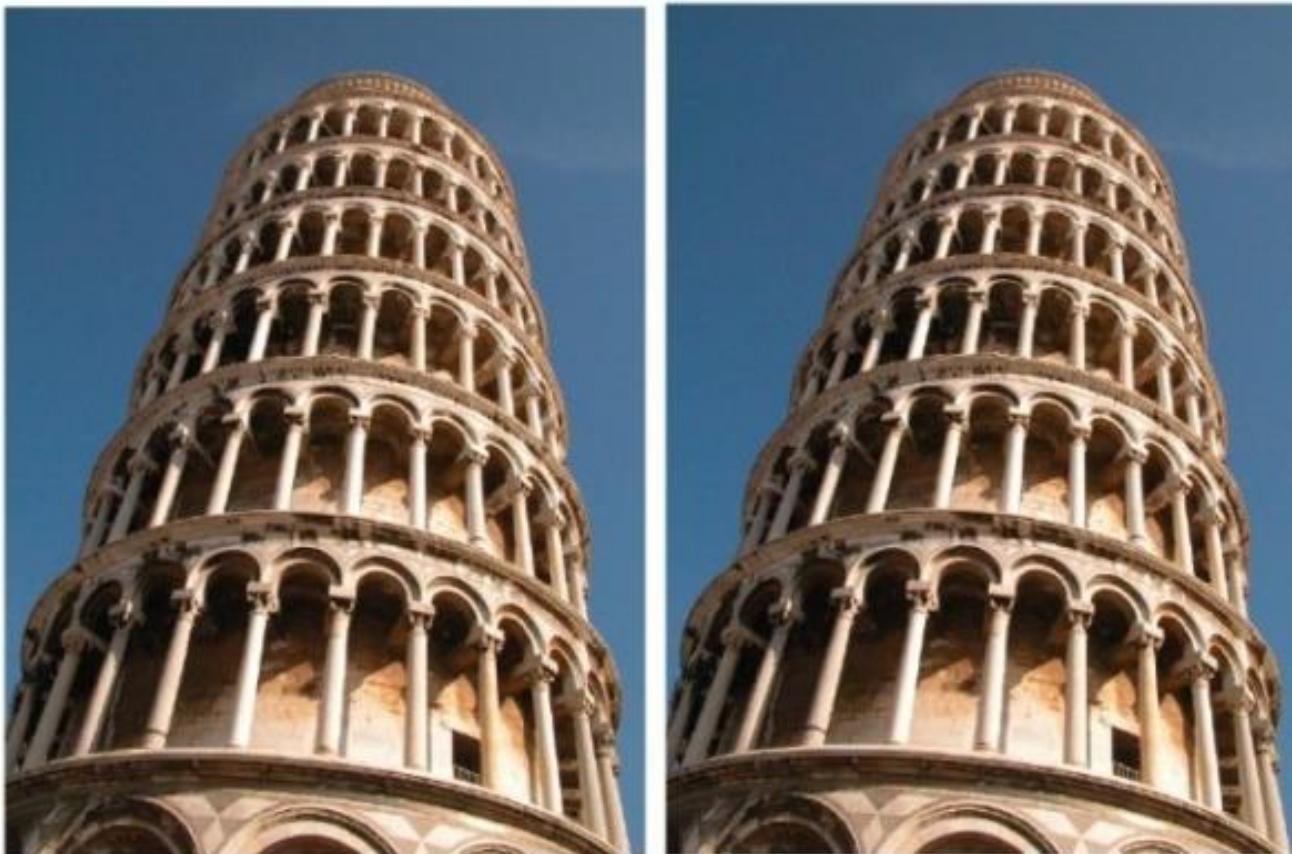


- Straight lines in the scene appear bent in the image
- There is no ‘distortion free’ way to make a full panorama

Photographic conventions

Visual illusion

- Which tower is leaning the most?



Multiple images and camera arrays

- Multiple images captured sequentially or simultaneously
 - produce new or enhanced images
 - Examples
 - Mosaicing
 - creation of collages and montages
 - refocusing, and light field rendering
 - achieve high dynamic range
 - extended depth of field

Refocusing and light field rendering

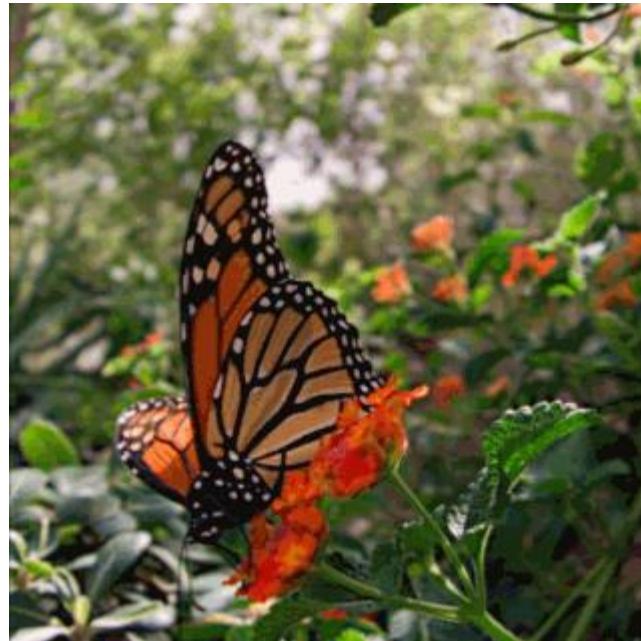


- Light field rendering is used to change the focus of an image that has been taken.

Extended depth of field



Multi-focus light-field camera



- Lytro
- www.lytro.com

3D light-field camera



- Raytrix camera
- 40,000 micro-lenses
- www.raytrix.de

Time-of-Flight + RGB systems



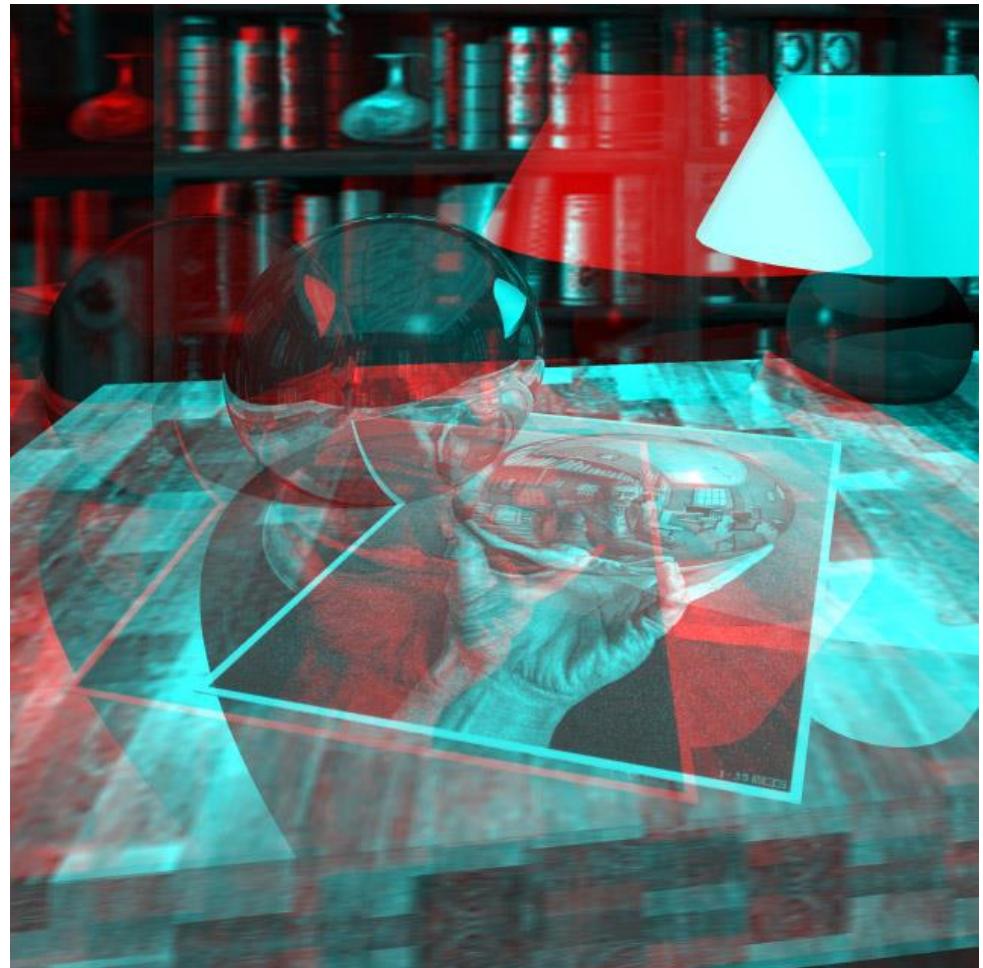
- Time-of-flight + 2xRGB cameras
- 3D from ToF, texture from RGB

Multiple images and camera arrays

- Multiple images captured sequentially or simultaneously
 - super-resolution 监控录像 放大看车牌号
 - denoising
 - multispectral imaging 非可见光
 - polarization imaging

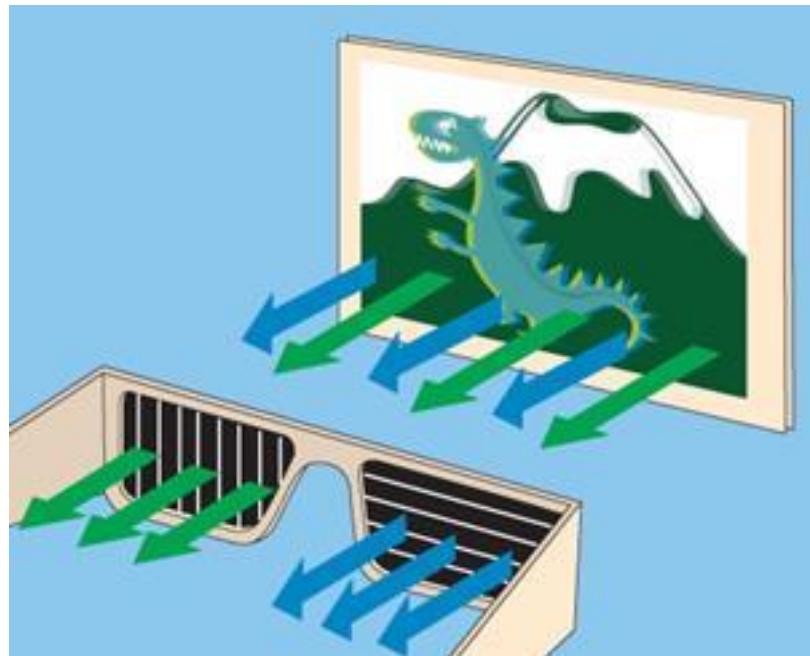
Old Style 3D

- Had to wear glasses so that one eye saw only the red image and the other only the blue.
- ‘Anaglyph’



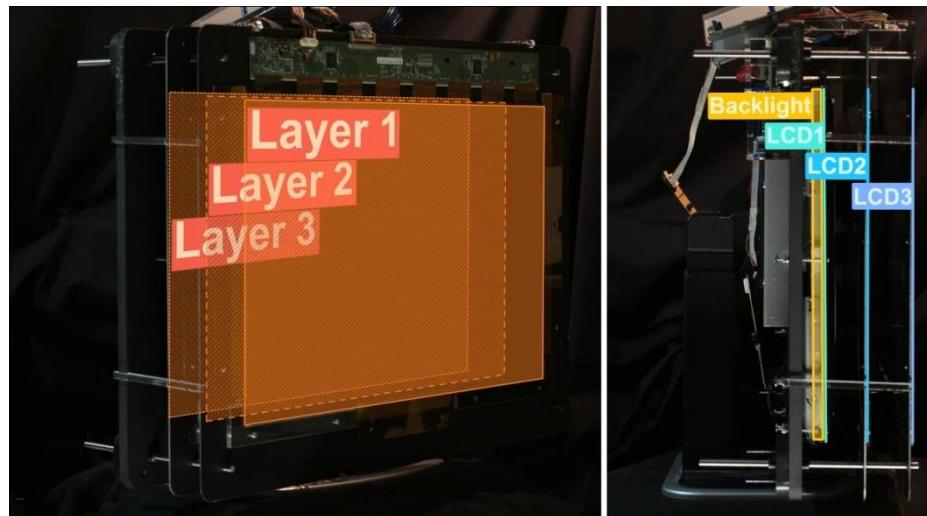
3DTV

- Use polarized light
- Or use synchronized shutter-glasses
- Still have to wear glasses



Tensor Displays

- Glasses-free multi-view prototype from MIT
- Uses directional backlight, and layered LCDs
- Sophisticated space-time image processing



New generation of VR headsets

- Will probably be released in 2015



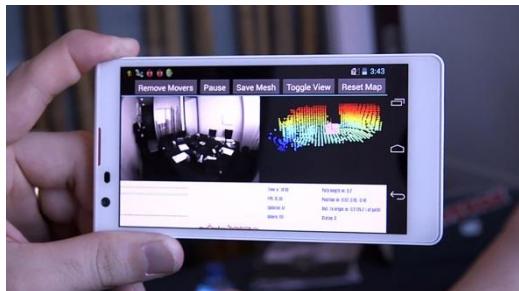
Oculus Rift



Sony Morpheus

Google Project Tango

- Aims to create Android devices (phone/tablet) that understand the **3D** world
- Uses cameras (front, back), and inertial sensors (compass, accelerometer, etc)
- Prototypes available this year



Project Tango examples



Depth map (Z colour-coded)



Full 3D construction (many views)

3D illusions

- See video...

What did we learn today?

- Introduction to EBU723U
 - Aims
 - Objectives
 - Course organisation
 - Resources
- Examples

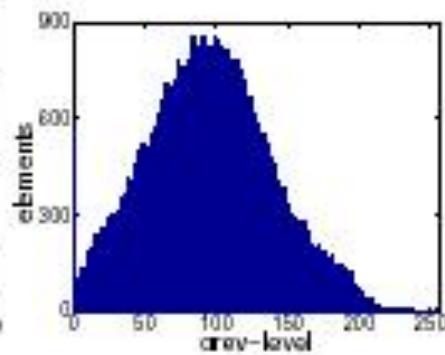
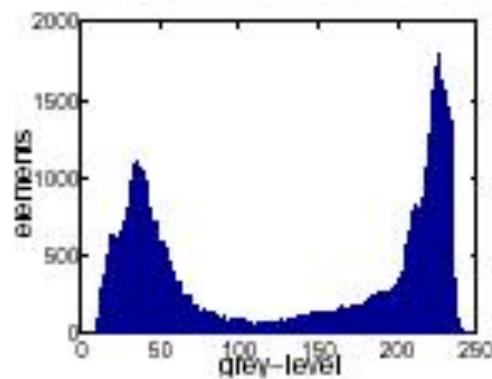
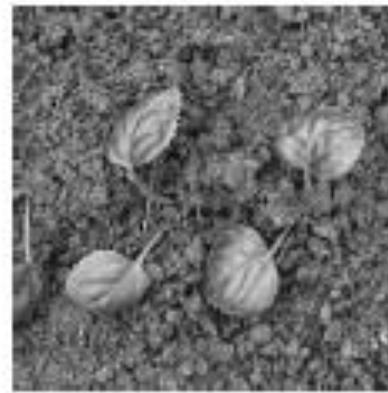
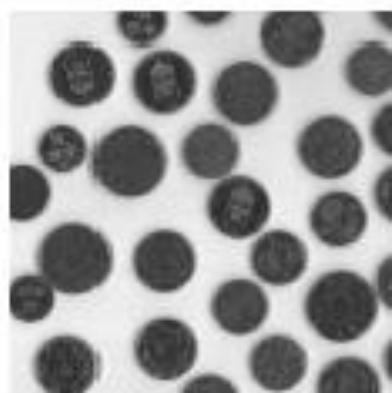
Image and Video Processing

(EBU723U)

Image Representation

Dr. Miles Hansard
miles.hansard@qmul.ac.uk

Segmentation



Today's agenda

- Digital image representation
- Sampling – pixels
- Quantization
- Sub-sampling
- Pixel interpolation

Today's agenda

- Digital image representation
- Sampling
- Quantization
- Sub-sampling
- Pixel interpolation

Image and video processing

- Relationship between image & video processing and signal processing

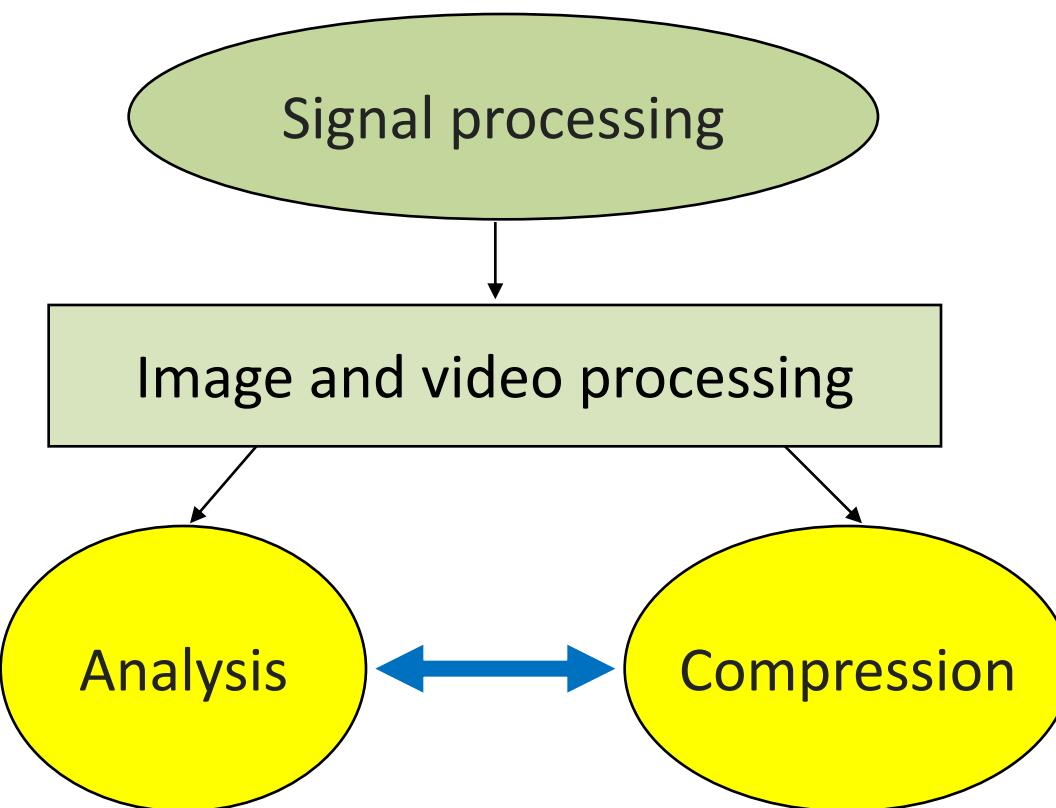
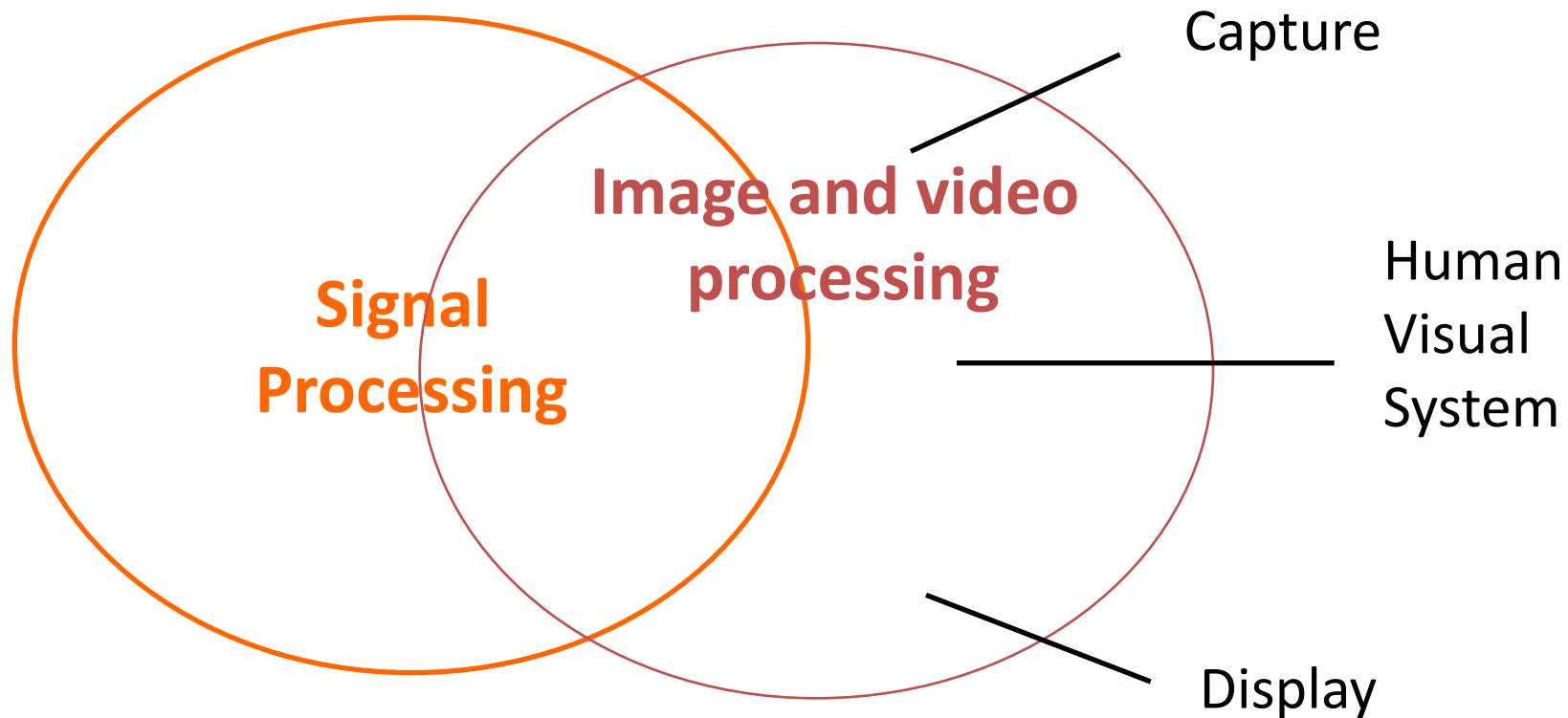


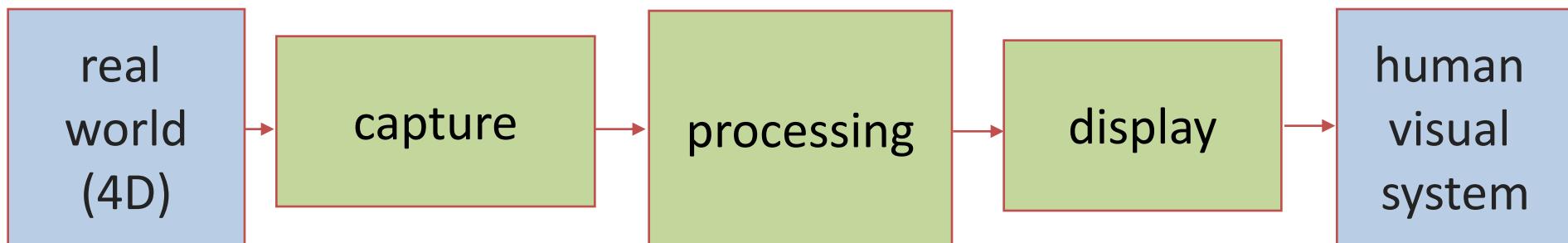
Image and video processing

- Relationship between image & video processing and signal processing

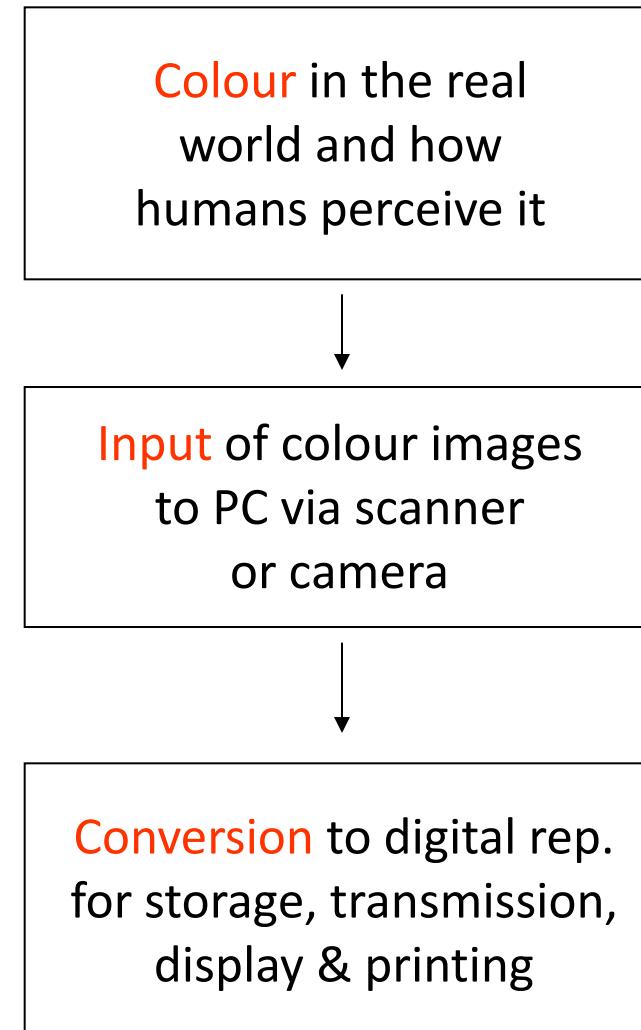


Introduction

- Complete image & video processing flow diagram

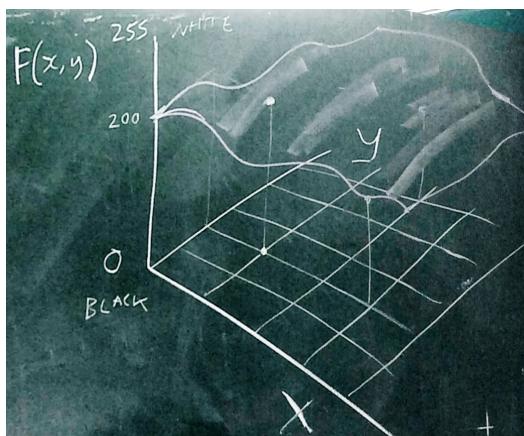


Analogue to digital

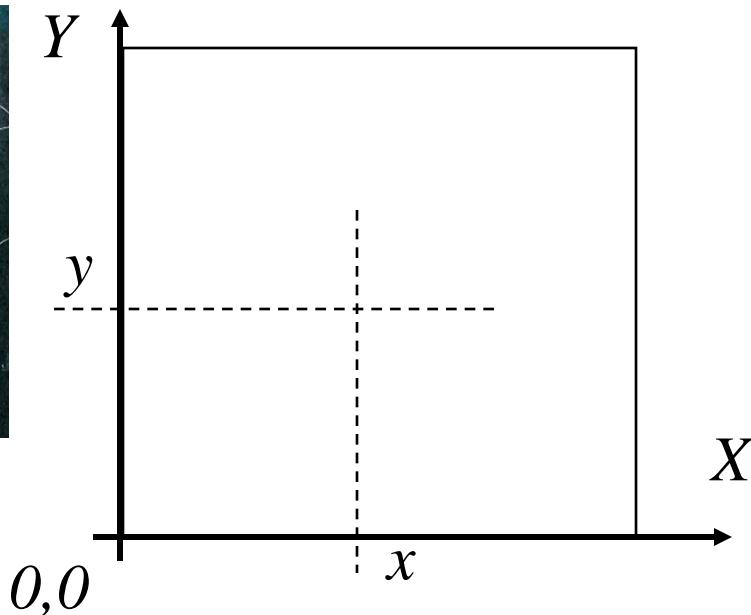


Continuous image representation

- An image can be represented
 - with a **function** $f(x, y)$ defined on a support D
 - the variables x and y represent the coordinates of an image point in the space, with the value of the function (**real number**) that defines the grey level associated to the point (x, y)



注：交点代表像素

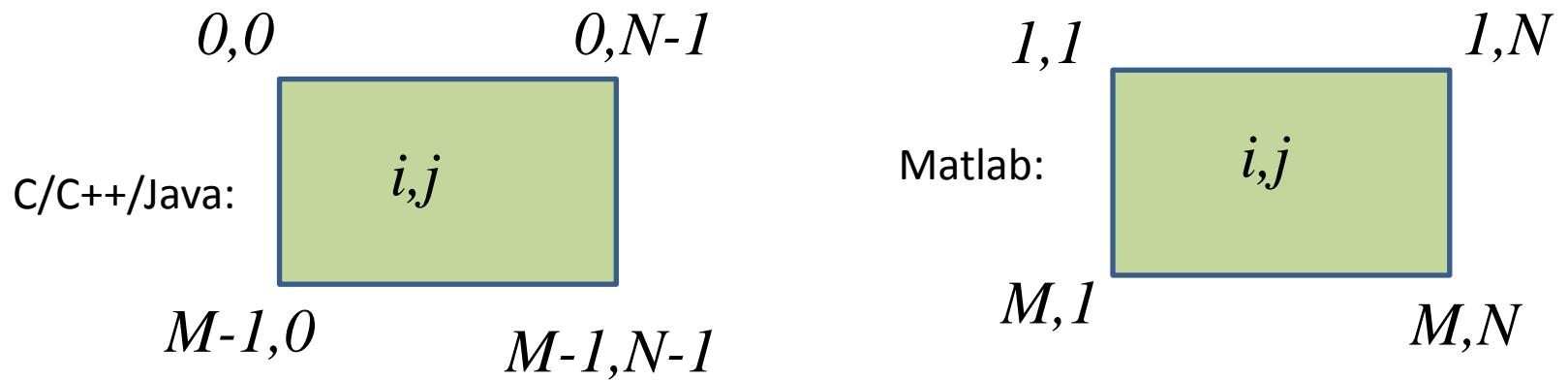
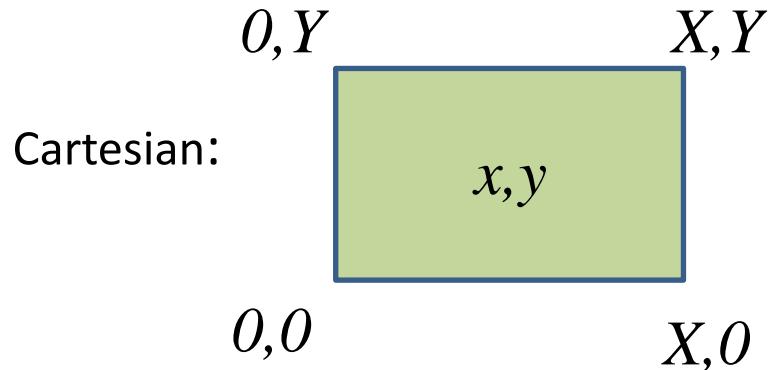


Discrete image representation

- Consider a digital image $g(i,j)$ of size $M \times N$.
- Array representation in C, C++ or Java:

$$g(i, j) = \begin{bmatrix} g(0,0) & g(0,1) & \cdots & s(0, N-2) & g(0, N-1) \\ g(1,0) & g(1,1) & \cdots & s(1, N-2) & g(1, N-1) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ g(M-2,0) & g(M-2,1) & \cdots & g(M-2, N-2) & g(M-2, N-1) \\ g(M-1,0) & g(M-1,1) & \cdots & g(M-1, N-2) & g(M-1, N-1) \end{bmatrix}$$

Be careful!

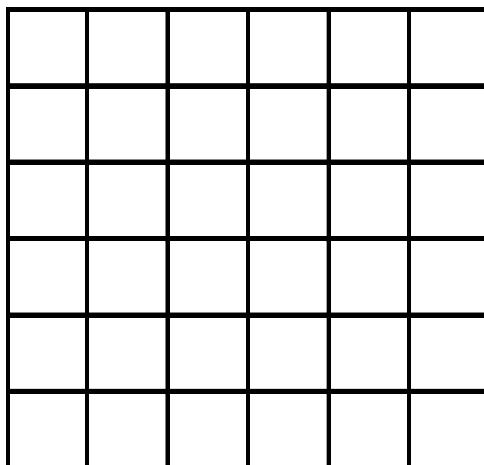


Pictorial intensity representation

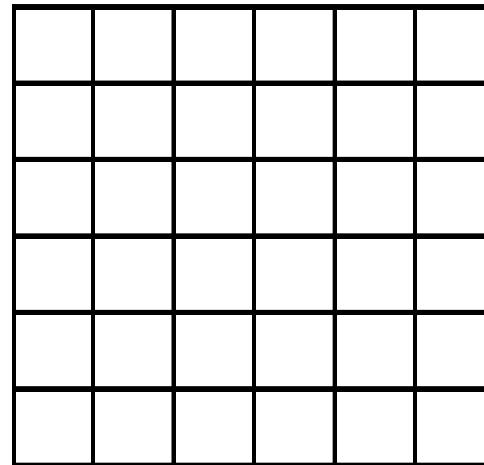
- For simplicity, a **grey-level image** can be represented as a table, with each cell representing a pixel of the image

Pictorial colour representation

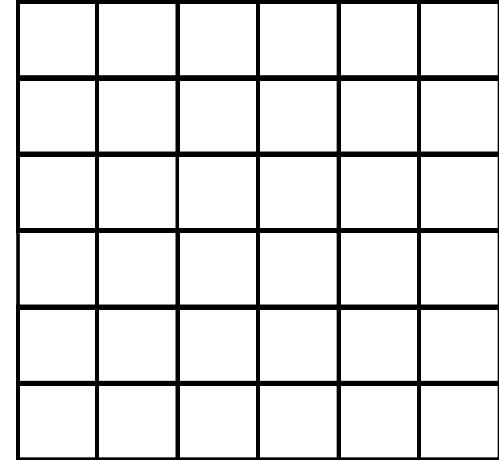
- A **colour image** can be represented with its **components**
- Each component is then represented as a grey-level image



R



G



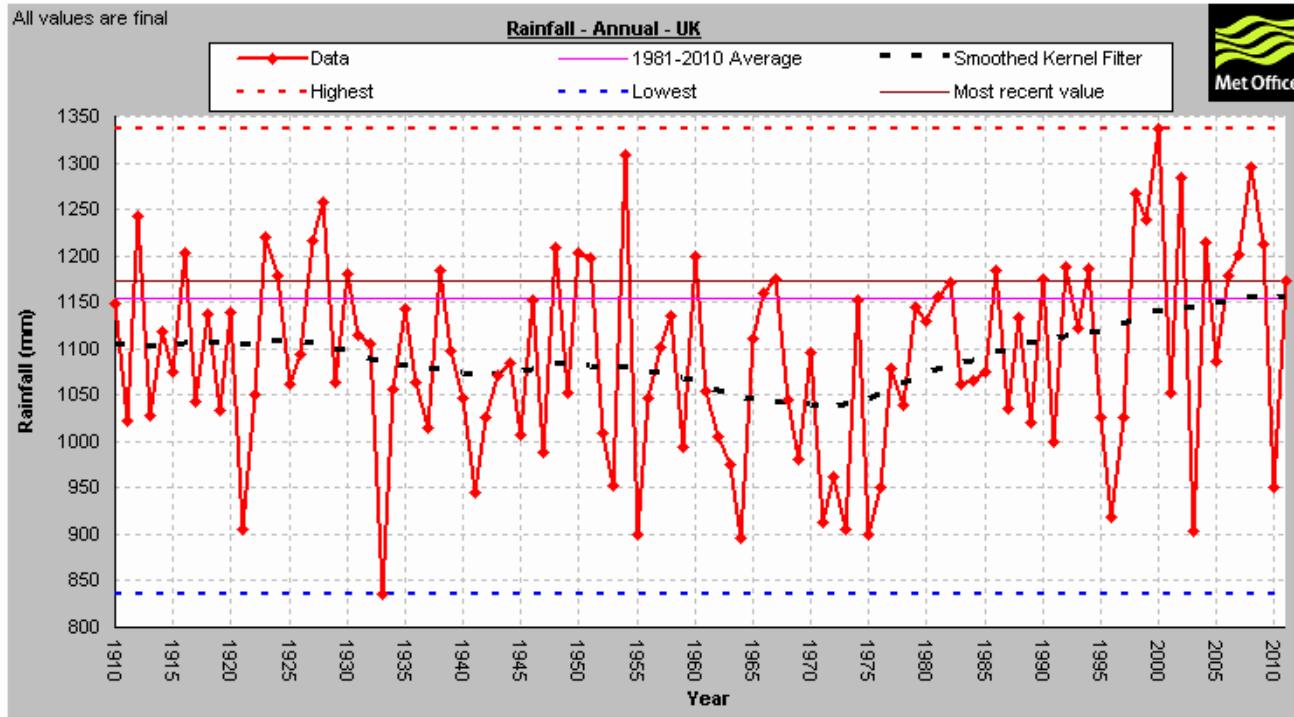
B

What is a pixel?

Pixel = picture element

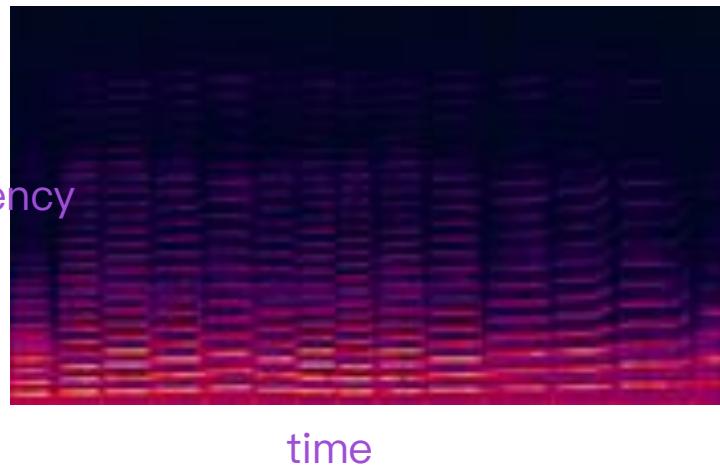
- It is *not* a little square!
- It is a *sample* from a **continuous** 2D function
- Where does the word come from?
- When was the word invented?

What is an image?

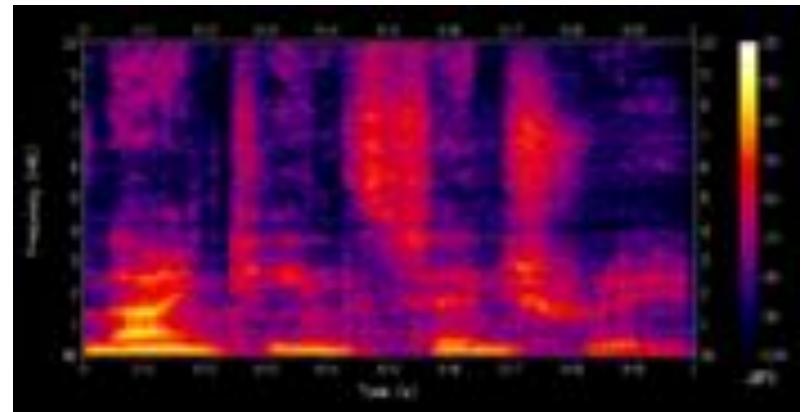


- This 1D function (time series) is not an image
- What about if we chop it up, and stack into a 2D table?
- Then it is a 2D function (month,year) → temperature
- But is *that* an image?

Are these images?



Violin sound



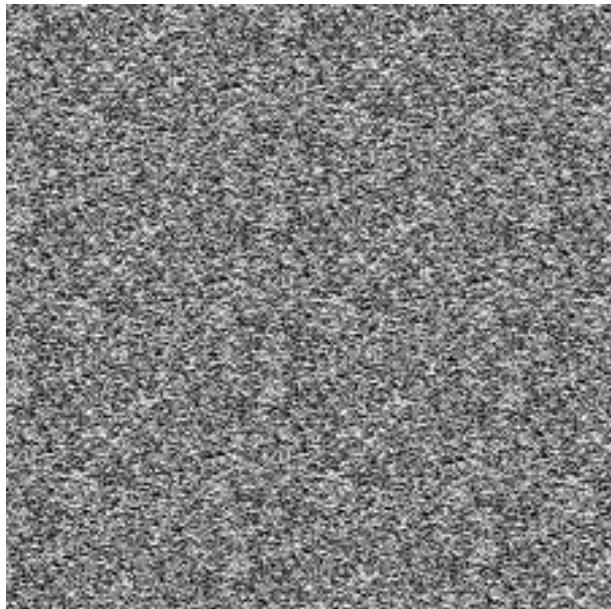
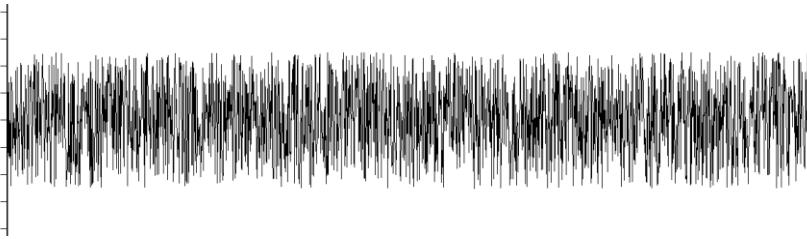
Speech: “nineteenth century”

- These are audio *spectrograms*
- They are 2D functions (time,frequency) → energy.
- But are they *images*?

An intuitive definition

- Recall the definition as a mapping $(x,y) \rightarrow F(x,y)$
- Intuitively, x and y should be *spatial* coordinates
- So they should have the same *units* Frequency – 1/sec
Time – sec
- Could be pixels or degrees of visual angle
- 2D rotation and translation should make sense
- No *special* coordinates, like time or temperature
- Think of looking down and photographing objects on a table-top (no horizon!)
- There would be no ‘preferred’ direction

An exception



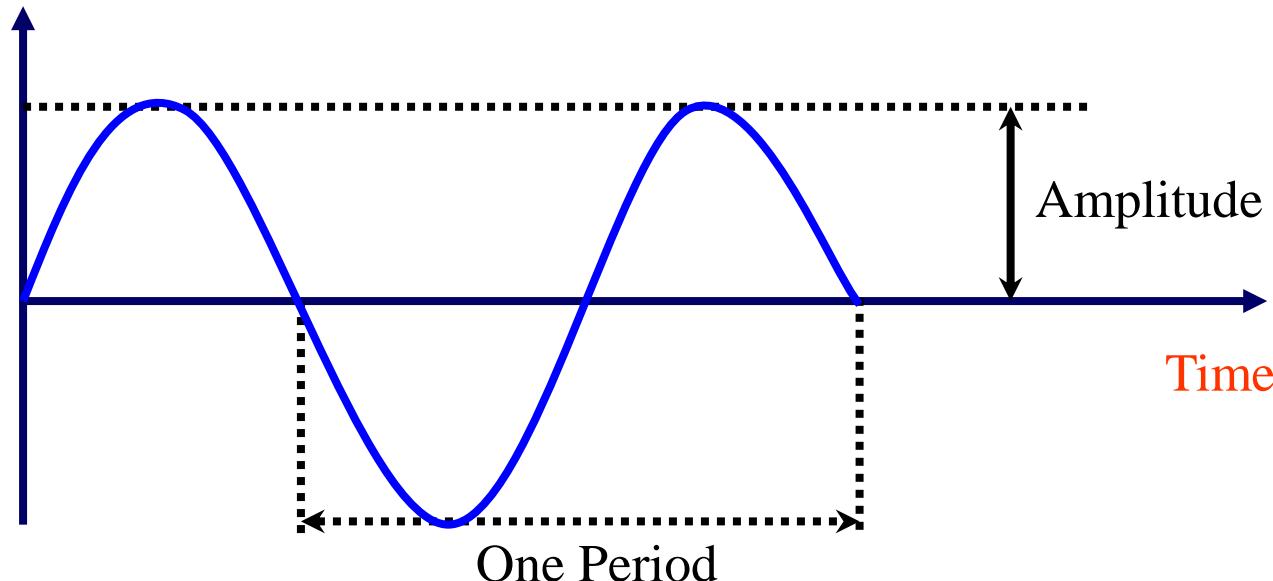
- White-noise time series $f(t)$
 - Chop and stack into a 2D table
-
- This 2D table *is* an image (in my opinion)
 - Rotation and translation make sense, because there is *no structure* in $F(x,y)$ by definition

Today's agenda

- Digital image representation
- Sampling
- Quantization
- Sub-sampling
- Pixel interpolation

Digitization 1D

- Why do we need to digitize?
 - Microphones, video cameras produce **analogue signals** (continuous-valued voltages)
 - To get audio or video into a computer, we must digitize it by converting it into a **stream of bits**

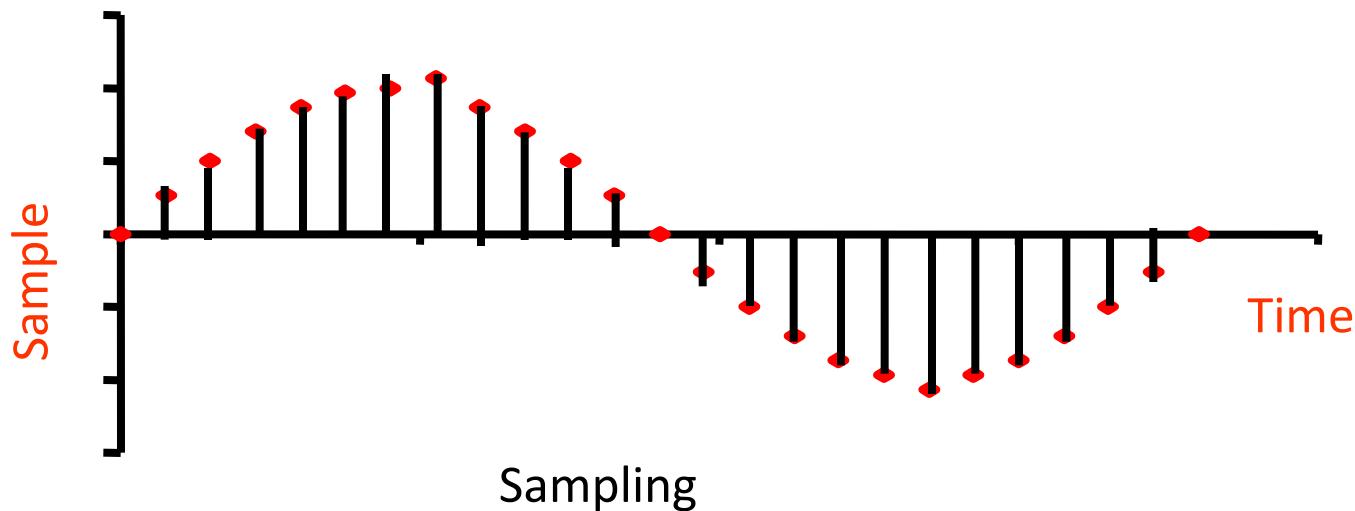


Digitization

- Sampling
 - Divide the time axis into **discrete** pieces
- Quantization
 - Divide the vertical axis (signal strength - voltage) into pieces
 - 8-bit quantization divides the vertical axis into 256 levels
 - 16 bit → 65536 levels.
 - The lower the quantization → the lower the quality of the signal

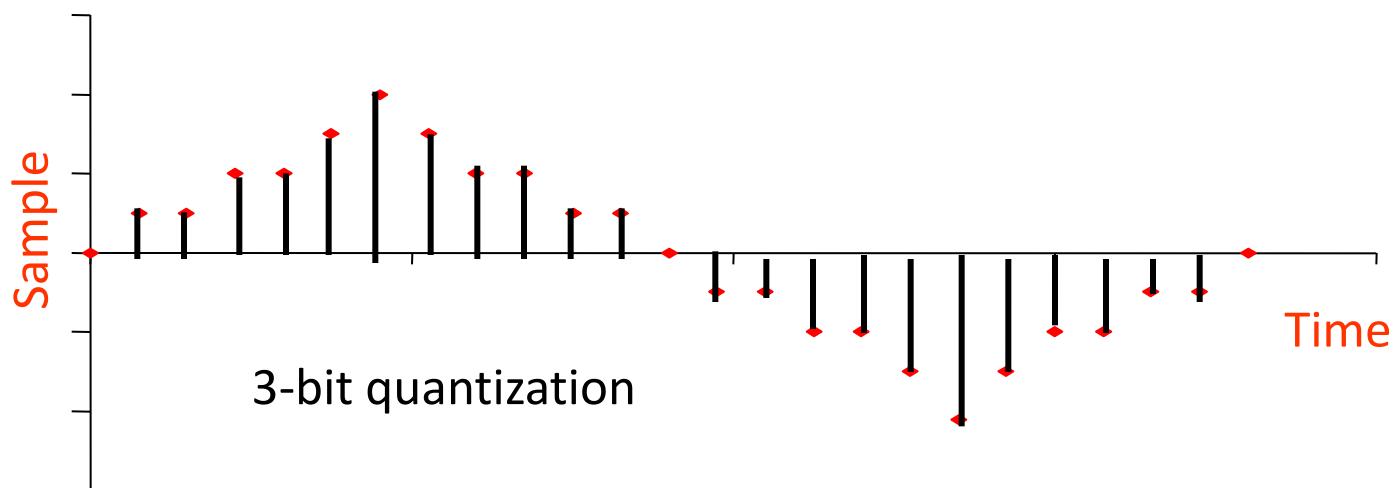
Sampling

- Sampling rate
 - number of samples per second (measured in Hz)



Quantization

- Example
 - 3-bit quantization → 8 possible sample values
 - Digitization is the reason to do quantization



Nyquist theorem

- For lossless digitization
 - the sampling rate should be **at least twice** the maximum frequency responses

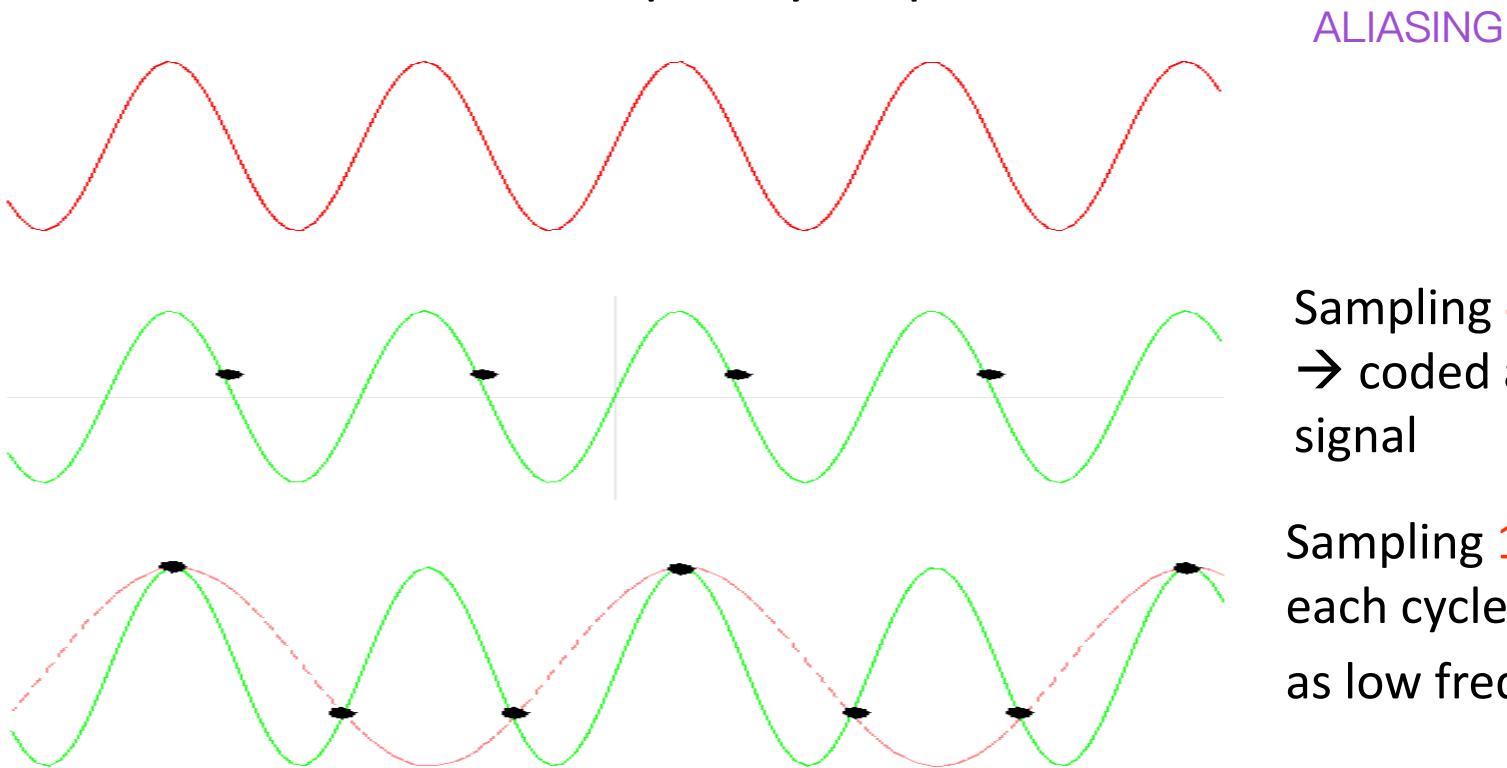
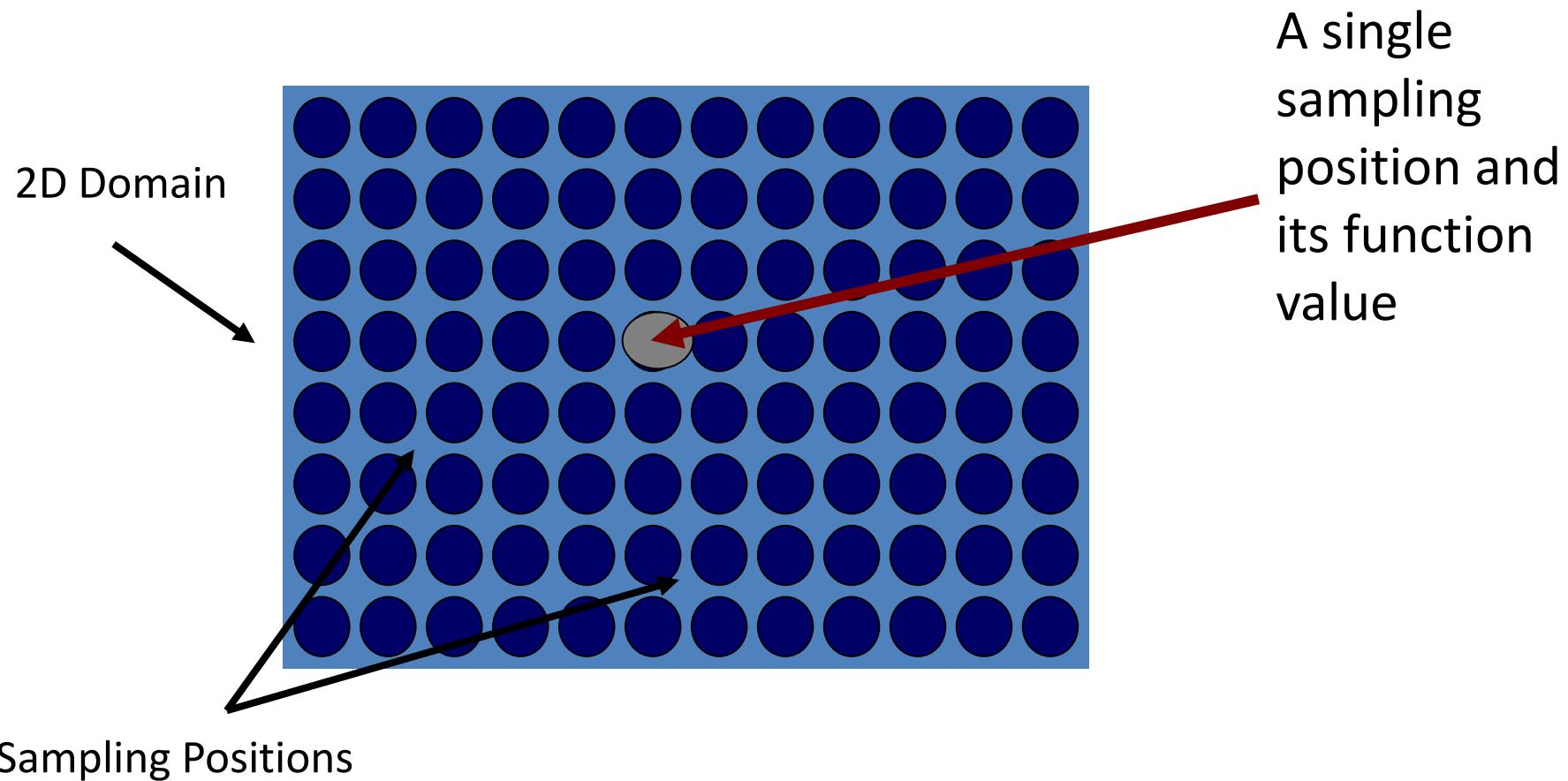
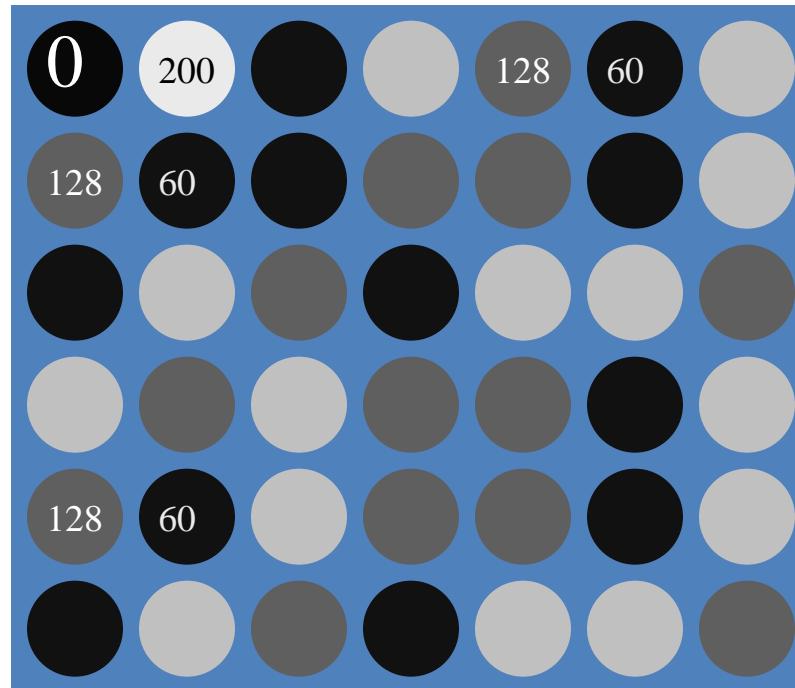
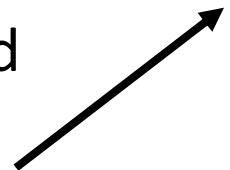


Image representation - pixels



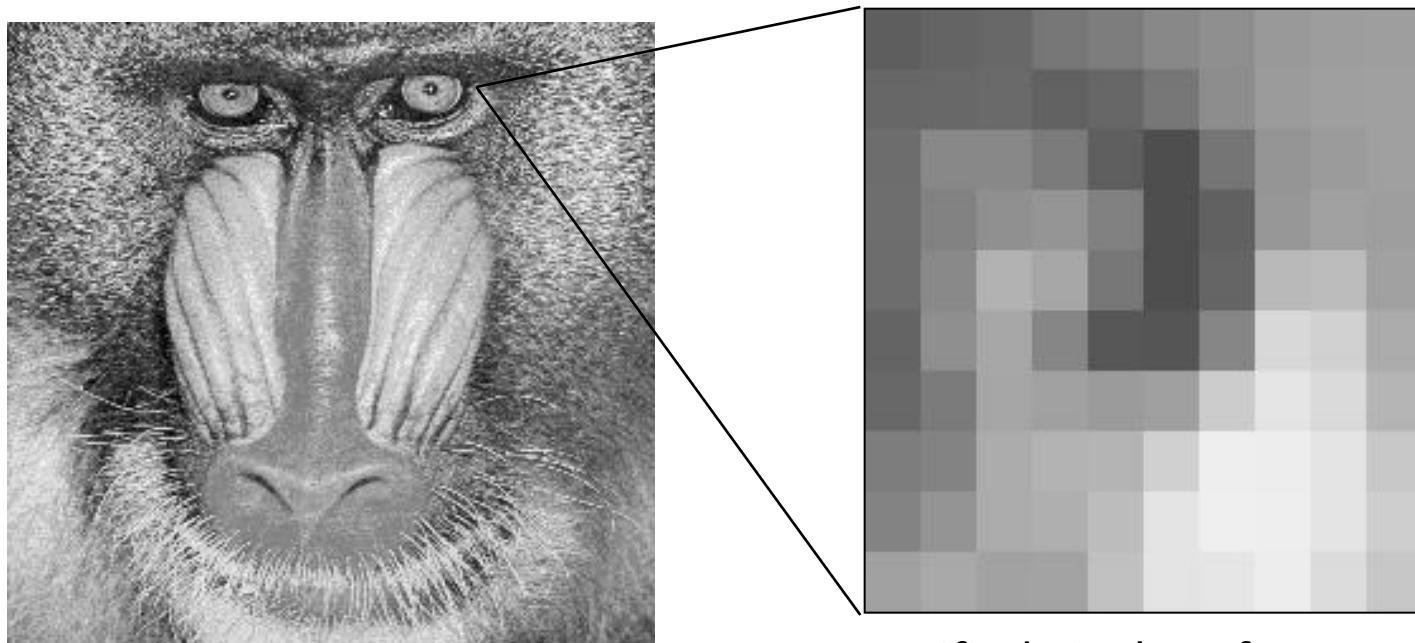
What does this mean?

Function
values at
discrete grid
points



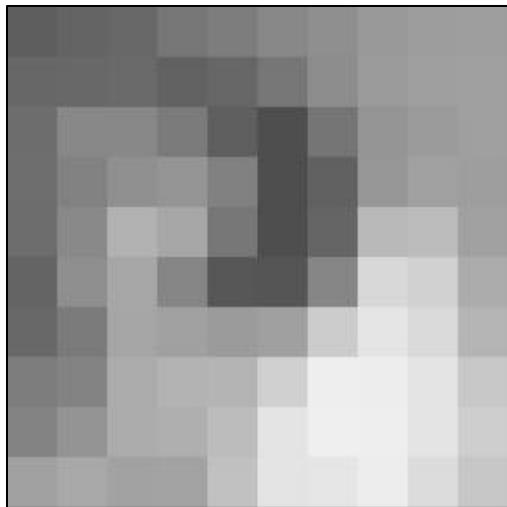
Digital images

- A digital image is a **array** of pixel values
 - E.g., in the 2D case the image data contains information of the graylevel at each position in the image



Magnified pixels at few sampling positions

Digital images

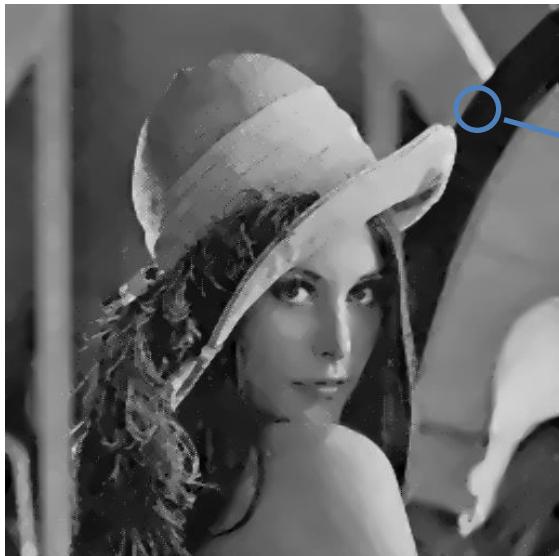


Pixels

94	100	104	119	125	136	143	153	157	158
103	104	106	98	103	119	141	155	159	160
109	136	136	123	95	78	117	149	155	160
110	130	144	149	129	78	97	151	161	158
109	137	178	167	119	78	101	185	188	161
100	143	167	134	87	85	134	216	209	172
104	123	166	161	155	160	205	229	218	181
125	131	172	179	180	208	238	237	228	200
131	148	172	175	188	228	239	238	228	206
161	169	162	163	193	228	230	237	220	199

Corresponding array

A digital image



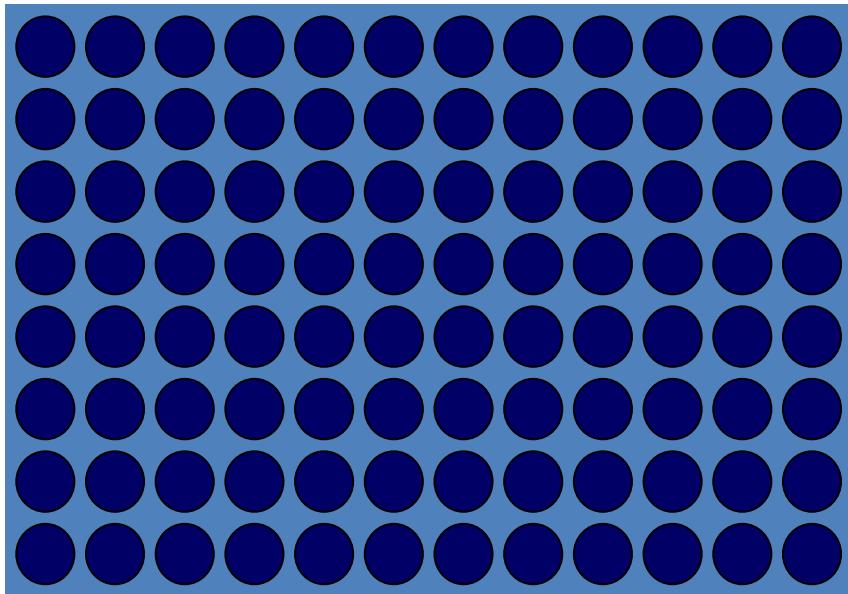
134	135	132	12	15...
133	134	133	133	11...
130	133	132	16	12...
137	135	13	14	13...
140	135	134	14	12...

Image digitization



Scene

digitizer
(e.g.,scanner)



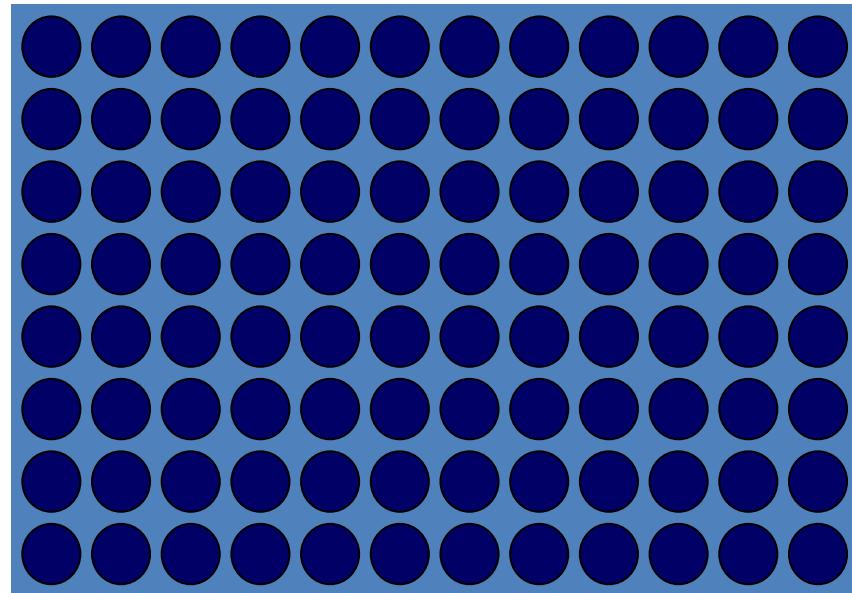
$g(i,j)$

Image digitization

- The continuous image domain is scanned
 - the brightness values are measured or sampled at discrete locations to form an array of intensity values



sampler
→



Continuous domain

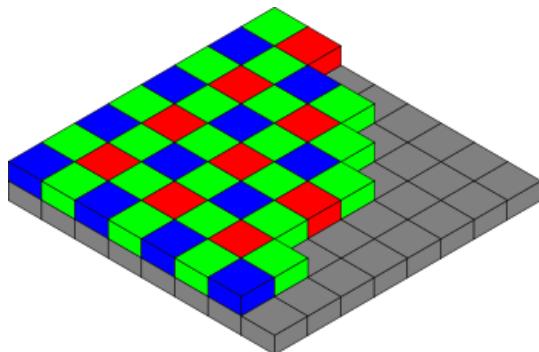
$g(i,j)$
array of sampling positions

What about cameras?

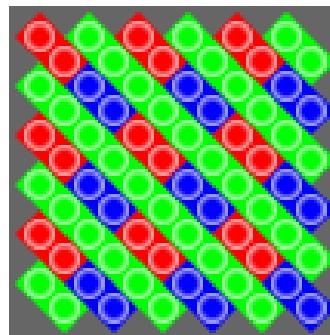
Charge-coupled Device

CCD是一种半导体器件，能够把光学影像转化为数字信号。

- Each CCD-element measures scalar *intensity*
- So we need to filter the light, and measure each spectral band
- How can this be done on a rectangular array?



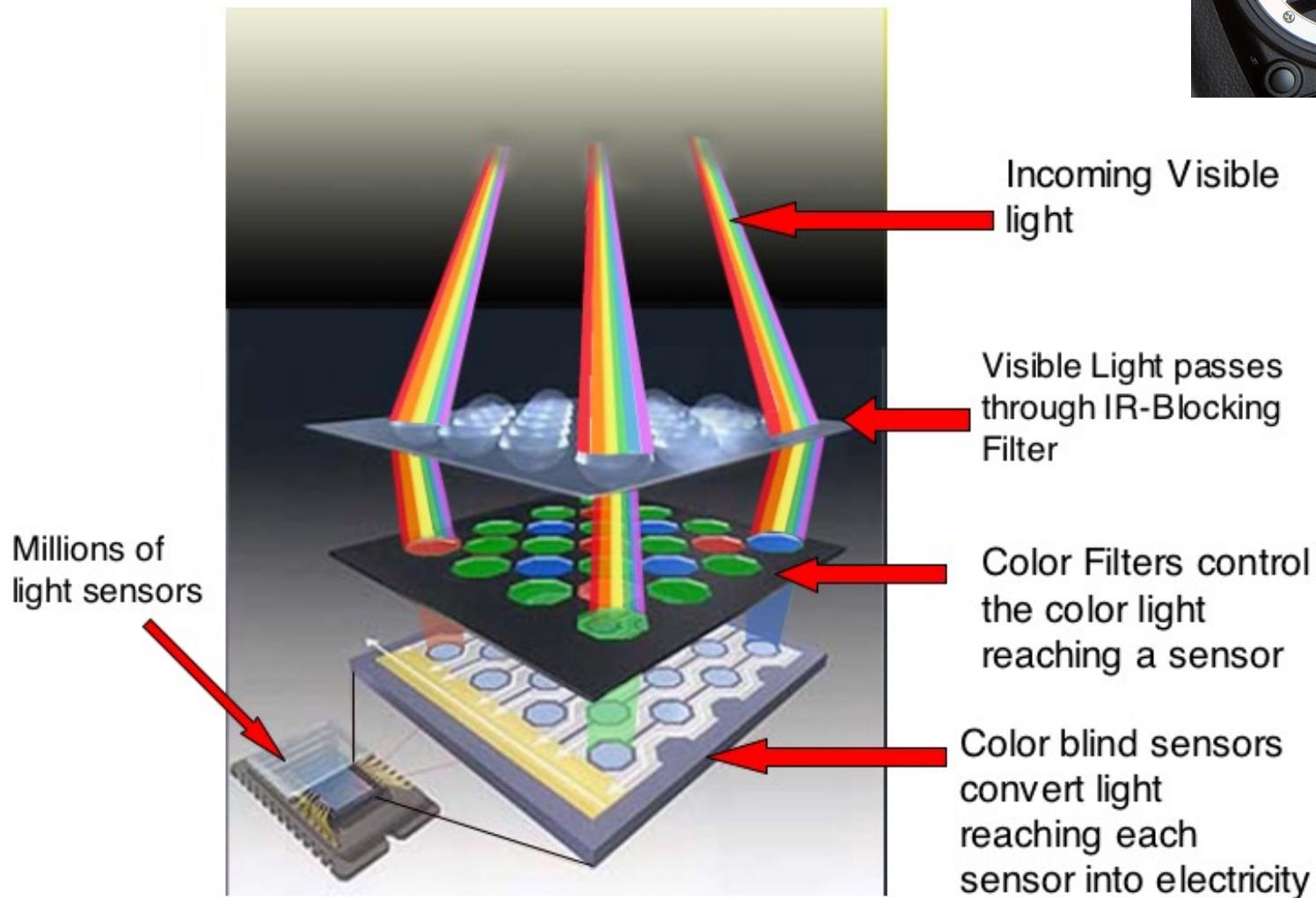
Bayer filter (standard)



Fuji EXR

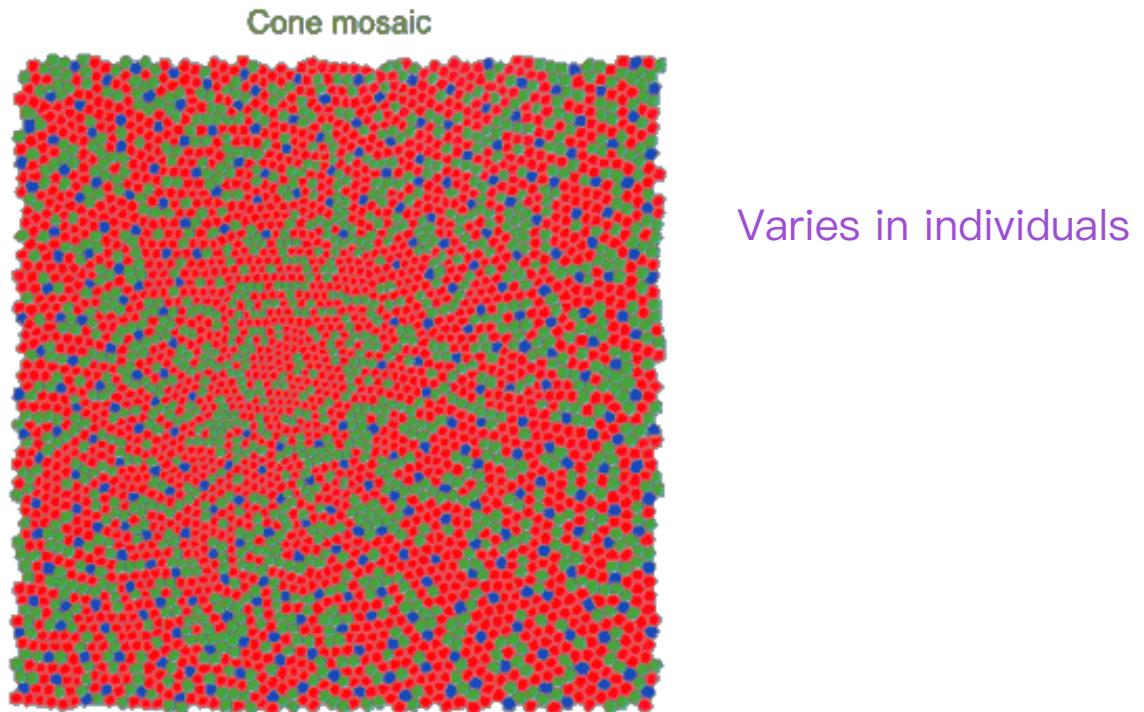
网上找的图：

RGB Inside the Camera



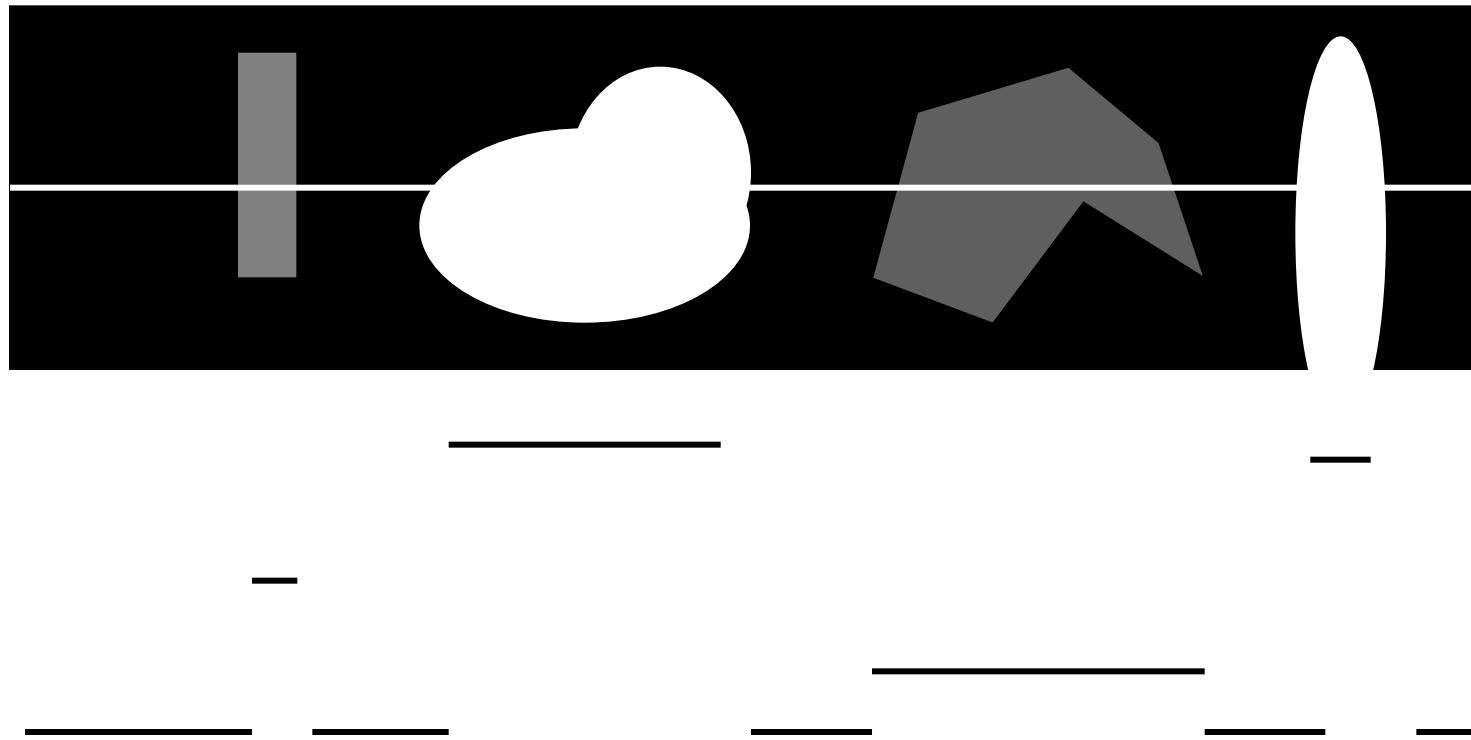
What about eyes?

- Retinal *cone mosaic*
- Very few blue cones (none in fovea)



Scanline

- Converting the continuous 2D signal in a digital image by sampling per scanlines



Example

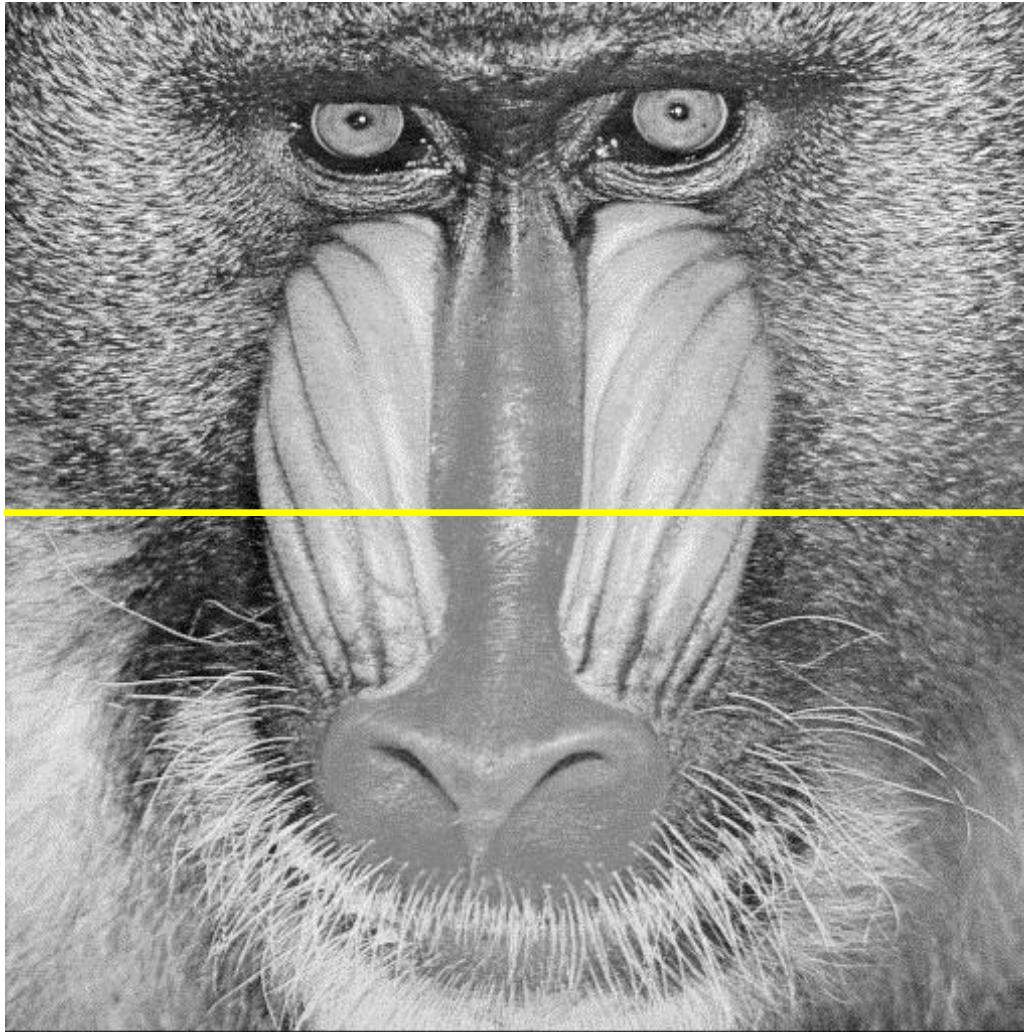


Image sampling

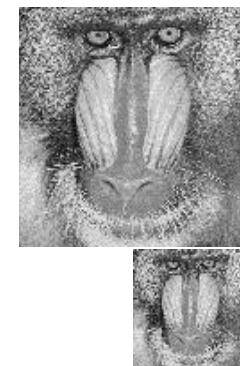
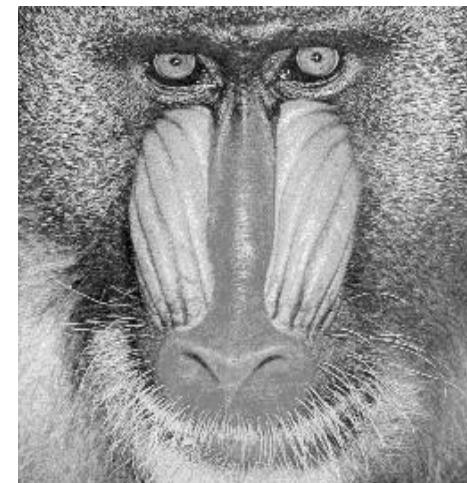
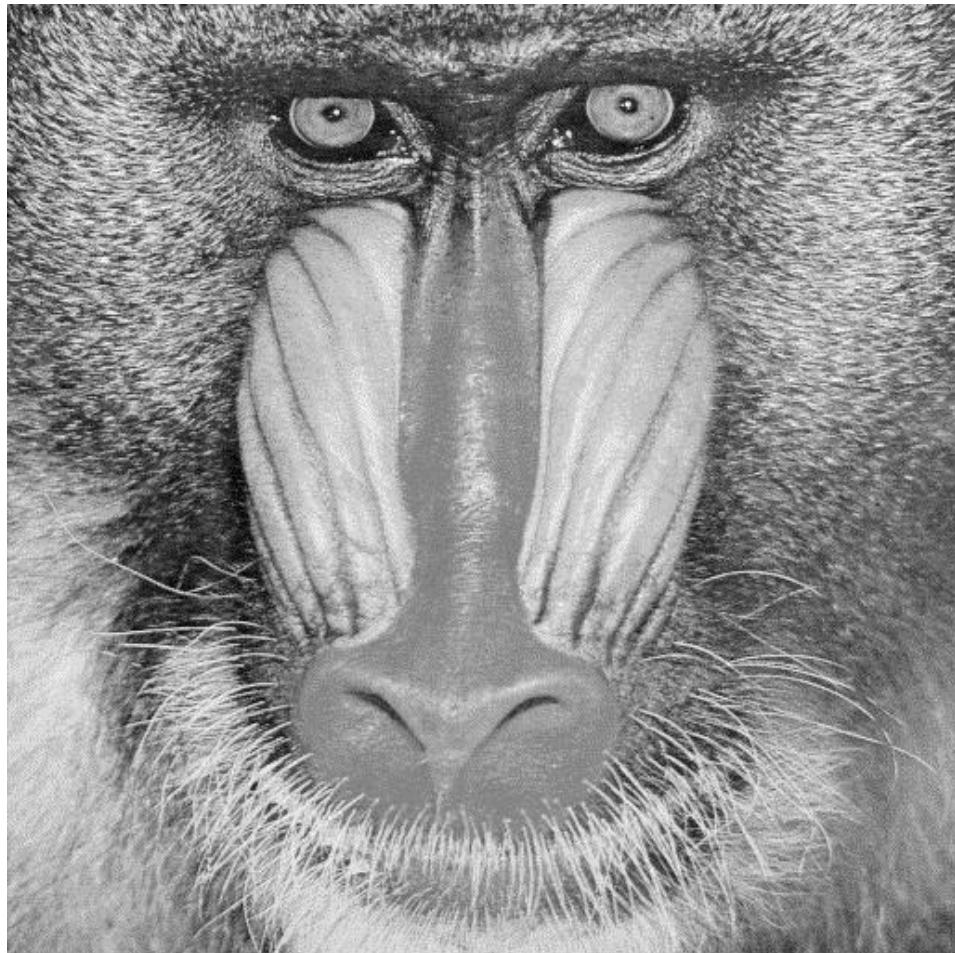
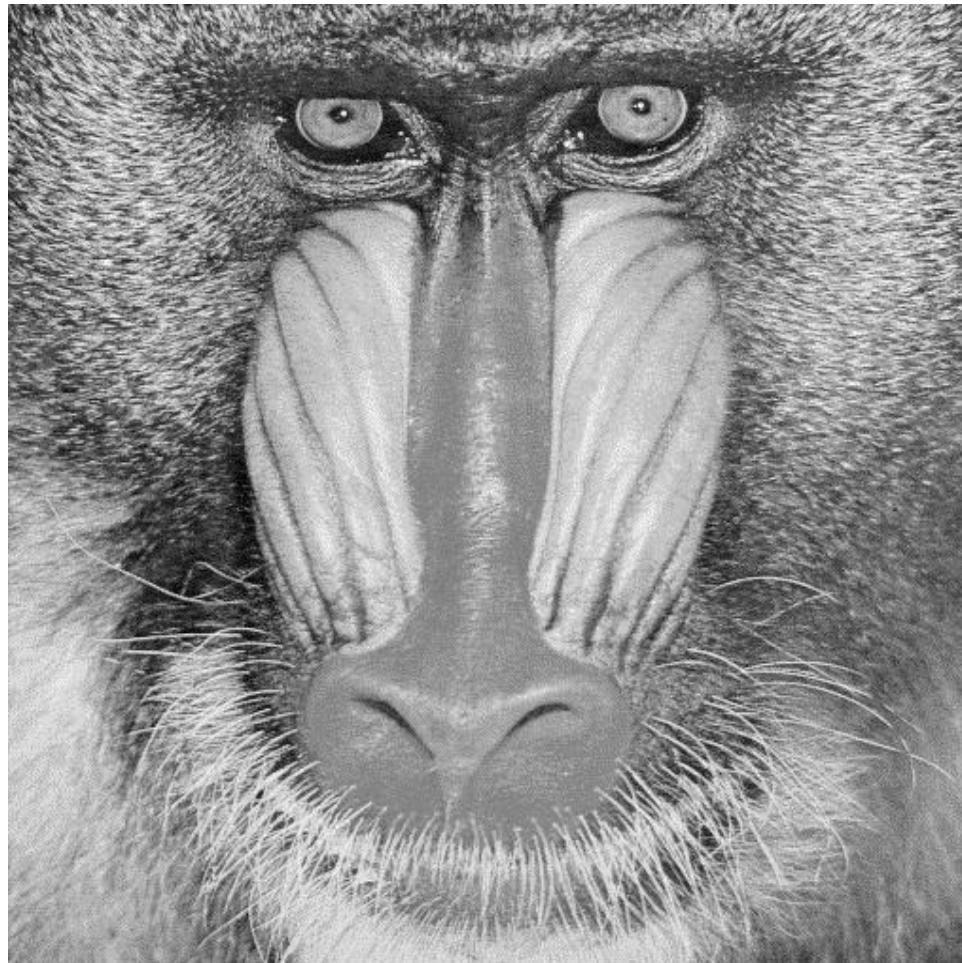
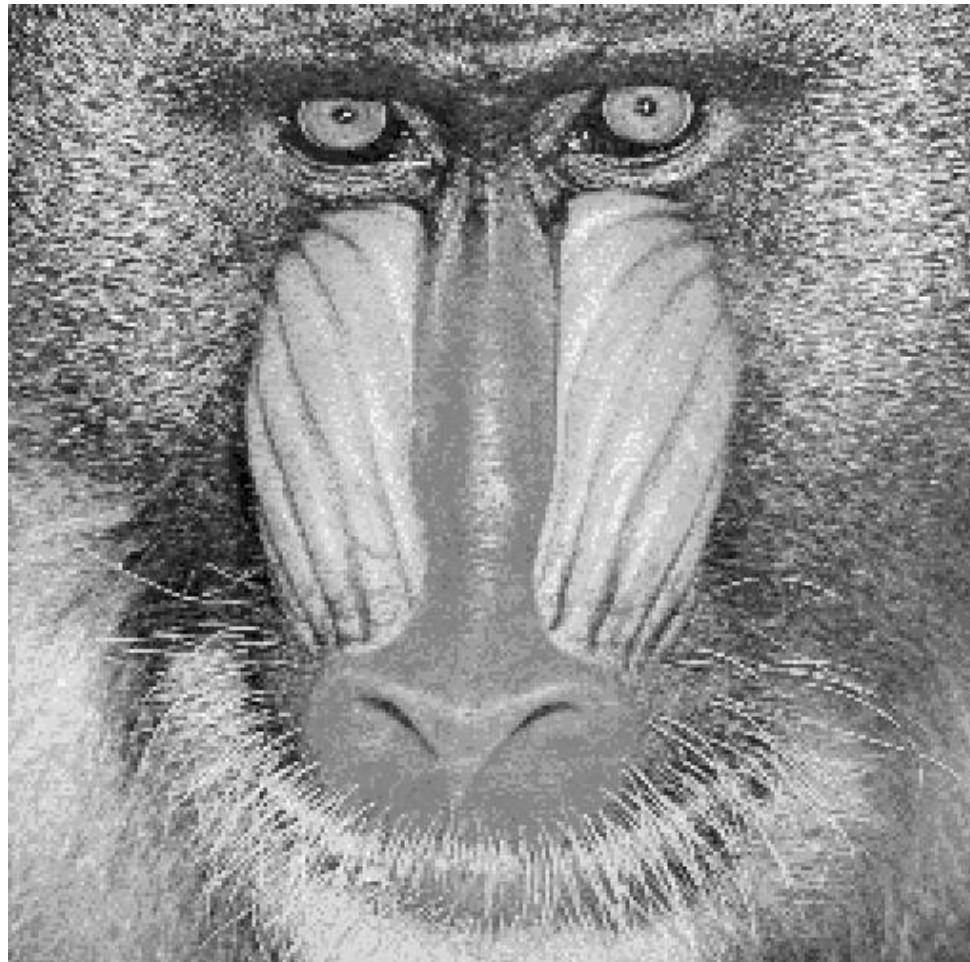


Image sampling



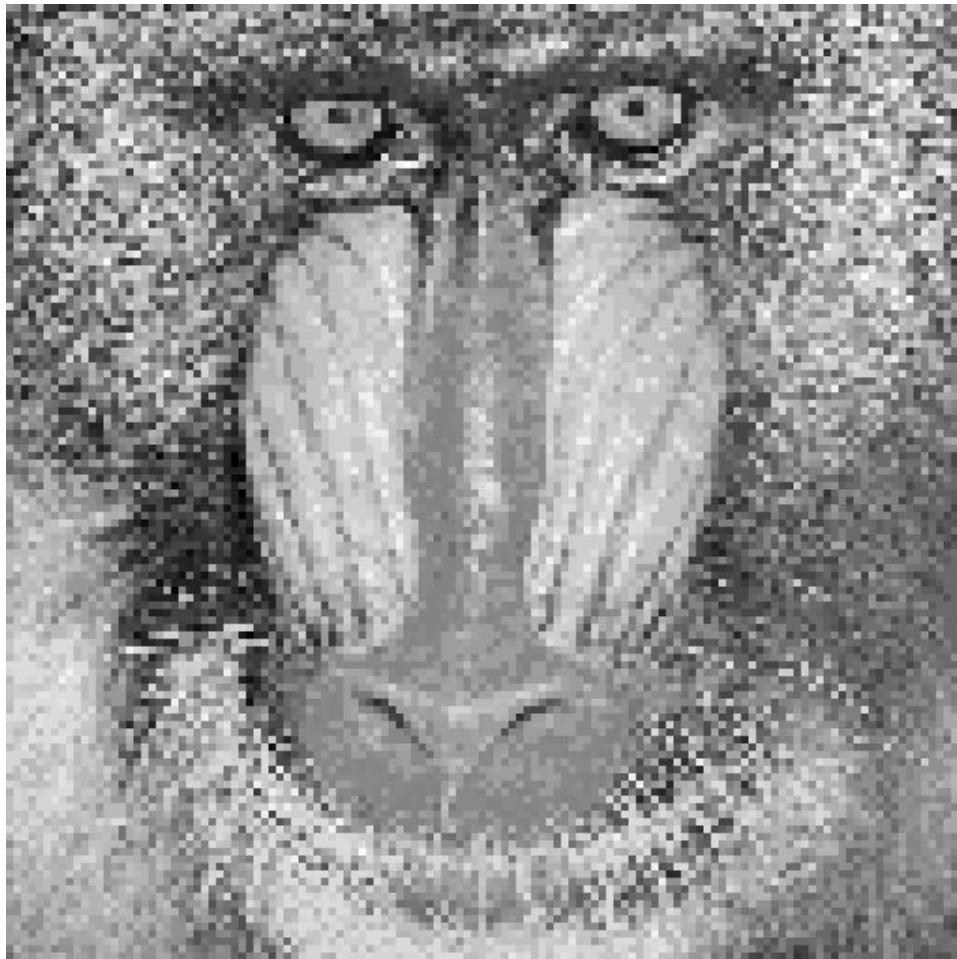
512

Image sampling



256

Image sampling



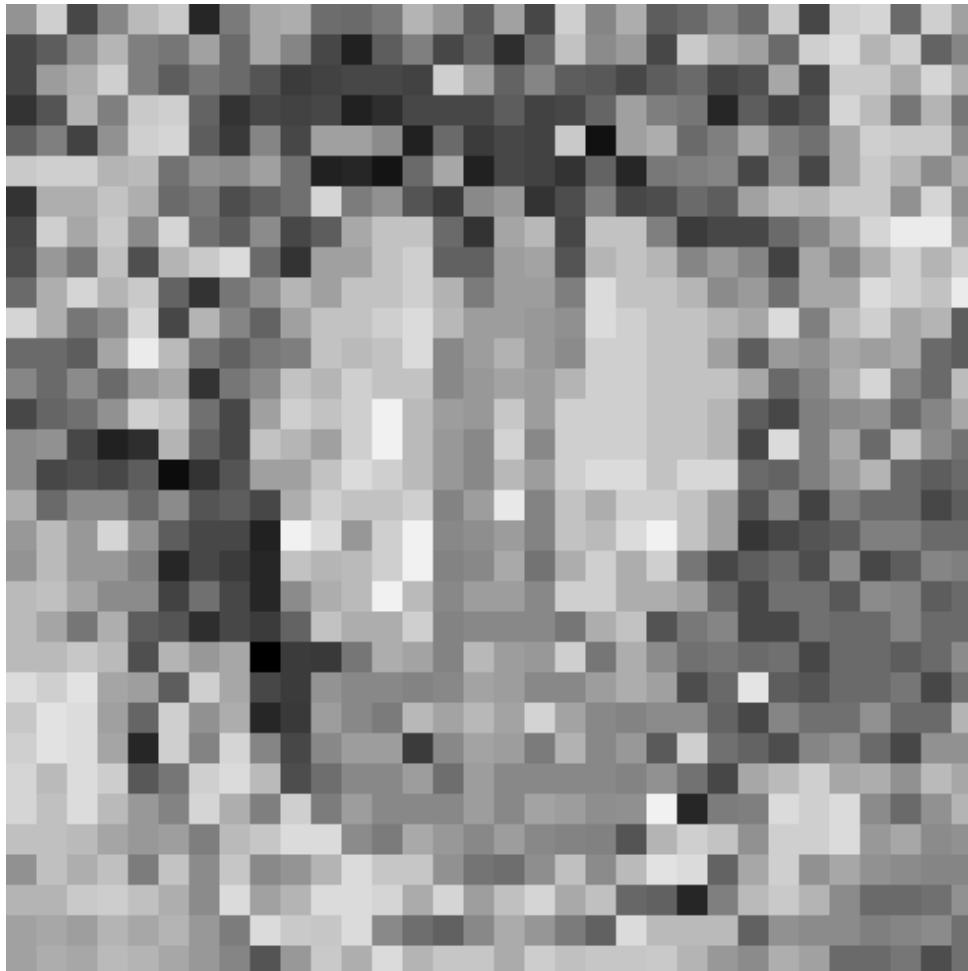
128

Image sampling



64

Image sampling



32

Image resolution



Full resolution

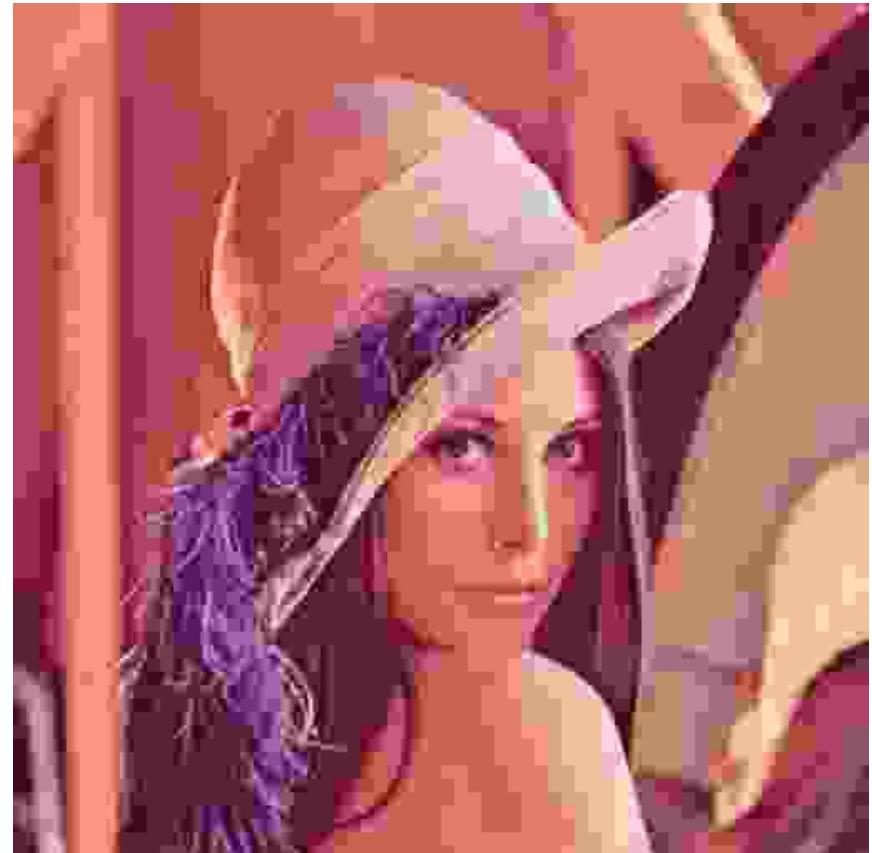


1/4 resolution

Image resolution



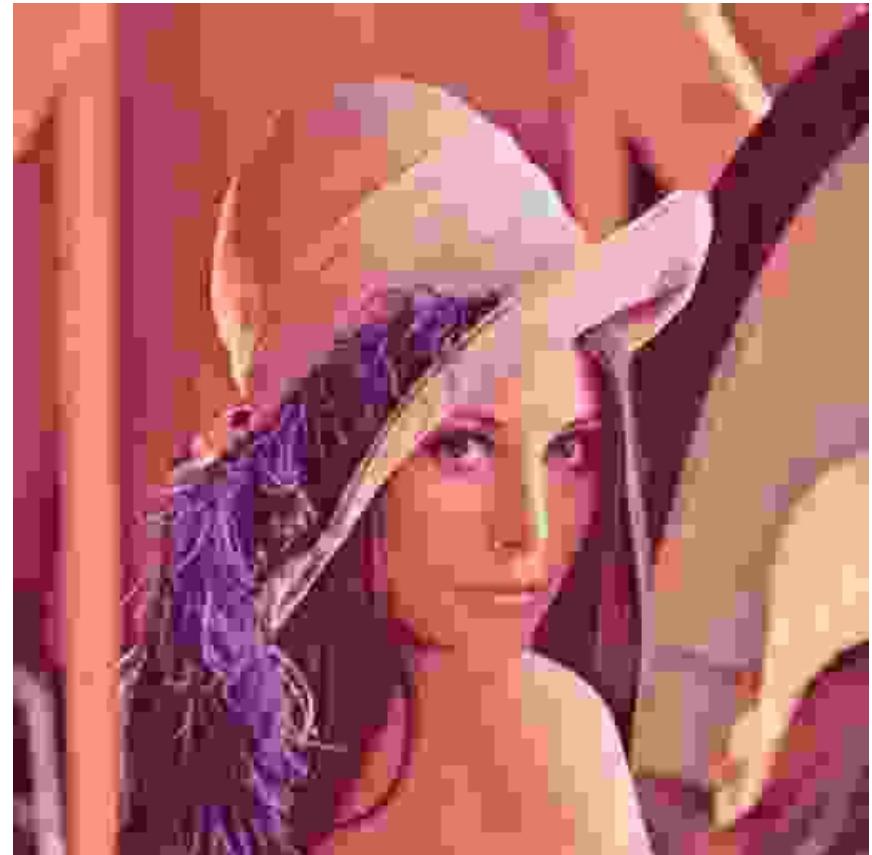
Full resolution



1/8 resolution

Low resolution

- The image appears blocky
- To accurately represent the original continuous scene, the **sampling rate** must be sufficiently high ...



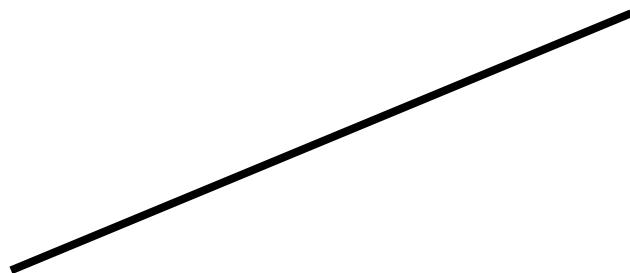
1/8 resolution

Image sampling methods

- Uniform
 - same sampling frequency everywhere
- Adaptive
 - higher sampling frequency in areas with greater **detail**
 - **compression** strategy

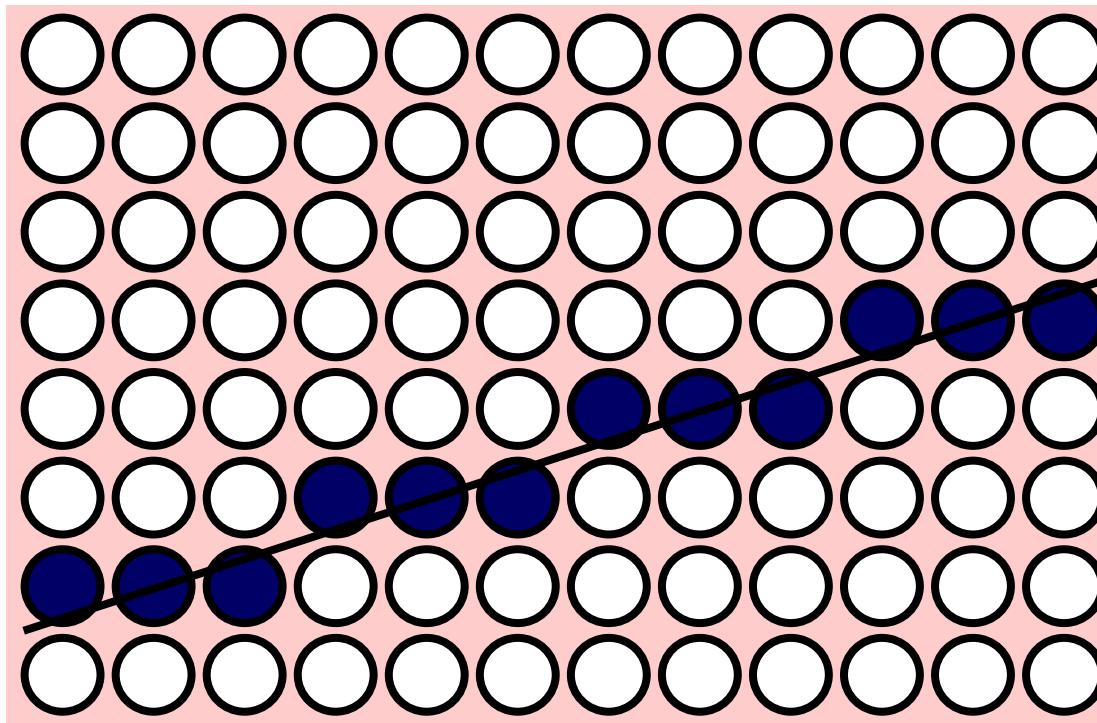
Sampling effects

- How to represent this line with discrete pixel values?



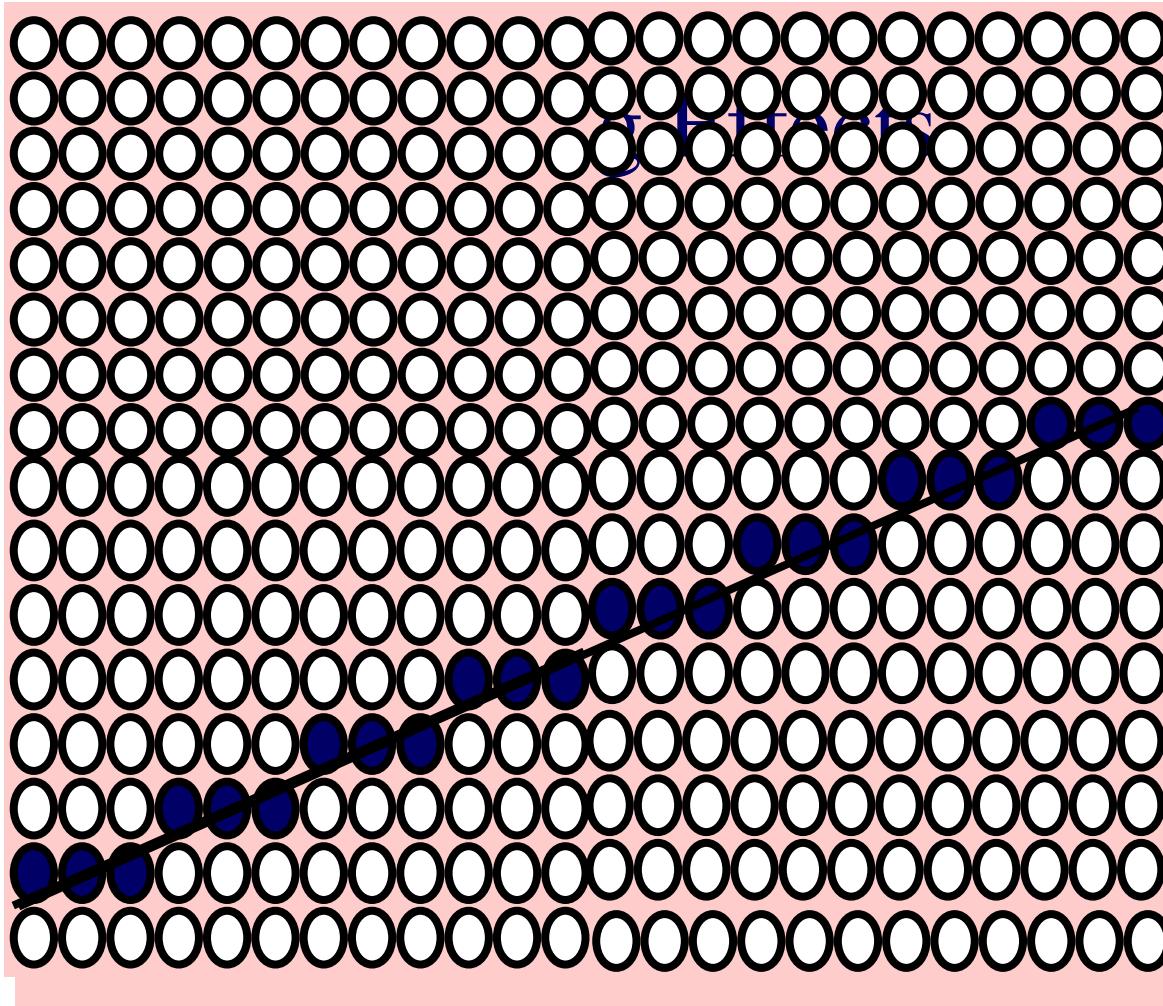
Sampling effects

Standard midpoint line on a binary representation



- Representing a line with discrete pixel values can lead to **sampling error** and loss of information

Sampling effects



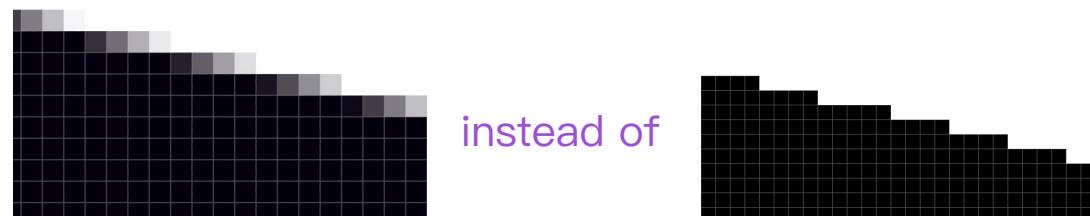
Same line with twice the linear resolution

Sampling effects

- Sampling errors in representing a line
 - Doubling resolution does not solve the problem
 - It costs 4 times memory, bandwidth and scan conversion time!

ANTI-ALIASING

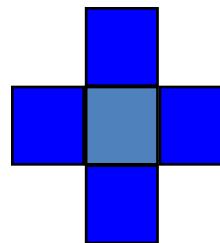
- The problem can be alleviated using **more grey-levels**



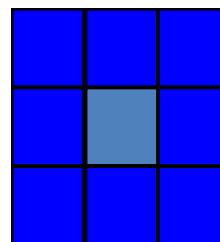
Relationship between pixels

- Depending on the neighbourhood definition, a pixel has 4 or 8 neighbours

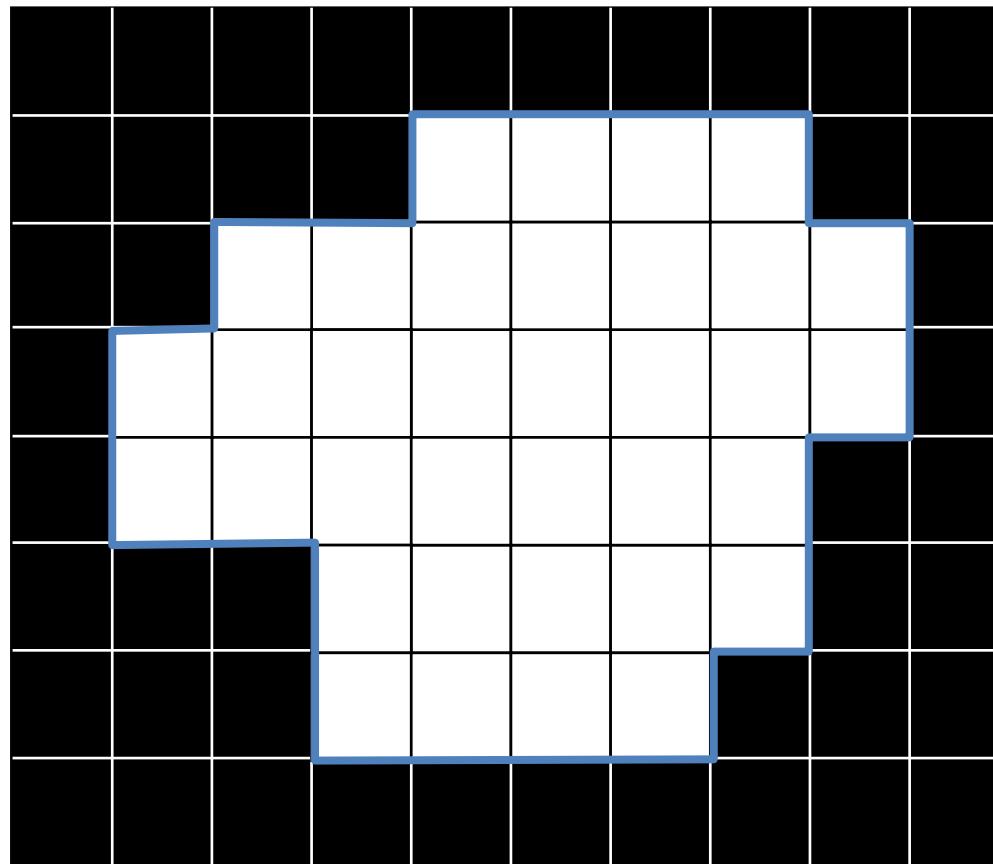
- 4-neighbourhood:
each neighbour shares a single edge with the pixel



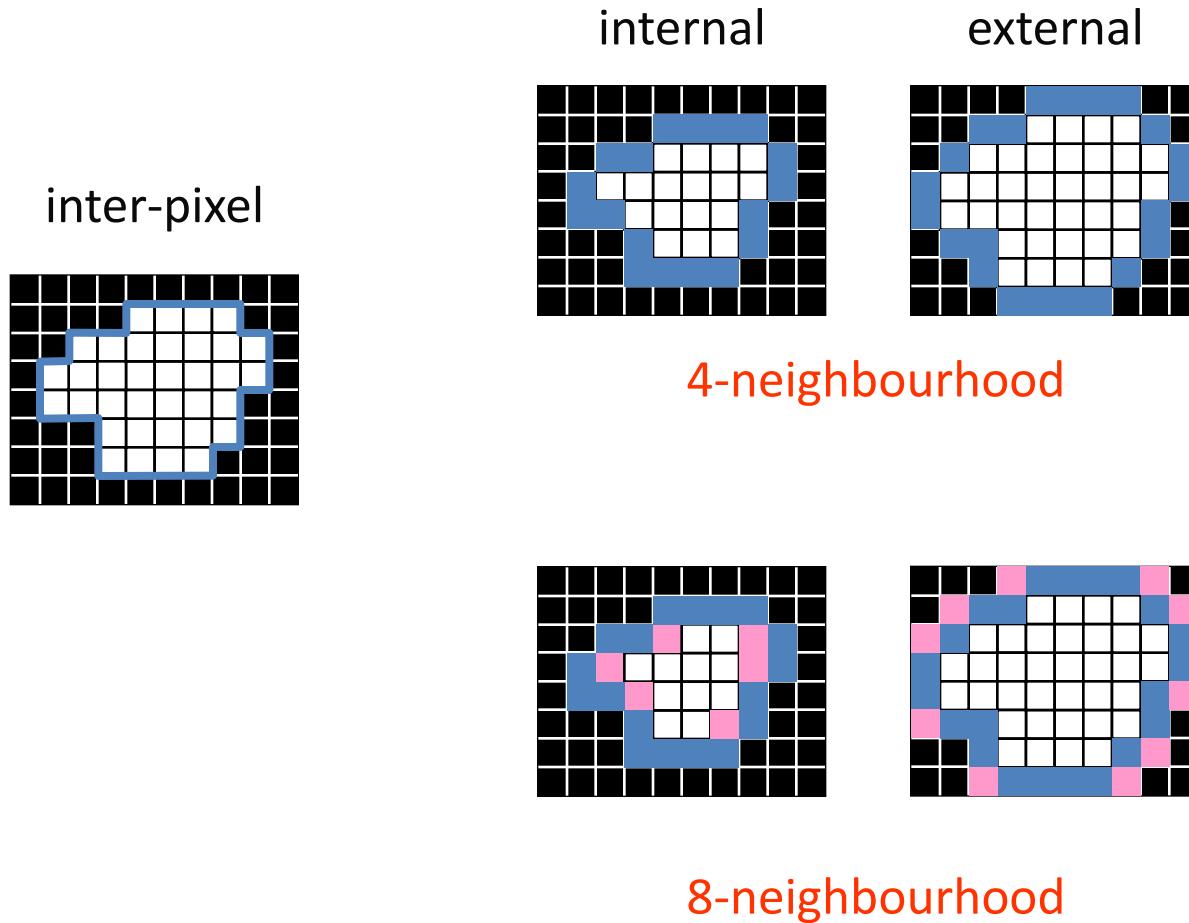
- 8-neighbourhood:
each neighbour shares an edge or a corner with the pixel



Shape representation

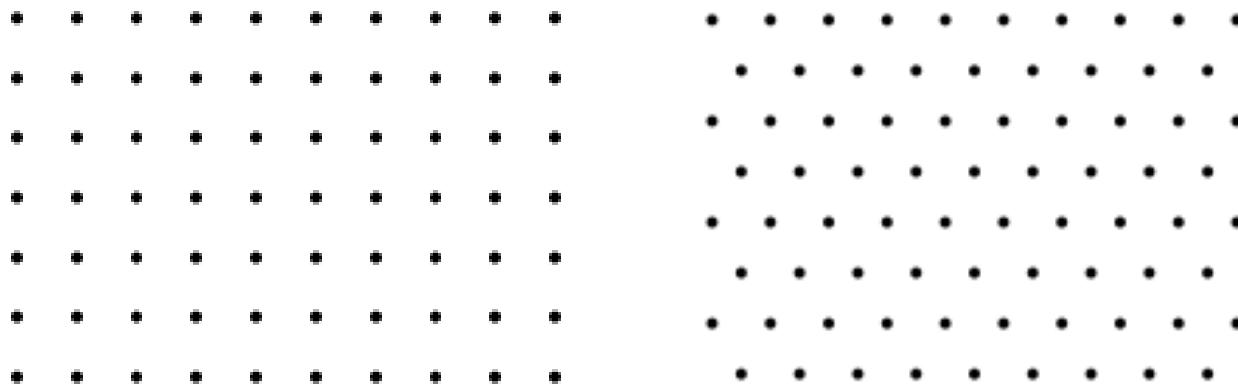


Shape representation

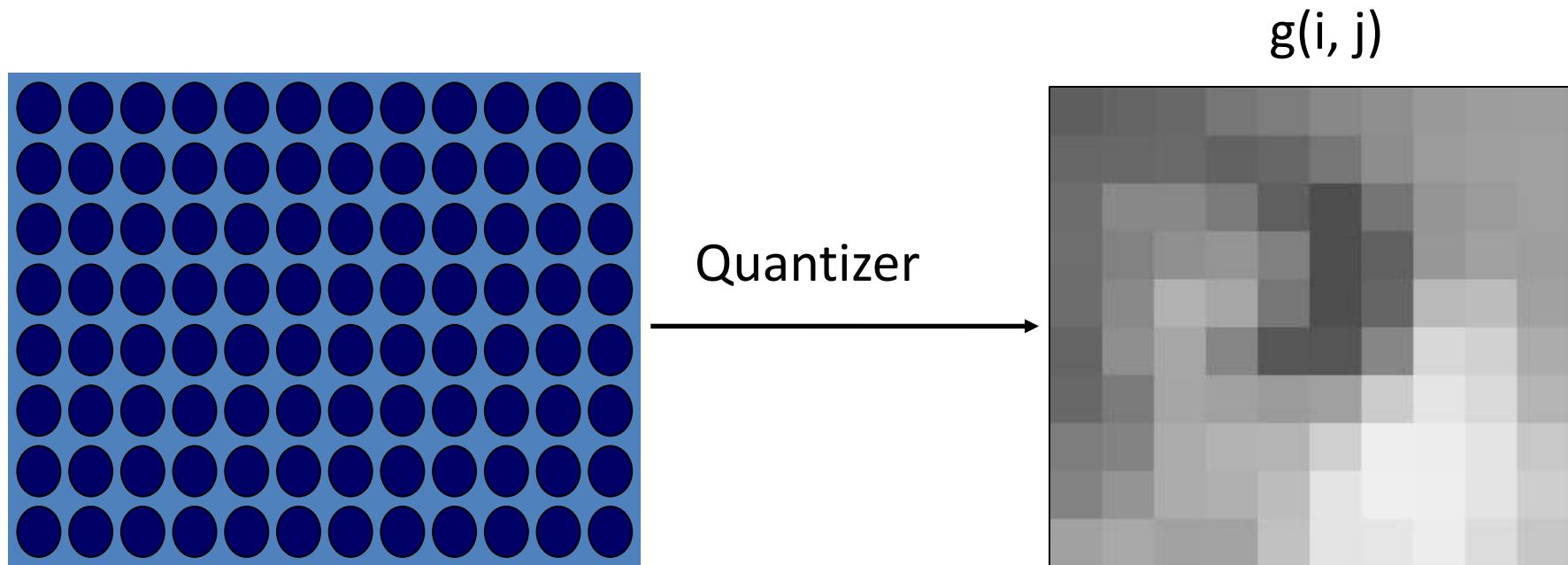


Alternative sampling schemes

- Cartesian sampling pattern is convenient
- Is it optimal in any formal sense?
- Hexagonal lattice is more *isotropic*.
- Less important for high-resolution images.



Quantization



Each element in the matrix is quantized,
i.e., replaced by an integer
Quantized values are called gray levels

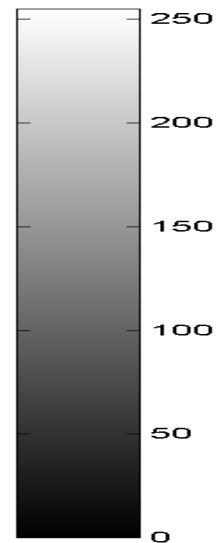
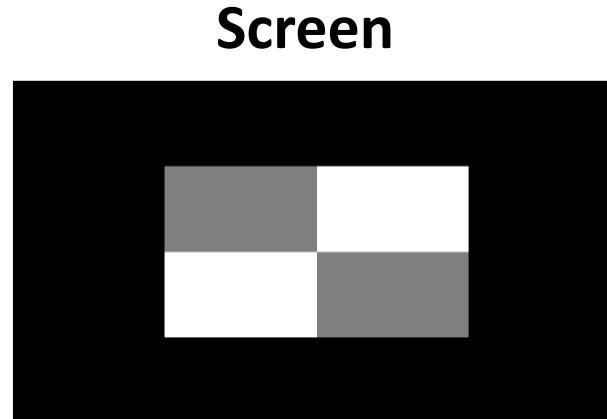
Digital image visualization

- Each pixel in the image is usually shown by a single pixel on the screen

Example → for $L = 256$ gray levels

- 0 maps into black
- 255 into white
- values in between map linearly into various levels of gray

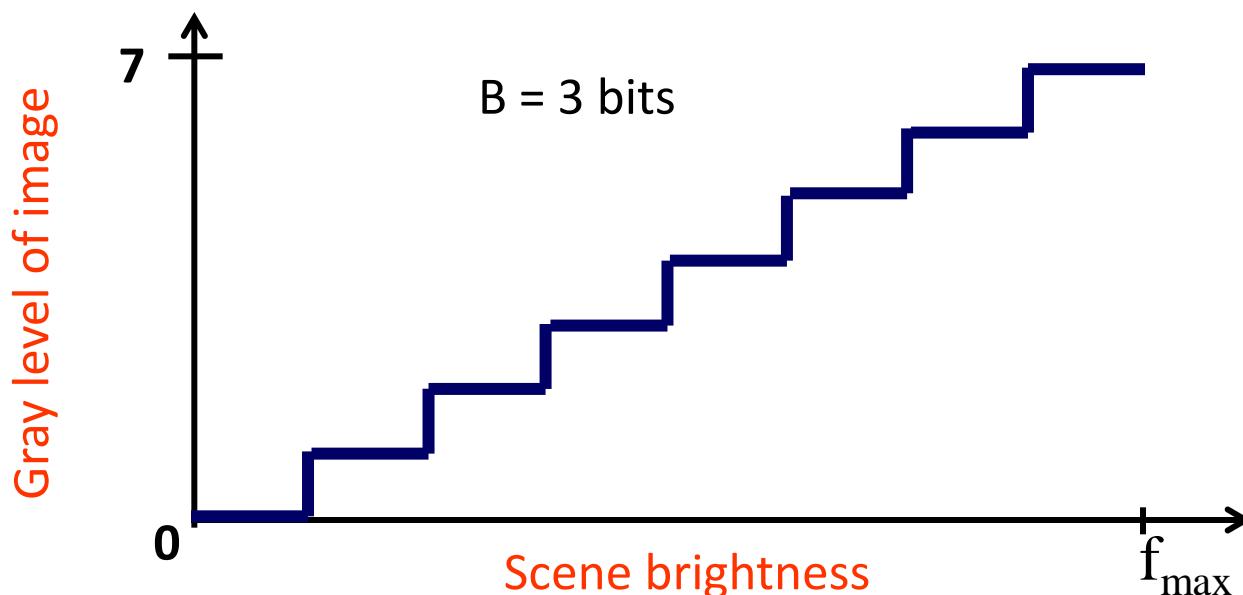
0	0	0	0
0	127	255	0
0	255	127	0
0	0	0	0



Intensity resolution

- Intensity resolution

- refers to how **accurately** a pixel's gray level represents the **brightness** of the corresponding point in the original scene
- during quantization, the brightness sampled at each point in the **continuous-tone** image is replaced by an **integer value**



Quantization formula

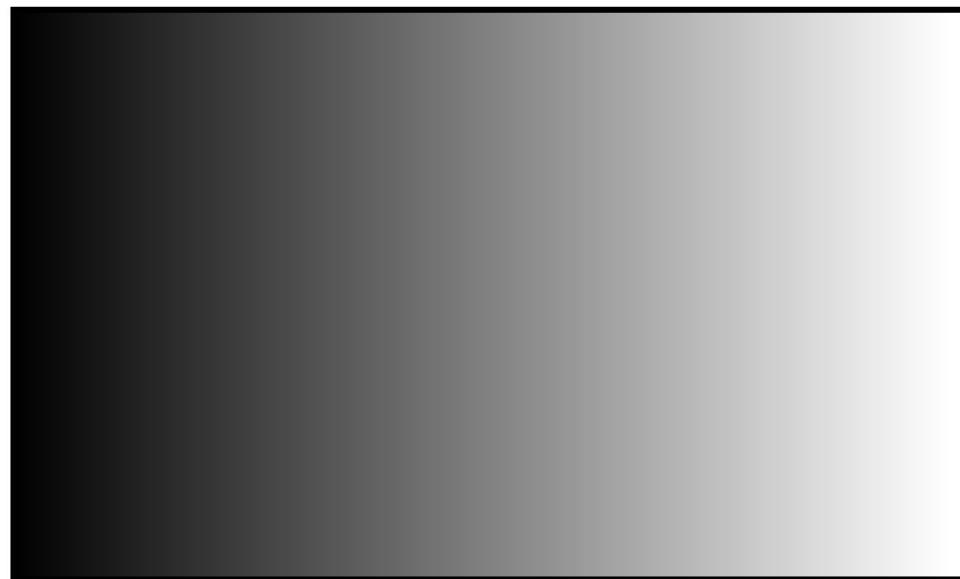
- First round the signal f to the nearest *number* of steps
- Then multiply the count by the step-size Δ

$$g = \text{sign}(f) \times \text{floor} \left(\frac{|f|}{\Delta} + \frac{1}{2} \right) \times \Delta$$

- $\text{floor}(x)$ is the largest integer that is smaller than x

Intensity resolution

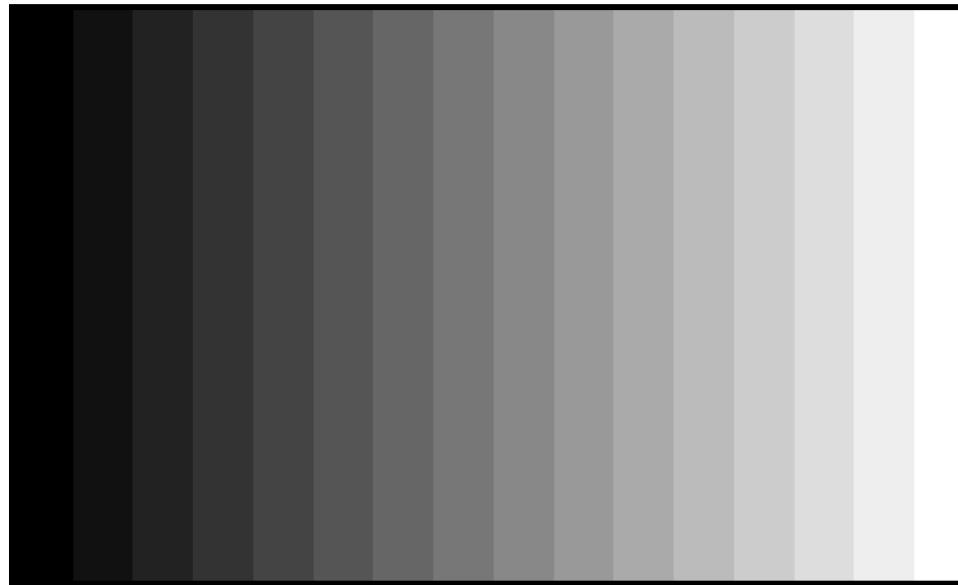
- Intensity resolution
 - depends on the number of bits available



Digital image quantized with 8 bits (256 gray levels)

Note that the image appears continuous

Intensity resolution



The same image quantized with only 4 bits (16 gray levels)

Now the image brightness appears **discontinuous**

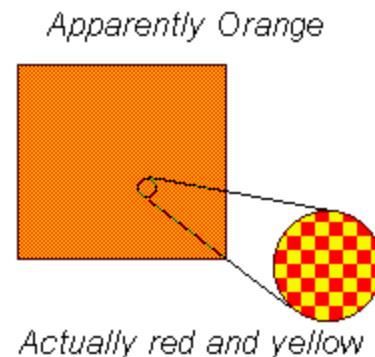
Intensity resolution

- Intensity resolution

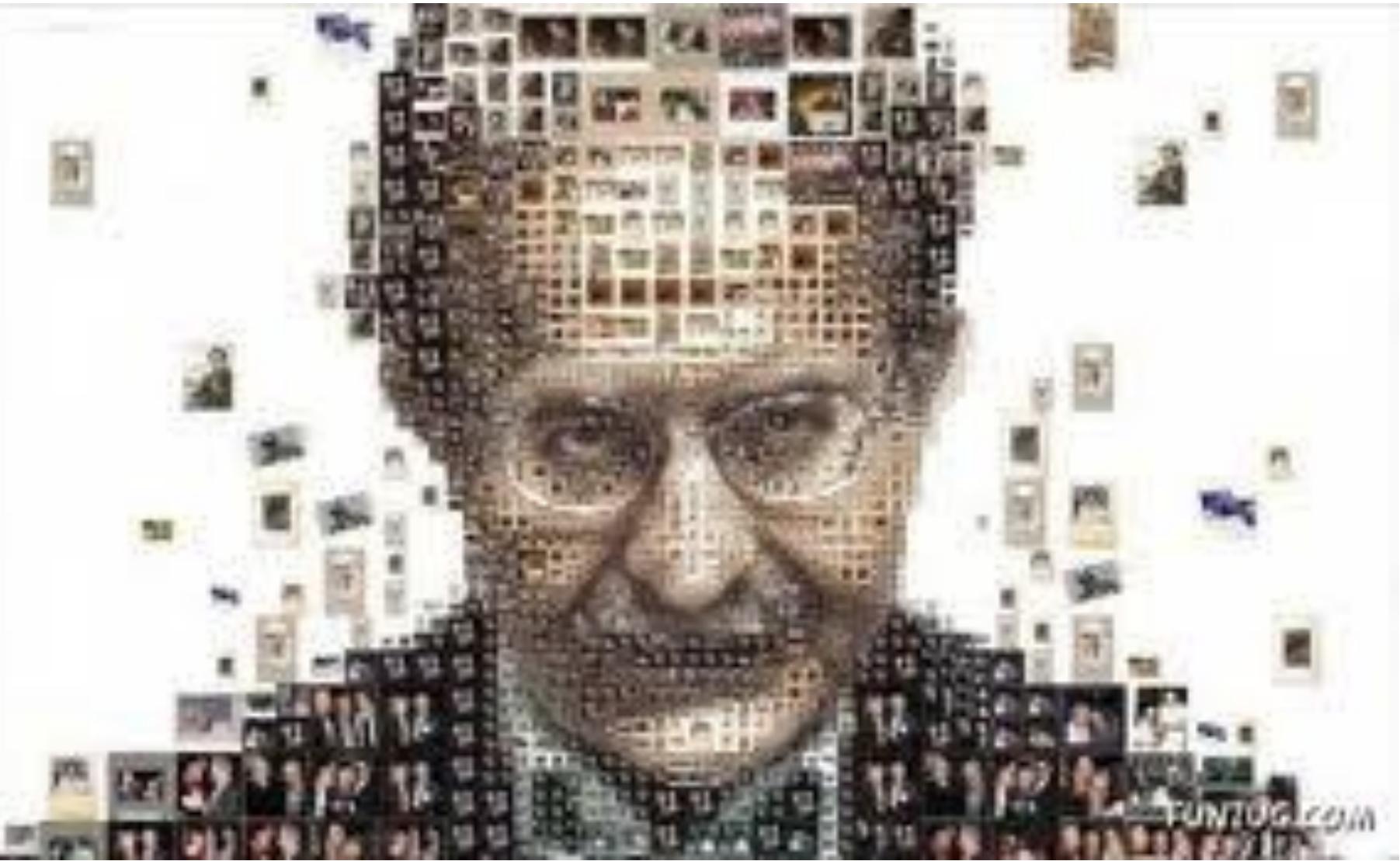
- with fewer bits, we cannot accurately represent the gradual intensity variations in the original scene because a **wider range of intensities** in the original scene is mapped into a **single gray level**
- generally: more bits → better brightness resolution

Dithering and halftoning

- Dithering and halftoning
 - used to render images and graphics with more **apparent colours** than are actually displayable
 - when the HVS is confronted with **large regions of high-frequency colour changes** they tend to blend the individual colours into uniform colour field
 - use this property of perception to represent colours that cannot be directly represented

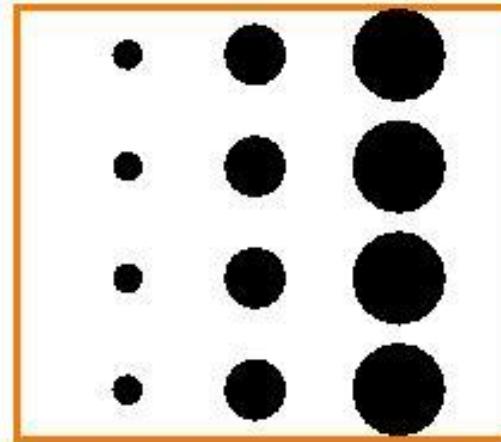


Artistic halftoning

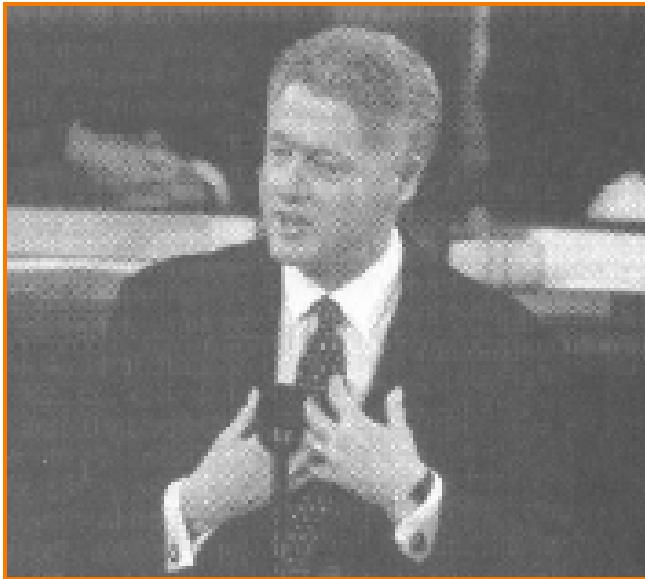


Classical halftoning

- Classical halftoning
 - uses **dots of various sizes** to represent intensity
 - used in newspapers and magazines



Example



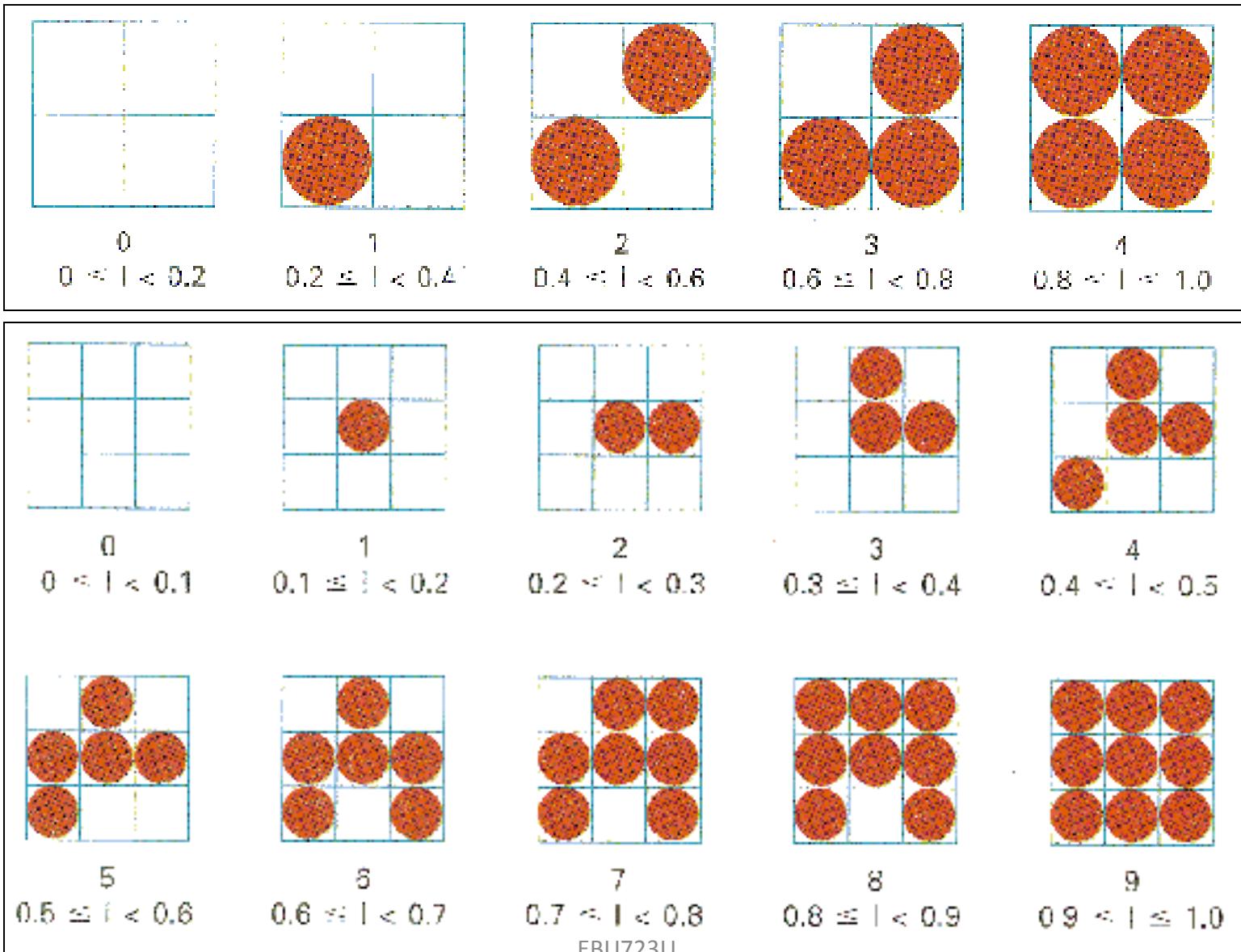
Newspaper Image



Halftoning examples



Halftoning with pixel patterns



Dithering

- HVS can discern ~100 brightness levels
 - depends on hue and ambient lighting
(e.g., we can see more distinct shades of green than blue)
 - True-colour displays
 - 256 colours available for each primary
 - usually adequate under normal indoor lighting
(when the nonlinearities of the display are properly compensated for)
 - usually no need to dither a true-colour display
 - High-colour displays
 - only 32 shades of each primary
 - HVS sees contours between two colours that vary by only one level
 - HVS even amplifies the variation!
 - This apparent amplification of contours is called Mach-band
- need dithering

Dithering

- Dithering

- process of **juxtaposing pixels of two colours** to create the illusion that a third colour is present
- largely used in printed media (newsprint, laser printers)

Original full-color photograph

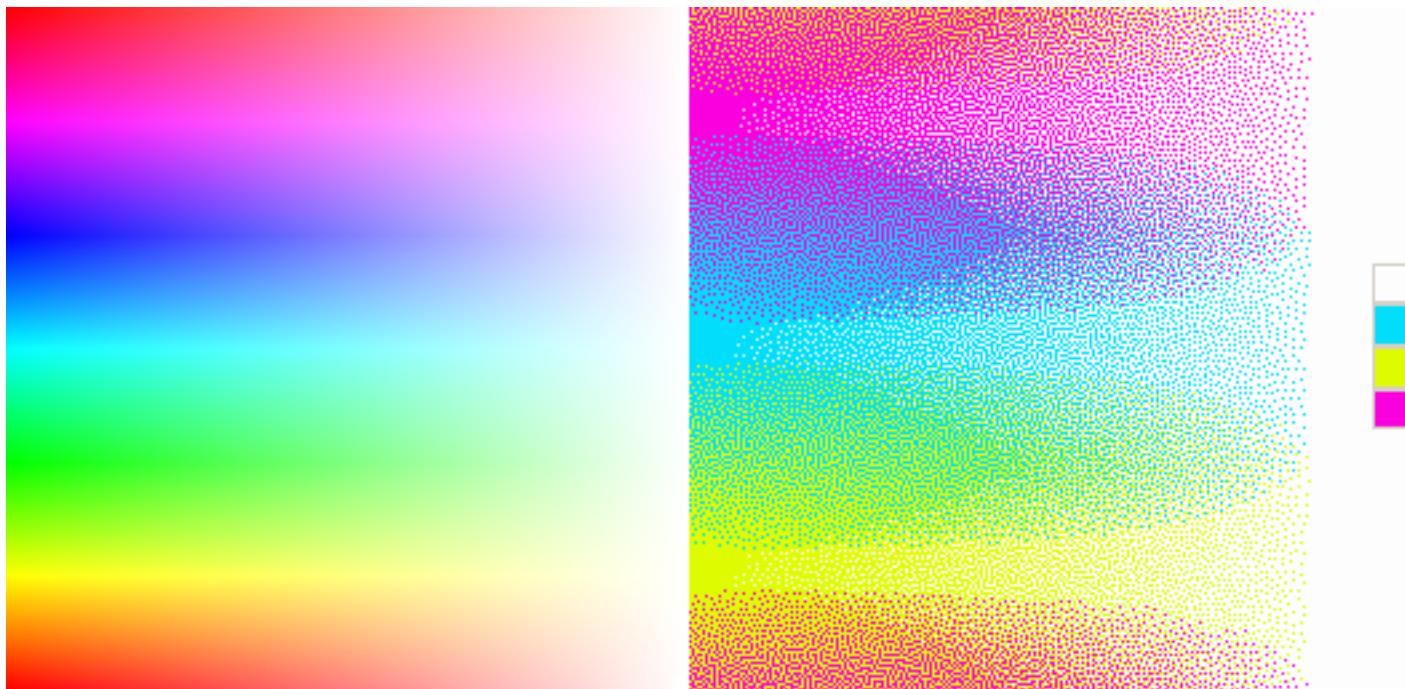


Dithered to 256 colors

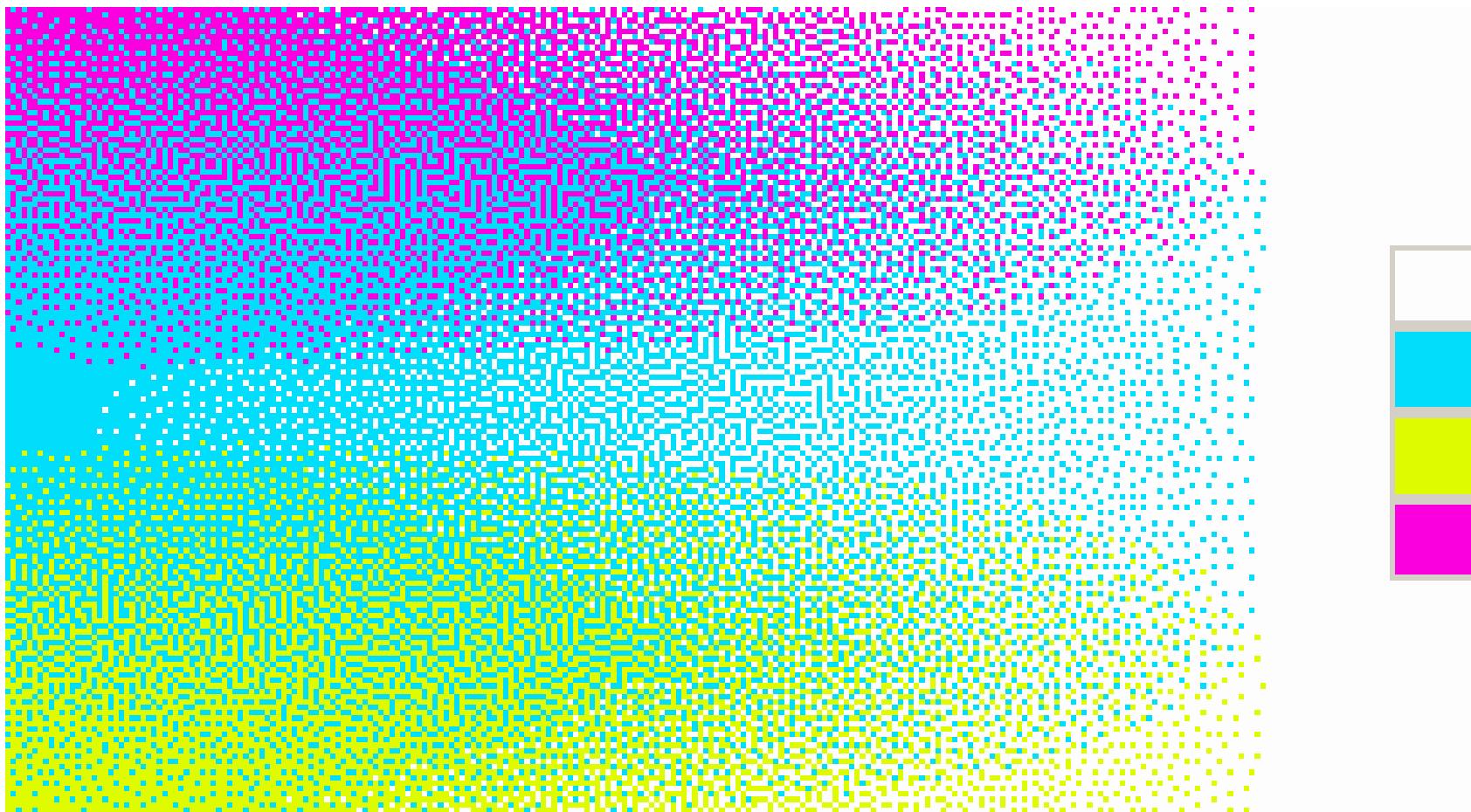


Another example

- Original image and four-colour version:



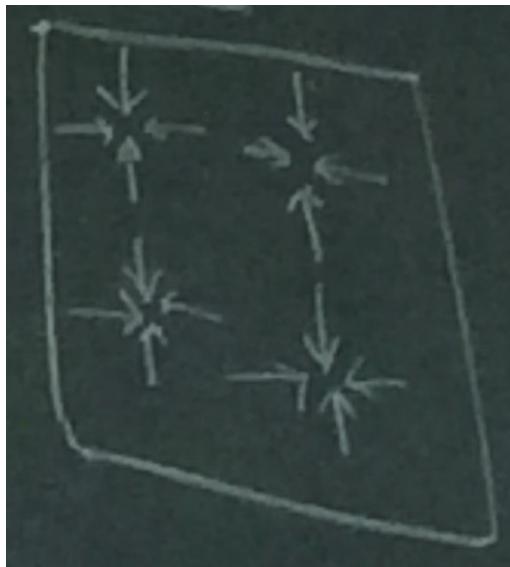
Zoomed



Dithering

- Dithering
 - requires the addition of **spatial offsets** to the original pixels
 - can be summarized as a **quantization process** where noise has been introduced to the input
 - The character of the dither is determined entirely by the structure of the noise
- This noise can be
 - **regular** (a repeated signal that is independent of either the input or output)
跟图像本身有关
 - **correlated** (a signal that is related to the input)
 - **random** 完全随机
 - or some combination

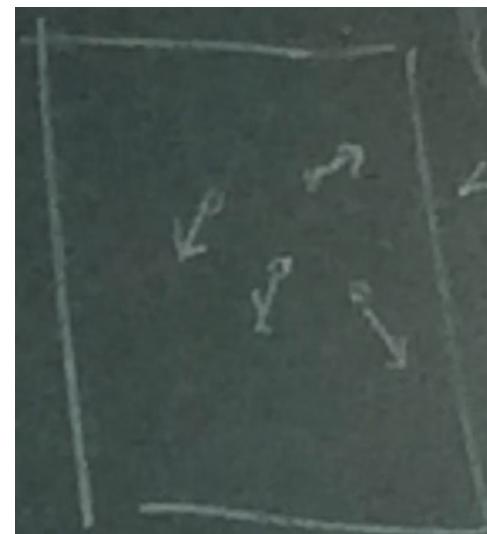
Note: Dithering decreases the SNR yet improves the perceived quality of the output



regular



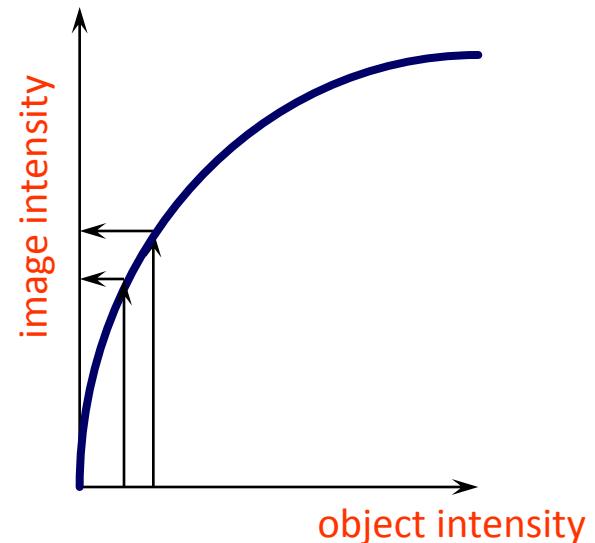
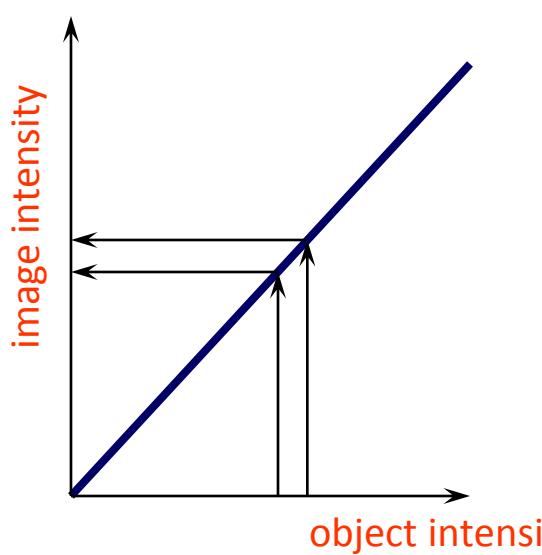
correlated



random

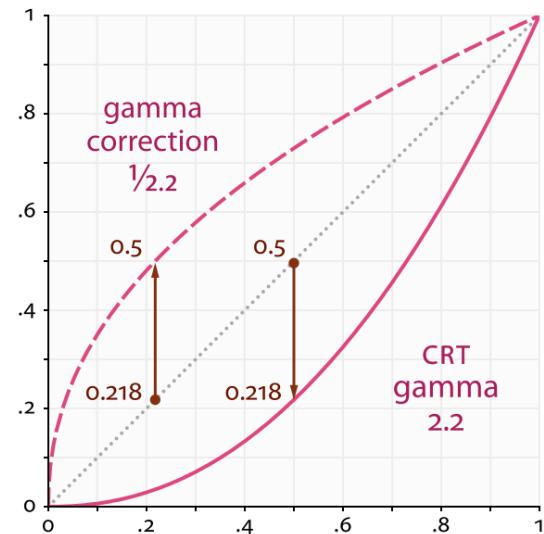
Quantization methods

- Uniform or linear
 - intensity of object is linearly mapped to gray-levels of image
- Logarithmic
 - higher intensity resolution in darker areas
(the human eye is logarithmic)



Gamma transformation

- Transform the intensity by power, gamma
- $f = f^\gamma$
- Emphasize dark regions if $\gamma < 1$, or emphasise bright regions if $\gamma > 1$
- Models human perception



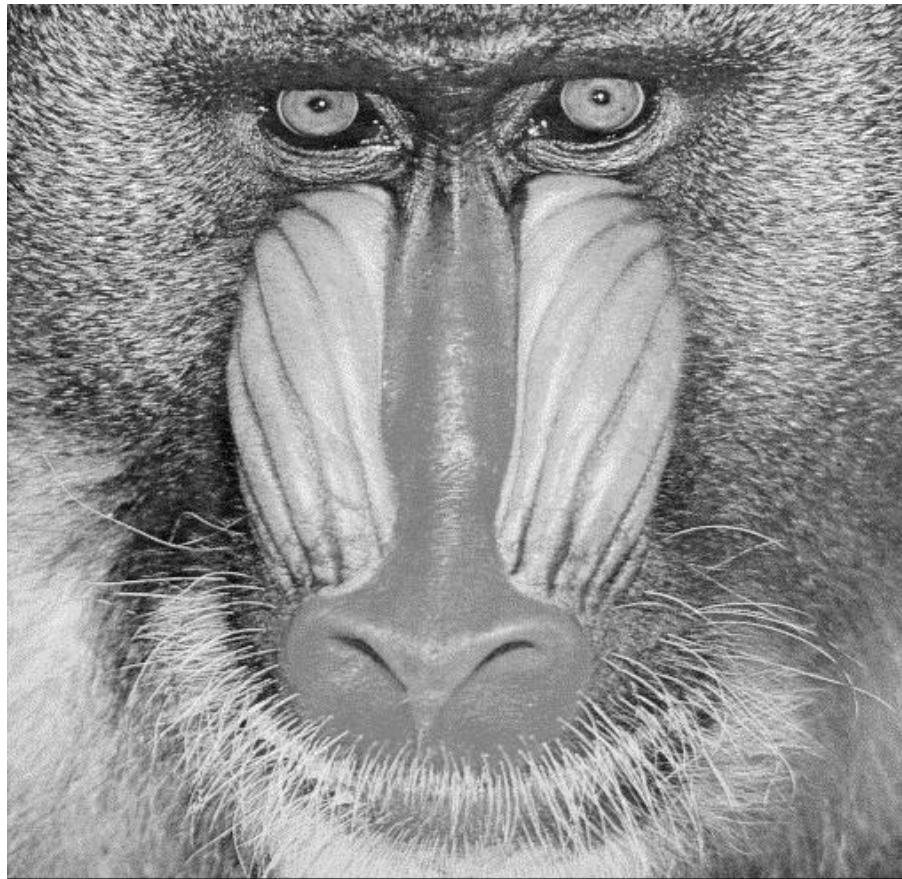
Common quantization levels

- $g(i,j)$ is given by integer values [0-max],
 $\text{max} = 2^n - 1$

$n=1$	-----	"binary image"
$n=8$	-----	1 byte, very common
$n=16$	-----	common in research
$n=24$	-----	common in color images (i.e. 3×8 for RGB)

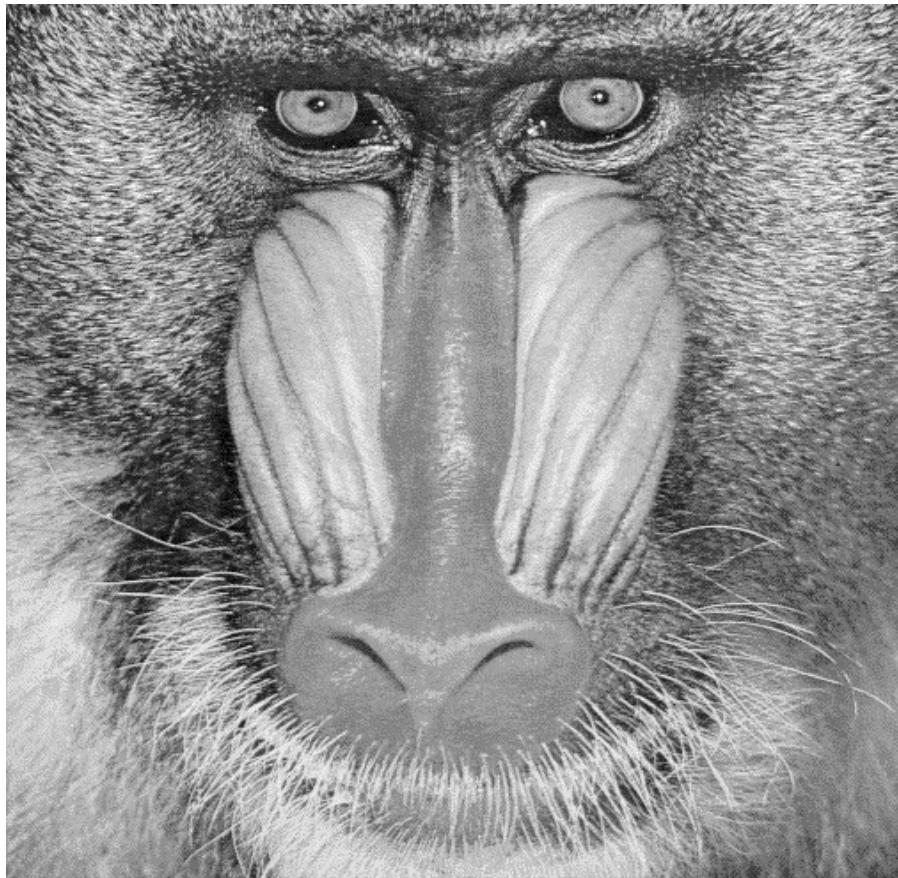
Gray-level quantization

256



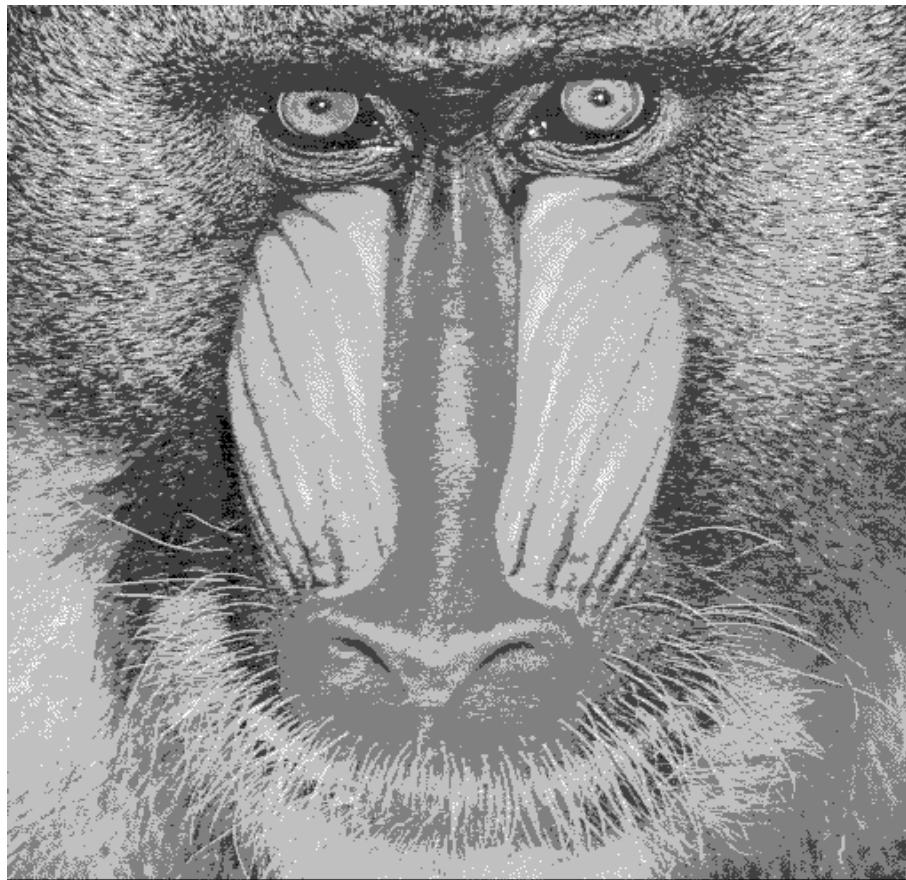
Gray-level quantization

32



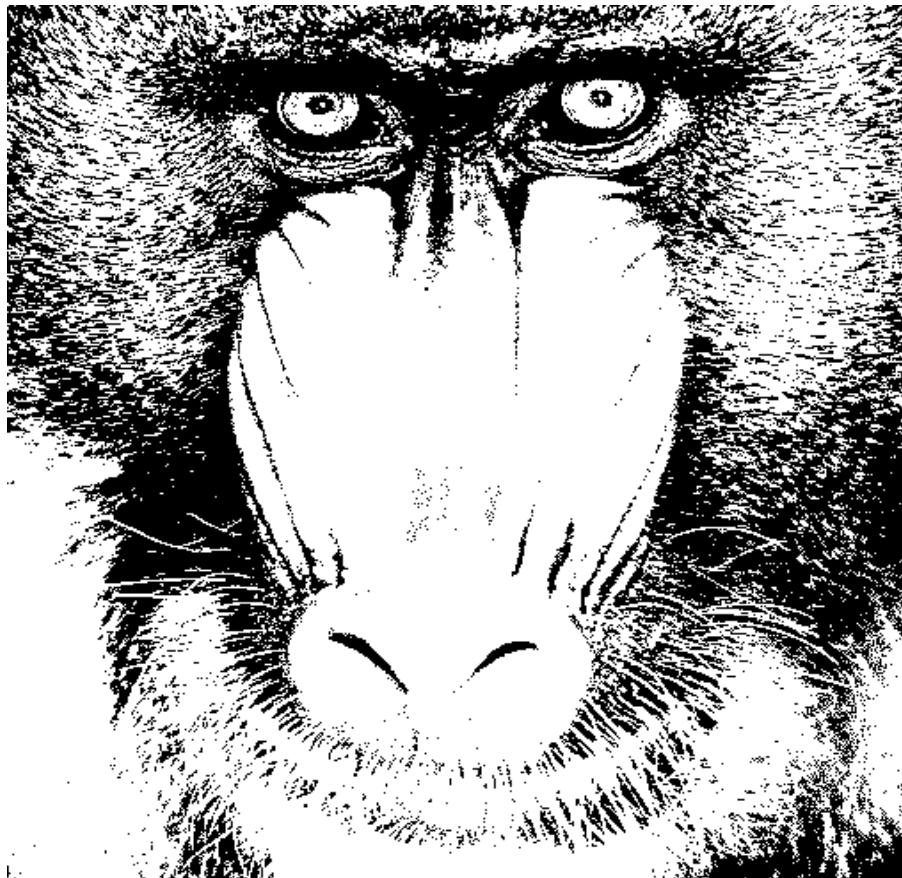
Gray-level quantization

8



Gray-level quantization

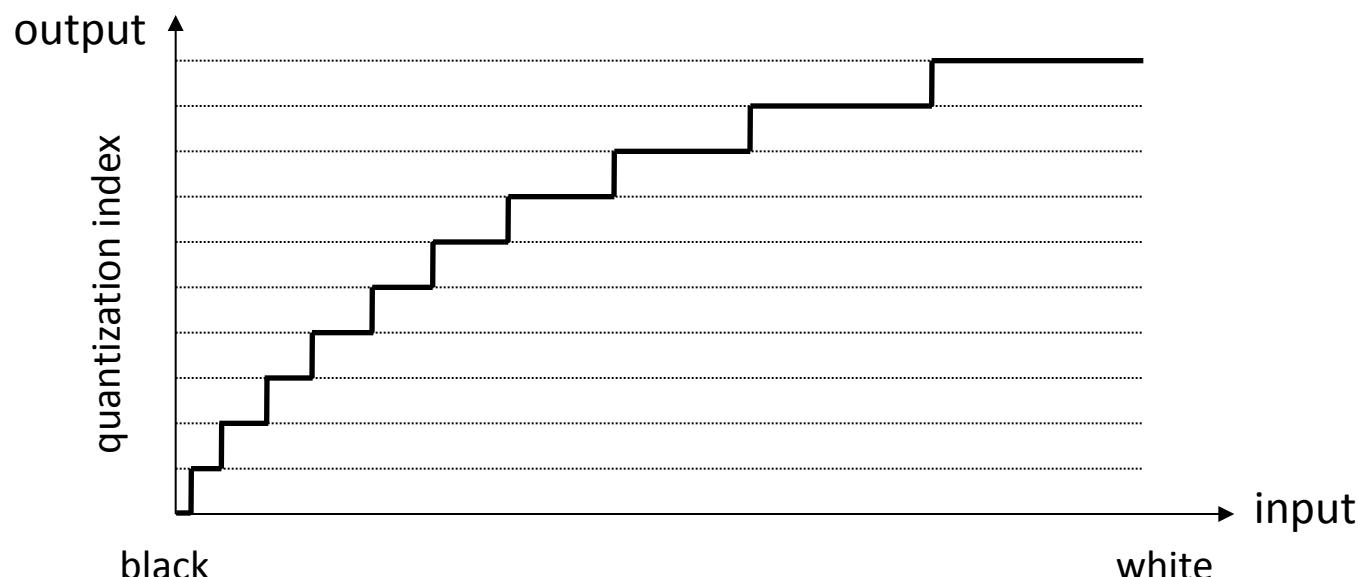
2



Non-uniform quantization

One reason: humans are more sensitive to variations in darker greyscale

- Non-uniform quantization
 - Better choice when probability density of a signal is not uniform
 - Allow to take into account the characteristics of the HVS

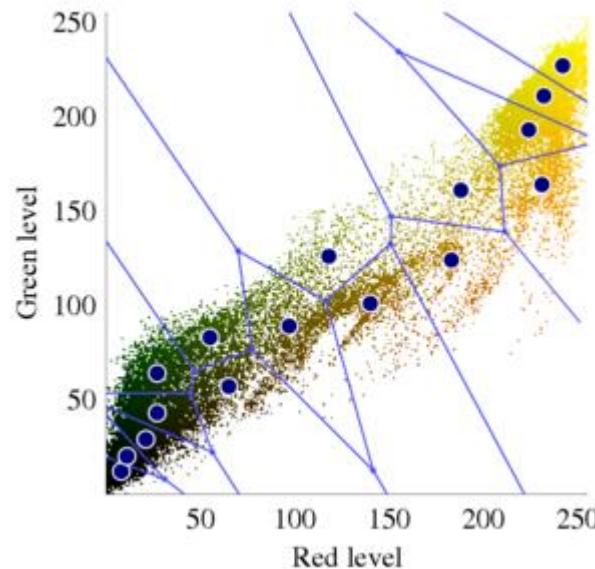


Quantizing colour images

- Each component can be quantized separately
- Easy, but not very satisfactory (see next slide)
- Some colour components can be
 - Quantized with different steps
 - Sampled with different steps
- Quantization of a colour image with a Look-Up Table (LUT)

Colour Quantization

original

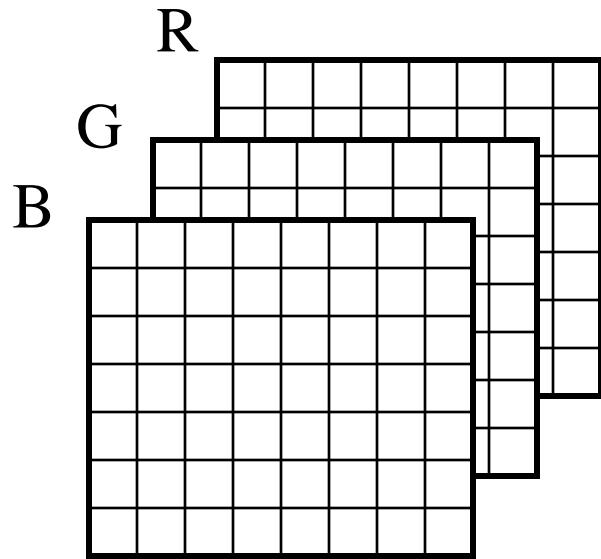


quantized

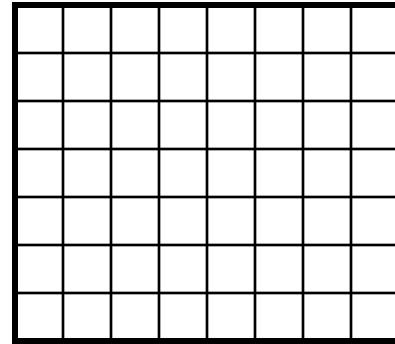


- Algorithm chooses a number of colours (sites) in the Red/Green plane
- Central plot shows Voronoi regions: all points closer to this site than any other

Look-up table (LUT)



True colours

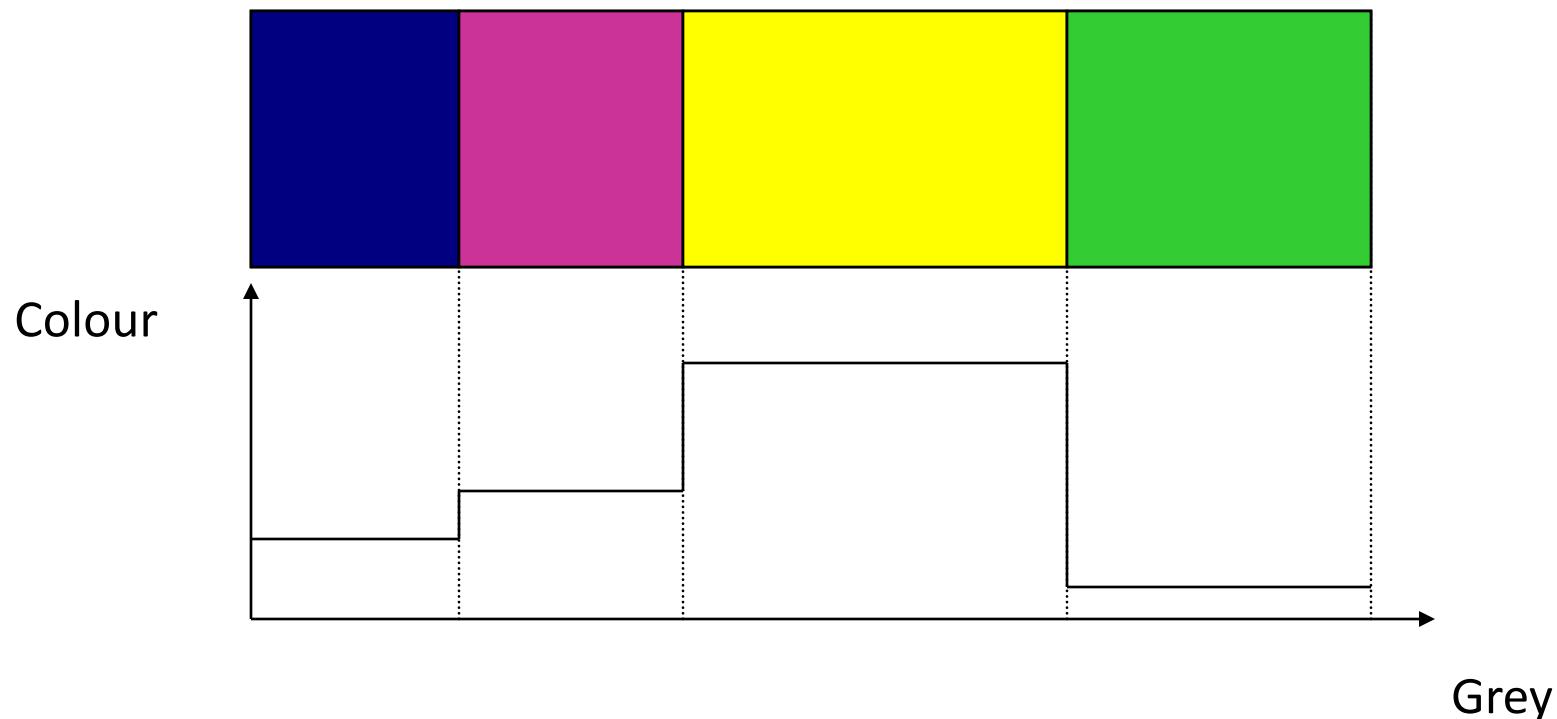


Look-up table

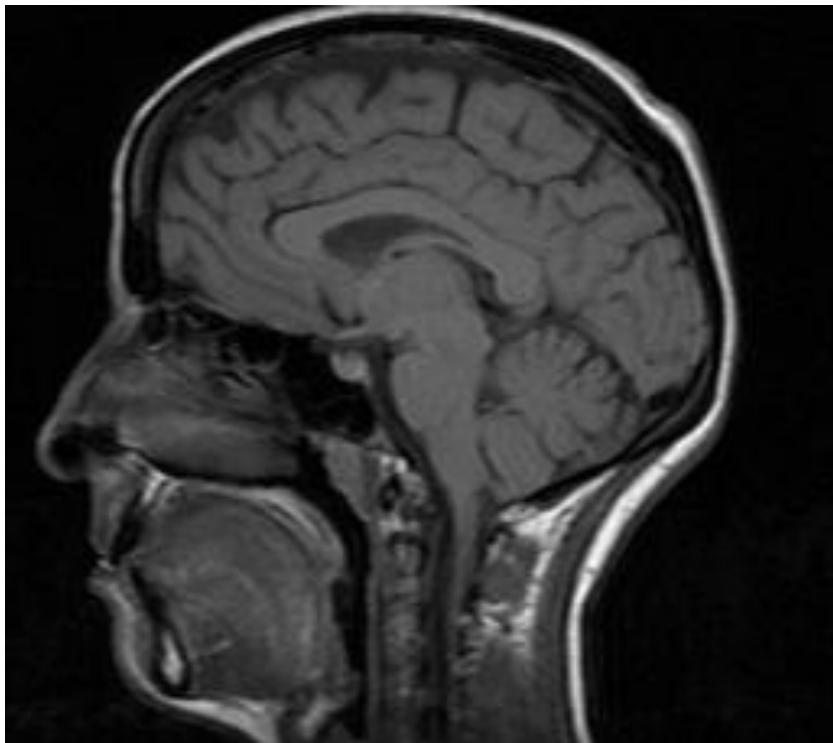
value	R	G	B
0	10	10	10
1	10	20	30
2	30	100	20
...

False colour images

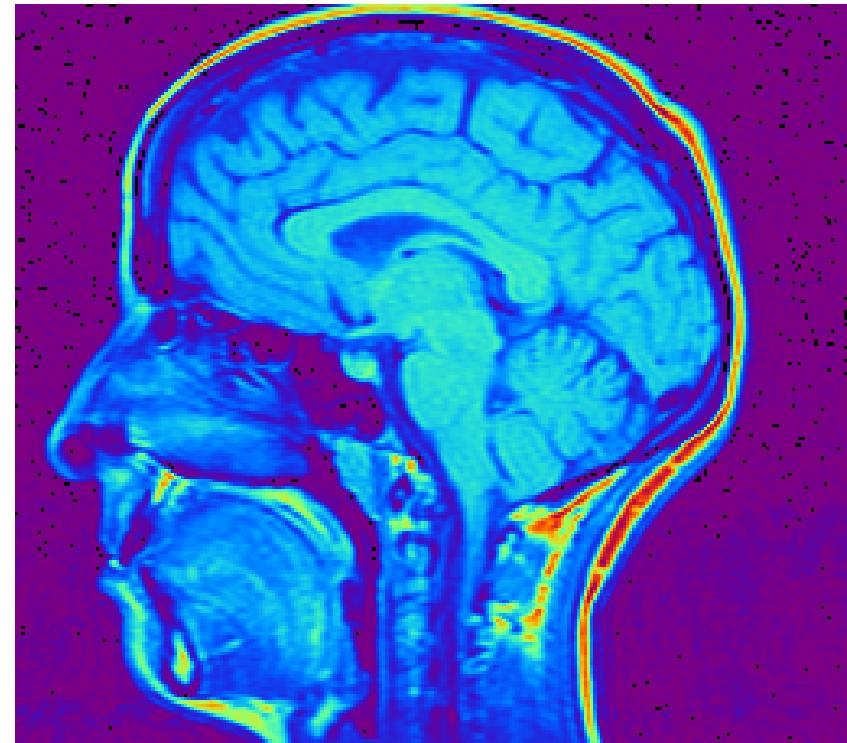
- A special look-up table ...



Example



original image



false colour image

Choice of sampling and quantization

- What will the image be **used for**?
- What are the limitations in **memory** and **speed**?
- Will the image be used for **visual** interpretation only or for any image analysis/processing?
- What **information** is relevant for the analysis (i.e. color, spatial and/or gray-level resolution)?

Old BBC TV test-card



- Note the grey-level steps, straight lines, and fine-stripes.

Zooming and shrinking

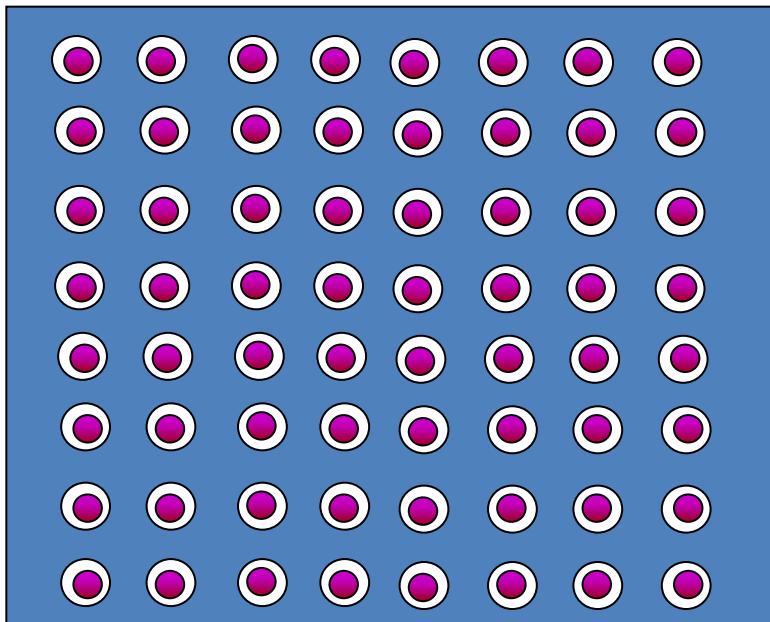
- Zooming
 - Can be seen as **over sampling**
 - *Creation of new pixel locations*
 - Assignment of grey level to those locations
 - Pixel replication (NN)
 - Bilinear interpolation
- Shrinking
 - Can be seen as **under sampling**

Today's agenda

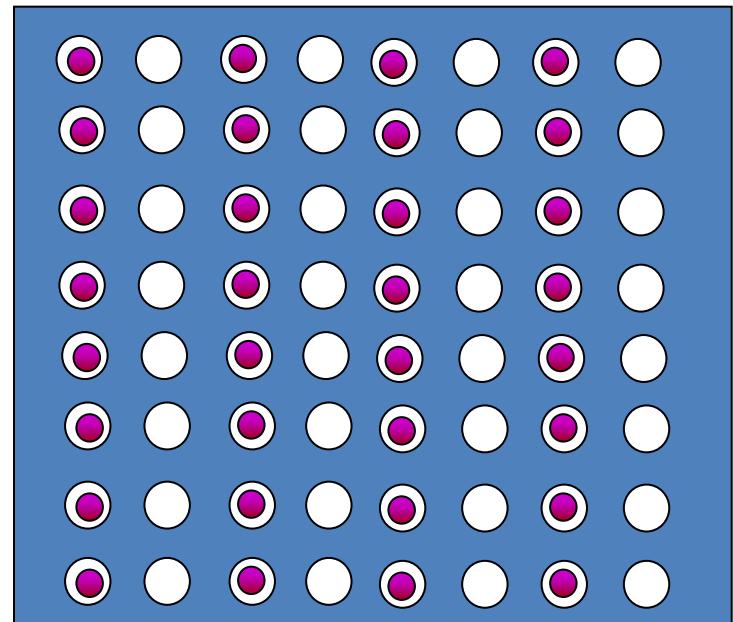
- Digital image representation
- Sampling
- Quantization
- Sub-sampling
- Pixel interpolation

Sub-sampling

Pixels are removed according to a given pattern

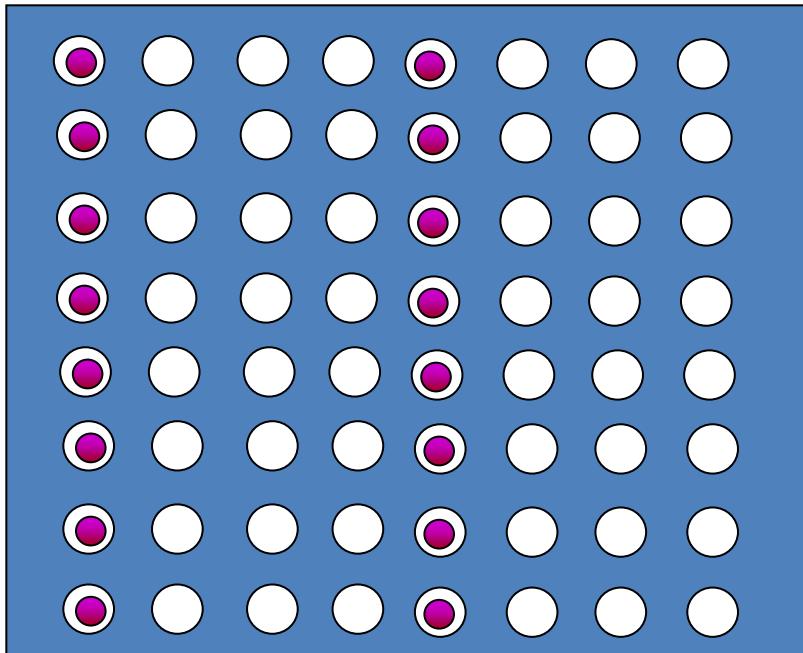


original sampling

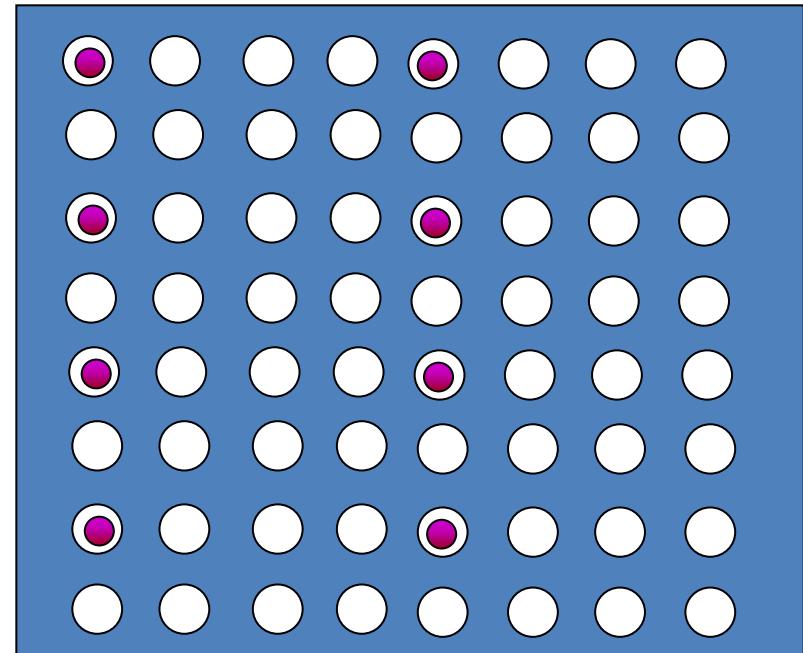


2:1 subsampling

Sub-sampling

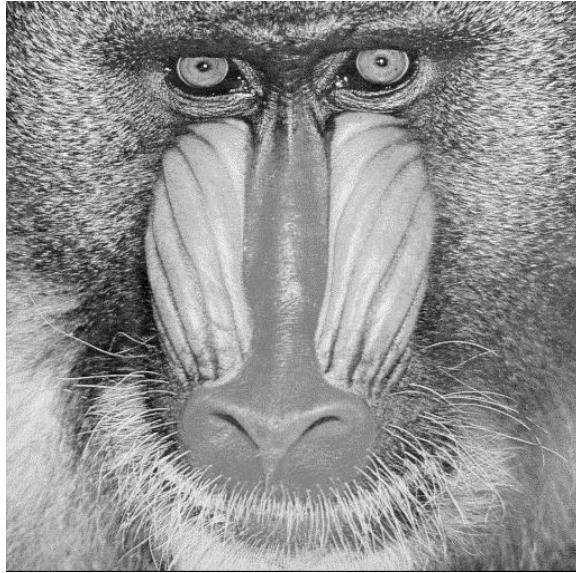


4:1 subsampling

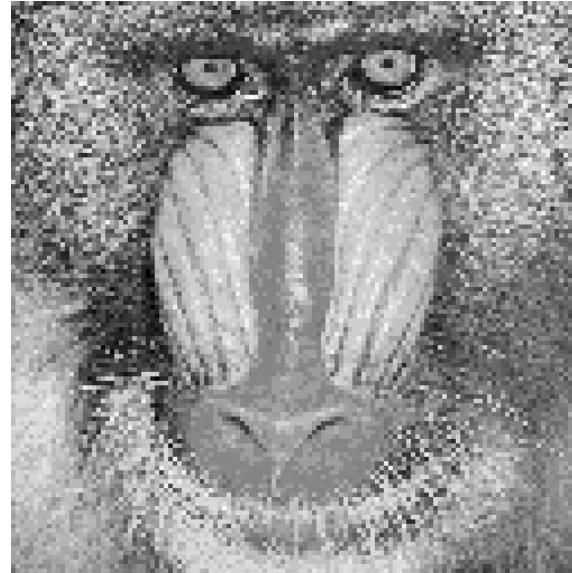


8:1 subsampling

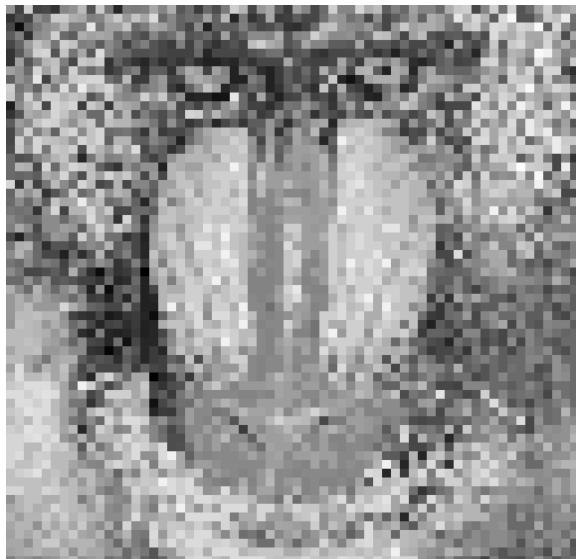
Image sub-sampling



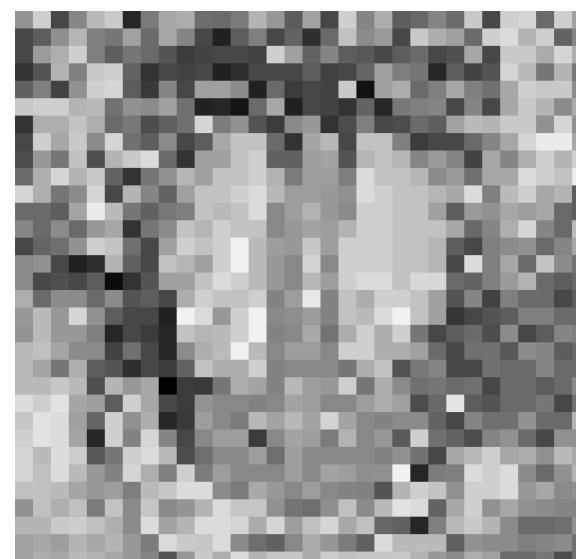
original
sampling
512x512



128x128



64x64



32x32

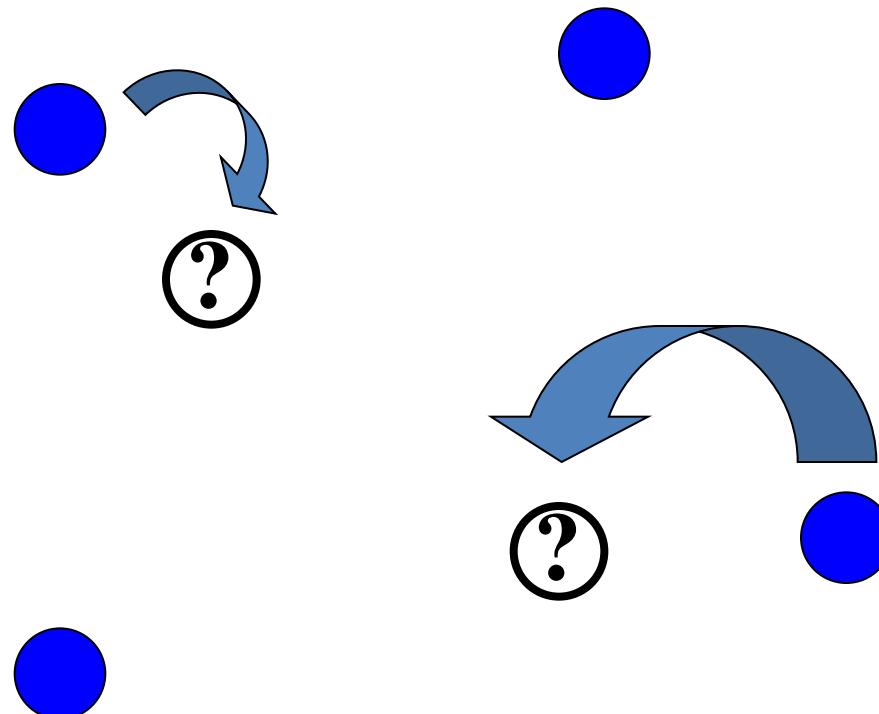
Up-sampling

- Objective
 - to increase the **resolution**
- Procedure
 - requires generation of **additional pixels** from available ones
 - usually geometric transforms require interpolation
- Example
 - the simplest form is approximation by the **nearest** available pixel



Nearest neighbour interpolation

- Pixels are generated by *copying* the nearest available pixel

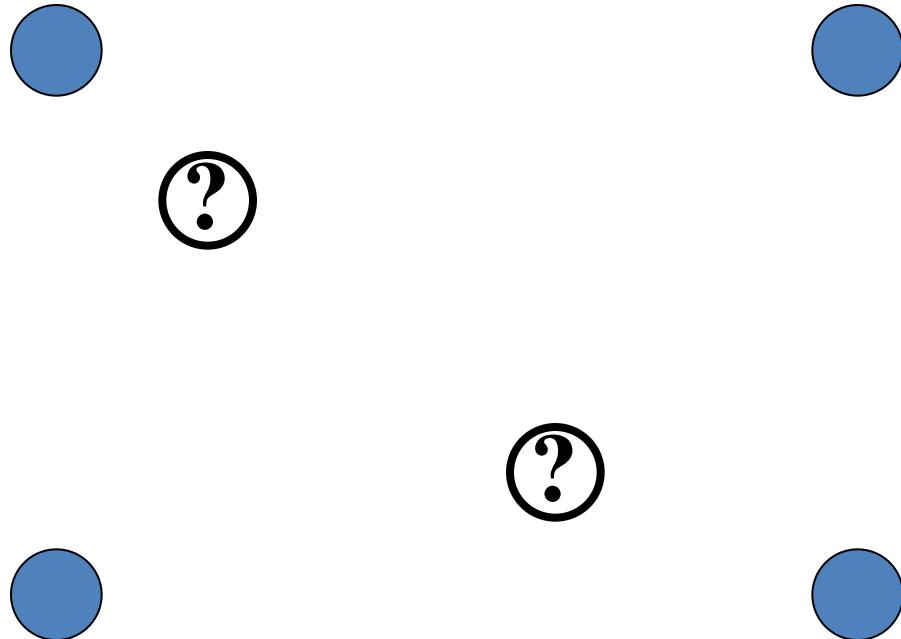


General interpolation

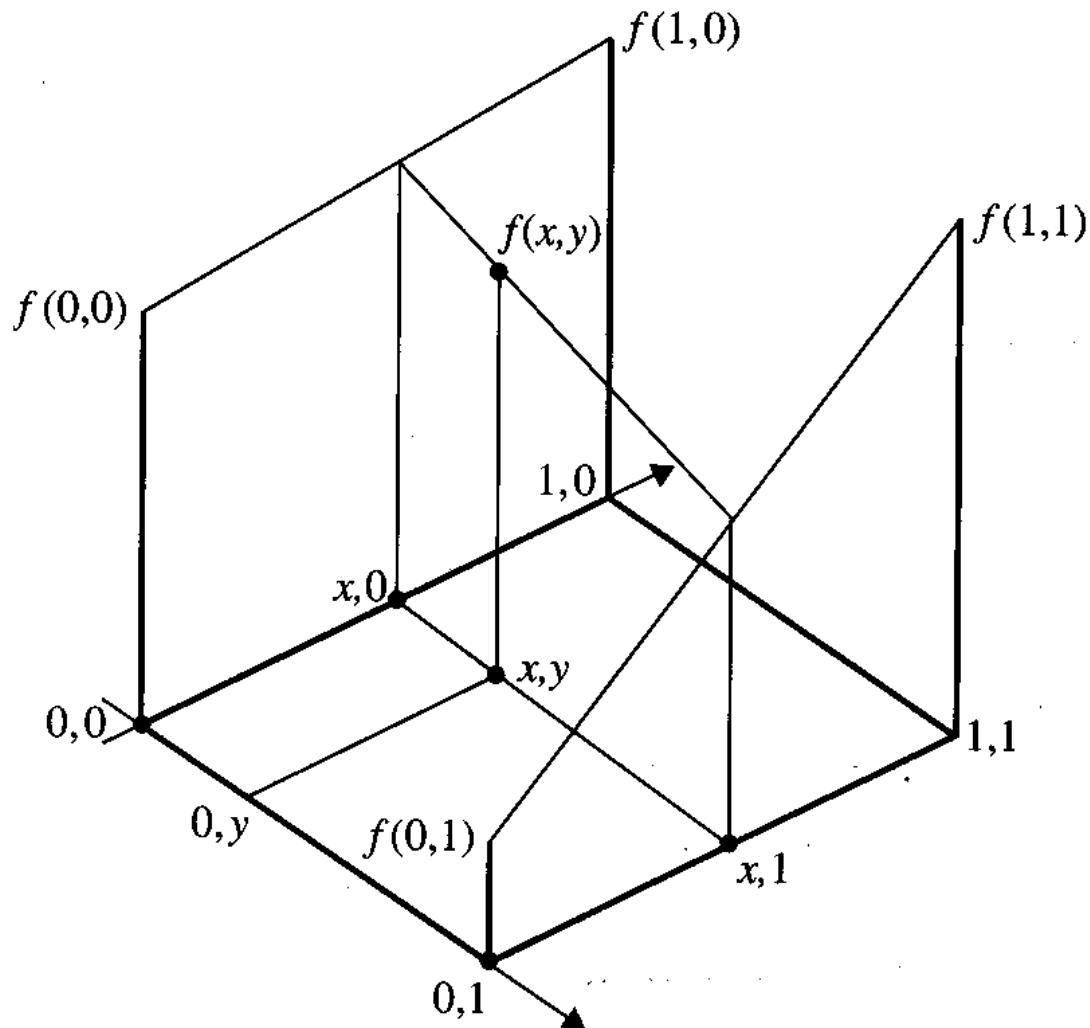
- Forward mapping
 - map the input grid to **non-integer**-located positions in the output image
 - **interpolate** to obtain pixels on the output image
 - problem → **redundant** conversions
(adjacent output pixels may use same data)
- Backward interpolation
 - estimate **integer** output grid → inverse map integer output grid
 - estimate pixels at input (continuous image by interpolation)
 - preferred method
 - it only converts data required for generating output pixels

Bilinear interpolation

- Assumes 4 pixels on a regular grid are known
- Pixels *inside* can be interpolated



Bilinear interpolation

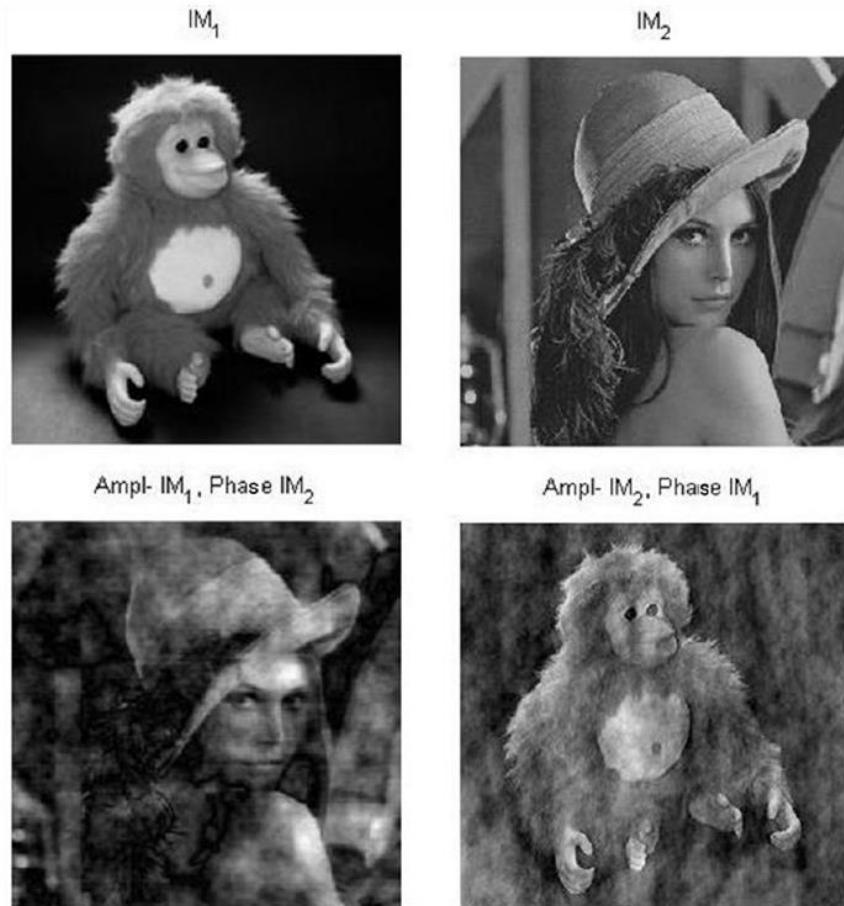


Bilinear interpolation

- Mathematical solution is over-determined using a plane (only 3 points needed)
- Bilinear
 - use the simplest second order form
$$f(x, y) = ax + by + cxy + d$$
 - interpolate *linear on each horizontal edge*
 - interpolate *linear in the vertical direction using obtained results*

Fourier information

- Fourier transform splits the image into Amplitude and Phase spectra
- Interesting experiment, using two images...
- Compute (A_1, P_1) and (A_2, P_2)
- Reconstruct from (A_1, P_2) and (A_2, P_1)
- **Phase is perceptually more important than amplitude.**



Example from Skarbnik et al.

What did we learn today?

- Digital image representation
- Sampling
- Quantization
- Sub-sampling
- Pixel interpolation

Image and Video Processing

(EBU723U)

Image histograms

Dr. Miles Hansard
miles.hansard@qmul.ac.uk

Today's agenda

- Histograms
 - definition
 - properties
 - colour histograms
 - thresholding
 - segmentation
 - equalization

Today's agenda

- Histograms
 - definition
 - properties
 - colour histograms
 - thresholding
 - segmentation
 - equalization

Definition

- Histogram
 - function that maps the **quantization levels** into the **frequency** of each quantization level in the image
 - Note that ‘frequency’ just means ‘count’, NOT spatial frequency!
 - ‘counts’ how many times each grey level appears in the whole image
- Grey-scale histograms
 - Number of pixels at *each* grey-level or in a *range* of grey levels
 - **Plot of frequencies** of grey levels as function of pixel value
 - Statistical equivalent to the Probability Density Function (**pdf**)

Example

P2

16 8

8

0000000000000000

0333300777700111

0300000700000100

0333000777000111

0300000700000100

0300000777700111

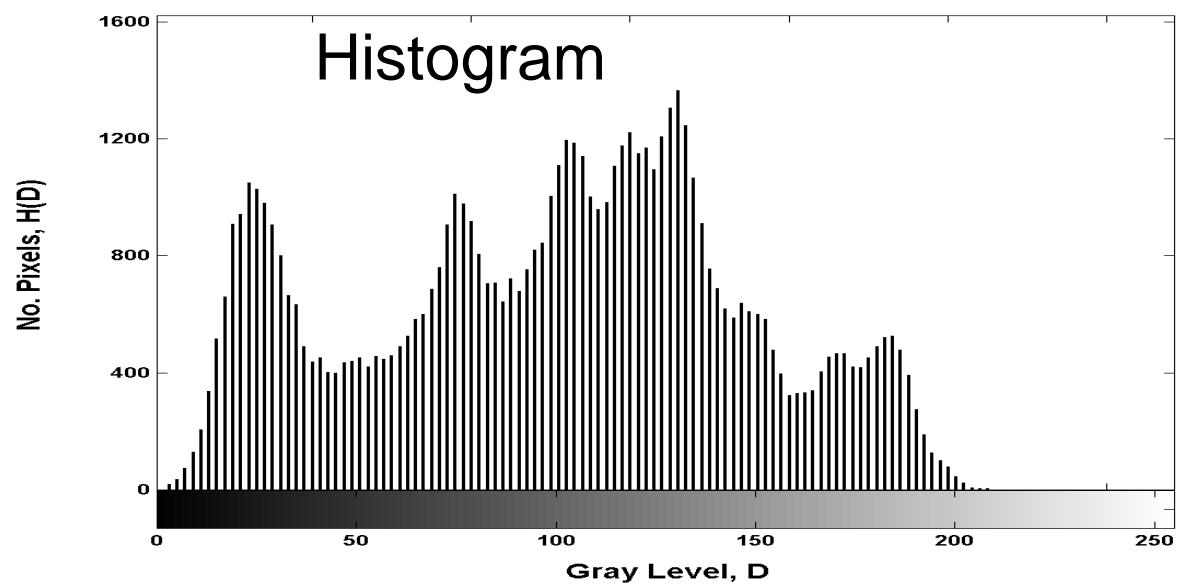
0000440000000000

0000000011233001

Frequencies

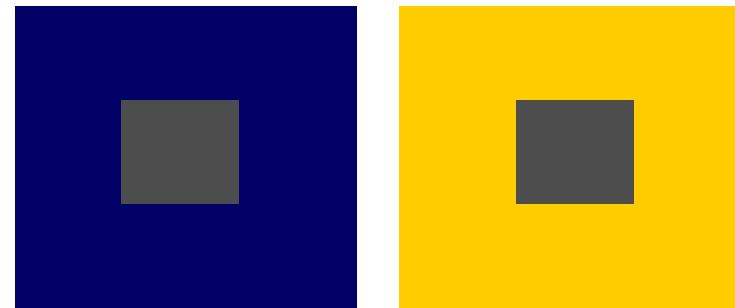
86
14
1
12
2
0
0
13

Example



Encapsulated information

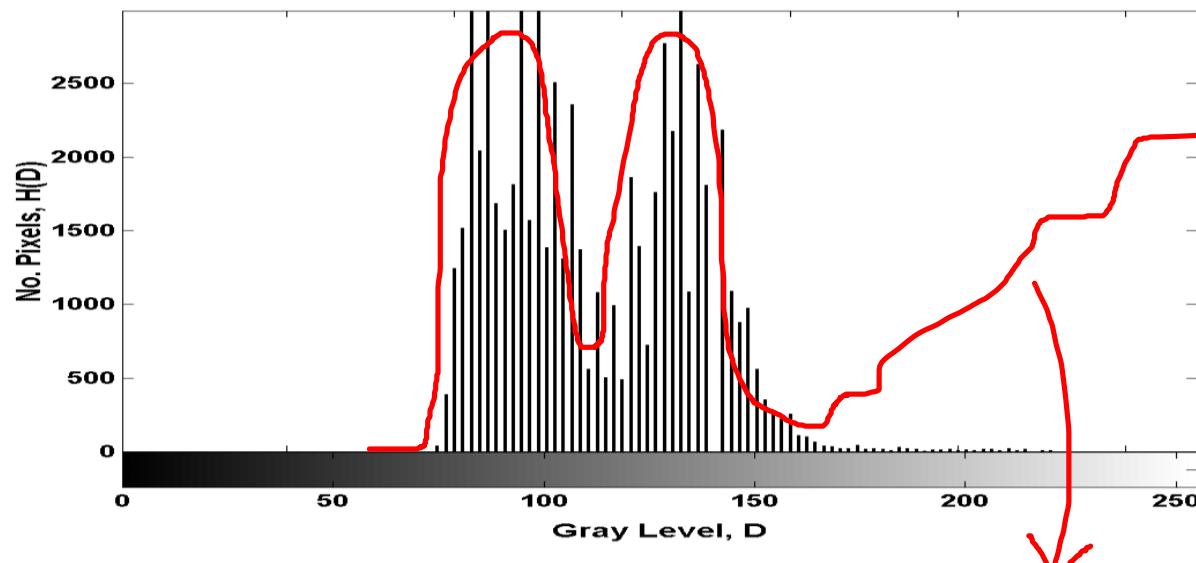
- Contrast
 - Intuitively: how *vivid or washed-out* an image/object appears
 - Amount of difference between **average** grey level of an object and that of surroundings
 - Example. The grey squares are identical, but the background contrast is different



- Brightness
 - **brightness range**: brightness span of the grey scale of an image
 - dynamic range of an 8-bit image → 256 grey levels

Example

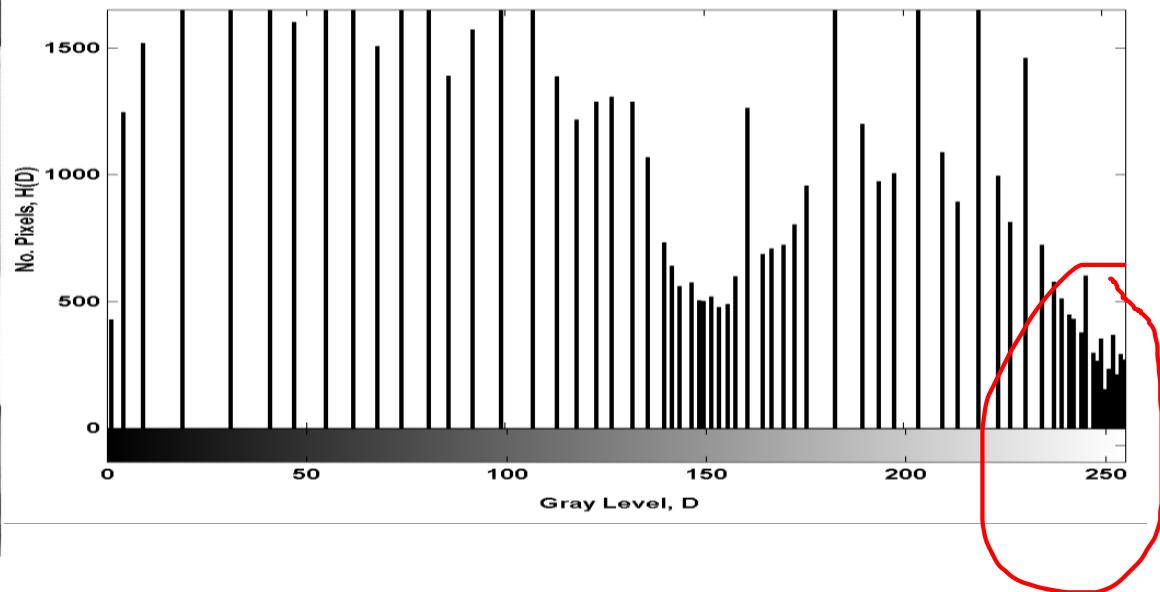
Low contrast & low brightness range



clipped

Example

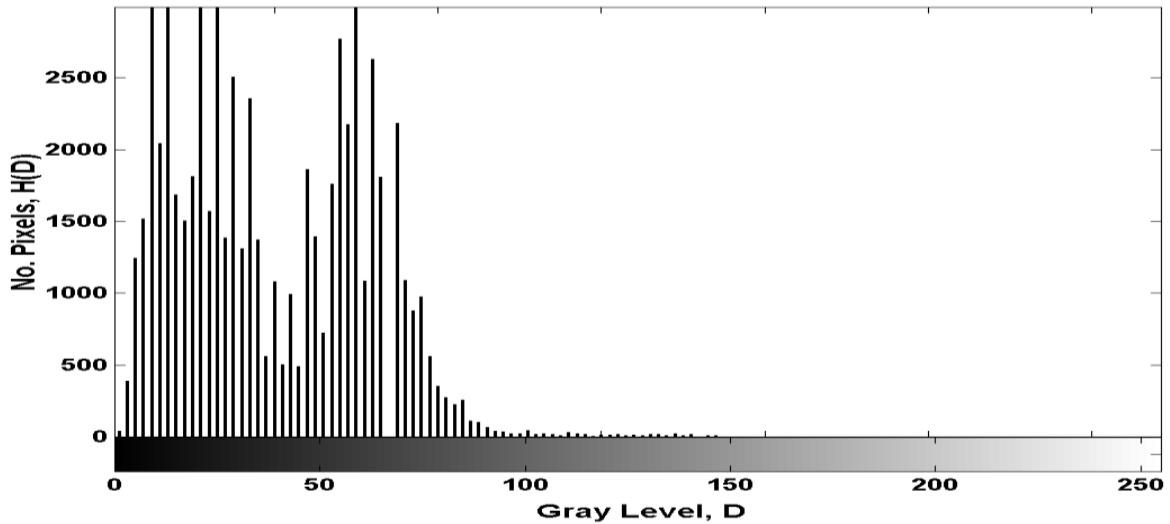
Higher contrast & brightness range



saturated

Example

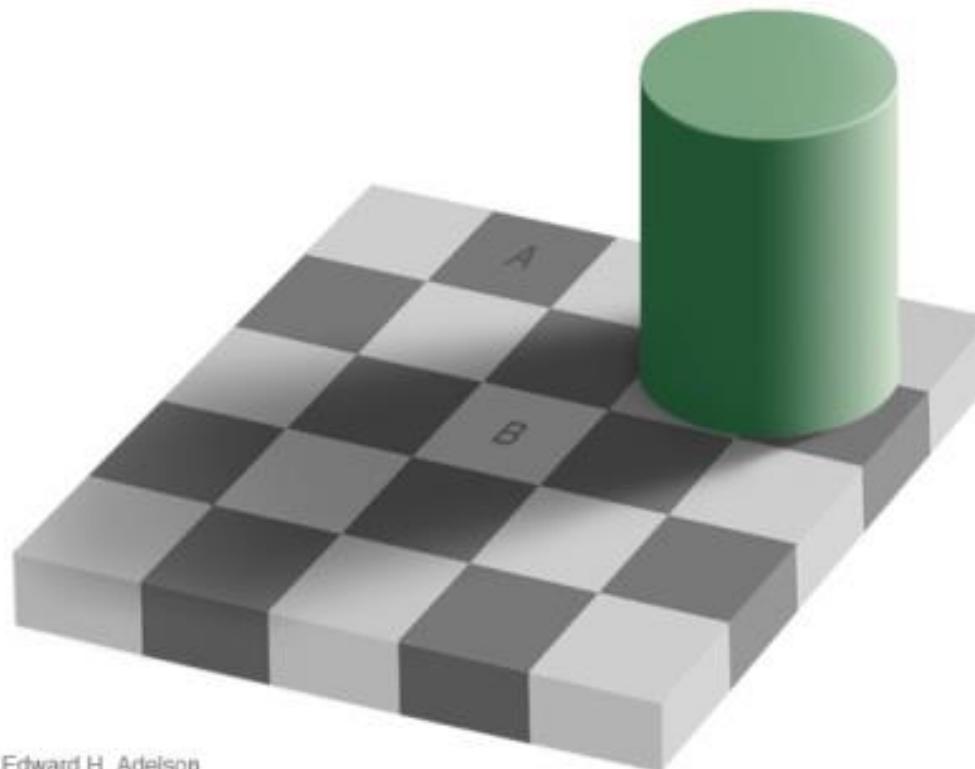
Dark image



Brightness perception

- Which square is darker, A or B?

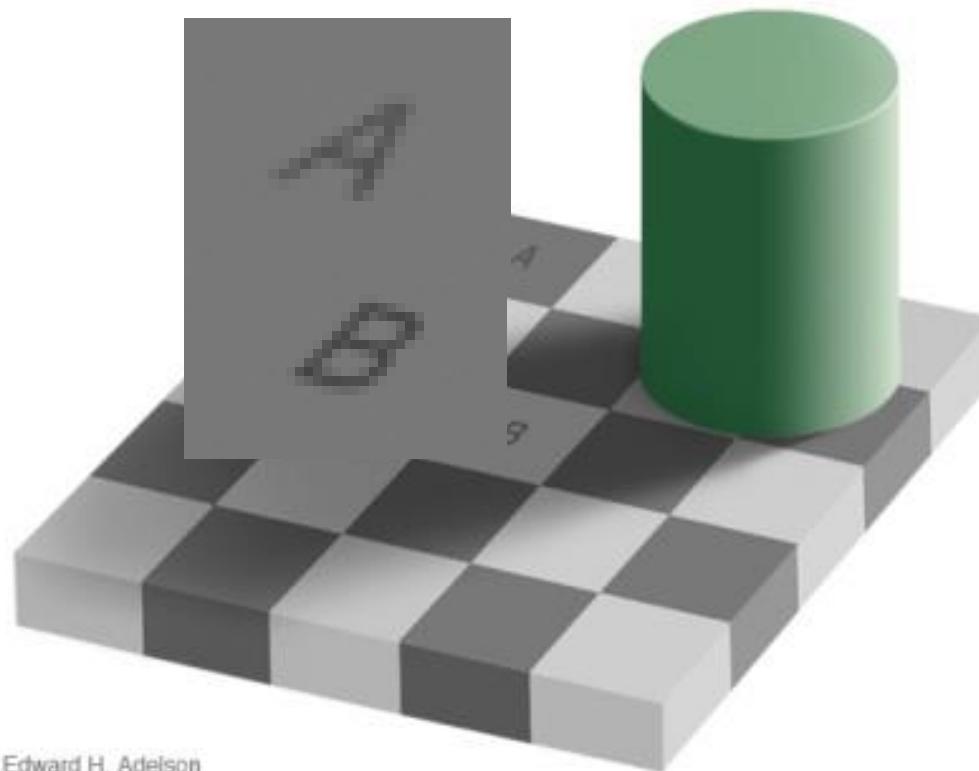
Photometric



Edward H. Adelson

Brightness perception

- They are the same! (Adelson brightness illusion)



Edward H. Adelson

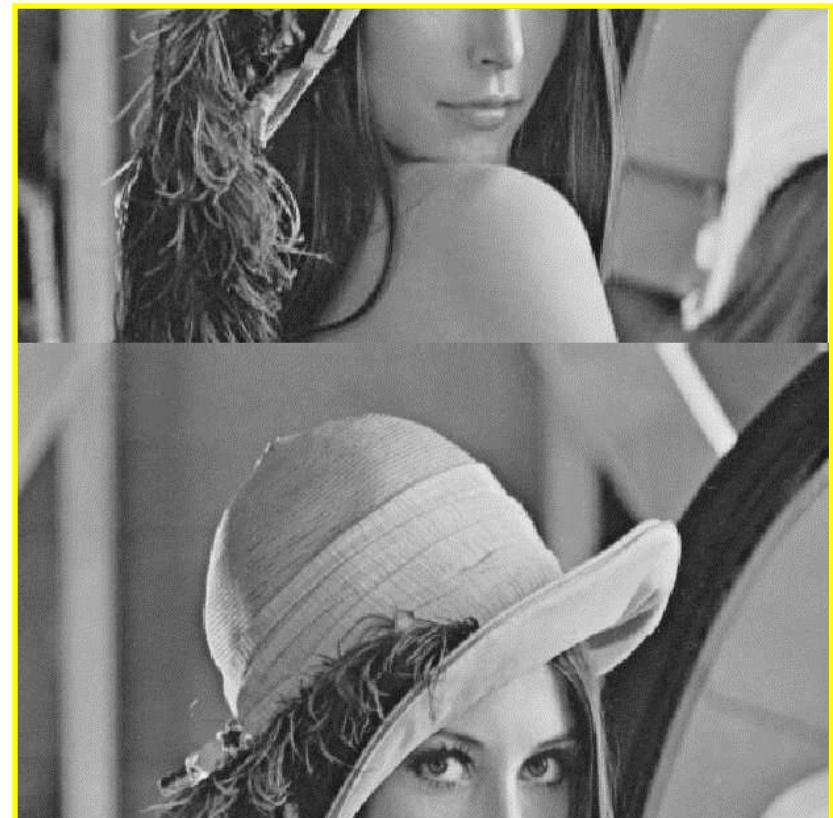
Today's agenda

- Histograms
 - definition
 - properties
 - colour histograms
 - thresholding
 - segmentation
 - equalization

Histograms of different images

- What is the relationship between the histograms of these two images?

the same. Histograms do not contain spatial information about the image.



Histograms of different images



These images have the same colour histogram!

Histogram properties

- Histogram is a **many-to-one mapping**
- Different images can have the same histogram
→ Non invertible mapping!

c.f. Fourier transforms are invertible



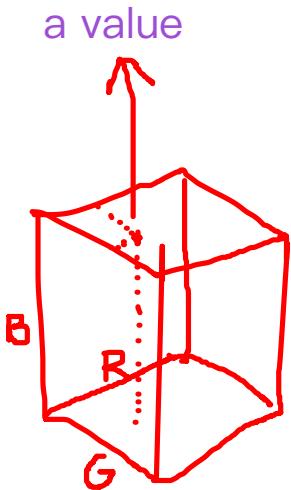
Histogram properties

- Histogram is **invariant** to certain geometric image operations
 - Rotation
 - Scaling 虽数值变大，但最后会normalise
 - Mirroring
 - Skew, ...

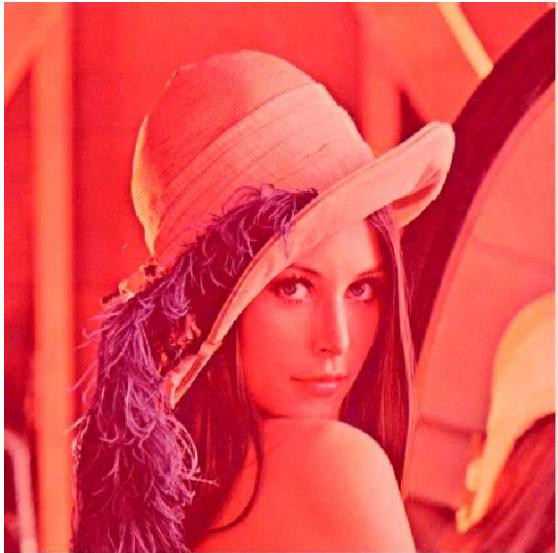
Today's agenda

- Histograms
 - definition
 - properties
 - colour histograms
 - thresholding
 - segmentation
 - equalization

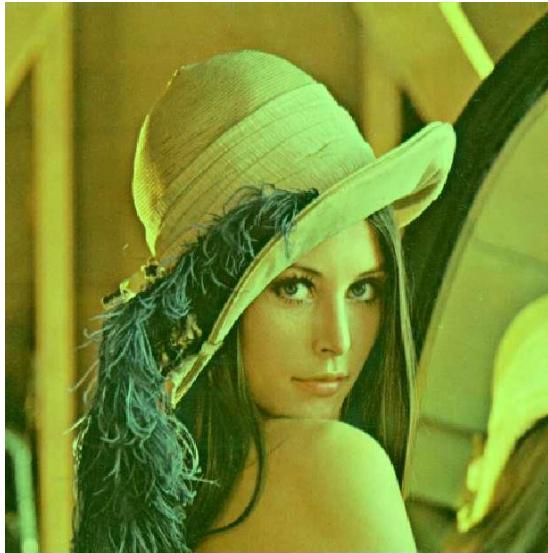
Colour image



Colour histograms



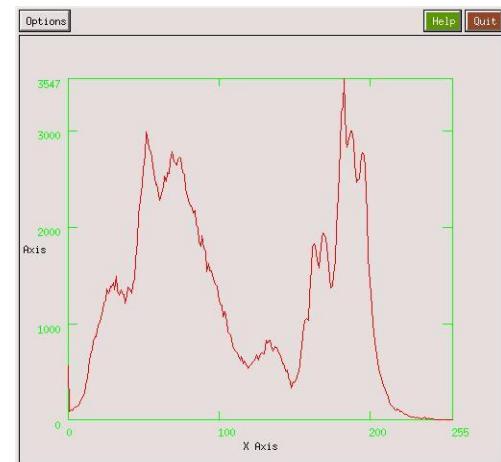
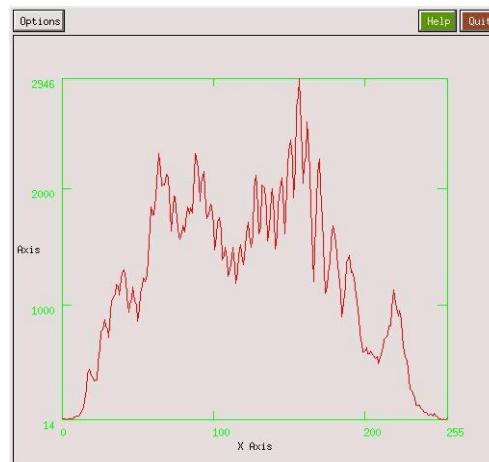
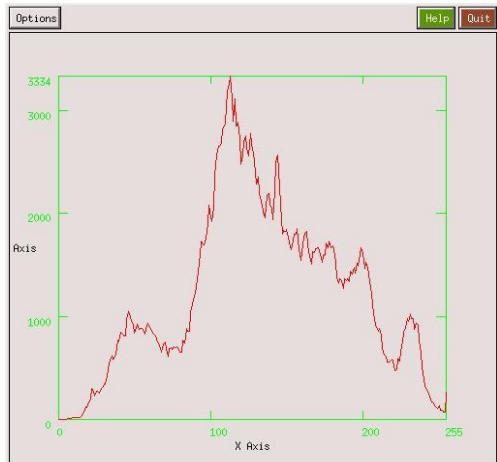
red



green



blue



... 3 x 1D or 1 x 3D ?

Histograms: summary

- Histograms
 - store the grey level / colour distribution
 - can be used to **manipulate images**
 - to change contrast
 - to change grey level range
 - does not provide us with spatial information
 - two different images may have the same histogram
- Applications
 - Histogram provides us with a quick indication as to whether or not the image covers the whole brightness range of digitizer
 - Histogram are a powerful tool for **image segmentation**
 - e.g, for separating objects from background ('background subtraction')

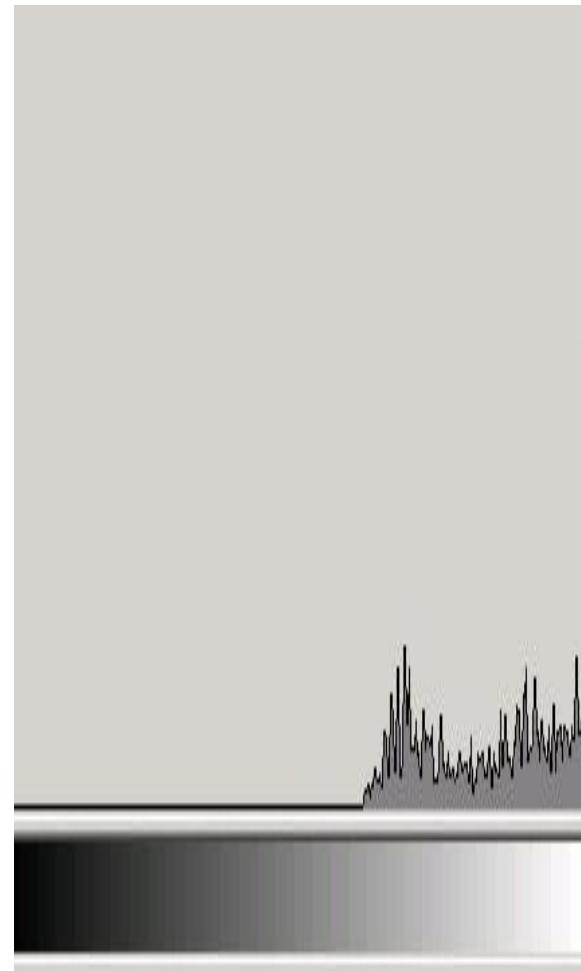
Image manipulation



adding 128 & clipping

clip掉>255的值

EBU723U

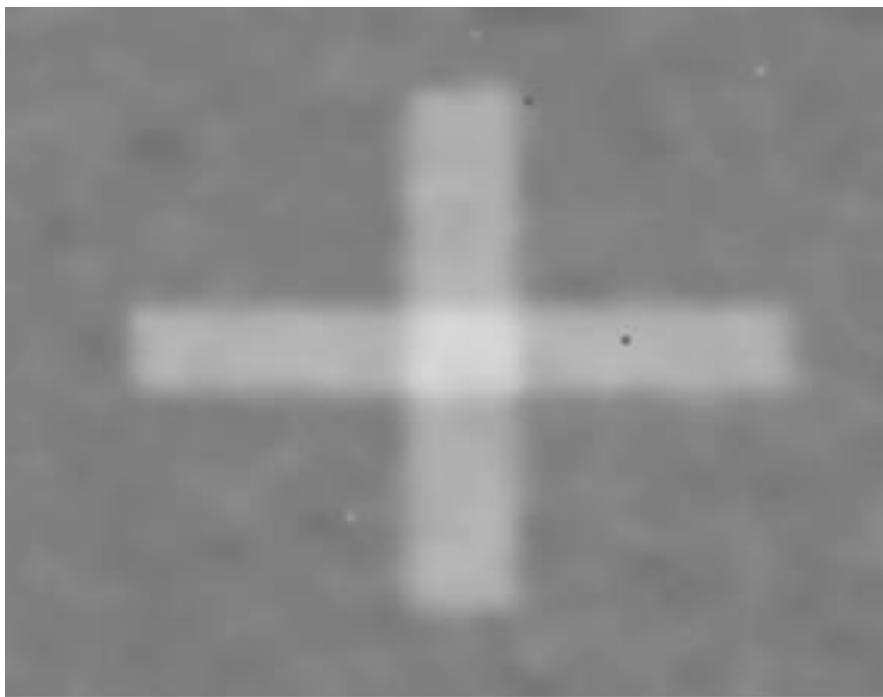


Today's agenda

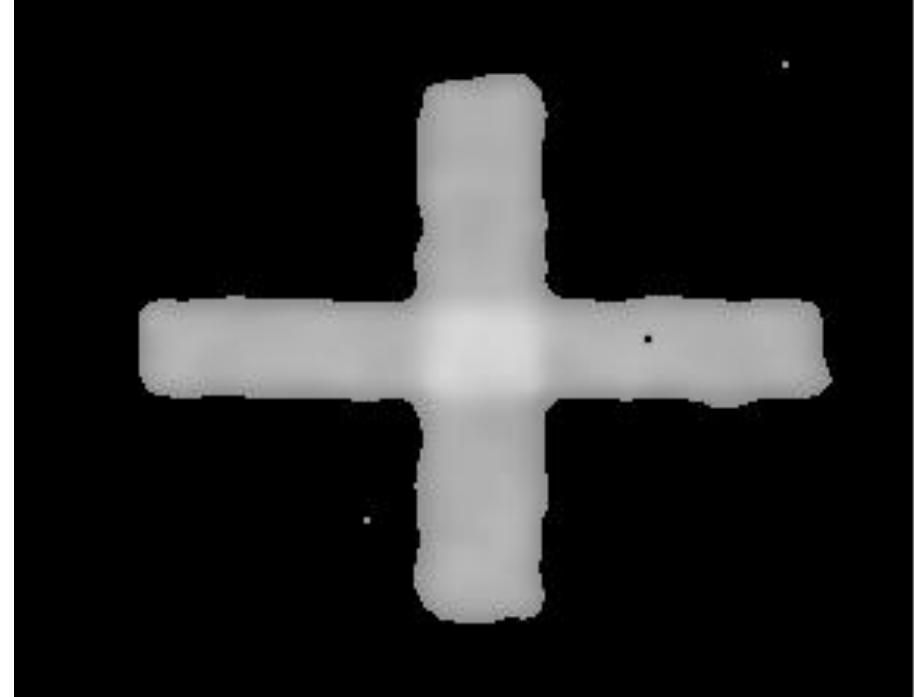
- Histograms
 - definition
 - properties
 - colour histograms
 - **thresholding**
 - segmentation
 - equalization

Thresholding

Original



Left-truncated



Thresholding

elements (section 4.3.2b).
f an optical system is a perspective
a) models the imaging geometry ac
y described by the position of the
focal length (section 4.3.2c). For th
determine the distance range that
of field, section 4.3.2d) and to lear
d hypercentric optical systems (se
al optical system from a perfect

Original



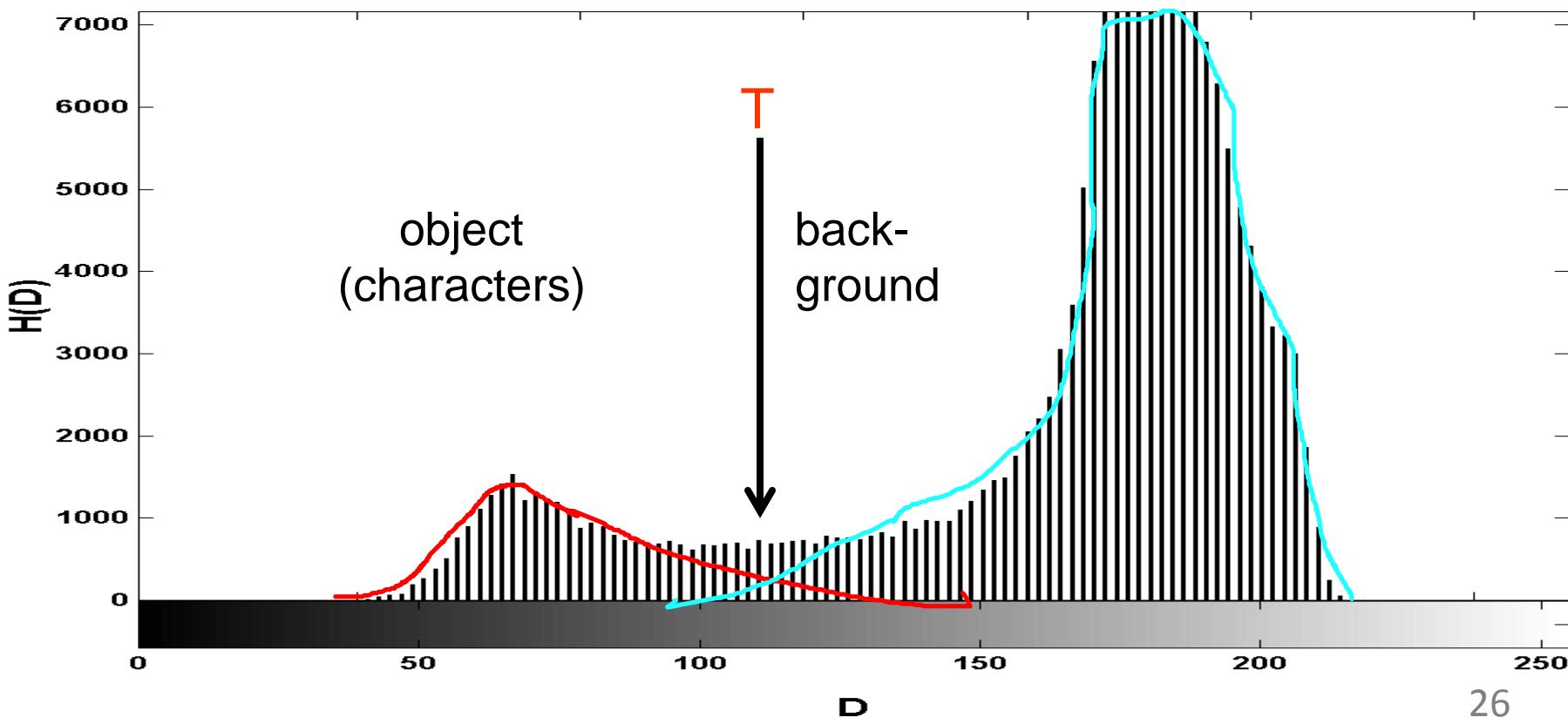
elements (section 4.3.2b).
f an optical system is a perspective
a) models the imaging geometry ac
y described by the position of the
focal length (section 4.3.2c). For th
determine the distance range that
of field, section 4.3.2d) and to lear
d hypercentric optical systems (se
al optical system from a perfect

Binary

To separate *contrasting* objects from background

Thresholding

- For an object on a contrasting background
 - histogram is **bimodal** (i.e., contains two peaks)
 - threshold **T** separates object from background



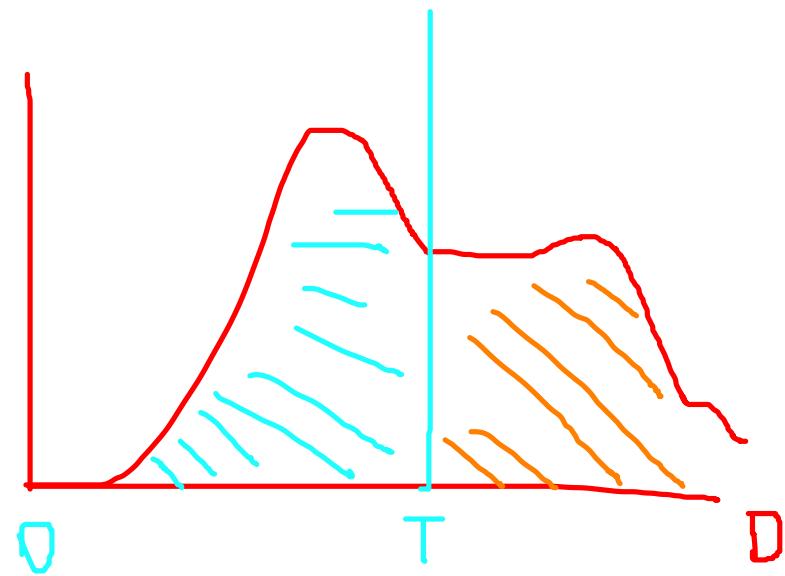
Measurements with histograms

- Area of the segmented object
 - The total **area A** of dark objects on a contrasting light background can be computed from the **histogram $H(D)$** and the **threshold T** .

$$A = \sum_{D=0}^T H(D)$$

$A + A' = \text{size of the image}$

$$A' = \sum_{D=T}^{\text{MAX}} H(D)$$



Interactive thresholding

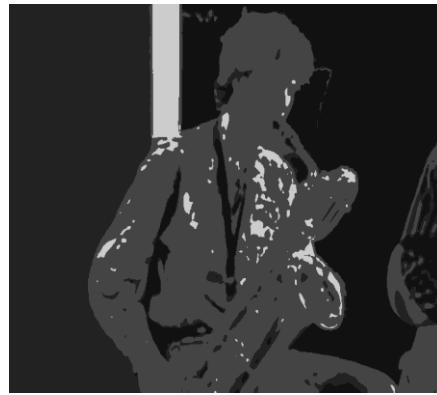


E.g. in GIMP or Photoshop

Today's agenda

- Histograms
 - definition
 - properties
 - colour histograms
 - thresholding
 - segmentation
 - equalization

Thresholding & segmentation



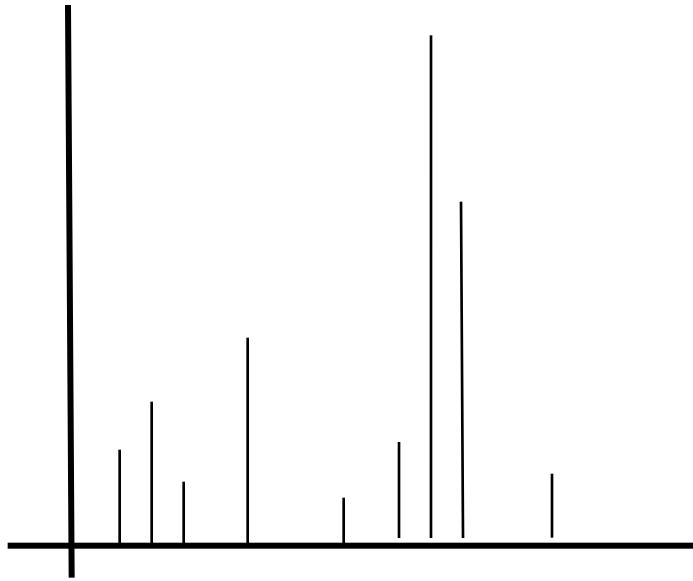
Today's agenda

- Histograms
 - definition
 - properties
 - colour histograms
 - thresholding
 - segmentation
 - equalization

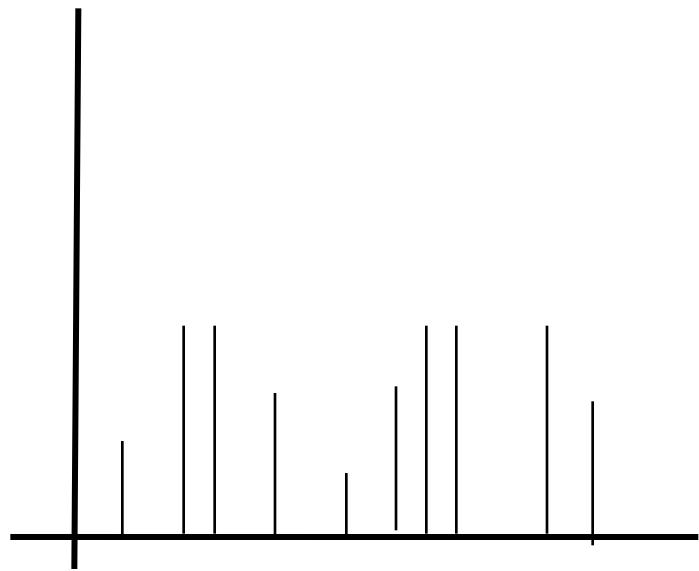
Histogram equalization

- Objective
 - To produce images with *evenly distributed* histograms without changing the number of grey levels
 - To map grey levels of an image into a target image with an **even distribution** of grey levels
- Application
 - Improved interpretability (e.g. scientific applications)
 - Automatic **contrast adjustment**

Histogram equalization



Histogram of the **original** image

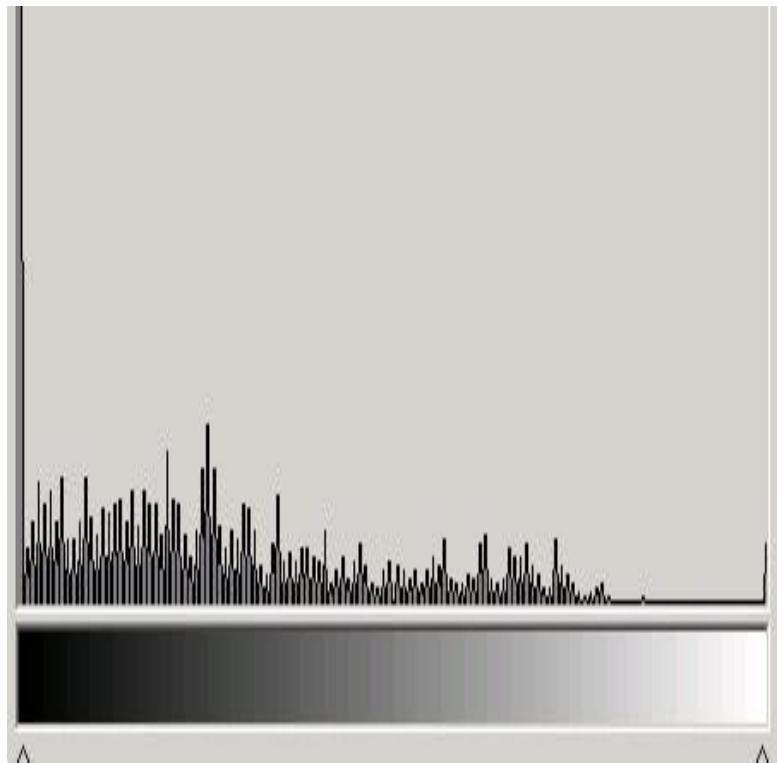


Histogram of the **target** image

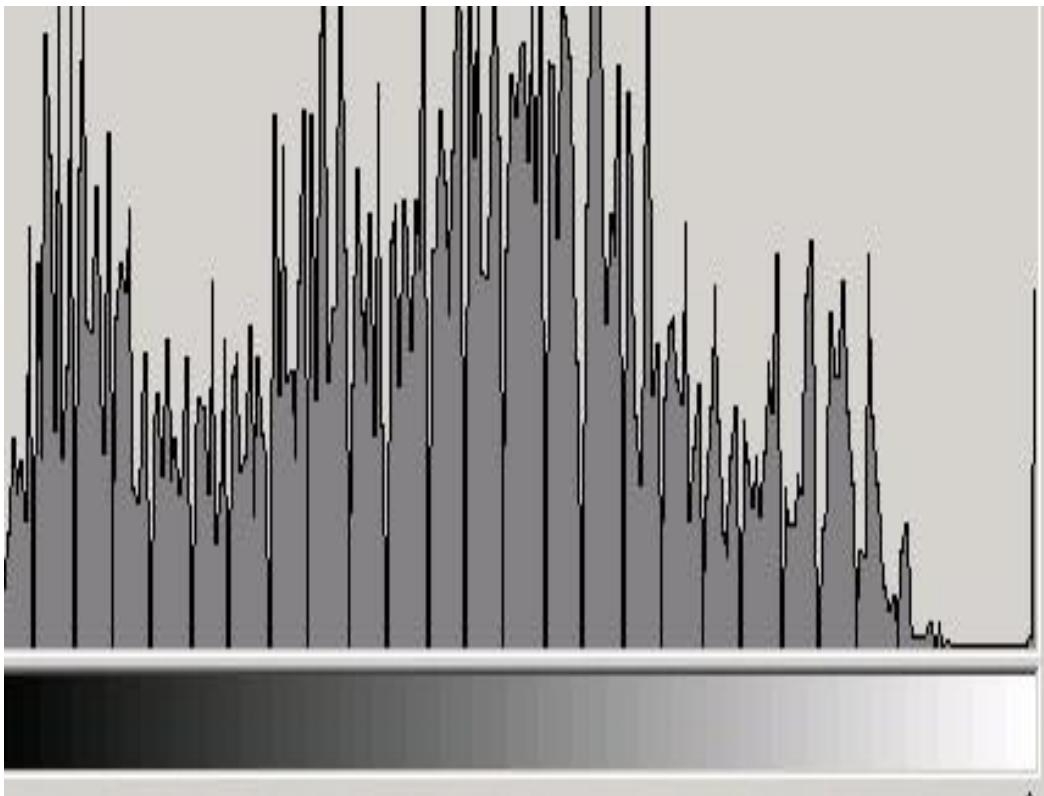
Example



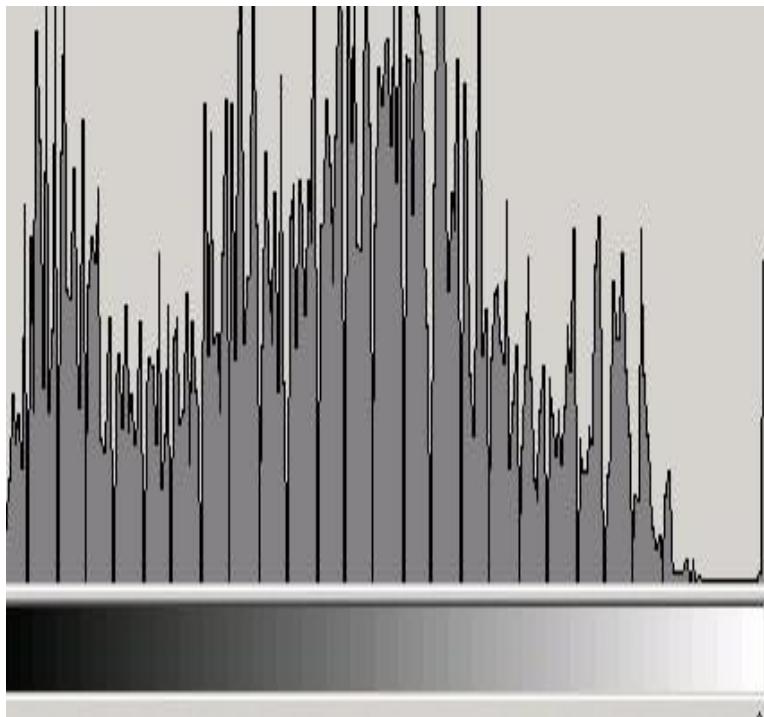
bad contrast



Equalized image



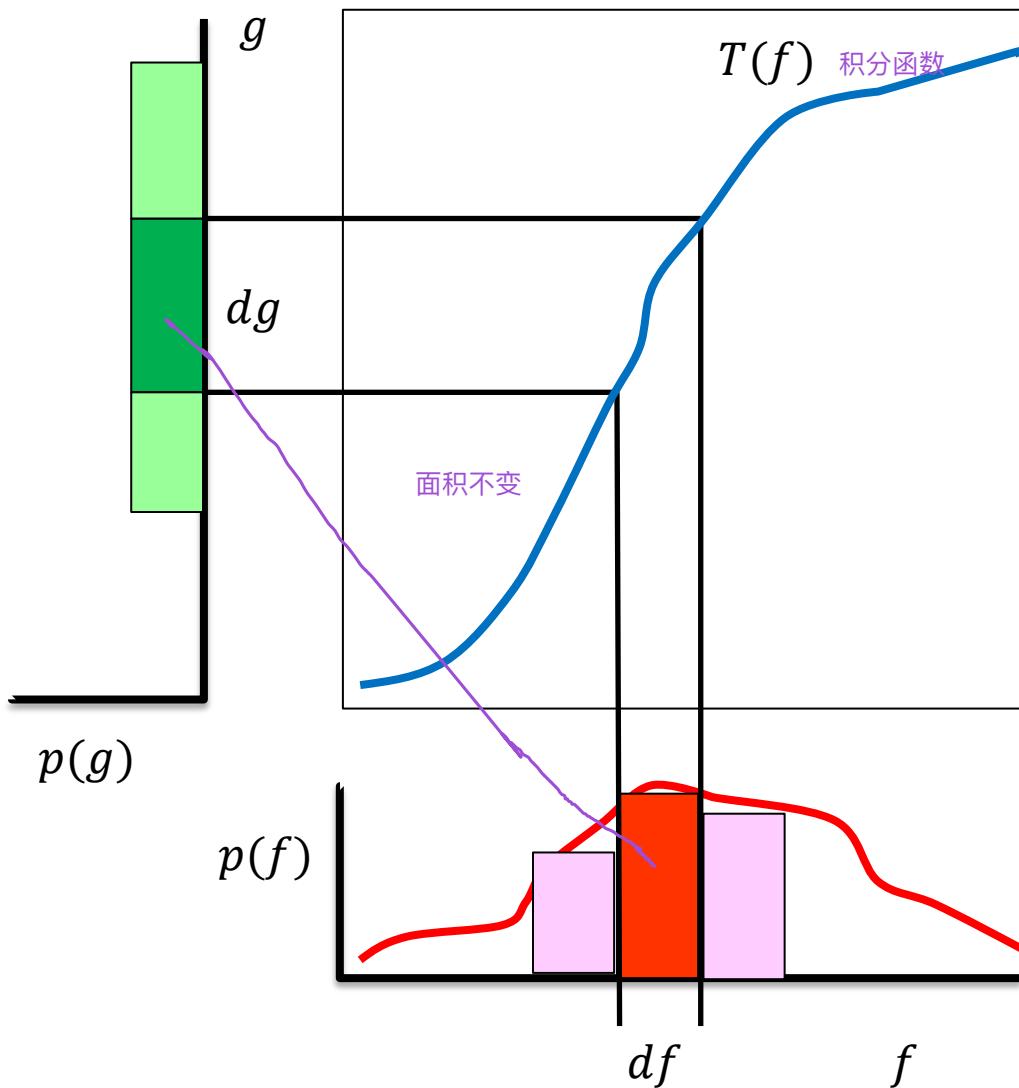
Cumulative distribution function



Comparison



Transformation of probability density



- $p(f)$ transforms to $p(g)$
- $T(f)$ is monotonic
- Probability of f in df must be equal to probability of g in dg
- So the area of the two rectangles must be equal
- Note that the width of dg varies
- Also called *change-of-variables*

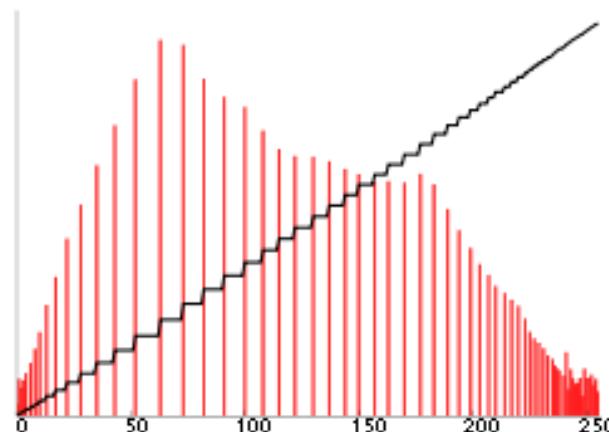
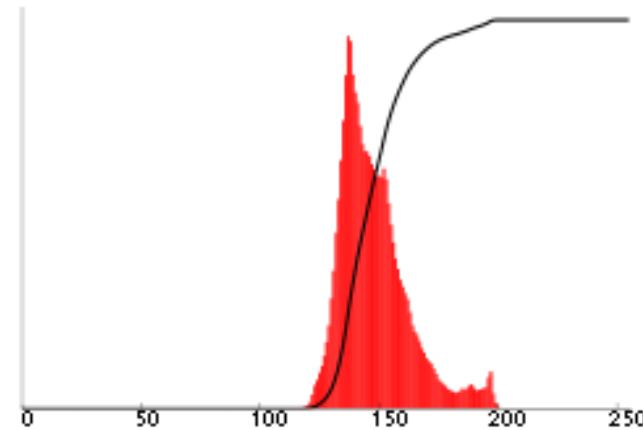
Continuous histogram equalization

- Suppose we have a transformation $g = T(f)$
- Let $p(f)$ and $p(g)$ be the densities (histograms) of f and g
- If $T(f)$ is a monotonic function, then $p(f)df = p(g)dg$
- Set $p(g) = 1$ to make the output distribution *uniform*
- Then $p(f) = dg/df$ from before. We want to find $g = T(f)$
- Integrate both sides: $g = \int_0^f p(x)dx = P(f)$
- Where $P(f)$ is the *cumulative distribution* function of f
- If $p(f)$ is high, then $P(f)$ is steep so dg is wide and $p(g)$ is low
- If $p(f)$ is low, then $P(f)$ is shallow so dg is narrow and $p(g)$ is high
- The histogram is evened-out, so that $P(g)$ is linear

Discrete histogram equalization

- The principle is the same in the discrete case
- Instead of the g axis being continuously deformed, the *spacing* between the histogram bins will vary
- So the output histogram won't *look* flat, and there will be gaps, in general
- But the *cumulative* histogram *will* be globally linear, as in the continuous case
- See Gonzalez & Woods or another book for details

Complete example



What did we learn today?

- Histograms
 - definition
 - properties
 - colour histograms
 - thresholding
 - segmentation
 - equalization

Image and video processing

(EBU723U)

Image transformations

Dr. Miles Hansard

Today's agenda

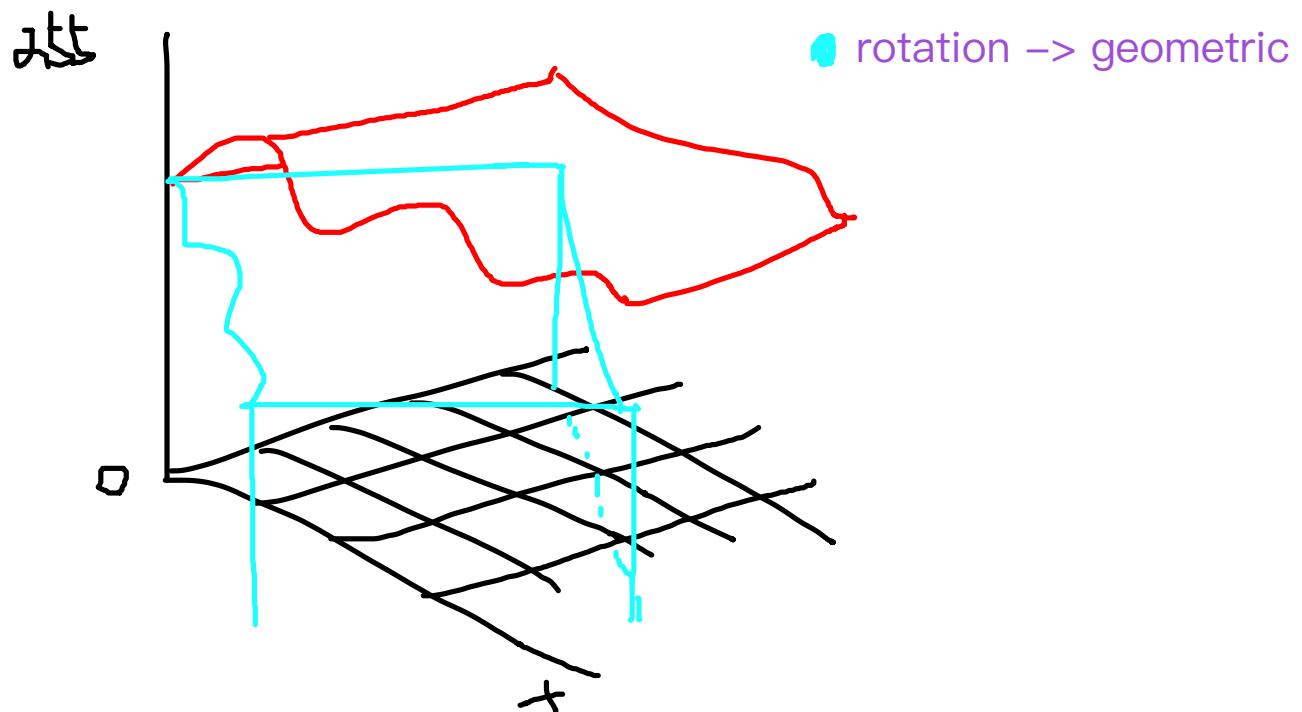
- Algebraic transformations
- Geometric transformations

The image shows handwritten notes on a light blue background. At the top left, the word "Algebraic" is written in red, with a small "CS>" superscript. To its right, the word "PHOTOMETRIC" is written in red. Below these, the equation $f(x, y) + d$ is written in red. In the center, the word "GEOMETRIC" is written in green, with a small "0" superscript. Below it, the equation $f(x-t, y)$ is written in green. A green bracket under the term $x-t$ is labeled "x'".

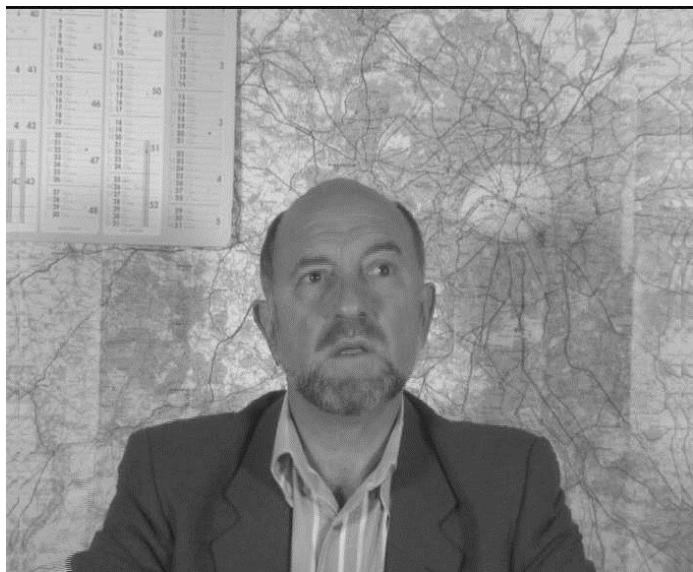
$$\text{Algebraic} \quad CS>$$
$$\text{PHOTOMETRIC}$$
$$f(x, y) + d$$
$$\text{GEOMETRIC} \quad 0$$
$$f(x-t, y)$$
$$x'$$

Today's agenda

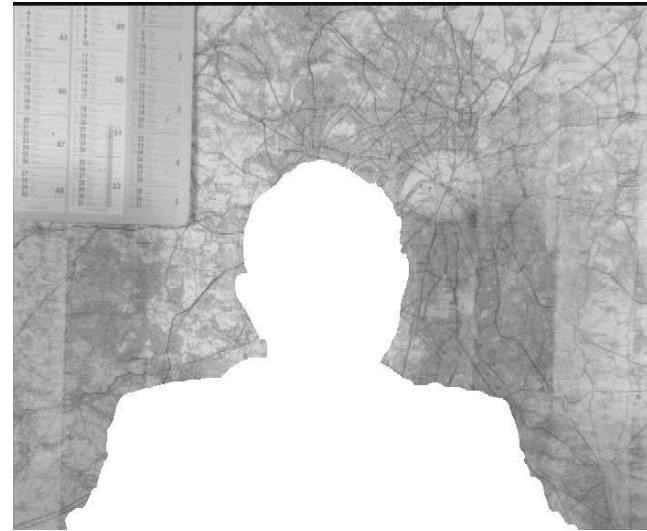
- Algebraic transformations
- Geometric transformations



Example: background subtraction



-



=



Algebraic operations

- Binary operators involving more than one image
- Usually the images are of the **same dimension**
- Examples
 - addition of two or more images
 - subtraction of one image from another
 - multiplication and division

Algebraic operations

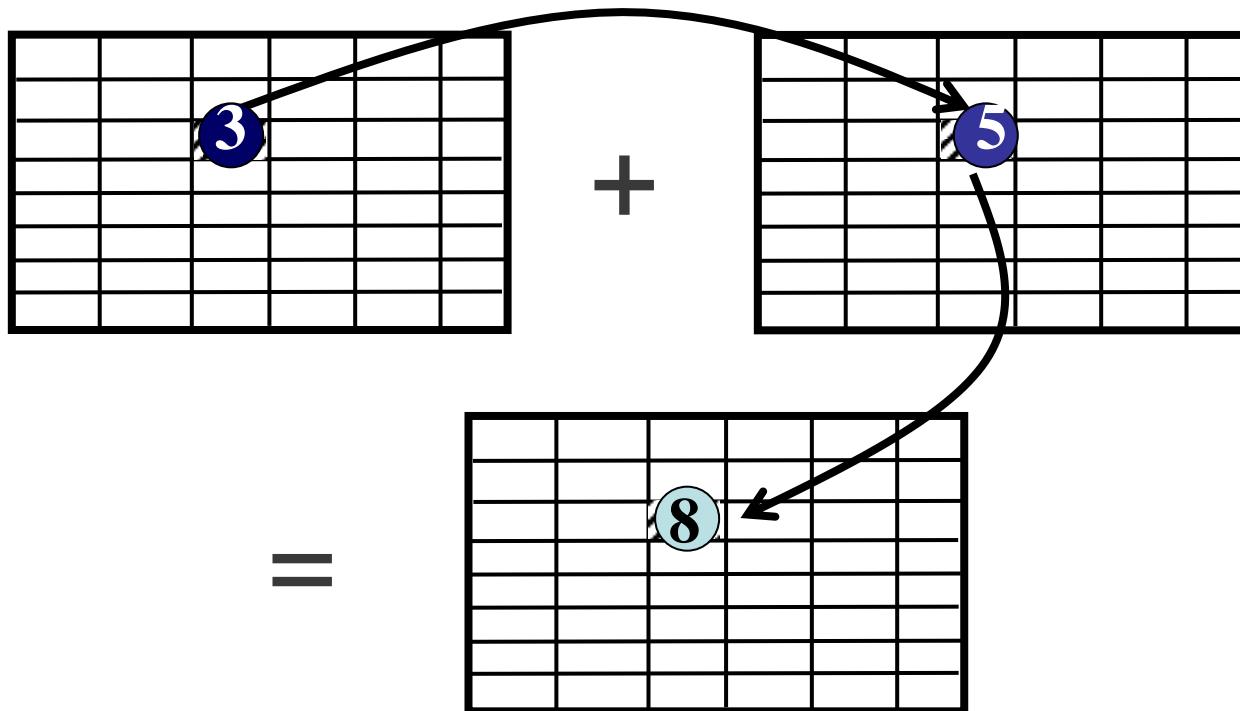
- Non invertible
 - Original information is lost (e.g., $2+2 = 3+1 = 1+3 = \dots$)
- Clipping
 - Combination of pixel values may exceed range of output variable
 - then clipping occurs, e.g., if unsigned byte, $230 + 80 = 310 > 255$
- Normalization
 - In general, may need to map $f \in [a, b]$ to $g \in [c, d]$

$$g = c + \frac{f - a}{b - a} (d - c)$$

Where typically $c = 0$ and $d = 255$ for eight-bit images

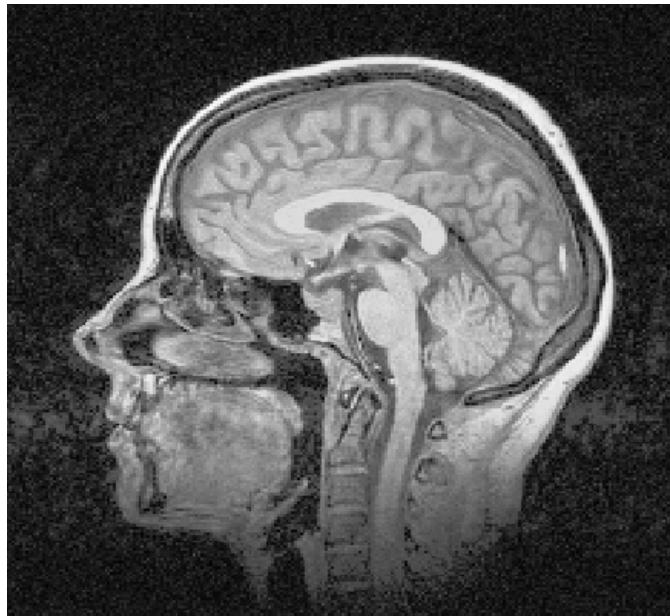
Example

$$3 + 5 = 8$$

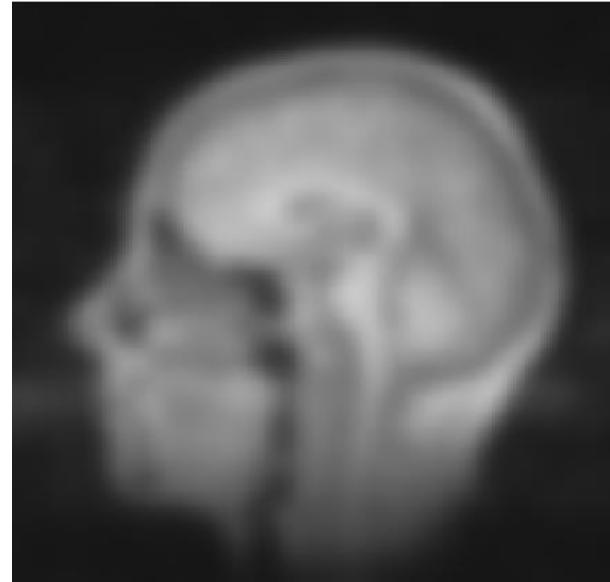


Example: noise removal

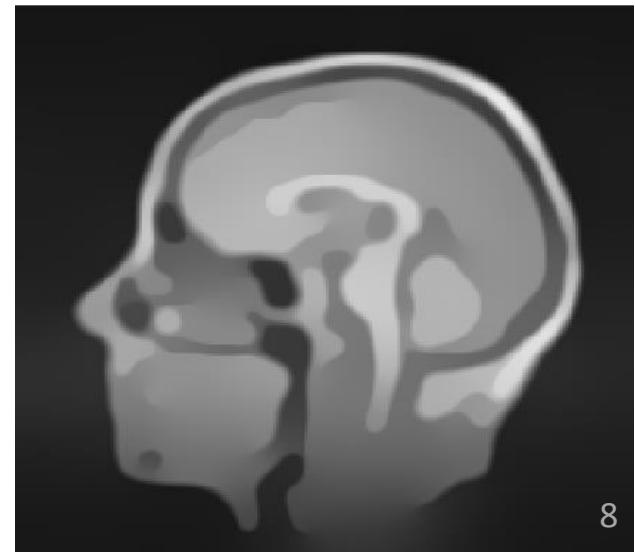
linear



+



$\times 1/2 =$



Noise Removal

Alternative: Median Filter

Single image



Average of 50



Average of 10

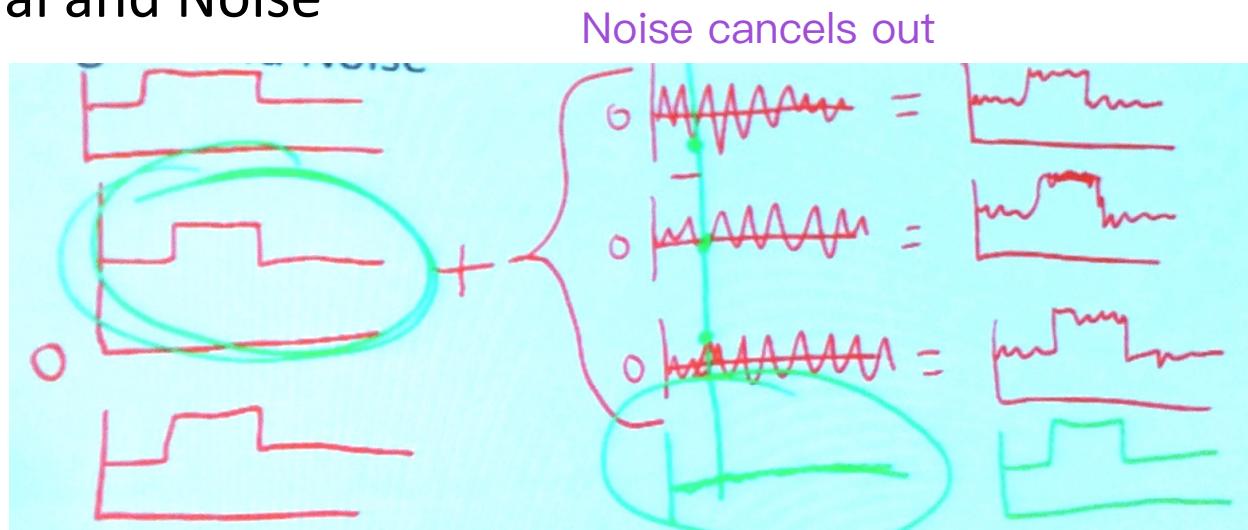


Average of 100



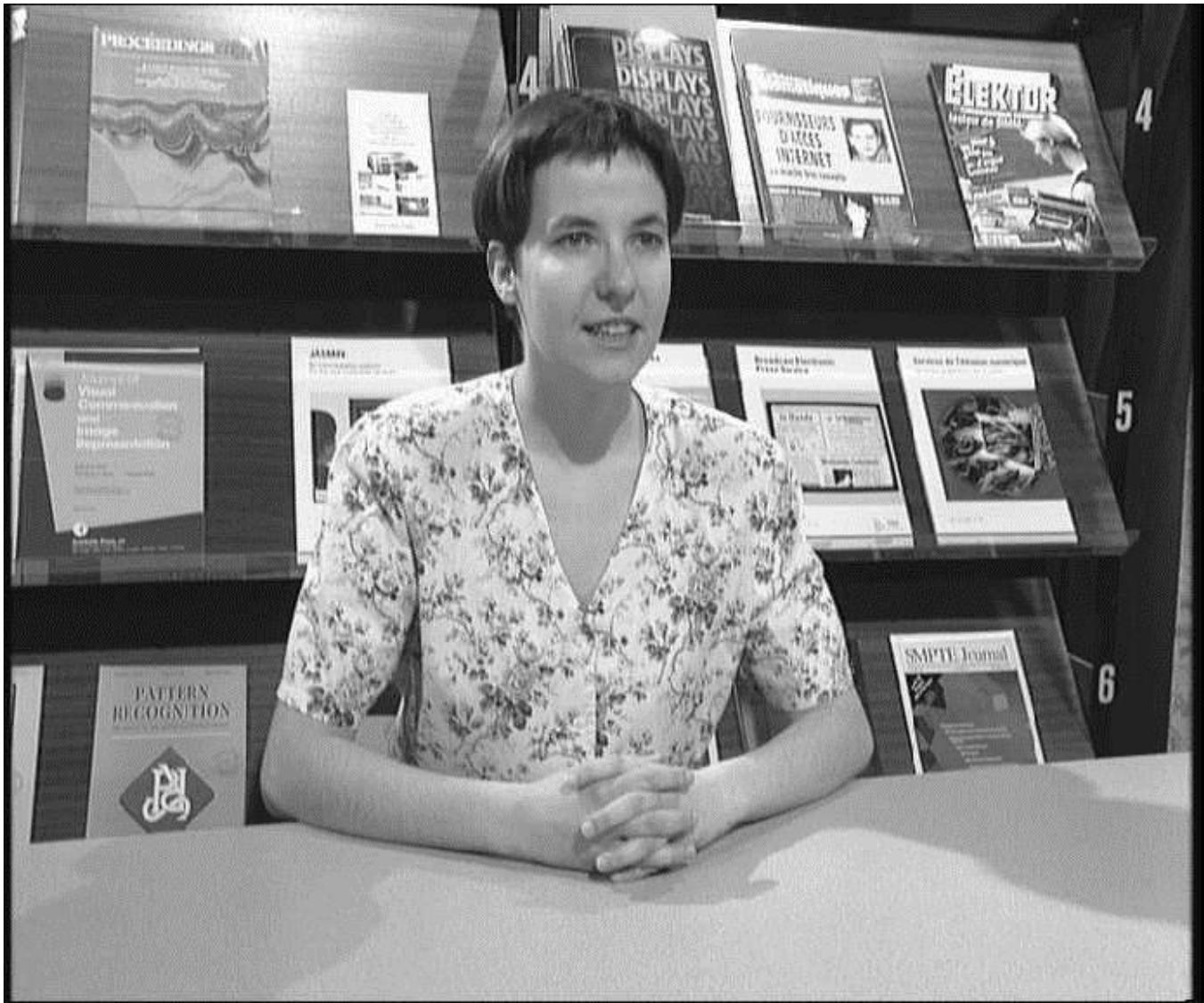
Sum for noise reduction

- If statistically independent noise is added to each of several identical images
- Or multiple noisy images are summed
 - The sum image has a higher SNR
- SNR can be computed by taking ratio of expected values of Signal and Noise



Problems: 1. Noise not cancelling out
2. Object is not still

Edge detection



EBU723U



Queen Mary
University of London

Edge detection using subtraction

- If an image is displaced (translated) relative to another image, then the difference between them approximates the first derivative

$$\frac{\Delta y}{\Delta x} = \frac{f(x + \Delta x) - f(x)}{\Delta x}$$

where Δx is the displacement

- This is called a *forward difference* approximation because $f(x)$ is compared to a value on the right. The *central difference* is often a better approximation:

半个像素? –取相邻两像素的平均值

$$\frac{\Delta y}{\Delta x} = \frac{f(x + \Delta x/2) - f(x - \Delta x/2)}{\Delta x}$$

Better and faster
indeed

But not applicable to time domain

Finite-difference filters

- Derivative operators can be implemented as linear filters.
- The backward x-difference (applied to image-rows) filter is
 $t = [-1, 1]$
- The central x-difference is between values at $x+1/2$ and $x-1/2$
- Let the image-row (locally) be $f = [a, b, c]$
- Values between a,b and between b,c are computed by the *averaging* filters
 $u = [1/2, 1/2, 0]$ and $v = [0, 1/2, 1/2]$ respectively.
- Then the central difference is $v \cdot f - u \cdot f = (v-u) \cdot f$
- So we can define the central x-difference filter as
 $w = v-u = [-1/2, 0, 1/2]$
- The y-difference filters can be similarly constructed.

Edge detection using subtraction

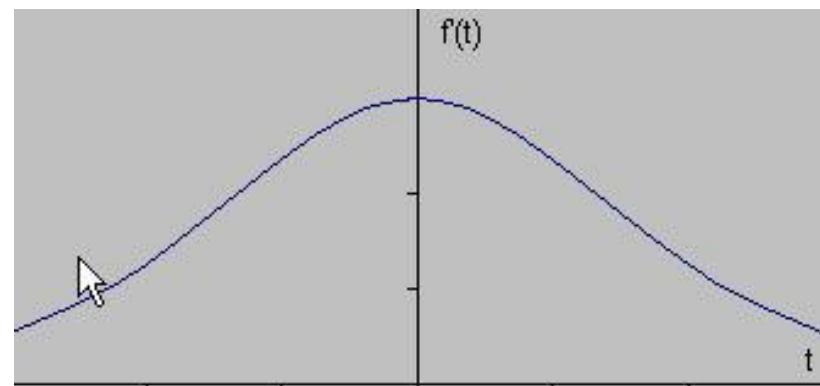
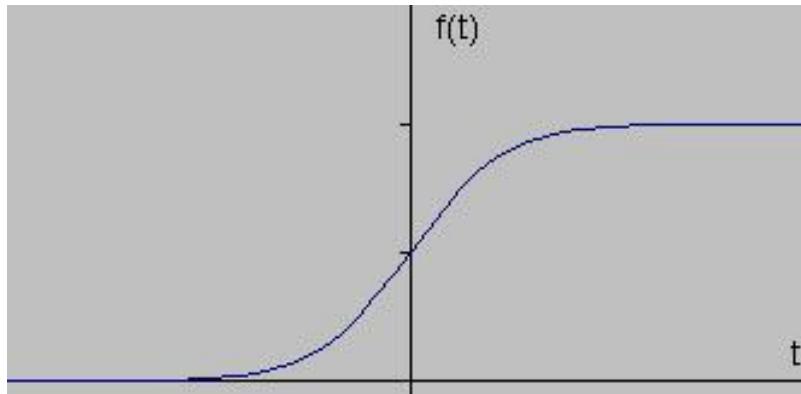


Original



Gradient Image

Edge-profile and derivative



Generalization of scalar derivative: Gradient

Given an image: $f(x, y)$, *then*

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

where

∇ is called the **gradient operator**

Note that the gradient ‘image’ contains
a *vector* for each pixel!

Gradient properties

- Points in direction of max upward slope
- Is a vector *perpendicular* to the *iso-contour* of the intensity
- Gradient magnitude
 - Useful scalar function of gradient
 - is equal to value of slope

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

Gradient magnitude

- Square root
 - computationally expensive → sometimes simpler approximation

$$|\nabla f(x,y)| \approx \max \left[\begin{array}{l} |f(x,y) - f(x+1,y)| \\ |f(x,y) - f(x,y+1)| \end{array} \right]$$

Directional derivative

- More generally, given the horizontal and vertical derivatives, the derivative in any other direction θ can be obtained:

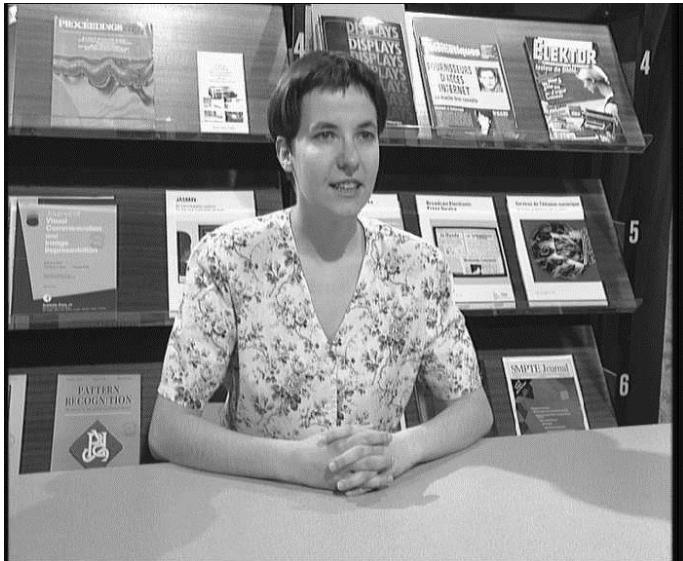
$$\frac{\partial}{\partial \theta} f(x, y) = \cos(\theta) \frac{\partial}{\partial x} f(x, y) + \sin(\theta) \frac{\partial}{\partial y} f(x, y)$$

- This is just the dot-product of the gradient with the unit-vector in direction θ .
- E.g. the derivative at angle zero is just $\frac{\partial}{\partial x} f(x, y)$
- This property is called *steerability* of the gradient

Today's agenda

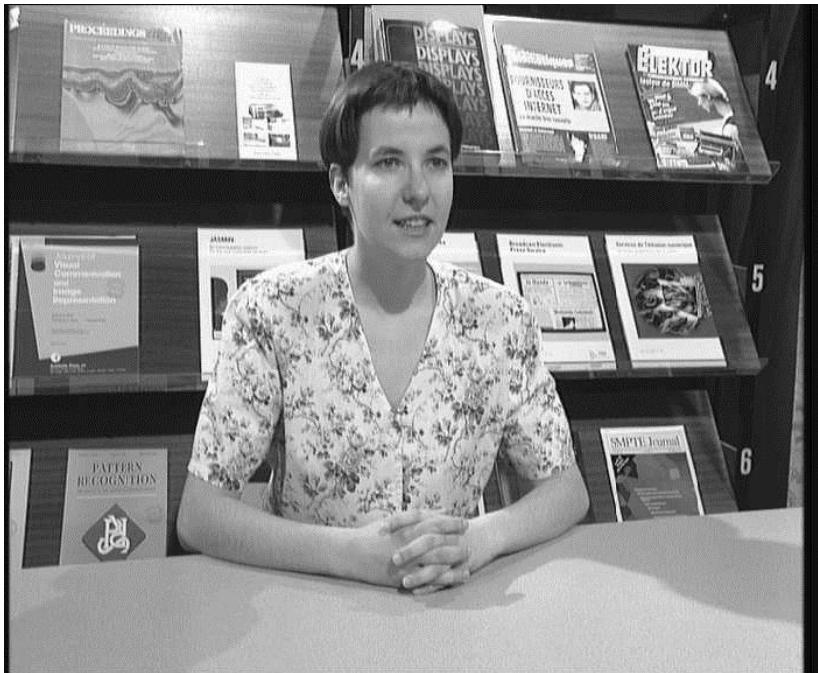
- Algebraic transformations
- Geometric transformations

Spatial transformations

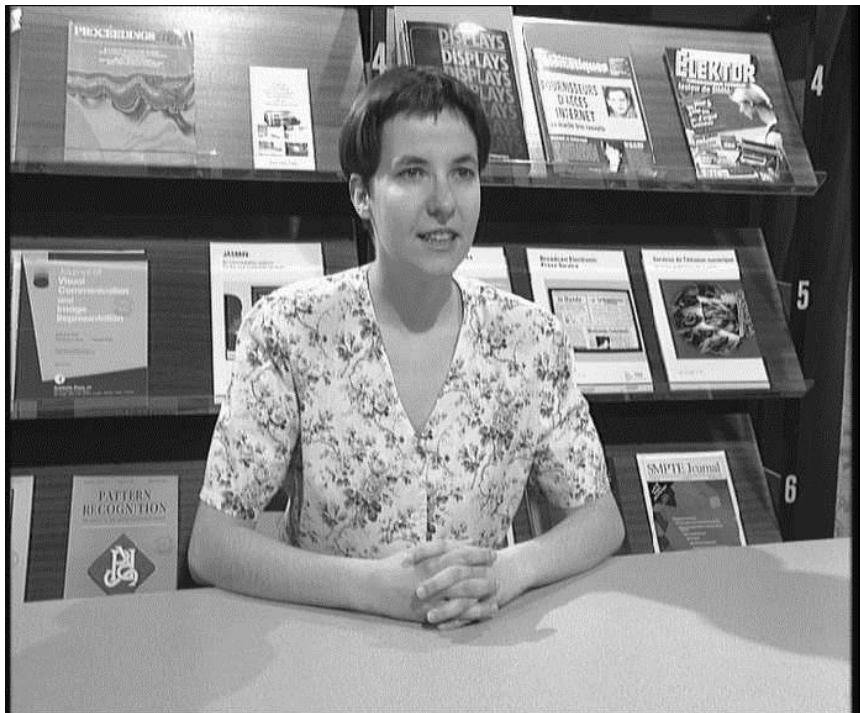


Example: rotation

- Simply a matrix multiplication

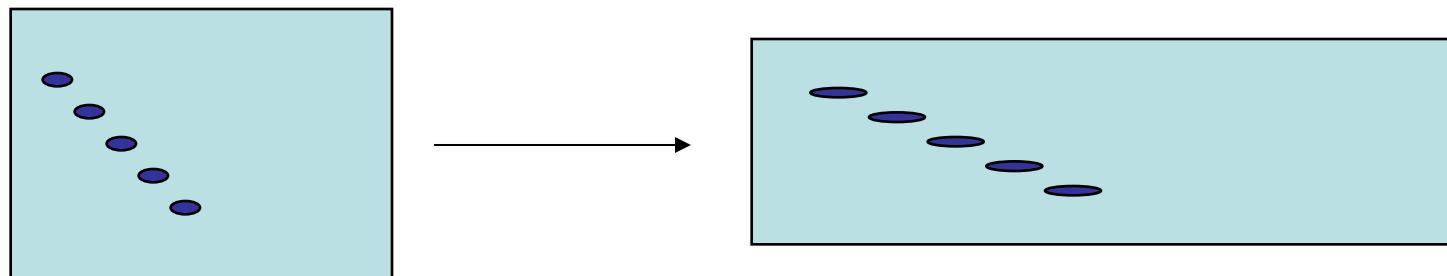


Example: scaling

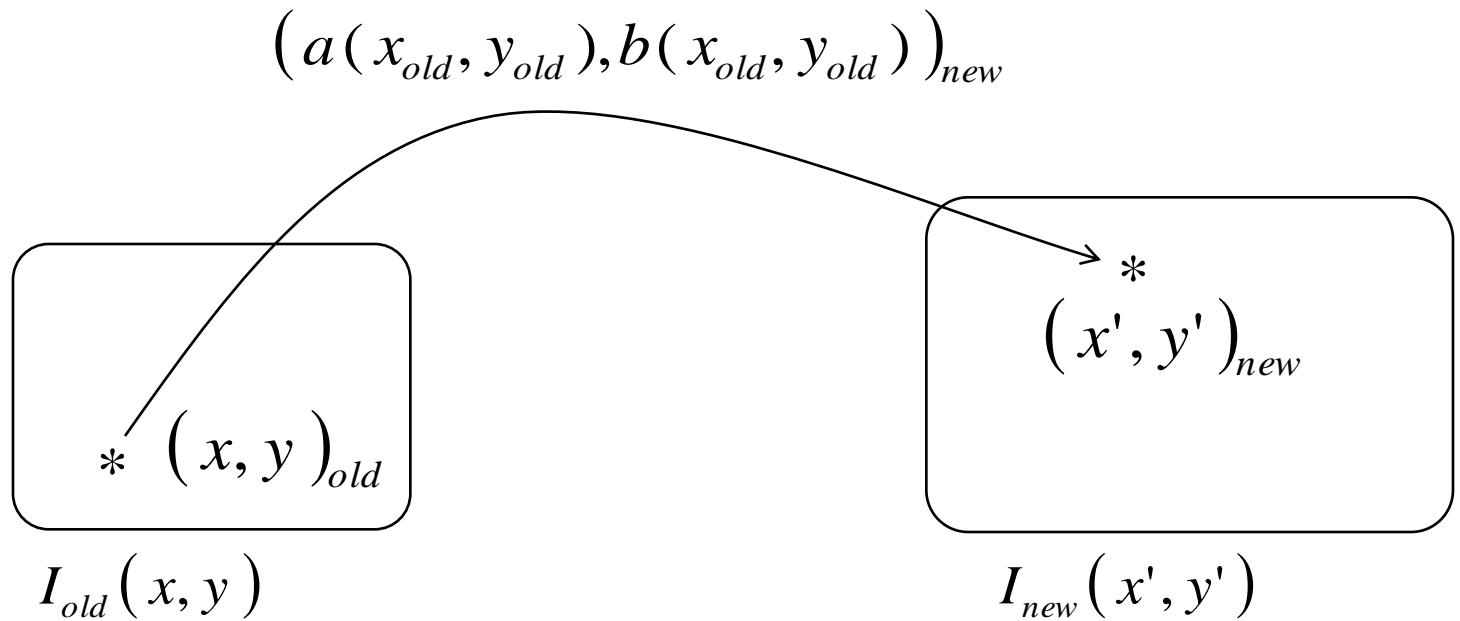


Geometric operations

- Change spatial relationship among pixels in an image
 - **Continuous**: rubber-sheet geometry
 - **Discontinuous**: change pixel neighborhoods
- Two considerations
 - Spatial transformation
 - Grey-level interpolation



Coordinate transformations



Scaling

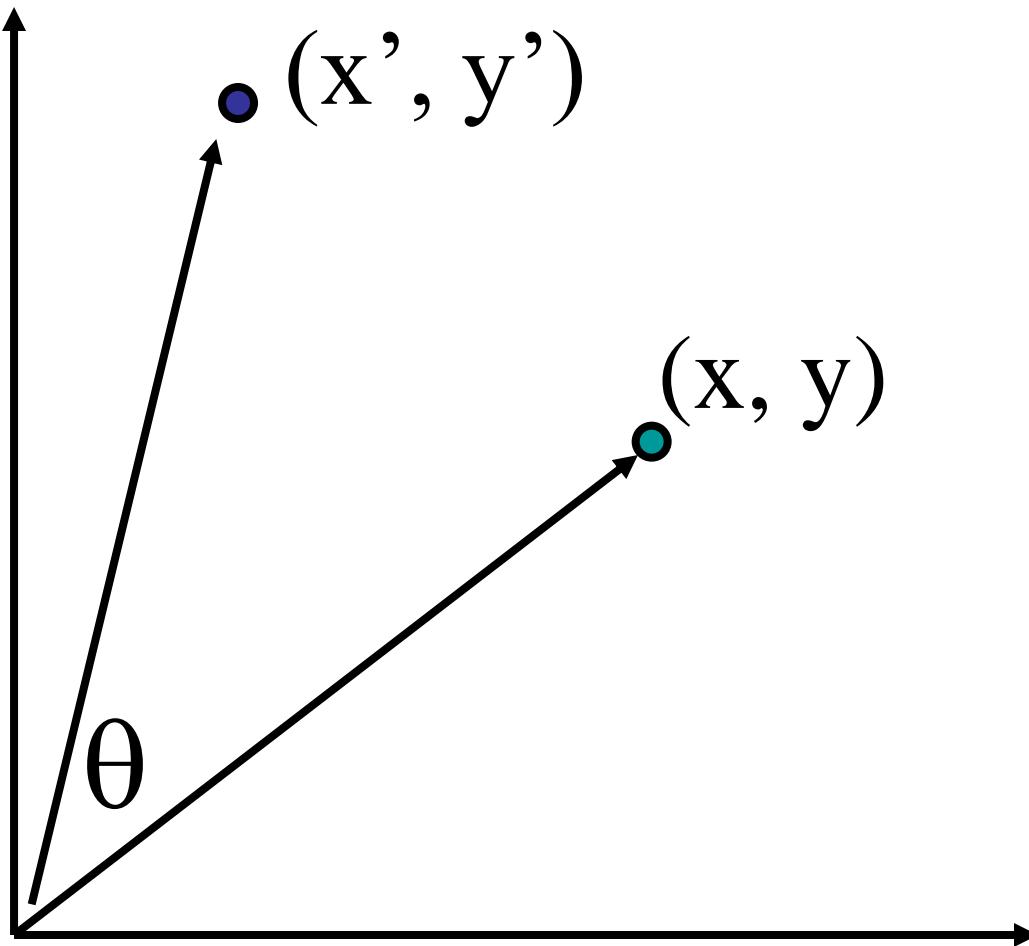
- Scaling operation:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} ax \\ by \end{bmatrix}$$

- Or, in matrix form:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix} \underbrace{\begin{bmatrix} x \\ y \end{bmatrix}}_{scaling\ matrix}$$

2D rotation



2D rotation

- Or, in matrix form:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

- Even though $\sin(\Theta)$ and $\cos(\Theta)$ are nonlinear functions of q ,

- x' is a **linear combination** of x and y
 - y' is a **linear combination** of x and y

- To derive this, represent $[x,y]$ in the form $[r \cos(\Phi), r \sin(\Phi)]$
 - Represent $[x',y']$ in the form $[r \cos(\Phi+\Theta), r \sin(\Phi+\Theta)]$
 - Use the trigonometric identities:

$$\cos(\Phi+\Theta) = \cos(\Theta) \cos(\Phi) - \sin(\Theta) \sin(\Phi)$$

$$\sin(\Phi+\Theta) = \sin(\Theta) \cos(\Phi) + \cos(\Theta) \sin(\Phi)$$

Matrix representation - summary

- Represent 2D transformation by a matrix

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

- Multiply matrix by column vector
↔ apply transformation to point

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \quad \begin{aligned} x' &= ax + by \\ y' &= cx + dy \end{aligned}$$

Determinant = $ad - bc$

A transformation is meaningless when
determinant = 0

Matrix representation - summary

- Transformations combined ('composed') by multiplication
- Matrix multiplication is *associative*

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} e & f \\ g & h \end{bmatrix} \begin{bmatrix} i & j \\ k & l \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

Matrices are a convenient and efficient way
to represent a sequence of transformations

- Matrices are *invertible* if the determinant is nonzero
- The determinant of $\begin{bmatrix} a & b \\ c & d \end{bmatrix}$ is the scalar $ad - bc$

2x2 matrices

- What types of transformations can be represented with a 2x2 matrix?

2D identity

$$\begin{aligned}x' &= x \\y' &= y\end{aligned}$$

$$\begin{bmatrix}x' \\ y'\end{bmatrix} = \begin{bmatrix}1 & 0 \\ 0 & 1\end{bmatrix} \begin{bmatrix}x \\ y\end{bmatrix}$$

2D scale around (0,0)

$$x' = s_x * x$$

$$y' = s_y * y$$

$$\begin{bmatrix}x' \\ y'\end{bmatrix} = \begin{bmatrix}s_x & 0 \\ 0 & s_y\end{bmatrix} \begin{bmatrix}x \\ y\end{bmatrix}$$

2x2 matrices

- What types of transformations can be represented with a 2x2 matrix?

2D rotate around (0,0)

$$\begin{aligned}x' &= \cos \Theta * x - \sin \Theta * y \\y' &= \sin \Theta * x + \cos \Theta * y\end{aligned}$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos \Theta & -\sin \Theta \\ \sin \Theta & \cos \Theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

2D shear

$$x' = x + sh_x * y$$

$$y' = sh_y * x + y$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & sh_x \\ sh_y & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

2x2 matrices

- What types of transformations can be represented with a 2x2 matrix?

2D mirror about Y axis

$$\begin{aligned}x' &= -x \\y' &= y\end{aligned}$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

2D mirror over (0,0)

$$\begin{aligned}x' &= -x \\y' &= -y\end{aligned}$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

2x2 matrices

- What types of transformations can be represented with a 2x2 matrix?

2D translation

$$x' = x + t_x$$

$$y' = y + t_y$$

NO!

Only linear 2D transformations
can be represented with a 2x2 matrix

Homogeneous coordinates

- How can we represent translation as a 3x3 matrix?

$$x' = x + t_x$$

$$y' = y + t_y$$

Homogeneous coordinates

- Homogeneous coordinates
 - represent coordinates in 2 dimensions with a 3-vector
 - seem unintuitive, but they make graphics operations much easier

$$\begin{bmatrix} x \\ y \end{bmatrix} \xrightarrow{\text{homogeneous coords}} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Homogeneous coordinates

- How can we represent translation as a 3x3 matrix?
 - Using the rightmost column

$$\begin{aligned}x' &= x + t_x \\y' &= y + t_y\end{aligned}$$

$$\text{Translation} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}$$

Homogeneous coordinates

- In a plane, the **homogeneous coordinates** of a point whose Cartesian coordinates are (x, y) are any three numbers (a_1, a_2, a_3) for which

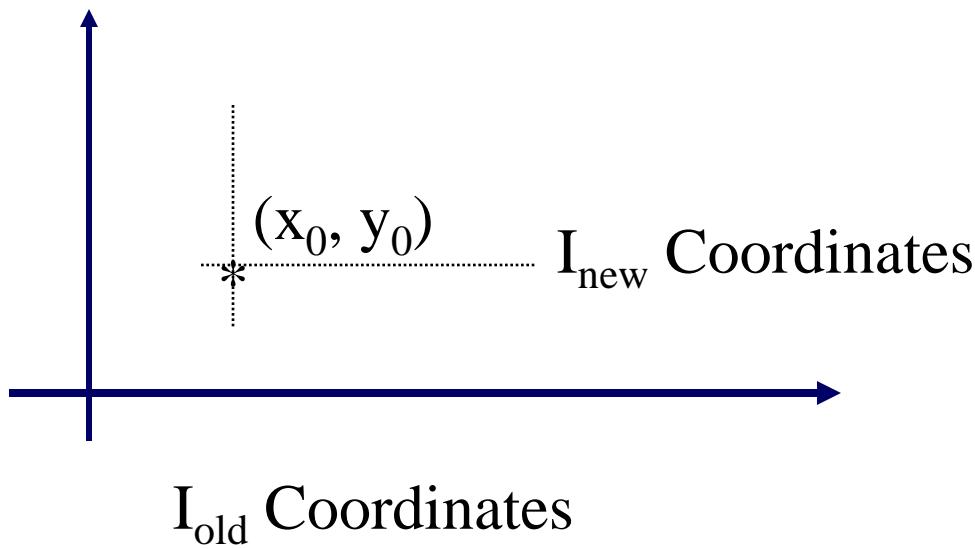
$$a_1/a_3 = x$$

$$a_2/a_3 = y$$

Let $a_3 = 1$

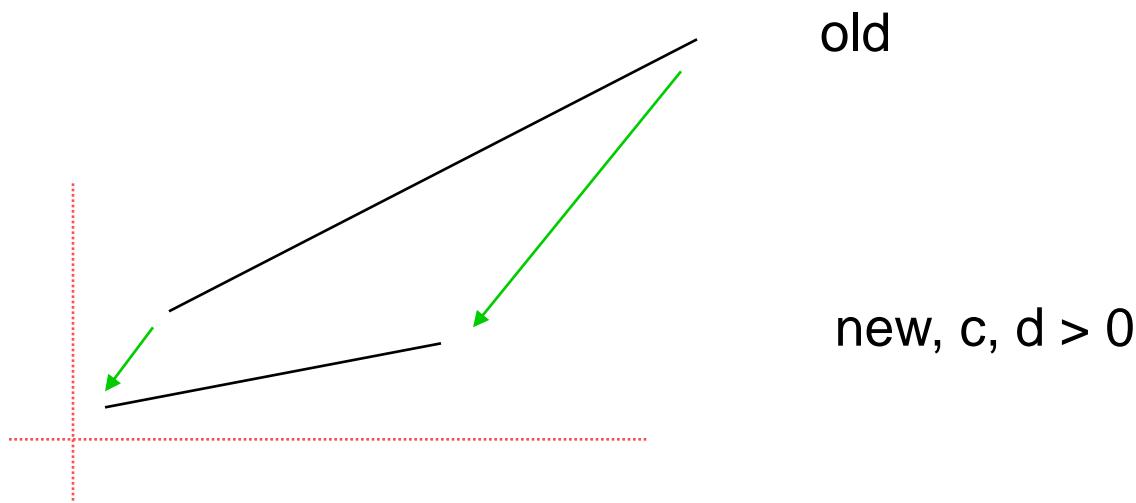
Coordinate translation

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix}_{new} = \begin{bmatrix} a(x, y) \\ b(x, y) \\ 1 \end{bmatrix}_{new} = \begin{bmatrix} 1 & 0 & x_0 \\ 0 & 1 & y_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}_{old}$$



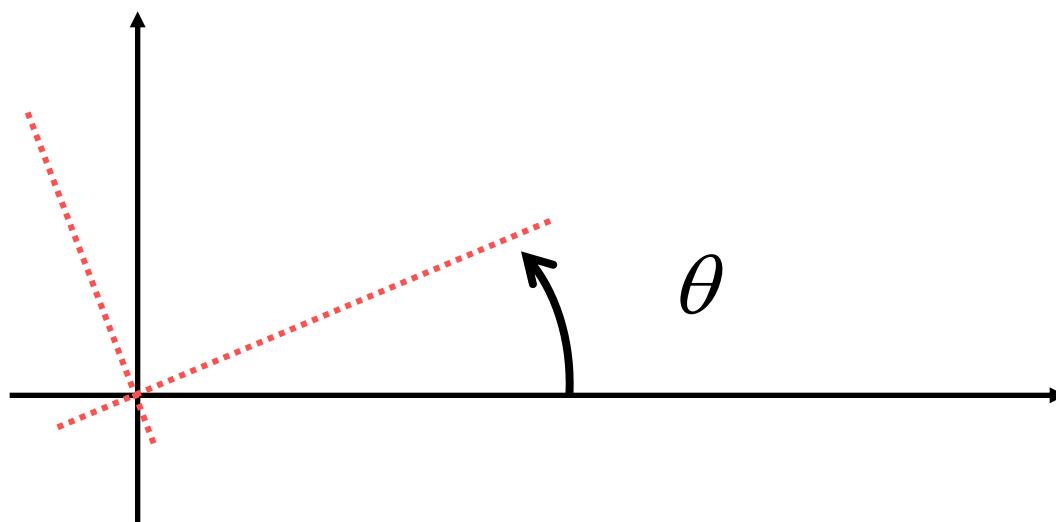
Coordinate scaling

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix}_{new} = \begin{bmatrix} a(x, y) \\ b(x, y) \\ 1 \end{bmatrix}_{new} = \begin{bmatrix} \frac{1}{c} & 0 & 0 \\ 0 & \frac{1}{d} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}_{old}$$



Coordinate rotation (around the origin)

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix}_{new} = \begin{bmatrix} a(x,y) \\ b(x,y) \\ 1 \end{bmatrix}_{new} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}_{old}$$



Compound transformations

- Compound transformations
 - order of operations is important
 - perform individual operations and combine into single function
 - Example: **rotation around arbitrary point** (x_0, y_0)
 - Translate from (x_0, y_0) to (0, 0)
 - Rotate
 - Translate back to (x_0, y_0)

Rotation around arbitrary point

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix}_{new} = \begin{bmatrix} a(x, y) \\ b(x, y) \\ 1 \end{bmatrix}_{new} = \begin{bmatrix} 1 & 0 & x_0 \\ 0 & 1 & y_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}_{transl}$$

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}_{transl} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}_{trans \& rot}$$

3D basic transformation in space

- Rotation around X-axis

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos \theta & -\sin \theta & 0 \\ 0 & \sin \theta & \cos \theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- Rotation around Y-axis

$$\begin{bmatrix} \cos \theta & 0 & \sin \theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin \theta & 0 & \cos \theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- Rotation around Z-axis

$$\begin{bmatrix} \cos \theta & -\sin \theta & 0 & 0 \\ \sin \theta & \cos \theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

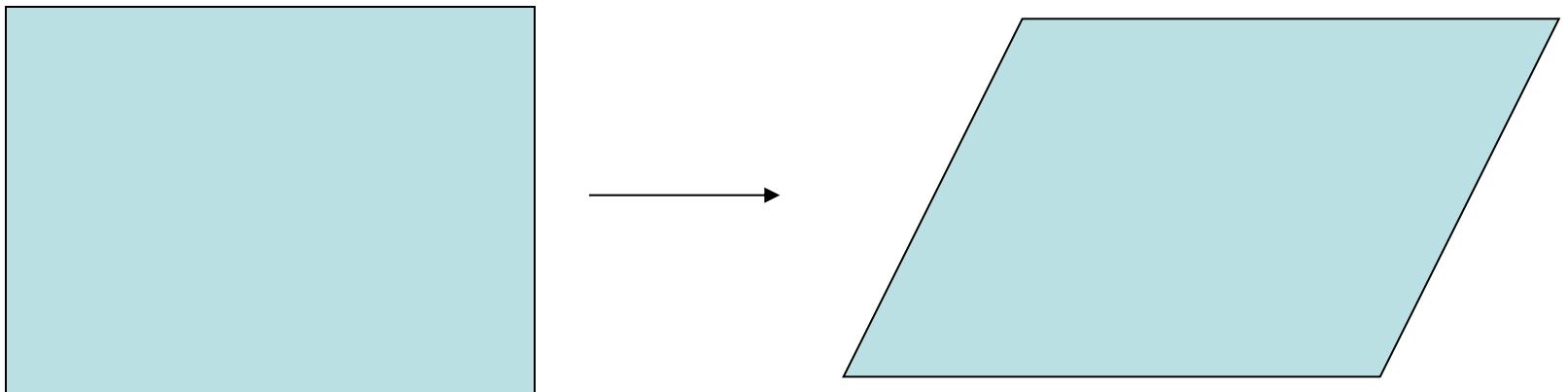
Skew and shear

- Take an image and *skew (or shear)* it to the side

$$Shear_{\theta} = \begin{bmatrix} 1 & \tan(\theta) \\ 0 & 1 \end{bmatrix}$$

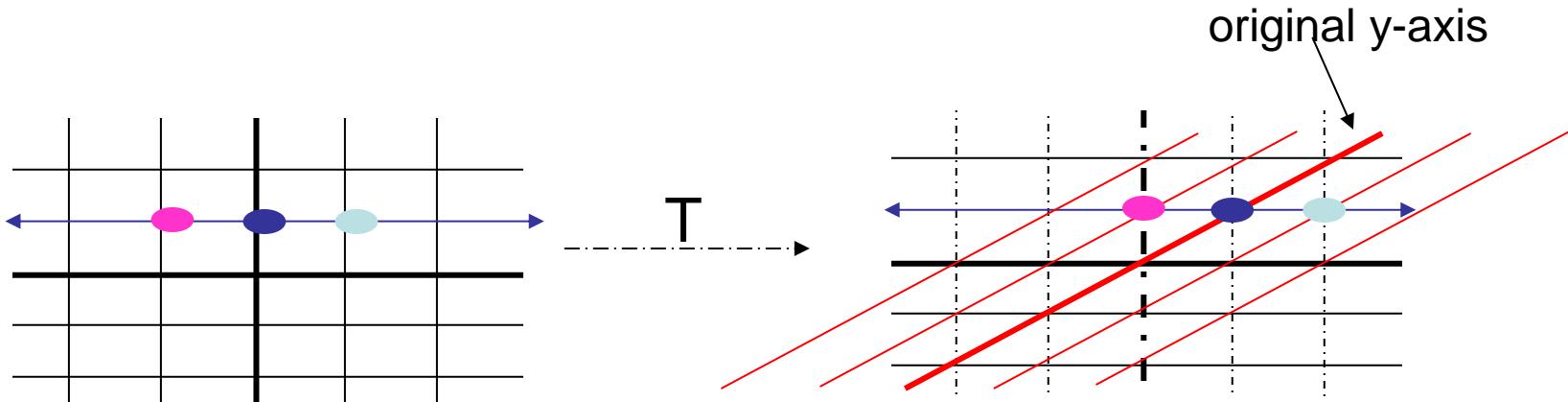
- Effect of the transformation
 - Squares become parallelograms
 - *x* coordinates skew to the right
 - *y* coordinates stay the same
 - Each scan line of the original image shifts relative to the one below it

Skew and shear

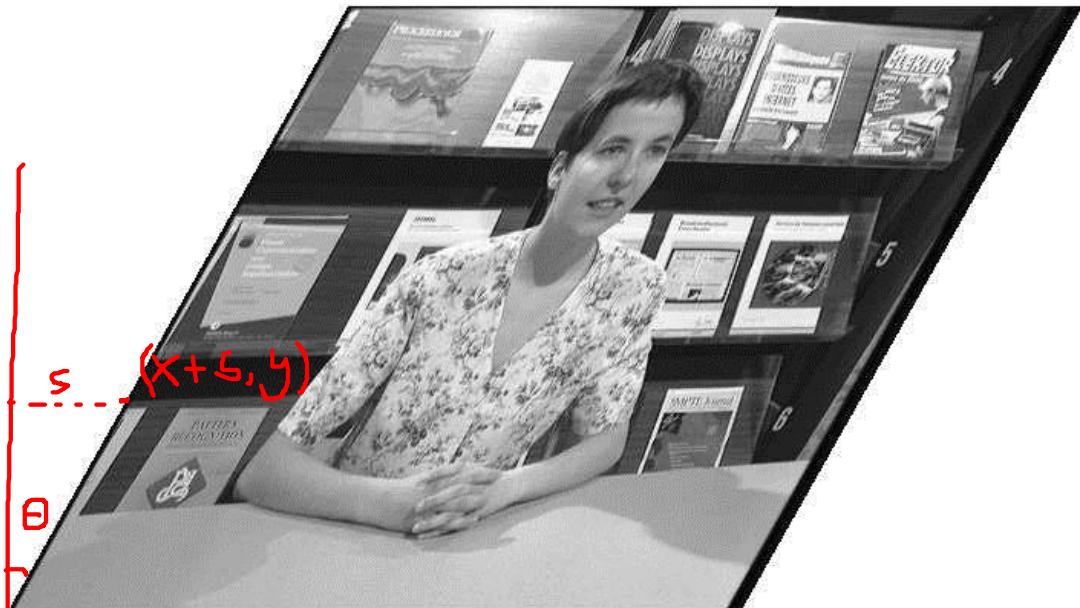


Skew and shear

- Skew and shear
 - Everything along the line $y=1$ stays on the same line $y=1$, but is translated to the right
 - Distance between points on this line is preserved



Example

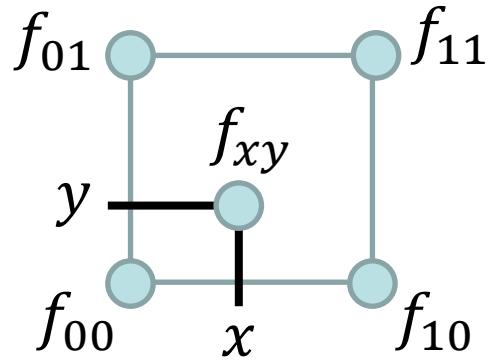


$$s = y * \tan\theta$$

$$\begin{bmatrix} 1 & \tan\theta & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Inverse warping

- In general, if we map each pixel $P = [x, y]$ to a new position $P' = W(x, y)$, we would end up with *holes* in the new image!
- This can be avoided by using the *inverse* of the warp W .
 - Loop through *all* destination pixels $[x', y']$
 - Find corresponding *source* position $[x, y] = W^{-1}(x', y')$
 - Find colour $f_{x'y'} = f_{xy}$ by **interpolation** in *source* image.
- Recall bilinear interpolation:
- $$f_{xy} = [1 - y, y] \begin{bmatrix} f_{00} & f_{10} \\ f_{01} & f_{11} \end{bmatrix} \begin{bmatrix} 1 - x \\ x \end{bmatrix}$$
- Interpolation may blur the result slightly.



What did we learn today?

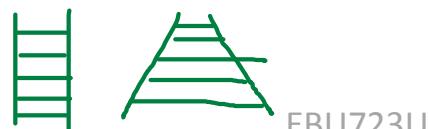
- Algebraic transformations
- Geometric transformations

Rigid (Euclidean) transformation: The shape does not change
E.g. Rotation, Translation

Similarity transformation
e.g. Scaling

Affine transformation: non-uniform, but reserve parallel relationships
e.g. Shear

Projective transformation
e.g. Perspective



source
image

$$F(x, y)$$

$$(1-x)f_{00} + xf_{10}$$

$$(1-x)f_{01} + xf_{11}$$

$$\begin{bmatrix} A \\ B \end{bmatrix}$$

$$\begin{bmatrix} 1-y & y \end{bmatrix} \begin{bmatrix} f_{00} & f_{10} \\ f_{01} & f_{11} \end{bmatrix} \begin{bmatrix} 1-x \\ x \end{bmatrix}$$

BILINEAR
INTERPOLATION

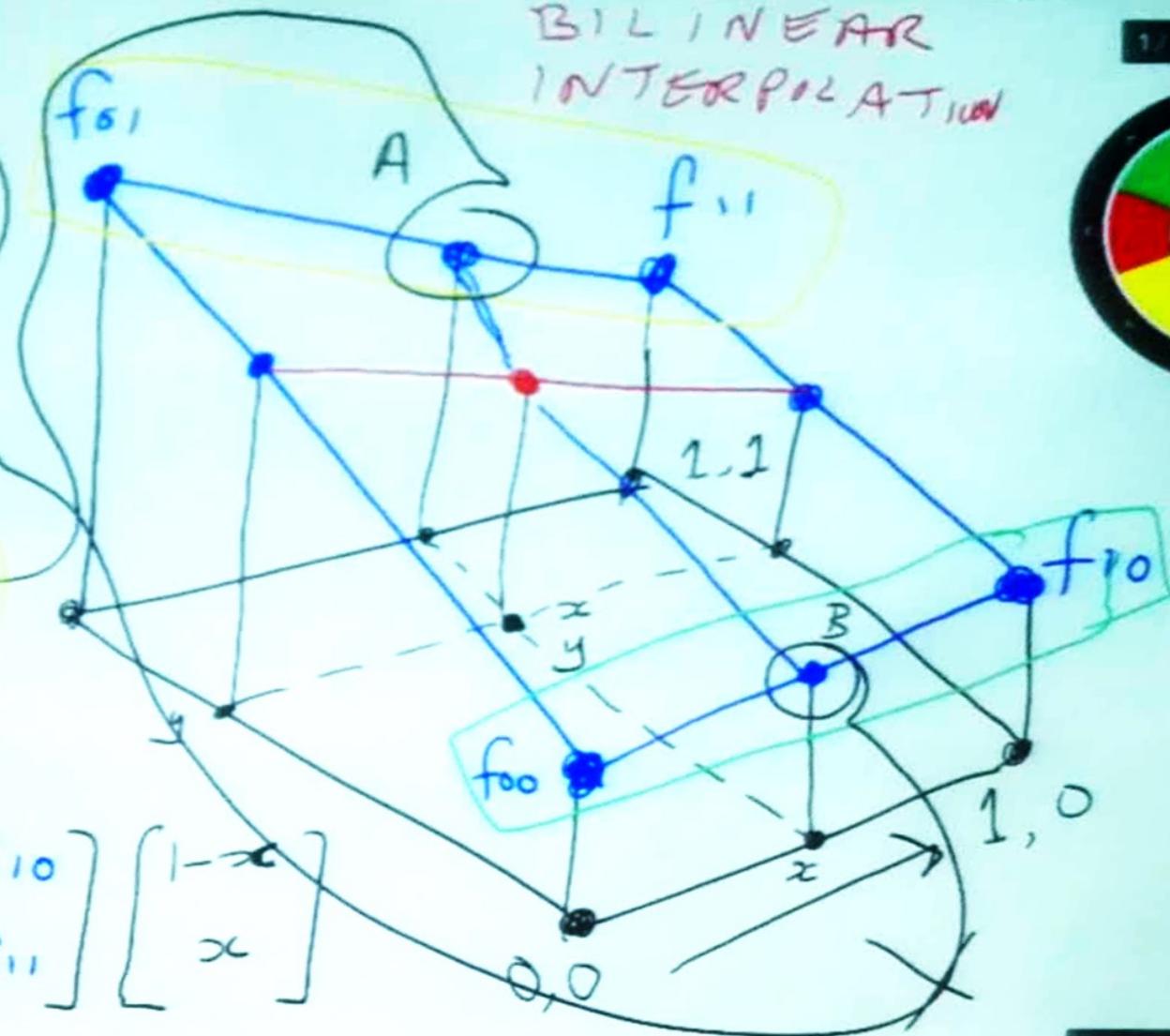


Image and video processing (EBU723U)

Colour Images

Dr. Miles Hansard

Today's agenda

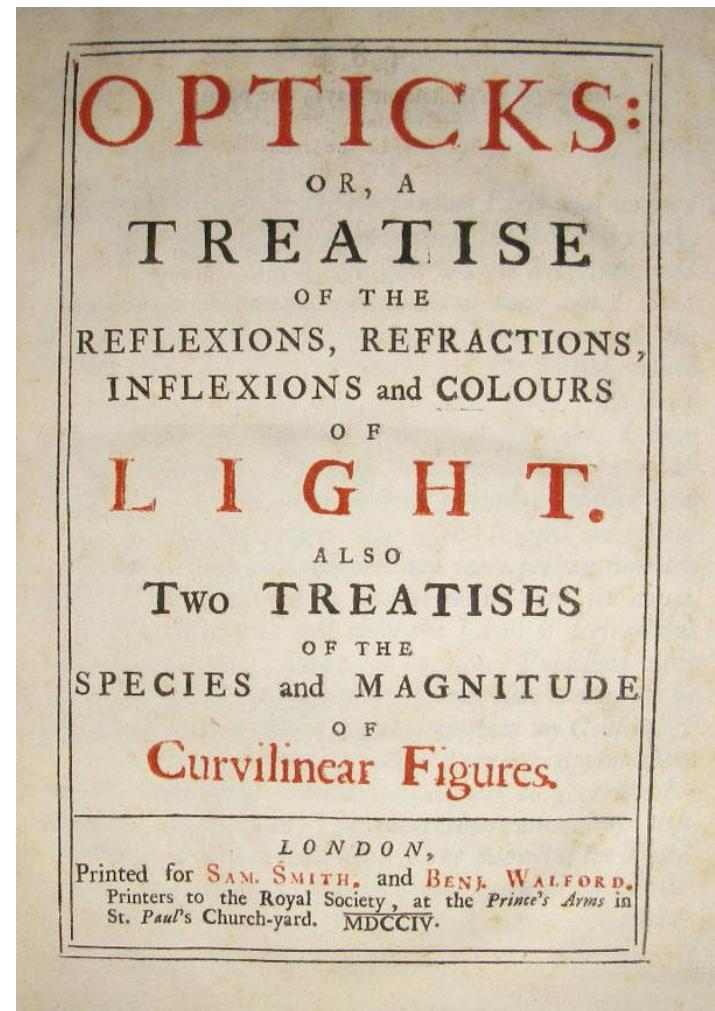
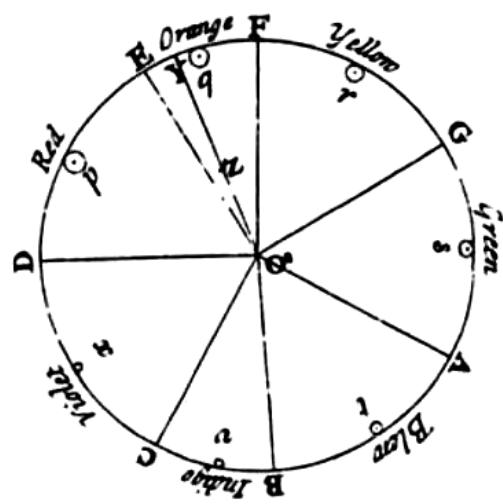
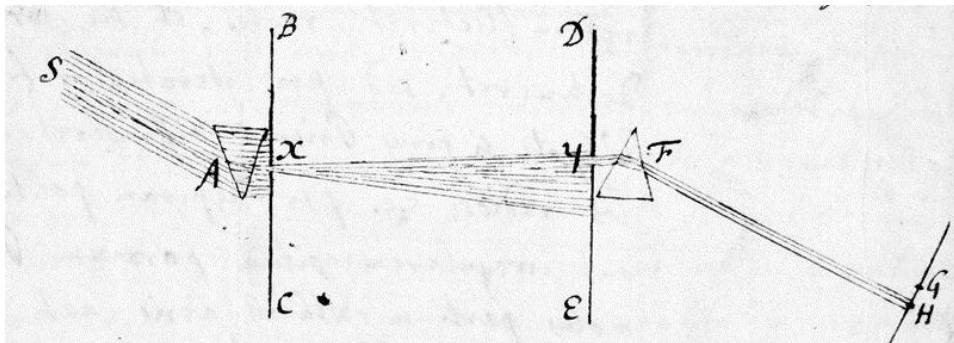
- Colour spaces
- Colour images
- PGM/PPM images

Today's agenda

- Colour spaces
- Colour images
- PGM/PPM images

History

- Isaac Newton (1642-1727)
- White light as a mixture



Beginning of colour science

- Goethe (1749-1832), colour perception
- Grassman (1809-1877), laws of colour mixing
- Maxwell (1831-1879), colour photography

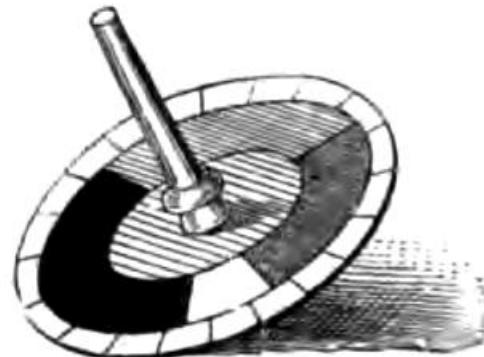
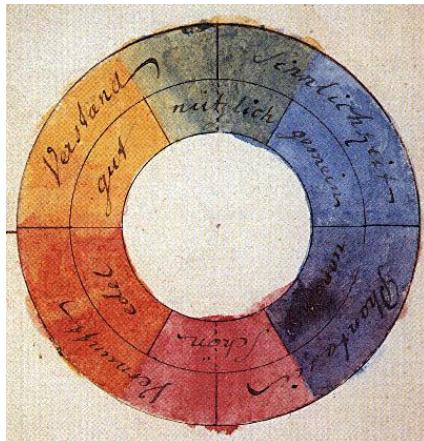
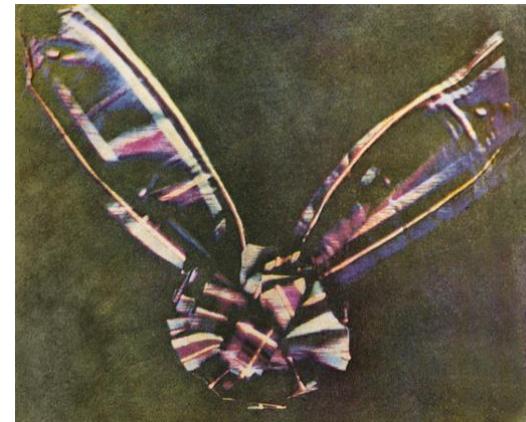


Fig. 83. The Color-top.



Some definitions...

- JND (Just Noticeable Distance)
 - we can distinguish ~7 million colors when samples placed side-by-side (JNDs)
 - ~128 fully saturated hues are distinct
 - Human Visual System (HVS) is
 - less discriminating for less saturated light
 - less sensitive for less bright light

Colour models

- **Hardware-oriented models:** not intuitive
 - RGB used with colour CRT monitors
 - YUV the broadcast TV colour system
 - CMY (cyan, magenta, yellow) for colour printing
 - CMYK (cyan, magenta, yellow, black) for colour printing
- **User-oriented models**
 - HSV (hue, saturation, value)
also called HSB (hue, saturation, brightness)
 - HLS (hue, lightness, saturation)
 - The Munsell system
 - CIE L*a*b*

Some definitions...

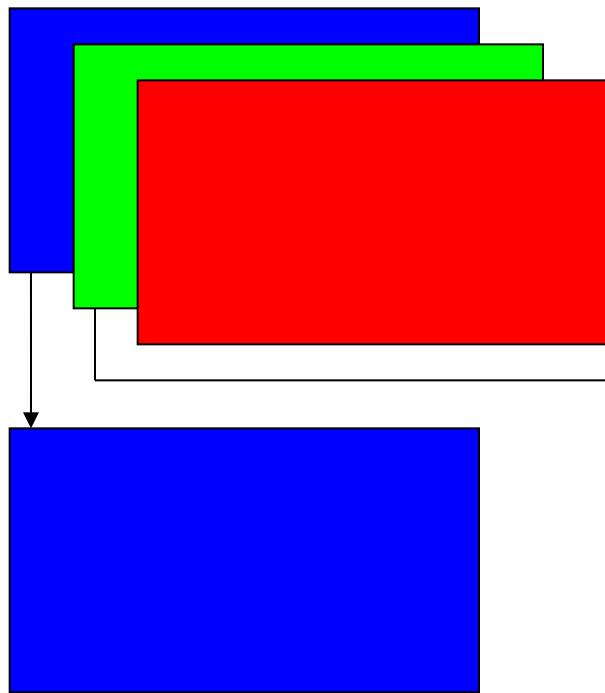
- Lightness
 - embodies the achromatic notion of perceived intensity of a **reflecting** object
- Brightness
 - is used instead of lightness to refer to the perceived intensity of a self-luminous (i.e., **emitting** rather than reflecting light) object, such as a light bulb, the sun, or a CRT

Colour spaces

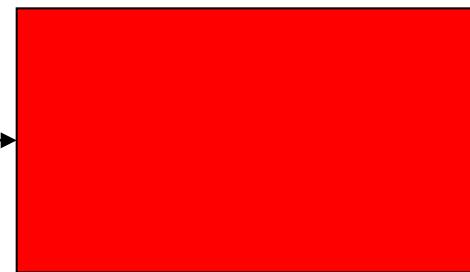
- Lab, RGB, YUV, CMY

Channels in colour images

3 samples per pixel



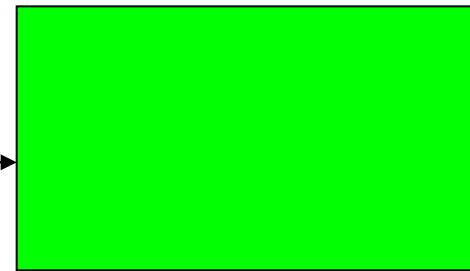
Red channel of image
1 sample per pixel



Blue channel of image
1 sample per pixel



Green channel of image
1 sample per pixel



Colour space representation

RGB



L



a



b



Chrominance low-pass filtering

Original



L



a



b



LP



LP



Result

L



a'



b'

Luminance low-pass filtering

Original



L



a



b



LP



Result

a



b



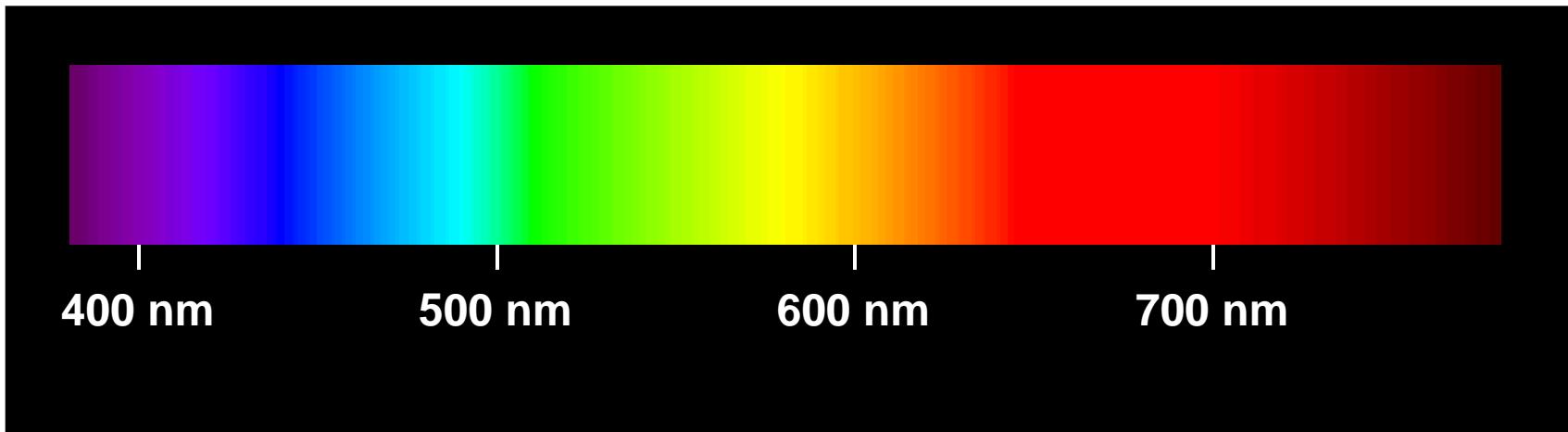
Light as a waveform

- Light
 - can be split into its component **wavelengths** and **intensity**
 - wavelength of visible light ~ between **400nm** and **700nm**
 - most light we see is a **combination** of many wavelengths (not a single wavelength)
- Human perception of colour
 - derives from the eye's responses to three different groups of wavelengths
 - those corresponding to **red, green and blue (RGB)**

Physical representation

Spectral for pure colours

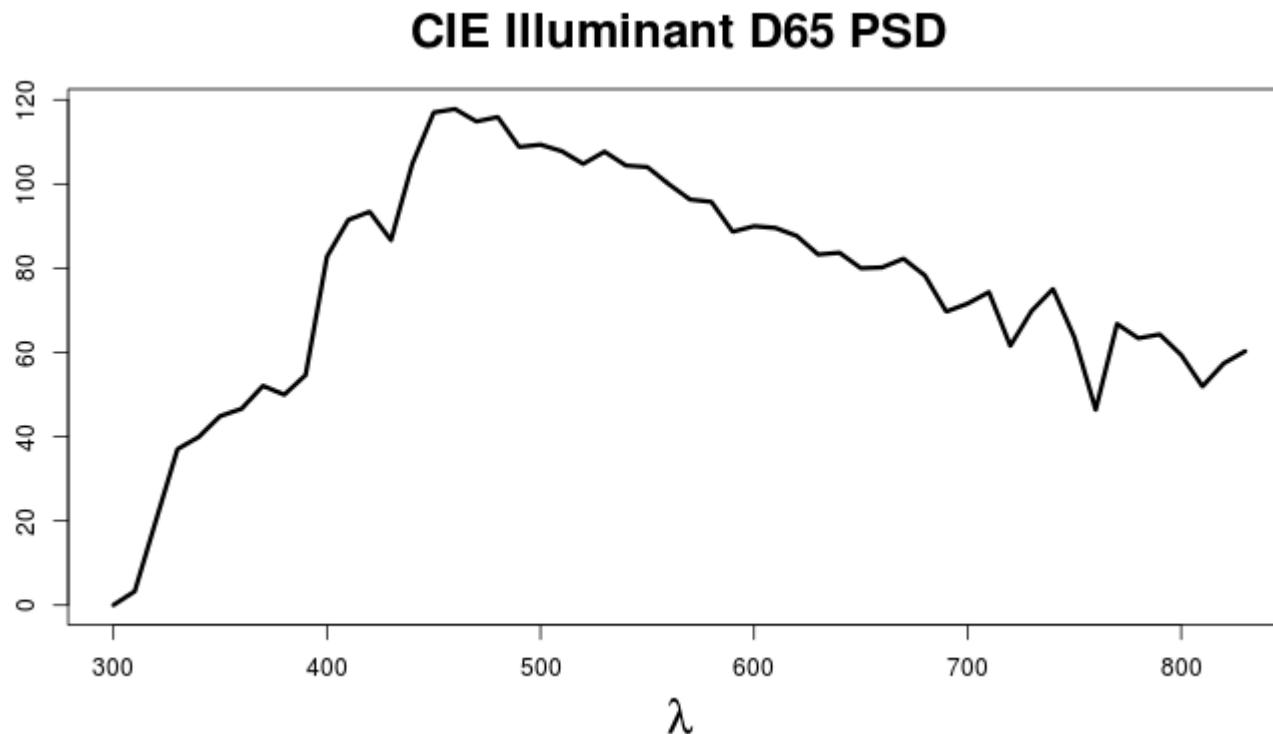
- Visible spectrum



- Frequency representation
- Colour spaces

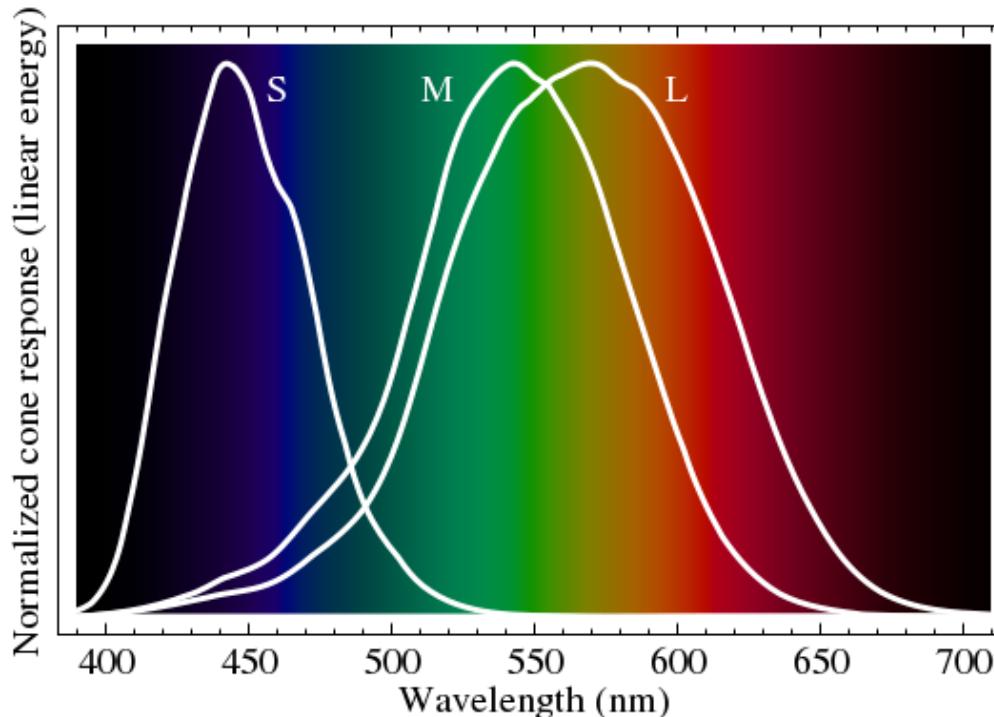
Spectral power distribution

- This is the physical spectrum of typical daylight



Trichromatic theory of colour vision

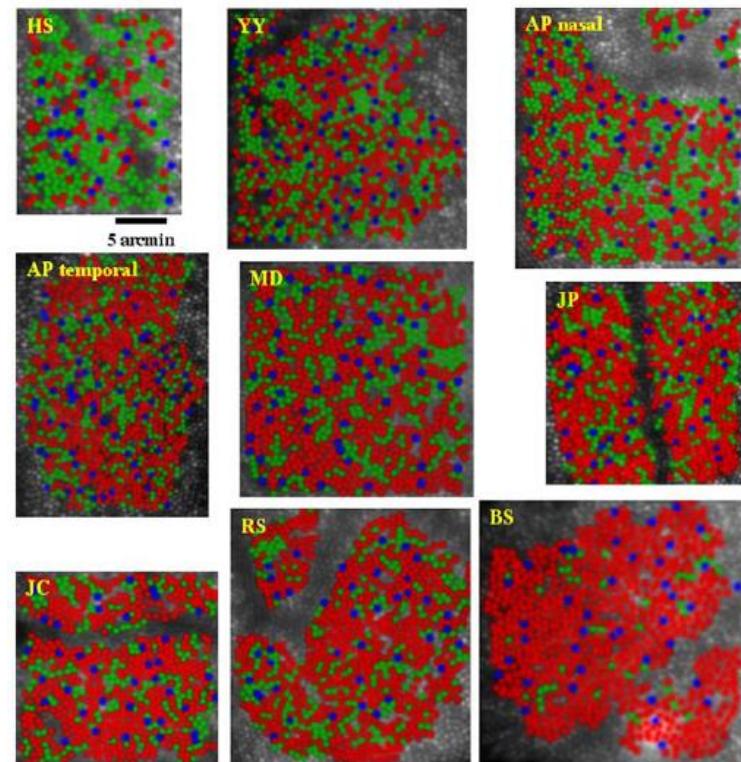
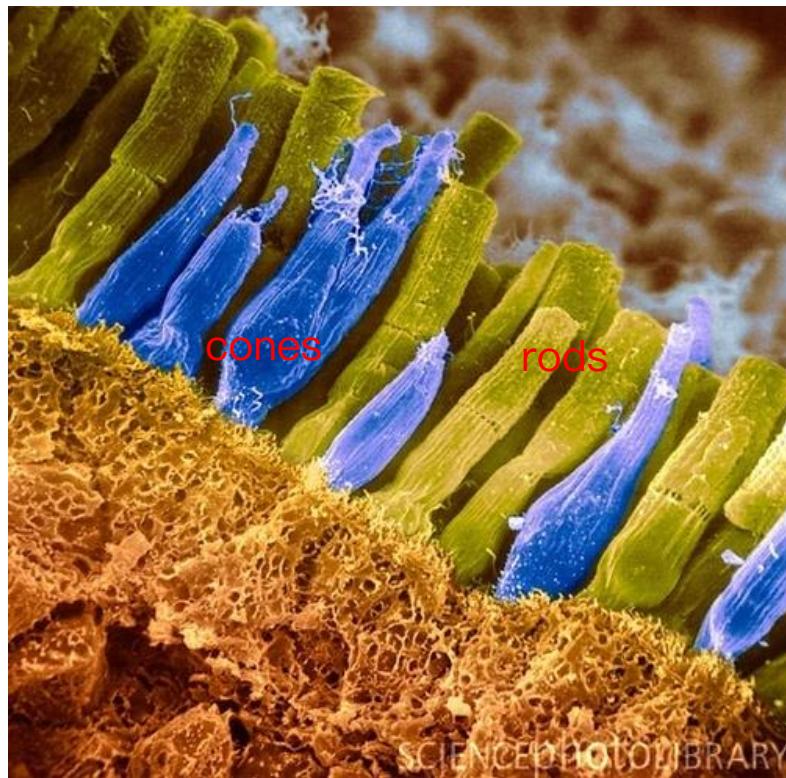
- ANY sensation of colour can be produced
 - by **mixing** together suitable amounts of **THREE** colours
 - R G B are called the **additive primary colours**
 - These are related to the three types of cone cells in the retina.



The retina: simple model

- The eye functions on the same principle as a camera
- Each neuron is either a rod or a cone
- Rods (130 M)
 - contain the elements that are sensitive to light intensities
 - are not sensitive to color
- Cones (6.5 M) → 3 types
 - red, green and blue (or *long, medium & short wavelength*).
 - each type responds differently to various frequencies of light

Retinal cone mosaic



Spatial acuity and color vision

RGB Image

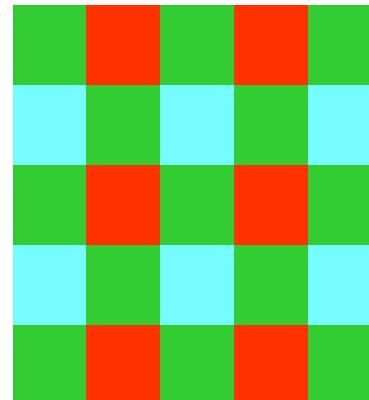
Red

Green

Blue



1 photoreceptor by spatial position

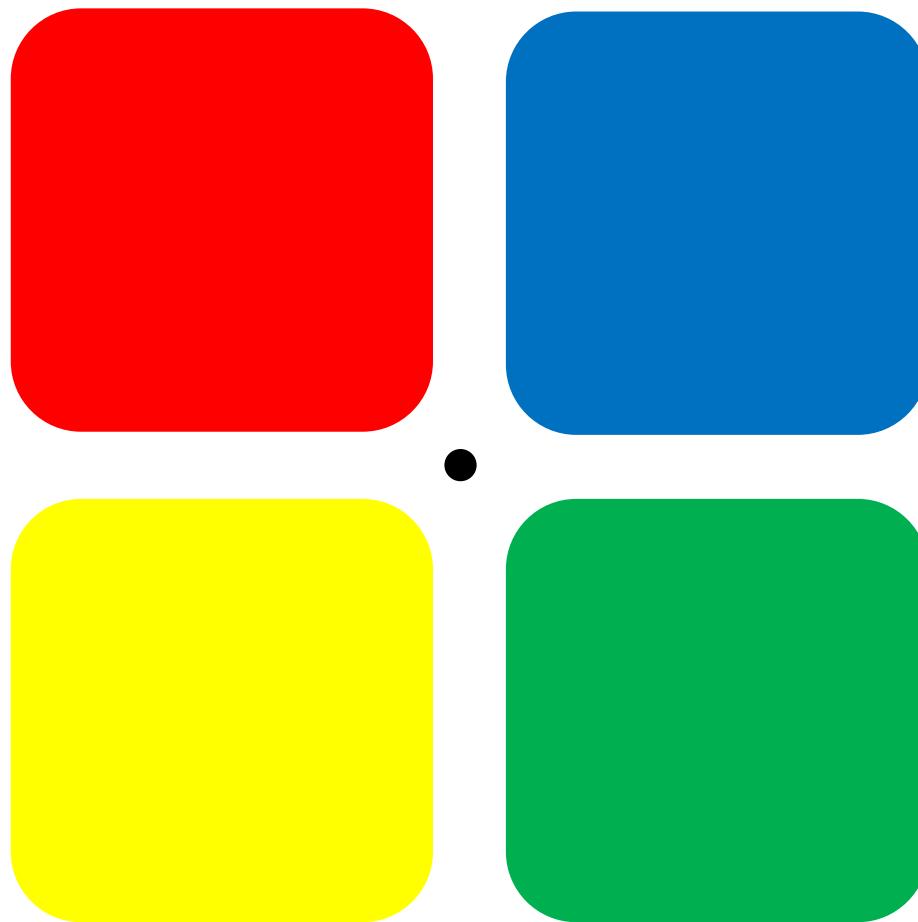


Spatial multiplexing of colors

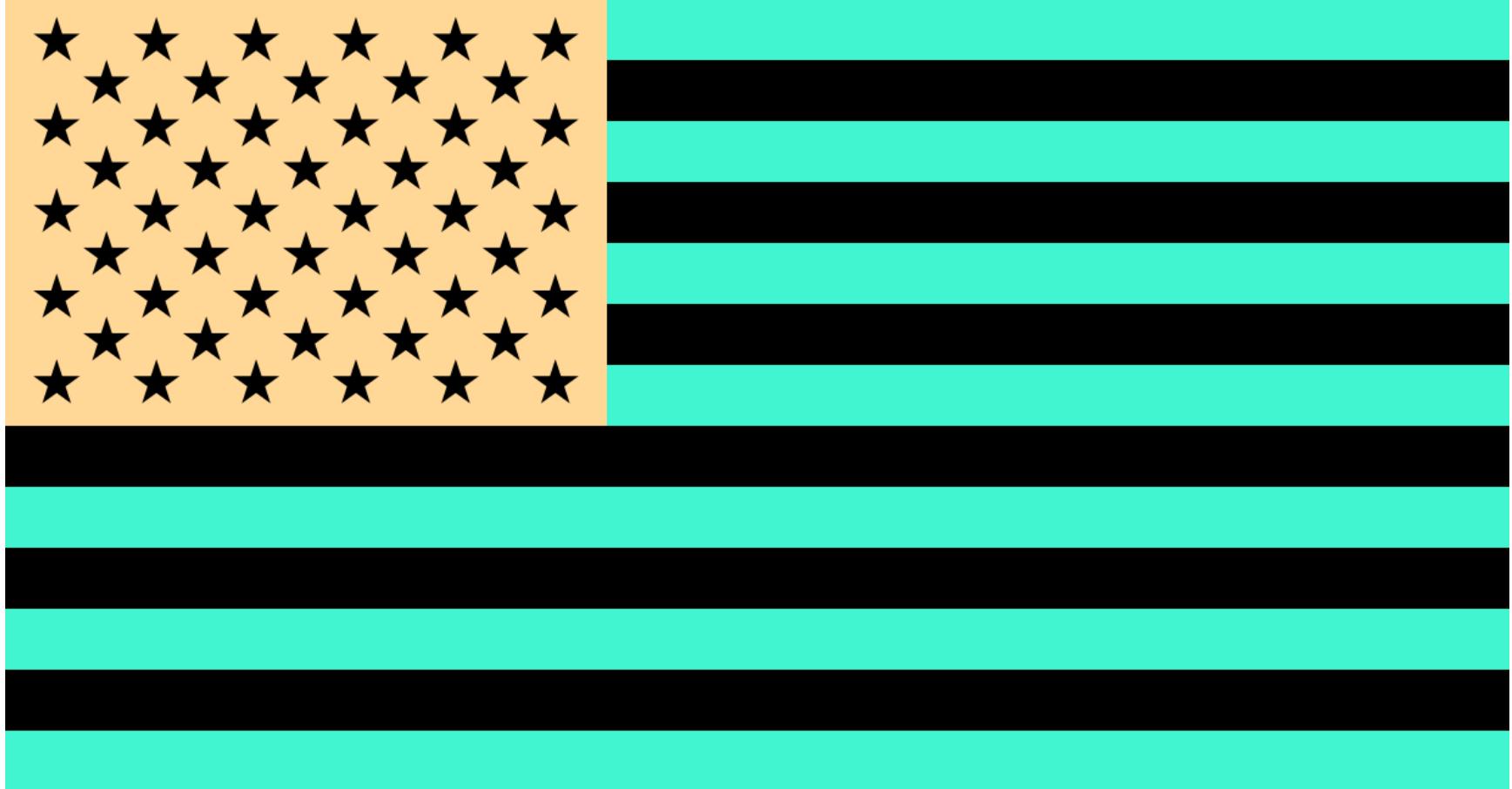
Properties of the Human Visual System

- The eye
 - more **sensitive** to changes in brightness than colour
 - unable to perceive brightness levels above or below certain **thresholds**
 - can't distinguish minor changes in brightness or colour
- Note: certain ranges of brightness or colour are more important visually than others
 - more sensitive to minor changes in shades of **green** than other colours
- Sensitivity of the eye is not linear

Colour after-images



Opponent colours demo



Impossible colours

- We can see ‘reddish-yellow’ = orange
- Or ‘greenish-blue’ = turquoise
- So what is ‘greenish-red’ ?
- How about ‘bluish yellow’ ?
- There must be a biological explanation...

Opponent-process theory

- Hypothesis that the cone-signals are re-coded as sums and differences.
- Luminance channel, and two opponent-colour channels:
 - $L = R + G + B$
 - $S = G - R$
 - $T = B - Y = B - (R + G)$
- So G and R ‘cancel each other out’, as do B and Y.
- Both Trichromatic theory (retina) and Opponent-process theory (brain) are needed.

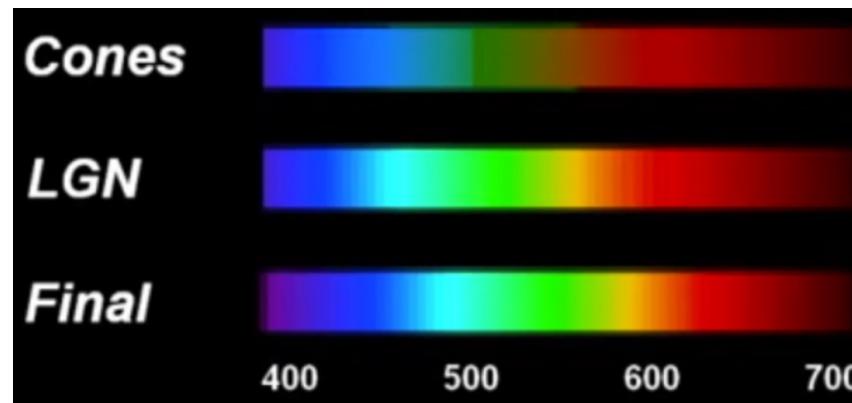
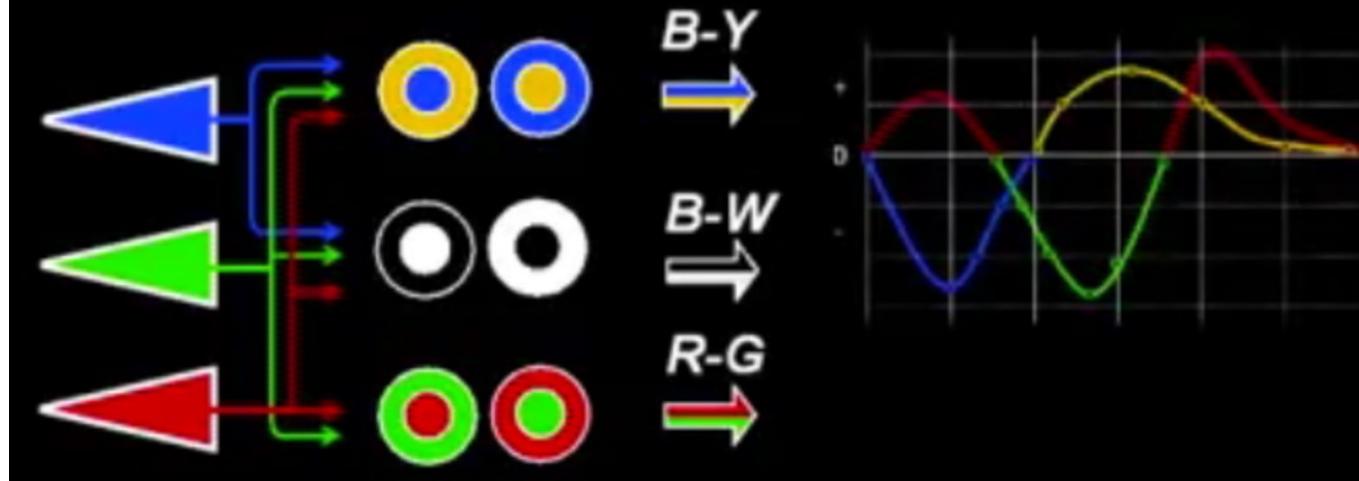
Stage 1: Cone Oppon Retina

$$\begin{aligned}T &= B - Y \\&= -R - G + B\end{aligned}$$

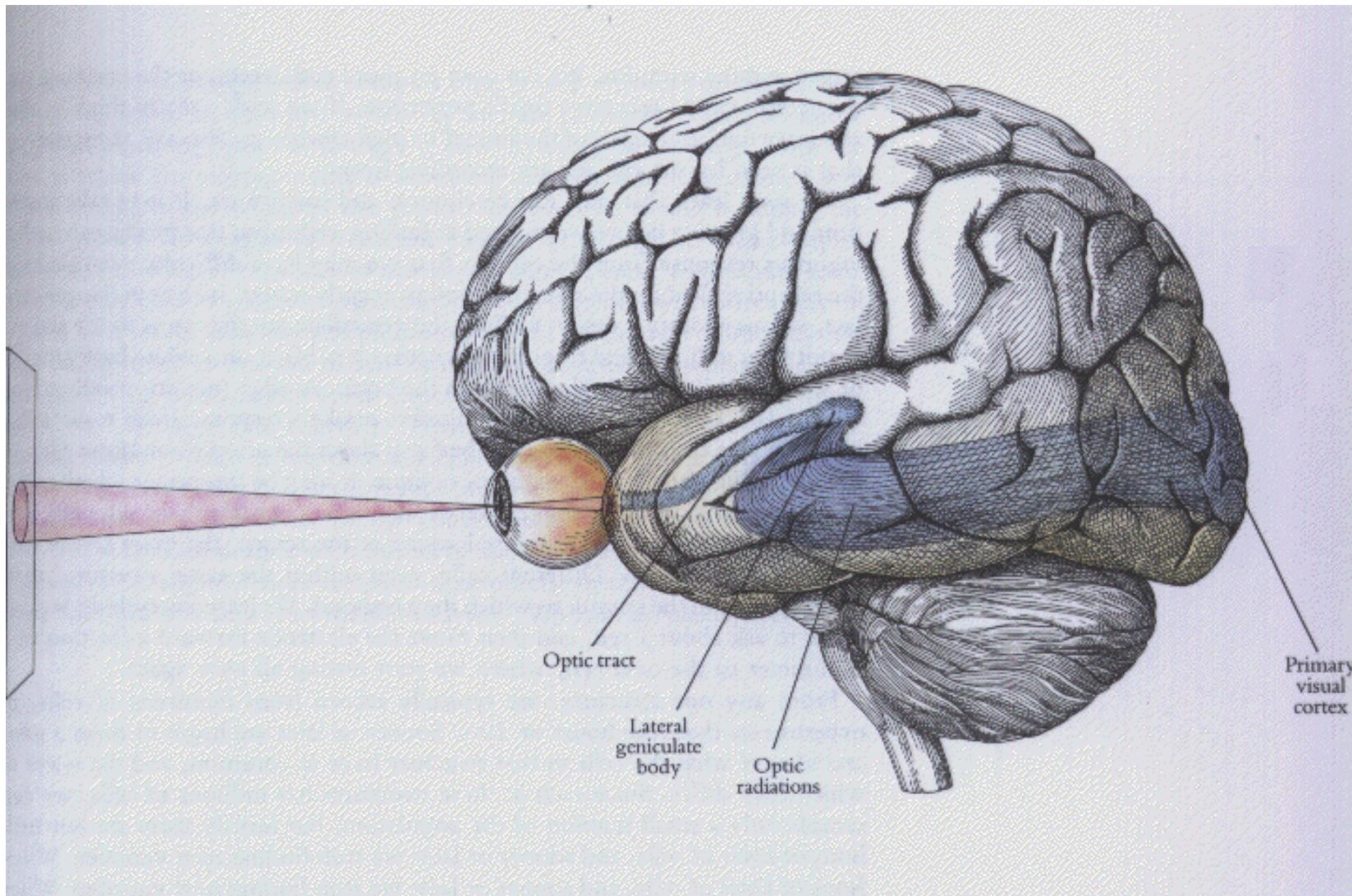
$$L = R + G + B$$

$$S = -R + G$$

Stage 2: Color Oppon Cortex

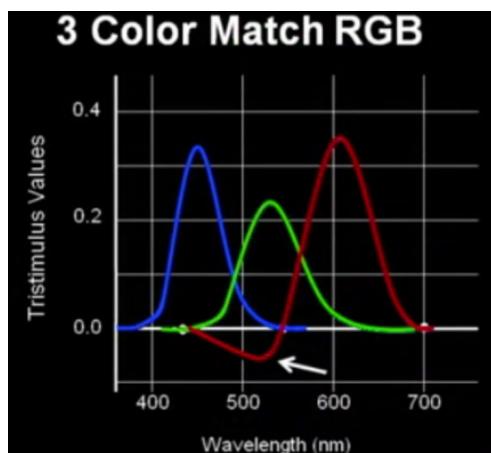


Human Visual System

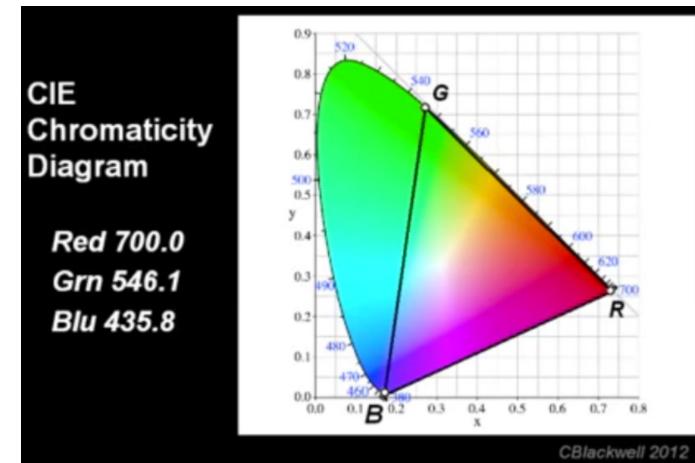
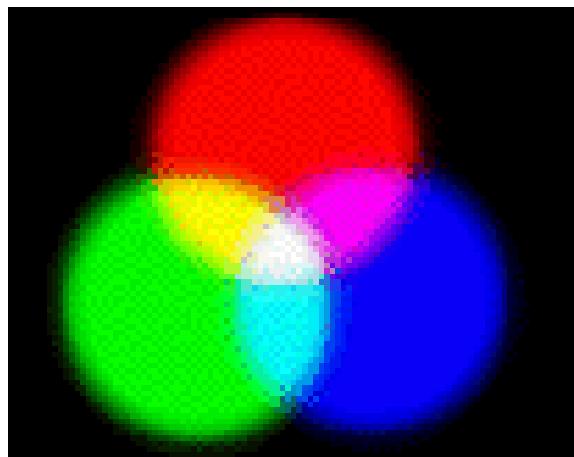


Primary and secondary colours

- Primary colours of light are Red, Green and Blue.
- **Misconception** – the three standard primaries, when mixed together in various intensity proportions can produce *all* visible colours.
- This is not true unless the wavelength is also allowed to vary – no longer have the three primaries.



the negative values in red!

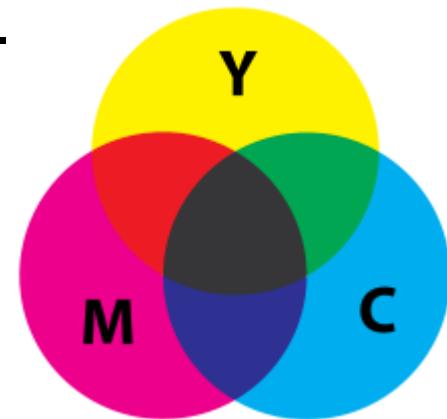


Additive and subtractive colours

- The three primary colours R, G, B are the three additive primaries.
- Adding them all together gives white light.
- These three colours are used in *light producing devices* to produce different colours. Eg. TV
- However, when you print some colour, the paper is *not* producing light waves – it is *absorbing* light.
- So the colour we see is the one that *is not absorbed* by the ink on the page.
- Eg. Magenta absorbs green light, and we only see red and blue – magenta.
- So a primary colour of pigment is defined as one that *subtracts* a primary colour of light and reflects the other two.

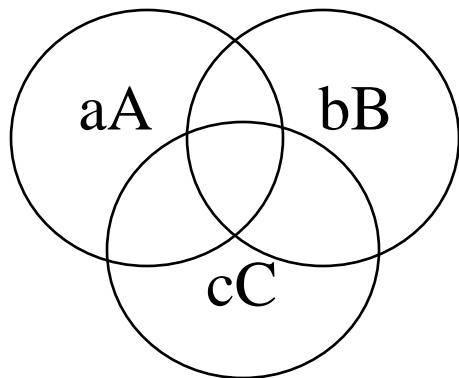
Subtractive primaries

- So a primary colour of pigment is defined as one that *subtracts* a primary colour of light and reflects the other two.
- Magenta – absorbs green and reflects red and blue.
- Cyan – absorbs red and reflects green and blue.
- Yellow – absorbs blue and reflects red and green.
- Mix them all together and you get black.

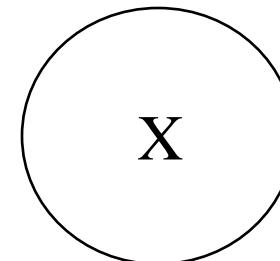


Additive primaries

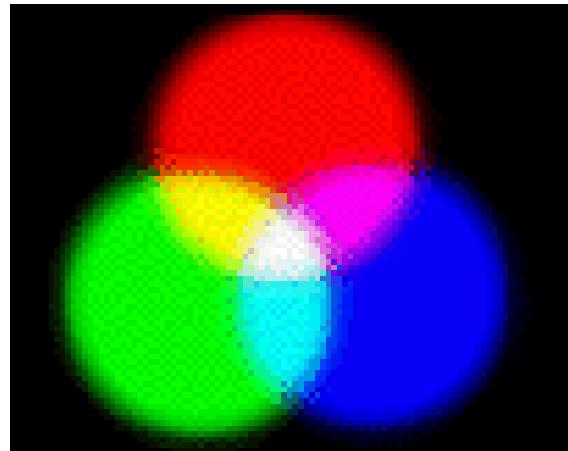
- Additive colours



$$X = aA + bB + cC$$

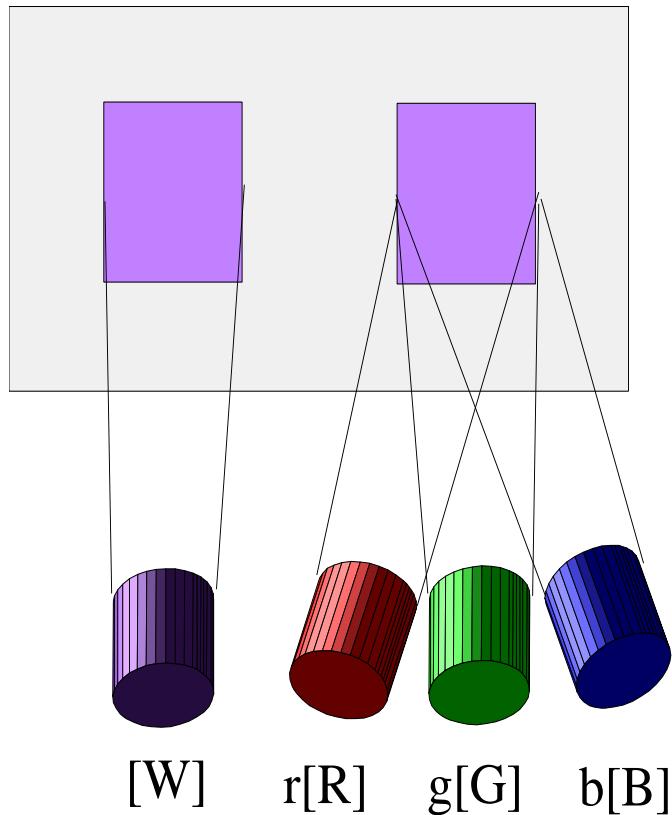


- Example: TV screen

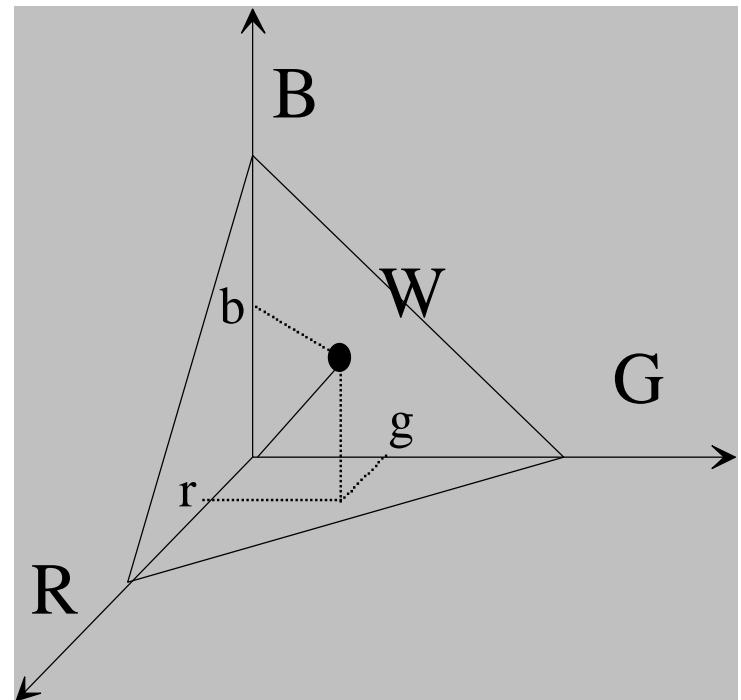


Colour mixture

Colour matching experiment



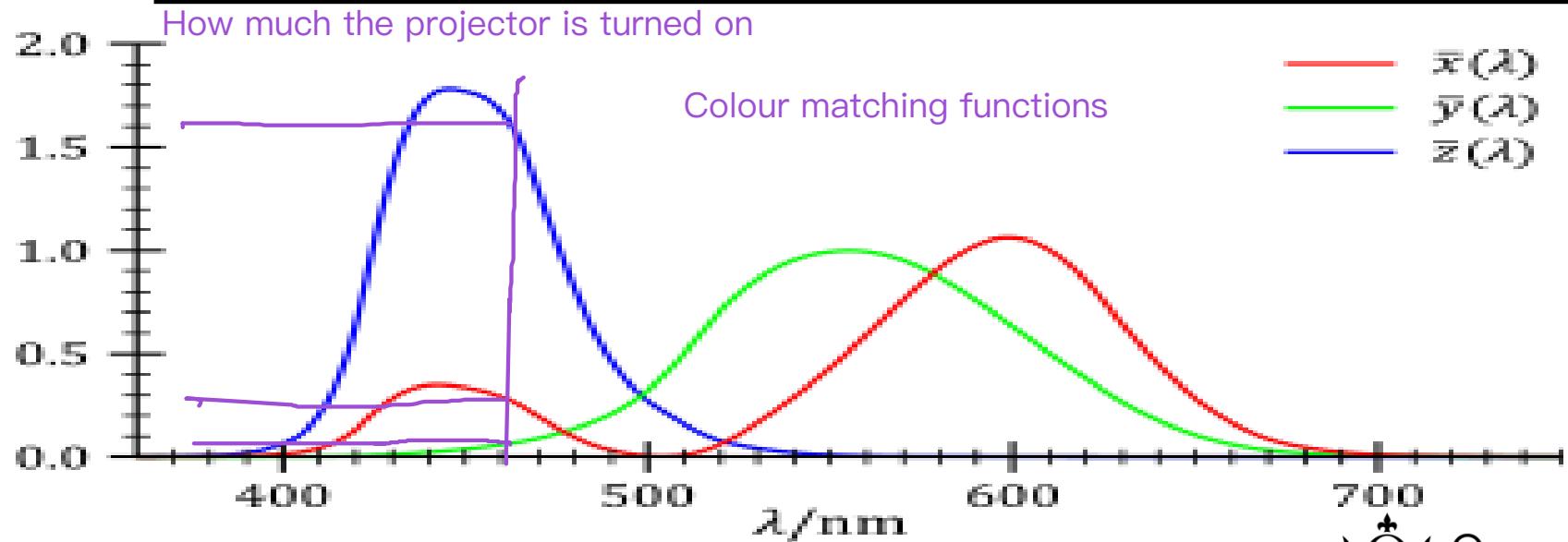
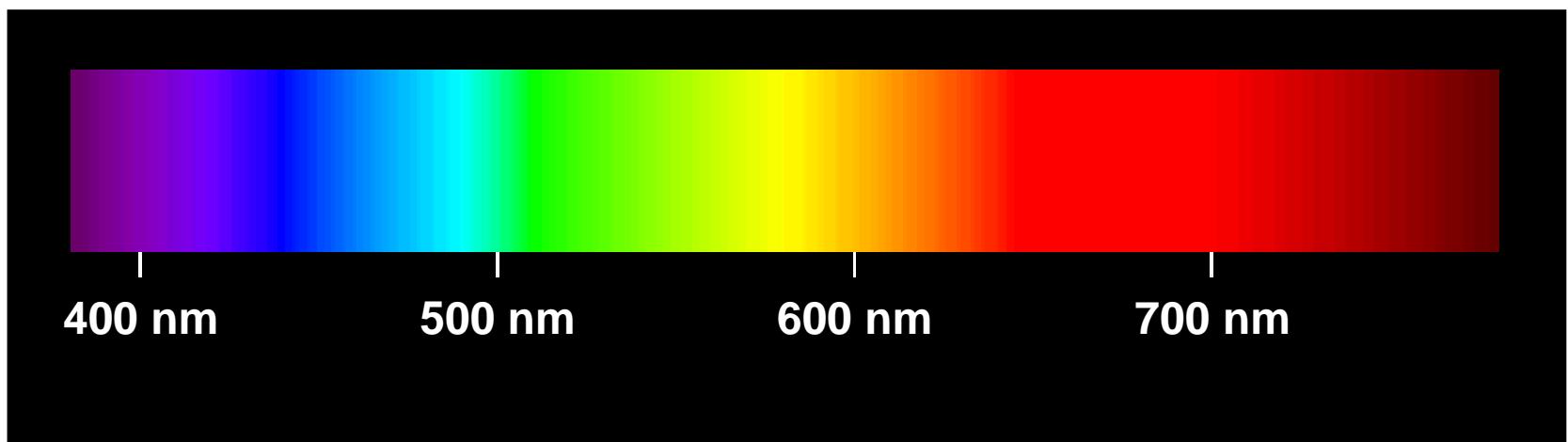
Vectorial colour space



Matching

$$[W] = r[R] + g[G] + b[B]$$

Experimental results



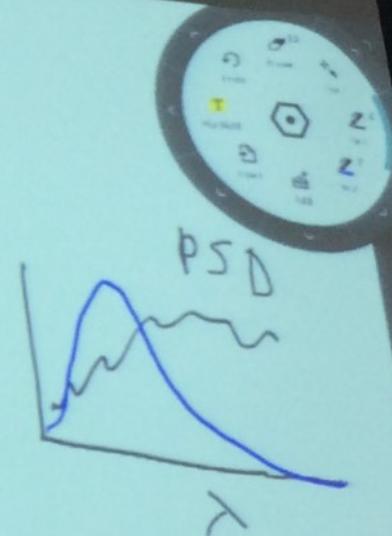
$$X = \int_{400\text{nm}}^{700\text{nm}} \bar{x}(\lambda) \text{PSD}(\lambda) d\lambda$$

$$Y = \int \bar{y}(\lambda) \text{PSD}(\lambda) d\lambda$$

$$Z = \int \bar{z}(\lambda) \text{PSD}(\lambda) d\lambda$$

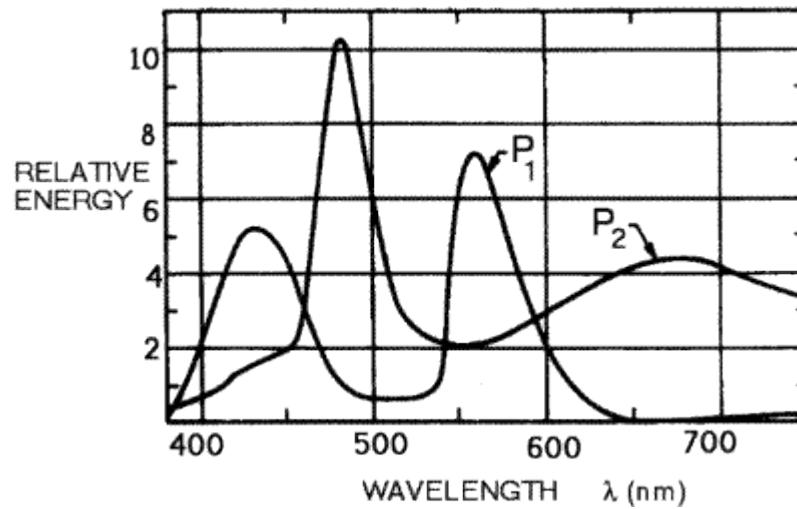
↑
color matching
functions

Physical
spectrum



Perceptual colour coordinates

- The dot-product (integral) of the three CIE basis functions with an input spectrum (PSD) gives us three perceptual coordinates for the corresponding colour
- If two different spectra have the same perceptual coordinates, then they are **metamers**
- Here are example metamers, P₁ and P₂



RGB-CIE colour model

- Standardization by the *Commission Internationale de l'Eclairage (CIE)* in 1931
 - The three primary sources:

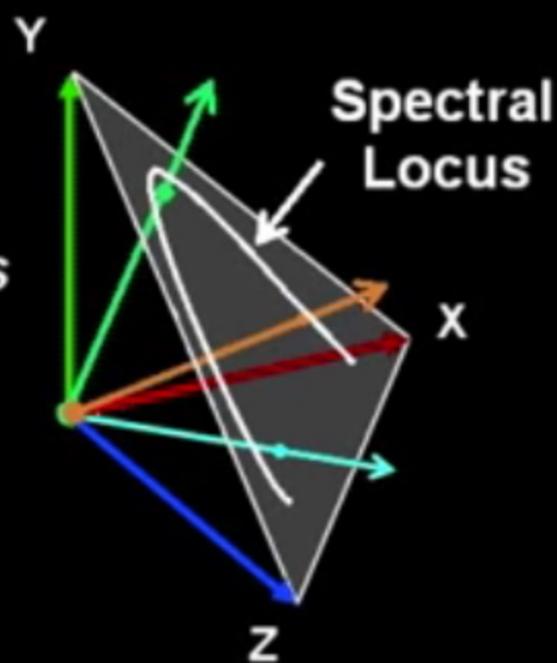
$$R_0(\lambda) = \delta(\lambda - 700\text{nm})$$

$$G_0(\lambda) = \delta(\lambda - 546.1\text{nm})$$

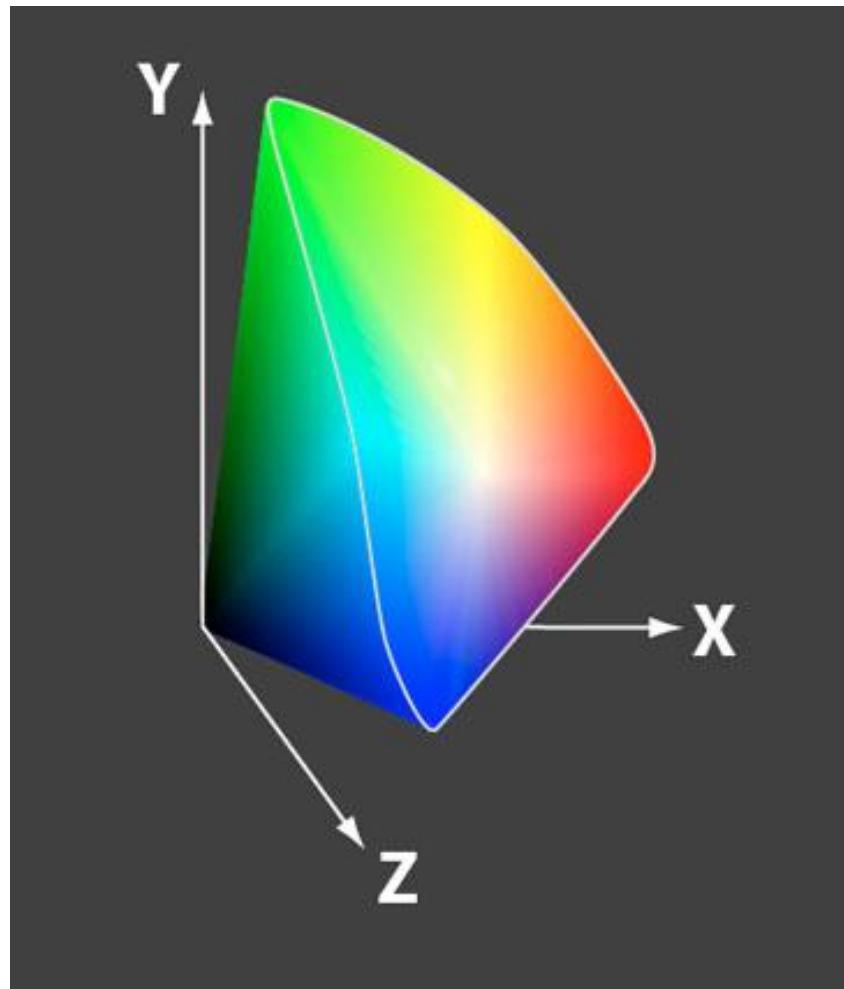
$$B_0(\lambda) = \delta(\lambda - 435.8\text{nm})$$

Color Space X-Y-Z

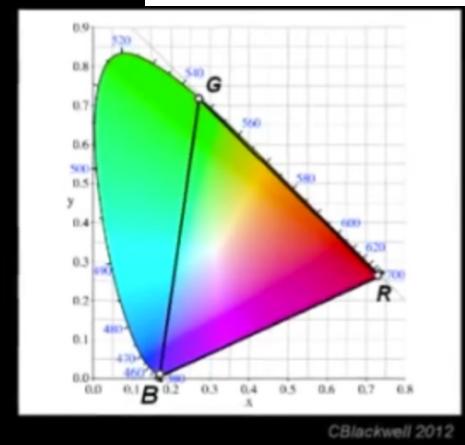
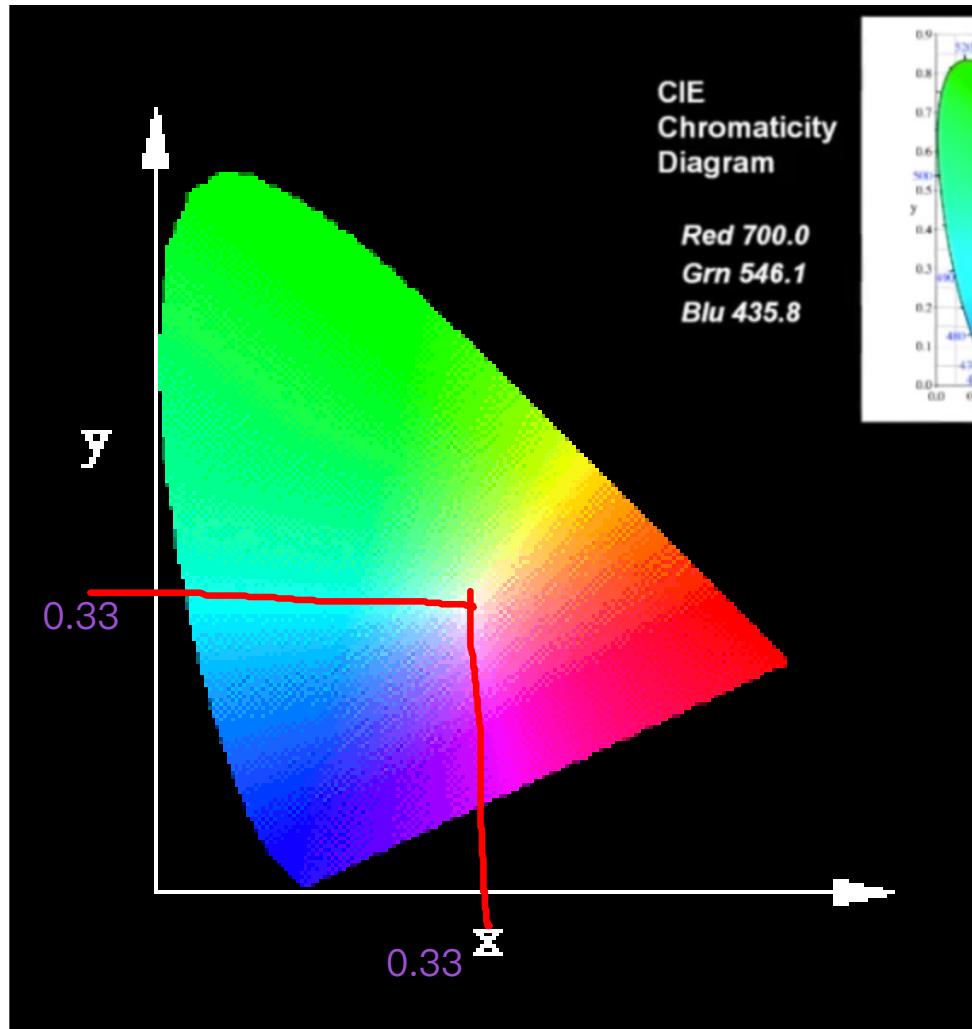
*Series of
Tristimulus Vectors
Map Out the
Chromaticity
Diagram*



CIE Colour XYZ



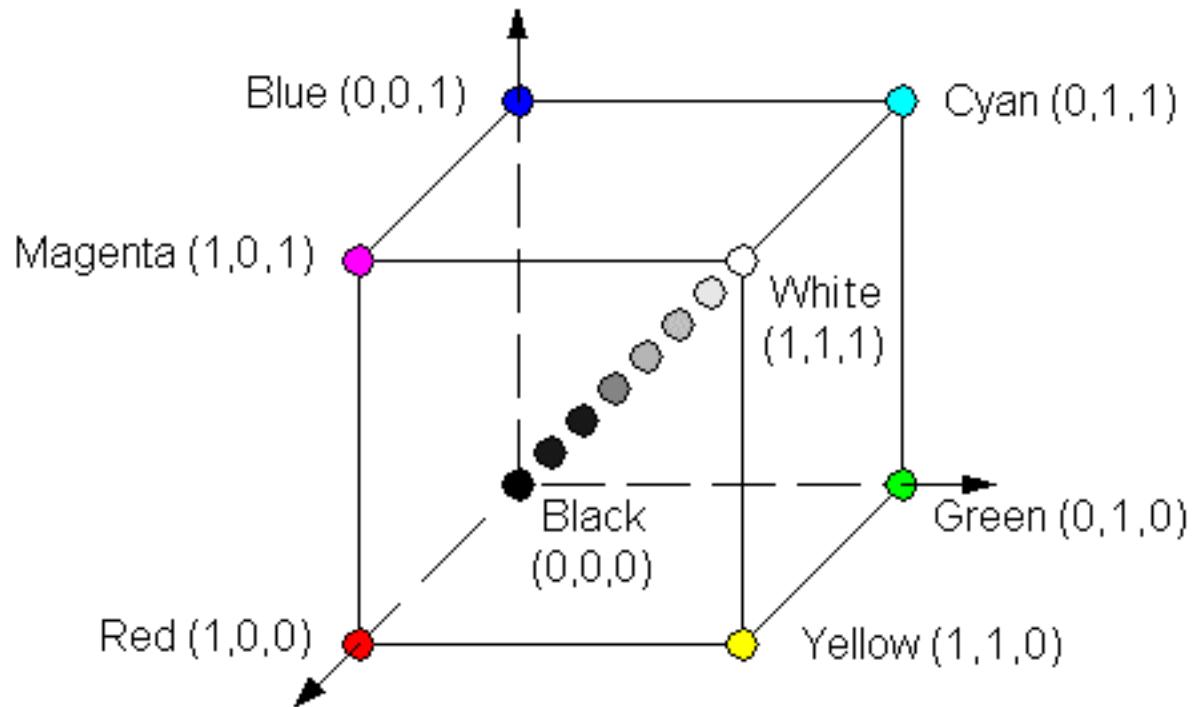
CIE xy chromaticity diagram



Not perceptually uniform

RGB-CIE

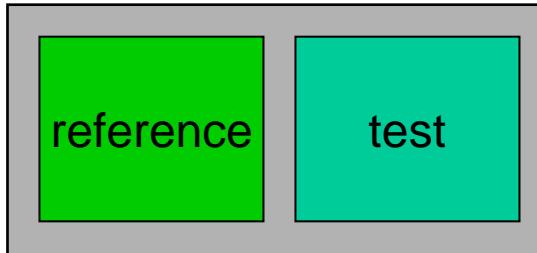
- RGB cube



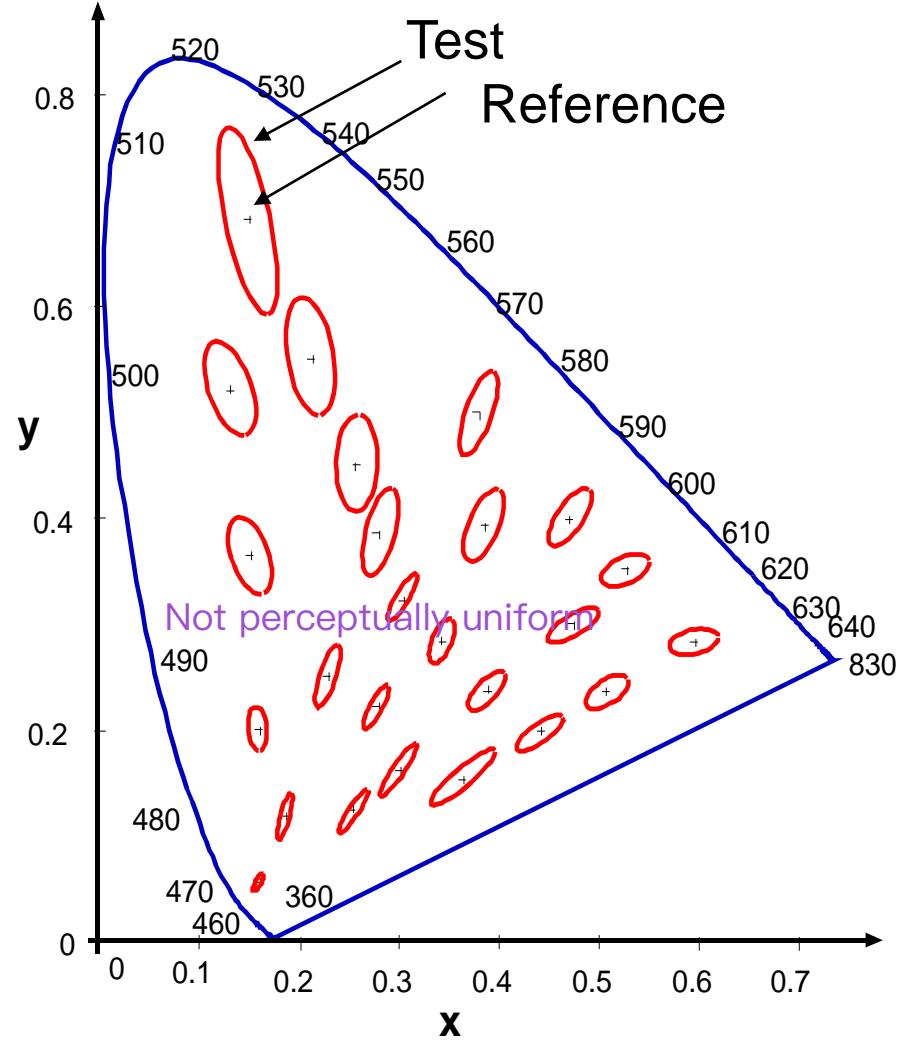
$L^*a^*b^*$

- Standardized by the *Commission Internationale de l'Eclairage (CIE)* in 1976
 - Perceptually uniformity for **chrominances and luminance**
 - Non-linear transformation from CIE
- L = luminosity
- a and b are chrominance

Colour non-linearity: Mac Adam ellipses

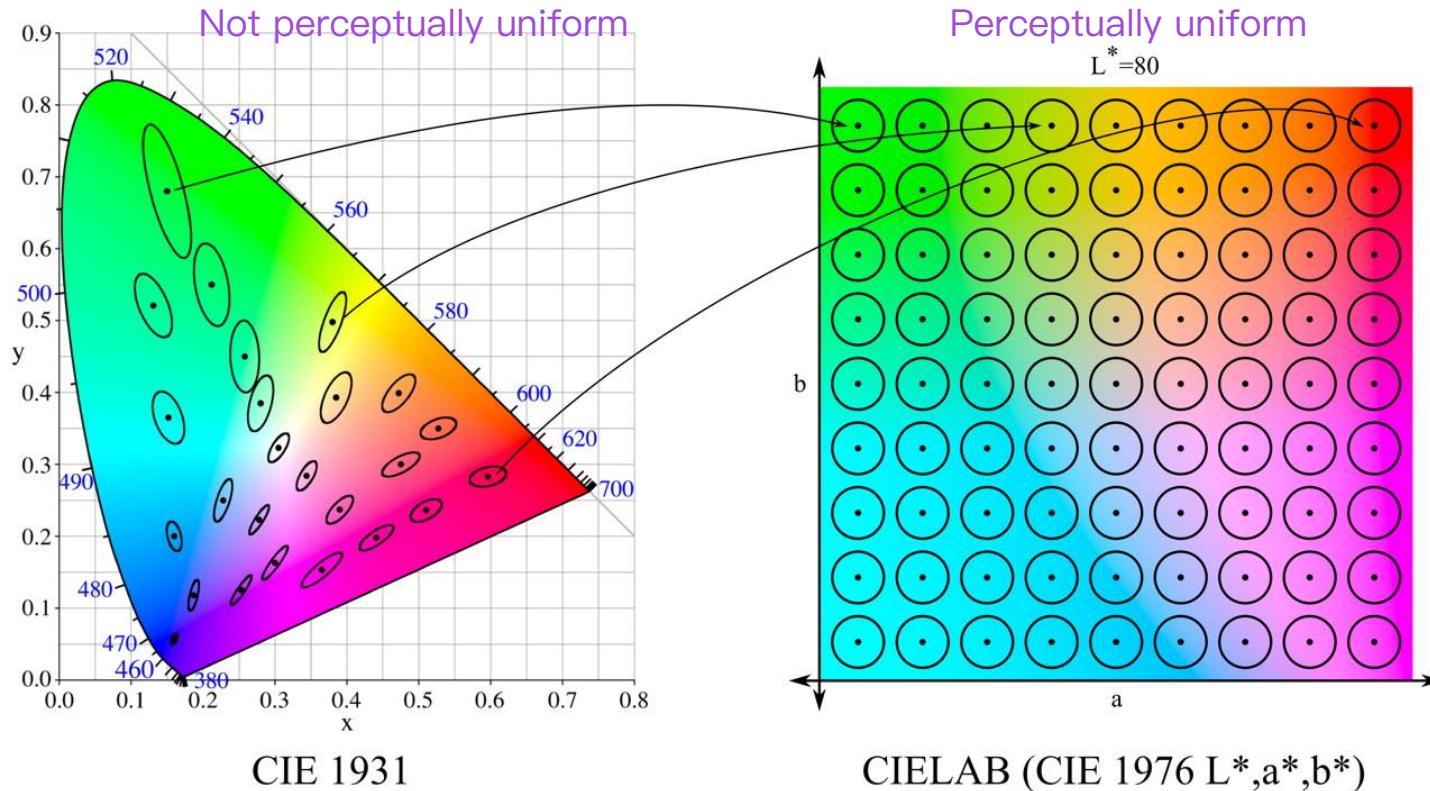


Just noticeable difference JND



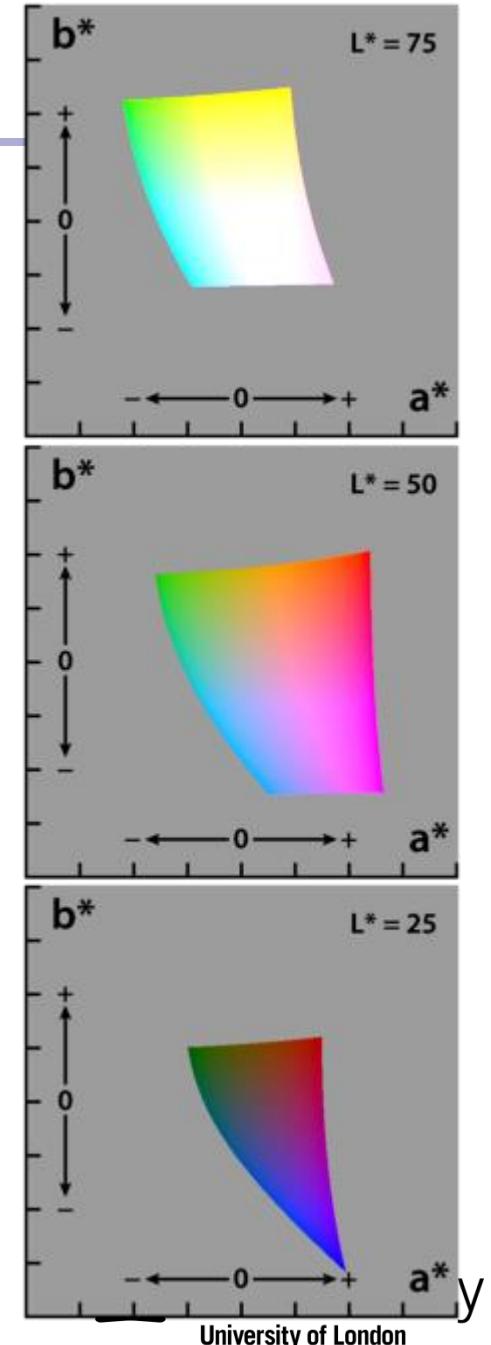
CIE to Lab mapping

- MacAdam ellipses are circles in Lab space
- The mapping is non-linear



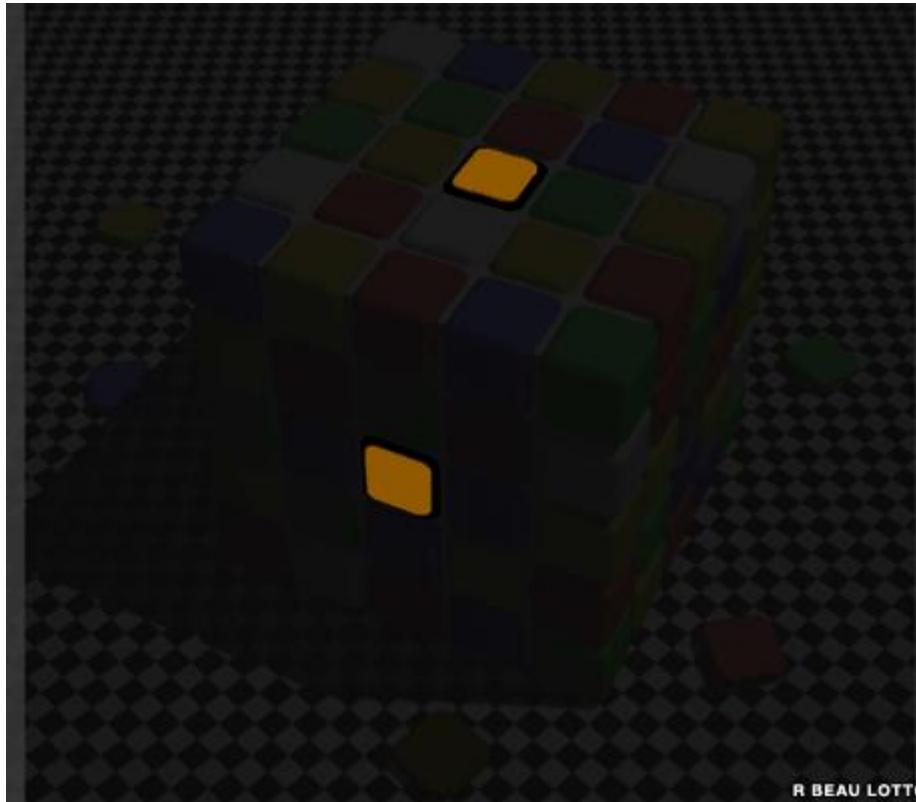
$L^*a^*b^*$ colour space

- Perceptually uniform.
- Space must be recalculated for different intensities.
- A certain distance d_1 has the same perceived distance anywhere in the colour space



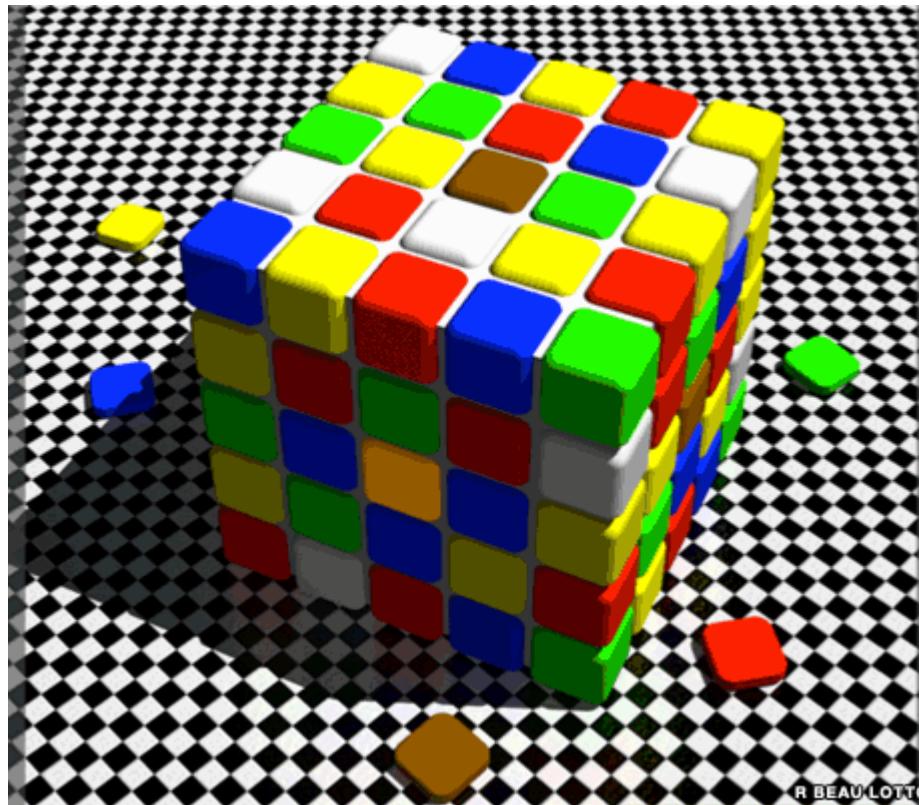
CIE Limitations

- These patches have the same colour



CIE limitations

- The CIE experiments do not explain everything



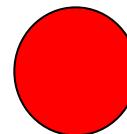
HSV colour model

- HSV models colour in terms of
 - HUE
 - the **dominant wavelength**
i.e. where most of the energy of the light is concentrated
 - hues usually identified by names:
mixtures of red, yellow, green and blue
 - SATURATION
 - a measure of the **colour's purity** or intensity
 - the presence of other hues makes the colour paler
 - BRIGHTNESS
 - a measure of how **light or dark** the colour is
- HSV corresponds more closely to the way humans think about colour than RGB

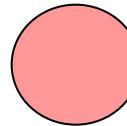
Colours

- Saturation
 - refers to how pure the color is
(i.e., how much white/grey is mixed with it)

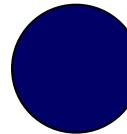
- red is highly saturated



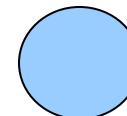
- pink is relatively unsaturated



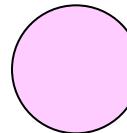
- royal blue is highly saturated



- sky blue is relatively unsaturated

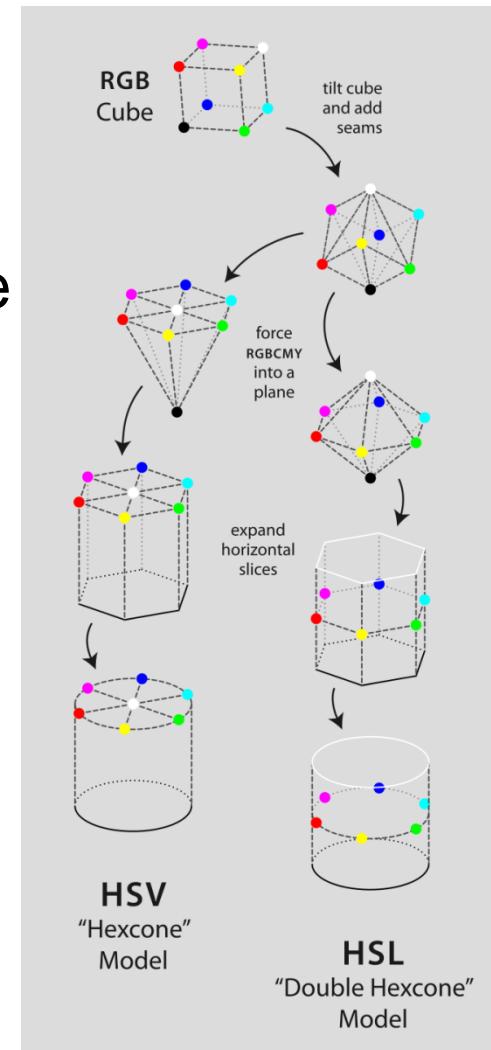
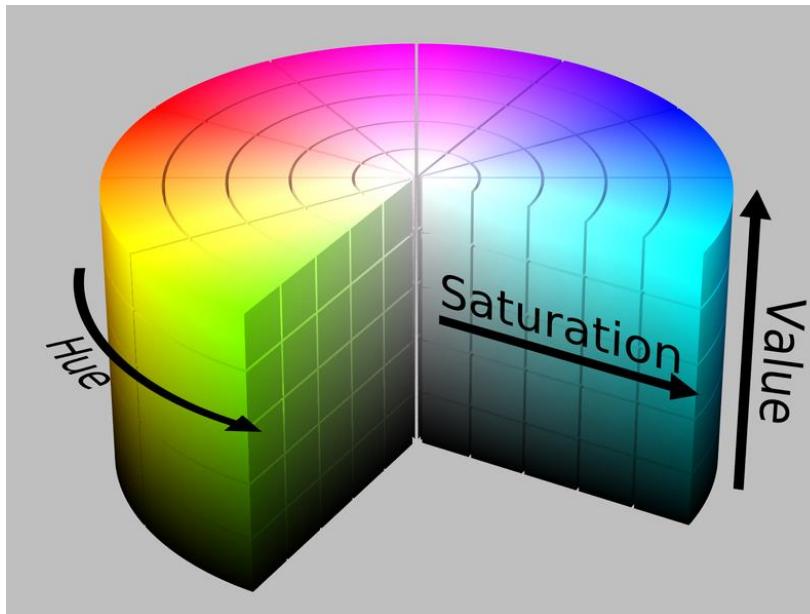


- pastels are less vivid, less intense

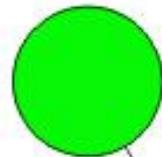


Transformation from RGB to HSV

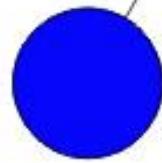
- Tilt the RGB cube onto the black corner
- Connect the R,G,B corners to White
- Squash R,G,B,C,M,Y corners into a plane
- Add a vertical Value dimension



Green

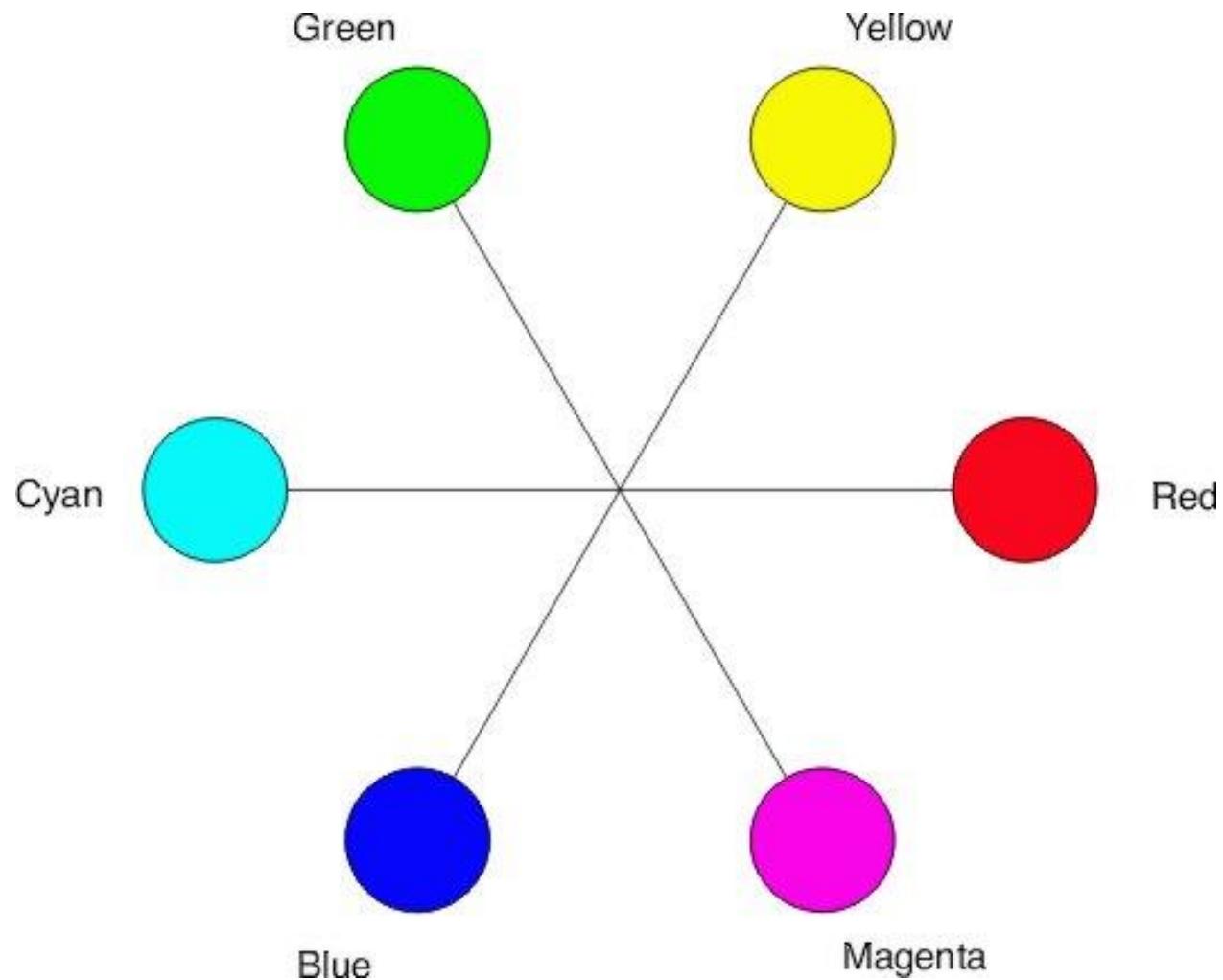


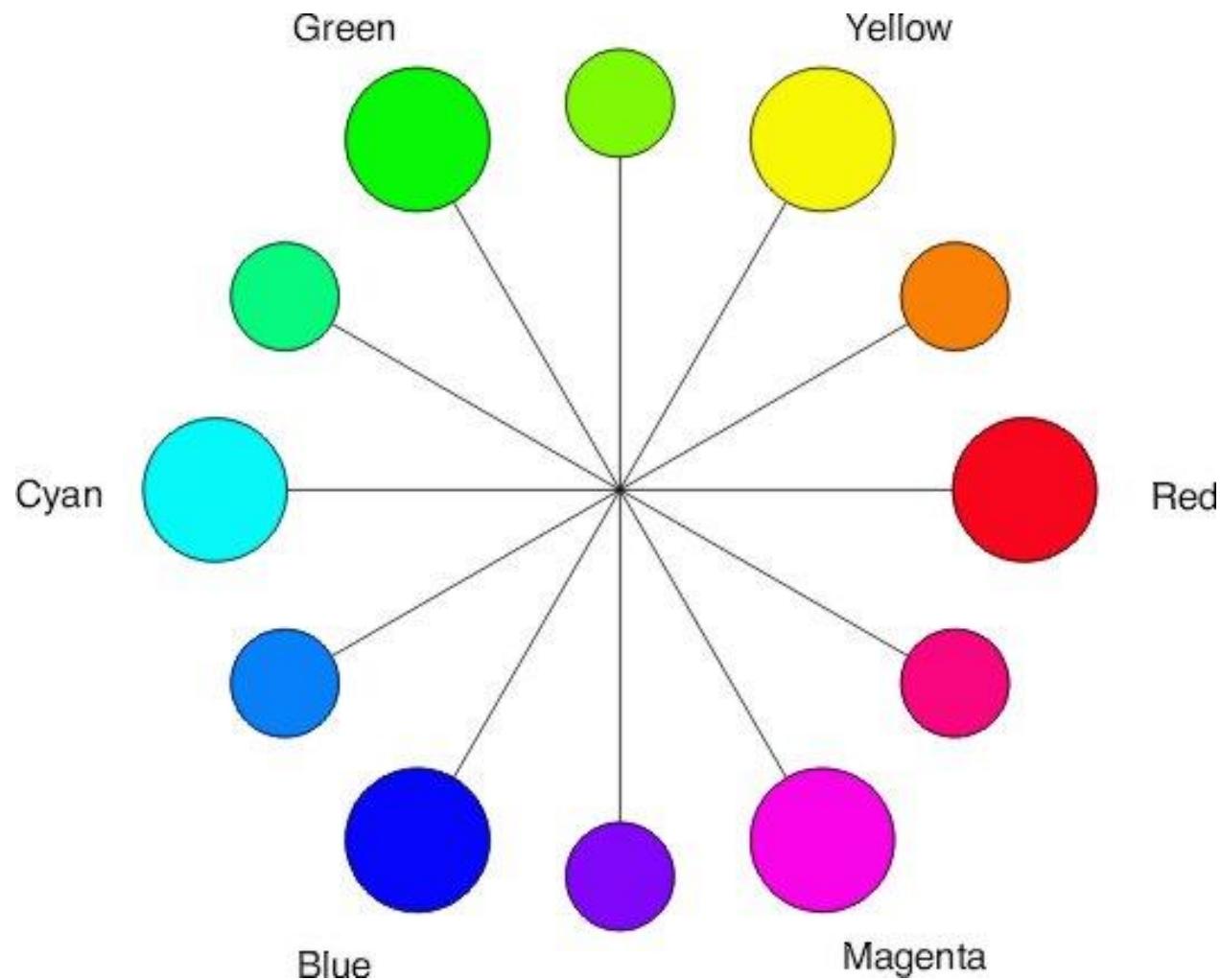
Red

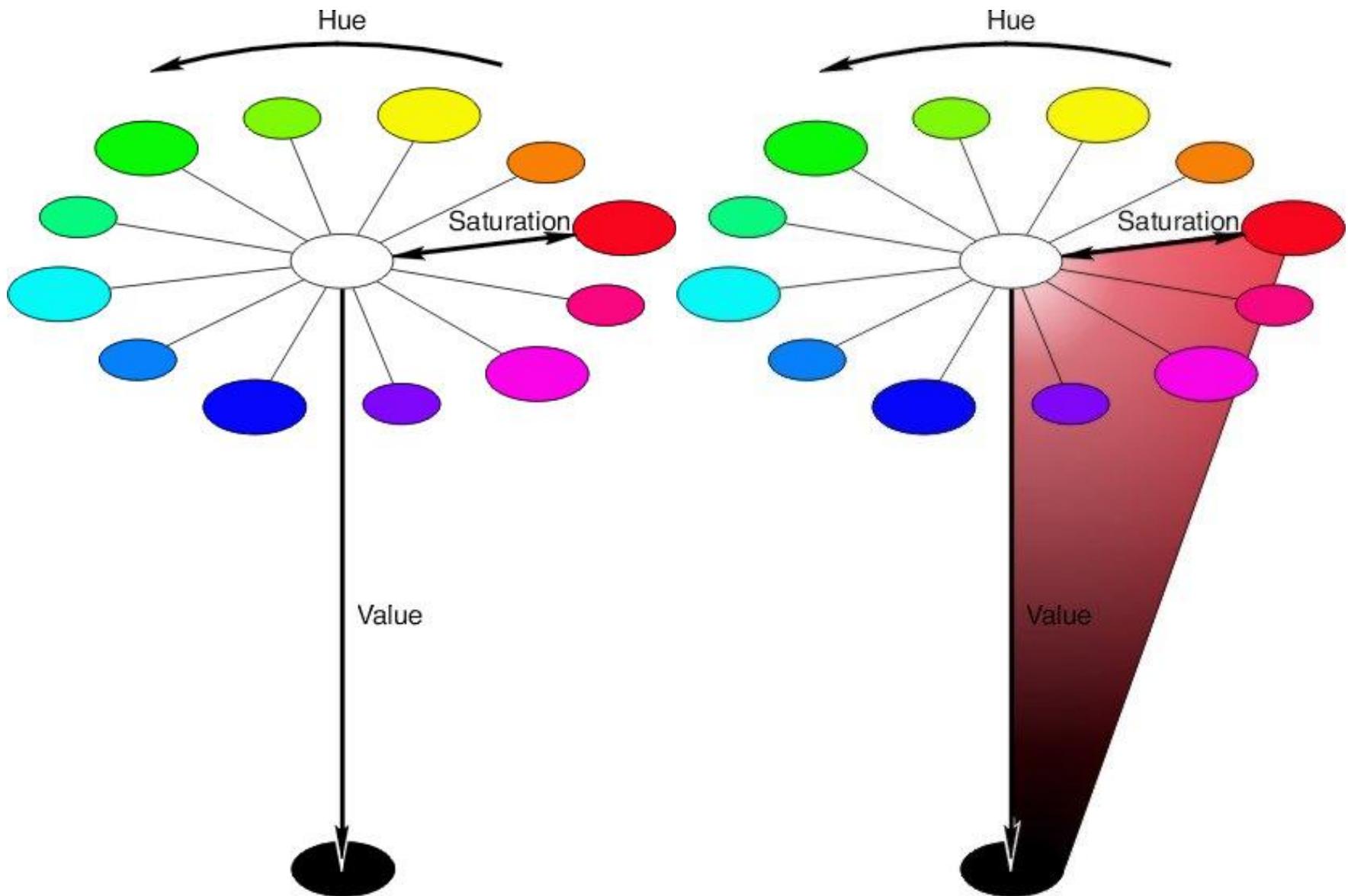


Blue









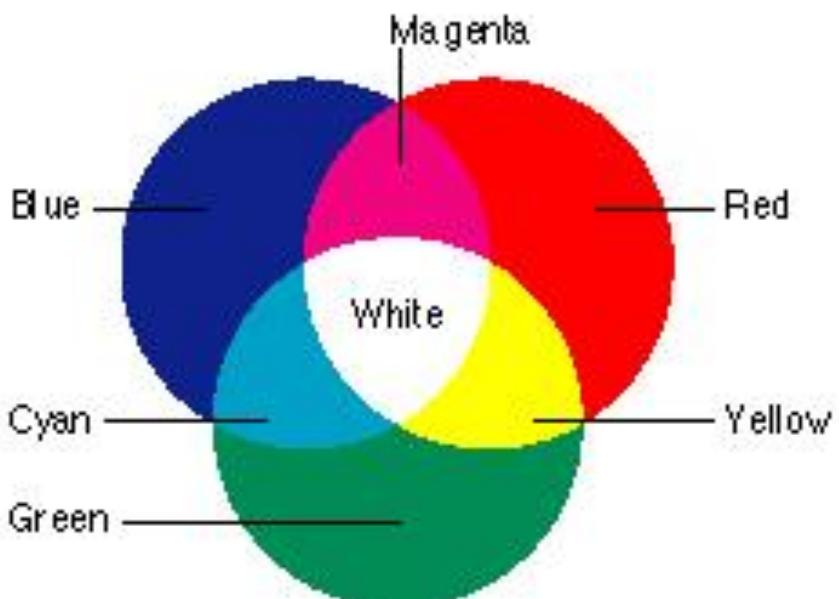
CMY

- Subtractive mixture
- Used in colour printers
- Three basic colours: Cyan, Magenta, Yellow
- Complementary to primary colours Red, Green, Blue

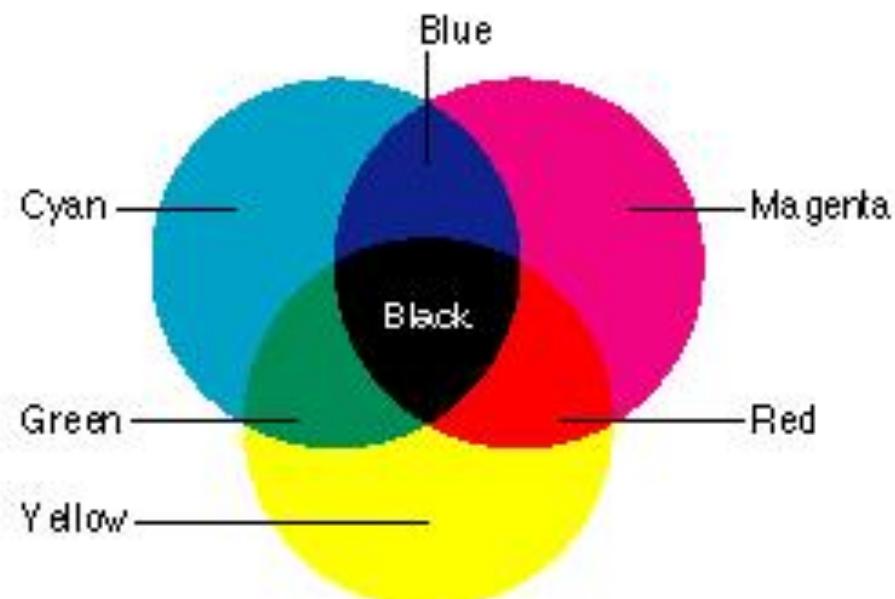
$$\begin{bmatrix} C \\ M \\ K \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

Comparison

Additive colours (in RGB)



Subtractive colours (in CMYK)

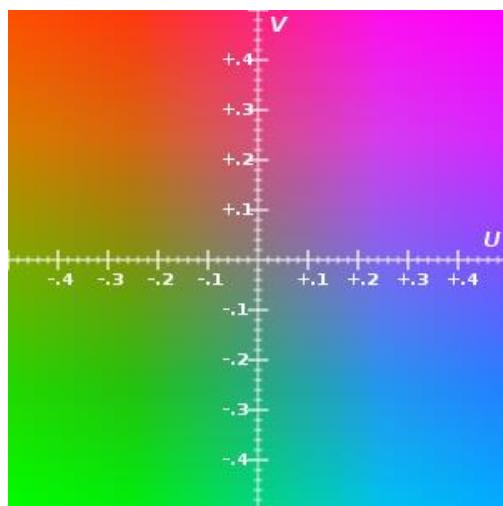


YIQ

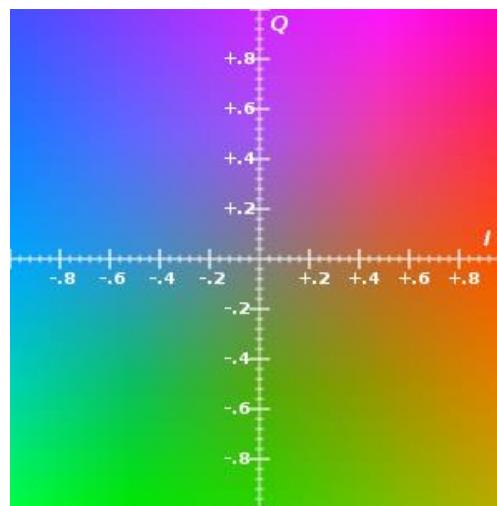
- Colour space used for TV transmission (**NTSC**)

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.275 & -0.321 \\ 0.212 & -0.523 & 0.311 \end{bmatrix} \begin{bmatrix} R_N \\ G_N \\ B_N \end{bmatrix}$$

YUV



YIQ



YUV

- Colour space used for TV transmission (**PAL, SECAM**)
- U and V obtained from I and Q with a rotation

$$\begin{bmatrix} I \\ Q \end{bmatrix} = \begin{bmatrix} \cos(33^\circ) & -\sin(33^\circ) \\ \sin(33^\circ) & \cos(33^\circ) \end{bmatrix} \begin{bmatrix} U \\ V \end{bmatrix}$$

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.148 & -0.289 & 0.437 \\ 0.615 & -0.515 & -0.100 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

Today's agenda

- Colour images
- PGM/PPM images

Colour vs. Grey



Colour vs. B&W



Today's agenda

- Colour spaces
- Colour images
- PGM/PPM images

PPM Header

- PPM header
 - consists of at least **three parts**
normally separated by carriage returns and/or linefeeds
but the PPM specification only requires white space
 - first "line" → a magic PPM **identifier** (P3 or P6)
 - next line → **width** and **height** of the image as ascii numbers
 - last part of the header → **maximum value of the colour**
components for the pixels, this allows the format to describe more
than single byte (0..255) colour values
- In addition to the above required lines, a comment can be placed anywhere with a "**#**" character, the comment extends to the end of the line

PPM Header

- The following are all valid PPM headers

- Example 1

P6 1024 788 255

- Example 2

P6

1024 788

My own comment on this image

255

- Example 3

P3

1024 # the image width

788 # the image height

Another comment

255

PPM Header

- Magic PPM identifier
 - Defines the format of the image data itself
 - "P3"
 - the image is given as **ascii** text
 - the numerical value of each pixel ranges from 0 to the maximum value given in the header
 - the lines should not be longer than 70 characters
 - "P6"
 - the image data is stored in **binary** format, one byte per colour component (r,g,b)
 - Comments can only occur before the last field of the header and only one byte may appear after the last header field, normally a carriage return or line feed
- Note1. "P6" image files are smaller than "P3" and much faster to read
- Note2. "P6" image files can only be used for single byte colours

PPM format

- Image
 - while not required by the format specification it is a standard convention to store the image in
 - top to bottom, left to right order
- Pixels
 - Each pixel is stored as a byte
 - value **0** == black
 - value **255** == white.
 - The components are stored in the "usual" order, **red** - **green** - **blue**

PGM format

- PGM format
 - stores grey-scale information
i.e., 1 value per pixel instead of 3 (R,G,B)
 - the header section: magic identifiers which are P2 and P5
 - **P2** corresponds to the ascii form of the data
 - **P5** corresponds to the binary form of the data
- Magic numbers - summary
 - **P2** PGM grey scale image, stored in ASCII, one value per pixel
 - **P3** PPM color image, stored in ASCII, 3 values rgb per pixel
 - **P5** PGM grey scale image stored in binary (compressed) format
 - **P6** PPM color image stored in binary (compressed) format

PGM example

P2

24 7

7

```
000000000000000000000000  
033330077770011110051550  
030000070000010000050010  
033300077700011100011110  
030000070000010000010000  
030000077770011110010000  
000000000000000000000000
```

What did we learn today?

- Colour spaces
- Colour images
- PGM/PPM images

Image and video processing

Image filtering

Dr. Miles Hansard

Today's agenda

- Noise
- PSNR
- Convolution
- Image smoothing and noise reduction

Today's agenda

- Noise
- PSNR
- Convolution
- Image smoothing and noise reduction

Image & video processing flow diagram

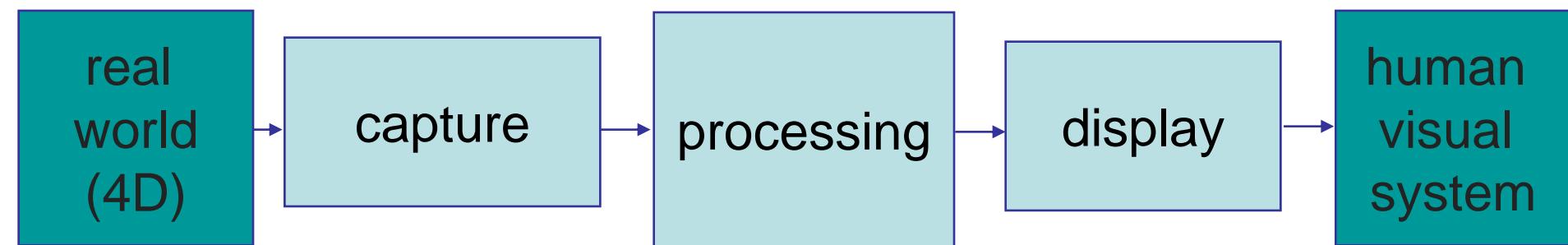
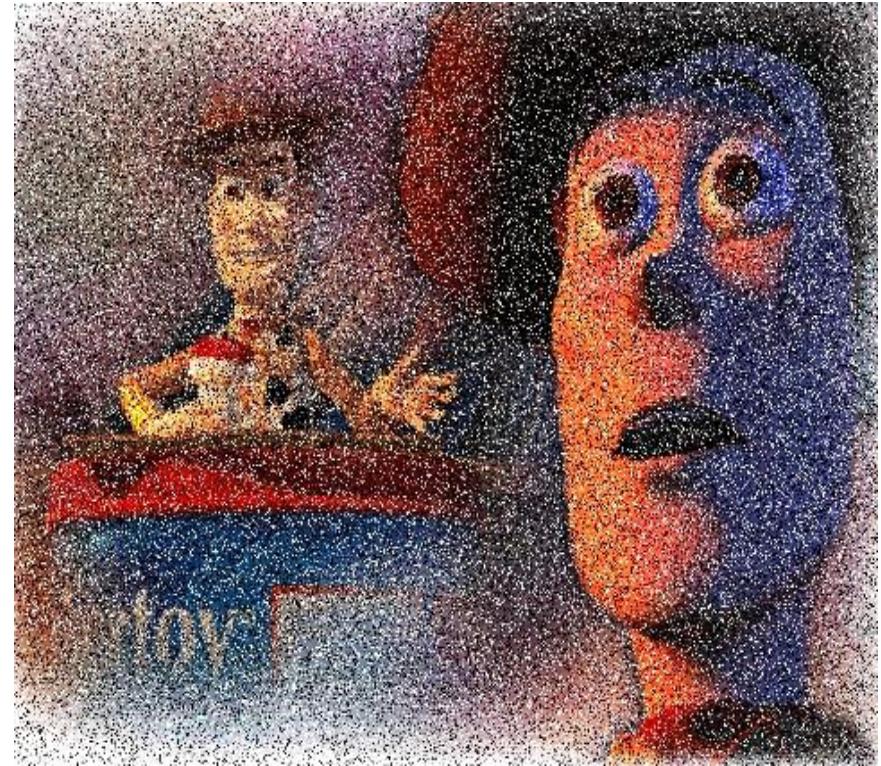


Image noise



original image



noisy image

Causes

- Incident Signal-to-Noise Ratio (SNR)
 - Power ratio
 - Source signal strength
 - Propagation losses
 - Background noise
 - Temperature noise at detector

Sensor induced noise

- Sensor induced noise
 - Interference among adjacent detectors
 - Thermal noise in infra-red detectors
 - Cooled to reduce
 - Leakage noise
 - Conductance at surface of detector
 - Shot noise
 - Electron devices

Also analog-to-digital conversion noise, and quantum noise.

Noise Removal – De-noising

- De-noising
 - Select or build a **low-pass filter**
 - Slide the filter over each successive sample point
 - Compute the **convolution integral** at that point
(i.e., the area under the product curve)
(the **weighted average** of the current pixel and its nearby neighbors)
 - Most filters are
 - **symmetric** around their origin and
 - **fall off** rapidly from their center

Note1: For 2D images/3D surfaces → the filter is a 2D image/3D surface

Note2: For digital images → we do not compute the integral,
but the sum of discrete values of the filter function

Convolution example (1D)

- Convolve these two sequences

Today's agenda

- Noise
- PSNR
- Convolution
- Image smoothing and noise reduction

Image quality measurement: SNR

- Signal-to-Noise Ratio (SNR)
 - estimates of the quality of a processed image $I'(x, y)$ compared with an original image $I(x, y)$
 - Basic idea → to compute a **single number** that reflects the quality of a distorted image
 - The higher the value of the metric → the better the quality of the distorted images

NB In general SNR measures do not reflect human subjective perception

PSNR

- Peak signal-to-noise (distortion) ratio PSNR
 - For a given source image $I(x, y)$ of dimension $N \times M$ and a distorted image $I'(x, y)$, the **error or distortion** is computed as follows:
 - Mean squared error (MSE) of the reconstructed image

why square? To ensure positivity.
Why normalize? To remain independent of size of the image.

$$MSE = \frac{\sum [I(x, y) - I'(x, y)]^2}{NM}$$

- PSNR in decibels (dB)

$$PSNR = 20 \log_{10} \left(255 / \sqrt{MSE} \right)$$

PSNR

- PSNR values 意义：比较算法的优越性
 - typically range between **20 and 40**
 - are usually estimated with two decimal points (e.g., 25.47)
 - the actual value is not meaningful, but the **comparison** between two values for different reconstructed images gives one measure of quality
- Importance of a PSNR difference 1.122
 - The MPEG committee used an informal threshold of **0.5 dB** PSNR to decide whether to incorporate a coding optimization because they believed that an improvement of that magnitude would be visible

Problem: 不care图片上noise的分布，比如一大块像素的损失和分布零散的小噪点可能有一样的PSNR。

Alternative: SSIM

Distortion errors

- Visualisation of errors
 - a technique to visualize errors is to construct an **error image** which shows the pixel-by-pixel errors
 - to create the pixel-by-pixel image **difference** between the distorted and original images
 - problem: zero difference is black → most errors are small numbers → shades of black → hard to see!
 - solution: multiply the difference image by a constant → increase the visible difference

$$E(x, y) = 255 \left[|I(x, y) - I'(x, y)| - \min \right] / (\max - \min)$$

Today's agenda

- Noise
- PSNR
- Convolution
- Image smoothing and noise reduction

Convolution: applications

- Deconvolution
 - Remove effects of previously applied linear operations
- Noise removal
 - Filtering to separate noise from signal for estimating noise
 - Feature detection
 - Periodic noise removal
- Feature enhancement
 - Using high pass filter, for instance

Blurring – low pass

1D continuous convolution

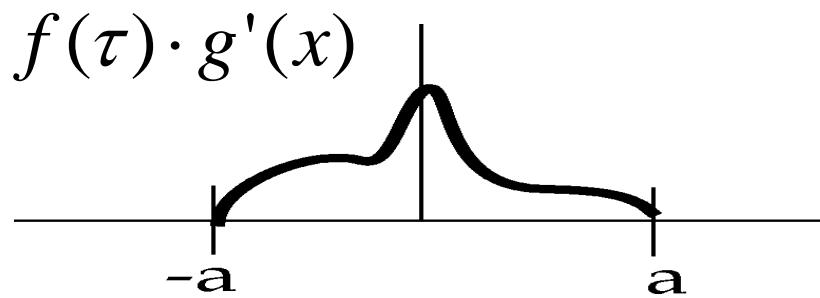
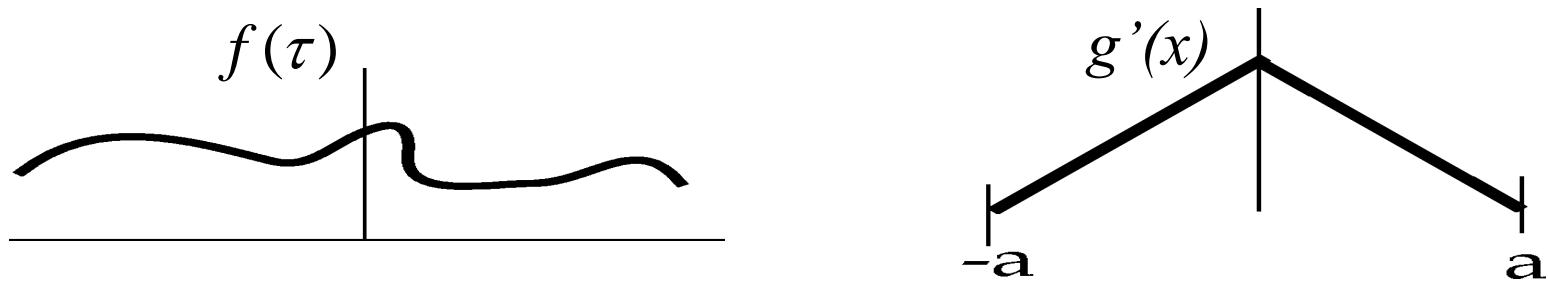
- If a signal $r(t)$ is input into a **linear system**, the output is the convolution of the input signal with the transfer function $h(t)$

$$c(t) = \int_{-\infty}^{+\infty} h(t - \tau) r(\tau) d\tau$$

- A useful ‘function’ is the **Dirac delta**, which is like a ‘spike’, defined as:
- $\delta(x) = \infty$ if $x = 0$
- $\delta(x) = 0$ if $x \neq 0$
- $\int_{-\infty}^{\infty} \delta(x) dx = 1$
- Convolution of a function with $\delta(x)$ returns the original function. Convolution with $\delta(x - t)$ performs a *shift*.

Example: 1D continuous

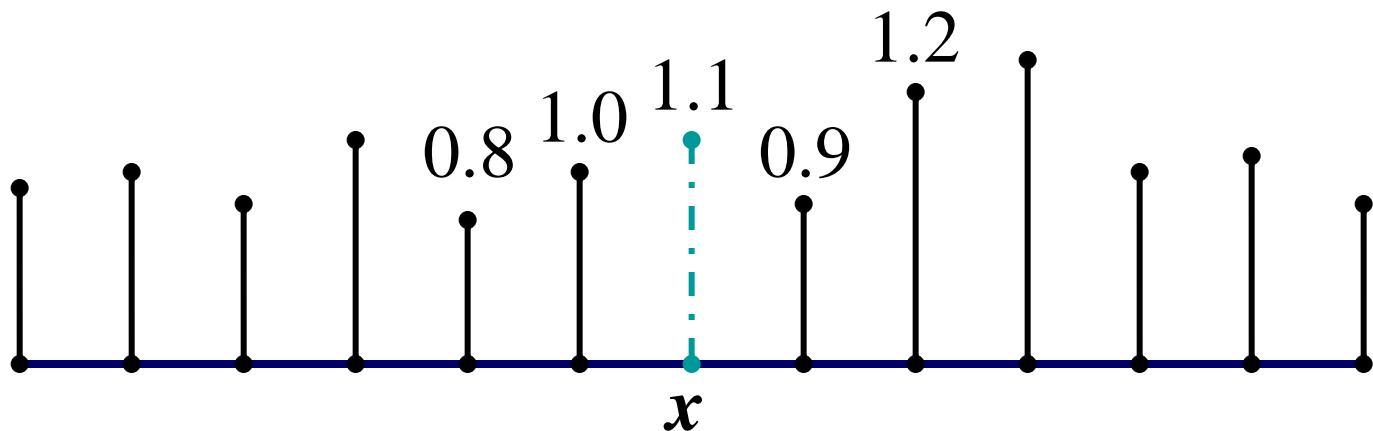
- Filtering by convolution
 - computes the area under the product curve $h(x) = f(\tau) \cdot g'(x)$



Discrete 1-D convolution

- Input
 - The discrete signal r with m sampling points
- Convolution kernel
 - Filter h of length n
- Output
 - c is a sequence of length m
 - the i^{th} element is
$$c(i) = r(i) * h(i) \\ = \sum_j r(j)h(i-j)$$

Example: 1D discrete



Filter

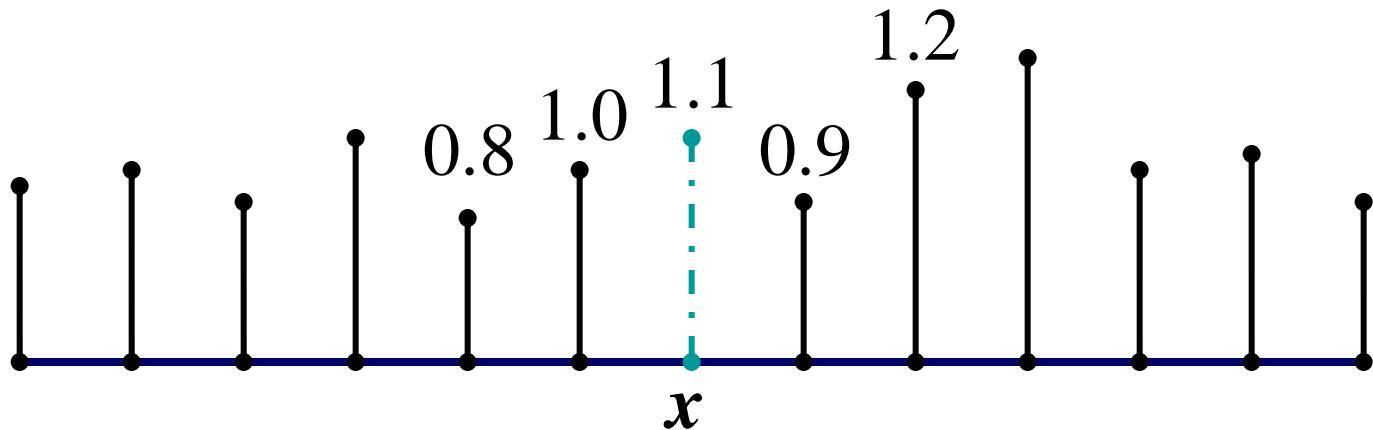
$$c = (1/9, 2/9, 1/3, 2/9, 1/9)$$

$$c(0) = 1/3, c(1) = c(-1) = 2/9, c(2) = c(-2) = 1/9$$

Amplitude or intensity value at $x=1.1$

Convolution (filter response at x)

Example: 1D discrete



Filter

$$c = (1/9, 2/9, 1/3, 2/9, 1/9)$$

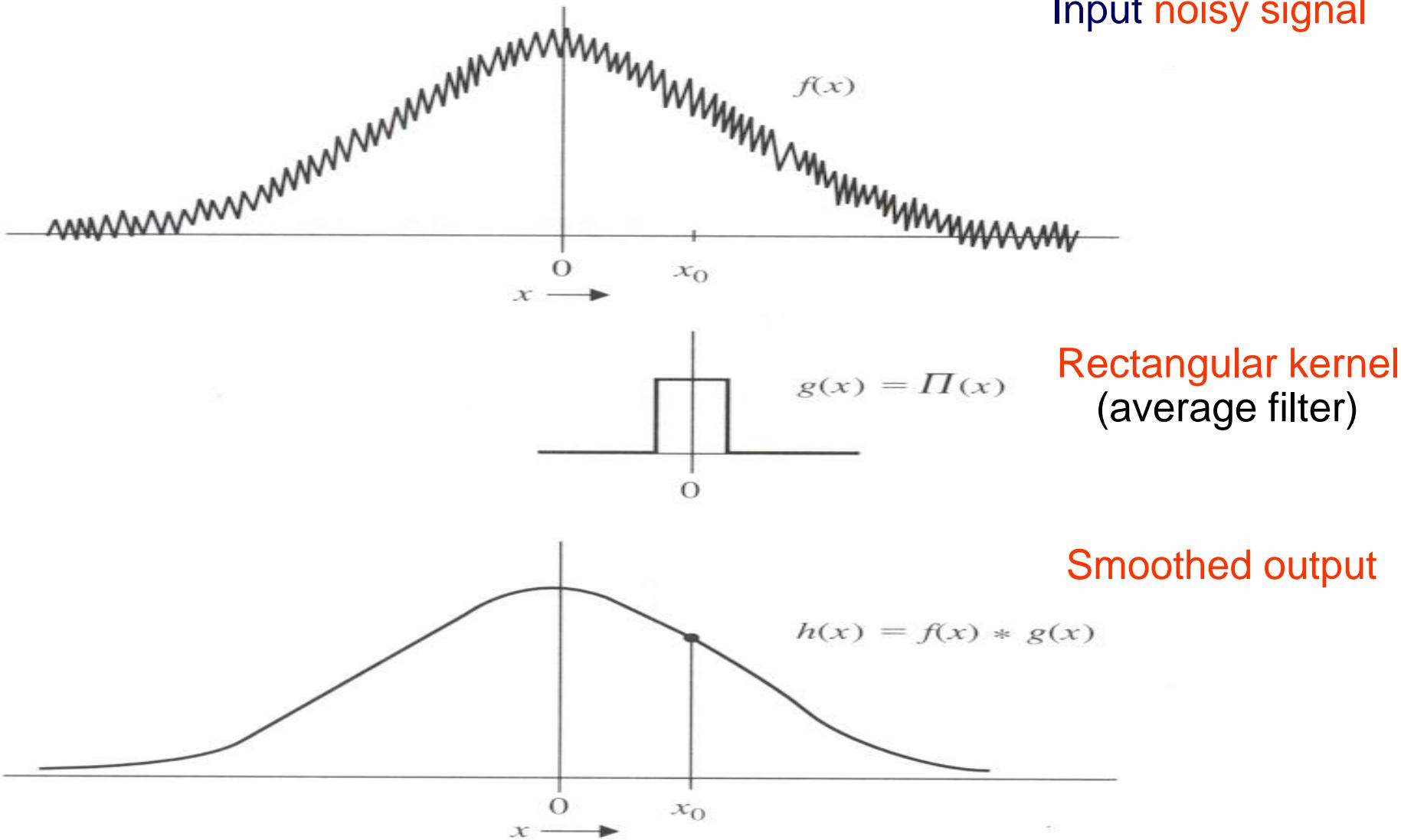
$$c(0) = 1/3, c(1) = c(-1) = 2/9, c(2) = c(-2) = 1/9$$

Amplitude or intensity value at x=1.1

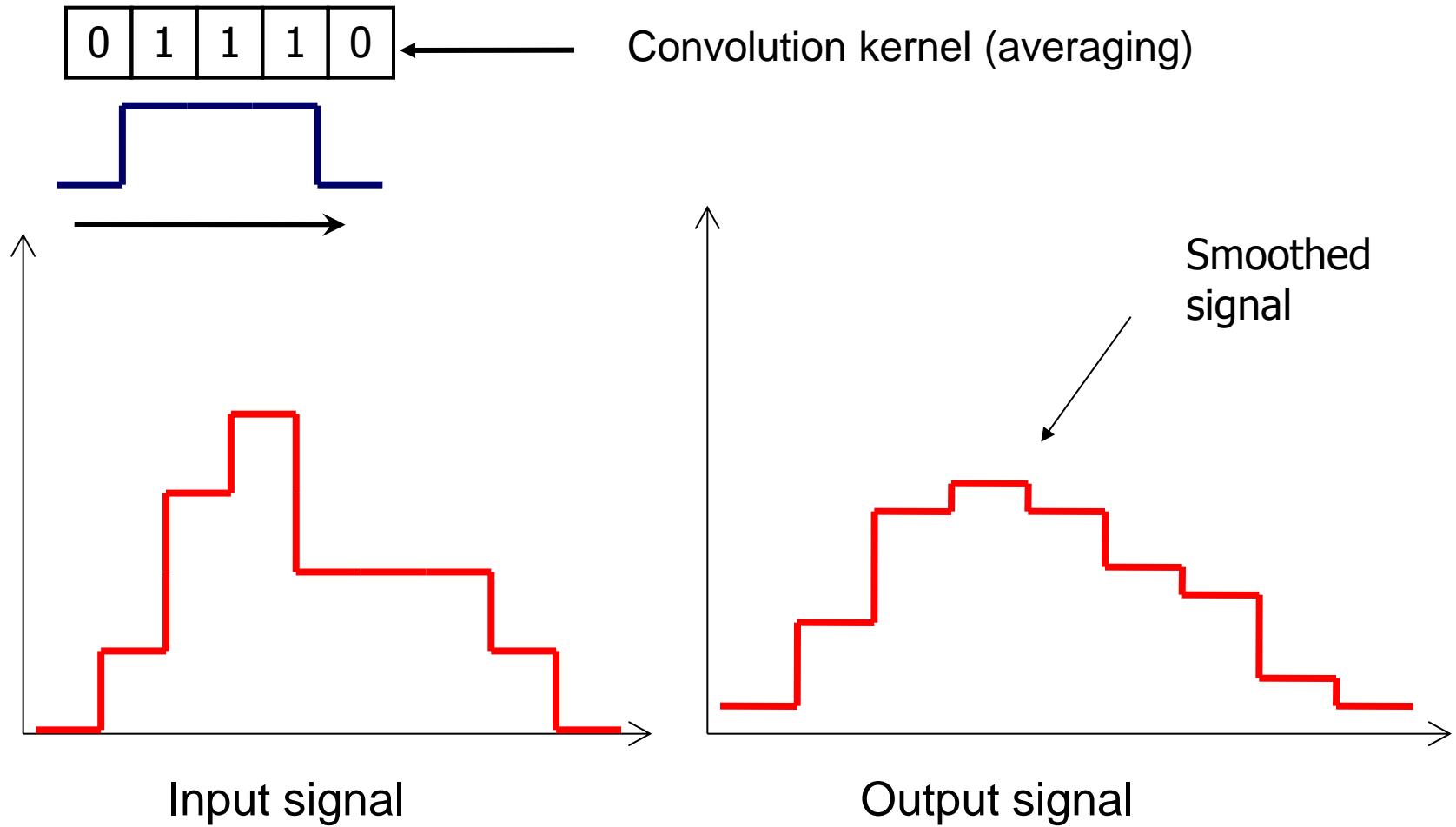
Convolution (filter response at x)

$$\sum_{i=x-2}^{x+2} I(i).c(i - x) = (0.8)x(1/9) + (1.0)x(2/9) + (1.1)x(1/3) + (0.9)x(2/9) + (1.2)x(1/9) = 1.011$$

Example: 1D



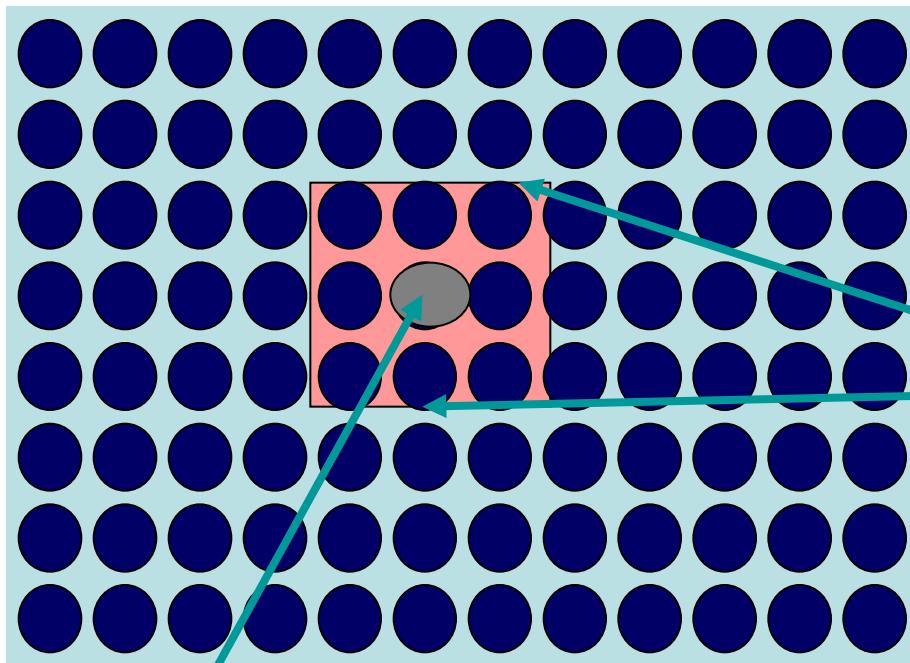
Example: 1D discrete



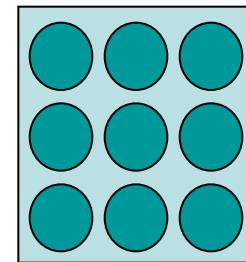
Convolution operator: properties

- Linearity: $f * (\alpha h + \beta g) = \alpha f * h + \beta f * g$
- Associativity: $(f * g) * h = f * (g * h)$
- Derivative: $d/dt (f * g) = f' * g = f * g'$

Discrete 2D convolution



Pixel p to be processed



3X3
convolution
kernel

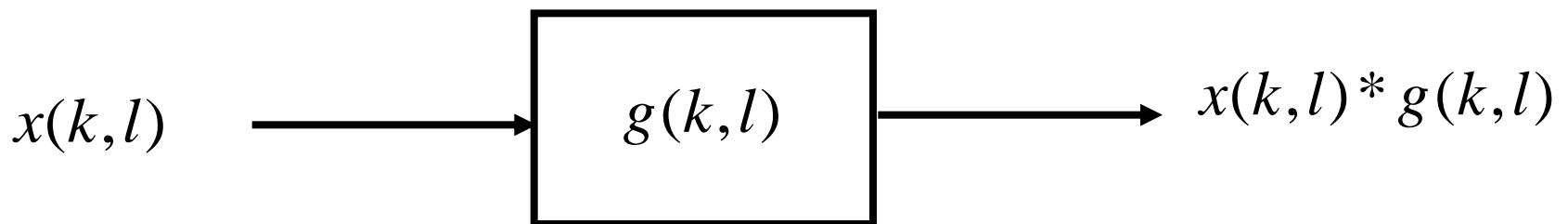
Pixels involved in
the convolution
process for the
single pixel p

2D convolution

- 2D convolution

$$x(k, l) * g(k, l) = \sum_{k'=-\infty}^{+\infty} \sum_{l'=-\infty}^{+\infty} x(k', l') g(k - k', l - l')$$

- Notation



2D convolution: properties

- Commutativity

$$\begin{aligned}x(k, l) * g(k, l) &= \sum_{k'=-\infty}^{+\infty} \sum_{l'=-\infty}^{+\infty} x(k', l') g(k - k', l - l') \\&= \sum_{k'=-\infty}^{+\infty} \sum_{l'=-\infty}^{+\infty} g(k', l') x(k - k', l - l') = g(k, l) * x(k, l)\end{aligned}$$

2D convolution: properties

- Associativity

$$[x(k,l) * g(k,l)] * h(k,l) = x(k,l) * [g(k,l) * h(k,l)]$$

- Linearity

$$x(k,l) * [g(k,l) + h(k,l)] = x(k,l) * g(k,l) + x(k,l) * h(k,l)$$

Choices for convolution

- Determine the most appropriate **shape** of the window
- Determine the **size** of the window
- Solve the **border problem**

Size of the window

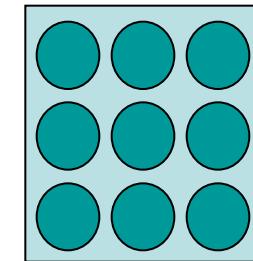
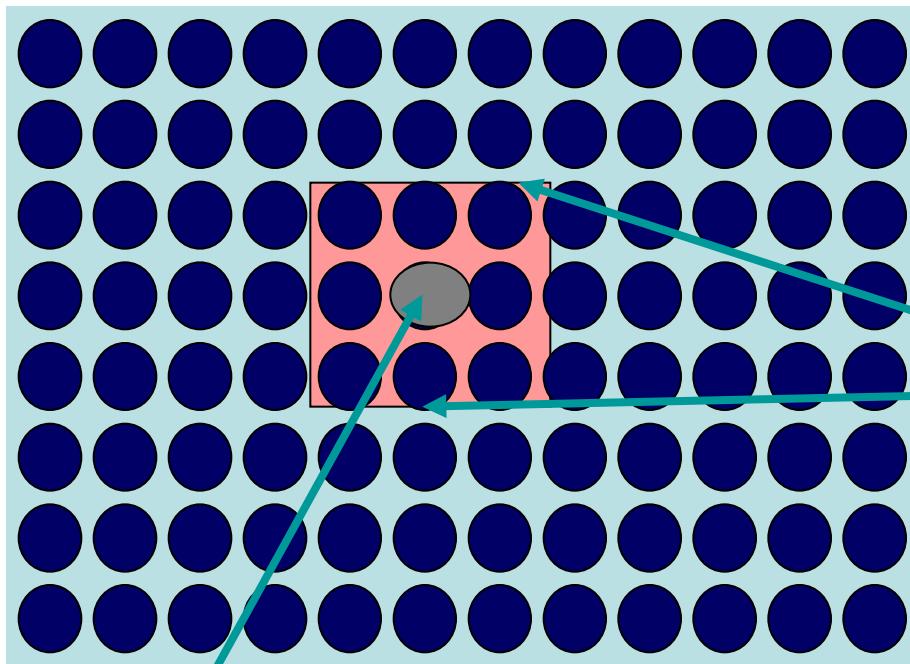
- The **limits of the sums** are defined by the size of the window (i.e., size of the filter) $\rightarrow M_g \times N_g$

$$x(k, l) * g(k, l) = \sum_{k'=0}^{M_g-1} \sum_{l'=0}^{N_g-1} g(k', l') x(k - k', l - l')$$

Convolution procedure

- Positioning
 - Place the **centre** of the window over the pixel position to be filtered
 - $M_g \times N_g$ pixels will be covered by the filter's window (mask, kernel)
- Multiply
 - the original pixel values of the image by the filter values corresponding to their location
- Add
 - Together the obtained multiplication results
 - The result of this addition is the **convolution result**

Discrete 2D convolution



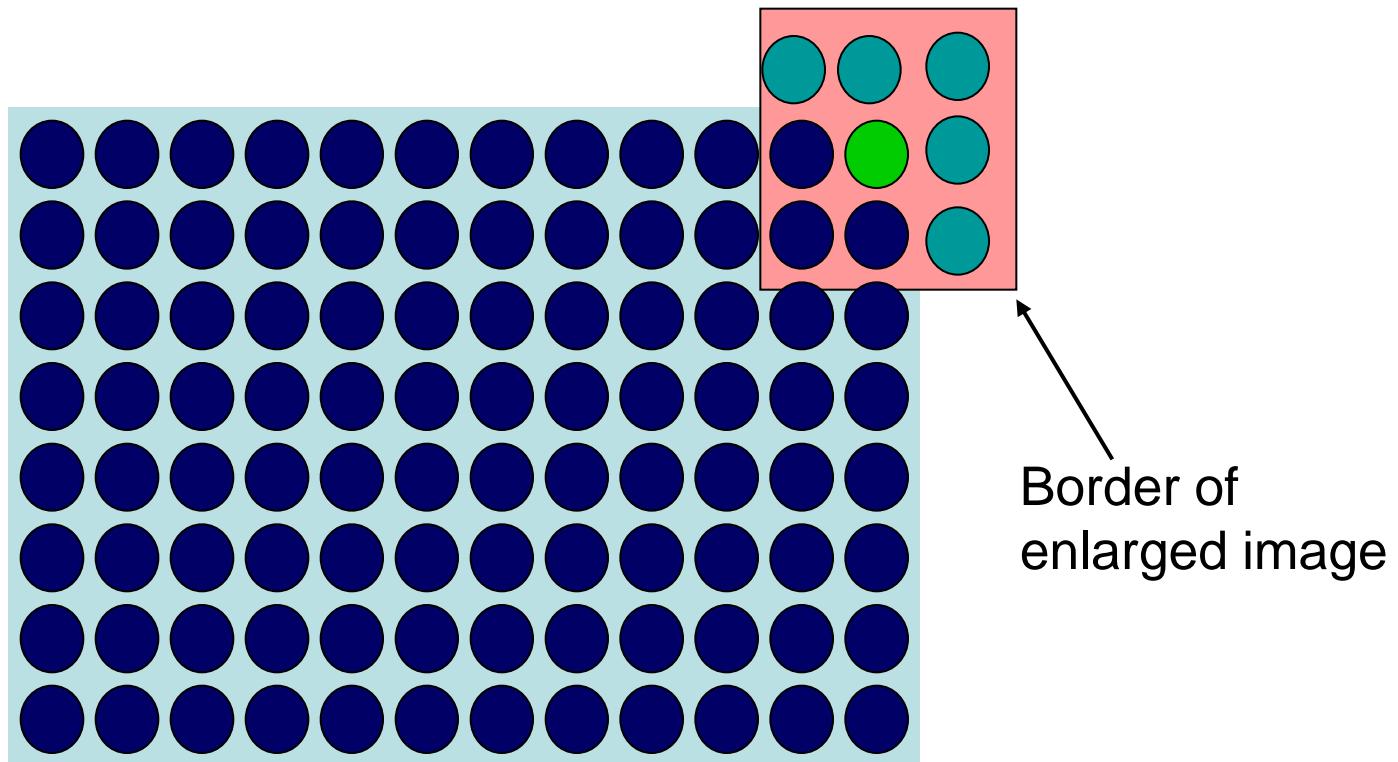
3X3 convolution kernel

Pixels involved in the convolution process for the single pixel p

Pixel p to be processed

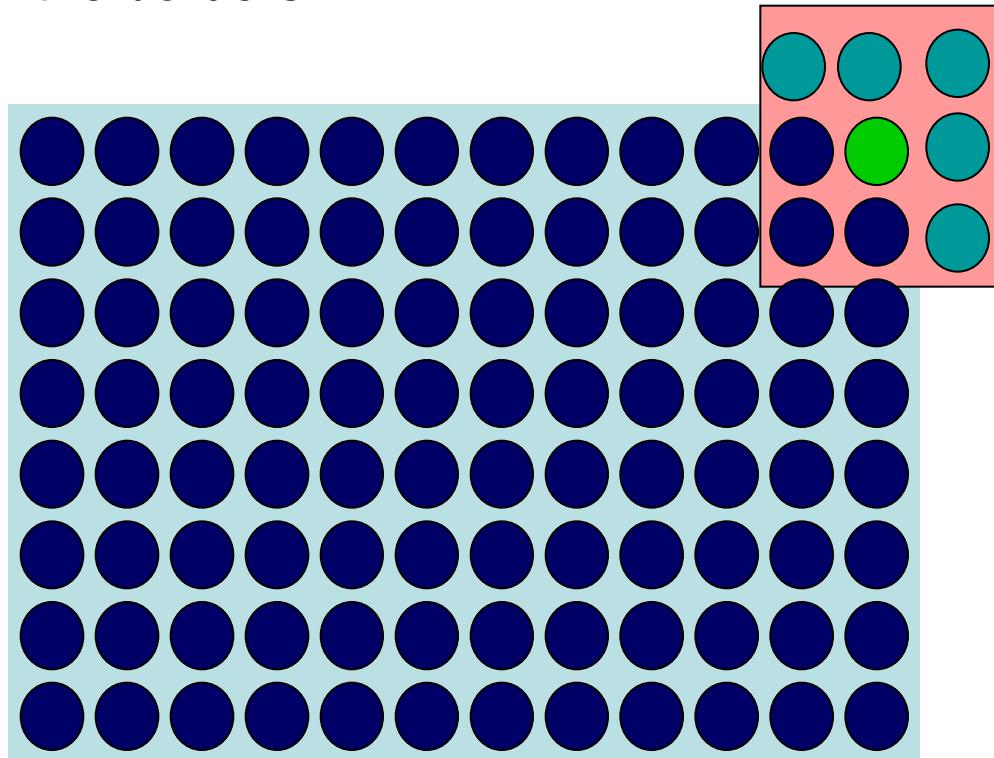
Border problem

- The values of the pixels **outside** the image **but** involved in the convolution process need to be estimated



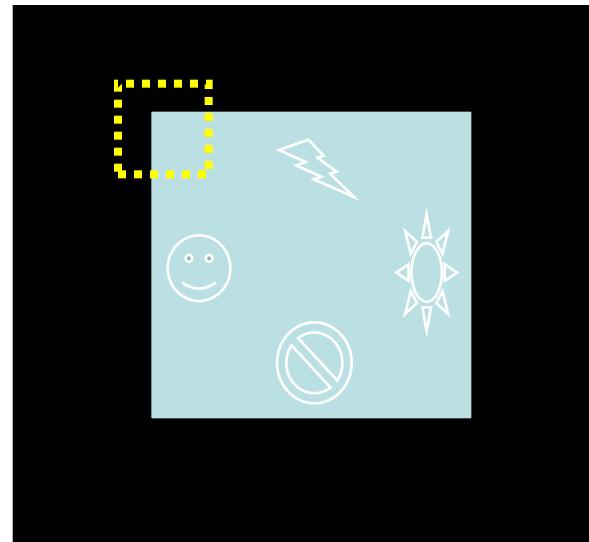
Border problem: solutions

- Change **filter** size along the border
- Enlarge **image**
 - Fill with zeros
 - Periodic extension of the image
 - Mirror the borders



Filling with zeros

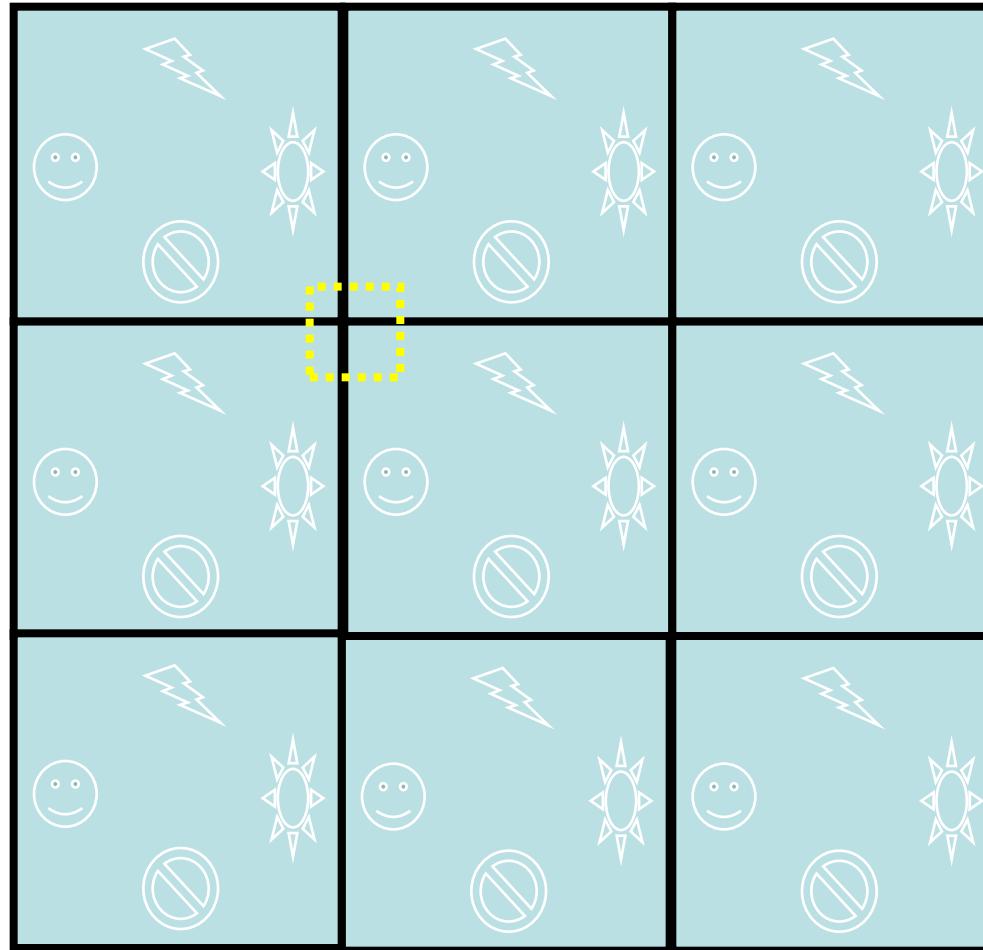
- Simple
- Generate border effects



Periodic extension

- Facilitate the software implementation of the filtering
- Coherent with the hypothesis of the Fourier Transform
- Better results than those obtained with filling with zeros

Periodic extension

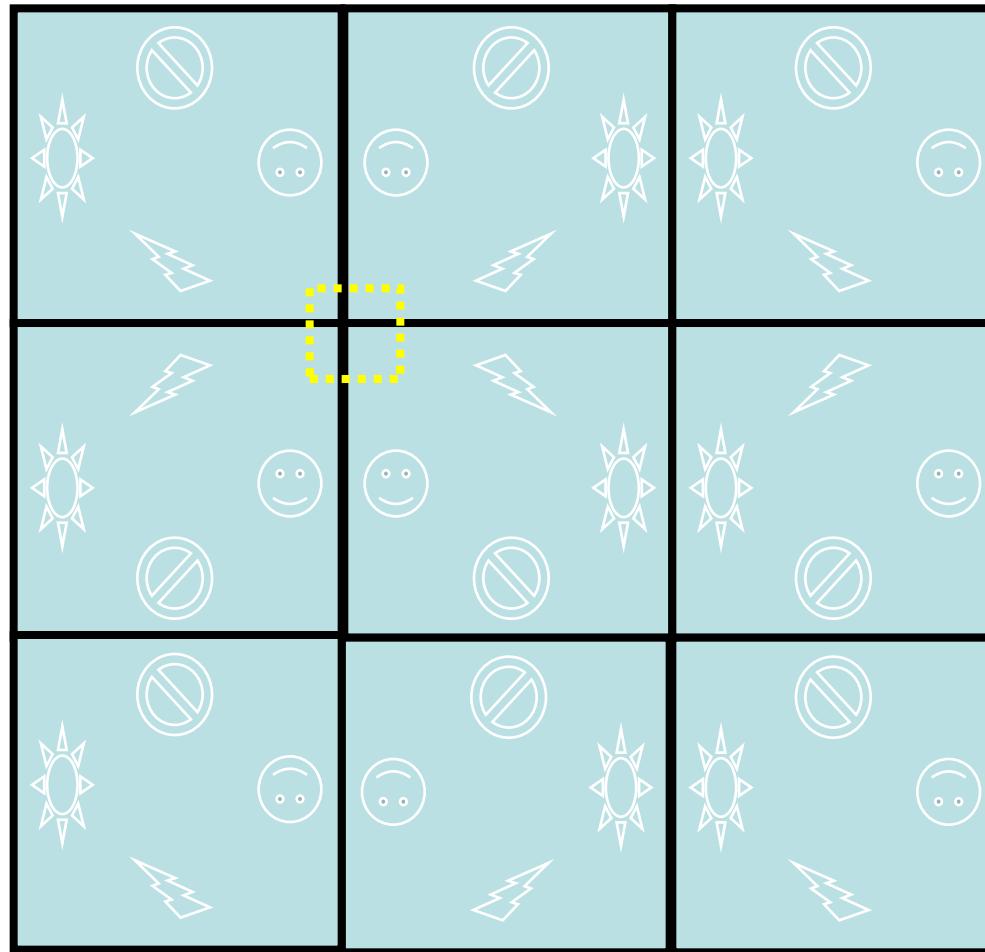


Mirroring

- More complex than the previous 2 solutions
- Better to eliminate the border effects

Mirroring

DCT



2D spatial filtering: summary

- Output of the filter
 - value calculated from the convolution with a local neighborhood in the input image
- Local neighborhood
 - enclosed in a window (mask, kernel) of $M_g \times N_g$ pixels
- Filtering
 - performed by shifting the window over the whole image

Today's agenda

- Noise
- PSNR
- Convolution
- Image smoothing and noise reduction

Spatial domain filtering

- Linear filters
 - Smoothing filters
 - mean filters
 - Gaussian filters
 - Edge enhancing filters
 - Sobel operator
 - Prewitt operator
 - Laplace operator
 -
- Non-linear filters
 - Median, Min, Max

Non-linear

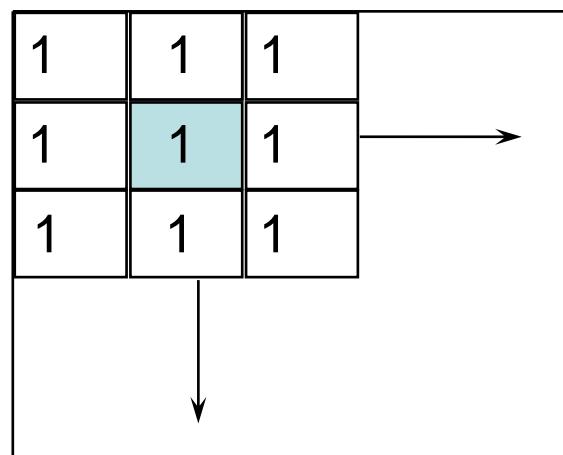
1. Fourier transforms are of no use
2. Sorting is slow

Low-pass filtering

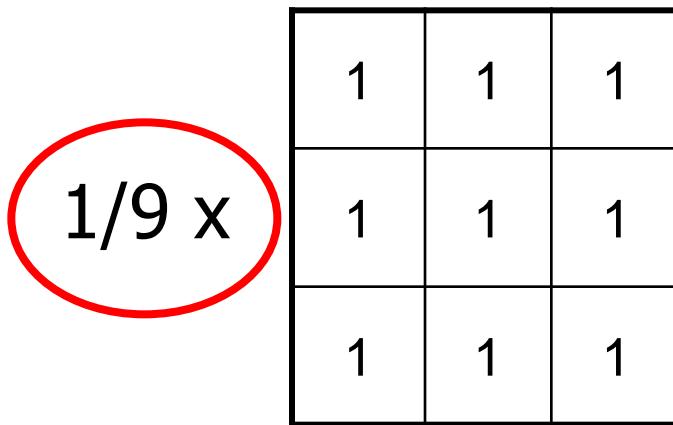
- Low-pass filtering
 - filter-out high frequencies
 - **simplest linear filter**: average= $1/n$
 - example
 - Convolving the discrete signal in the previous example with the average filter. What is the response at x ?
 - most popular filter: **Gaussian**

Smoothing filters

- Linear filters
- Adaptive filters (steerable)
- Non-linear filters
- Averaging → the simplest linear filter



Mean filter



- Need for normalization
 - To conserve the total “energy” of the image (sum of all grey levels)
- Quick
- Severe edge blurring

Mean filter: results



original image



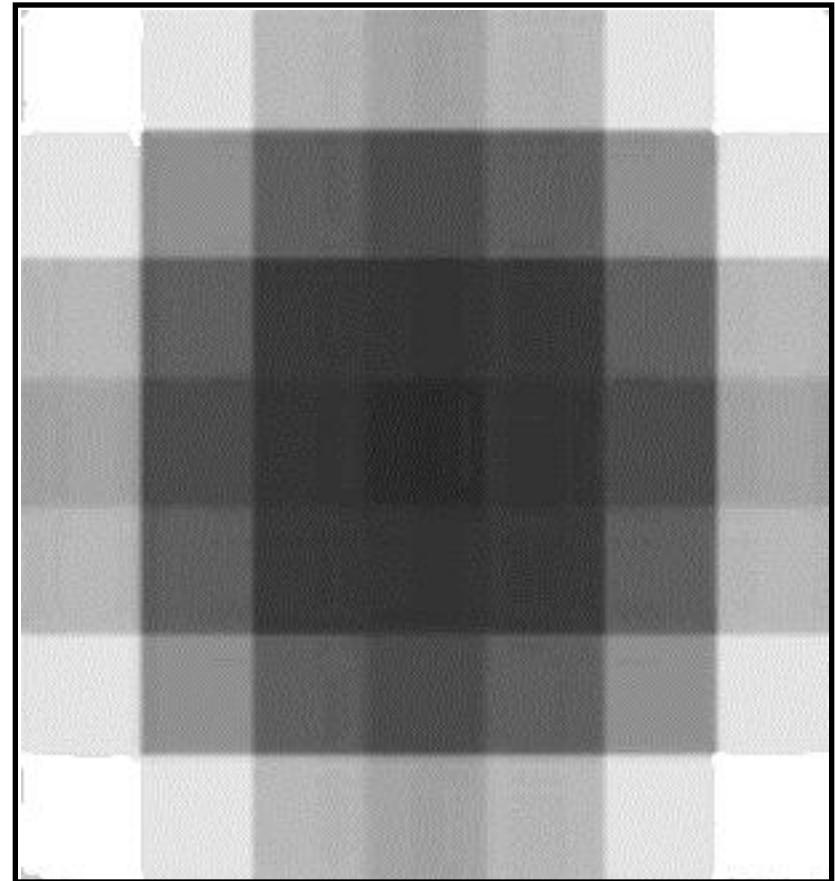
5x5 filter



9x9 filter

Gaussian filters

- 2D Gaussian kernel
 - (in the figure): the darker a pixel, the higher the filter value
 - the **weighting** values decrease proportionally to the distance from the center (**exponentially**)



Gaussian filter

- Gaussian filter
 - separable

$$G_\sigma(k, l) = \frac{1}{2\sigma^2\pi} e^{-\frac{k^2+l^2}{2\sigma^2}}$$

- The Fourier transform of a Gaussian is another Gaussian, of reciprocal width.
- The convolution of two Gaussians is another Gaussian.

$$G_\sigma(k, l) = G_\sigma(k)G_\sigma(l)$$

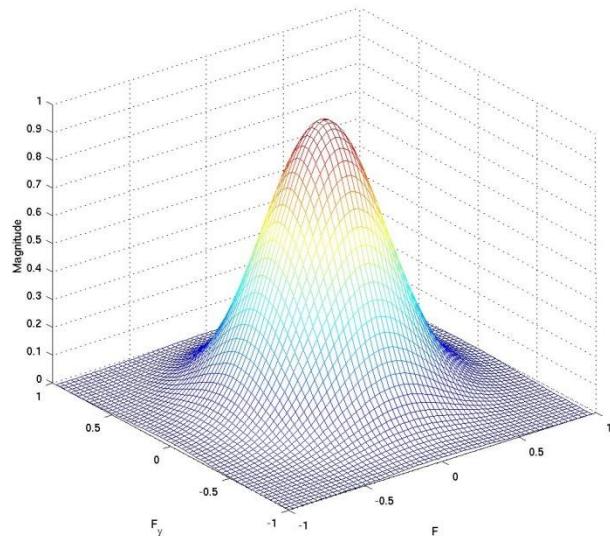
$$G_\sigma(k) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{k^2}{2\sigma^2}}$$

- We can perform the 2D convolution by:
 - First filtering the rows of the input image.
 - Then filtering the columns of the result.
 - This is much faster than using the 2D kernel.

用二维高斯的效果和先用一次横向一维+再用一次纵向一维

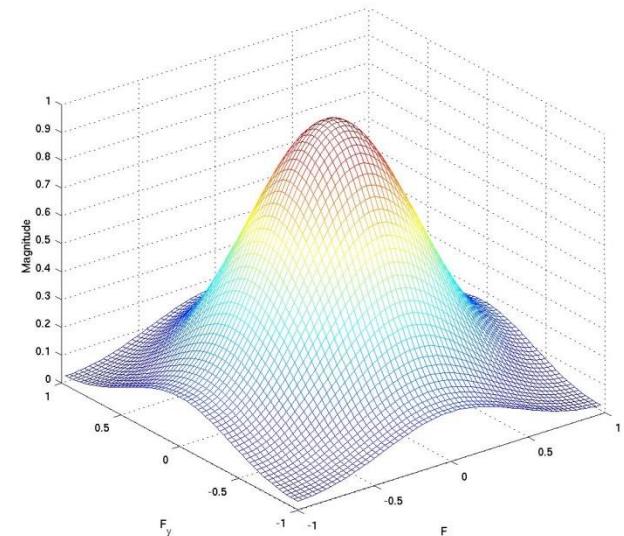
实际上，二维的复杂度是 $O(n^2)$ ，用两次一维的复杂度是 $O(2n)$

Gaussian filter 7x7

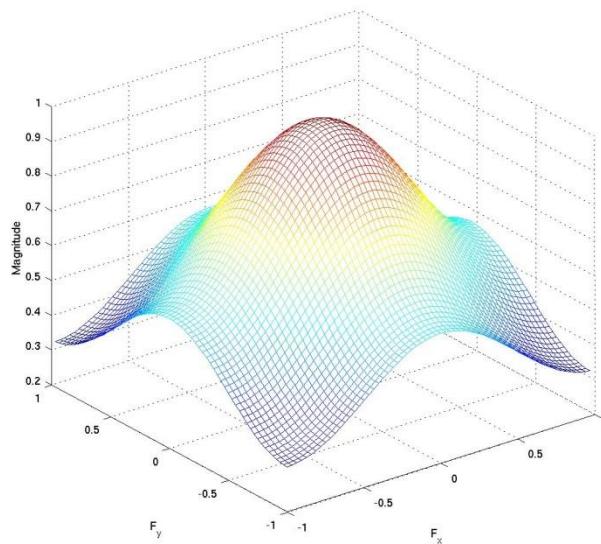


$\sigma=1$

$\sigma=0.5$



$\sigma=0.7$



Gaussian smoothing: results



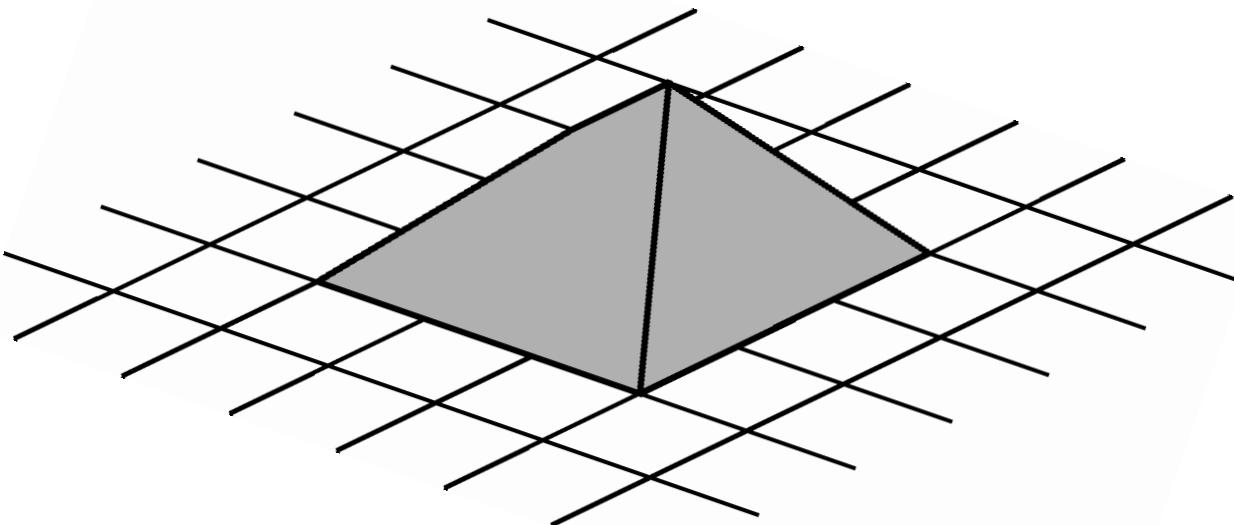
original image



13x13 filter kernel

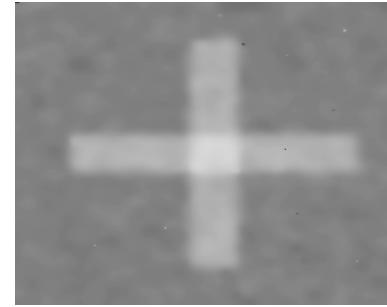
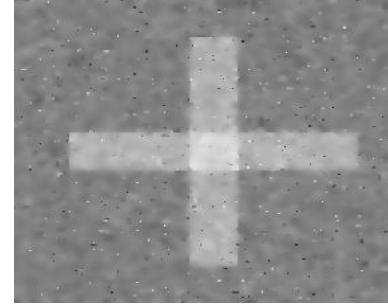
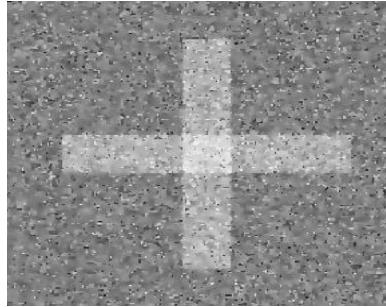
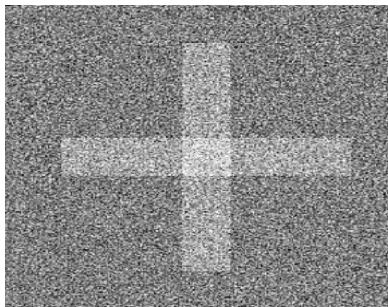
Linear pyramidal filter

- Simpler than the Gaussian filter
 - The kernel elements decrease **linearly** (not exponentially as in the Gaussian)

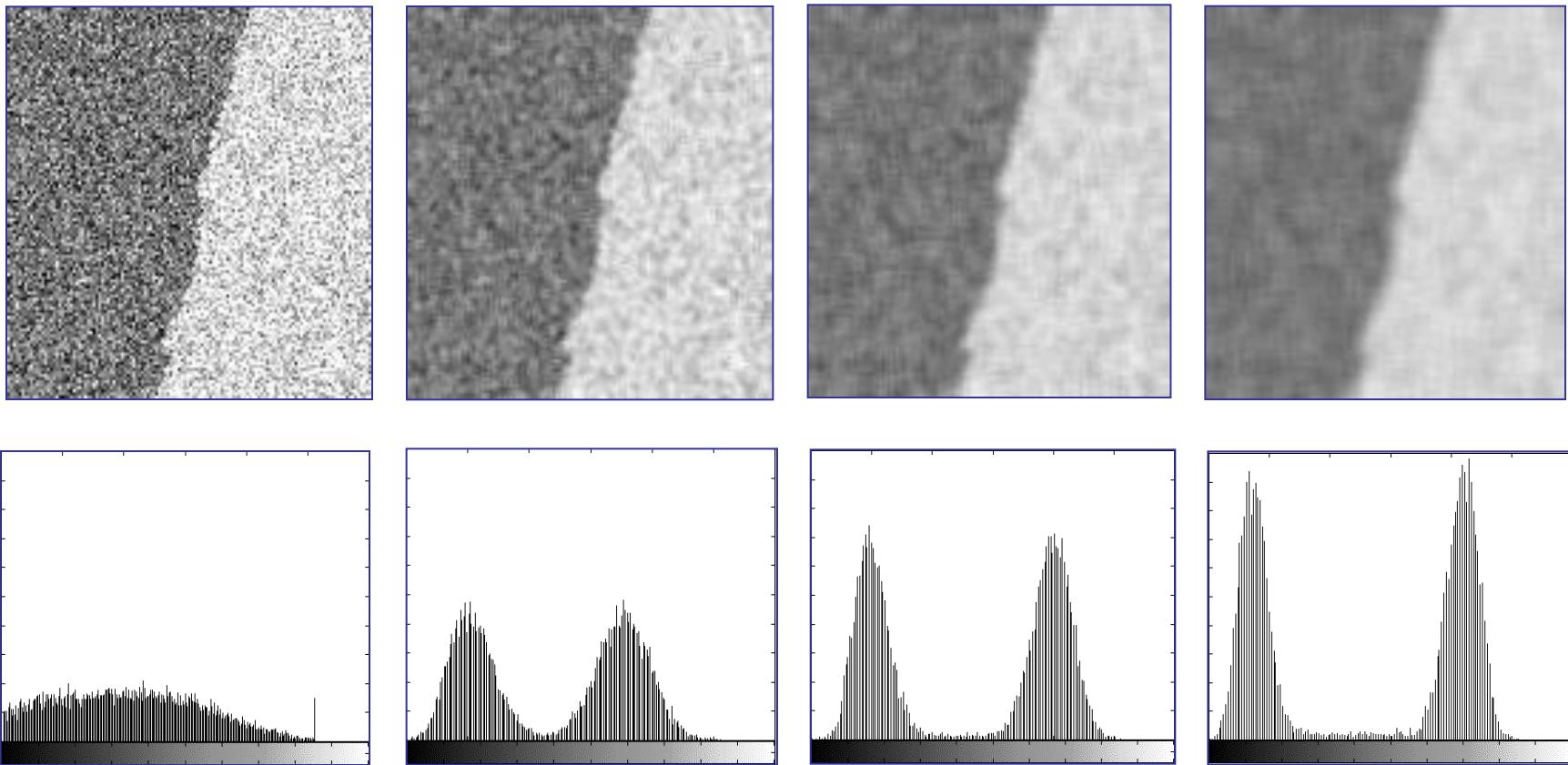


Non-linear de-noising and smoothing

- Advanced filters
 - remove high frequency components (noise) while keeping edges sharp



Noise in histogram thresholding



What did we learn today?

- Noise
- PSNR
- Convolution
- Image smoothing and noise reduction