



# Image Analysis

Rasmus R. Paulsen

Tim B. Dyrby

DTU Compute

<http://compute.dtu.dk/courses/02502>

# Week 1 - today

8:00 – 10:00

Exercises

---

10:00 – 12:00

Introduction and practical matters

---

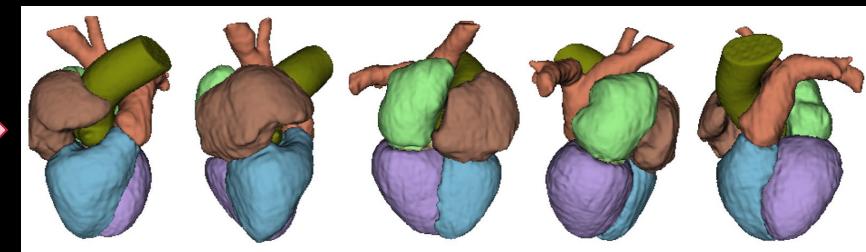
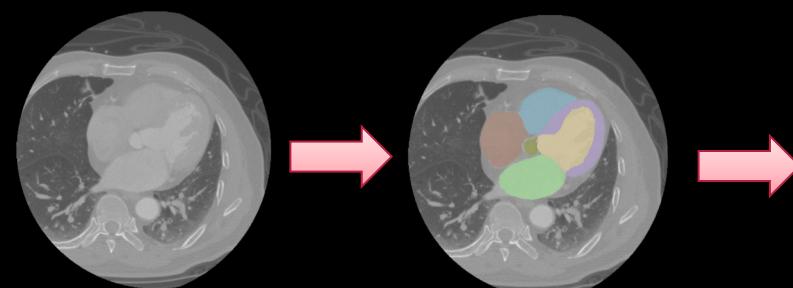
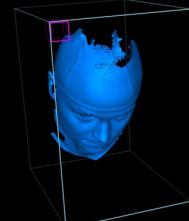
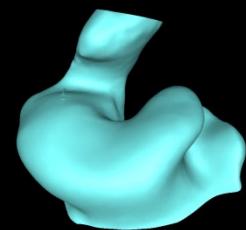
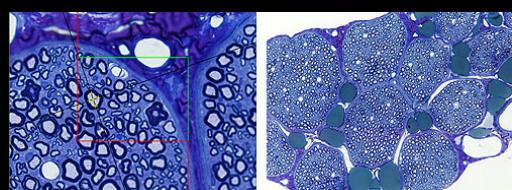
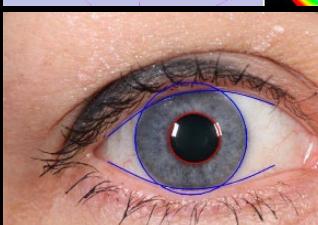
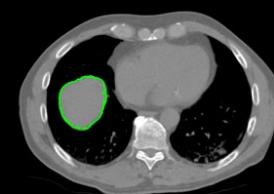
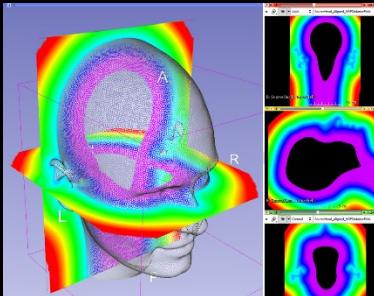
Lecture – An introduction to image analysis

---

Lecture – A tutorial on Principal Component Analysis (PCA)

---

# Rasmus R. Paulsen



- Master of Science (Eng). DTU 1998
- Industrial PhD with Oticon A/S
- Research and development at Oticon A/S
- Professor DTU Compute

# Tim B. Dyrby



■ Associate Professor at DTU Compute and Danish Research Centre for Magnetic Resonance (DRCMR)

# Teaching Assistants

- Mathias Micheelsen Lowes
- Bjørn Marius Schreblowski
- Thina Lundsgaard Thøgersen
- Andreas With Aspe

# Practical matters

- 13 days over the DTU 13 week semester
- Flipped class room
  - 8-10 Computer exercises (also on MS Teams)
  - 10-12 Lecture with quizzes
- Lectures are streamed, recorded and made available
  - Links to the stream will be posted on DTU Learn before the lecture
  - Links to video on the homepage (under schedule)
  - Courses.compute.dtu.dk/02502

## From 2021 on onwards

- The exercises are now much more related to the exam
- Learning objectives stated in all exercises
- You will be examined in these learning objectives
- You will also be examined in the more theoretical learning objectives from the lectures
- We expect that you can run Python during the exam!

**Very Important I: Do the exercises!**

**Very Important II:**  
**We can not help you after the course period!**

# Materials

## ■ Book:

- Rasmus R. Paulsen and Thomas B. Moeslund: *Introduction to Medical Image Analysis* (**MIA**)
- Polyteknisk boghandel
- <http://mediabook.compute.dtu.dk>

## ■ Notes

- Notes will be provided during the course

## ■ At the end of the course a complete reading list will be published

# DTU Learn and the homepage

- Homepage : The main entry to the course
  - <http://courses.compute.dtu.dk/02502>
  - Schedule / Exercises / Data
  - Updates happen!
- Course messages will be given through DTU Learn



#	Date	Topic	Video	Material	Exercise
1	4/2	Introduction to image analysis (Rasmus) Introduction to Principal Component Analysis (PCA) (Rasmus)	Stream	MIA 1, 2, app. A. PCA Note (except Section VI (SVD) and App. A)	1
2	11/2	Cameras, lenses, image compression, image storage and change detection in videos (Rasmus)	Stream	MIA 2, 3 + CDV Note	1 + 1b
3	18/2	Pixelwise operations, Colour images. PCA Analysis on images (Claes)	Stream	MIA 4, 8 Eigenfaces article (only sections marked with yellow)	2 + 2b
4	25/2	Neighborhood Processing (Filtering) and Morphology (Tim)	Stream Recordings Fall 2023	MIA 5, 6	3 + 8
5	4/3	Blob analysis and object classification (Rasmus)	Stream	MIA 7	4 + 4b
6	11/3	Pixel classification and advanced classification (Tim)	Stream Recordings Fall 2023	MIA 9 + LDA note on Learn	5
7	18/3	Industry presentations:		none	Exercise catch-up
8	25/3	Geometric transformations and landmark based registration (Tim)	Stream Recordings Fall 2023	MIA 10, 11	6
9	1/4	Boundary Tracing (Hough) Transformation and Dynamic Programming (Tim)	Stream Recordings Fall 2023	MIA 12	6b
10	8/4	Advanced registration (Tim)	Stream Recordings Fall 2023	Elastix manual (5.2.0) chapter 2.	7
11	22/4	Real time face detection using the Viola Jones method (Rasmus)	Stream	Rapid Object Detection using a Boosted Cascade of Simple Features	9
12	29/4	Statistical models of shape and appearance and active shape models (Rasmus)	Stream	Statistical Models of Appearance for Computer Vision (p. 12 - 20 and p. 29 - 43)	Exercise catchup
13	6/5	Advanced topics (Claes Nørh Ladefoged)	Stream	none	Digital test exam and exercise catch-up

# Learning Objectives (Læringsmål)

- A list of learning objectives for each lecture and exercise
- A learning objective describes what you can do after the lecture/exercise
- If you fulfil all learning objectives you get 12
- Low-level learning objective
  - Apply the Prewitt edge filter to an image
- High-Level learning objective
  - Evaluate and compare the performance of a selection of image analysis algorithms

# Exam

- Four hours multiple-choice exam
- Please see details here:
  - <http://courses.compute.dtu.dk/02502/exam.html>
- Previous exam sets are also available
  - Most relevant is from Spring 2021 and onwards

# AI assisted tools during the course and the exam

- You are allowed to use AI tools like ChatGPT and Copilot
  - Both during the course and at the exam
- It is your responsibility to
  - install and keep your tools up to date
  - Verify if the output of the tools are correct
- The exam can ALSO be solved without the use of AI assisted tools

# AI related learning objectives

- General 02502 course learning objective:
  - Estimate the correctness of the answer given by an AI assisted tool like ChatGPT and Copilot
- We are gradually adding AI tools related learning objectives to the exercises

PollEverywhere quizzes

<https://pollev.com/rasmuspulse538>



## What programming language are you most comfortable with?



Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)

## What is your practical experience with Python programming

I implement and run Python scripts (.py) in an IDE like Jupyter, PyCharm or Visual studio code

80%

I use Notebooks in JupyterLab or similar

14%

I use google Colab or a similar online service

1%

I use another method or tool for my Python programming

2%

I do not have any Python experience

3%

## What is your experience with Jupyter Notebooks or JupyterLab

I am a complete beginner - I need instructions on software installation and a crash course in Jupyter notebooks.

0%

I have tried it a few times but I need a refresher

0%

I am comfortable with Jupyter notebooks but would not mind a quick refresher

0%

I am all ready - hit me!

0%

I am an expert Jupyter notebook user and have made my own

0%

Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)

## What is your experience with image manipulation, image processing and image analysis?

I have never manipulated an image

5%

I have done cropping, rotation and colour enhancements on my phone or in a photo editor

28%

I have used Photoshop or similar to do advanced image manipulation

23%

I have used an image analysis tool in Python, Matlab, C# or similar

37%

I have implemented and tested my own image analysis program

7%

Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)

## To what extent are you currently using AI assisted tools like ChatGPT and Copilot

105

They are invaluable tool in my learning and university life

17%

I use them actively and often

45%

I am an occasional user

32%

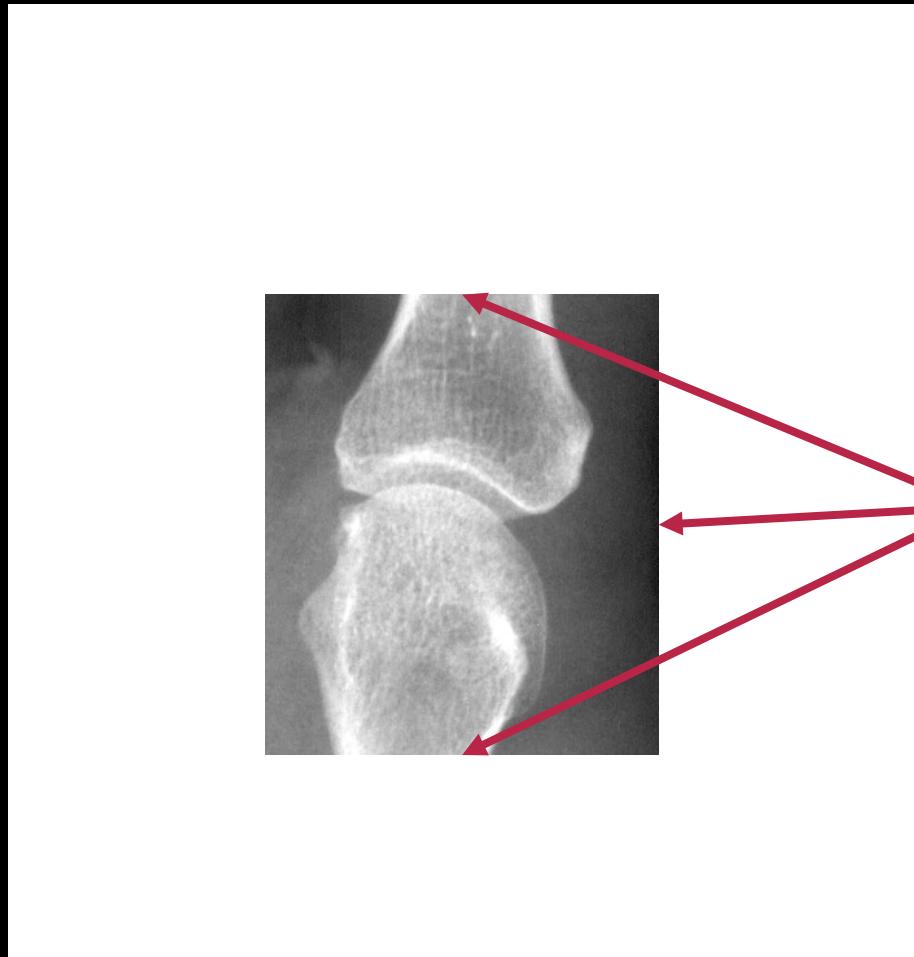
I rarely use the tools

3%

I have never or very rarely used these tools

3%

# Why are my slides black?

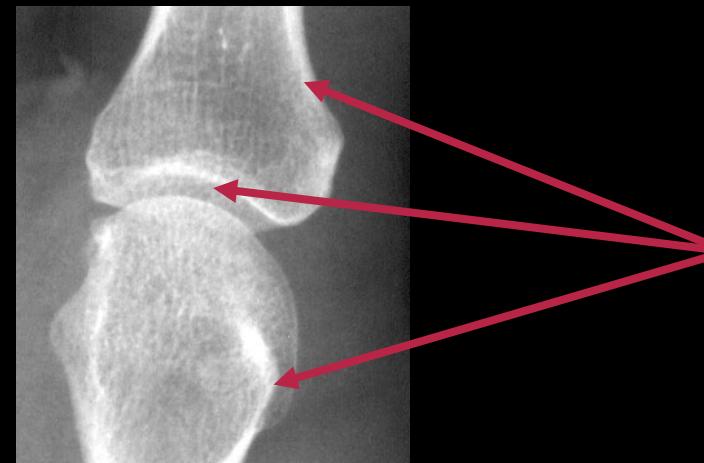


?

Norwegian Black Metal

With a white background,  
the strongest visual  
contrast is here

# Why are my slides black?



With a dark background, the strongest visual contrast is here  
(which I find more important)

# What is image analysis

- Automatic extraction of information from images
- A sub-topic within
  - Pattern recognition
  - Machine learning
  - Deep learning

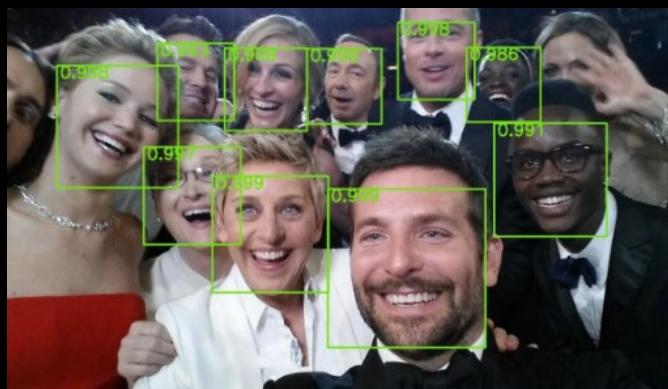
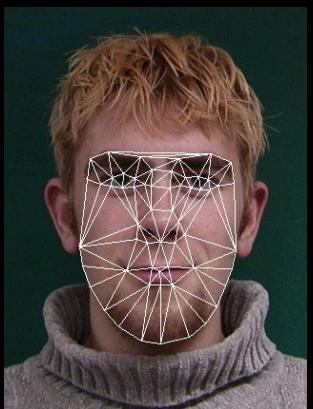
# What is image processing

- Changing the information in images – but not necessarily getting any knowledge
  - Photoshopping
  - Changing the visual appearance of photos
  - Cropping / rotating
  - Filters / effects



# Face tracking – all features including eyes

- For digital cameras / phones
  - Automatic focus on the face + face beautification
- Tracking and manipulation for apps
  - Messenger / WhatsApp / SnapChat ...
- Awareness tracking for car drivers
  - Warning if you fall a sleep



# A 100 million \$ industry



- This image is worth 100 of millions of dollars!
- Well – perhaps not that exact photo.
- The ability to track faces fast and accurate
  - Including estimates of 3D structure
  - App developers pays buckets of money for that
- It all started in 2001 with:  
P. Viola and M. Jones. "Rapid object detection using a boosted cascade of simple features.". CVPR 2001
- Suddenly you could track faces fast and relatively accurate
- Now a lot of focus on deep learning

# Self driving cars

## ■ Modern self driving cars rely on many sensors

- Lidar – radar system
- GPS
- Accelerometers, gyroscopes, magnetometers etc.
- Stereo cameras or multiple cameras
- Lots of advanced image analysis – sensor fusion



# Sports tracking – human body tracking

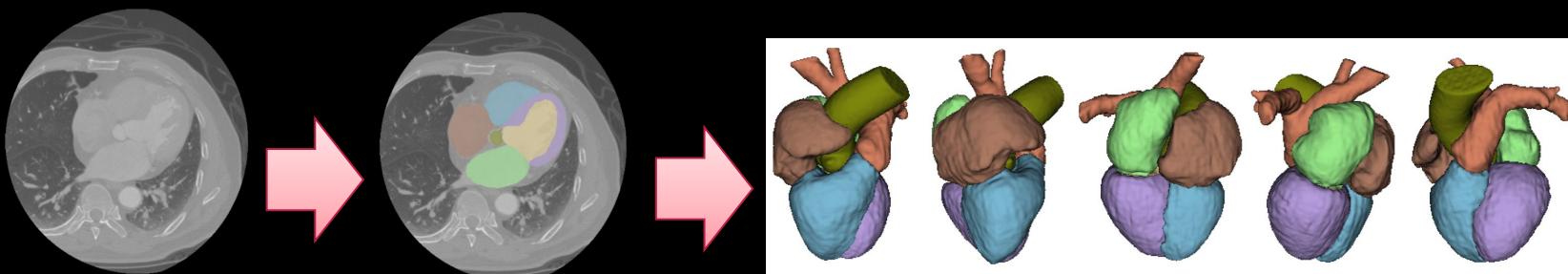


- Huge commercial impact
- Lots of research in human body tracking
- Personal trainers
- TV player tracking and smart overlays



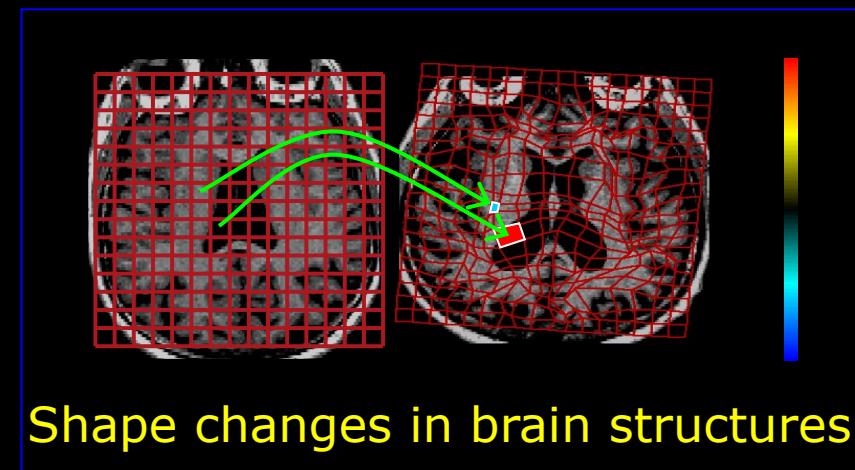
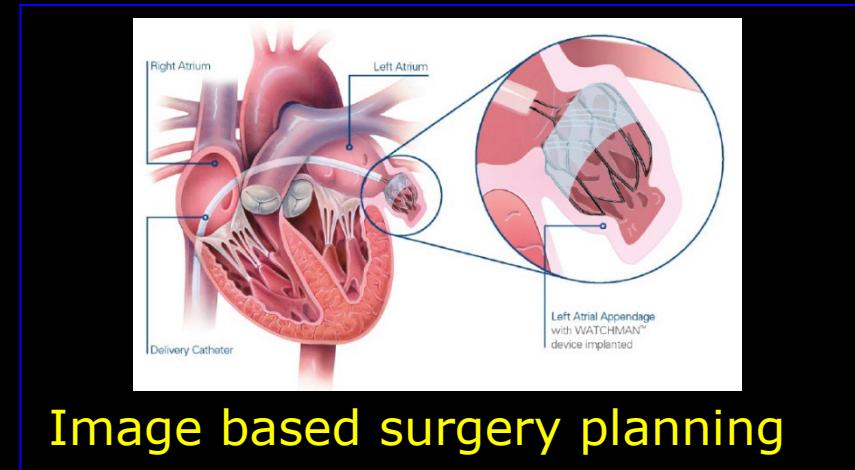
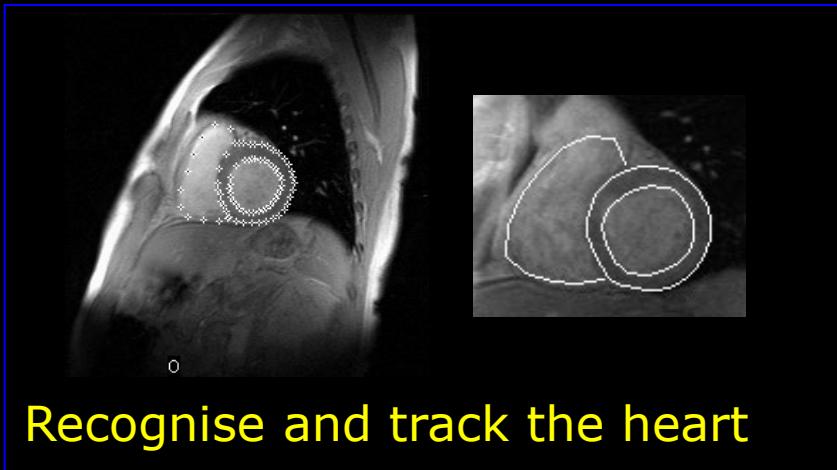
# What is medical image analysis?

- Extraction of information from digital images
- Find unknown connections between diseases and what can be seen in images
- Can enhance the signs of diseases
  - Tumours / heart diseases / brain diseases / bone fractures
- Reproduce expert diagnostics
  - More accurate
  - Variation between doctors opinions removed
- Computer aided diagnostics

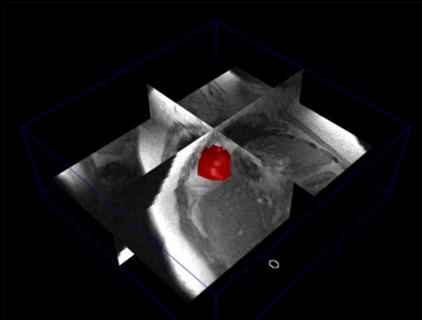


Automatic localization of the heart and its major substructures

# Medical image analysis examples



# Relevance



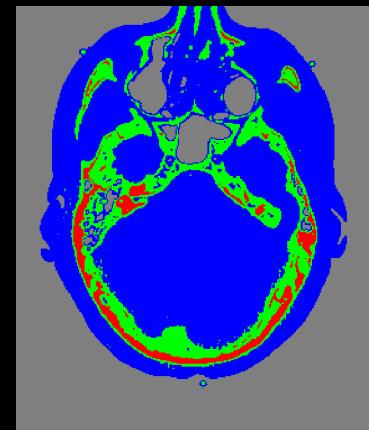
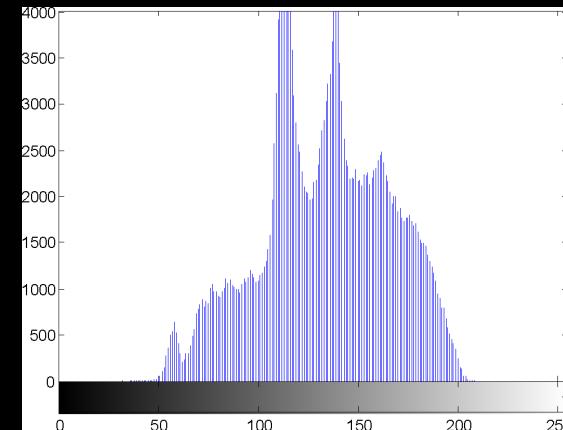
Now – PET/MR

1980  
Magnetic  
resonance  
prototype

- Images is an important tool in
  - Diagnosis
  - Treatment
  - Follow-up
- Very high-tech!
- New imaging technologies are developed all the time.

# Digital Images - Learning Objectives

- Describe the fundamental properties of a digital image
- Describe and use the commonly used image coordinate systems
- Describe pixel types
- Describe the binary, the color, the label, the multispectral, the floating point, and the 16-bit image



# A digital image

23	216	120	55
4	89	158	130
65	76	189	34
19	234	7	45

- Consists of pixels (picture elements)
- Each pixel has a value between 0 and 255? Why?

# Bits and Bytes!

- A **bit** is a tiny tiny little switch that can be either 0 or 1 – the “memory of a computer” consists of insanely many bits
- One **byte** is 8 bits together. It is the “basic” unit in a computer.
- With 8 bits how many possible values can be made?
  - $(2^8 = 256)$
  
- 00000001 = 1
- 00000010 = 2
- 00000100 = 4
- 00001010 = 10
- 00001111 = 15
- 11111111 = 255

128	64	32	16	8	4	2	1
<input type="checkbox"/>							



What is decimal 67 as a binary number?

1010 1010

0100 0011

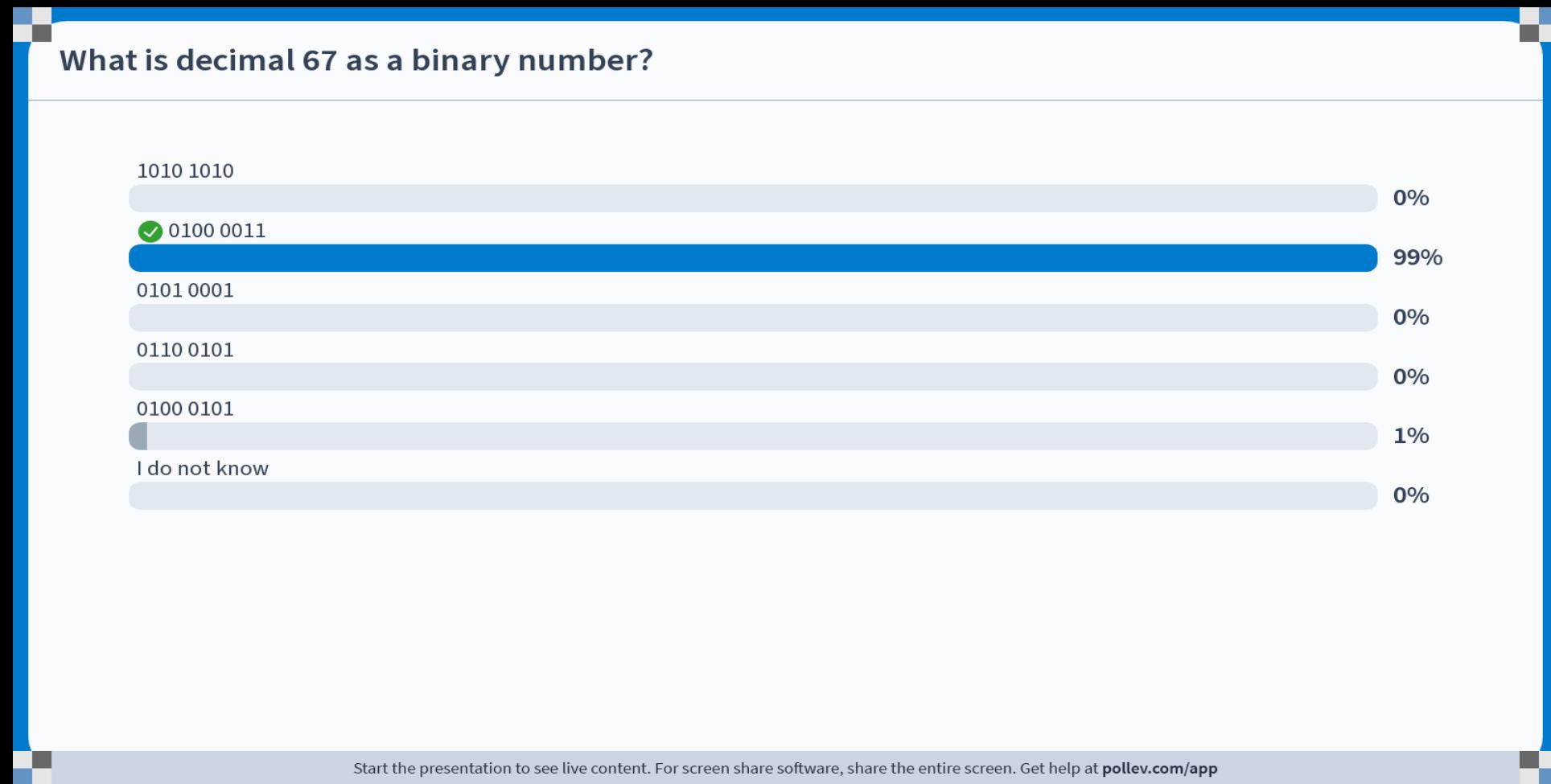
0101 0001

0110 0101

0100 0101

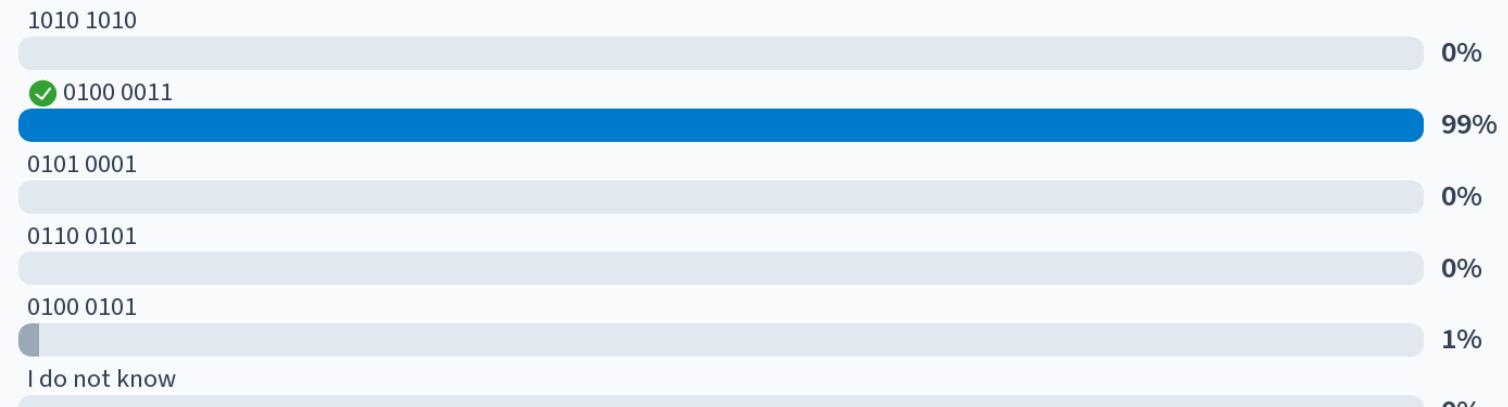
I do not know

Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)





### What is decimal 67 as a binary number?



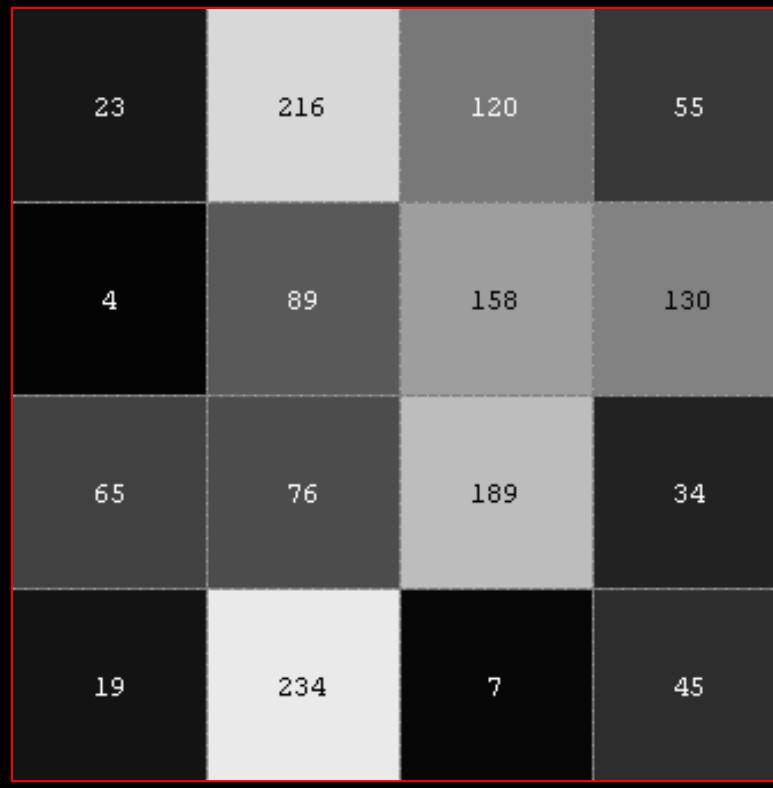
Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)

# A digital image

23	216	120	55
4	89	158	130
65	76	189	34
19	234	7	45

- between 0 and 255.
- The pure image data takes up 16 bytes of computer memory

# Grayscale digital images



- 0 is black and 255 is white!
- The values in between are shown as shades of gray



# Typical Grayscale image



- Traditional film X-ray
- Scanned on a flatbed scanner
- Bone is white and air is black
  - The more radiation the darker
- What are they used for?
  - Fractures
  - Arthritis
  - Osteoporosis

# Image Resolution

- Determines how much the image fills in the memory and on the hard disk
- Spatial resolution
- Gray level resolution

# Spatial?

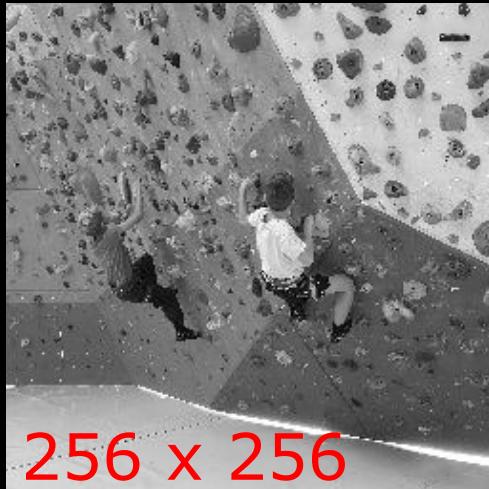
- Spatial
  - relating to the position, area and size of things

- Example:
  - This task is designed to test the child's *spatial* awareness

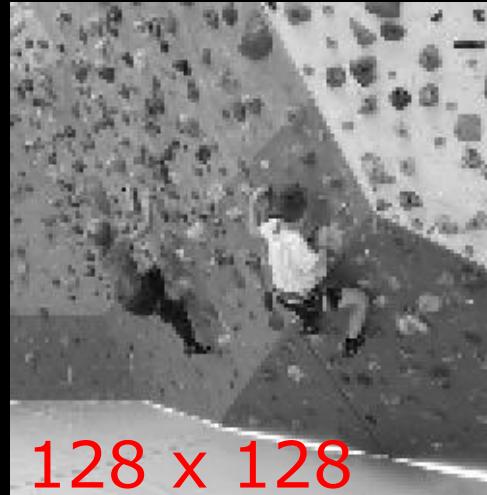
- Danish
  - Rumlig – jeg har en god rumlig forståelse

# Spatial resolution

The number of pixels used to represent the image



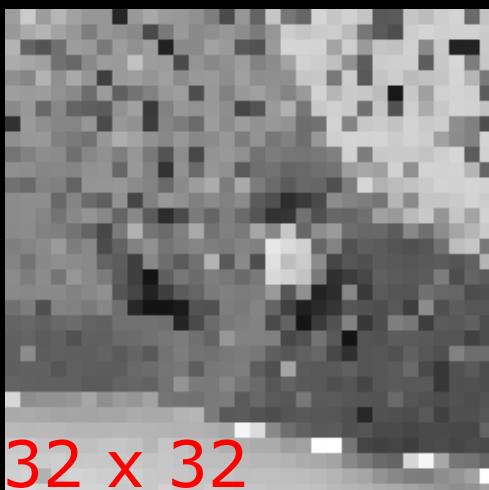
256 x 256



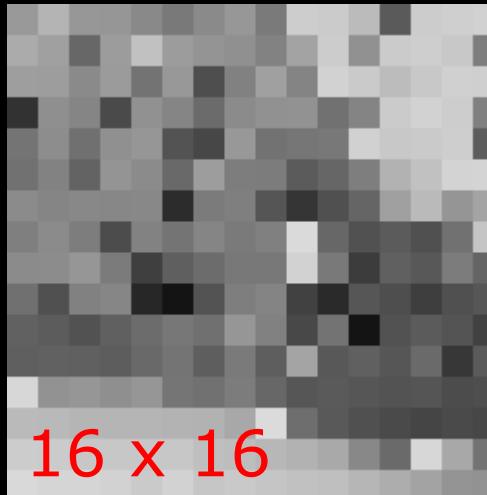
128 x 128



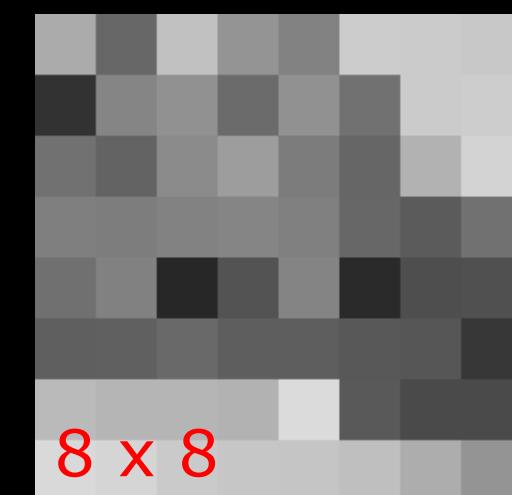
64 x 64



32 x 32

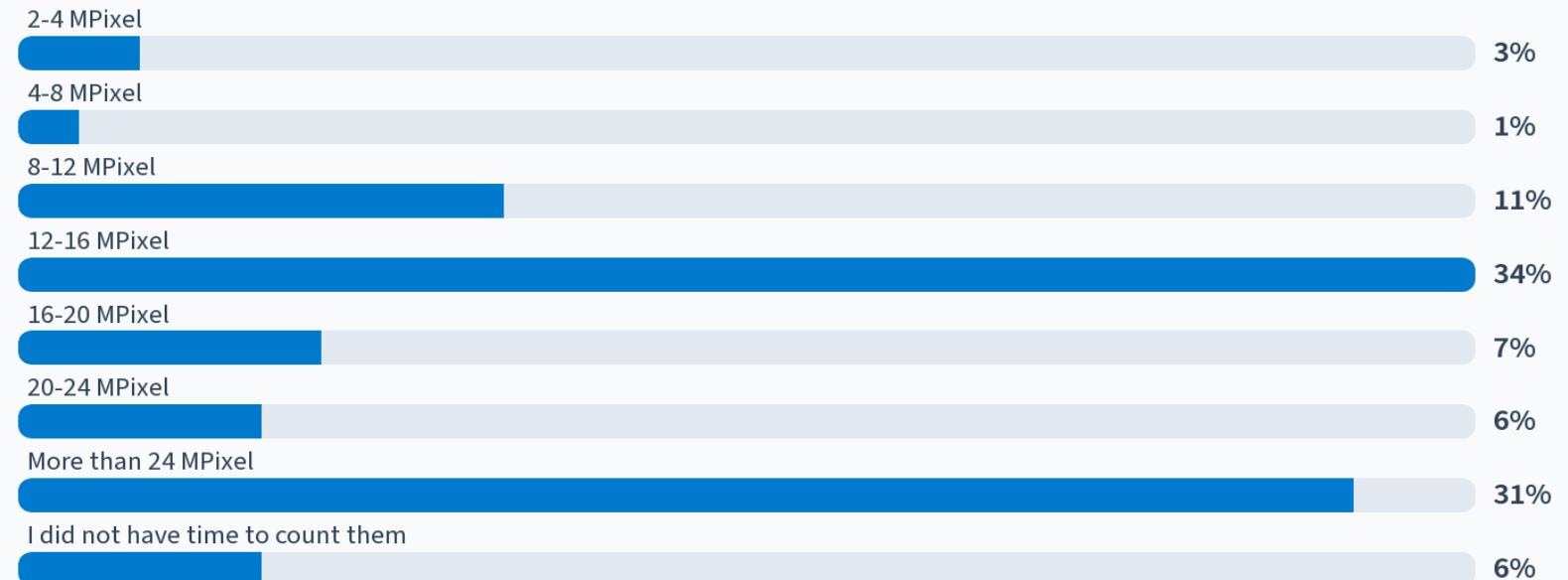


16 x 16



8 x 8

### How many megapixels (approximately) do the photos you take with your camera or phone have?



Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)

# How many pixels?

Width	Height	Pixels	Mega-pixels	Camera
320	240	10.000	0.01	Prototype 1975
1600	1200	1.920.000	2	Nikon Coolpix 950
4032	3024	12.192.768	12	Samsung Galaxy S7 edge
6240	4160	26.000.000	26	Canon EOS 6D M2
8984	6732	60.480.288	60.5	Phase One P65+

# Grey level resolution

## The number of grey levels in an image



256



64



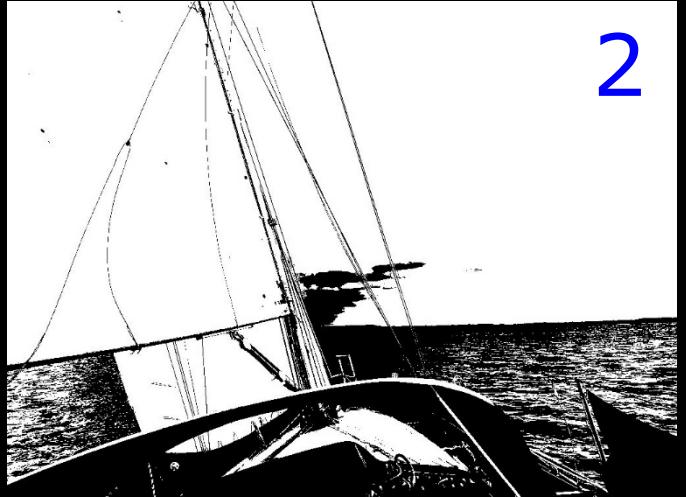
16



8

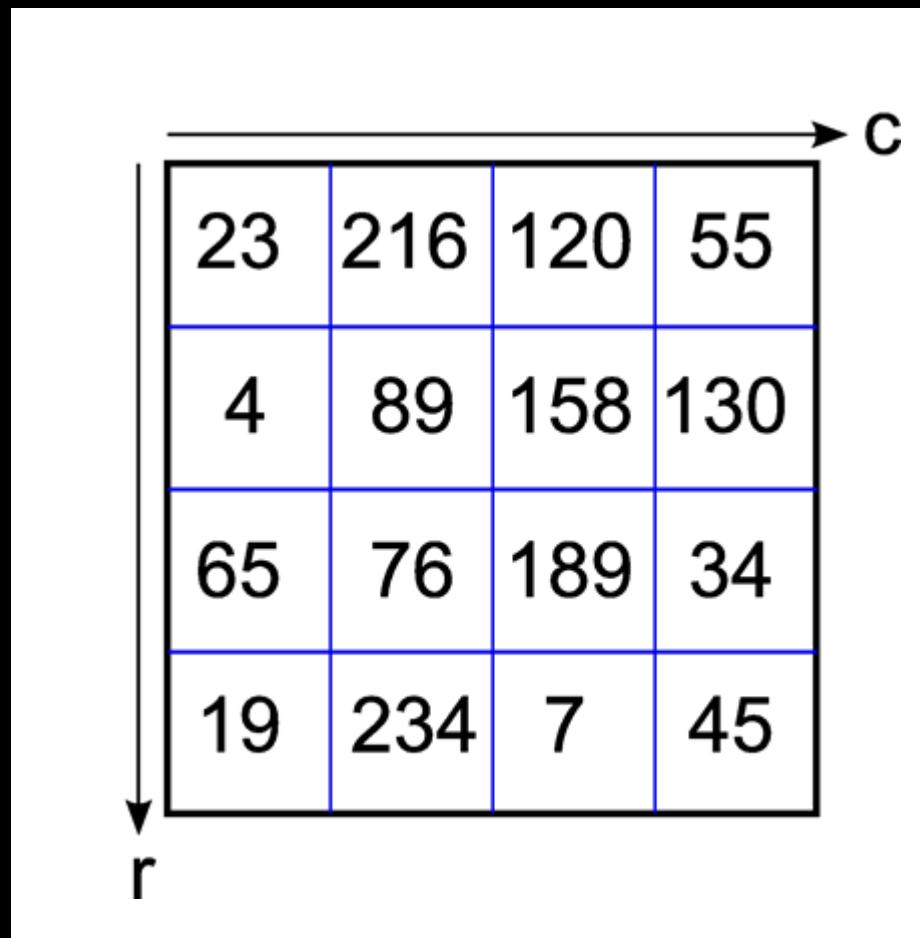


4



2

# An image as a matrix



- An image is stored in the computer memory as a 2 dimensional matrix
- 4 rows and 4 columns
- Can also be seen as a discrete function  $f(r, c)$
- In Python a pixel can be stored as an `uint8`
- `uint8` = Unsigned 8-bit integer = 1 byte

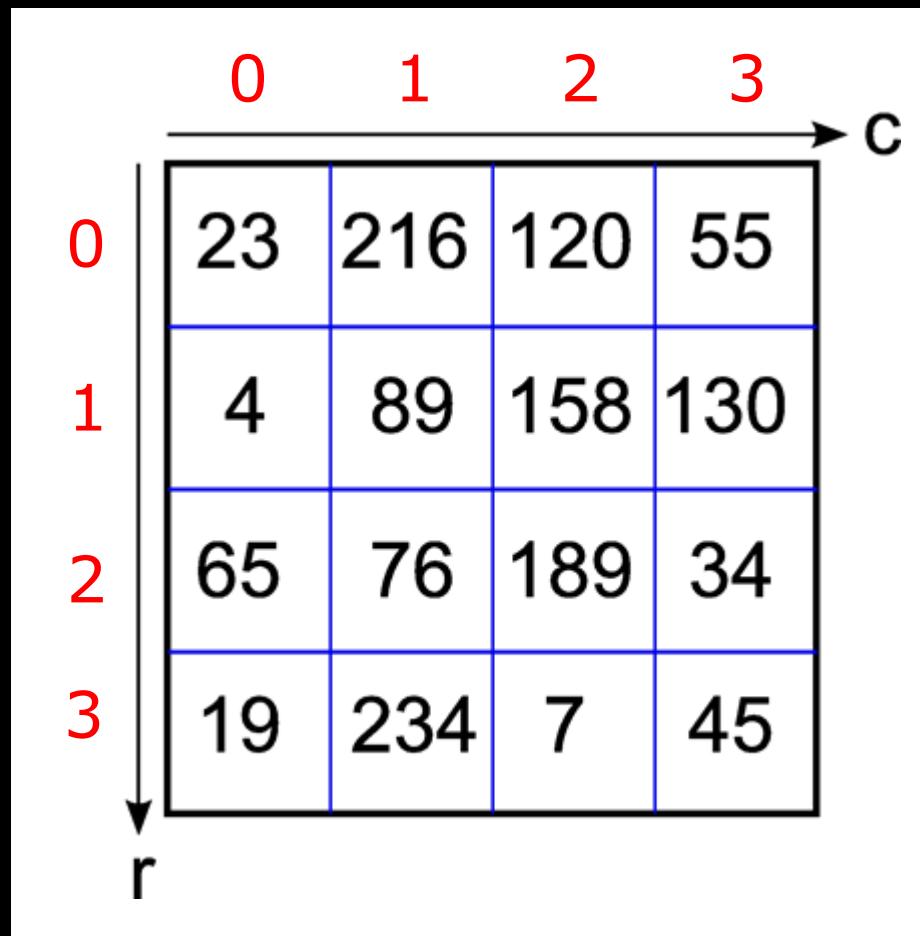
# Pixel types and their ranges

Data type	Range
uint8	0 to 255
uint16	0 to 65535
uint32	0 to $2^{32} - 1$
float	-1 to 1 or 0 to 1
int8	-128 to 127
int16	-32768 to 32767
int32	$-2^{31}$ to $2^{31} - 1$

- A pixel can be processed and stored as different *types*
- The uint8 is the most common type
- For processing a pixel is often transformed to a float
- When processing speed and memory space is an issue you should be careful about the pixel type – more about that later in the course.

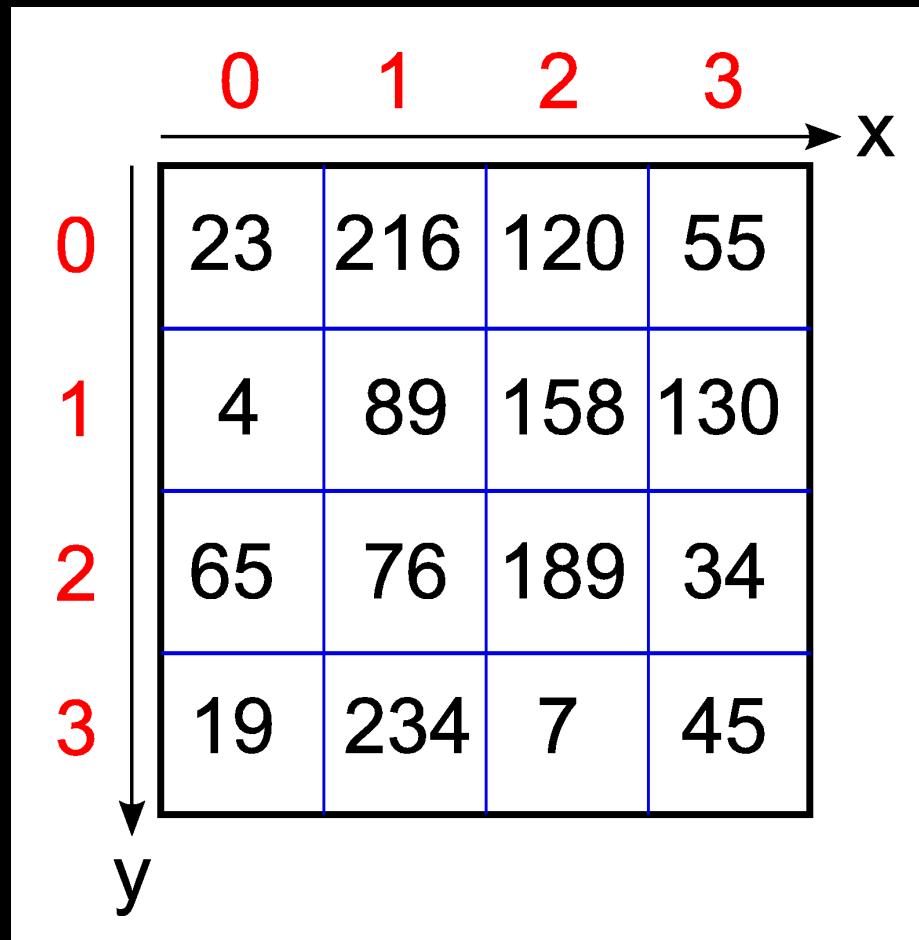
[https://scikit-image.org/docs/stable/user\\_guide/data\\_types.html](https://scikit-image.org/docs/stable/user_guide/data_types.html)

# Pixel coordinates – Python matrix



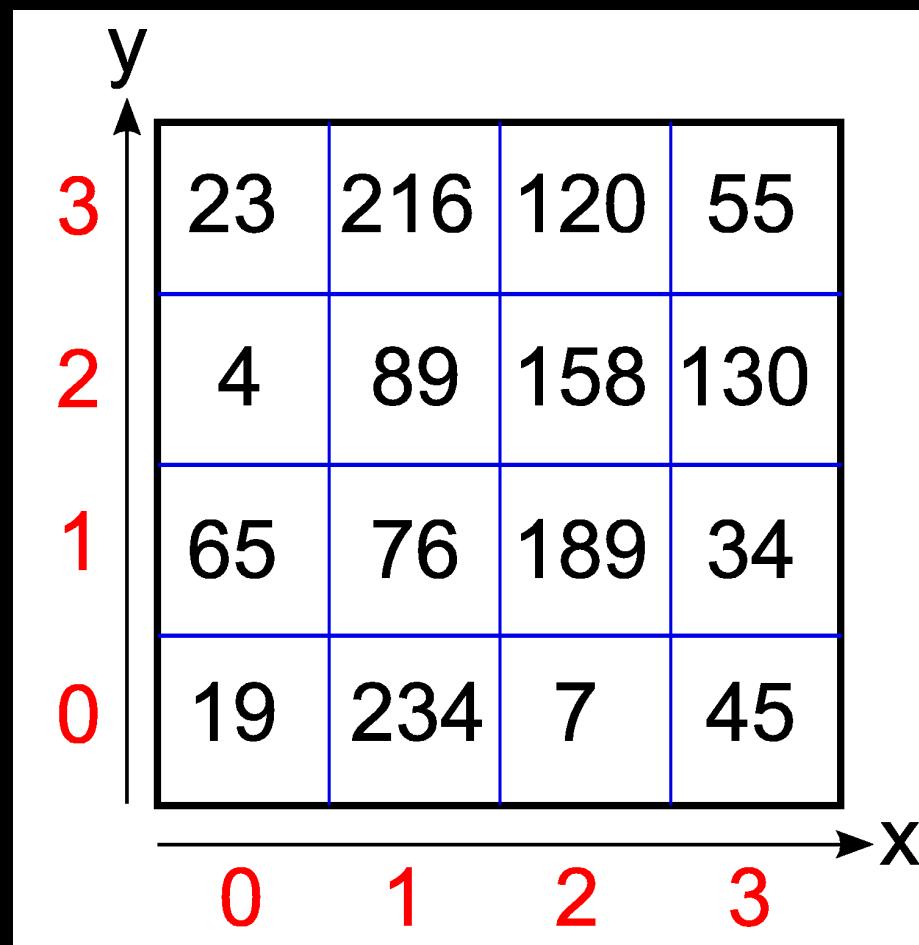
- Origin is in upper left corner
- 0-based
- (row, column) system
  - Vertical axis is the first axis
- M rows and N columns
- Row range [0, M-1]
- Column range [0, N-1]

# Pixel coordinates – Photoshop etc.



- Used in many graphics programs
- Origin in upper left corner
- 0-based
- (X,Y) system
  - Horizontal axis is the first coordinate
- Often width (W) and height (H) are used to denote image dimensions
- X range [0, W-1]
- Y range [0, H-1]

# Plot coordinates

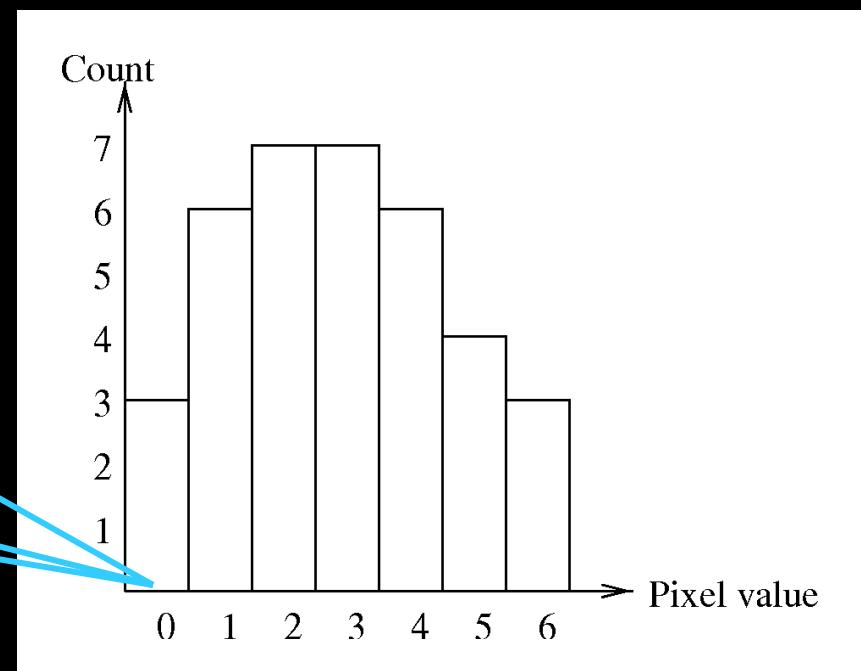


- Used when plotting – known from mathematics
- Origin in lower left corner
- 0-based
- ( $X, Y$ ) system
  - Horizontal axis is the first axis

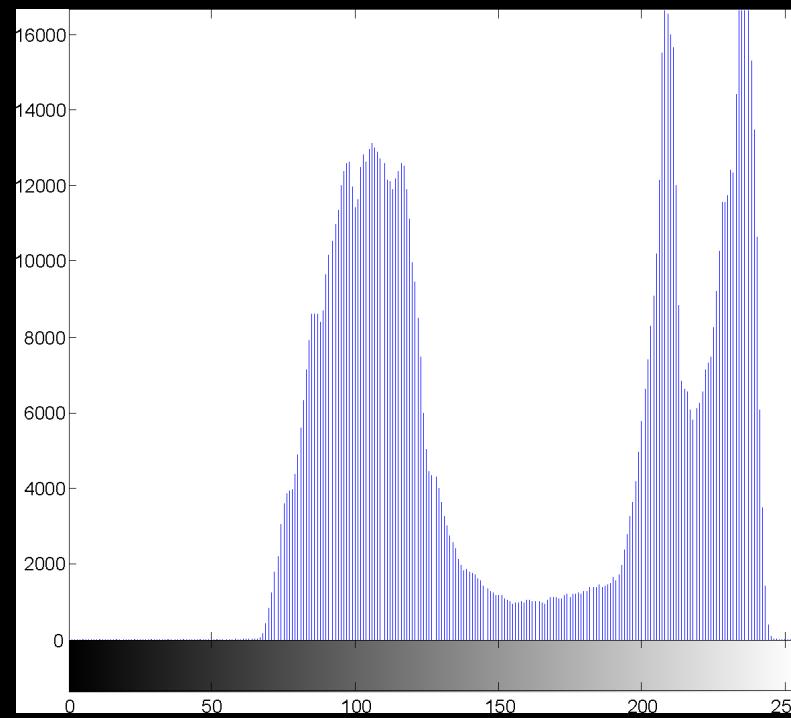
# The Image Histogram

- A histogram normally contains the same number of “bins” as the possible pixel values
- A bin stores the number of pixel with that value

0	2	6	6	3	3
1	4	3	4	4	4
3	2	5	1	5	2
1	4	2	1	3	1
2	5	3	0	2	0
4	2	5	6	3	1

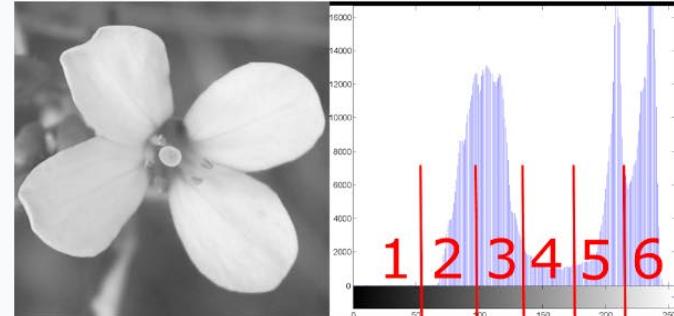


# A real grayscale image histogram



- 256 gray levels in the image = 256 bins in the histogram
- The shape of the histogram tells us something about the image

Where are the flower leaves in the histogram?



Range 1

Range 2

Range 3

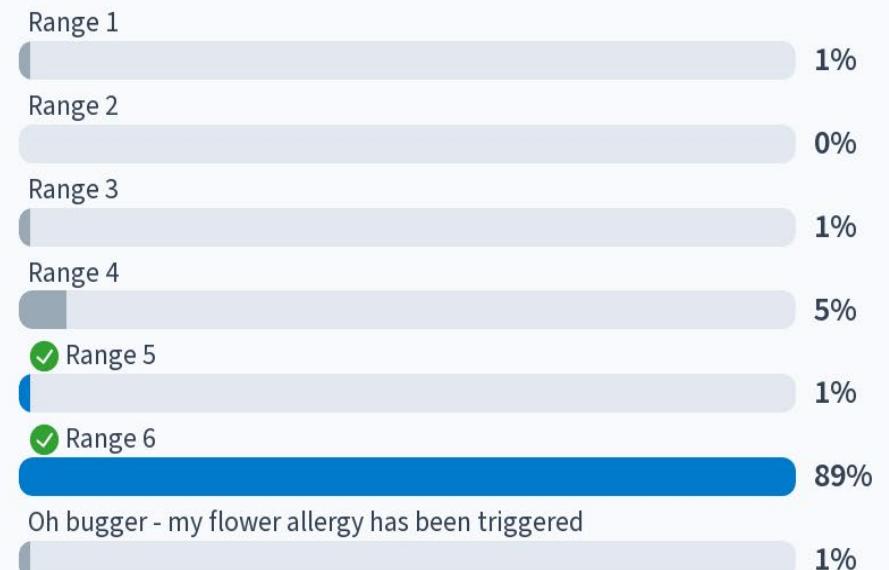
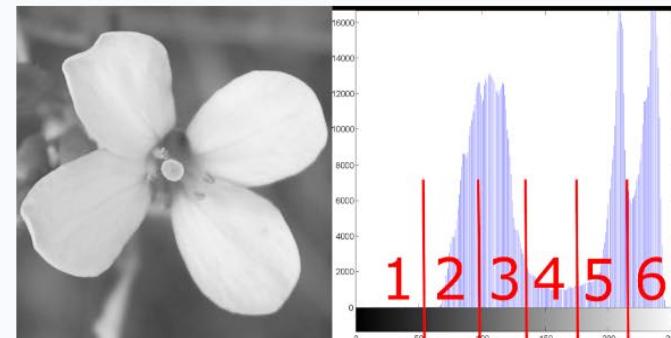
Range 4

Range 5

Range 6

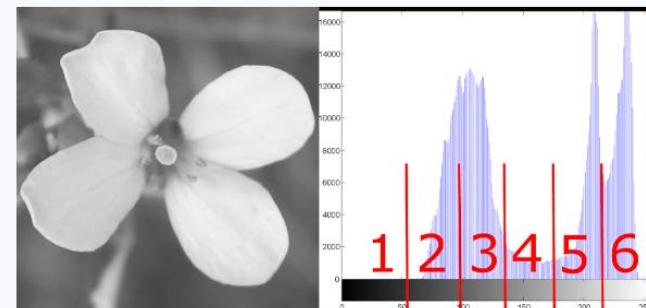
Oh bugger - my flower allergy has been trigge...

## Where are the flower leaves in the histogram?



Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)

## Where are the flower leaves in the histogram?



Range 1 1%

Range 2 0%

Range 3 1%

Range 4 5%

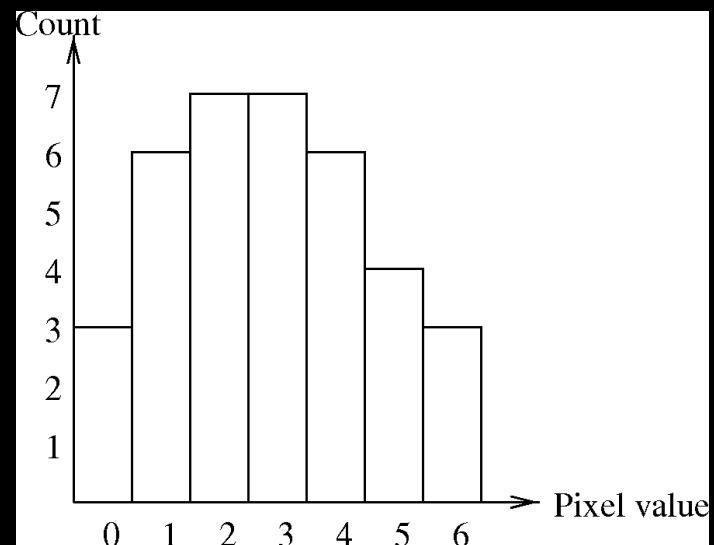
Range 5 1%

Range 6 89%

Oh bugger - my flower allergy has been triggered 1%

# Pixel value statistics

0	2	6	6	3	3
1	4	3	4	4	4
3	2	5	1	5	2
1	4	2	1	3	1
2	5	3	0	2	0
4	2	5	6	3	1



- Pick a random pixel in the image
- What is the probability of it having value 3?  $P(v=3)$
- $h(3) = 7$
- $N_p = 36$
- $P(v=3) = 7/36 * 100\%$
- The histogram divided by the total pixel count can be seen as a probability density function



A random pixel is chosen in the image. What is the probability that the value of the pixel is 3?

6%

28%

39%

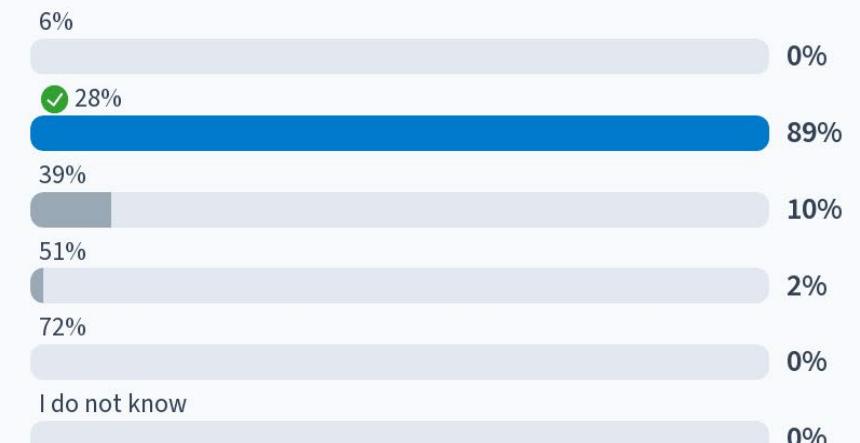
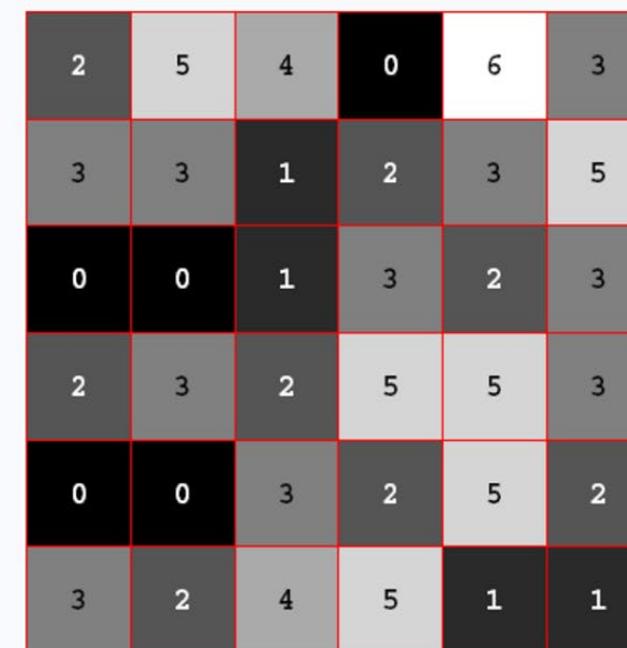
51%

72%

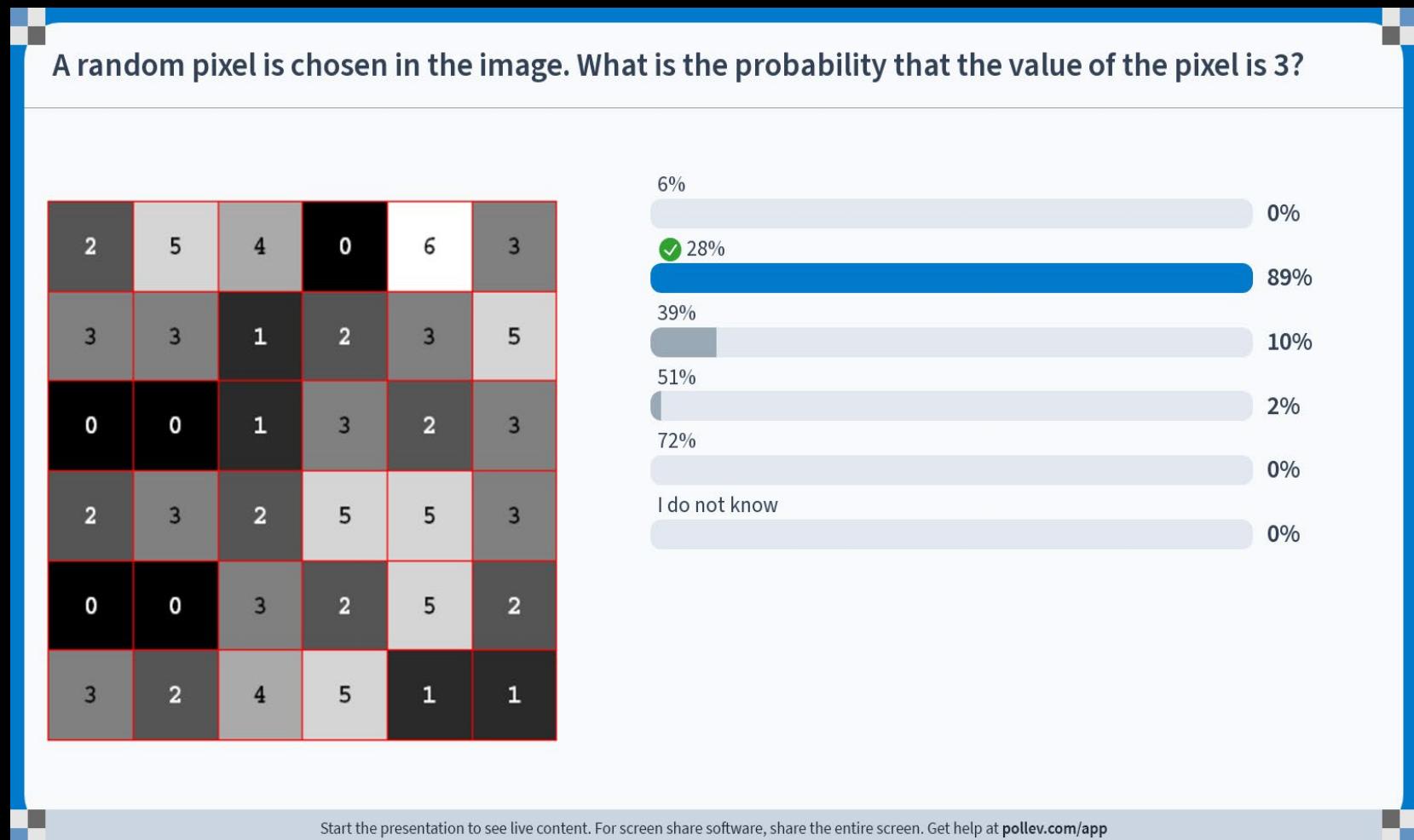
I do not know

Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)

A random pixel is chosen in the image. What is the probability that the value of the pixel is 3?



Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)



# Other Image Types

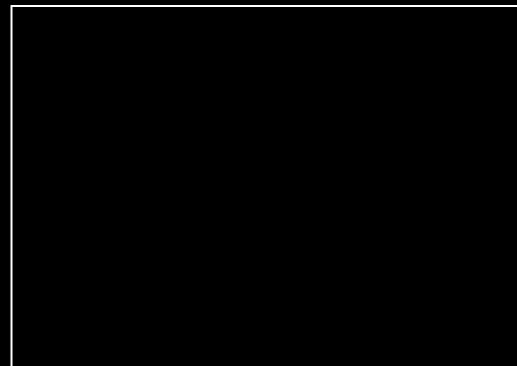
- Colour images
- Binary Images
- Label Images
- 16-bit images
- Floating point images

# Colour images



- RGB = Red, Green, and Blue
- Television, computers, digital cameras use the “RGB color space”
- Additive colours: Final colour is made by mixing red, green, and blue
- Typically the values of R, G, and B lie between 0 and 255 (total 3 bytes)!

# RGB Colours

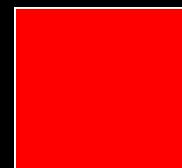


RGB = (0,0,0)

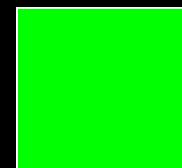


RGB = (255,255,255)

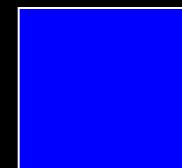
- When alle three "Lamps" are turned of we get black
- When all three "lamps" are on what do we get?



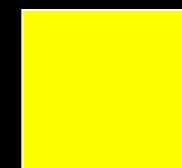
(255,0,0)



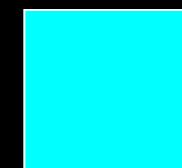
(0,255,0)



(0,0,255)



(255,255,0)

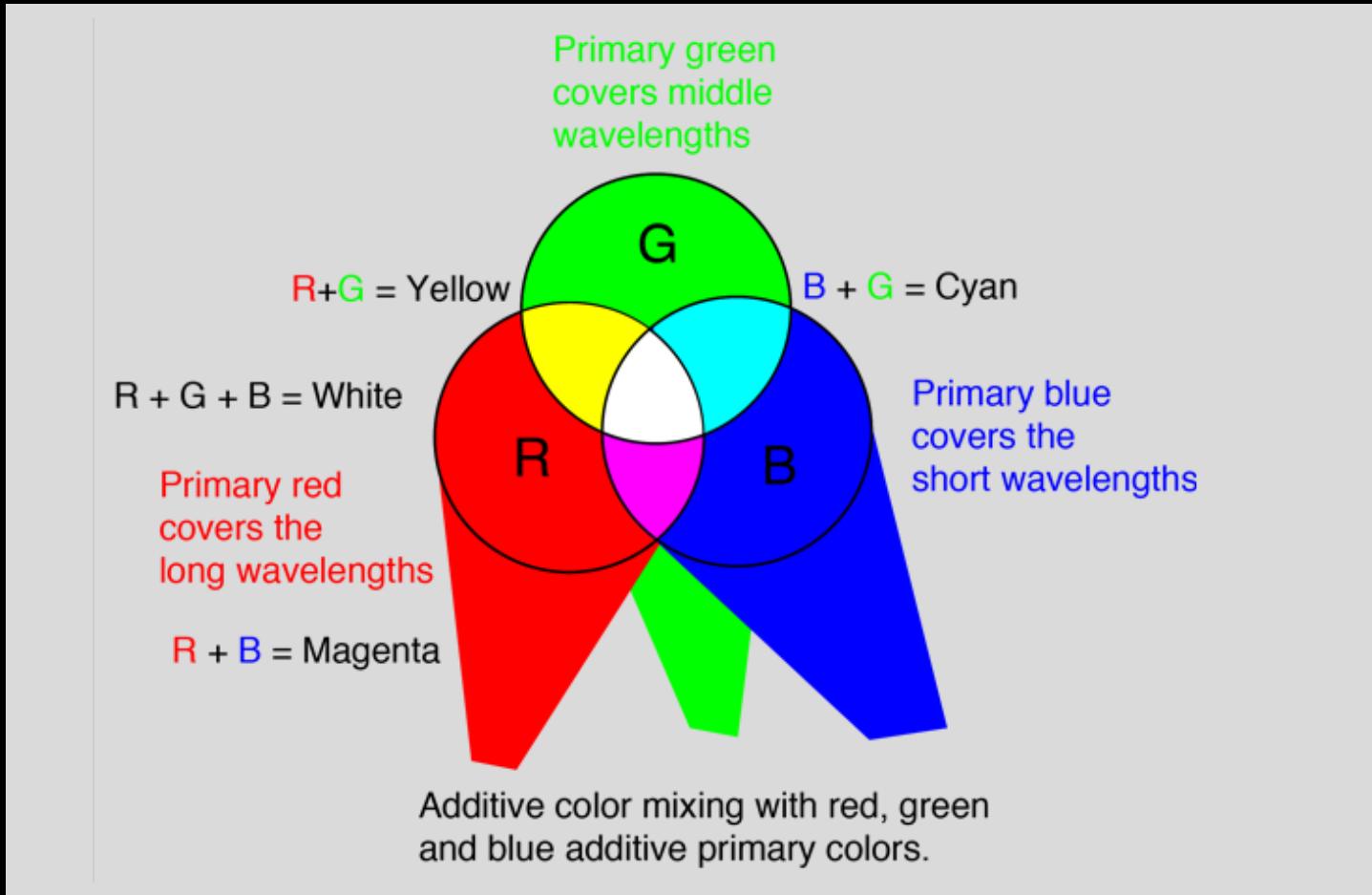


(0,255,255)



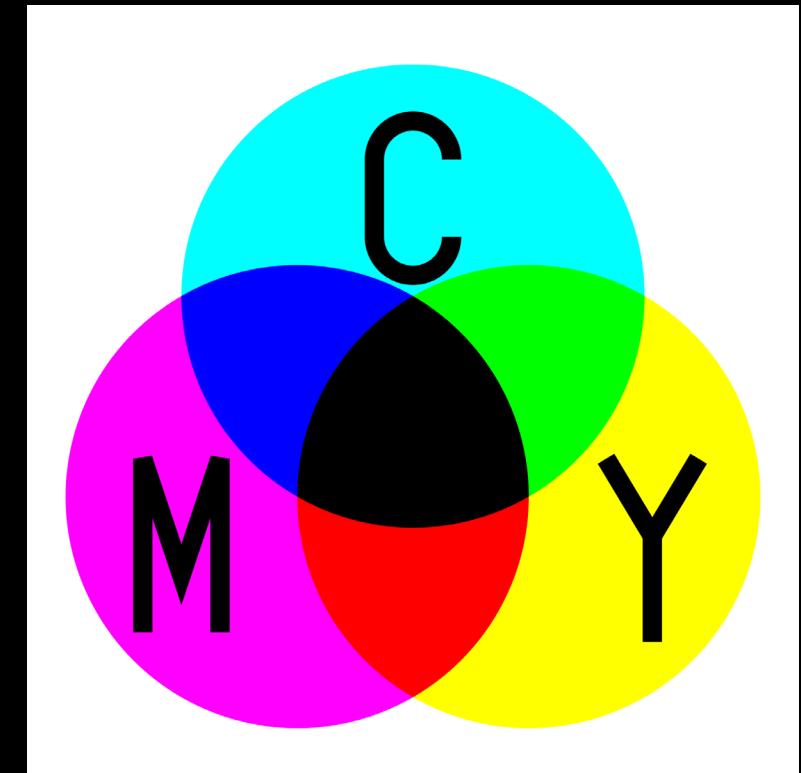
(255,0,255)

# Additive color mixing



<http://hyperphysics.phy-astr.gsu.edu/hbase/vision/addcol.html>

# Subtractive color mixing



Wikipedia

# Processing RGB images

- Each pixel in a colour image contains 3 values
- Equal to a “vector function” in mathematics
- More complicated to analyse
- Medical images are typically grayscale
  - Why?
- Often images are converted from colours to grayscale before the analysis

# Converting colour to grayscale

$$V = 0.2989 * R + 0.5870 * G + 0.1140 * B$$



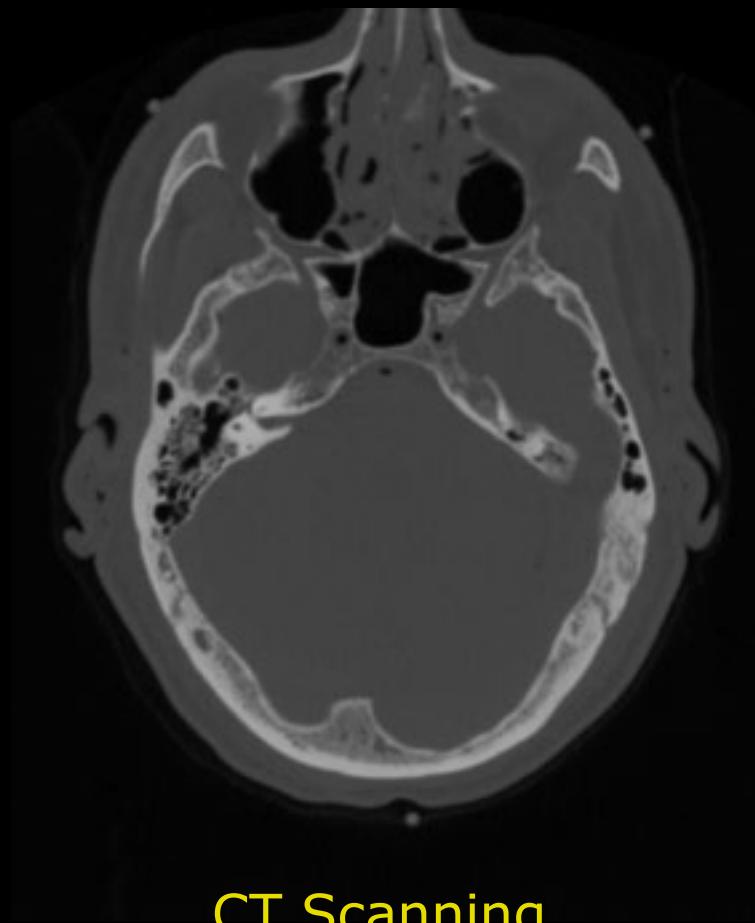
Is it possible to convert a grayscale image back to a color image?

# Binary images



- Binary – means on or off
- Binary image – only two colors
- Background (0 = black)
- Foreground (1 = white)
  
- Simple representation of CT scanning of the head

# Gray scale to Binary Image



CT Scanning



Threshold



“Bone Image”

# Binary image – why?

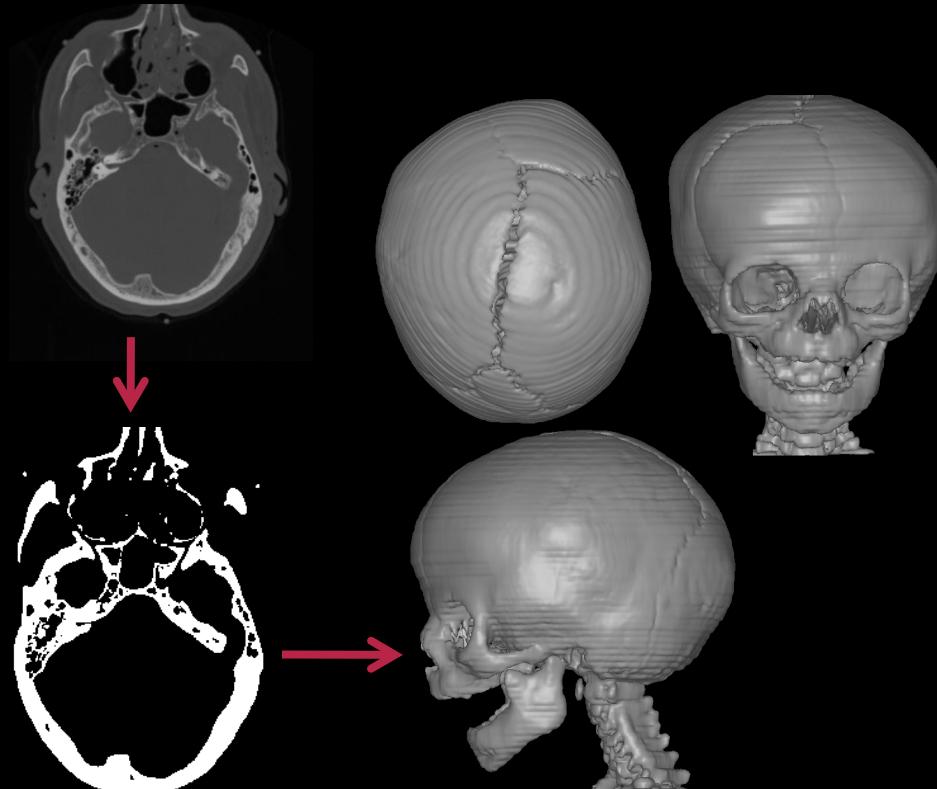
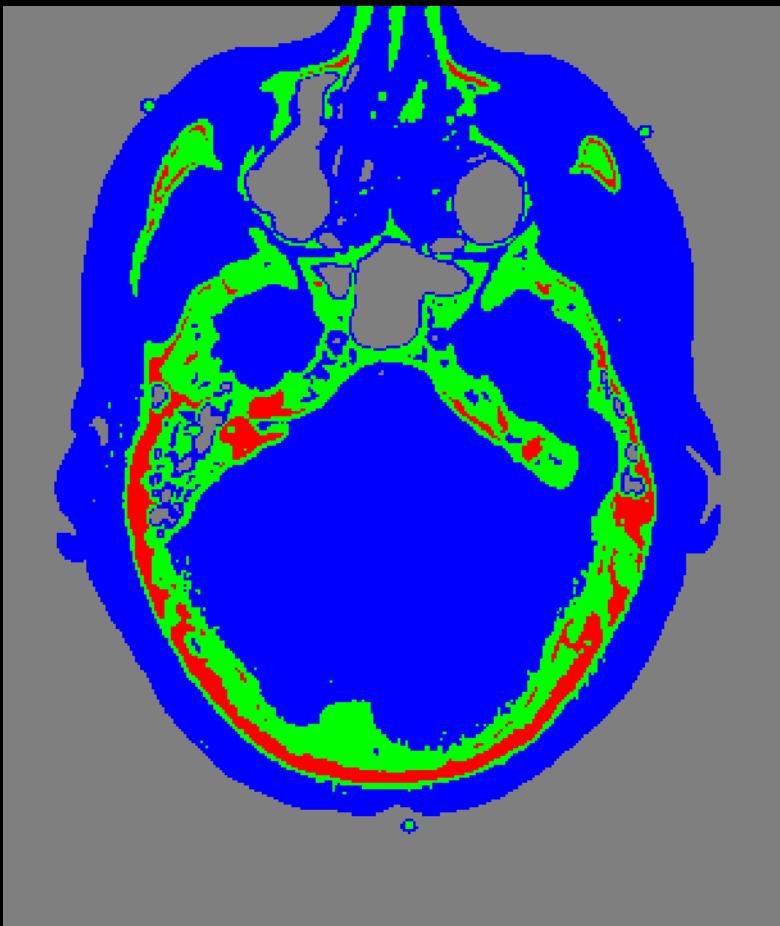


Image from 3D laboratory

- Separating objects from background
- Count the number of the objects
- Measure the size and shape of objects
- Advanced 3D visualisations

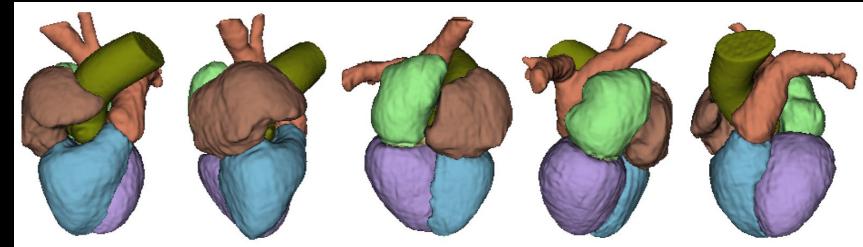
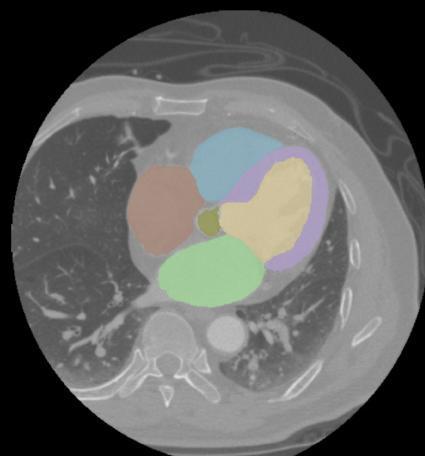
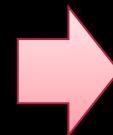
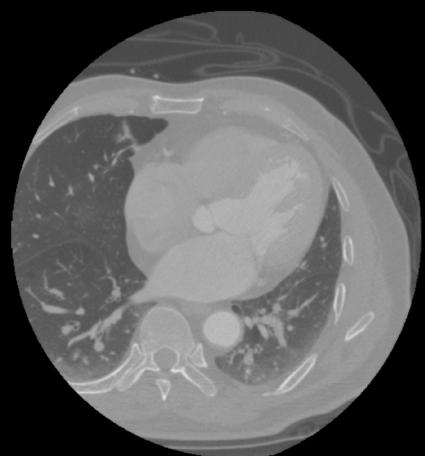
# Label Image



- The pixel value tells the *type* of the pixel
  - (0) Gray – background
  - (1) Blue – soft tissue
  - (2) Green – hard bone
  - (3) Red – spongy bone
- Only 4 different pixel values
- Colours made using a *look-up-table*

# Label Image – why?

- Segment images into regions
- **Example:** Recognize the major structures of the human heart as seen in a computed tomography image. Construct a 3D model of a given patient heart. Use the 3D model for diagnostics and surgery planning.



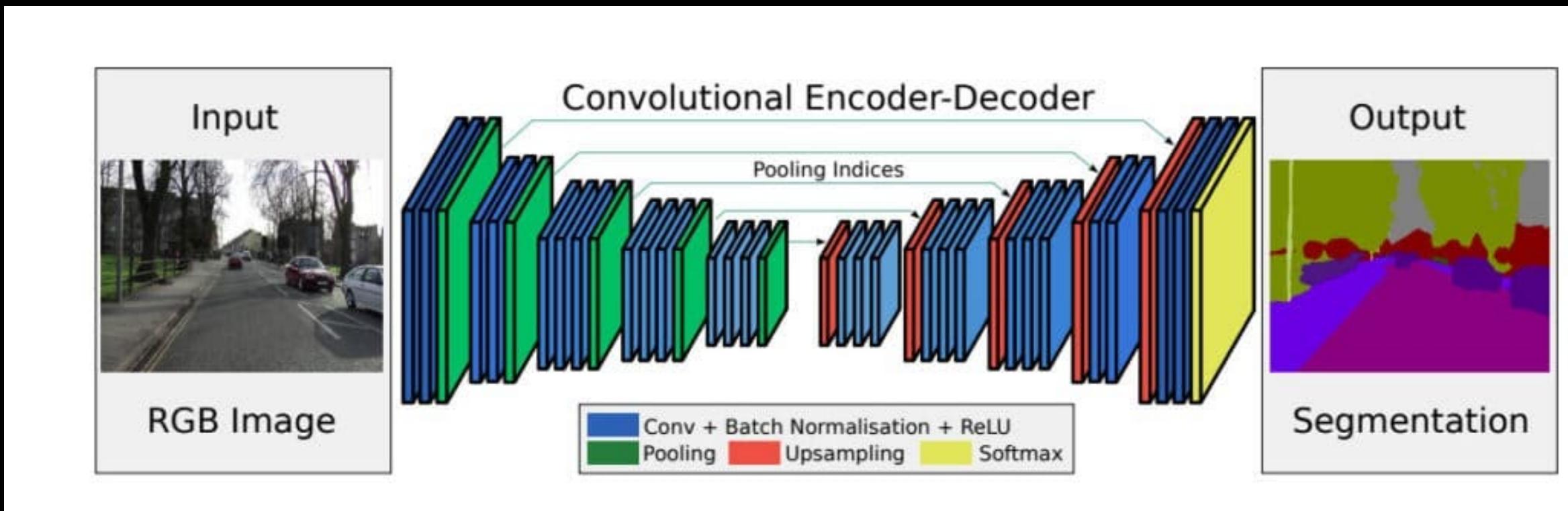
# Label image from semantic segmentation

- Scene understanding for self navigating vehicles



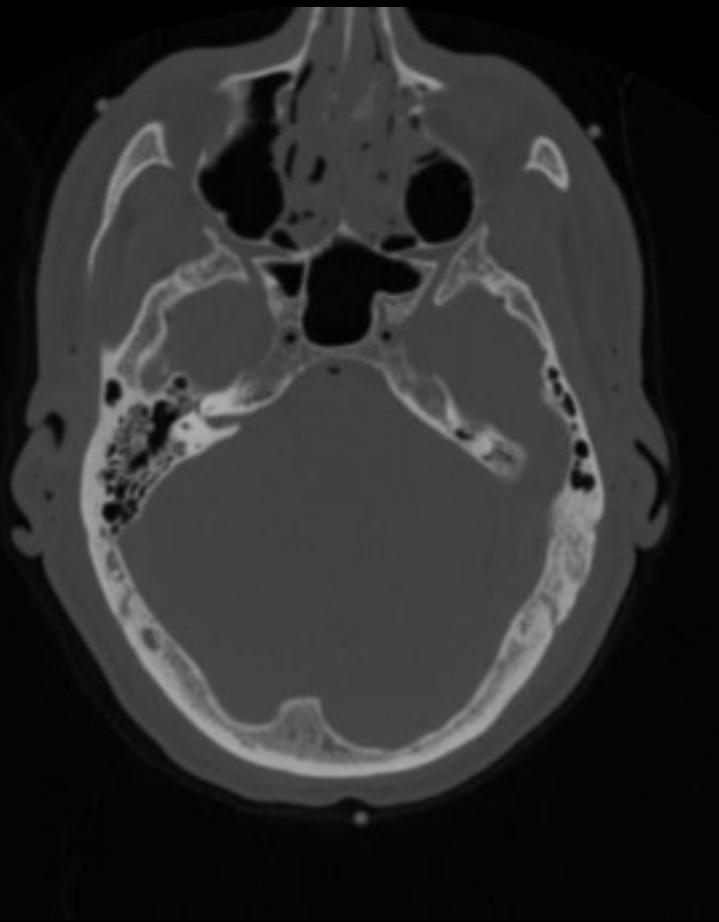
<https://towardsdatascience.com/semantic-segmentation-of-150-classes-of-objects-with-5-lines-of-code-7f244fa96b6c>

# Deep learning for semantic segmentation



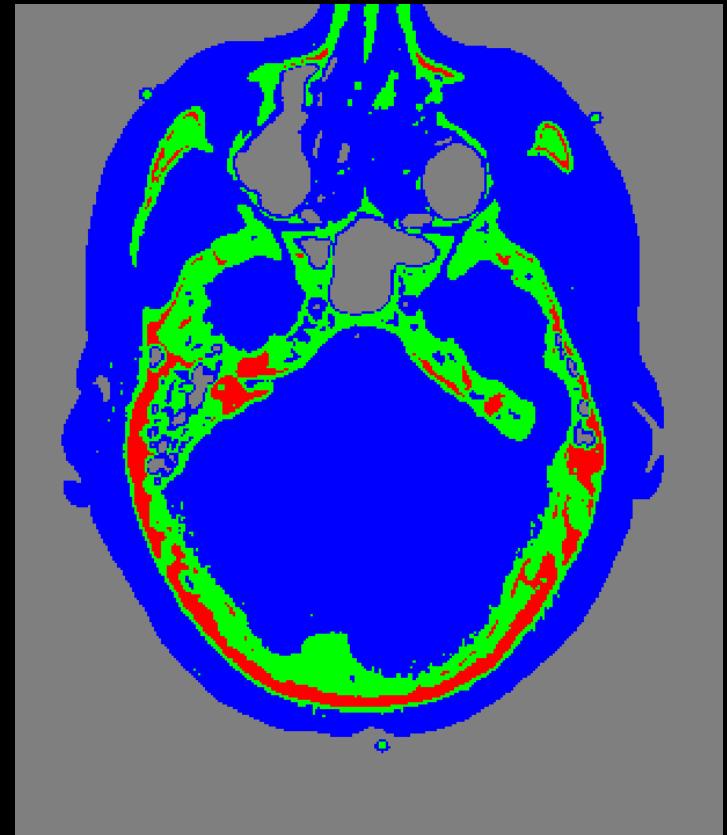
Badrinarayanan, Vijay, Alex Kendall, and Roberto Cipolla. "Segnet: A deep convolutional encoder-decoder architecture for image segmentation." *IEEE transactions on pattern analysis and machine intelligence* 39.12 (2017): 2481-2495.

# Label images in this course



Pixel Classification

BLOB analysis and classification



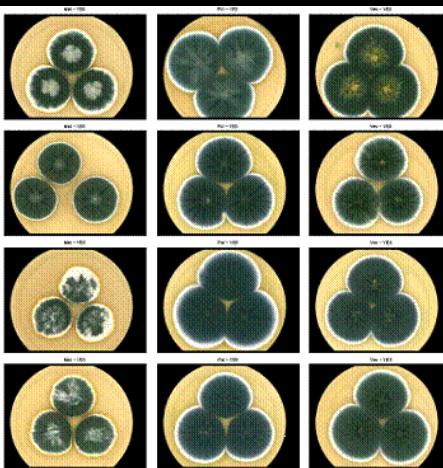
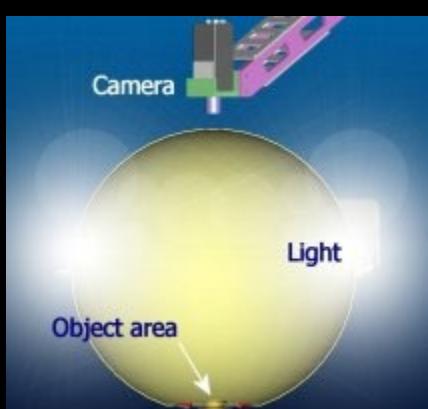
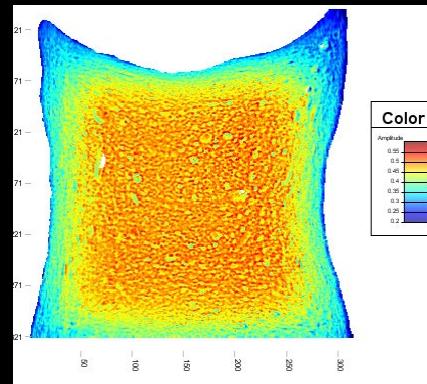
# Multispectral images



Infrared

- There are more visual information than what can be seen with the human eye
- Standard cameras captures the red, green, blue colours
- Capture systems that capture more bands and other frequencies exist
- Creates multispectral images
  - Each pixel contains perhaps 20 values from different spectral bands

# Multispectral System - VideometerLab



- Integrating sphere
- Light emitting diodes with different wavelengths
  - From near infrared to ultraviolet
- High resolution camera
- Water in bread
- Classification of fungi
- Skin diseases

# 16-bit images



- 256 values fine for the human eye
- Pixel values not only for display
  - Physical meaning
- Computed Tomography
  - X-ray attenuation
- Hounsfield units
  - 0 water
  - -1000 air
  - -120 fat
  - 400+ bone

# Floating point images

- The pixel type is often changed when applying image processing functions
- For example when scaling an image, the output will be a floating point image:

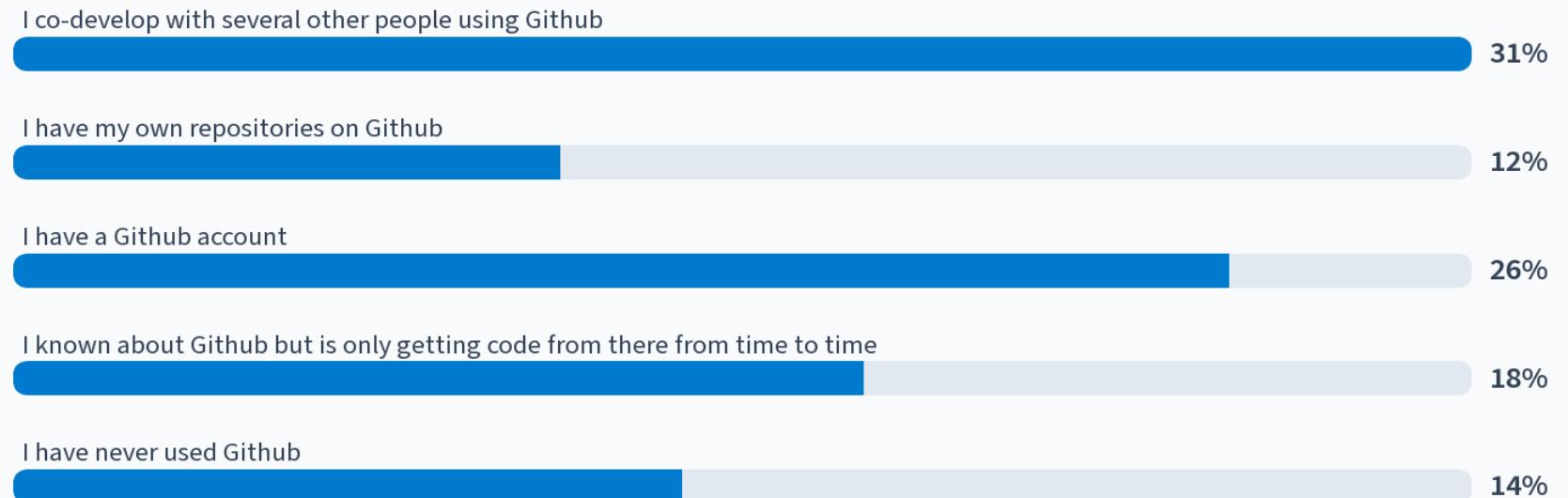
```
image_rescaled = rescale(im_org, 0.25, anti_aliasing=True, channel_axis=2)
print(image_rescaled.shape)
print(image_rescaled.dtype)
```

# Python scripts vs. Jupyter Notebooks

- In this course, you can do the exercises and the exam in both Jupyter Notebooks or as Python scripts
- Strengths and benefits of both approaches



## What is your experience with Github?



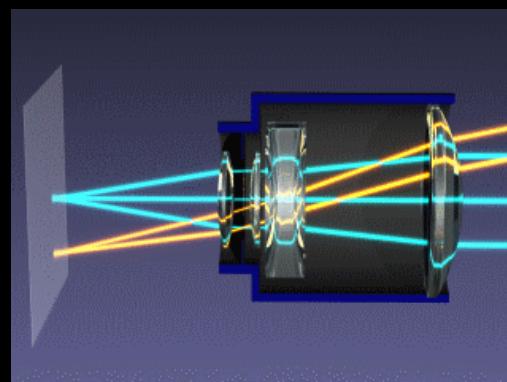
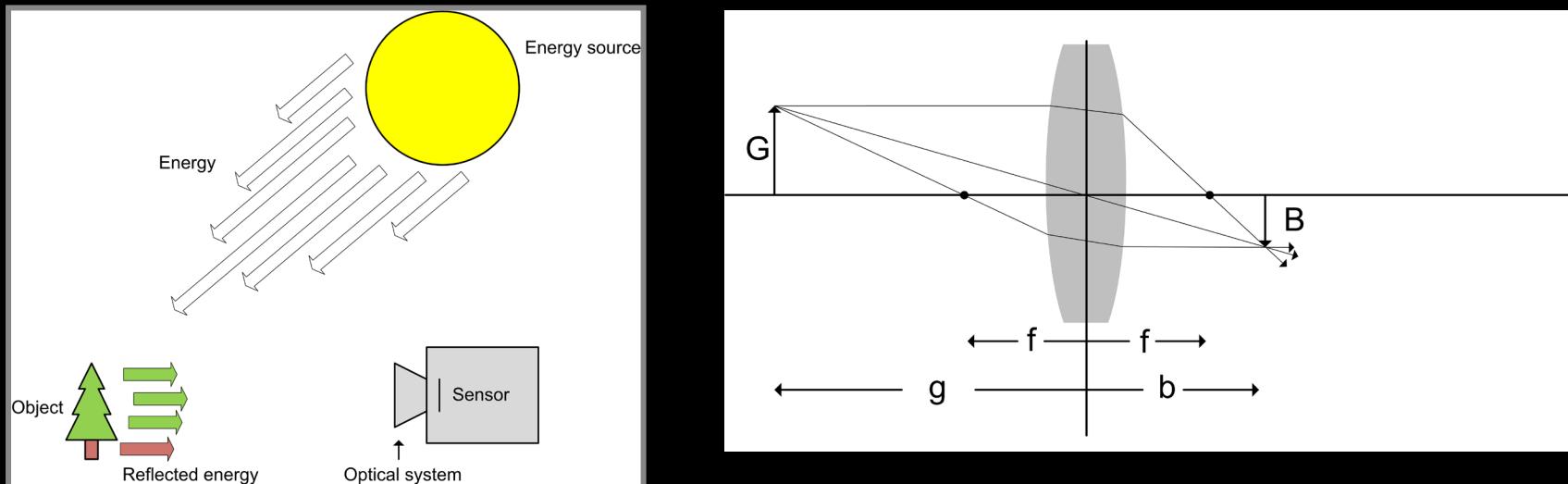
Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)



# PCA Analysis

# Next week:

## Image acquisition, digital cameras, compression and storage and real-time image analysis





# A tutorial on principal component analysis

Rasmus R. Paulsen

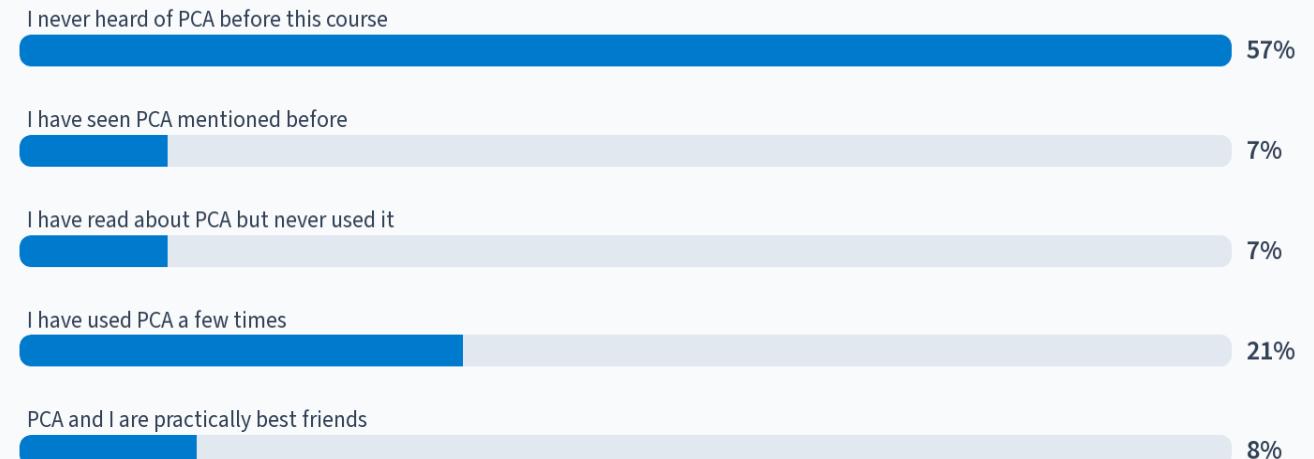
DTU Compute

Based on

Jonathan Shlens: A tutorial on Principal Component Analysis (version 3.02  
– April 7, 2014)

<http://compute.dtu.dk/courses/02502>

### What is your experience with Principal Component Analysis (PCA)



Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)

# Principal Component Analysis (PCA) learning objectives

- Describe the concept of principal component analysis
- Explain why principal component analysis can be beneficial when there is high data redundancy
- Arrange a set of multivariate measurements into a matrix that is suitable for PCA analysis
- Compute the covariance of two sets of measurements
- Compute the covariance matrix from a set of multivariate measurements
- Compute the principal components of a data set using Eigenvector decomposition
- Describe how much of the total variation in the data set that is explained by each principal component

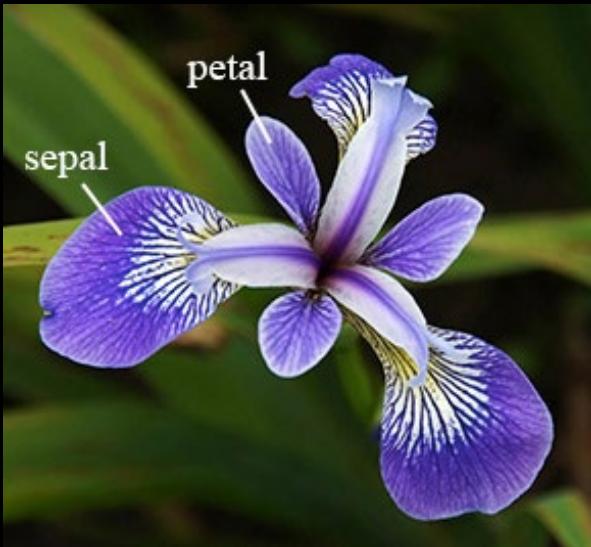
# Iris data

## The Iris flower data

set or Fisher's Iris data set is a data set introduced by Ronald Fisher in his 1936 paper *The use of multiple measurements in taxonomic problems*



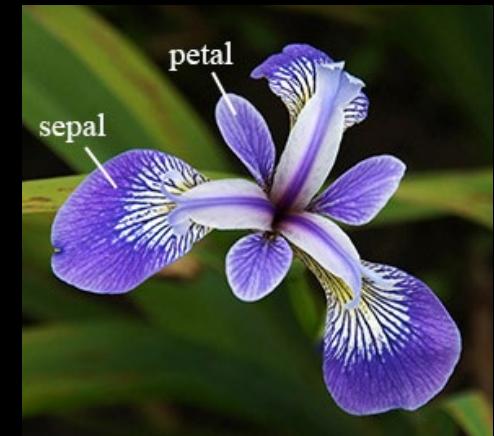
# Iris data



- 3 Iris types
  - 50 flowers of each type
- For each flower
  - Sepal length
  - Sepal width
  - Petal length
  - Petal width
- We use one type as example
  - 50 measured flowers

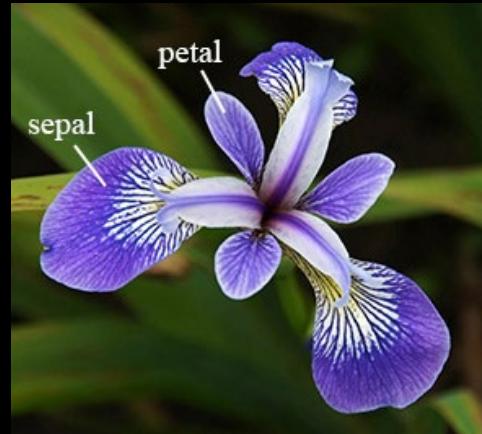
# Iris Data Matrix

- One column is one flower
- One row is all measurements of one type



$$\mathbf{X} = \begin{bmatrix} \text{1} & \text{Sepal length}_1 & \dots & \text{Sepal length}_{50} \\ & \text{Sepal width}_1 & \dots & \text{Sepal width}_{50} \\ & \text{Petal length}_1 & \dots & \text{Petal length}_{50} \\ & \text{Petal width}_1 & \dots & \text{Petal width}_{50} \end{bmatrix}$$

# What can we use these data for?



- The measurements can be used to:
  - Recognize a species of flowers
  - Classify flowers into groups
  - Describe the characteristics of the flower
  - Quantify growth rates
  - ...
- Do we need all the measurements?
  - Can we *boil down* or *combine* some measurements?
- Are some measurements *redundant*?

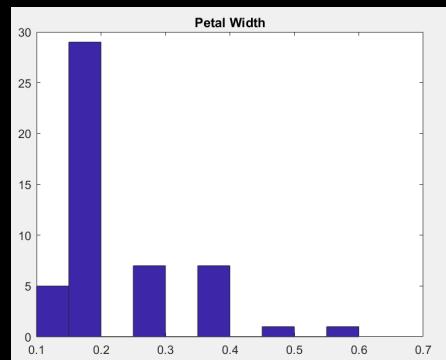
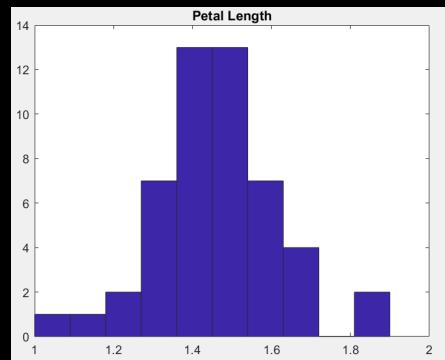
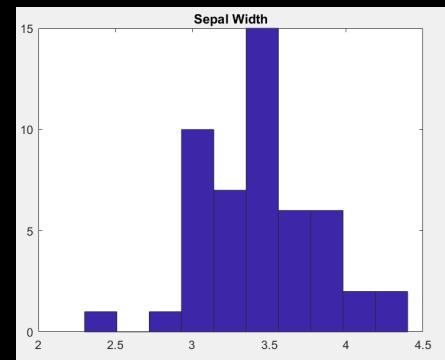
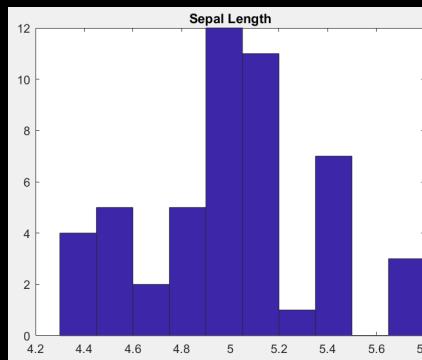
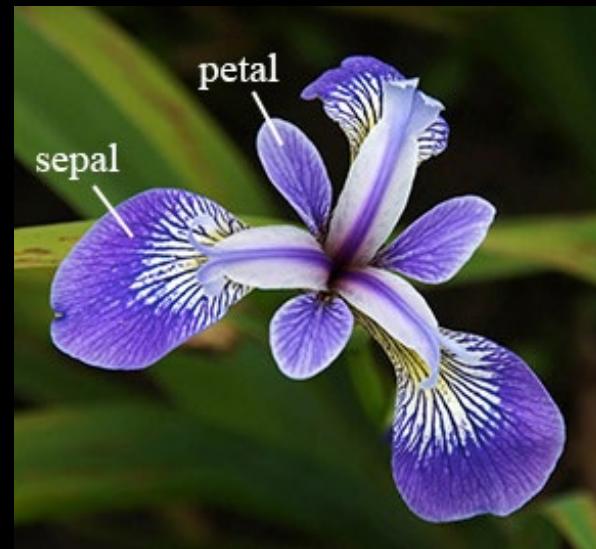
# Variance

$$\sigma_{SL}^2 = 0.1242$$

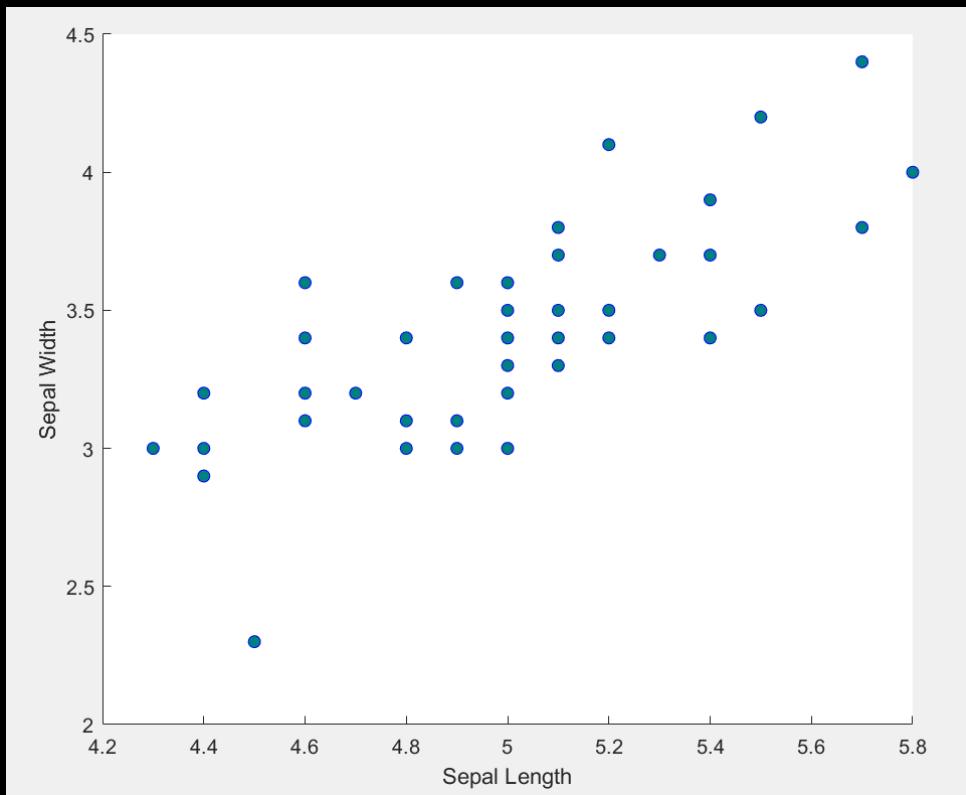
$$\sigma_{SW}^2 = 0.1437$$

$$\sigma_{PL}^2 = 0.0302$$

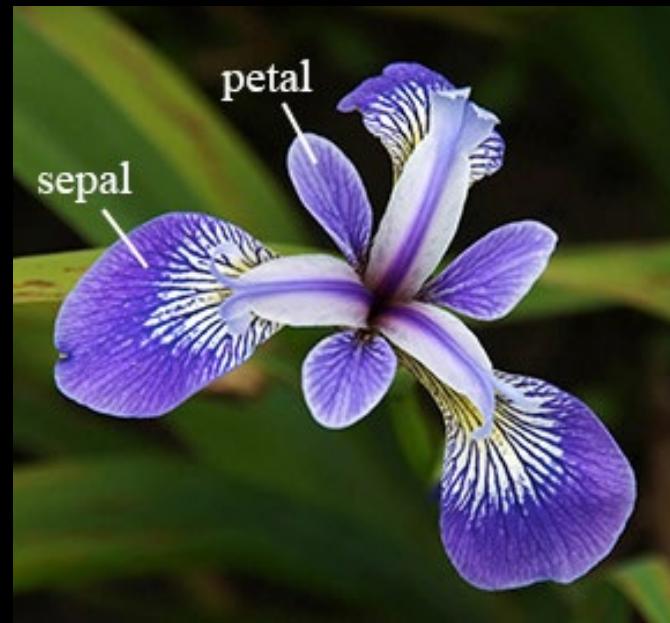
$$\sigma_{PW}^2 = 0.0111$$



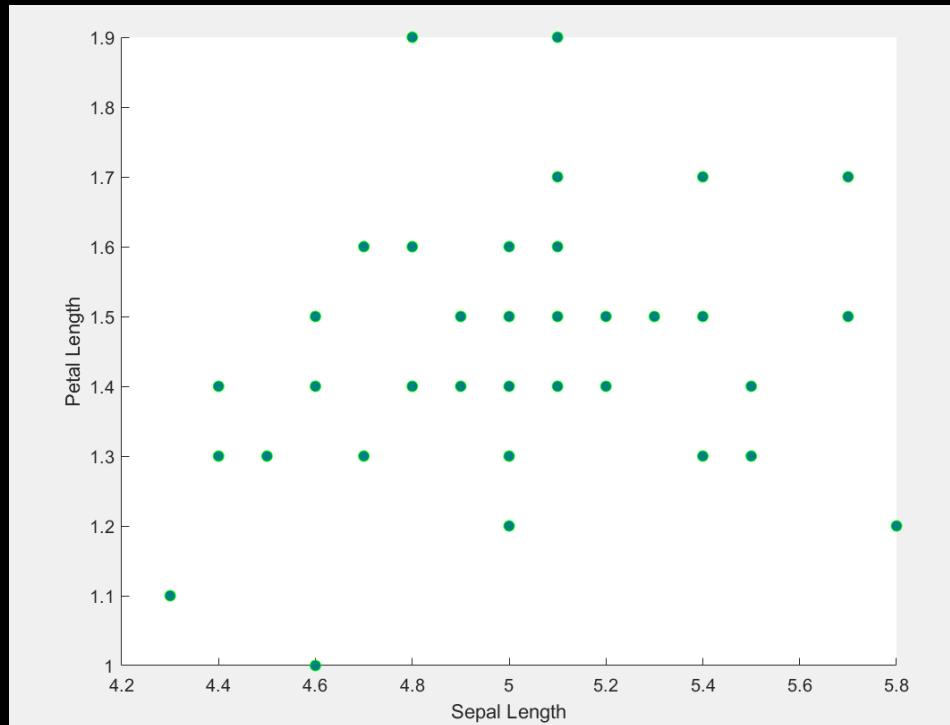
# High Redundancy



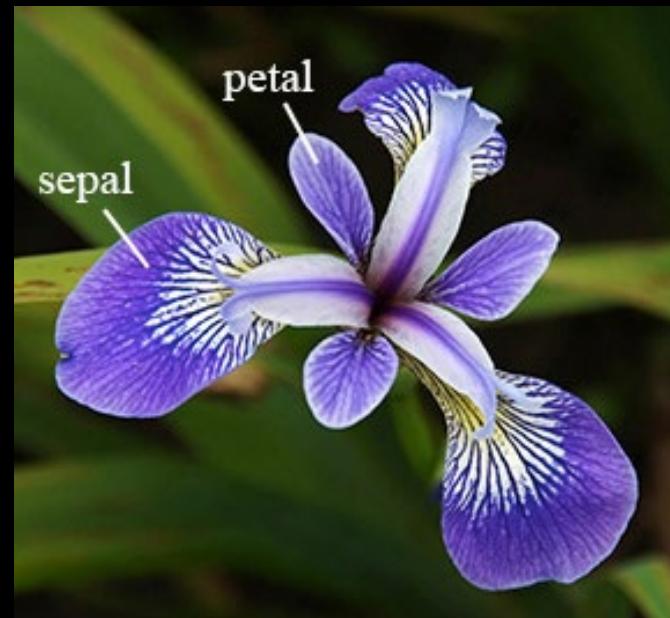
Observation: We can explain quite a lot of the sepal width if we know the sepal lengths



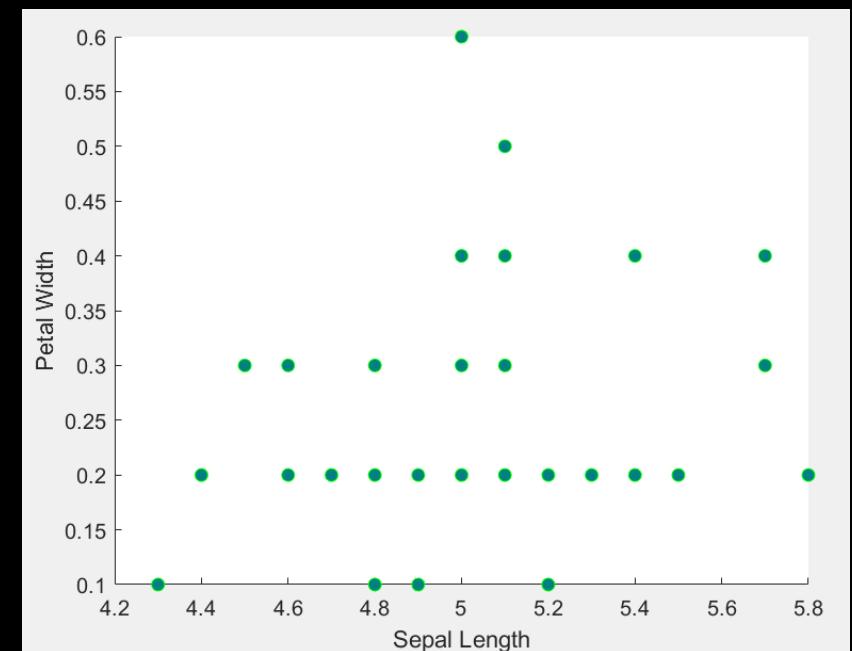
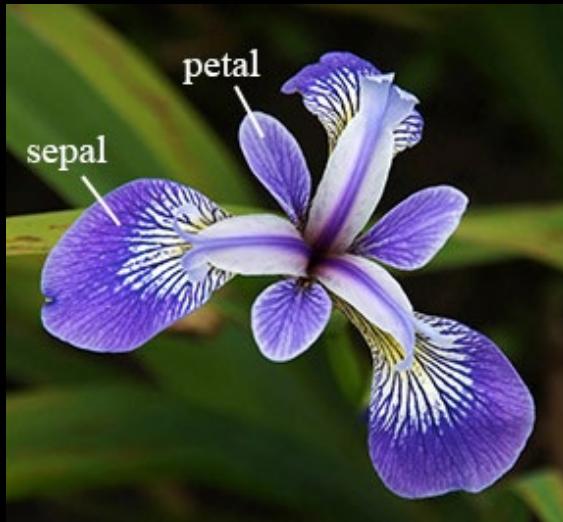
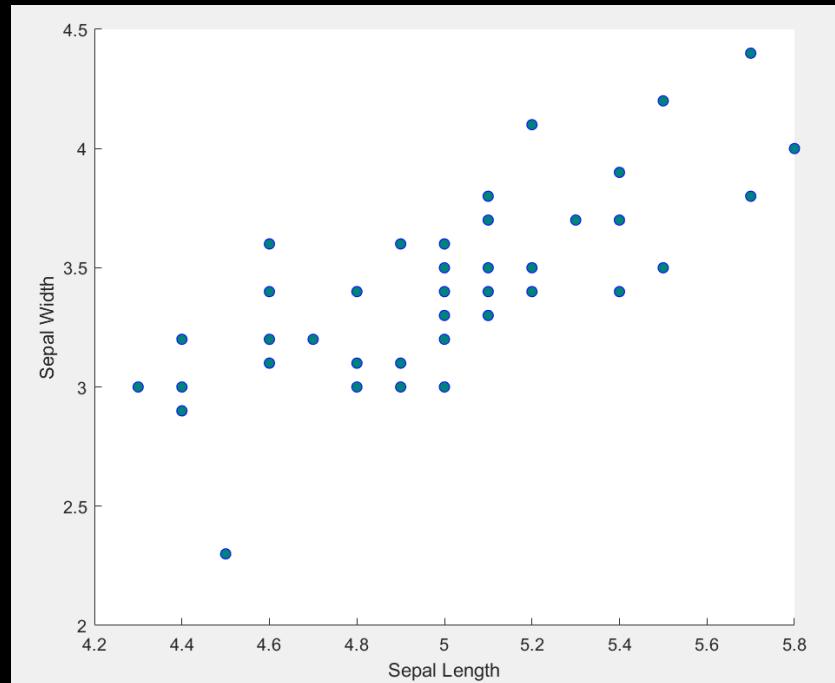
# Low Redundancy



**Observation:** We can NOT explain the petal length if we know the sepal lengths

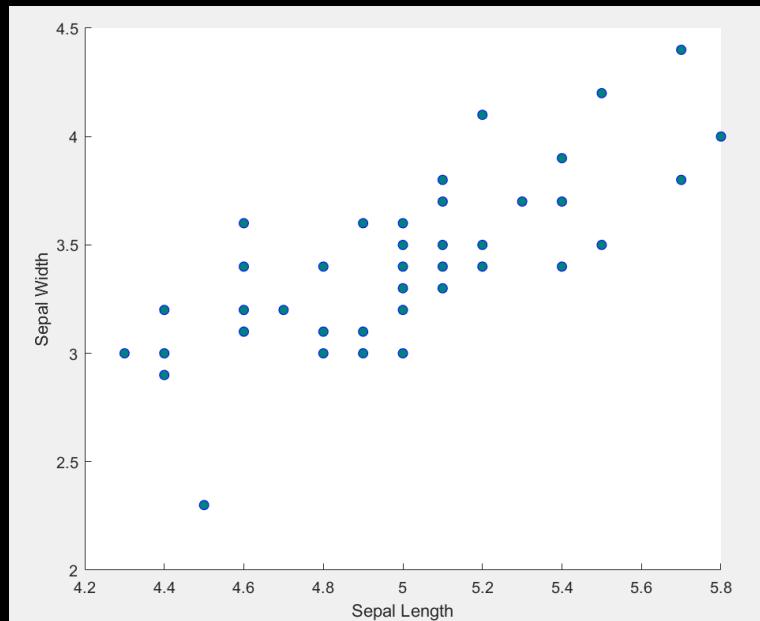


# Covariance



Covariance measures the *relationship* between measurements

# High Covariance



Sepal length and sepal width

$$a_i = \text{SL} = \{5.1, 4.9 \dots, 5\}$$

$$b_i = \text{SW} = \{3.5, 3, \dots, 3.3\}$$

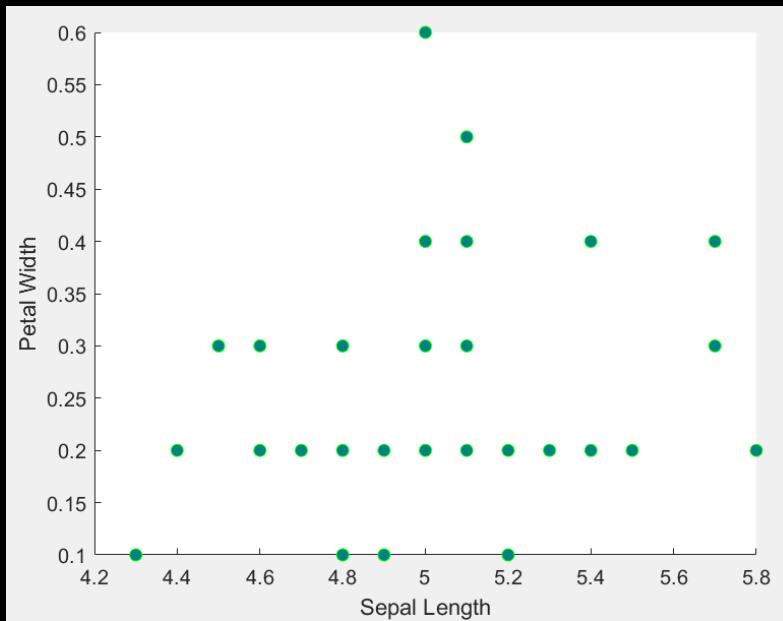
$$\sigma_{\text{SL,SW}}^2 = \frac{1}{n} \sum_i a_i b_i = 17.2578$$



Note that in practice  $n-1$  is used instead of  $n$

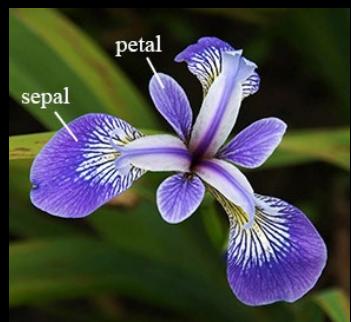


# Low covariance

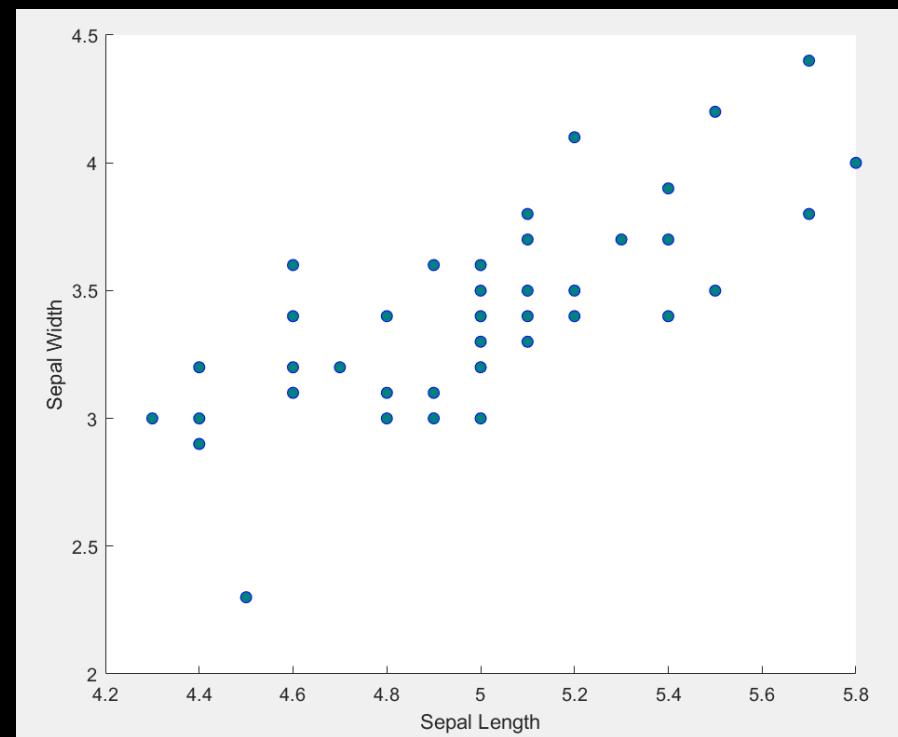


Sepal length and petal width

$$\sigma_{\text{SL}, \text{PW}}^2 = \frac{1}{n} \sum_i a_i b_i = 1.2416$$



# Vector notation for covariance



## Sepal length and sepal width

$$\mathbf{a} = \mathbf{SL} = [5.1, 4.9 \dots, 5]$$

$$\mathbf{b} = SW = [3.5, 3, \dots, 3.3]$$

$$\sigma_{\text{SL,SW}}^2 = \frac{1}{n} \mathbf{a} \mathbf{b}^T$$

# Matrix notation for covariance

$m \times n$  matrix ( $m=4$  and  $n=50$ )

$$\mathbf{X} = \begin{bmatrix} \text{Sepal length}_1 & \dots & \text{Sepal length}_{50} \\ \text{Sepal width}_1 & \dots & \text{Sepal width}_{50} \\ \text{Petal length}_1 & \dots & \text{Petal length}_{50} \\ \text{Petal width}_1 & \dots & \text{Petal width}_{50} \end{bmatrix}$$

$$\mathbf{C}_{\mathbf{X}} \equiv \frac{1}{n} \mathbf{X} \mathbf{X}^T$$

$m \times m$  square matrix  
( $m=4$ )

Note that in practice  $n-1$  is used instead of  $n$

# Covariance matrix autopsy

$$\mathbf{C}_X \equiv \frac{1}{n} \mathbf{X} \mathbf{X}^T$$

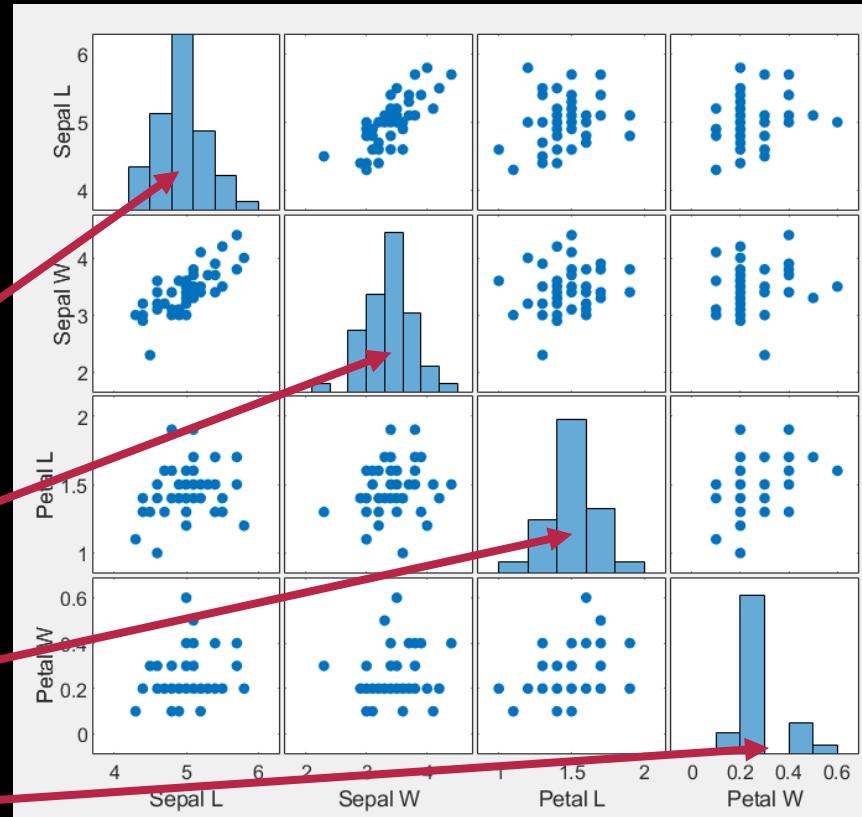
The diagonal elements are the variances

$$\sigma_{SL}^2 = 0.1242$$

$$\sigma_{SW}^2 = 0.1437$$

$$\sigma_{PL}^2 = 0.0302$$

$$\sigma_{PW}^2 = 0.0111$$



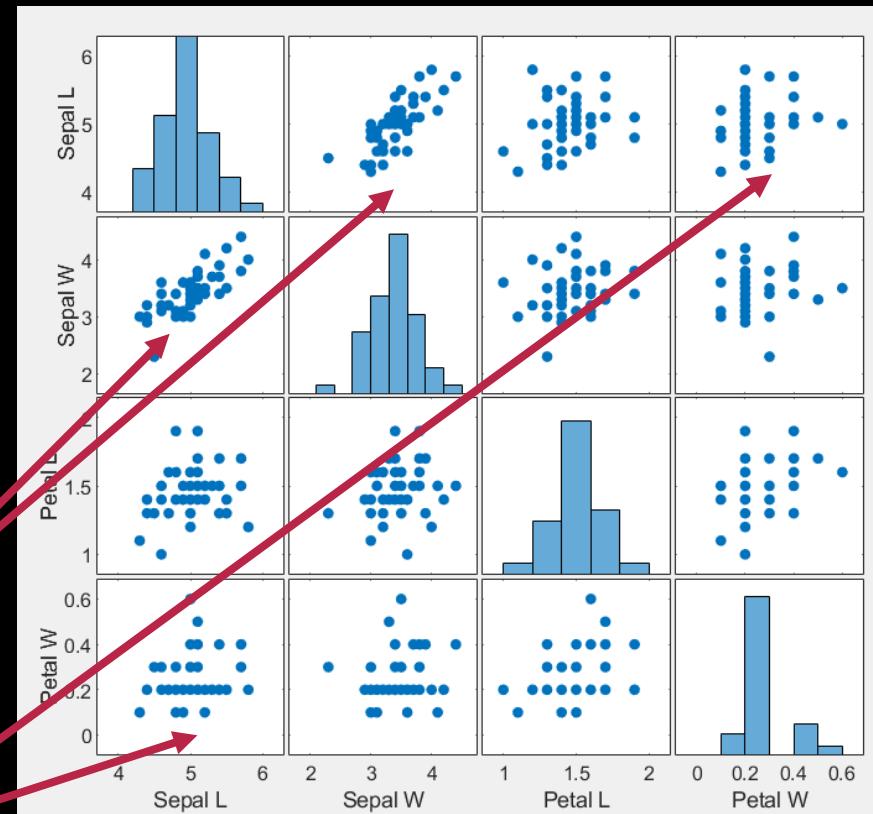
## Covariance matrix autopsy II

$$\mathbf{C}_X \equiv \frac{1}{n} \mathbf{X} \mathbf{X}^T$$

The off-diagonal elements are the covariance

$$\sigma_{SL,SW}^2 = \frac{1}{n} \sum_i a_i b_i = 17.2578$$

$$\sigma_{SL,PW}^2 = \frac{1}{n} \sum_i a_i b_i = 1.2416$$

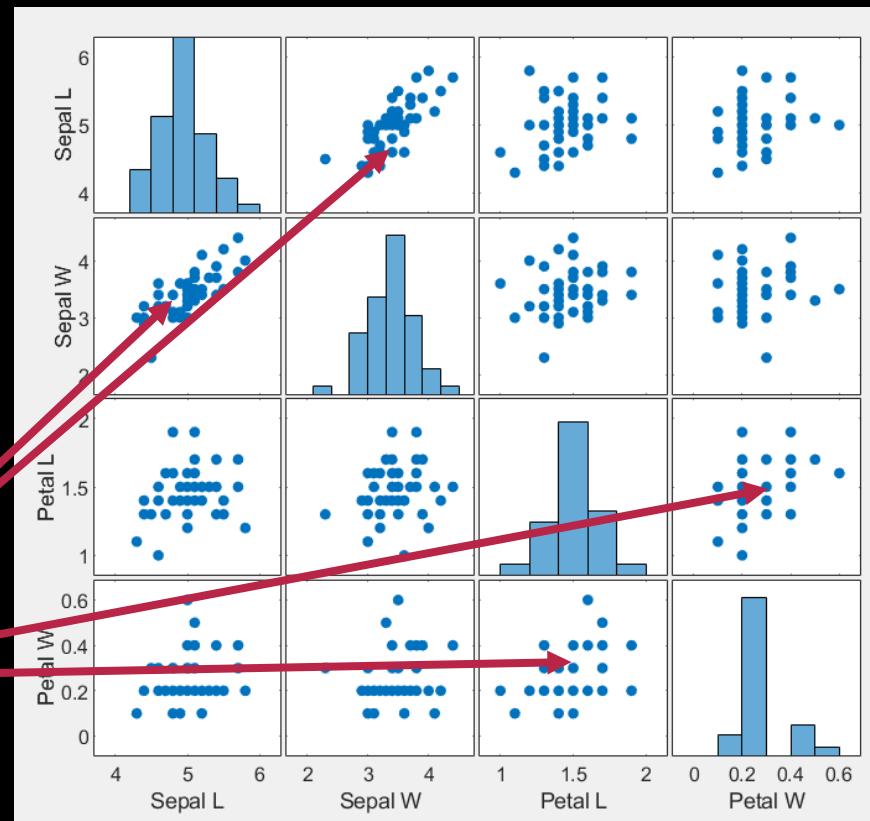


Symmetric!

# Covariance matrix autopsy III

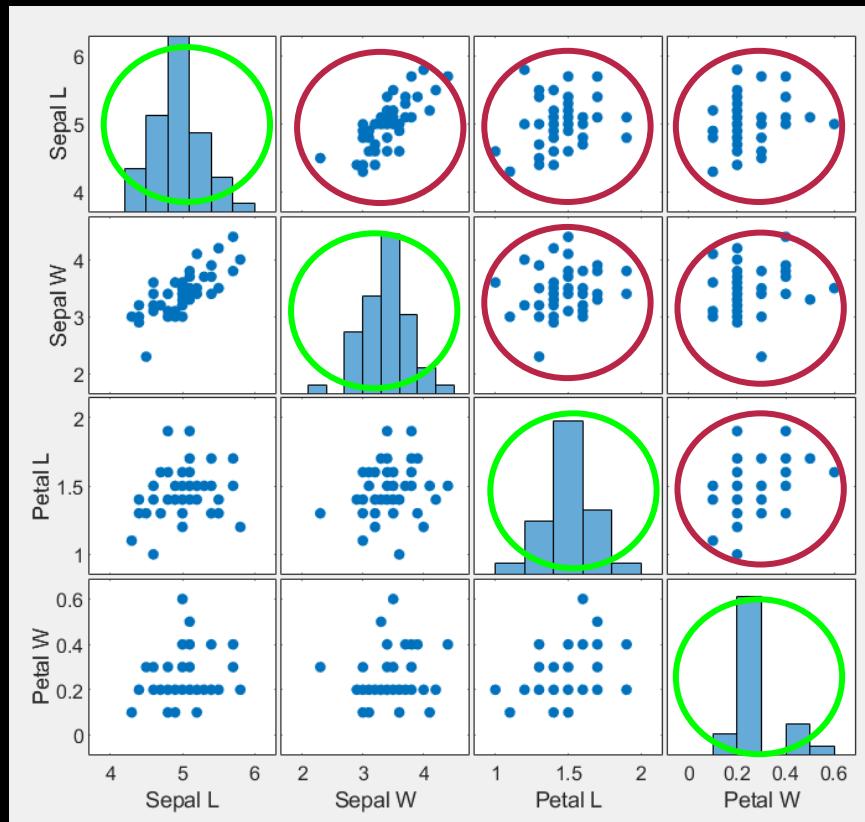
$$\mathbf{C}_X \equiv \frac{1}{n} \mathbf{X} \mathbf{X}^T$$

High redundancy



Symmetric!

# Goals

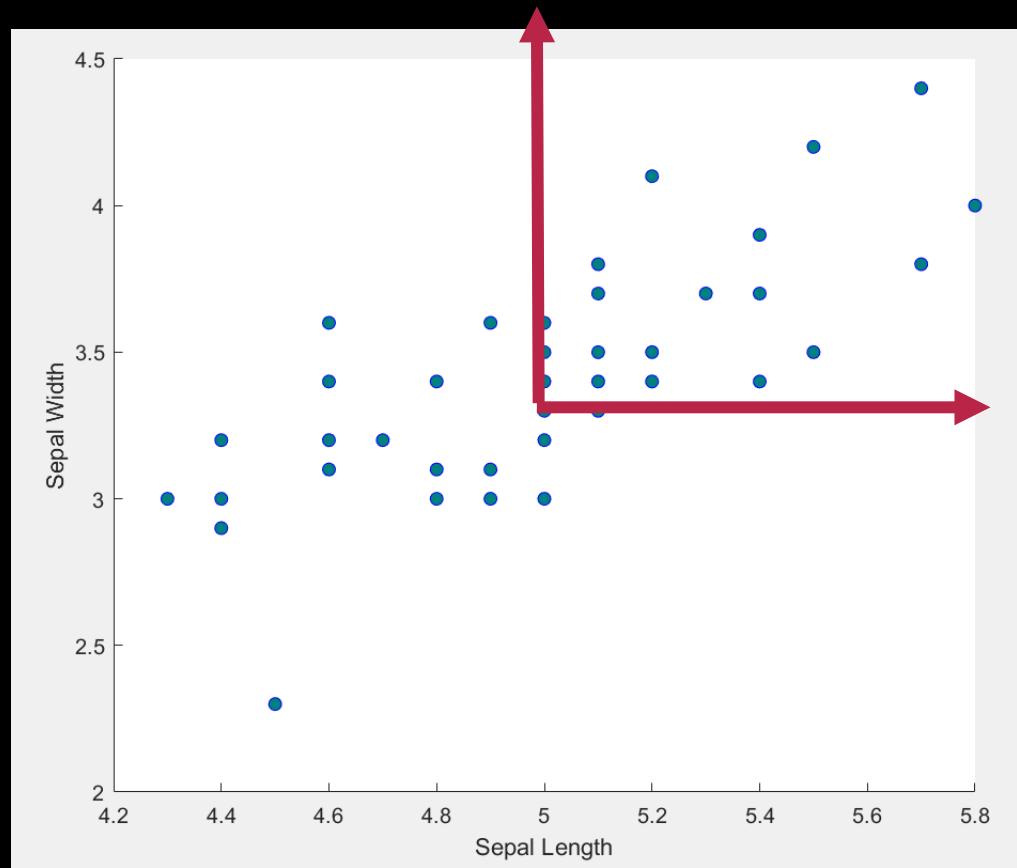


- Minimize redundancy
  - Covariance should be small
- Maximize signal
  - Variance should be large

Signal to noise ratio:

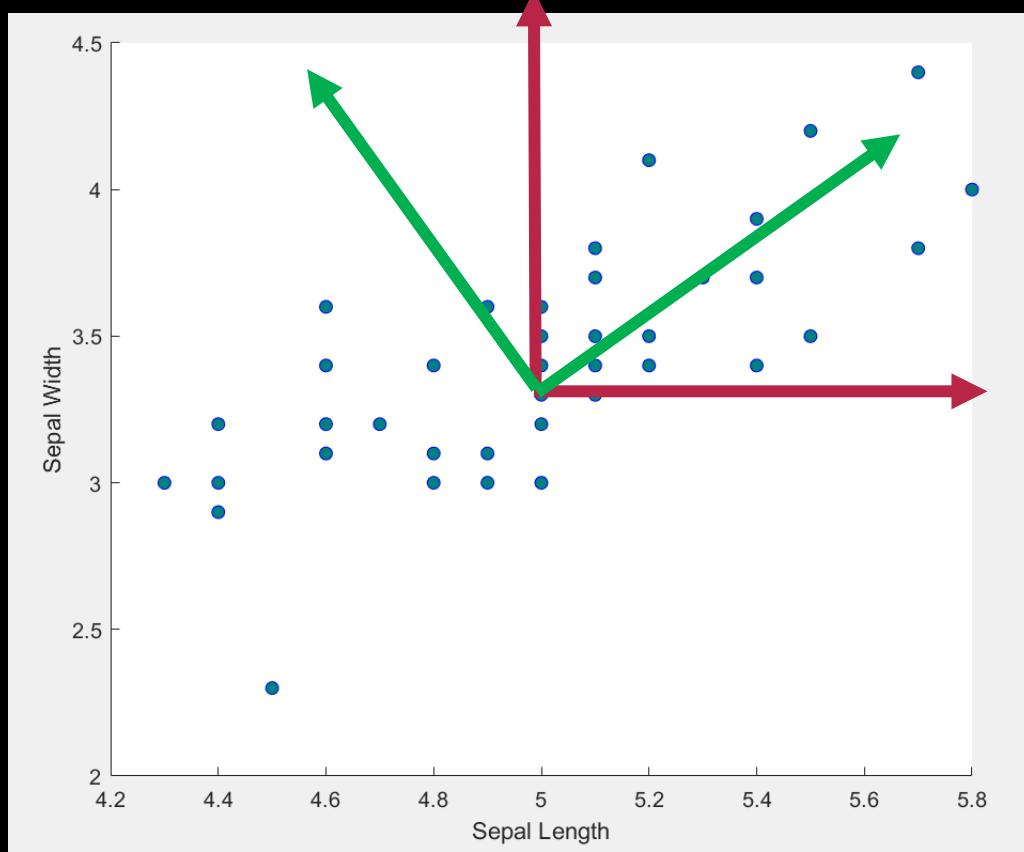
$$\text{SNR} = \frac{\sigma_{\text{signal}}^2}{\sigma_{\text{noise}}^2}$$

# Changing basis

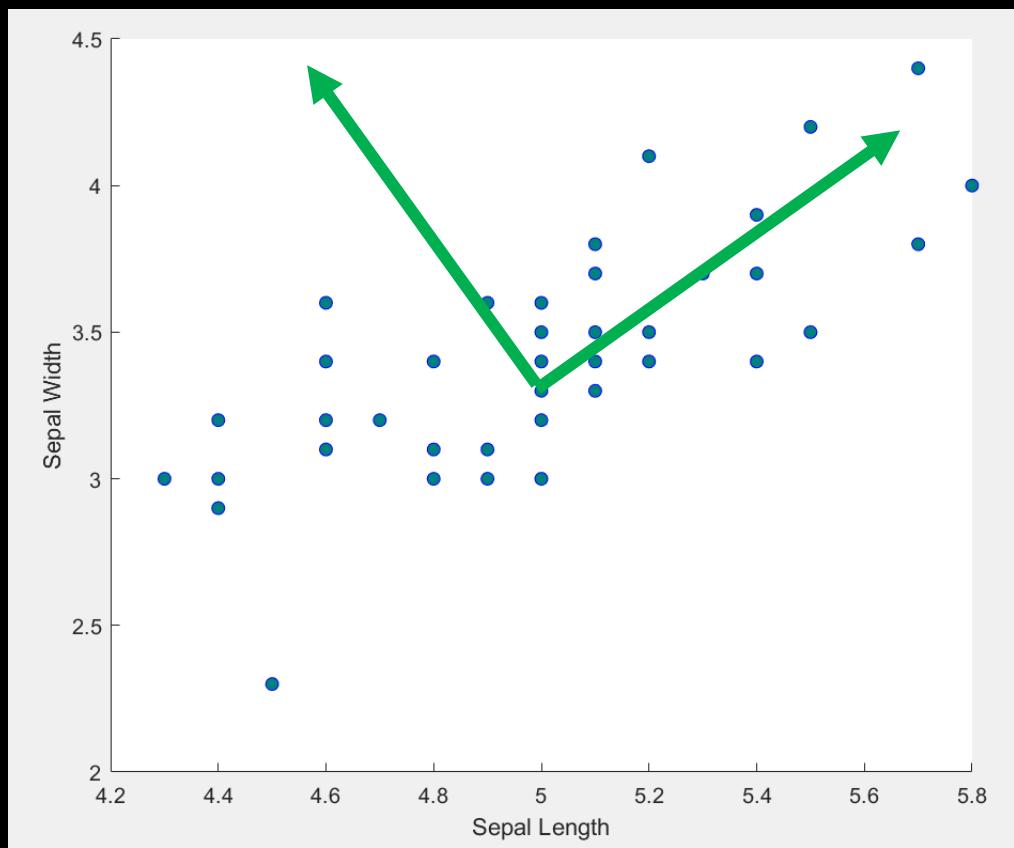


- We start by subtracting the mean
  - Centering data
- Red lines are the default basis

# Changing basis



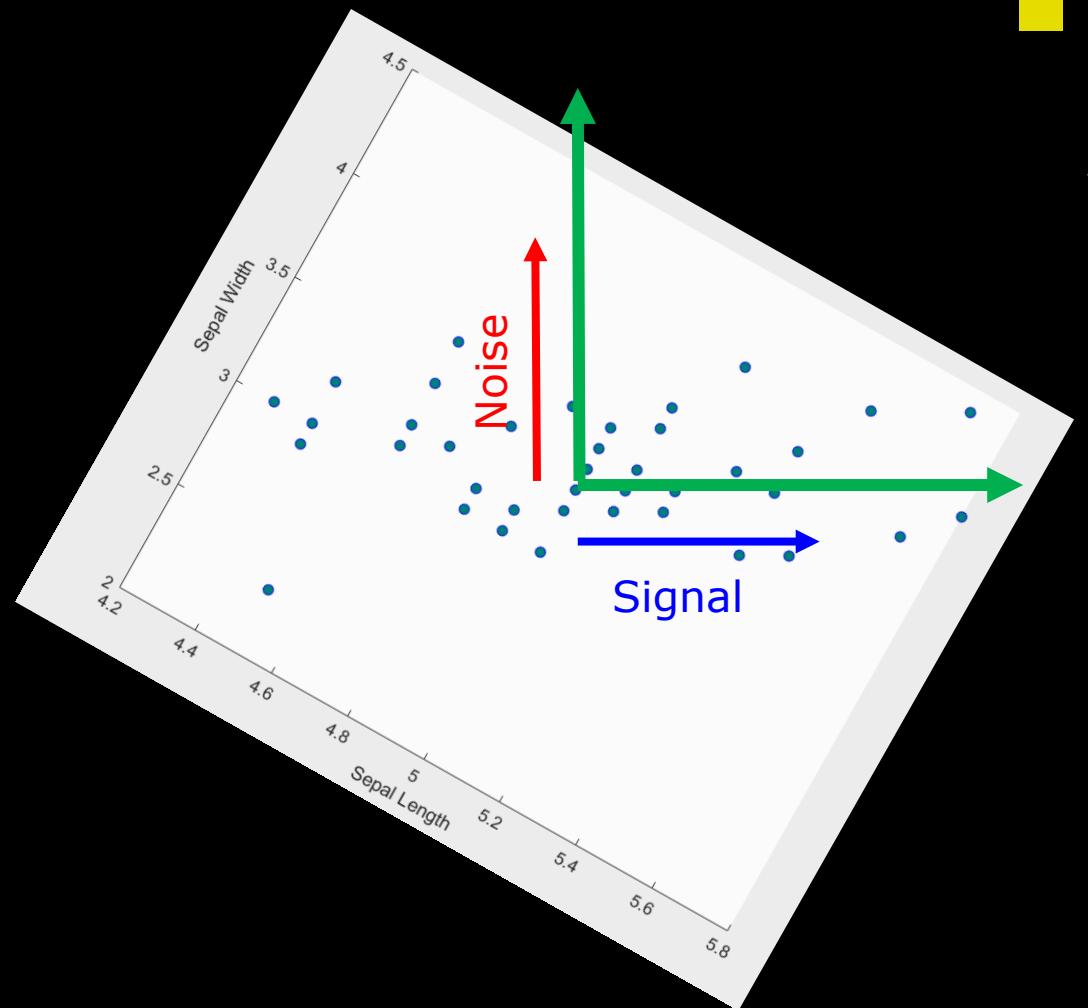
# Changing basis



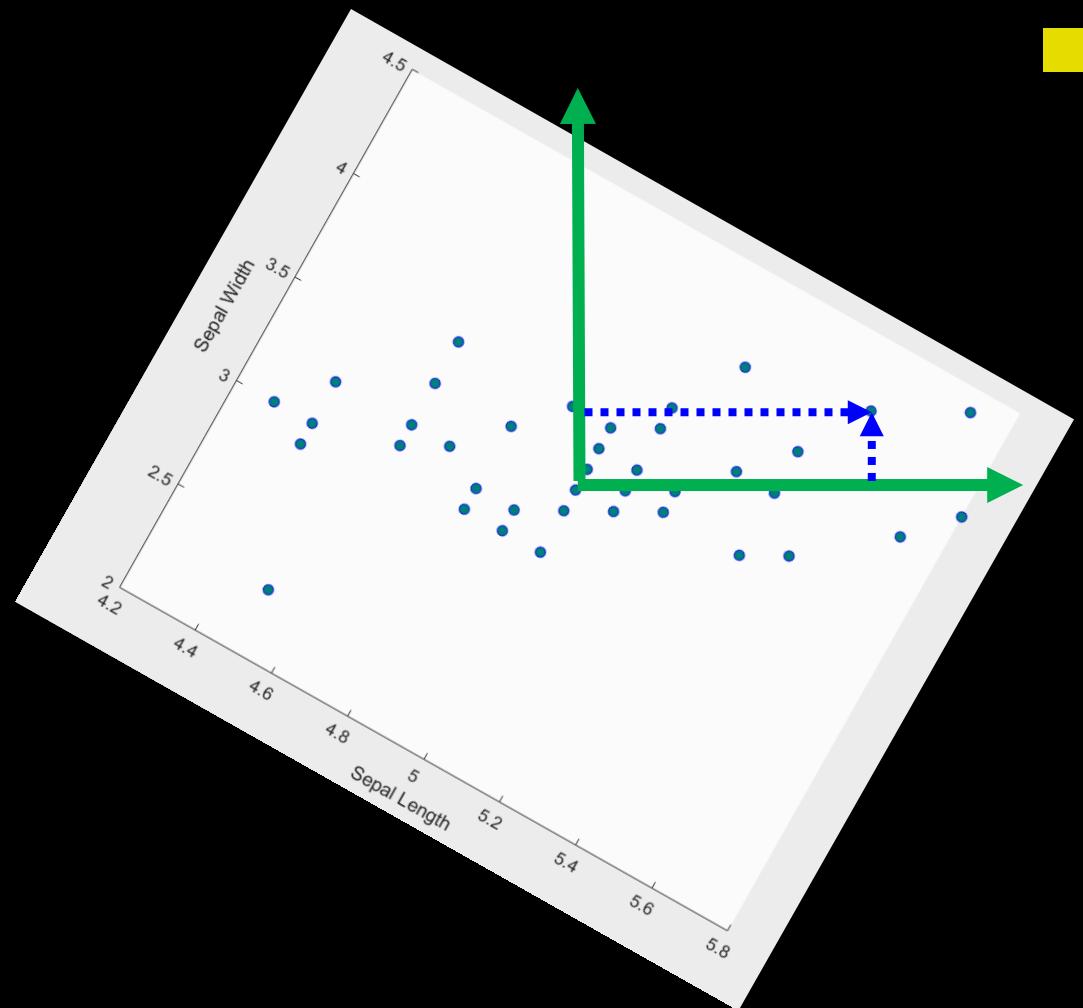
- A new basis that follows the *covariance* in the data

# Changing basis

- Lets try to rotate the data – for visualisation



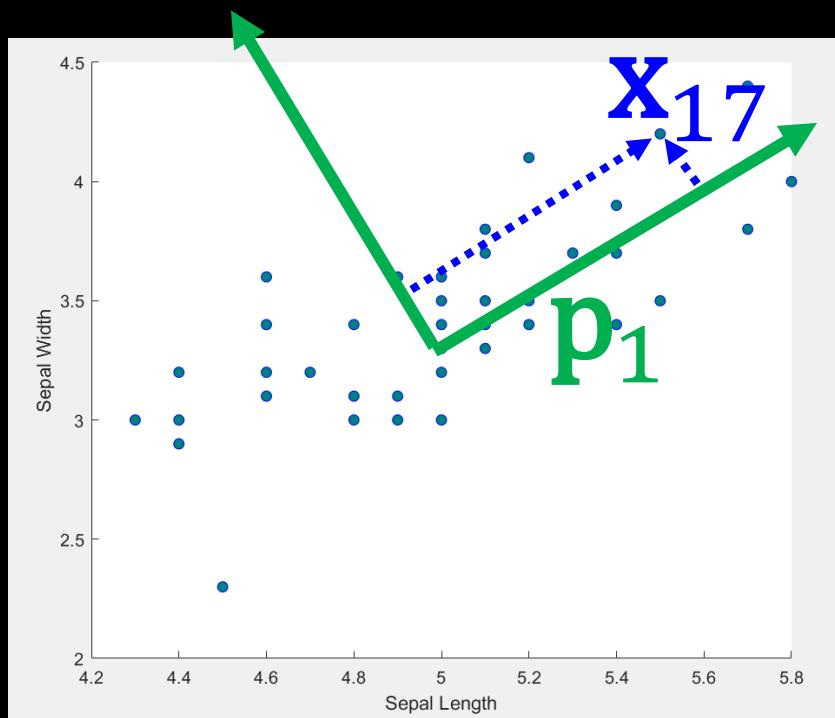
# Changing basis



- Finding the measurement values in the new basis

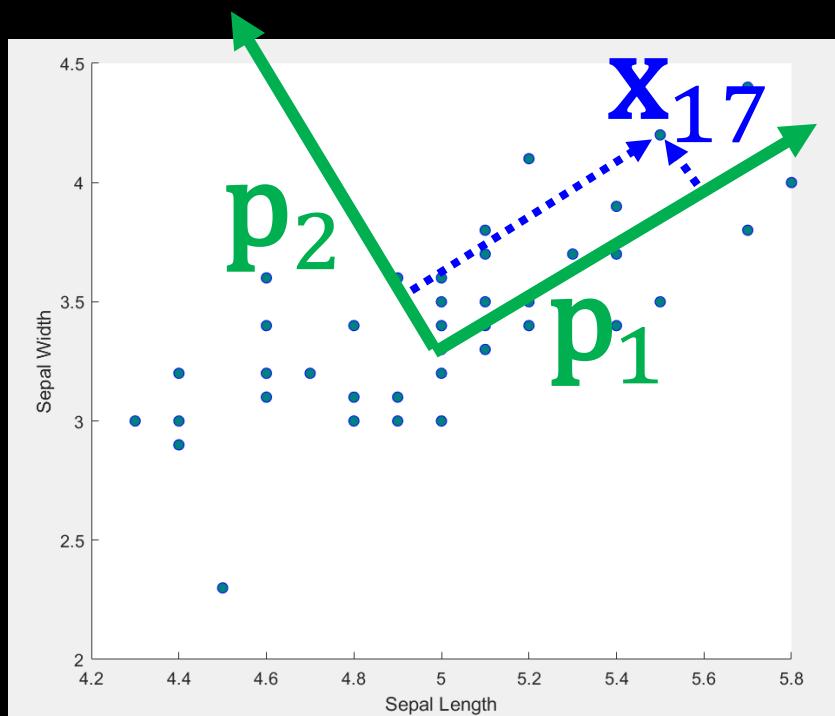
# Changing basis

- The dot product projects a point down to a new axis



$$\mathbf{x}_{17,\text{new}} = x_{17} \cdot p_1$$

# Changing basis



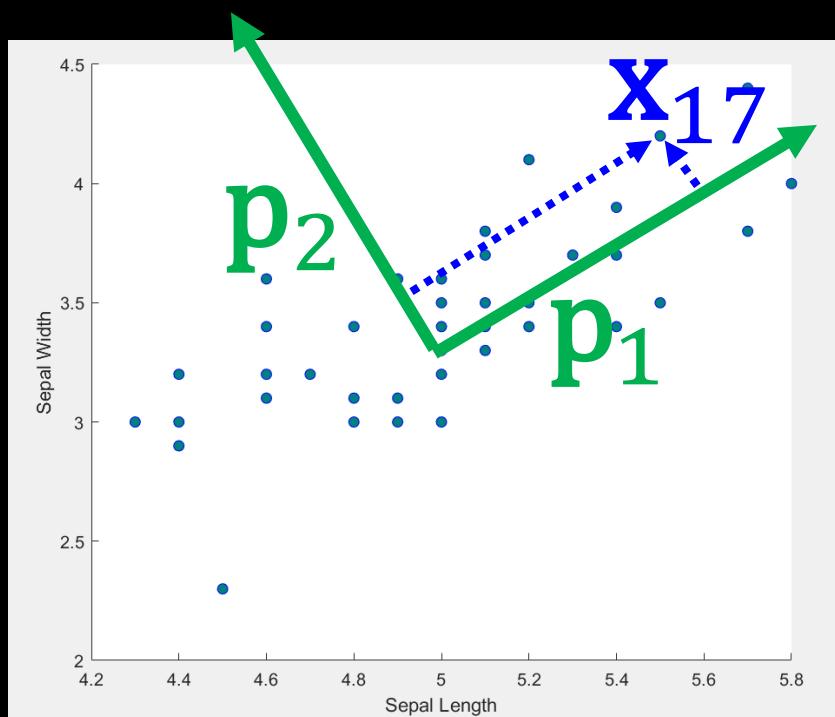
- The dot product projects a point down to a new axis

$$\mathbf{P}\mathbf{X} = \mathbf{Y}$$

- $p_1$  and  $p_2$  are the rows of  $\mathbf{P}$

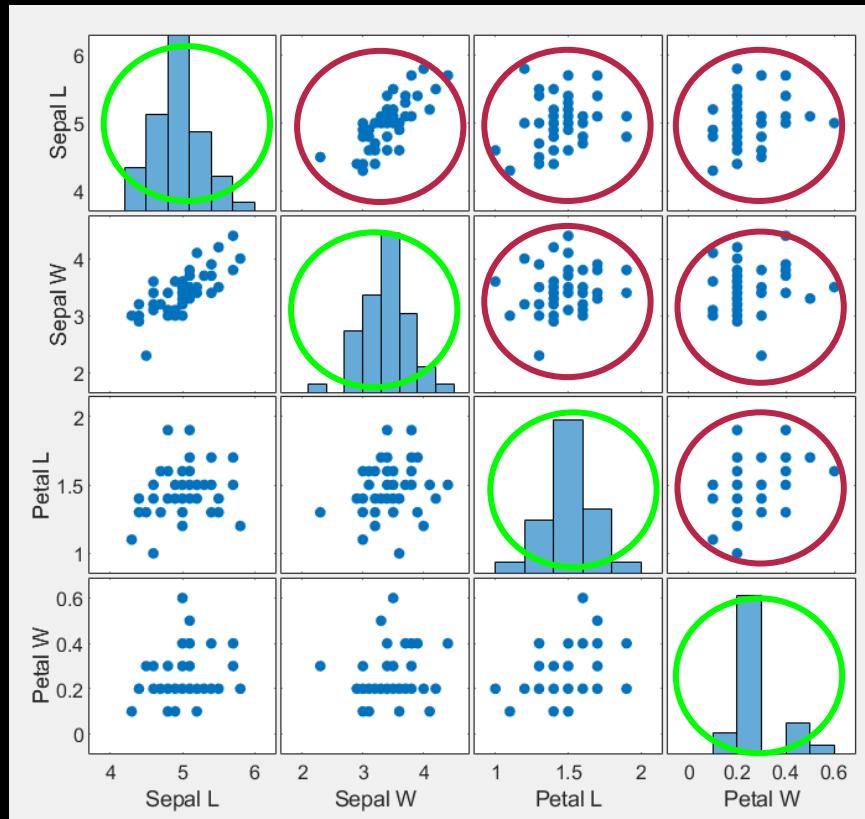
$$\mathbf{X} = \begin{bmatrix} \text{Sepal length}_1 & \dots & \text{Sepal length}_{50} \\ \text{Sepal width}_1 & \dots & \text{Sepal width}_{50} \\ \text{Petal length}_1 & \dots & \text{Petal length}_{50} \\ \text{Petal width}_1 & \dots & \text{Petal width}_{50} \end{bmatrix}$$

# Changing basis



- The dot product projects a point down to a new axis  
 $\mathbf{P}\mathbf{X} = \mathbf{Y}$
- Here  $\mathbf{Y}$  contains the new coordinates/measurements per sample

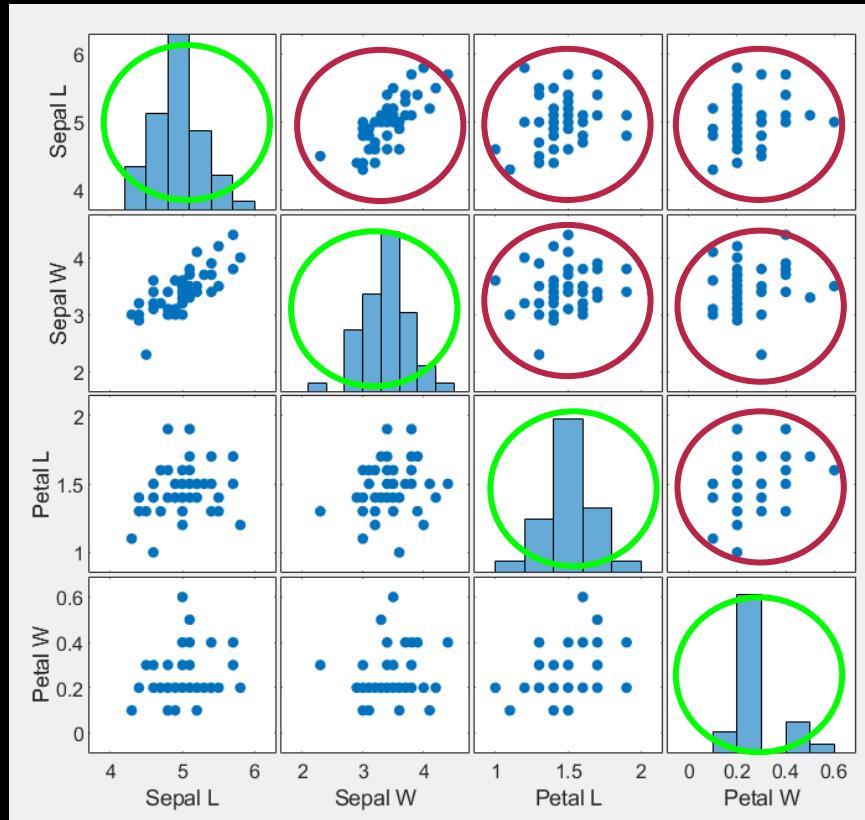
# Goals



- Minimize redundancy
  - Covariance should be small
- Maximize signal
  - Variance should be large
- Transform our data
  - Rotating and scaling the basis
- $\mathbf{Y} = \mathbf{P}\mathbf{X}$
- So it will have

$$\mathbf{C}_\mathbf{Y} \equiv \frac{1}{n} \mathbf{Y} \mathbf{Y}^T$$

# Goals



- The covariance matrix

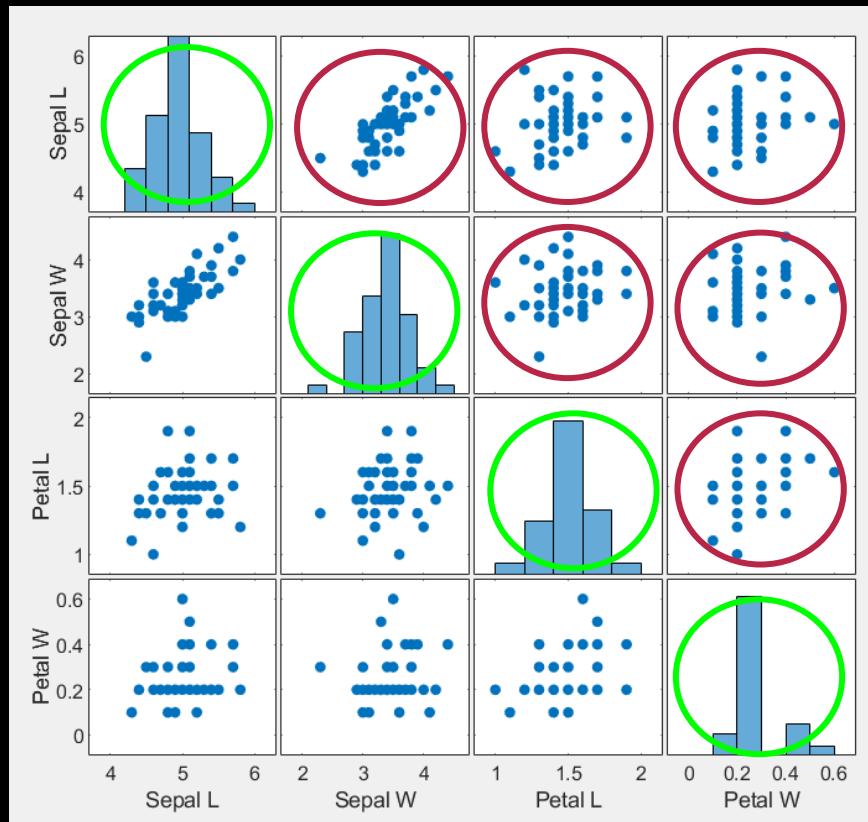
$$\mathbf{C}_Y \equiv \frac{1}{n} \mathbf{Y} \mathbf{Y}^T$$

- Should be *as diagonal as possible*
- We do this by

$$\mathbf{Y} = \mathbf{P}\mathbf{X}$$

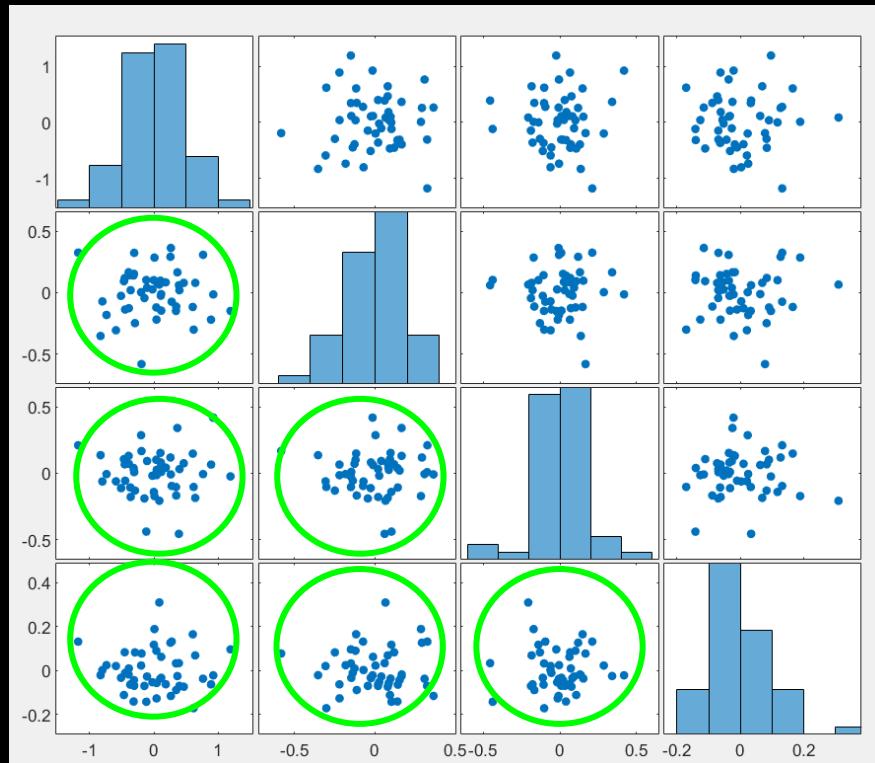
- Where **P** are the principal components

# Computing the principal components



- The Principal Components of  $\mathbf{X}$  are the **eigenvectors** of
$$\mathbf{C}_\mathbf{X} \equiv \frac{1}{n} \mathbf{X} \mathbf{X}^T$$
- The  $i$ 'th diagonal value of  $\mathbf{C}_Y$  is the variance along principal component number  $i$

# New covariance matrix for Iris data



Covariance: 0

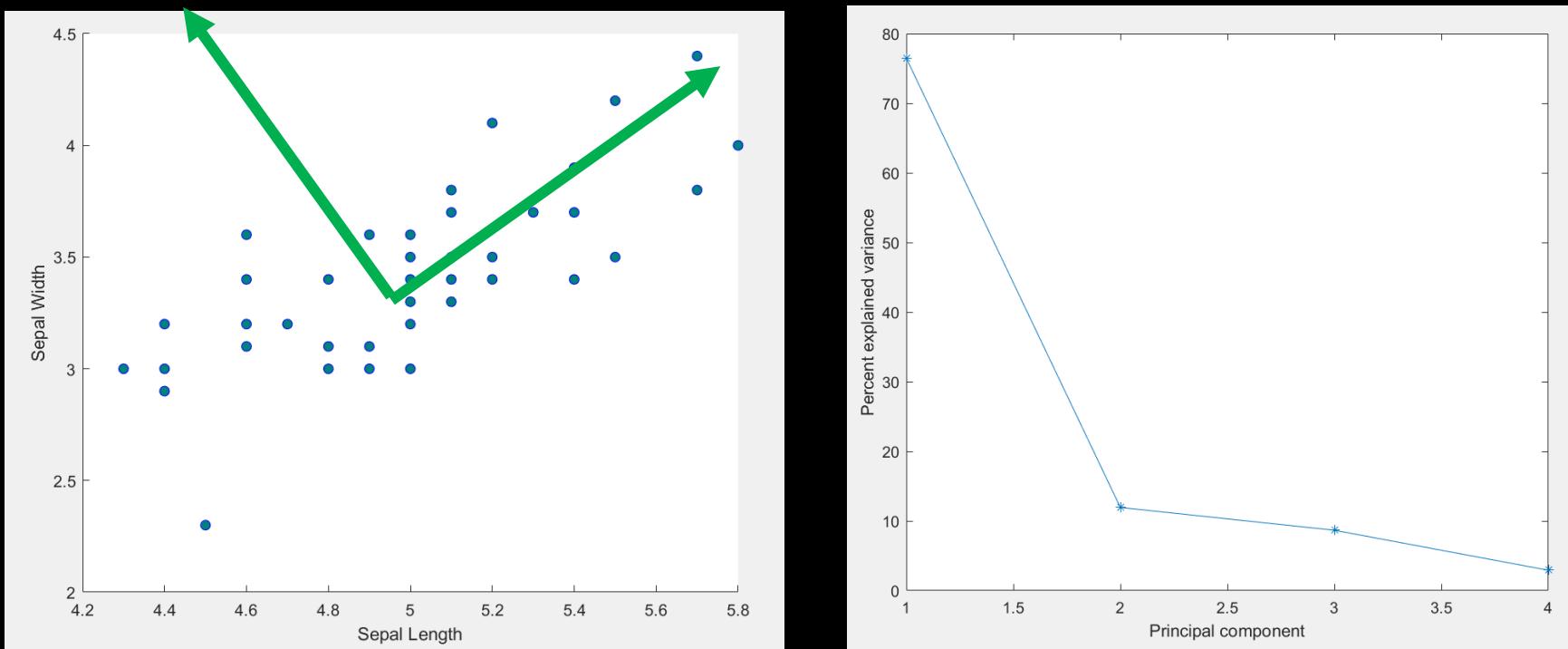
- The principal component are found and

$$\mathbf{Y} = \mathbf{P}\mathbf{X}$$

- With the covariance matrix

$$\mathbf{C}_{\mathbf{Y}} \equiv \frac{1}{n} \mathbf{Y} \mathbf{Y}^T$$

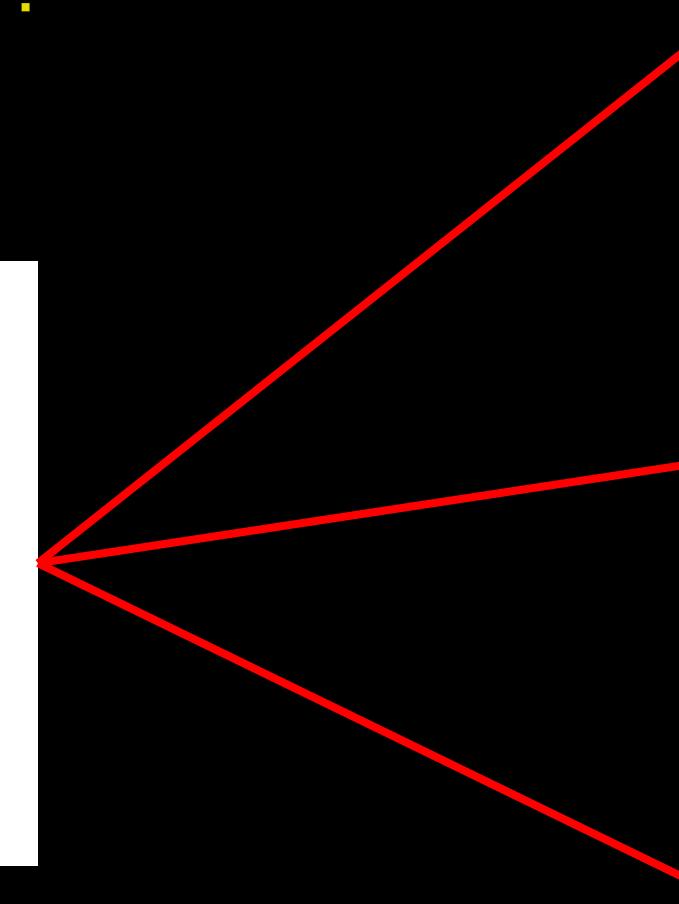
# Explained variance



One component explains 75% of the total variation – so for each flower we can have one number that explains 75% percent of the 4 measurements!

# What can we use it for?

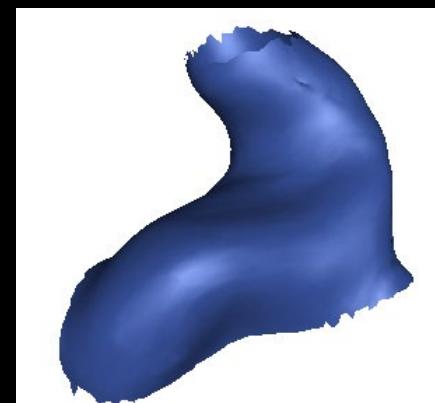
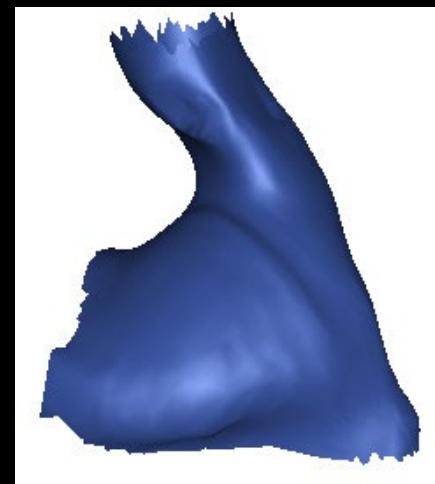
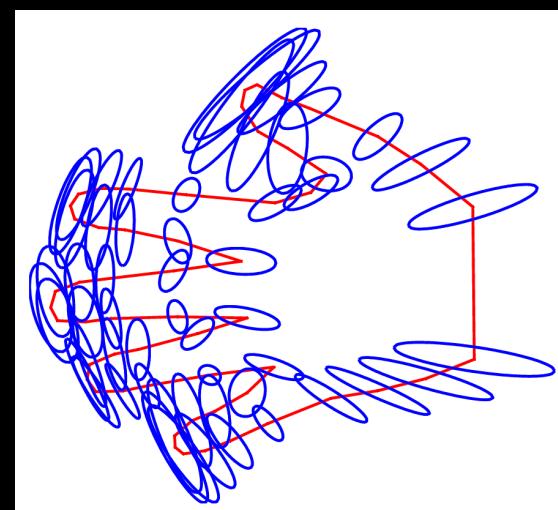
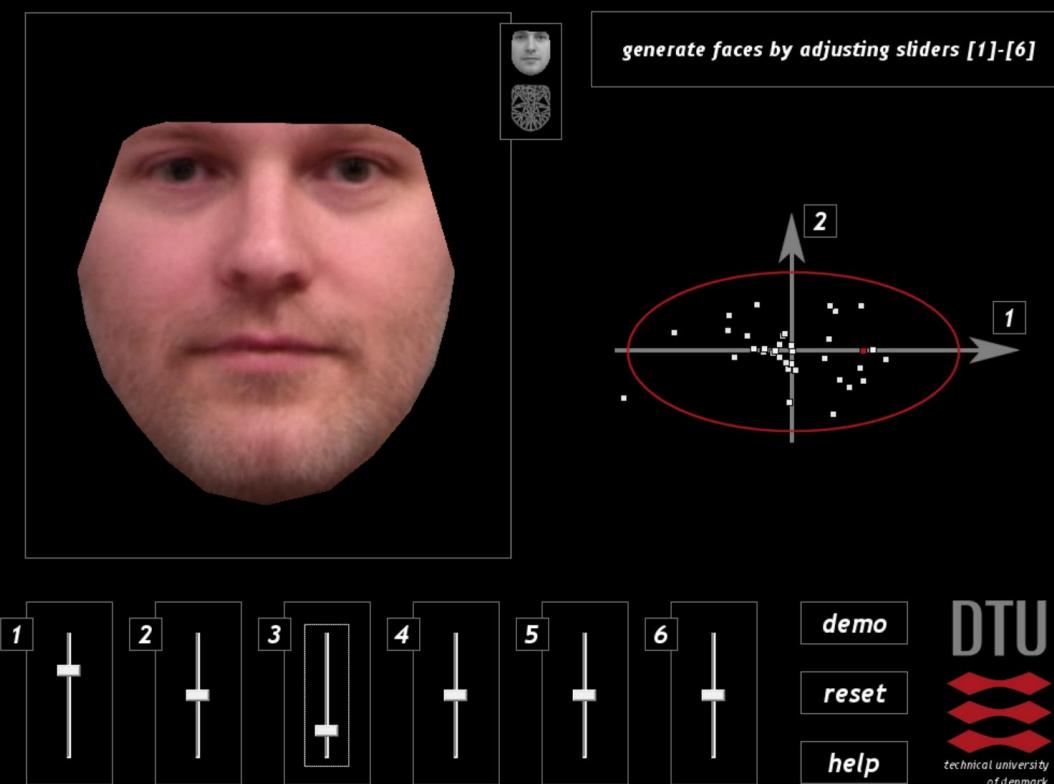
## ■ Classification



Based on one value instead of 4

# What can we use it for?

- Many more examples in the course



## Final note – practical estimation of covariance matrix

$$\mathbf{C}_\mathbf{X} \equiv \frac{1}{n} \mathbf{XX}^T$$

In practice  $n-1$  is used instead of  $n$  for exercises and in the exam.

$$\mathbf{C}_\mathbf{X} \equiv \frac{1}{n-1} \mathbf{XX}^T$$



# Image Analysis

Lecture 2: Image acquisition, compression, storage and change detection  
in videos

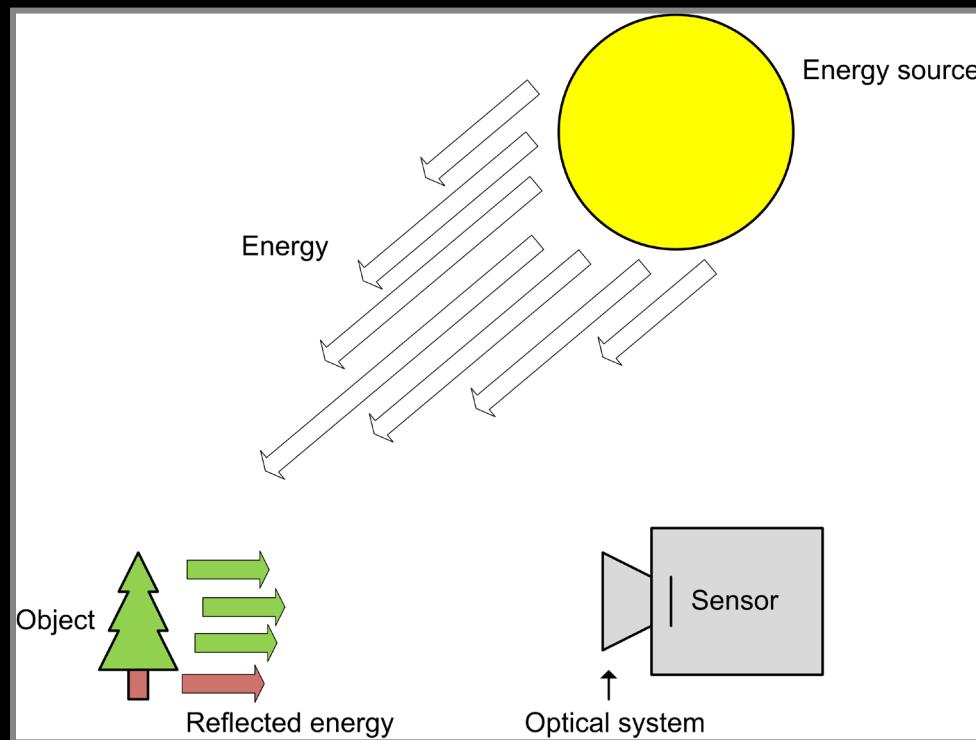
Rasmus R. Paulsen  
Tim B. Dyrby

DTU Compute

<http://compute.dtu.dk/courses/02502>

# Lecture 2

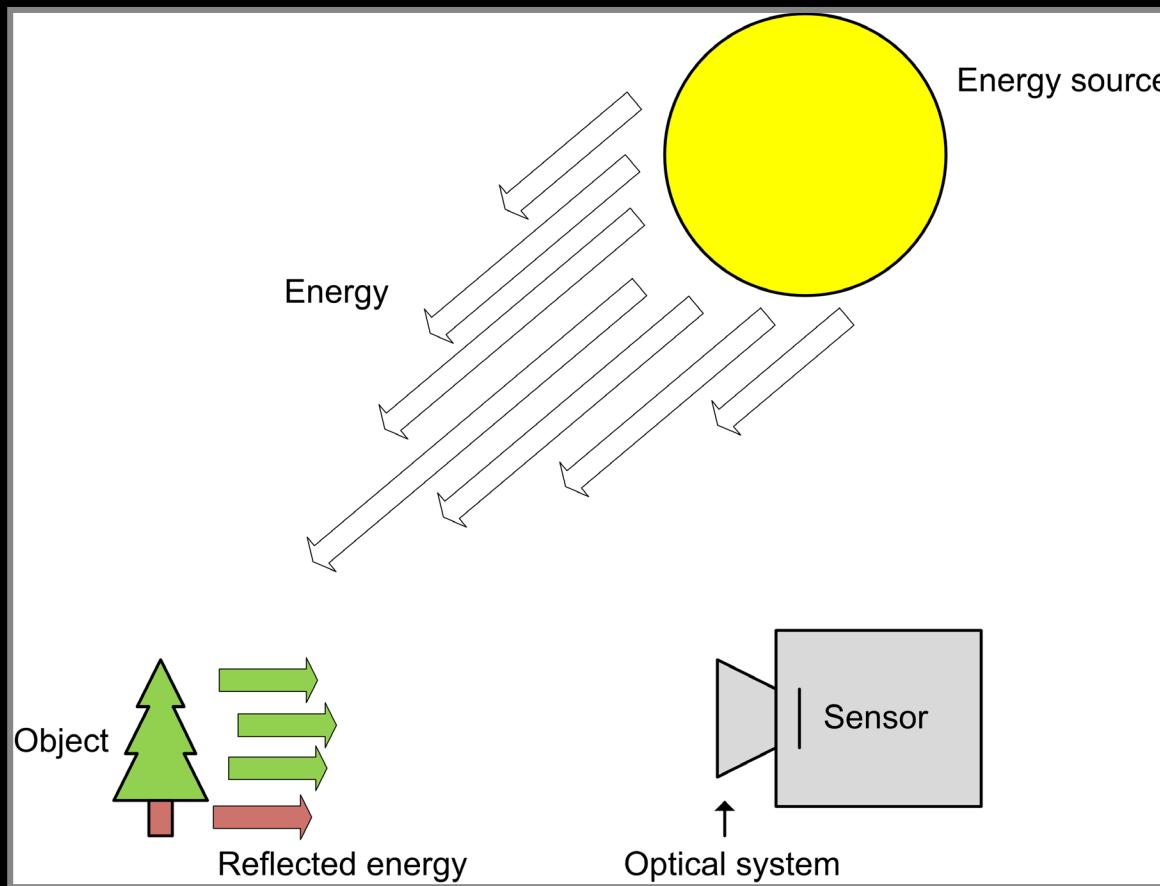
## ■ Image acquisition, compression, storage and change detection in videos



# Learning objectives – cameras and lenses

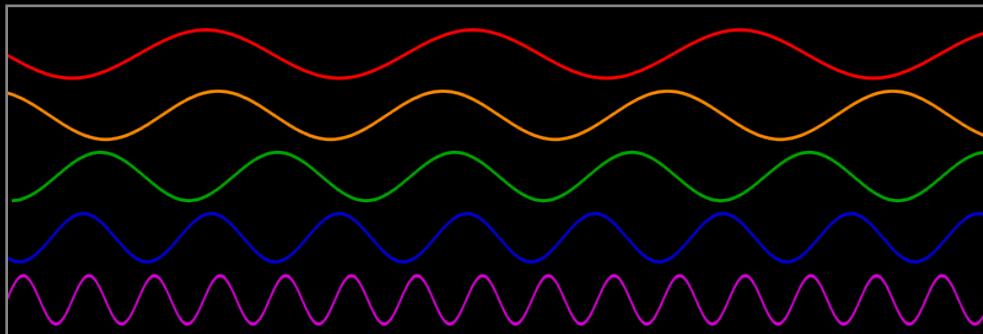
- Explain where visible light is in the electromagnetic spectrum
- Describe the pin hole camera
- Describe the properties of a thin-lens including focal-length, the optical center, and the focal point
- Estimate the focal length of a thin lens
- Compute the optimal placement of a CCD chip using the thin lens equation
- Describe depth-of-field
- Compute the field-of-view of a camera
- Explain the simple CCD model

# How is an image created?

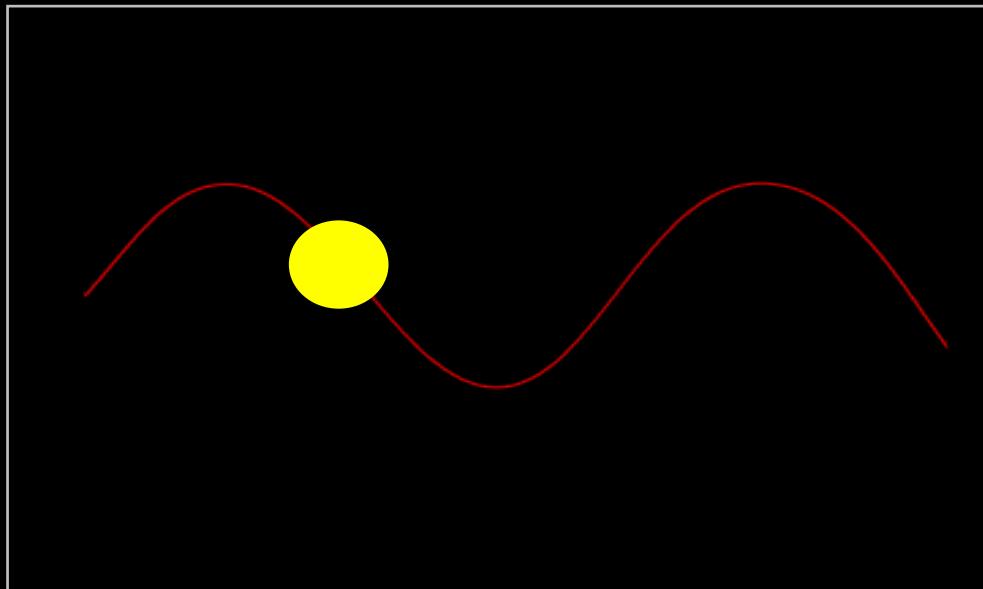


This is just one way! Other methods will be described later in the course.

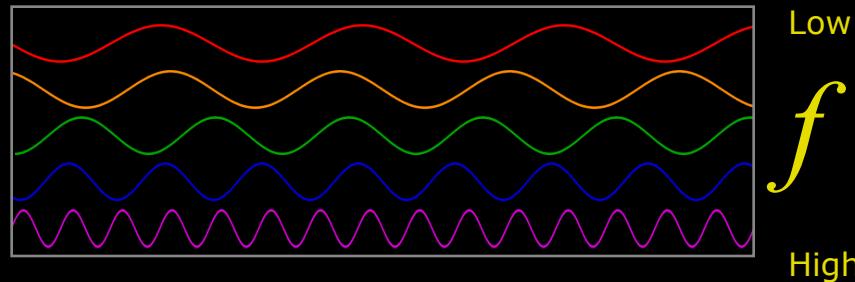
# What is light?



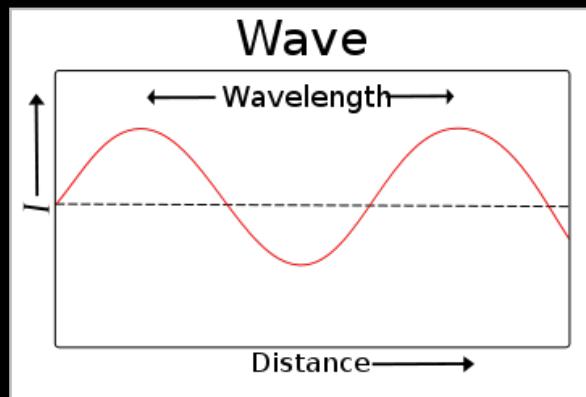
- Can be seen as electromagnetic waves
- Or as a photon (from Greek *phōtos*, "light")
  - Mass less fundamental particle



# Light as a wave



$$\lambda = \frac{c}{f}$$

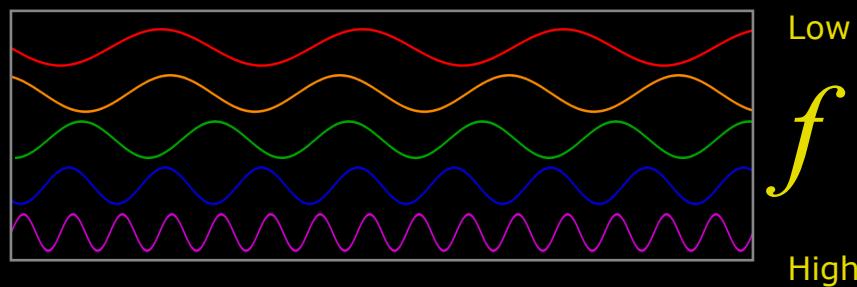


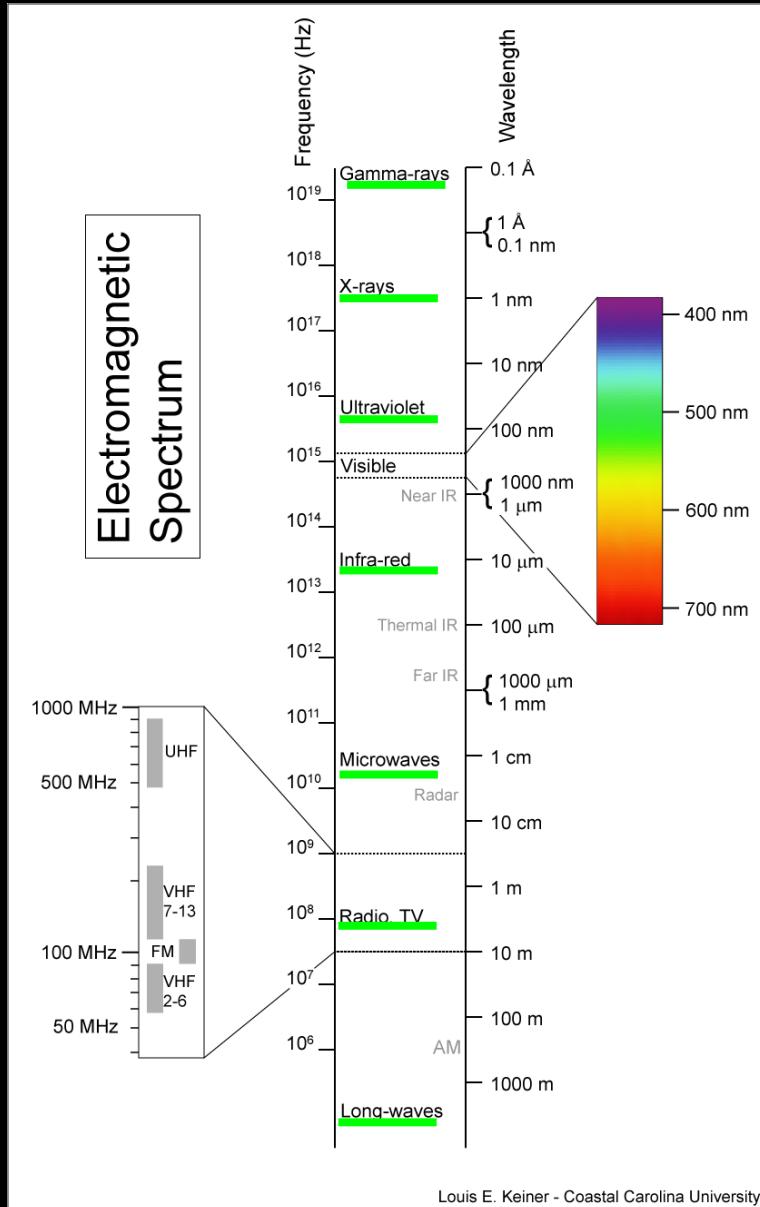
- It has a frequency  $f$ 
  - Measured in Hertz [Hz]
- It has a wavelength  $\lambda$  (lambda)
  - Measured in meters [m]
- It has a speed
  - "The speed of light"  $c$
  - 299.792 458 [m/s]
- High frequency -> short waves
- Low frequency -> long waves

# Energy of light

$$E = h \cdot f$$

- Light has energy
  - You can feel it in the sun!
- Planck's constant  $h$
- High frequency -> high energy
- Long waves -> low energy





## ■ Electromagnetic spectrum

- Range of all frequencies
- Divided into 7 regions

## ■ Wavelengths

- $1 \mu\text{m} = 1 \text{ micrometer} = 0.001 \text{ mm}$
- $1 \text{ nm} = 1 \text{ nanometer} = 0.0000001 \text{ mm}$

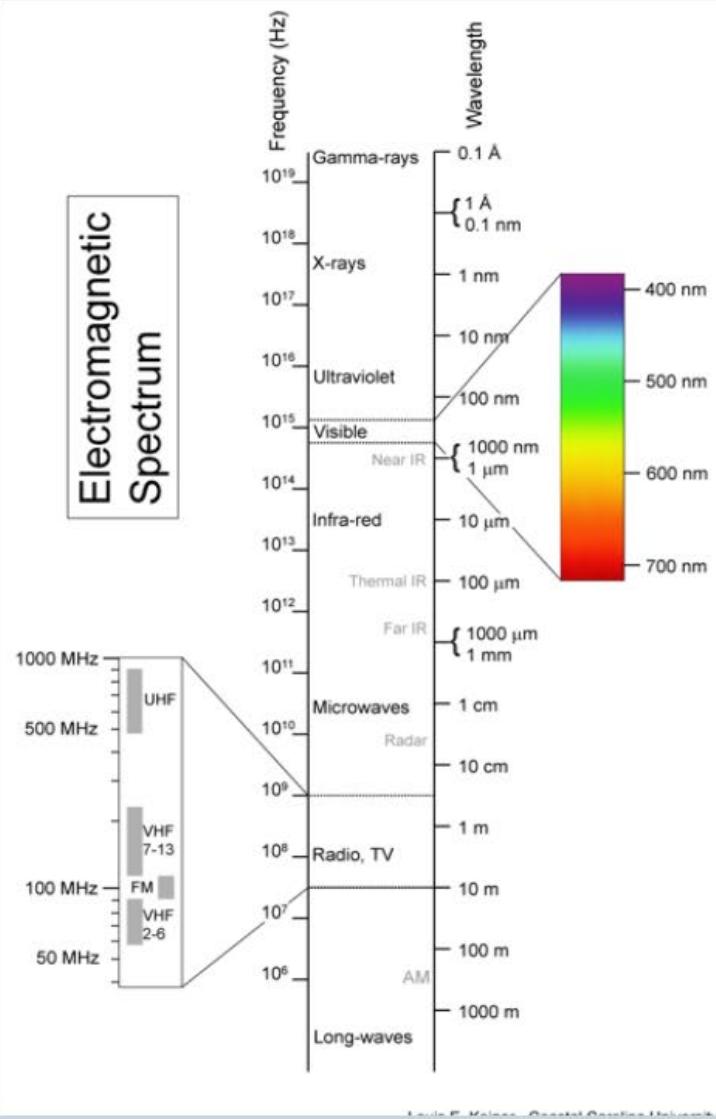
# PollEverywhere quizzes

<https://pollev.com/rasmuspulse538>



# What has the most energy?

Electromagnetic Spectrum

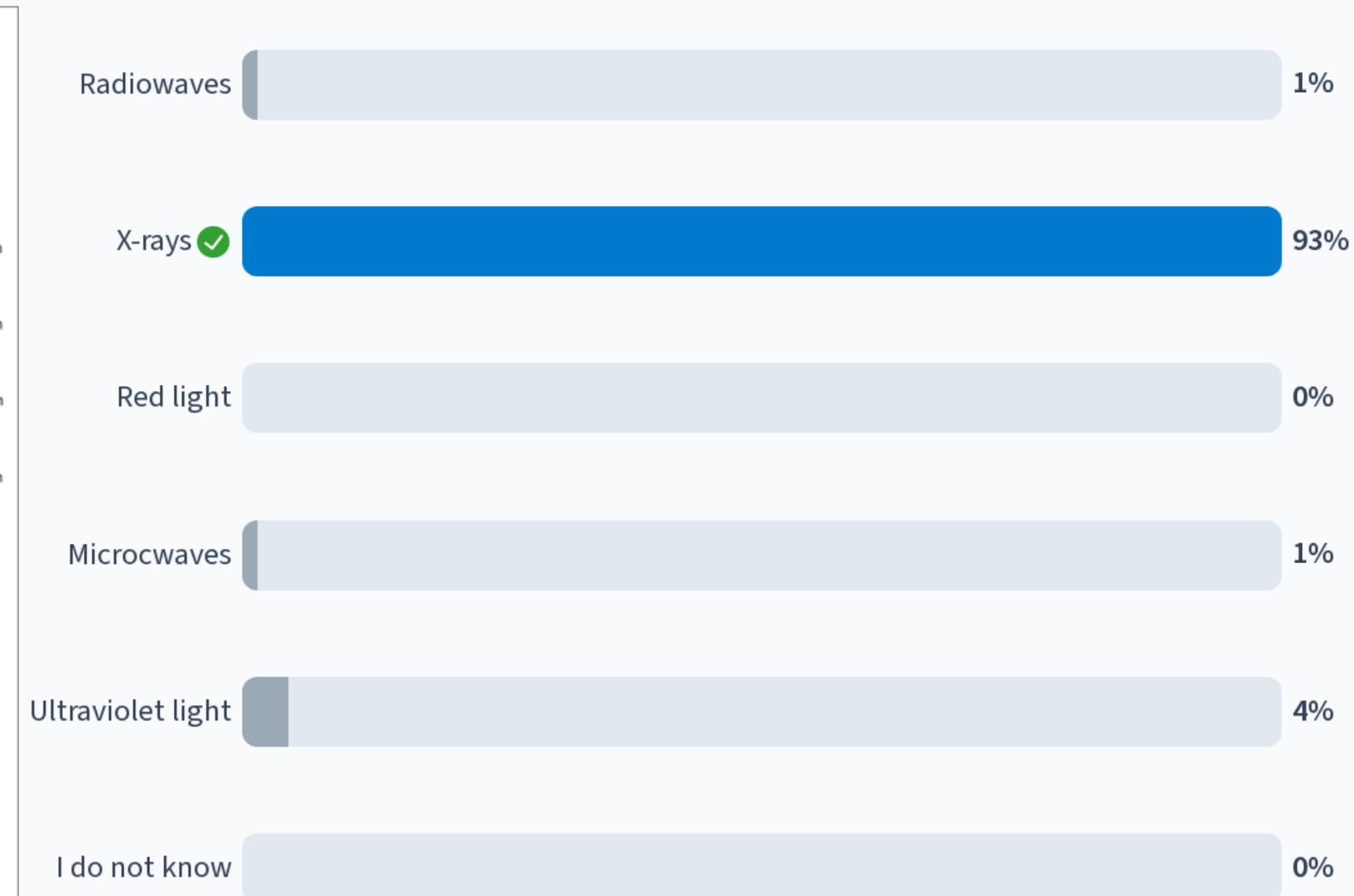
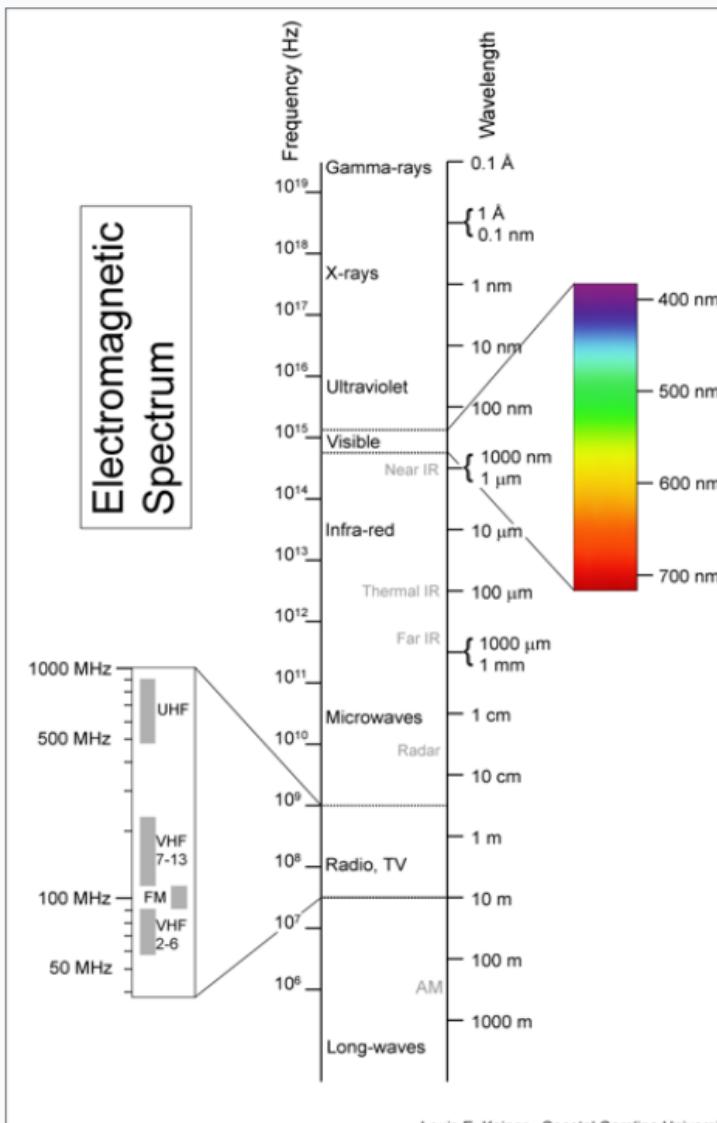


- Radiowaves
- X-rays
- Red light
- Microwaves
- Ultraviolet light
- I do not know

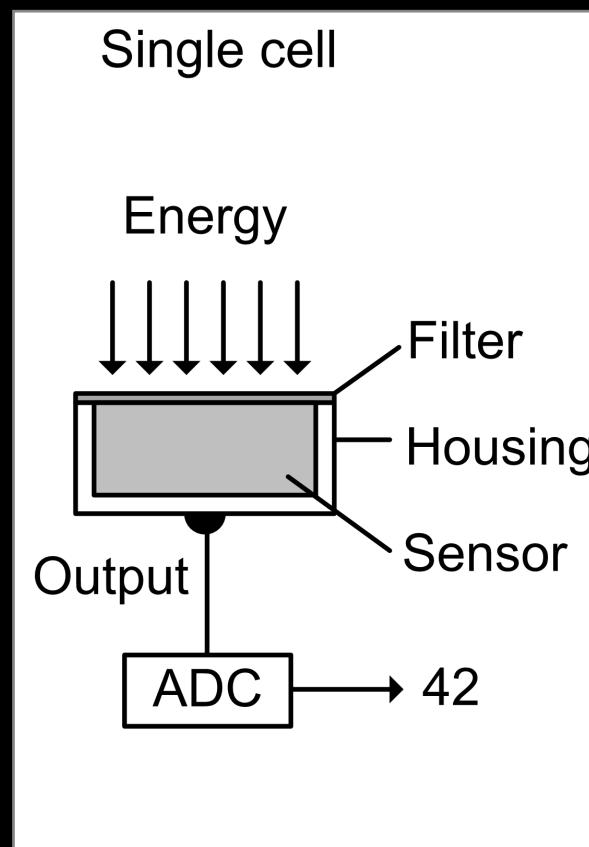


Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)

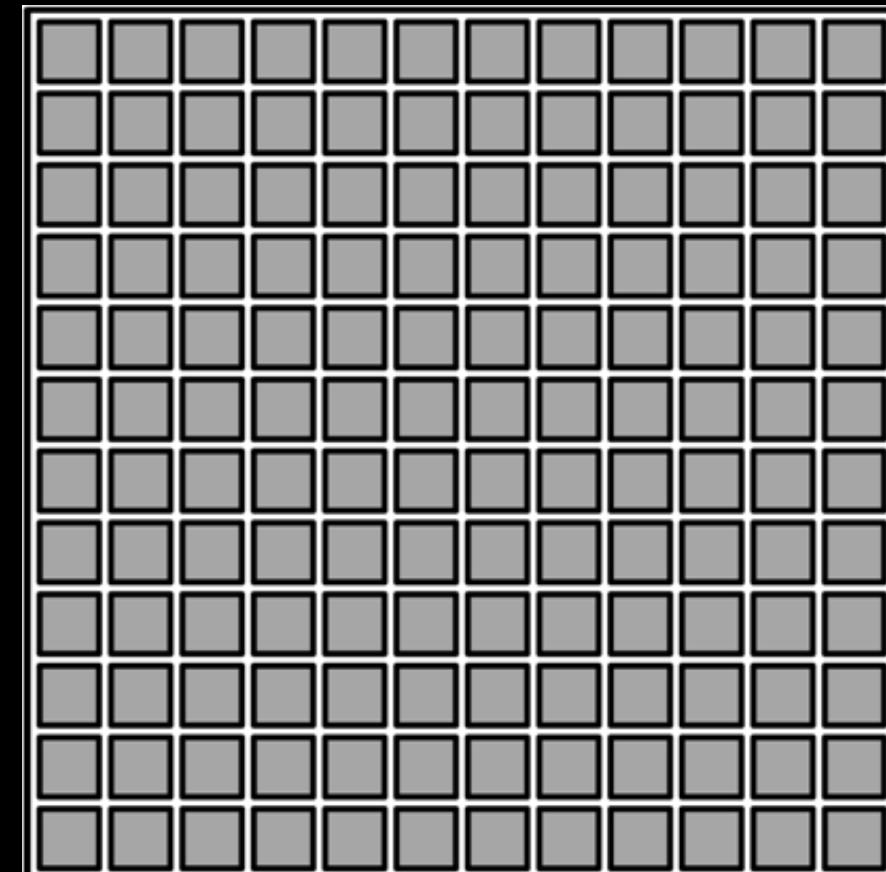
# What has the most energy?



# How do light become a digital image?

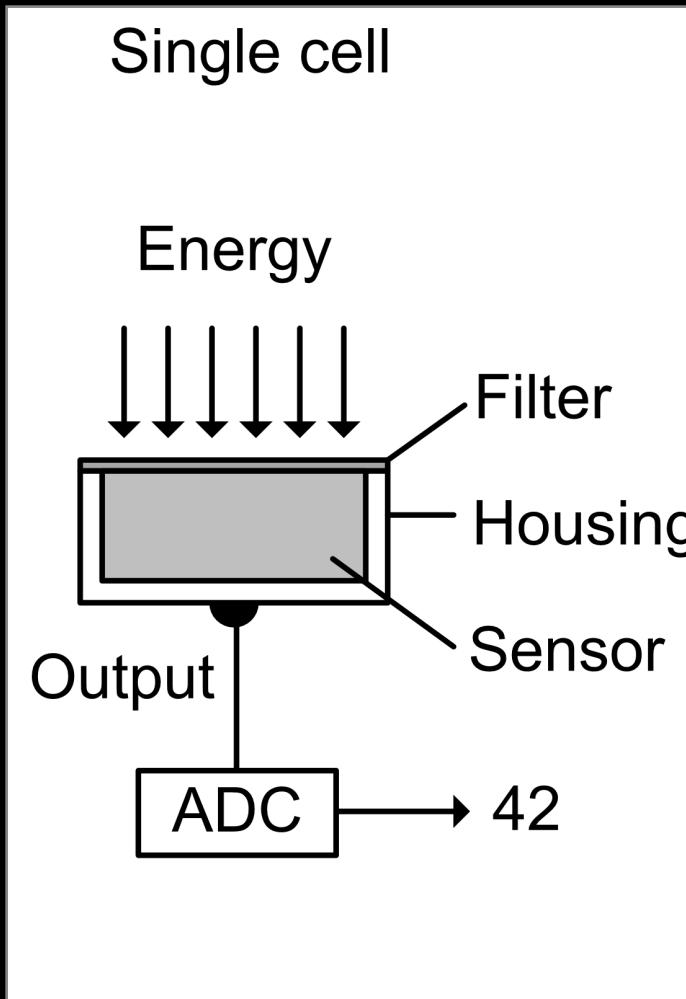


Charged coupled device (CCD-chip)



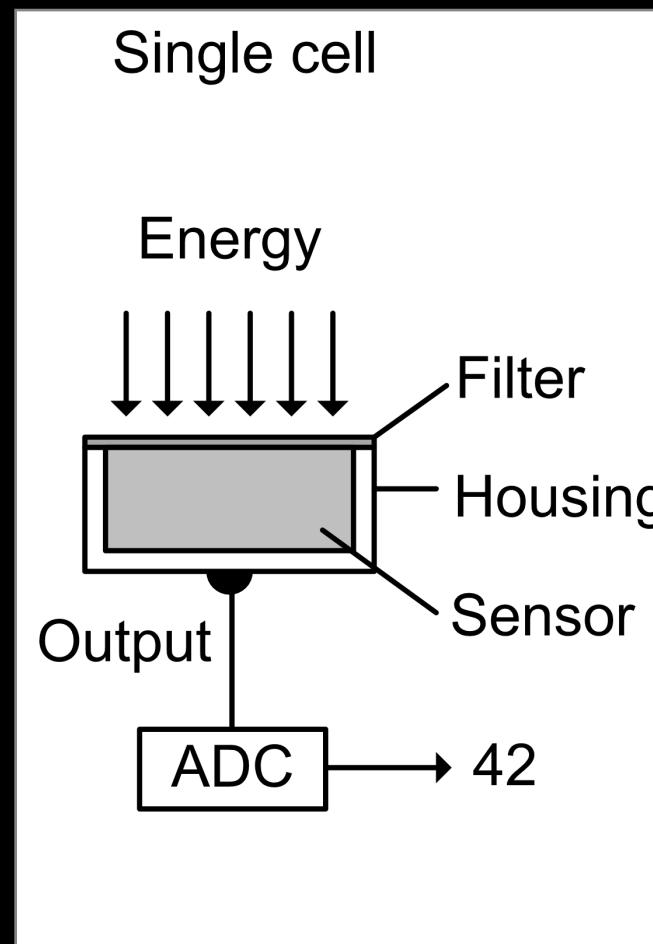
The digital film!

# The CCD cell

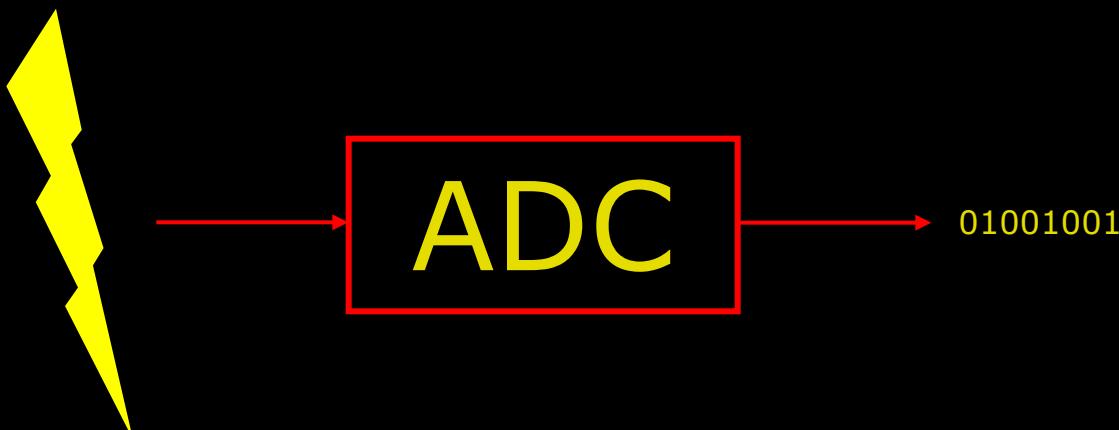


- The cell can be seen as a well that collects energy
- It collect energy for a limited time (*to be charged*)
  - Exposure time
  - Integration time
  - Shutter

# The CCD cell - conversion



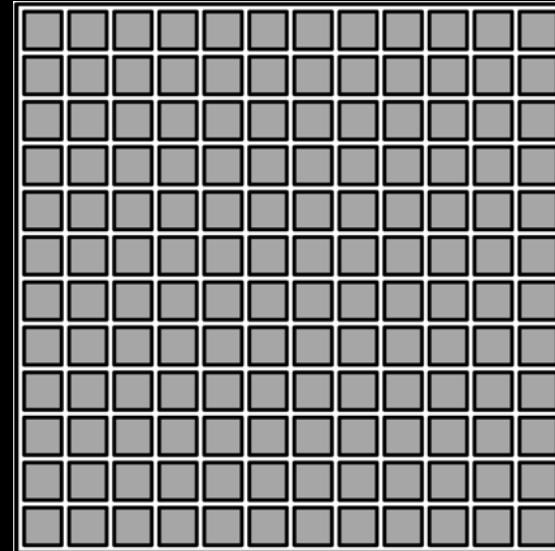
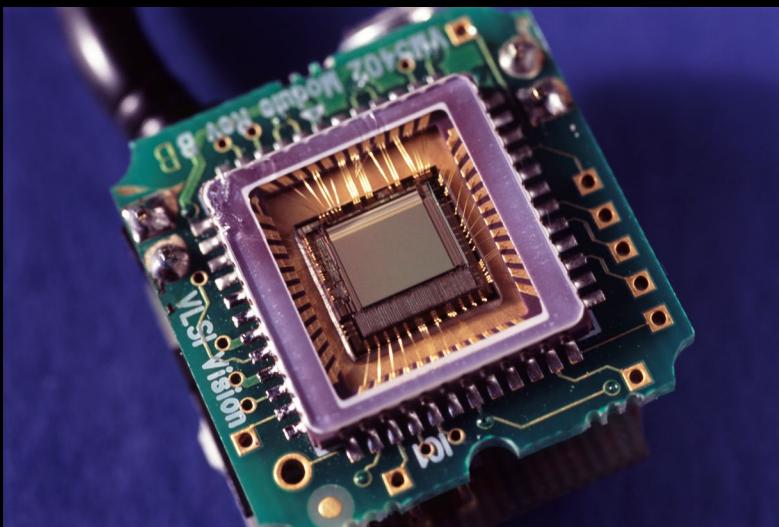
- Energy transformed to a digital number
  - Analog-to-Digital converter (ADC)
- Takes a an “analogue signal” and converts it to a digital signal



$$(01001001)_2 = (0 \times 2^7) + (1 \times 2^6) + (0 \times 2^5) + (0 \times 2^4) + (1 \times 2^3) + (0 \times 2^2) + (0 \times 2^1) + (1 \times 2^0) = (73)_{10}$$

# CCD and images

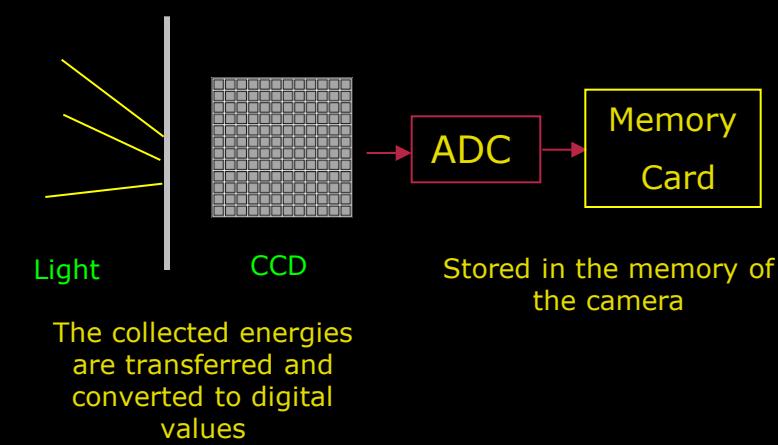
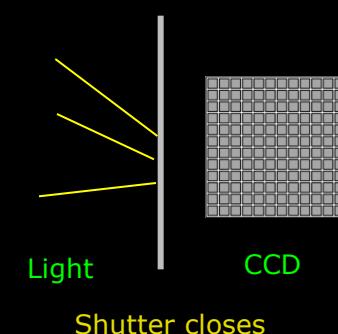
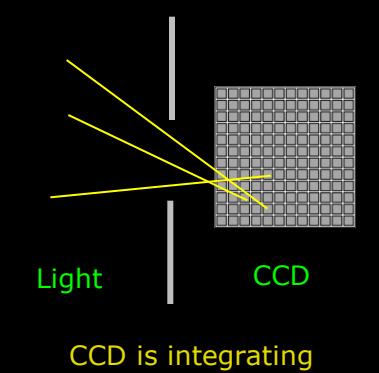
- Surprise! 1 CCD cell = 1 pixel
  - Only for grayscale images
  - More complex for RGB images
- 10 MPixel camera
  - 10 millions analog to digital conversions for one image!



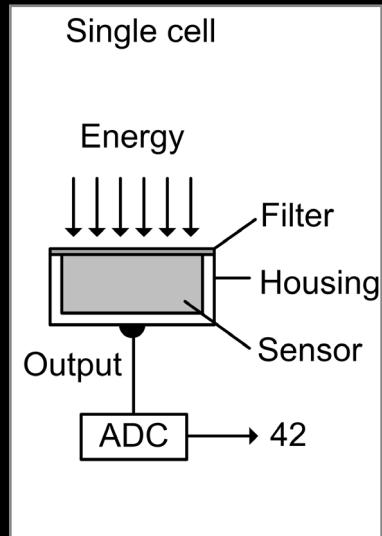
# What happens when you press the button?



The shutter opens and the CCD is hit by light



# Question: Integration time



■ What happens if we integrate over long time?

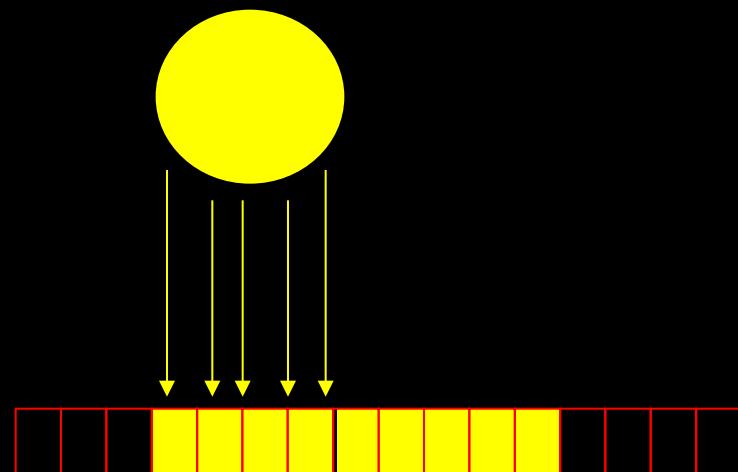
- Motion blur
- Over-exposure (the well is overrunning)
- Blooming

■ Short integration time

- Noise
- Lack of contrast

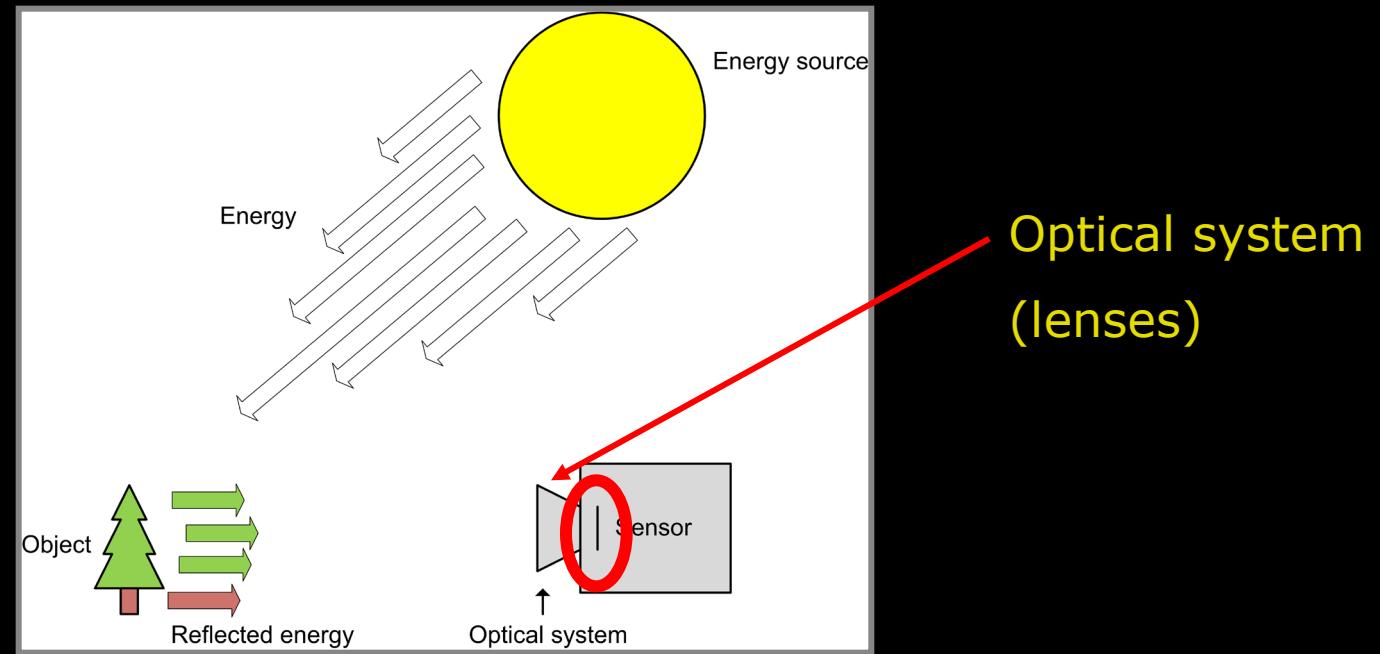
# Motion blur

- Causes blurring of the moving object



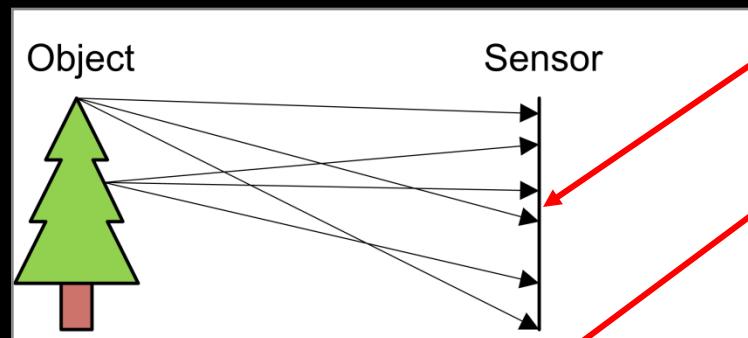
# The bigger picture

- A camera is more than a CCD!
- The CCD is the sensor!
- There is also “an optical system”



# Optical system

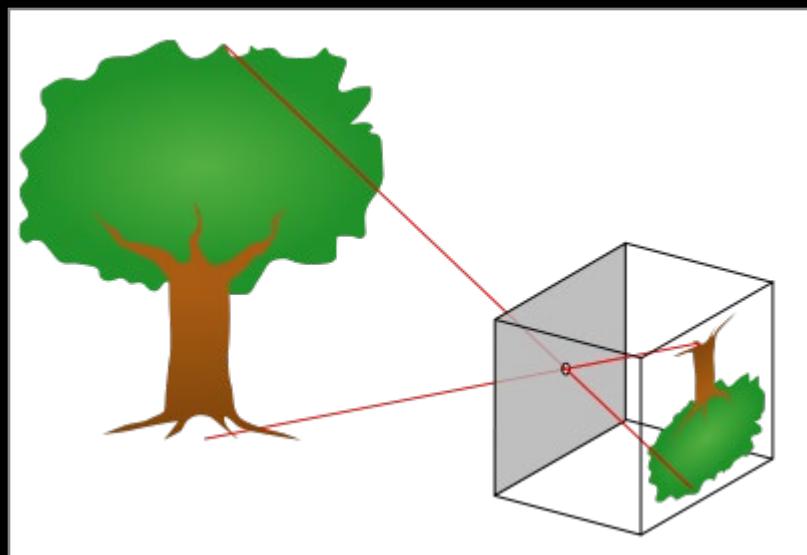
- How do we get an image on the CCD?
- Light follows a straight line
- Light that hit one spot reflects in many directions



Same point hit by rays from  
all over the object

Barrier with tiny hole

# Pinhole camera



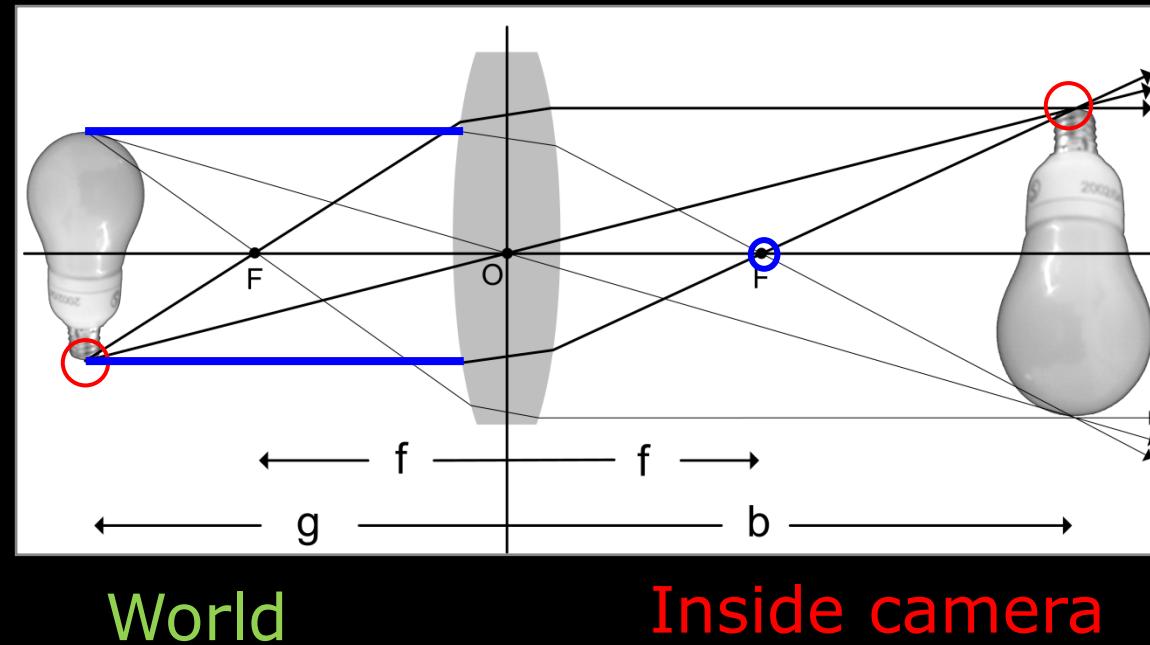
- Light coming through the tiny hole – any problems?
  - Very little light!
- How do we get more light inside the camera?
  - While keeping the focus?



A lens!

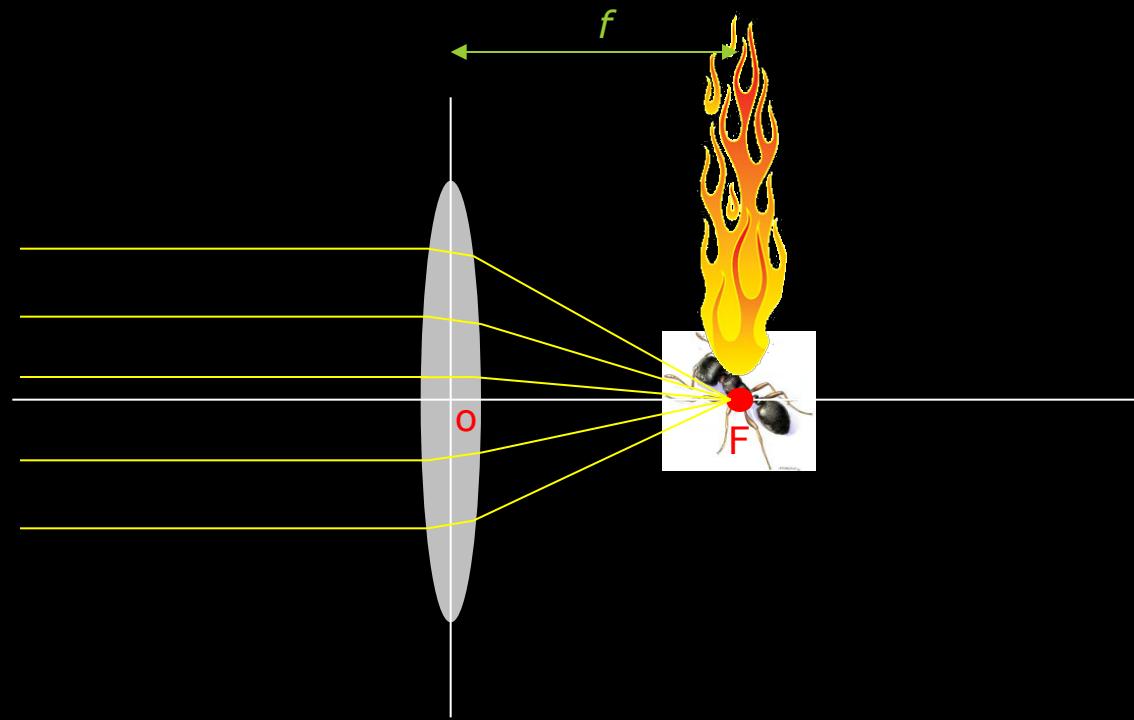
# The lens

- A lens focuses a bundle of rays to one point
- Parallel rays pass through a focal point **F** at a distance  $f$  beyond the plane of the lens.  
 $f$  is the focal length
- O is the optical centre. F and O span the optical axis

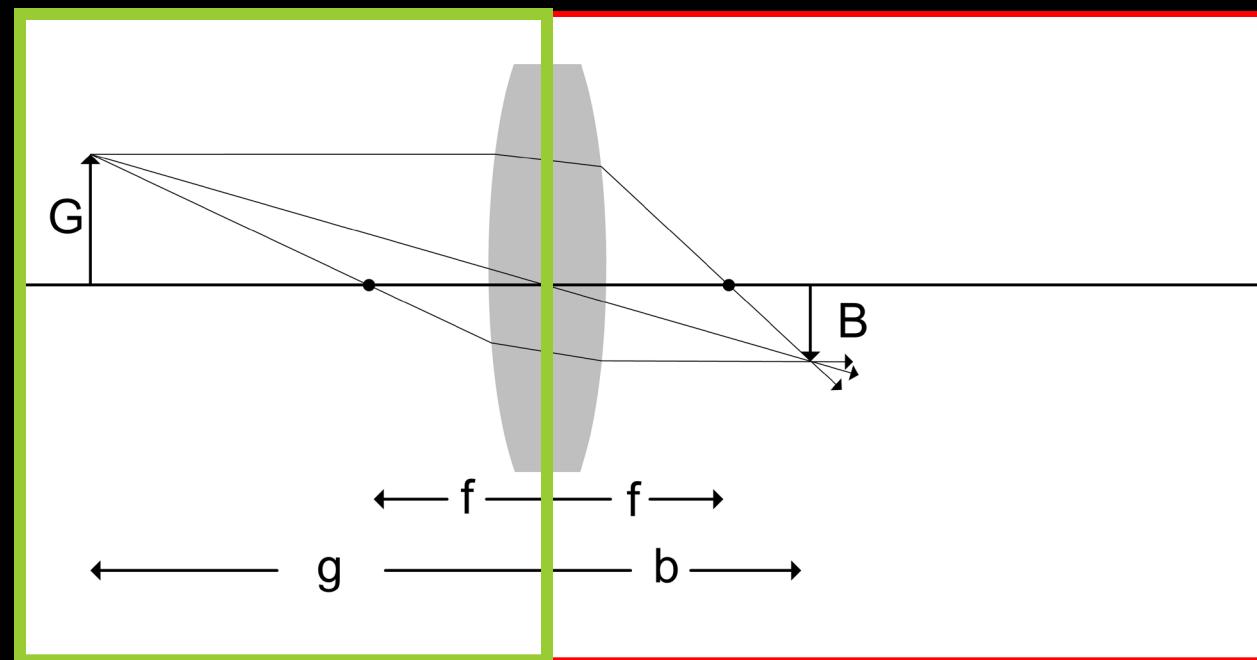


# Focal point – focal length

- Light coming from “really far away” can be seen as parallel rays
- Rays intersect at the focal point
- Distance from optical centre O to focal point F is called *focal length f*



# Where do non-parallel rays meet?



$g$  – distance to object

$b$  – distance to intersection

$$\frac{1}{g} + \frac{1}{b} = \frac{1}{f}$$

Thin lens equation

or

Gauss' lens equation

## Where do the rays meet

b=1 mm

b=4 mm

b = 5 mm

b=6 mm

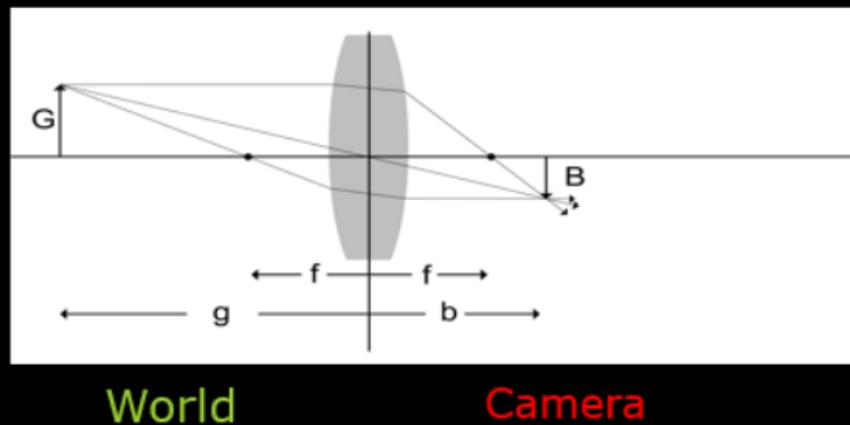
b=7 mm

Do not know

## Where do the rays meet

- Camera with focal length of 5 mm
- Rasmus is standing 3 meters away
- Where do the rays meet in the camera? (b)

$$\frac{1}{g} + \frac{1}{b} = \frac{1}{f}$$



b=1 mm 0%

b=4 mm 4%

b = 5 mm ✓ 93%

b=6 mm 4%

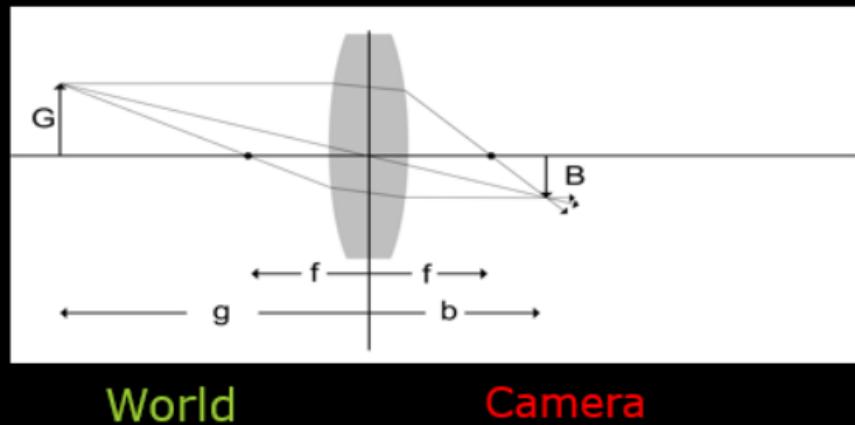
b=7 mm 0%

Do not know 0%

## Where do the rays meet

- Camera with focal length of 5 mm
- Rasmus is standing 3 meters away
- Where do the rays meet in the camera? (b)

$$\frac{1}{g} + \frac{1}{b} = \frac{1}{f}$$



b=1 mm

0%

b=4 mm

4%

b = 5 mm

93%

b=6 mm

4%

b=7 mm

0%

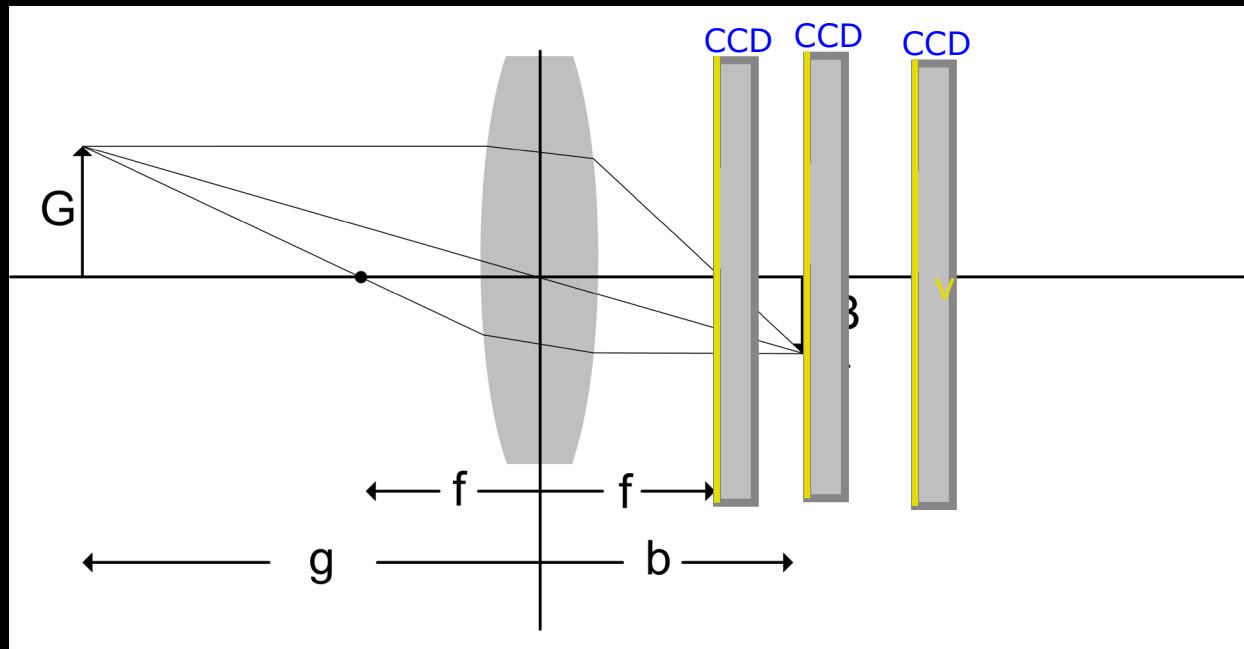
Do not know

0%

# Focus or not to focus?



# How do we make focused images? Placing the CCD right

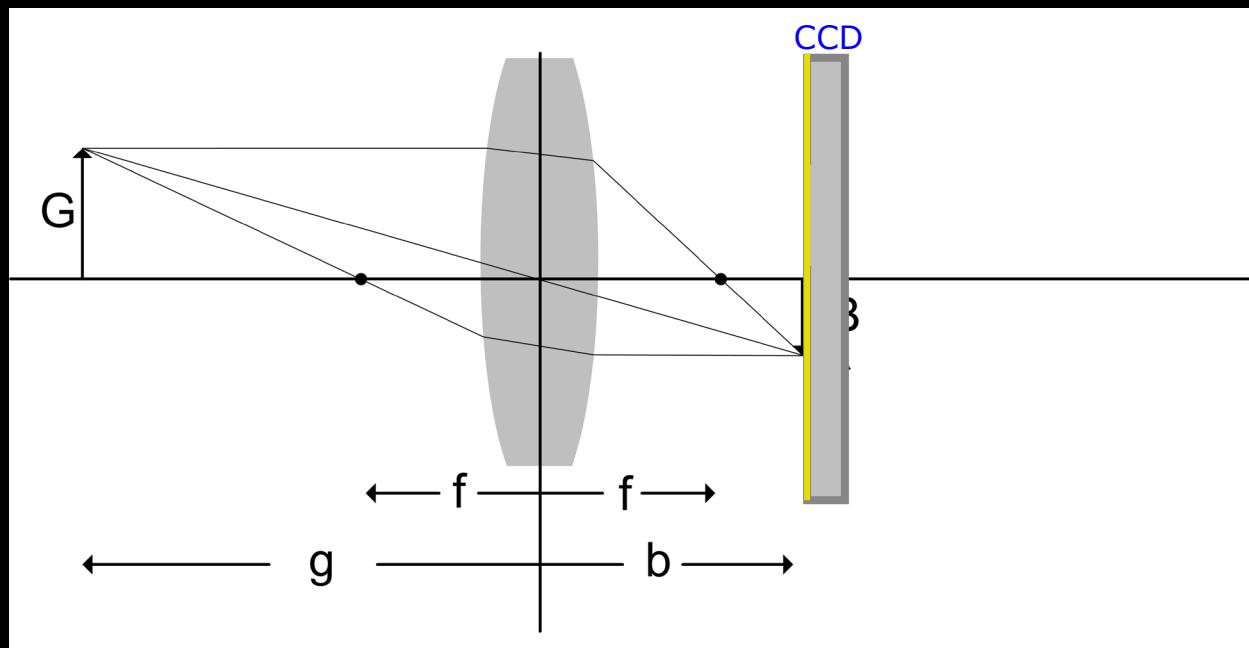


World  
 $g$  – distance to object

Camera  
 $b$  – distance to intersection

CCD should  
be placed  
at b!

# Focusing



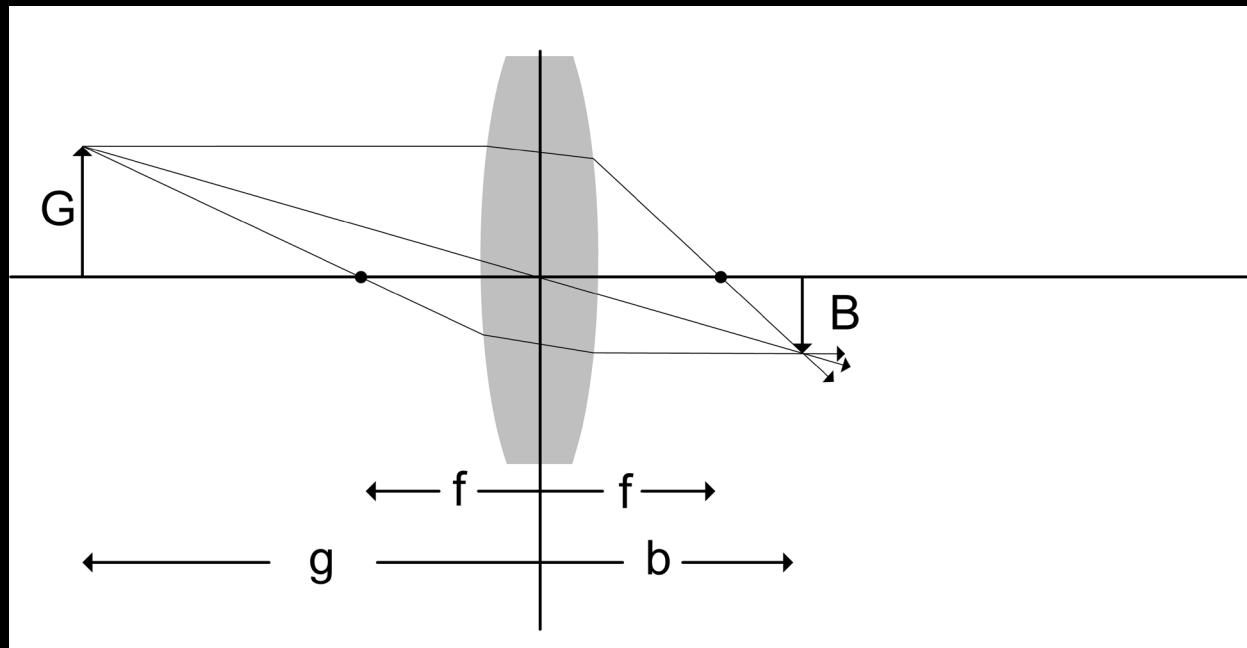
World  
 $g$  – distance to object

Camera  
 $b$  – distance to intersection

- We move the camera
- Distance to object ( $g$ ) changes
- $f$  is fixed
- $b$  changes
- Move CCD to  $b$ 
  - Focusing

$$\frac{1}{g} + \frac{1}{b} = \frac{1}{f}$$

# Object size



What is the size of  
an object on the  
CCD?

## World

$g$  – distance to object

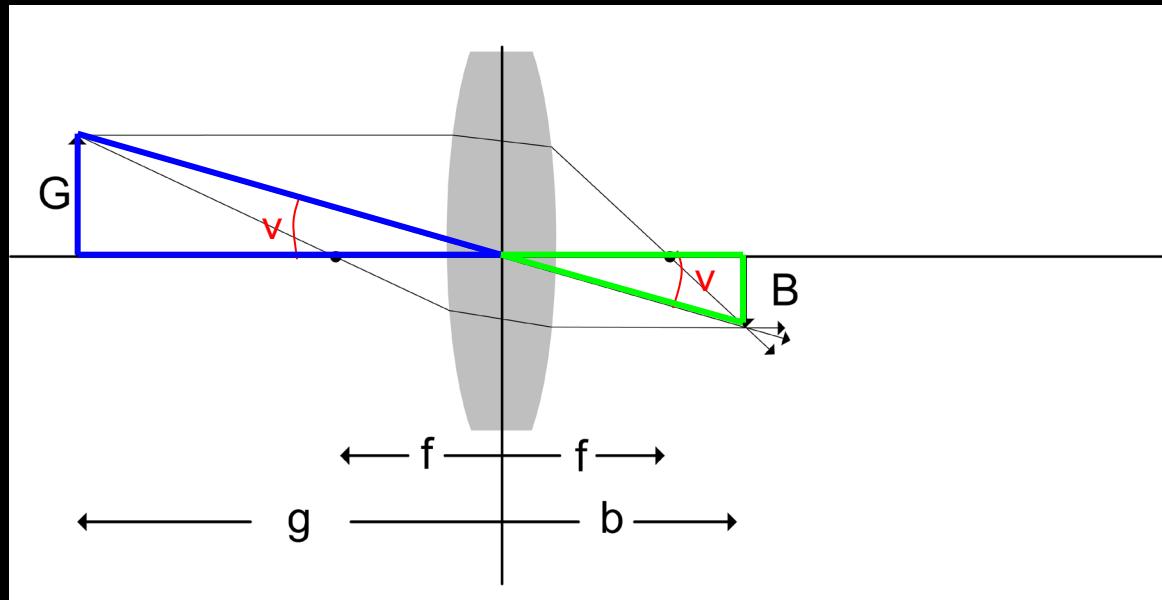
$G$  – Object height

## Camera

$b$  – distance to intersection

$B$  – object height on CCD

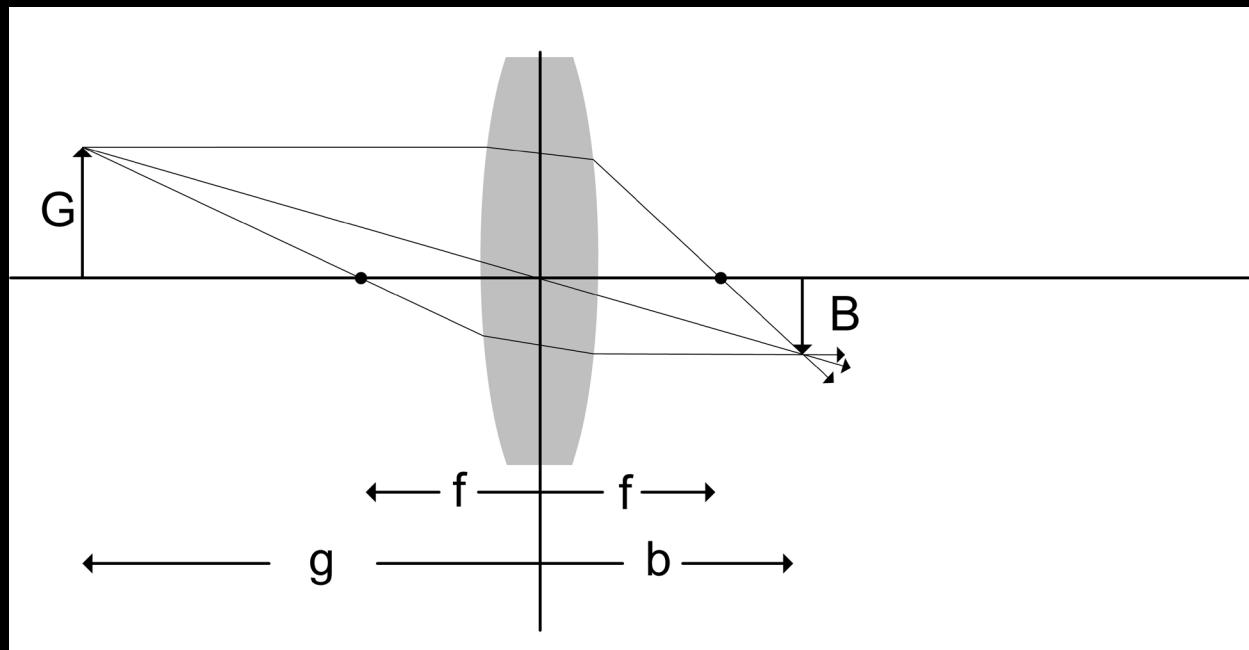
# An important relation!



- Two triangles
- One with side length  $g$  and one with  $b$
- $B$  and  $G$  are related! – how?
- tangent

$$\frac{b}{B} = \frac{g}{G}$$

# An important relation!



$$\frac{b}{B} = \frac{g}{G}$$

World

$g$  – distance to object

$G$  – Object height

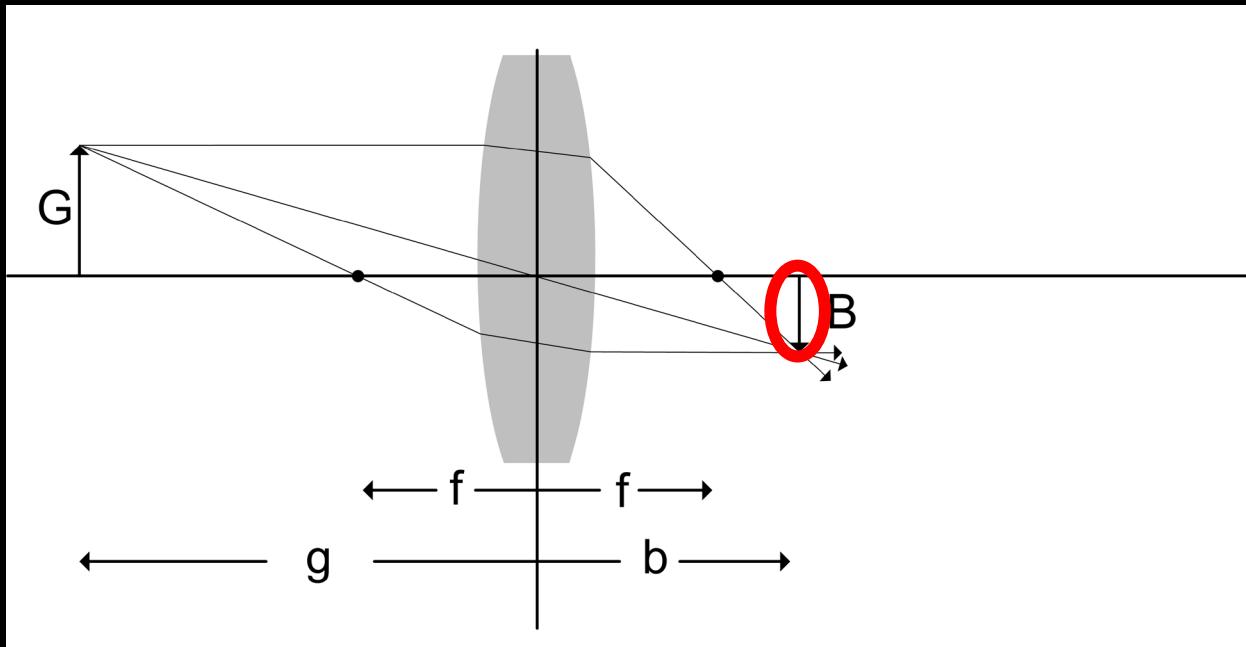
Camera

$b$  – distance to intersection

$B$  – object height on CCD

# How do we Zoom ?

We want to make B larger! How?



## World

$g$  – distance to object

$G$  – Object height

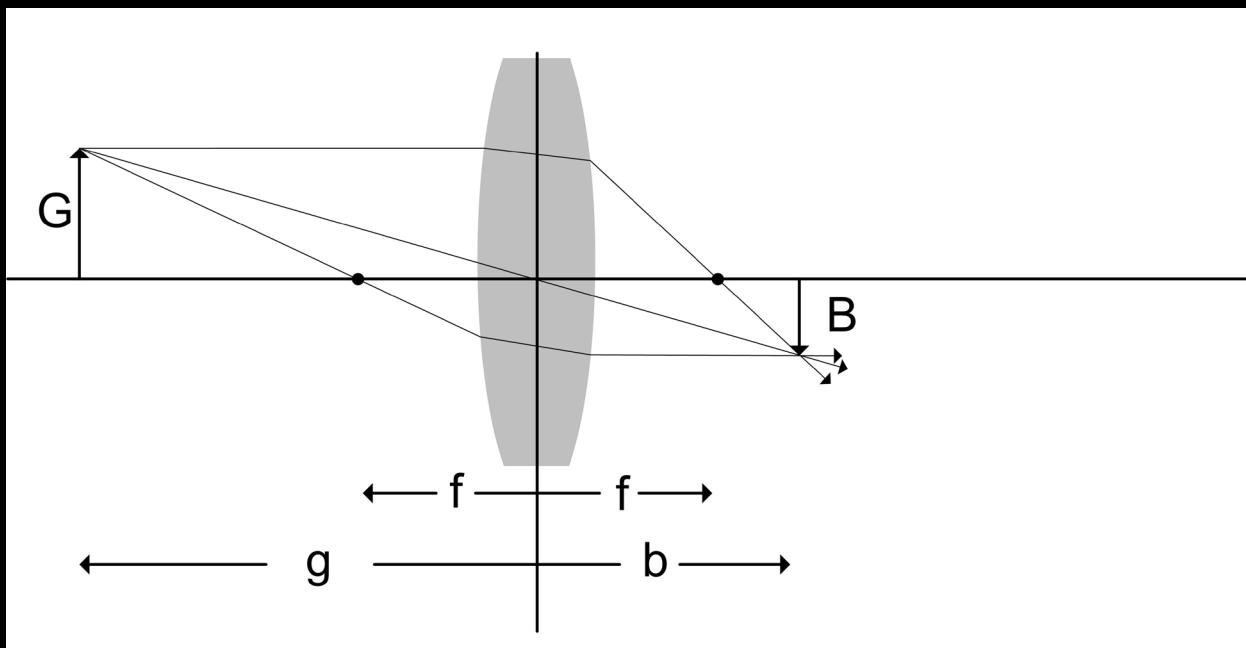
## Camera

$b$  – distance to intersection

$B$  – object height on CCD

# Zoom

We want to make B larger! How?



**World**  
 $g$  – distance to object  
 $G$  – Object height

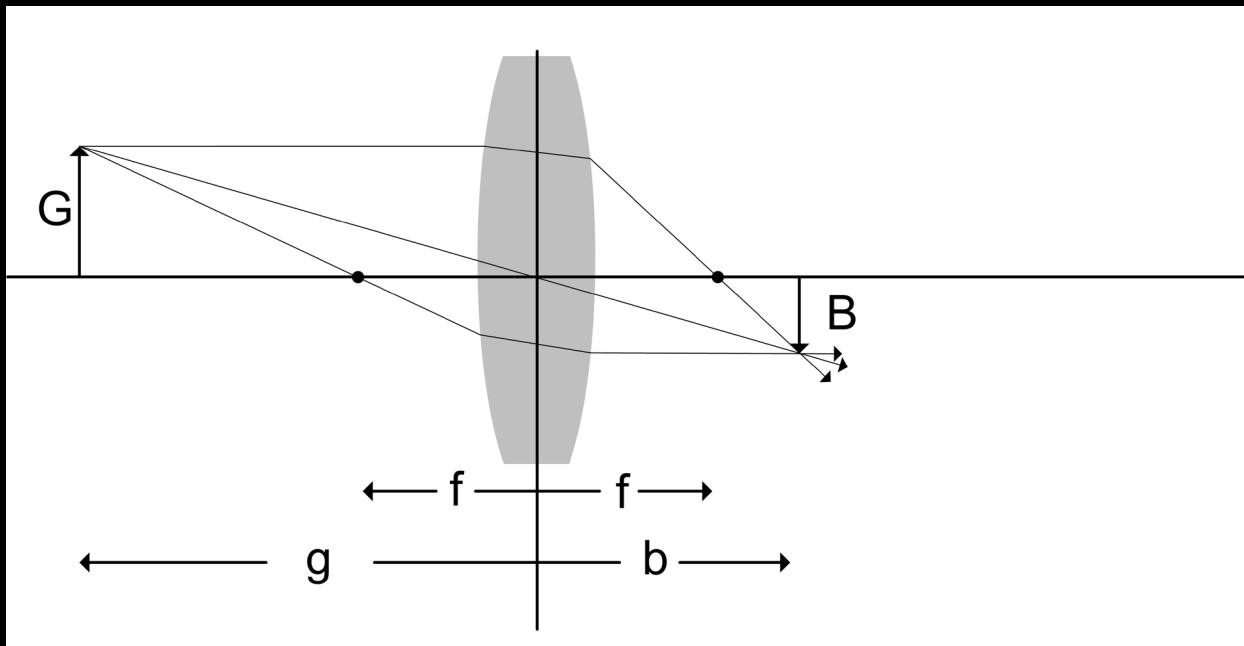
**Camera**  
 $b$  – distance to intersection  
 $B$  – object height on CCD

$$\frac{b}{B} = \frac{g}{G}$$
$$B = \frac{b}{g} G$$

Fixed

# Zoom

We want to make B larger – changing b!



**World**  
 $g$  – distance to object  
 $G$  – Object height

**Camera**  
 $b$  – distance to intersection  
 $B$  – object height on CCD

$$B = \frac{b}{g}$$

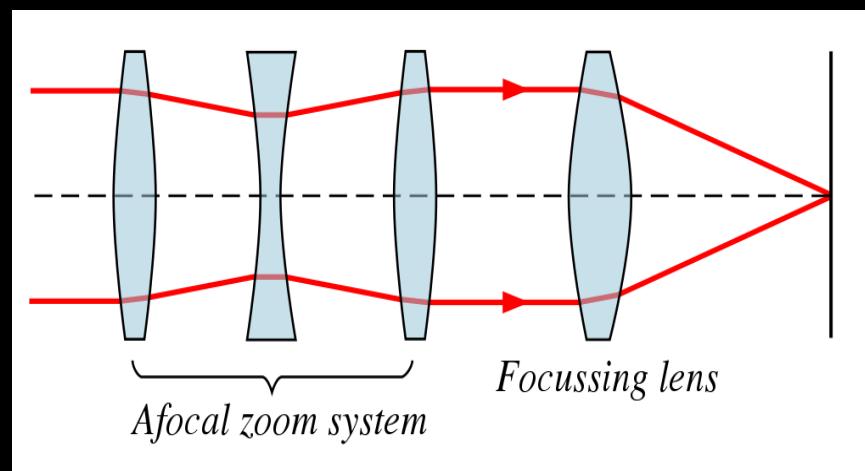
$$\frac{1}{g} + \frac{1}{b} = \frac{1}{f}$$

constant

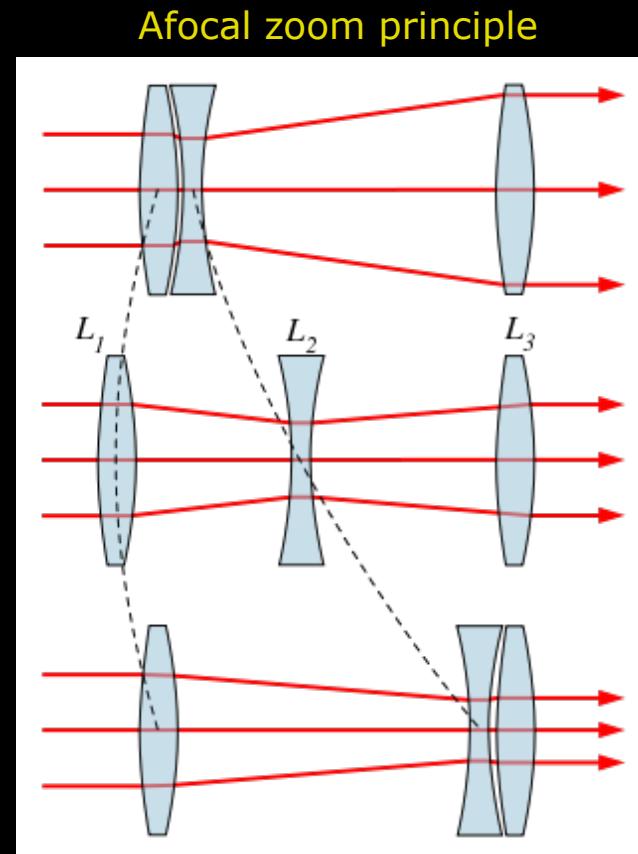
To change  $B$  we change the focal length!

# Changing the focal length?

- Not possible on a simple lens
- Need a “zoom lens”
- Several lenses together

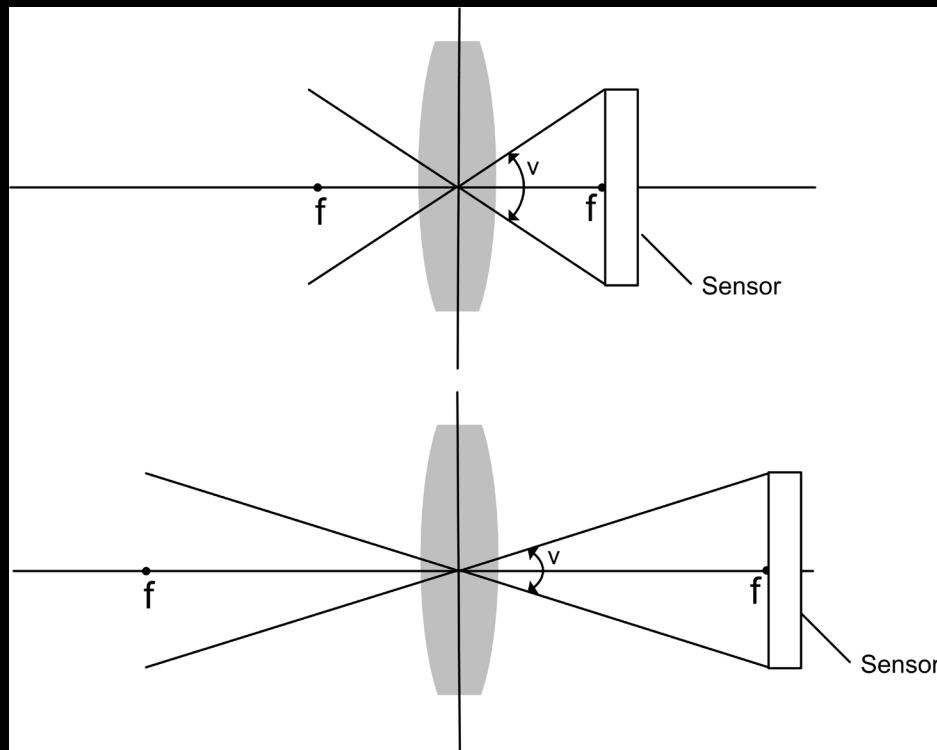


From Wikipedia: [wikipedia.org/wiki/Zoom\\_lens](https://wikipedia.org/wiki/Zoom_lens)



# Field of view (FOV)

Two cameras with different focal length



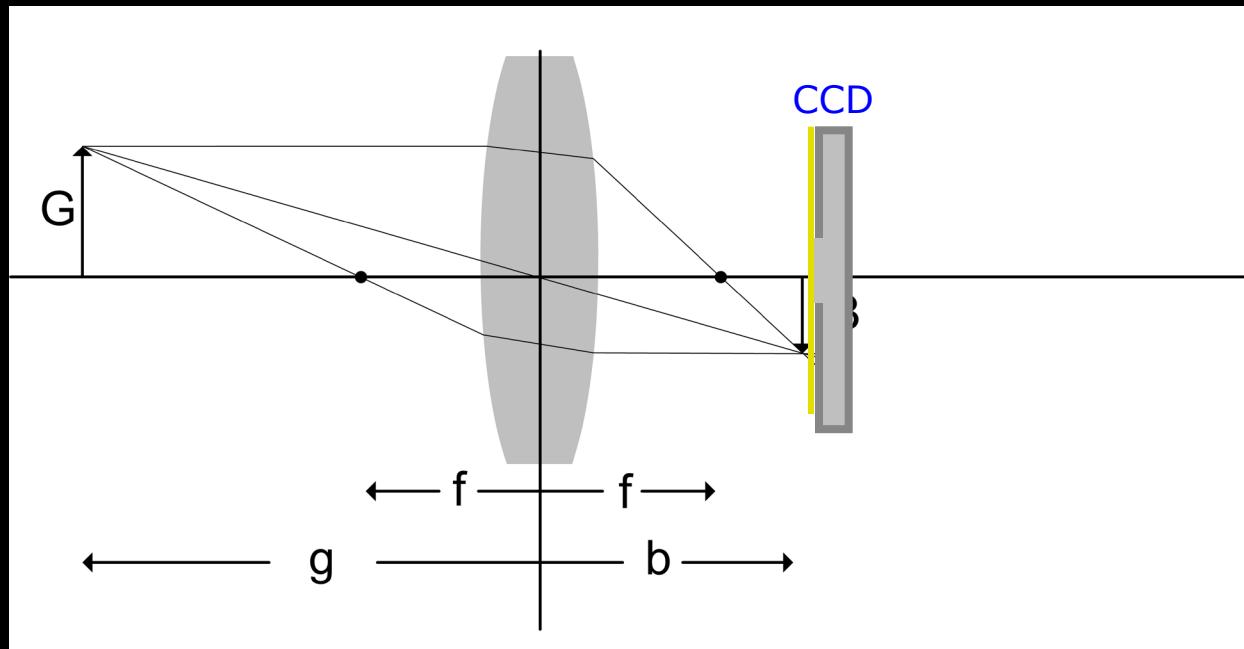
- Described by an angle
  - Large angle the larger FOV
- Depends on
  - CCD size
  - Focal length
- Fisheye lens
  - Small focal length
  - Large field of view
- CCD chip is a rectangle
  - Horizontal field of view
  - Vertical field of view
- Zoom changes field of view
  - Optical zoom
  - Digital zoom



# Depth of field - dybdeskarphed



# Depth of field

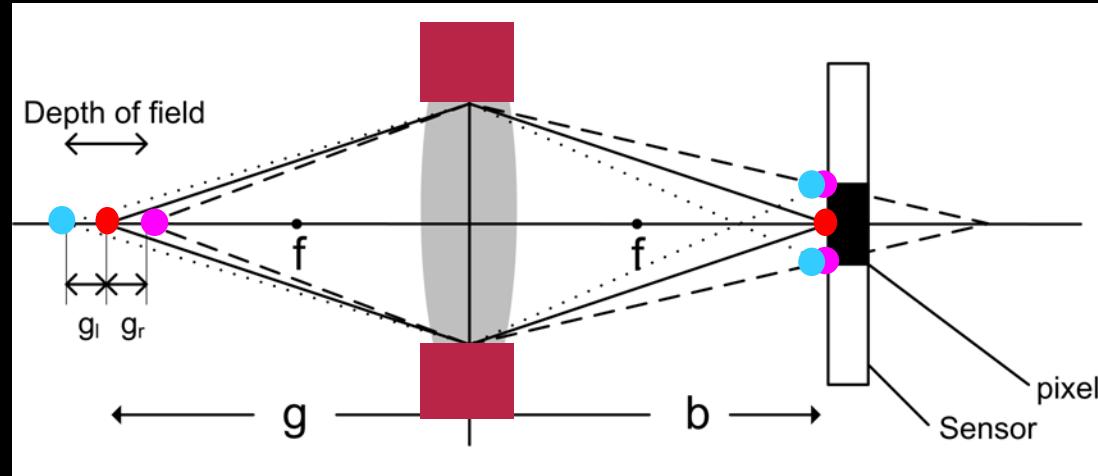


World  
 $g$  – distance to object

Camera  
 $b$  – distance to intersection

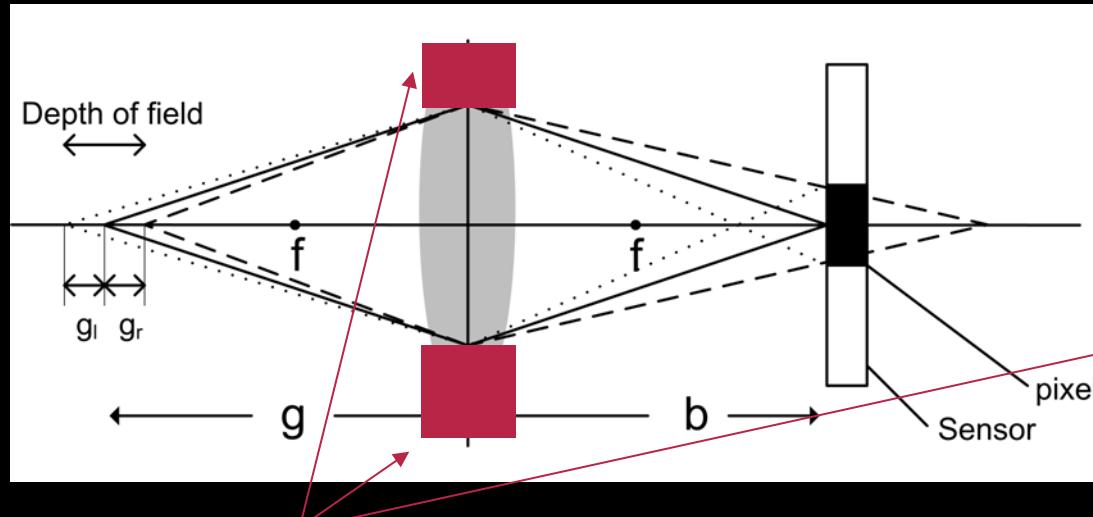
- CCD should be placed at  $b$
- $g$  is fixed – only focus at one distance!

# Depth of field



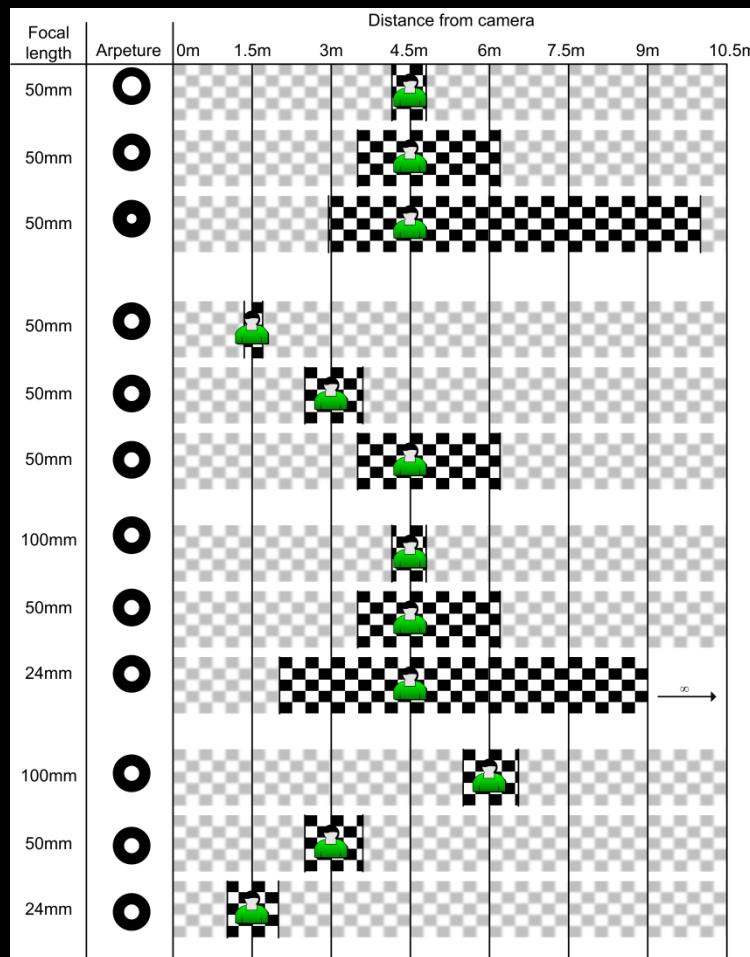
- Look at one pixel in the middle
- The object is placed at distance  $g$
- How much can we move the object?
  - Light has to hit the same pixel
- Move it to the left ( $g_l$ )
- move it to the right ( $g_r$ ) – still hit the same pixel (but twice)

# Depth of field – Aperture (blænde)

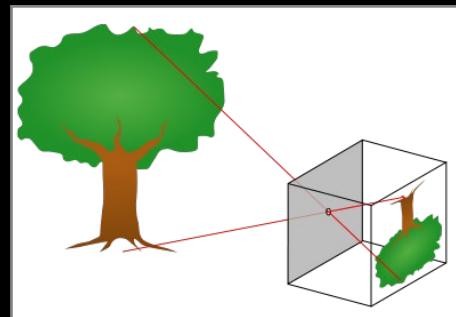


- The **aperture** controls the amount of light
- Small aperture
  - large depth of field
  - Less light -> longer exposure

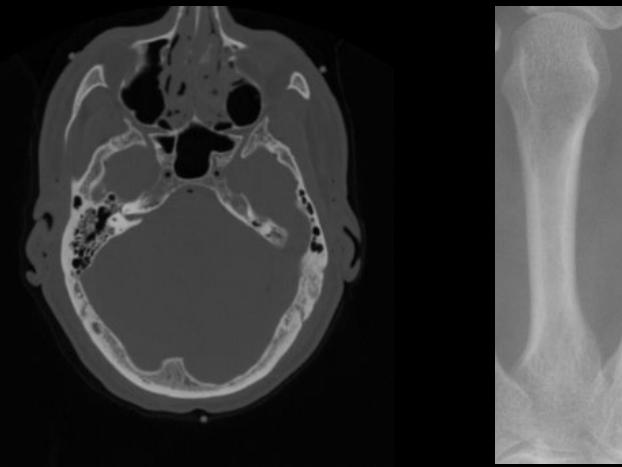
# How to acquire a good image?



- Distance to object
- Motion of object
- Zoom
- Focus
- Depth-of-fields
- Focal length
- Shutter
- Field-of-view
- Aperture (DK: blænde)
- Sensor (size and type)



# Image storage and compression



# Learning objectives – image storage and compression

- Compute the run-length code of a grayscale image
- Compute the chain coding of a binary image
- Compute the compression ratio
- Describe the difference between a lossless and a lossy image format
- Decide if a given image should be stored using a lossless or a lossy image format

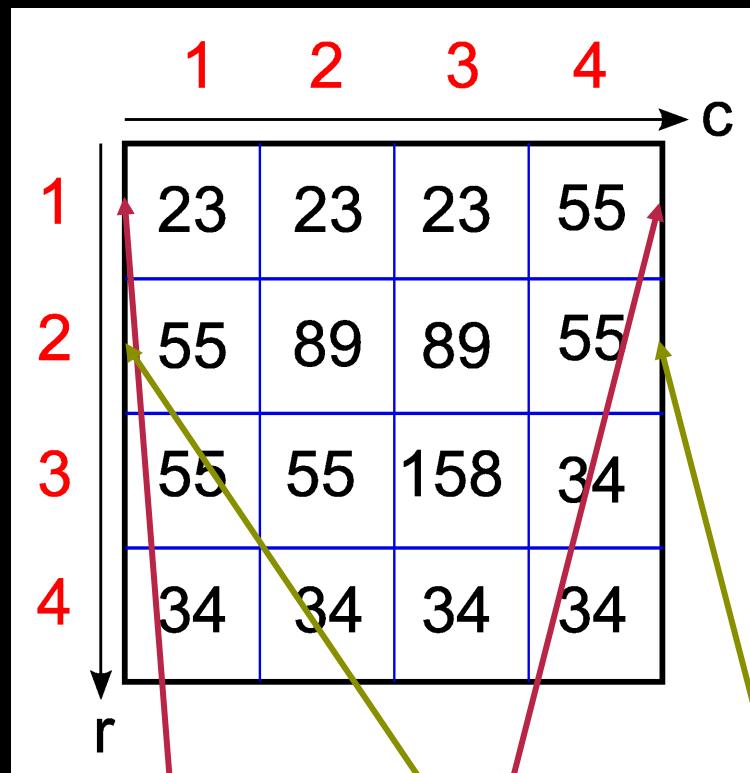
# Hard disks, memory cards, CDs etc



- Storage for bytes!
  - 500 GB?
  - 500 GigaBytes = 500.000.000.000 bytes!
- A hard disk does not know anything about images
- Stores data as lists of bytes
  - 17, 255, 1, 3, 87, 98, 11, ...
- File on a hard disk
  - It has a length (in bytes, MB, GB)
  - Contains numbers! (Bytes)

We want to make an “image file”

# Image as data

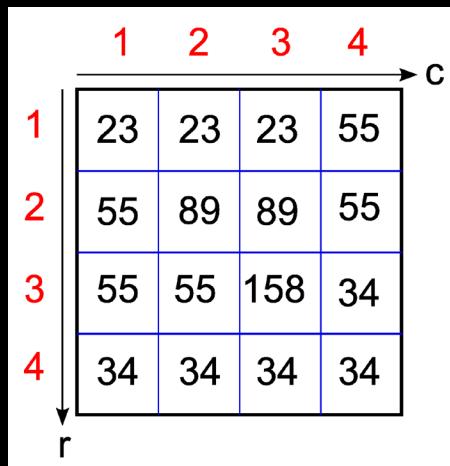


23,23,23,55,55,89,89,55,55,55,158,34,34,34,34,34

- How do we store this image as list of bytes?
- What do we need
  - Size of the image
    - Width as 2 bytes (0-65535)
    - Height as 2 bytes (0-65535)
  - The data

# Simple image format

- Stores the image as
  - A **header** with information about size
  - Data with no **compression**
- Windows Bitmap Format (BMP)



	1	2	3	4
1	23	23	23	55
2	55	89	89	55
3	55	55	158	34
4	34	34	34	34

# Compression - make something smaller

- Is there a more “compact” way to represent the data below?
- Look for patterns
  - A series of numbers can be represented how?
    - The count and the value
- What is the “count and value” code?
  - Reduced from 16 to 12 values

## Run length encoding

23,23,23|55,55,89,89,55,55,55,158|34,34,34,34,34

3,23,

2,55,

2,89,

3,55,

1,158,

5,34

# Run length encoding

- Simple but useful data compression
- General – not only for images
- Is also used by the Windows Bitmap Format (BMP)

## Run length coding of an image

1	5	5	5	3
3	2	3	3	201
201	19	19	19	147
147	130	130	130	130
147	147	147	88	88

1 1 3 5 2 3 1 2 2 3 2 2 0 1 3 1 9 2 1 4 7 4 1 3 0 3 1 4 7 2 8 8

1 1 3 5 2 3 1 3 2 2 0 1 3 1 9 2 1 4 7 4 1 3 0 3 1 4 7 2 8 8

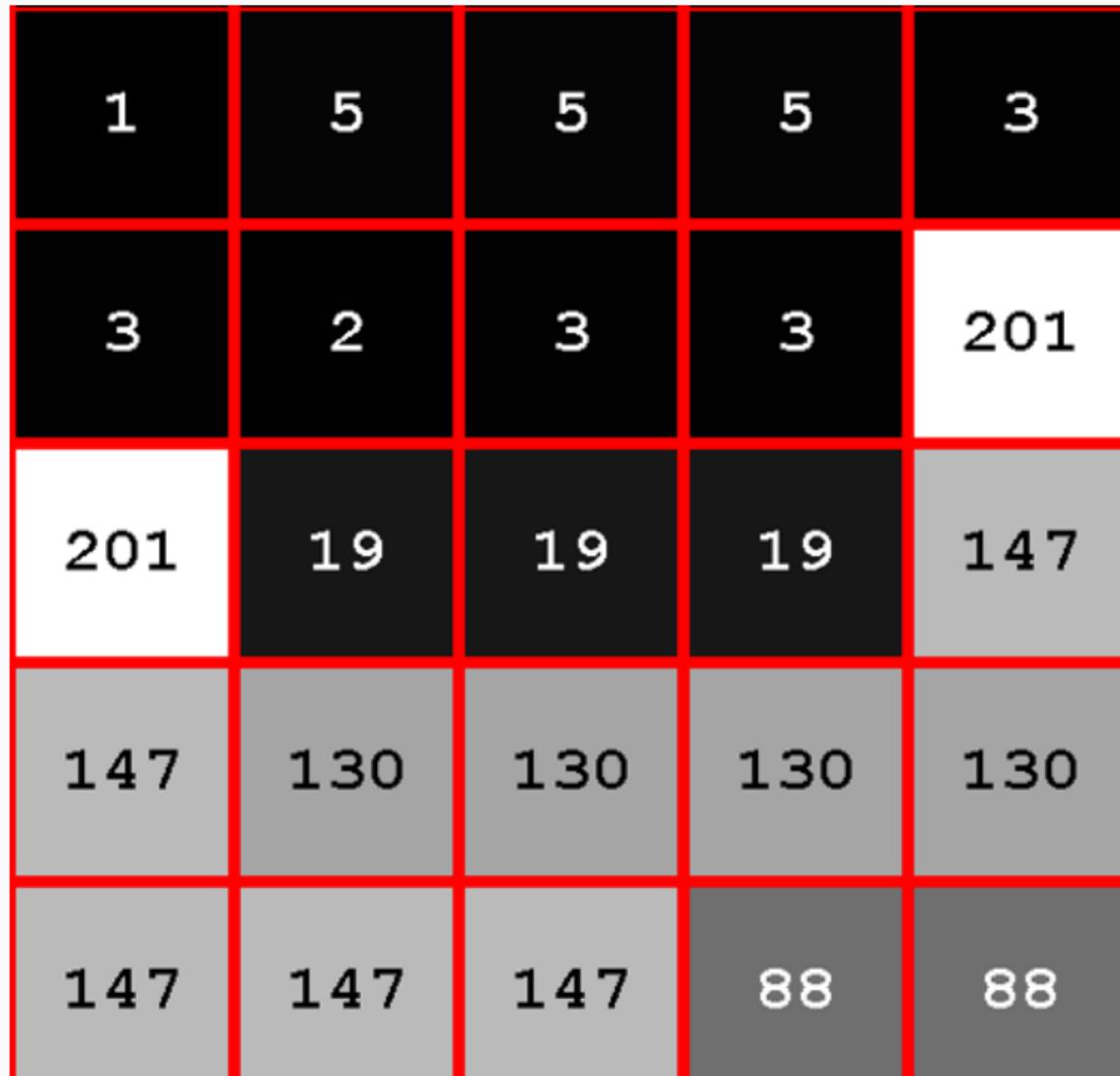
1 3 5 2 2 3 3 4 1 2 2 3 2 2 0 1 3 1 9 2 1 4 7 4 1 3 0 3 1 4 7 2 8 8

1 1 3 5 2 3 1 2 2 4 4 2 0 1 3 1 9 2 1 4 7 4 1 3 0 3 1 4 7 2 8 8

1 1 3 5 2 3 1 2 2 3 2 3 3 1 9 2 1 4 7 3 2 3 4 4 1 3 0 3 1 4 7 2 8 8

I do not know

## Run length coding of an image



1 1 3 5 2 3 1 2 2 3 2 2 0 1 3 1 9 2 1 4 7 4  
1 3 0 3 1 4 7 2 8 8

98%

1 1 3 5 2 3 1 3 2 2 0 1 3 1 9 2 1 4 7 4 1 3 0 3  
1 4 7 2 8 8

0%

1 3 5 2 2 3 3 4 1 2 2 3 2 2 0 1 3 1 9 2 1 4 7 4  
1 3 0 3 1 4 7 2 8 8

0%

1 1 3 5 2 3 1 2 2 4 4 2 0 1 3 1 9 2 1 4 7 4  
1 3 0 3 1 4 7 2 8 8

0%

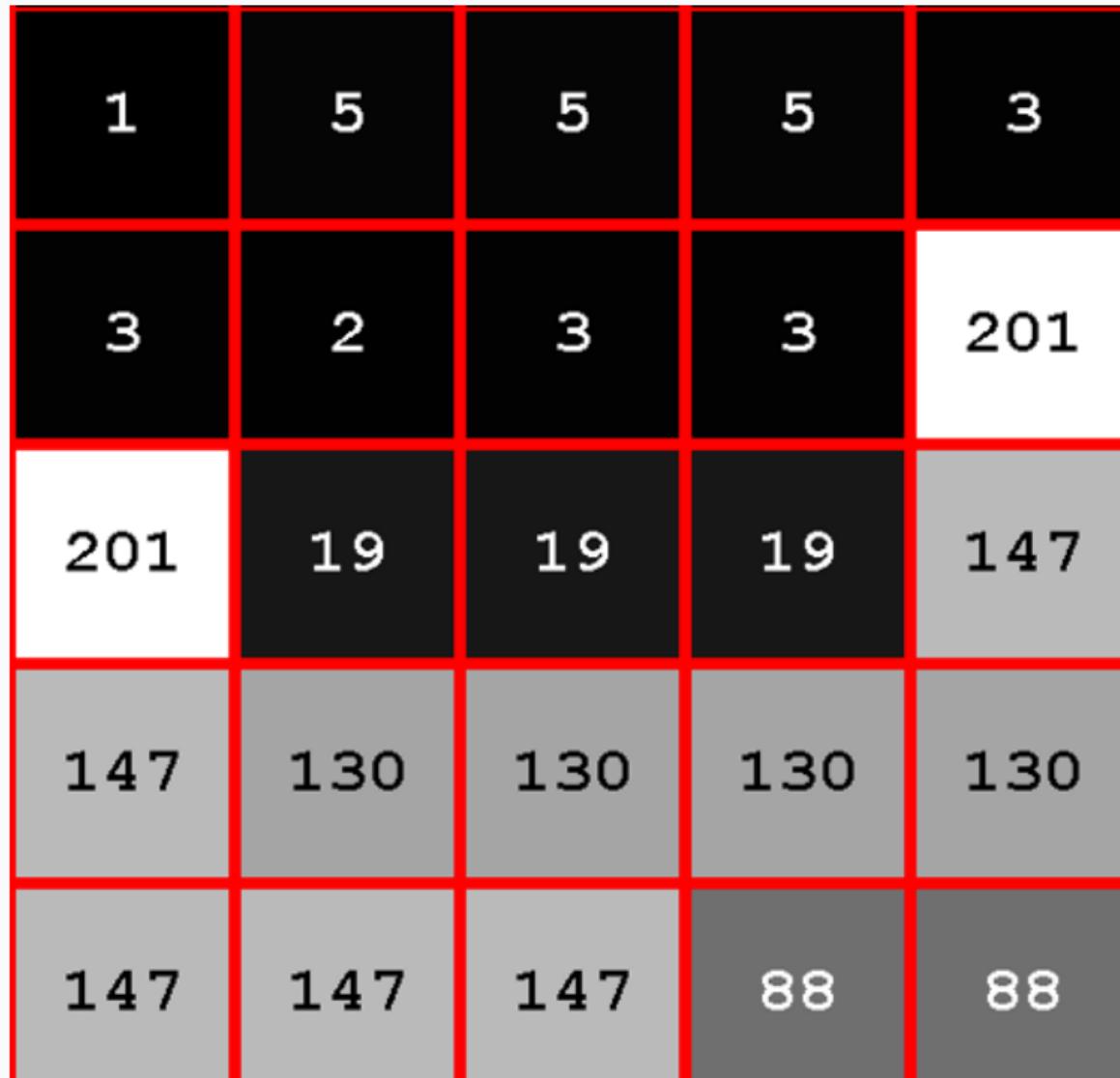
1 1 3 5 2 3 1 2 2 3 2 3 3 1 9 2 1 4 7 3 2 3 4  
4 1 3 0 3 1 4 7 2 8 8

2%

I do not know

0%

## Run length coding of an image



1 1 3 5 2 3 1 2 2 3 2 2 0 1 3 1 9 2 1 4 7 4  
1 3 0 3 1 4 7 2 8 8

98%

1 1 3 5 2 3 1 3 2 2 0 1 3 1 9 2 1 4 7 4 1 3 0 3  
1 4 7 2 8 8

0%

1 3 5 2 2 3 3 4 1 2 2 3 2 2 0 1 3 1 9 2 1 4 7 4  
1 3 0 3 1 4 7 2 8 8

0%

1 1 3 5 2 3 1 2 2 4 4 2 0 1 3 1 9 2 1 4 7 4  
1 3 0 3 1 4 7 2 8 8

0%

1 1 3 5 2 3 1 2 2 3 2 3 3 1 9 2 1 4 7 3 2 3 4  
4 1 3 0 3 1 4 7 2 8 8

2%

I do not know

0%

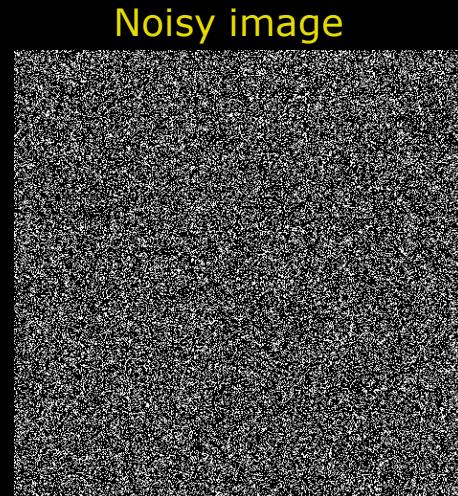
# Compression ratio – how compressed?

- Gives a measure for how much data is compressed
- Our example
  - From 16 to 12
  - $16 : 12 = 4 : 3$
  - Ratio 1.33

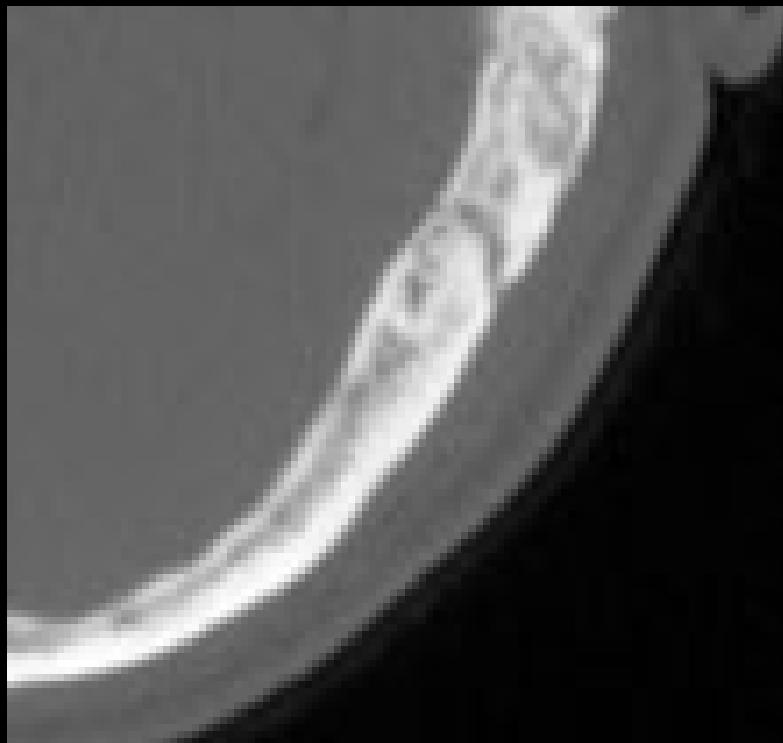
Compression ratio = uncompressed size / compressed size

# Lossless image formats

- Do not throw away information
- Good for storing medical images
  - We do not want to destroy any information
- Not very effective for photos. Why?
  - Too many changes in the image
- PNG (portable network graphics) is a good format

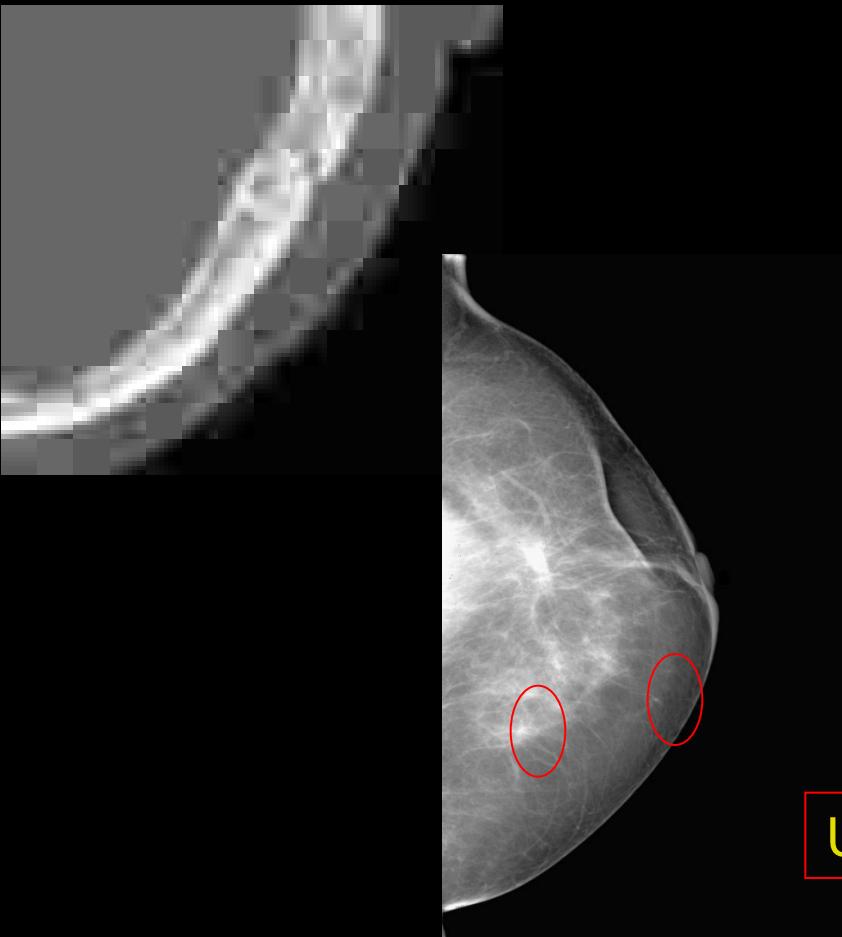


# Lossy image formats



- Removes “unimportant” information
- JPEG is an example
- Removes the “high frequencies”
- Similar to the MP3 sound format

# Compression artefacts



- Lossy compression changes the image
- Normally not a problem for photos
- BIG problem for medical images
- Mammogram
  - Looking for tiny bright spots
  - Would be changed by lossy compression

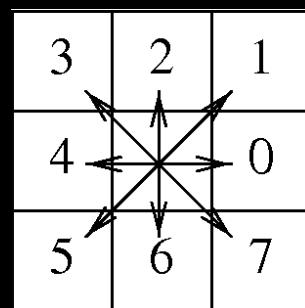
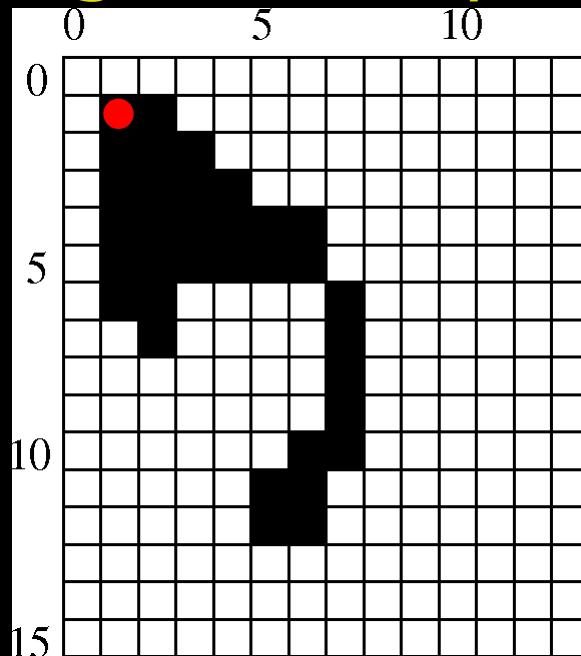
Use JPEG (JPG) for photos only

# Binary images



- Binary – means on or off
- Binary image – only two colors
- Background (0 = black)
- Foreground (1 = white)

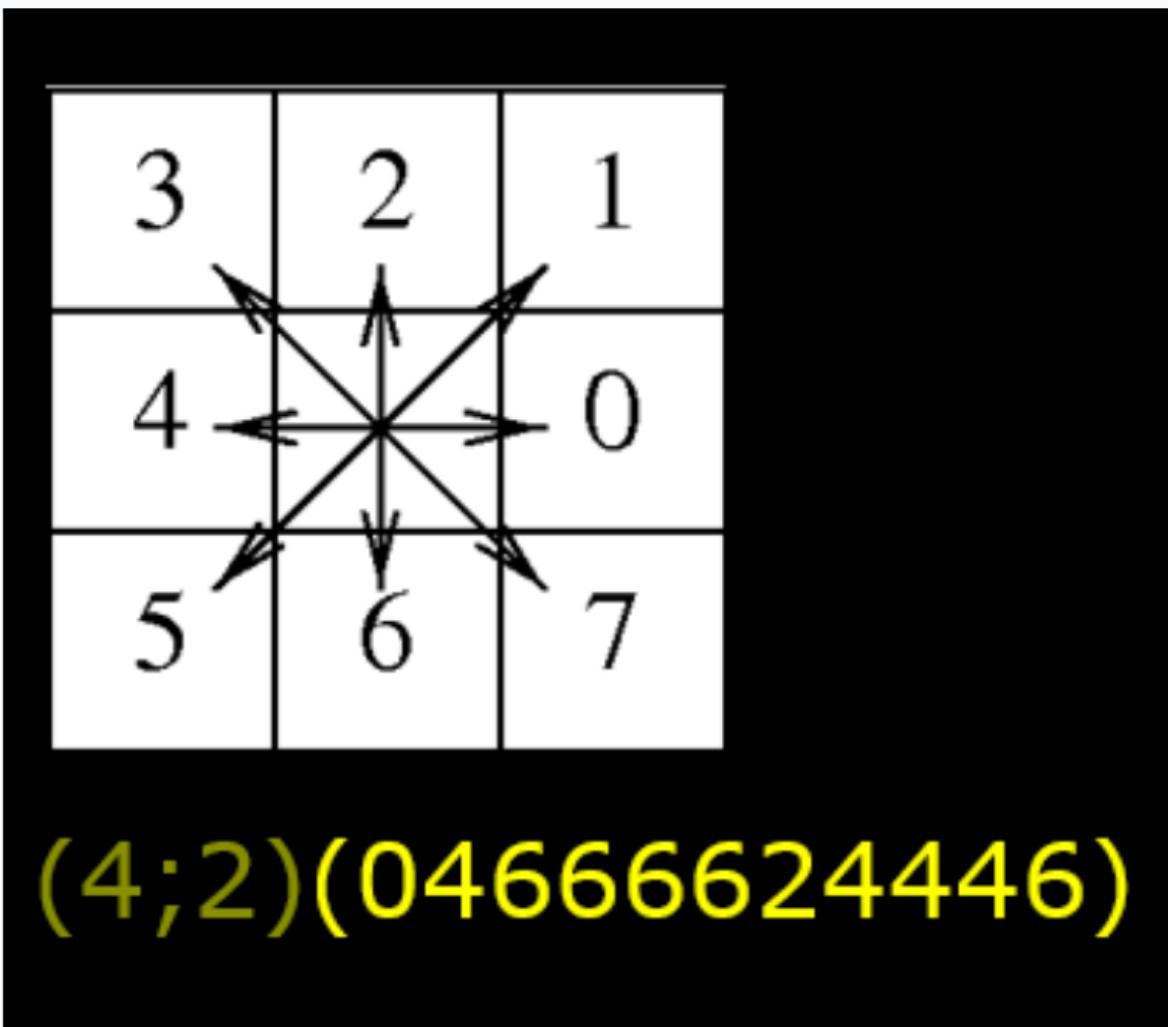
# Chain coding of binary images



(1; 1)(07770676666564211222344456322222)

- Sufficient to describe the foreground
- Background given by the foreground
- The coordinates of the starting pixel is stored
- Secondly the sequence of step directions is stored

## Chain code - what is in the image?



House

Flower

Giraffe

Dog

Teapot

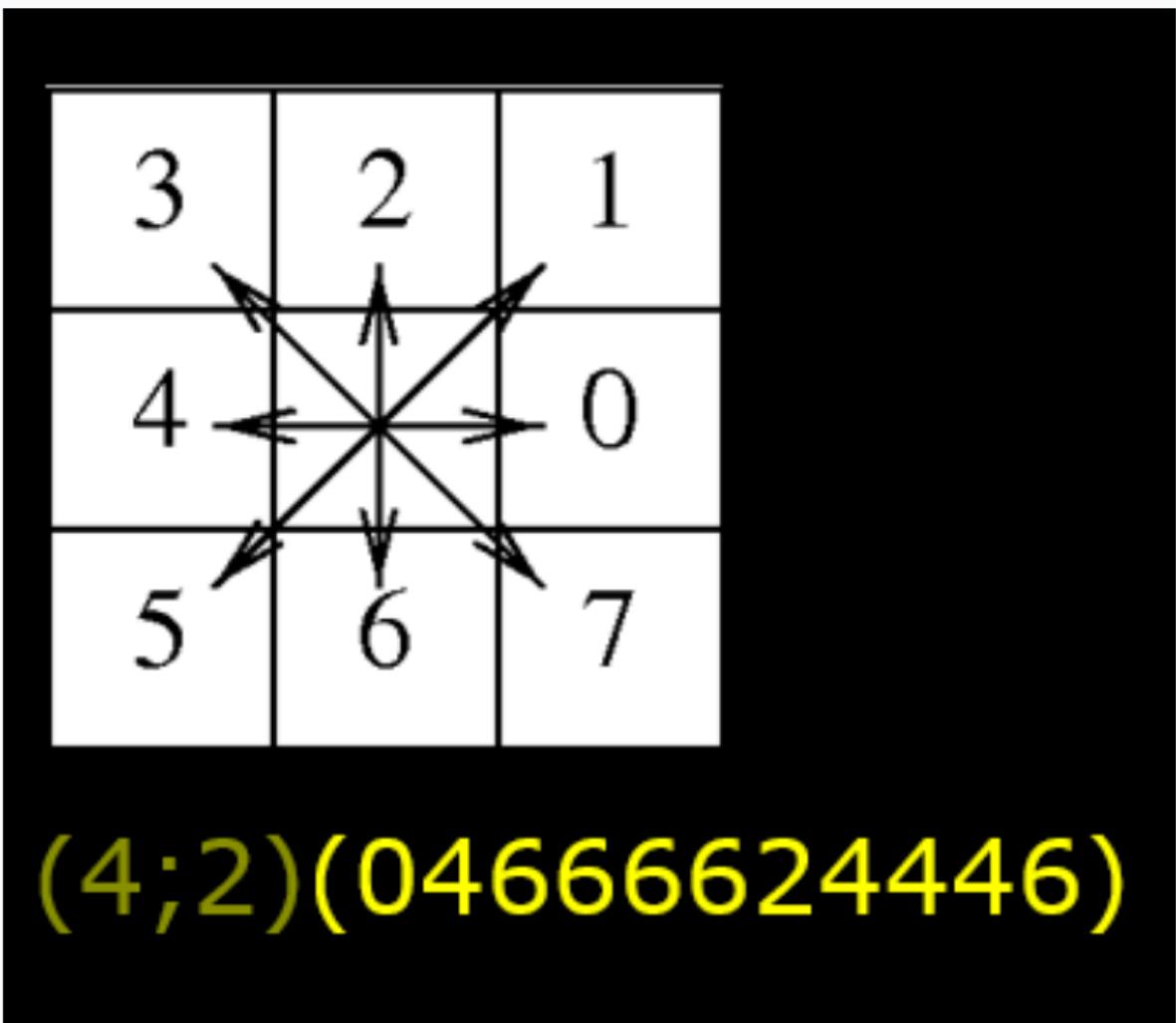
Car

Glass

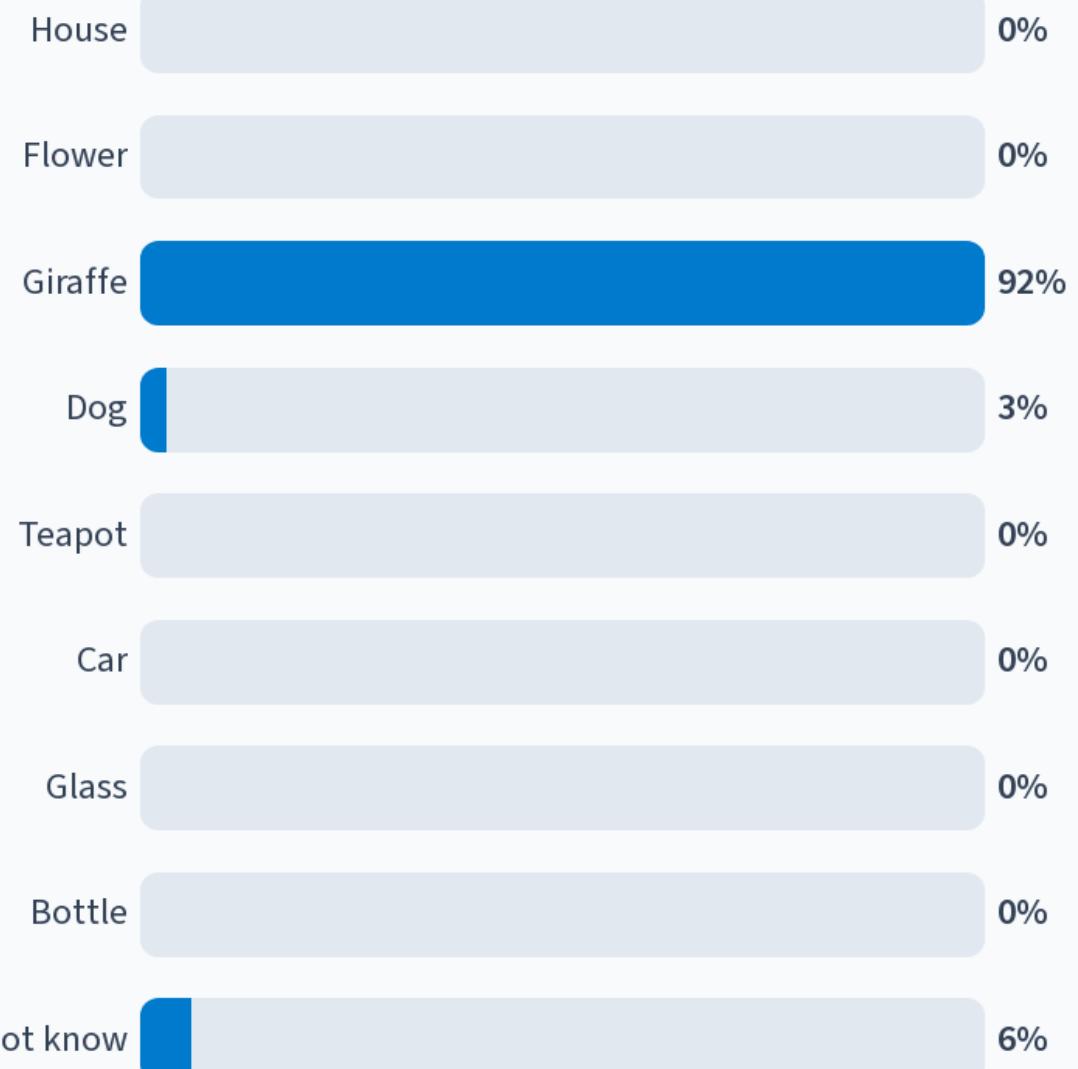
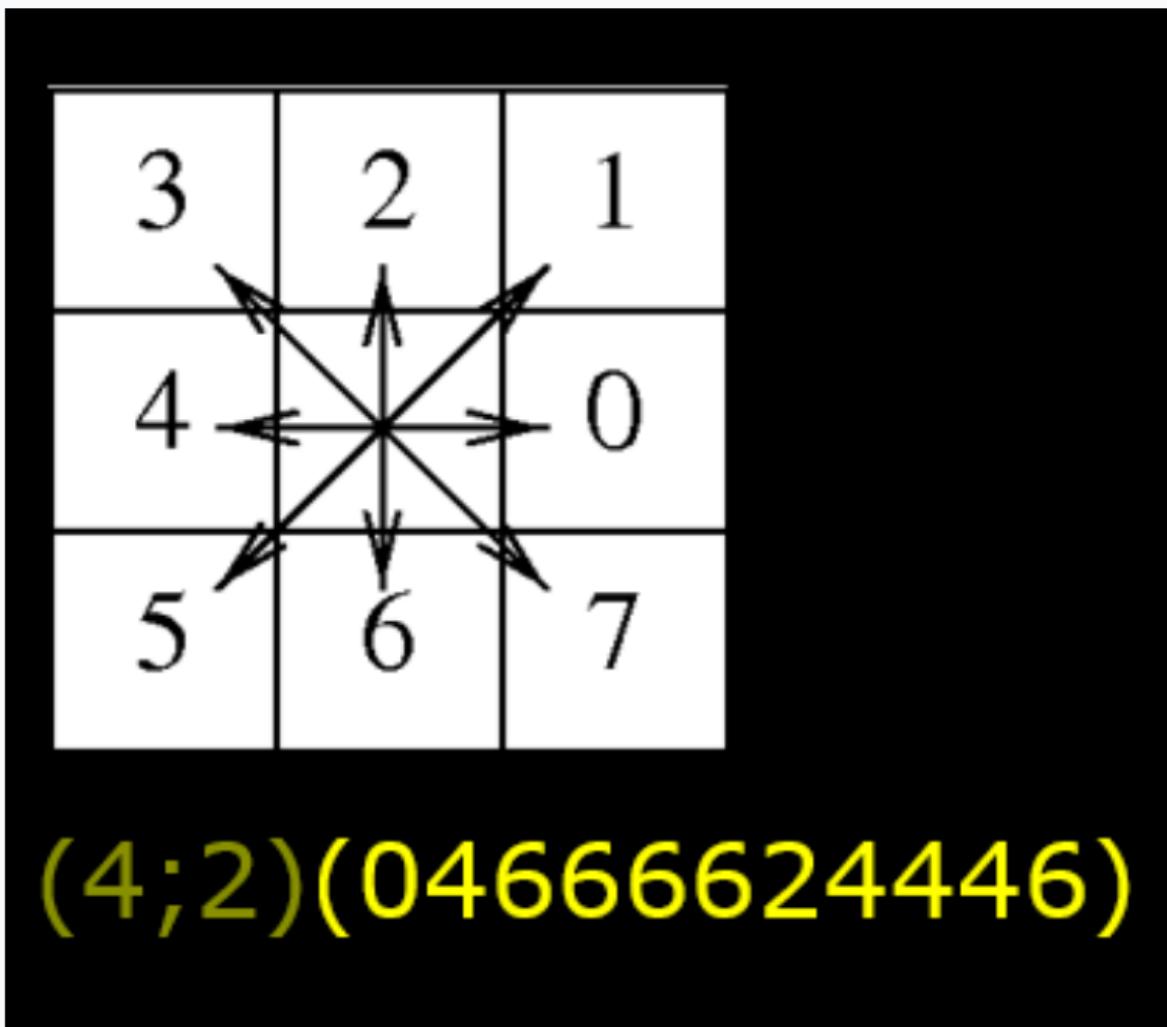
Bottle

I do not know

## Chain code - what is in the image?



## Chain code - what is in the image?

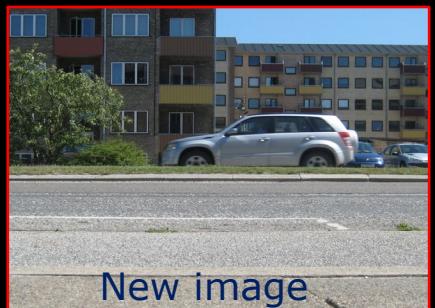


# Video Analysis



- Video – images coming in a stream from a camera
- Automated video analysis applications
  - Industrial / agricultural sorting machines
  - Activity alerts for surveillance cameras
  - Sports tracking
  - Self-driving vehicles
  - Driver awareness tracking / alerts
  - Space-ship navigation
  - Tracking of surgical instruments
  - And many more..

# Change detection in videos



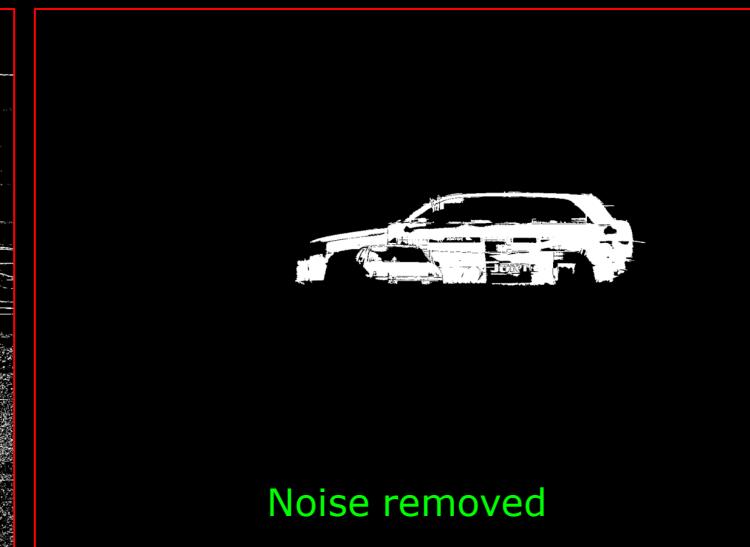
- Automatically detects changes in video stream
- The basis for many processing steps
  - Human pose tracking
  - Vehicle tracking
  - Alert systems
  - Cell tracking
  - ...

# Learning objectives – Video change detection

- Describe the concept of change detection
- Describe the camera, the processing and the total system frame rate
- Compute the maximum frame rate based on an algorithm processing time
- Compute a background/reference image when the scene is static or slowly changing
- Use pre-processing steps like color conversion and resizing to make images from a video stream ready to be analyzed
- Use image differencing to compute changes in a video stream
- Use background subtraction to compute changes in a video stream
- Use a threshold to create a binary image from a difference image
- Describe alternative approaches for background/reference image estimation
- Describe different scenarios where an action can be taken based on detected changes in a video stream

## Exercise 2b

- The goal of exercise 2b is to implement and test a small change detection system



# Cameras and videos – frame rate



- A camera delivers video in the forms of a stream of images (also called frames)
- The frame rate is the amount of frames per second.  
For example 20 frames/s (measured in Hz)
- For video processing we have two frame rates
  - How many frames can the camera deliver per second
  - How many frames can we analyze per second
- The *system frame* rate is the minimum of the camera and processing frame rate

## Camera frame rate

Your camera is attached to your computer using a USB-2 connection. On your system, the maximum transfer speed is 30 megabytes per second (MB/s). The images are RGB images (3 bytes per pixel) and their size is 640 x 480. What is the maximum camera frame rate on your system?

16

25

32

61

103

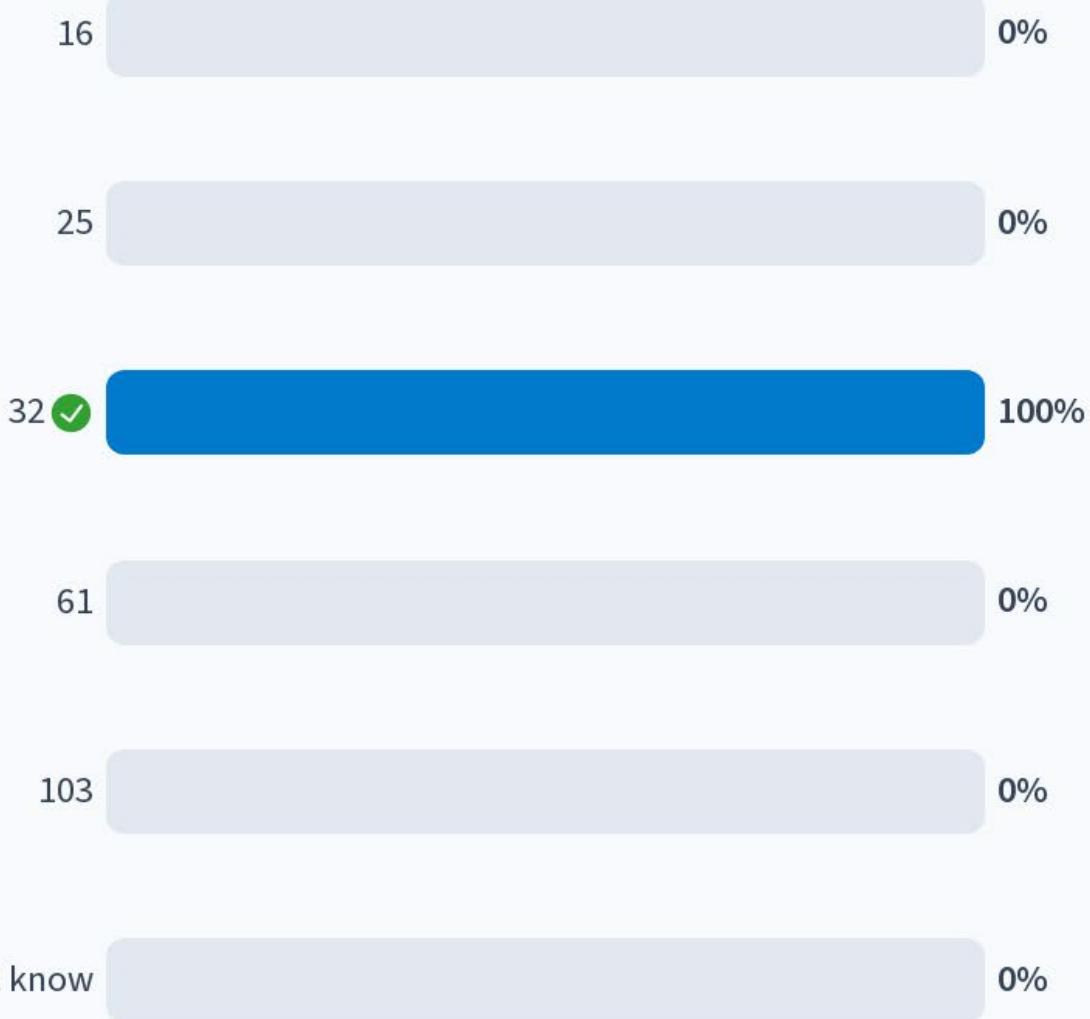
I do not know



Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)

## Camera frame rate

Your camera is attached to your computer using a USB-2 connection. On your system, the maximum transfer speed is 30 megabytes per second (MB/s). The images are RGB images (3 bytes per pixel) and their size is 640 x 480. What is the maximum camera frame rate on your system?



Your fancy image analysis algorithm uses 24 milliseconds to analyze one image. The camera delivers 60 frames per second. What is the maximum frame rate of your system?

12 Hz

27 Hz

38 Hz

41 Hz

67 Hz

I do not know

Your fancy image analysis algorithm uses 24 milliseconds to analyze one image. The camera delivers 60 frames per second. What is the maximum frame rate of your system?

12 Hz

0%

27 Hz

5%

38 Hz

0%

41 Hz

95%

67 Hz

0%

I do not know

0%

Your fancy image analysis algorithm uses 24 milliseconds to analyze one image. The camera delivers 60 frames per second. What is the maximum frame rate of your system?

12 Hz

0%

27 Hz

5%

38 Hz

0%

41 Hz

95%

67 Hz

0%

I do not know

0%

# How do we detect changes in a video stream?



## ■ One solution (out of many):

- Subtract the previous image from the current image and take the absolute value in each pixel
- *Image differencing*

## ■ Several drawbacks:

- Loses track of for example cars stopped for red light
- *Ghost differences*

# A change detection program

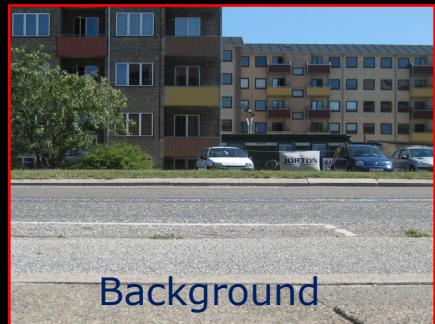
- Estimate and save a background/reference image
- Stop = False
- Do
  - Capture and pre-process one image
  - Compare with reference image (perhaps just subtraction)
  - Threshold difference image
  - Filter noise
  - Decide if something should be done
  - If 'q' key pressed:
    - Stop = True
- While not Stop

# A change detection program

- Estimate and save a background/reference image
- Stop = False
- Do
  - Capture and pre-process one image
  - Compare with reference image (perhaps just subtraction)
  - Threshold difference image
  - Filter noise
  - Decide if something should be done
  - If 'q' key pressed:
    - Stop = True
- While not Stop

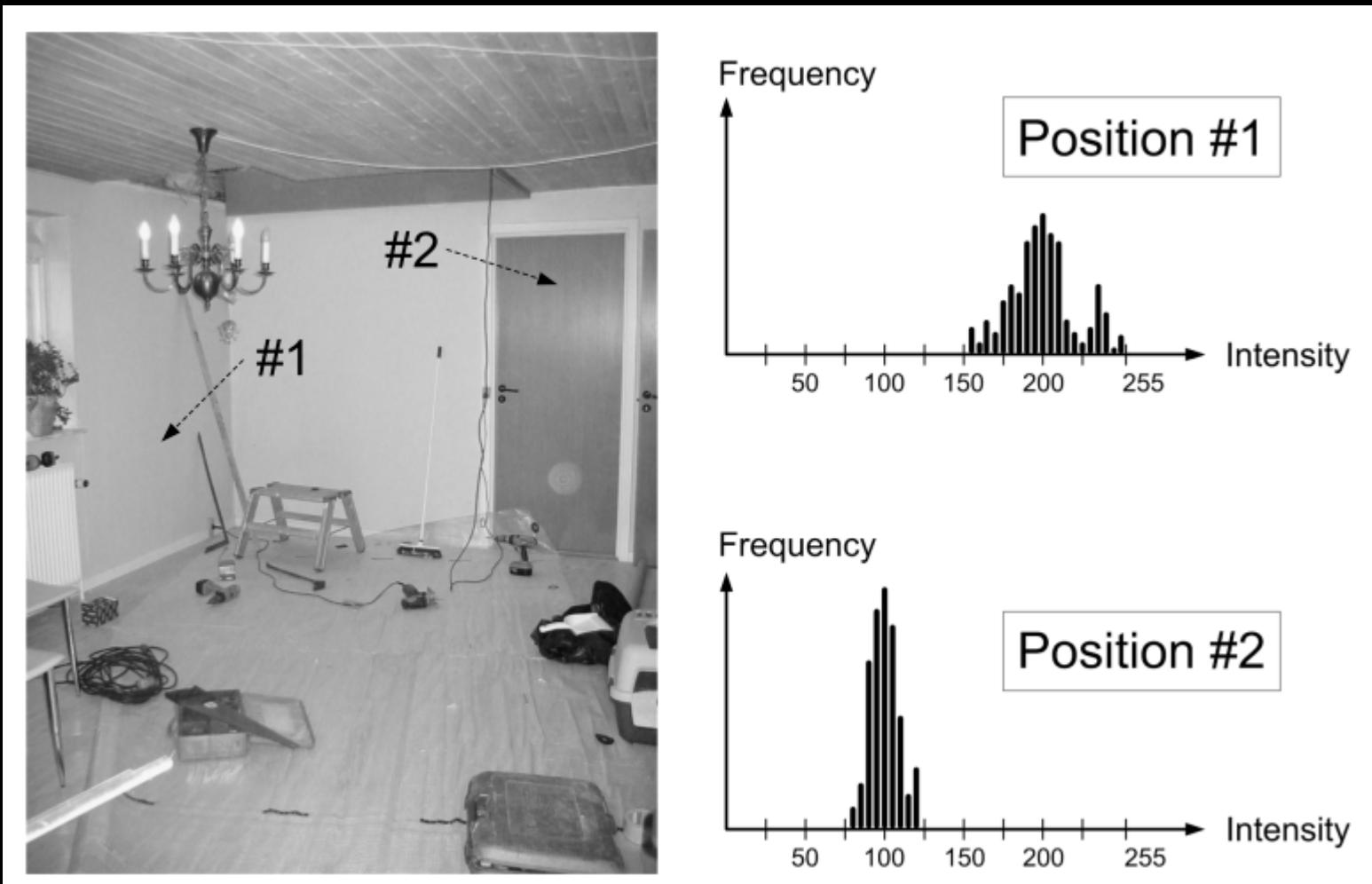
# Background estimation

- Estimate a robust background image that can be used to detect significant changes



- The scene can be more or less complicated
  - A static scene – very controlled light and no moving objects
  - Slowly changing scene – light changes due to the movement of the sun
  - Rapidly changing scene – Fast movement of leaves due to wind

# Naturally occurring changes



A camera is mounted on the parking space at building 101. The goal is to count the cars coming in and out. What could cause smaller rapid changes that should not be analyzed?

entrance) someone  
recognised cyclists direction  
enter area birds camera falling  
used clouds going weather biking  
traffic parking people (for pf simply  
pedestrians shadows leaf  
leafs buses walking bikes cars running  
outside change leaves daylight  
giraffes blowing rainsnow changes wanted  
nature darkness bicycles insects  
mosquitoes example regrets

# Background estimation – slowly changing scene



- Estimating a slowly changing background/reference image

```
ref_image = get_image_from_camera()  
stop = False  
alpha = 0.95  
  
do  
    new_image = get_image_from_camera()  
    old_ref = ref_image  
    ref_image = alpha * old_ref + (1 - alpha) * new_image  
    ...  
    ...(do something more)  
while not stop
```

# A change detection program

- Estimate and save a background/reference image
- Stop = False
- Do
  - Capture and pre-process one image
  - Compare with reference image (perhaps just subtraction)
  - Threshold difference image
  - Filter noise
  - Decide if something should be done
  - If 'q' key pressed:
    - Stop = True
- While not Stop

# Get new image and make it ready for processing



4032 x 3024

RGB (3 bytes per pixel)



4032 x 3024

Gray (1 byte per pixel)



640 x 480

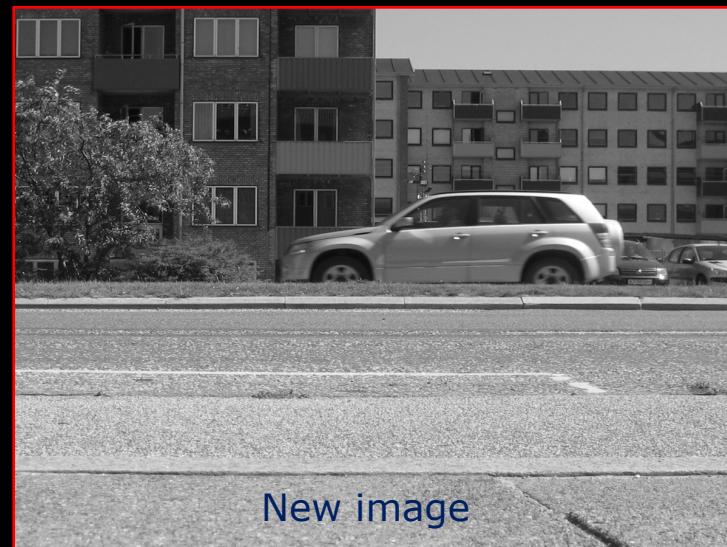
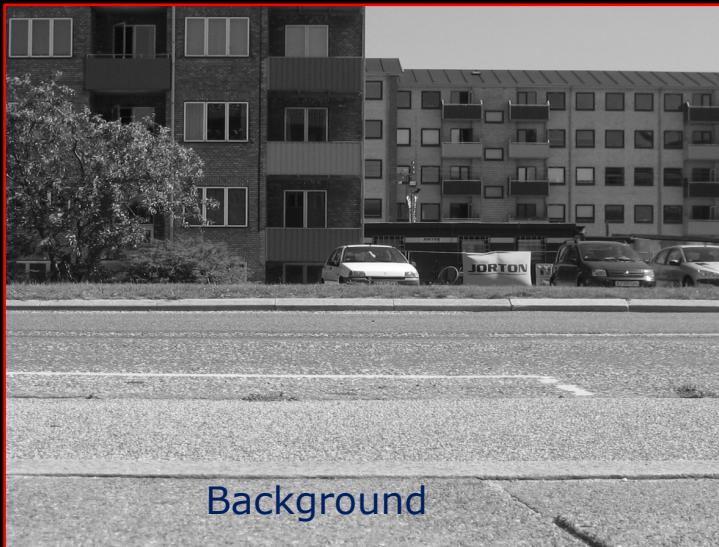
Gray (1 byte per pixel)

# A change detection program

- Estimate and save a background/reference image
- Stop = False
- Do
  - Capture and pre-process one image
  - Compare with reference image (perhaps just subtraction)
  - Threshold difference image
  - Filter noise
  - Decide if something should be done
  - If 'q' key pressed:
    - Stop = True
- While not Stop

# Compare with reference image

- Simplest approach:
  - Absolute difference between background and new image
- More advanced approaches based on pixel-wise statistics exists



# A change detection program

- Estimate and save a background/reference image
- Stop = False
- Do
  - Capture and pre-process one image
  - Compare with reference image (perhaps just subtraction)
  - Threshold difference image
  - Filter noise
  - Decide if something should be done
  - If 'q' key pressed:
    - Stop = True
- While not Stop

# Threshold difference image

- Identify the pixel that have significantly changed
  - Set a threshold,  $T$ , in the difference image
  - Pixels with a value higher than the threshold is set to 1 the rest to 0
- Complicated to choose the correct threshold



Absolute difference



Thresholded image (Binary)

# A change detection program

- Estimate and save a background/reference image
- Stop = False
- Do
  - Capture and pre-process one image
  - Compare with reference image (perhaps just subtraction)
  - Threshold difference image
  - Filter noise
  - Decide if something should be done
  - If 'q' key pressed:
    - Stop = True
- While not Stop

# Remove noise from binary image

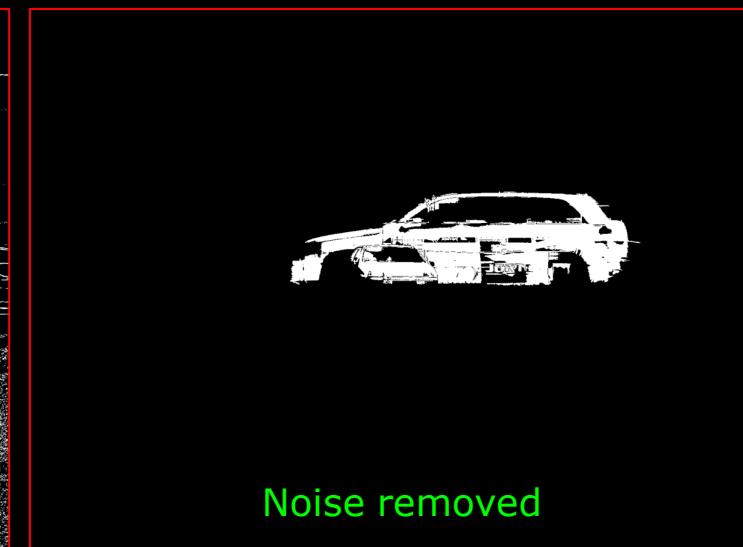
- Remove pixels that can be considered noise
  - Isolated pixels
  - Pixels in small groups
- Filtering, morphological operations, BLOB analysis – more about this later in the course



Absolute difference



Thresholded image (Binary)



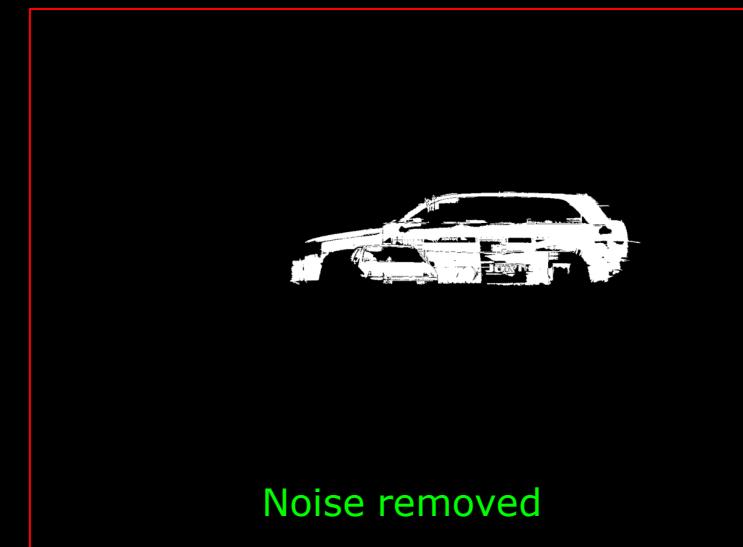
Noise removed

# A change detection program

- Estimate and save a background/reference image
- Stop = False
- Do
  - Capture and pre-process one image
  - Compare with reference image (perhaps just subtraction)
  - Threshold difference image
  - Filter noise
  - Decide if something should be done
  - If 'q' key pressed:
    - Stop = True
- While not Stop

# Decide if something should be done?

- Depends on the application and the scene
- Certain percentage of the total amount of pixel have changed
  - Sound an alarm?
- The changed pixels has the same size and shape as a car
  - Tell that a car is here or start analyzing the car
- The changed pixels look like a face or a person
  - Recognize the face
  - Track the human
- ...



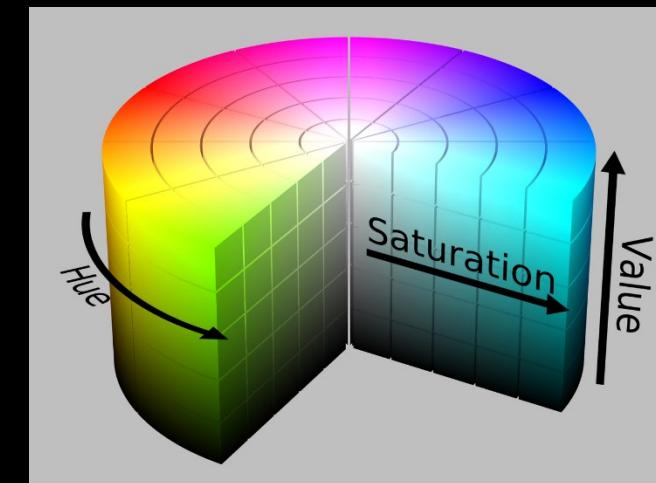
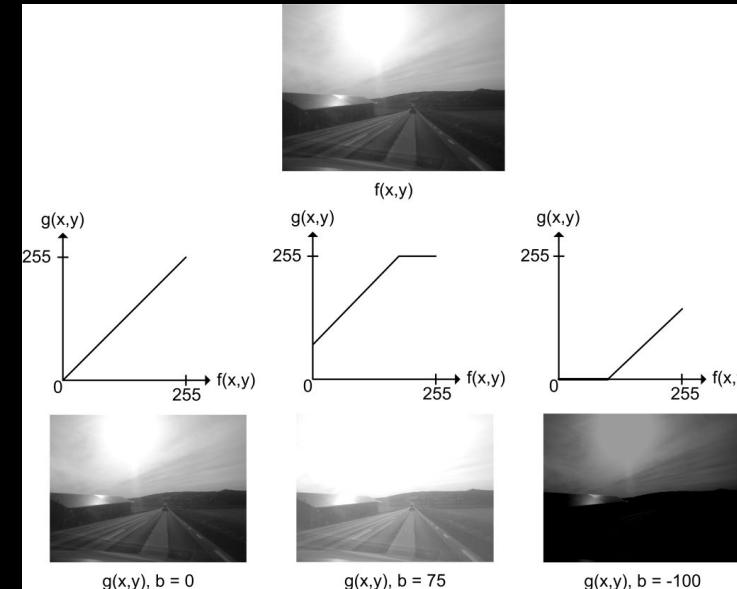
# Advanced change detection techniques



- Active research and development for 30+ years
- Advanced reference image estimation
  - Pixel wise multi-class estimation
  - Statistical testing per-pixel to detect changes
  - Other color spaces
  - ...

# Next week

- Pixel wise operations
- Colour images
- PCA on images





# Image Analysis

Rasmus R. Paulsen

Tim B. Dyrby

DTU Compute

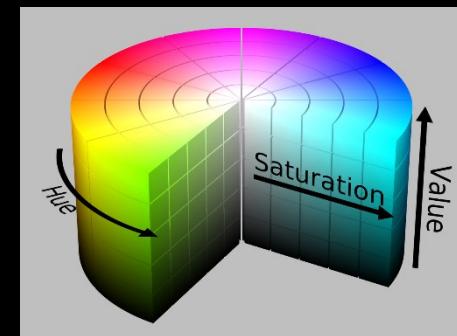
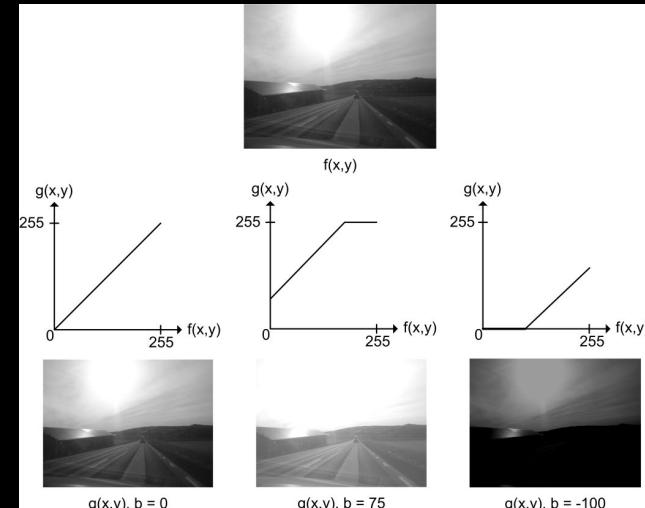
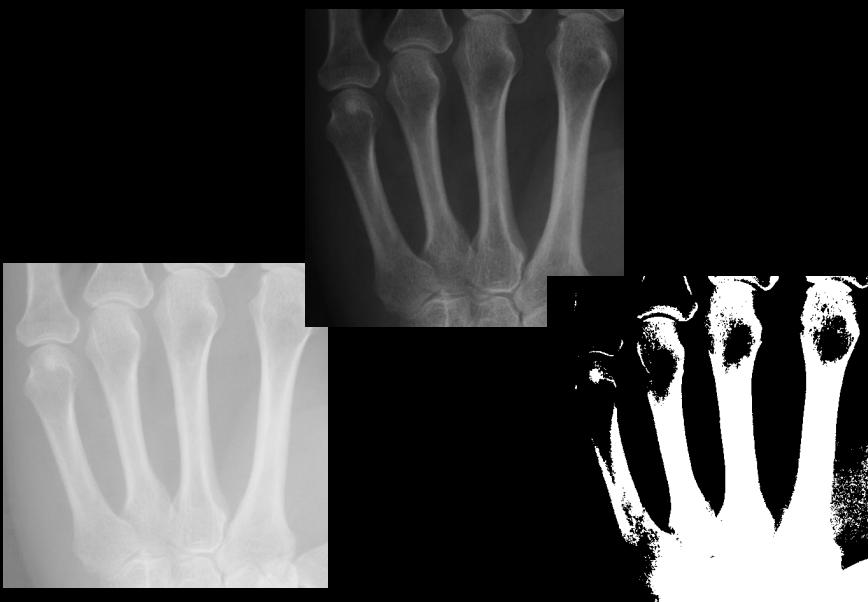
<http://courses.compute.dtu.dk/02502>

Plenty of slides adapted from Thomas Moeslunds lectures

# Week 3

## Pixelwise operations and colour images

### PCA on images



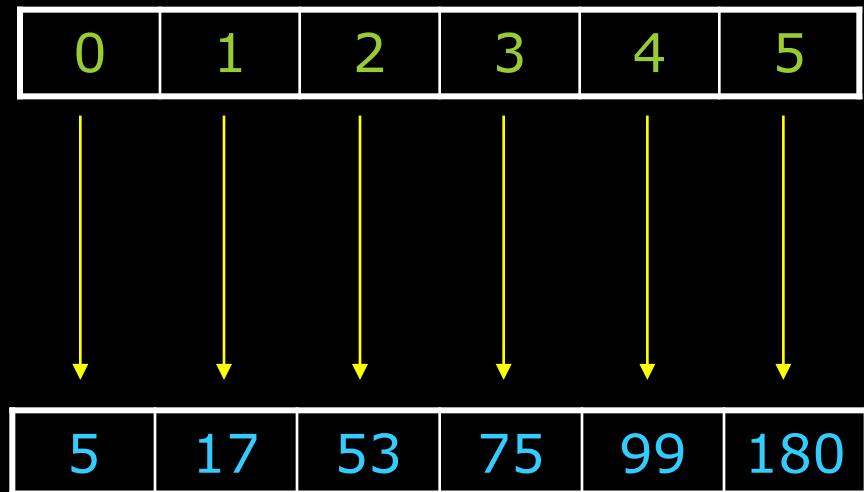
# What can you do after today?

- Compute and apply a linear gray transformation
- Describe and compute the image histogram
- Implement and apply histogram stretching
- Implement and apply gamma transformation
- Implement and apply log and exp mappings
- Describe and use thresholding
- Describe and use automatic thresholding
- Perform conversions between bytes and doubles
- Use addition and subtraction of images
- Explain the benefits of bi-modal histograms
- Identify images where global thresholding can be used for object extraction

# ...and you can even more

- Describe the basic human visual system including rods and cones
- Describe subtractive colors
- Describe additive colors
- Describe the RGB color space
- Describe the normalised RGB color representation
- Describe the use of the Bayer pattern in digital cameras
- Describe the HSI color space
- Convert from an RGB to a grey level value
- Convert from an RGB value to an HSI value
- Describe the use of different color spaces
- Implement and use color thresholding in RGB space
- Implement and use color thresholding in HSI space

# Gray value mappings



- Mapping
  - To make correspondence between two sets of values
- Look-up-table
  - A table of mappings



### Mapping Function

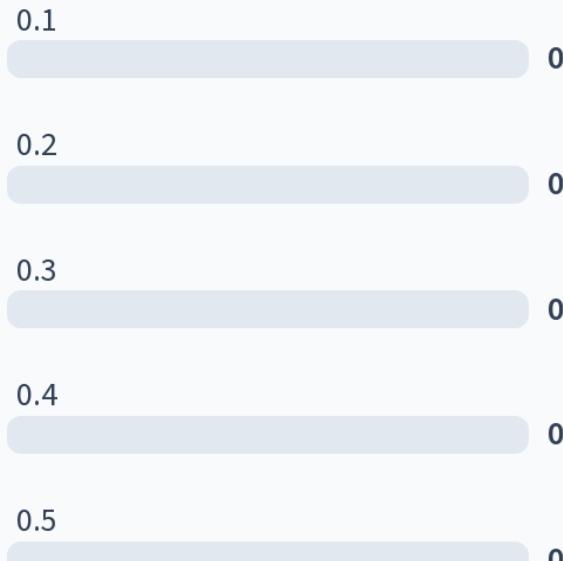
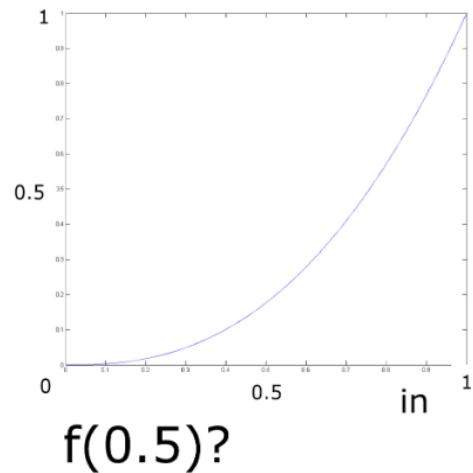
A graph showing a mapping function  $f$  from the interval  $[0, 1]$  to itself. The x-axis is labeled "in" and the y-axis has numerical ticks from 0 to 1. The curve starts at (0, 0), passes through approximately (0.2, 0.05), (0.4, 0.15), (0.6, 0.35), (0.8, 0.75), and ends at (1, 1). The curve is concave up, indicating an increasing function.

$f(0.5)?$

- 0.1
- 0.2
- 0.3
- 0.4
- 0.5

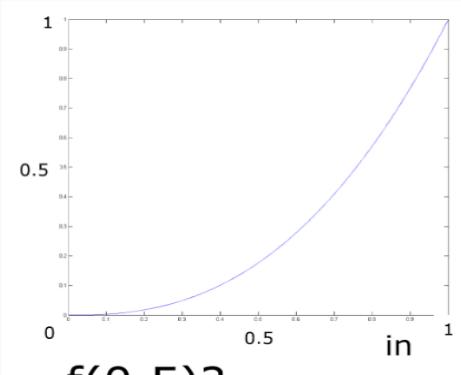
Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)

## Mapping Function



Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)

### Mapping Function



$f(0.5)?$

0.1 0

0.2 0

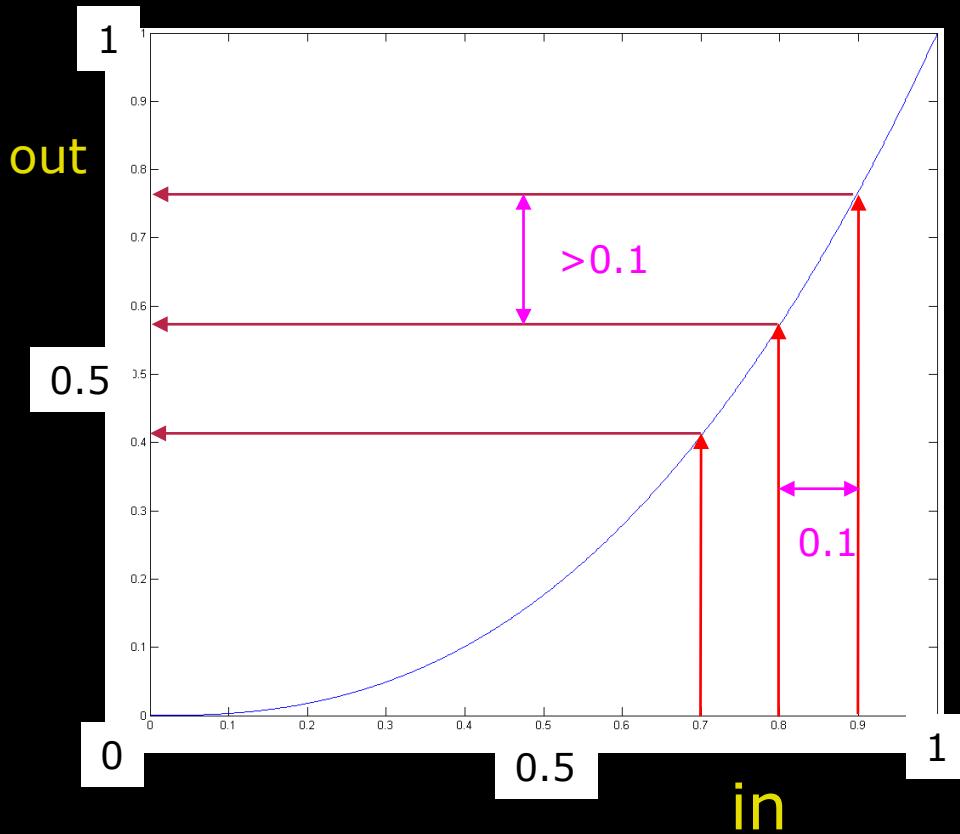
0.3 0

0.4 0

0.5 0

Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)

# Gray value mappings



- Mapping
  - To make correspondence between two sets of values
- Mapping function
  - $\text{out} = f(\text{in})$
- What happens with the values?
  - Values with difference 0.1
  - Output values “spread out”



When is it a good idea to change pixel values and how will it change the image?

Nobody has responded yet.

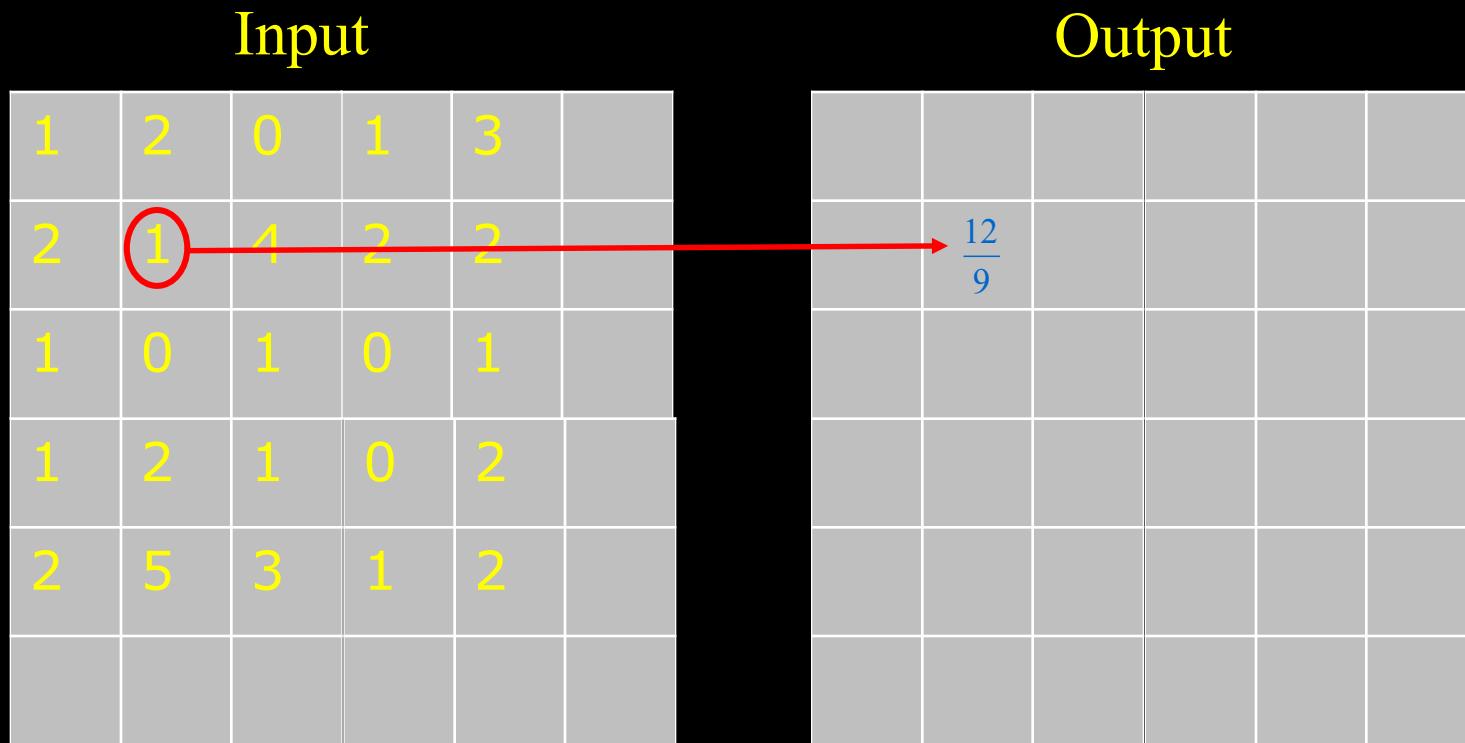
Hang tight! Responses are coming in.

Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)

# Why change gray level values

- When could it be good to change the gray level values?
  - Lack of contrast
  - Very dark image
  - Very bright image

# Point processing



- The value of the output pixel is only dependent on the value of one input pixel
- A global operation – changes all pixels

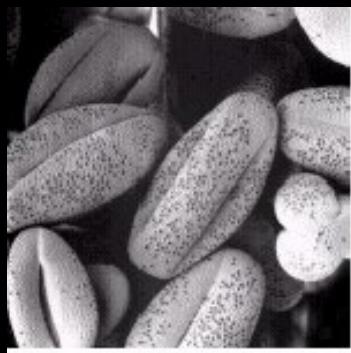
# Point processing

## ■ Grey level enhancement

- Process one pixel at a time independent of all other pixels
- For example used to correct Brightness and Contrast
  - Known from the television remote control



Correct



Too high  
brightness



Too low  
brightness



Too high  
contrast



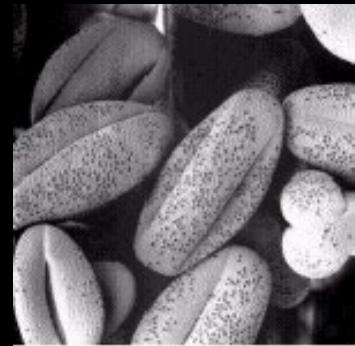
Too low  
contrast



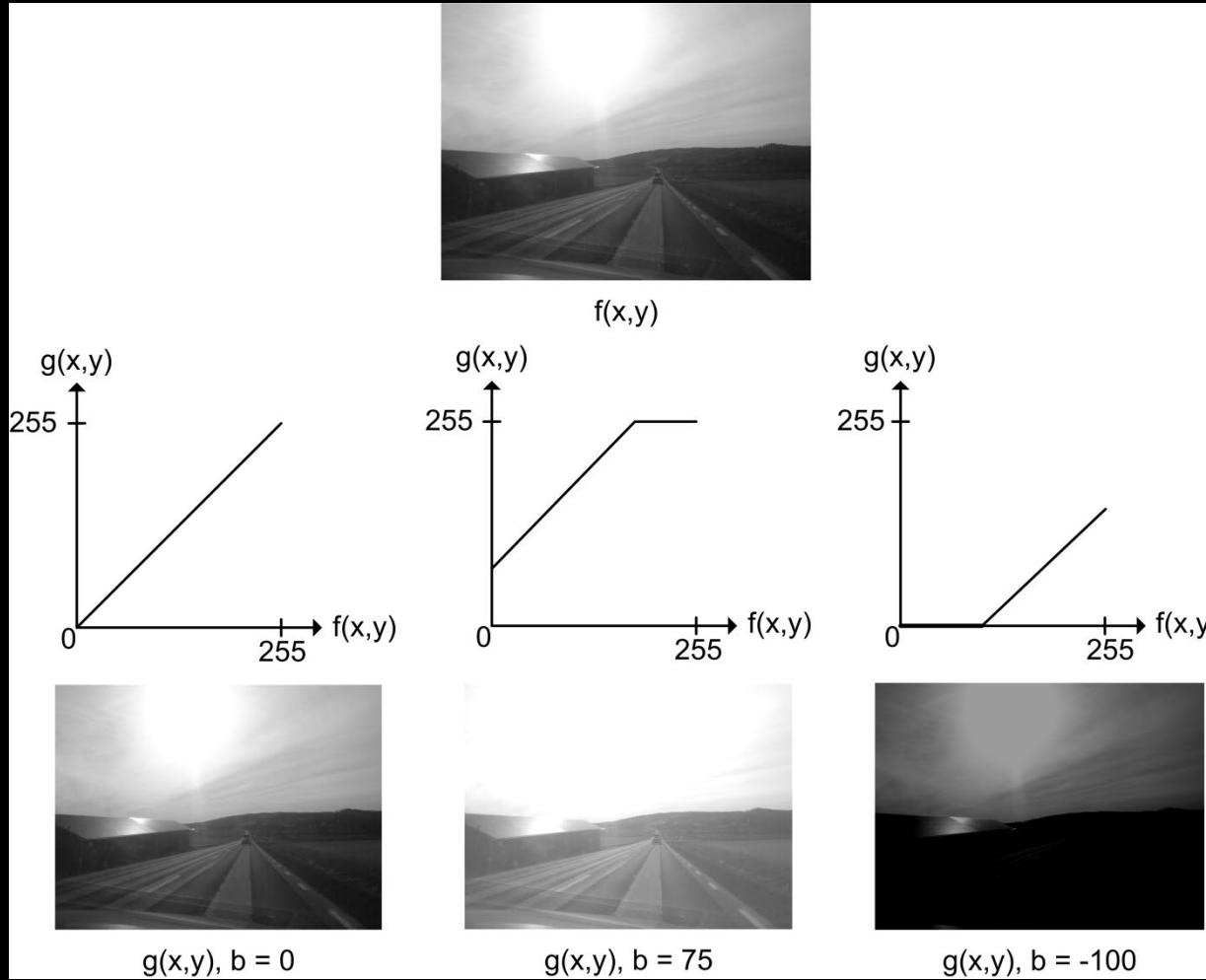
# Brightness

- The brightness is the intensity
- Change brightness:
  - To each pixel is added the value  $b$
  - $f(x, y)$  is the input image
  - $g(x, y)$  is the (enhanced) output image
- If  $b > 0$  : brighter image
- If  $b < 0$  : less bright image

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

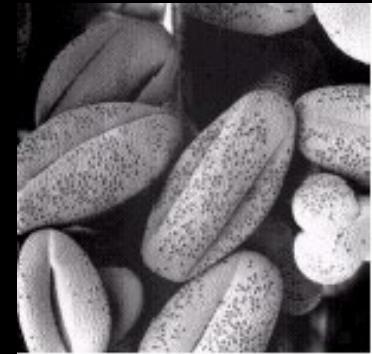


# Brightness



# Contrast

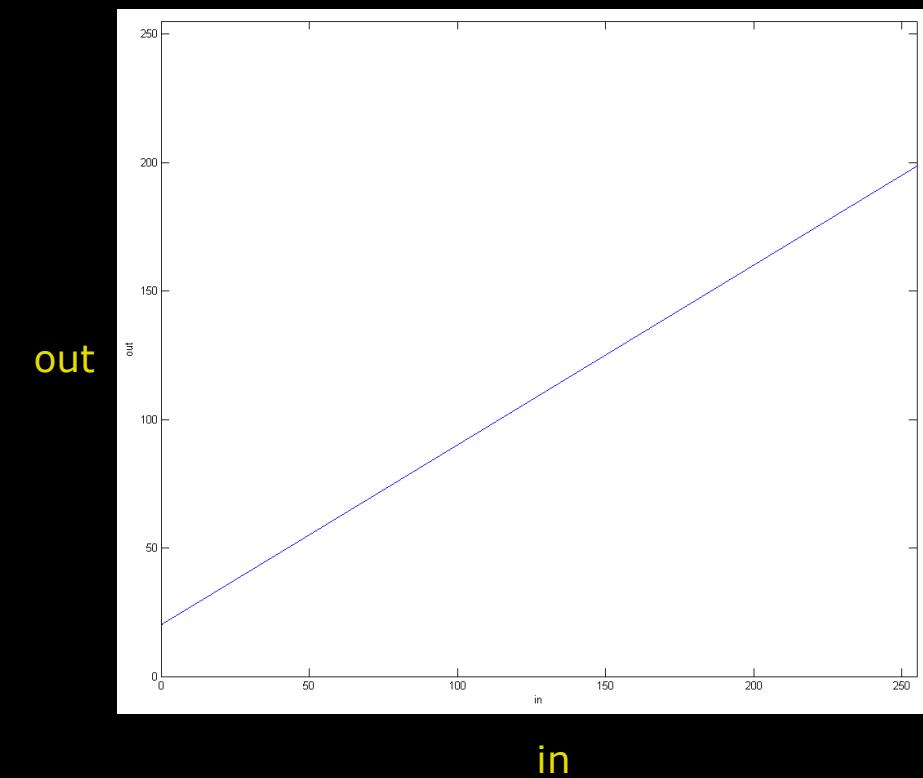
- The contrast describes the level of details we can see
- Change contrast
- Each pixel is multiplied by  $a$ 
  - $f(x, y)$  is the input image
  - $g(x, y)$  is the (enhanced) output image
- If  $a > 1 \Rightarrow$  more contrast
- If  $a < 1 \Rightarrow$  less contrast



$$g(x, y) = a * f(x, y)$$

# Combining brightness and contrast

- A straight line
- Called a *linear transformation*
- Here  $a = 0.7$  and  $b = 20$

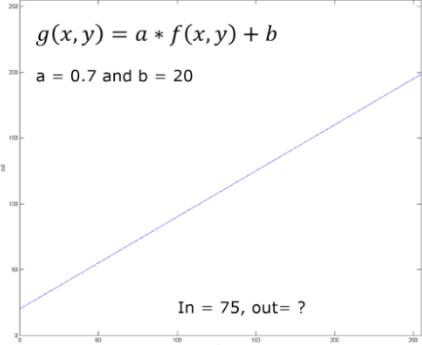


$$g(x, y) = a * f(x, y) + b$$

Linear Transformation

$$g(x, y) = a * f(x, y) + b$$

a = 0.7 and b = 20



In = 75, out= ?

20

45

72

103

230

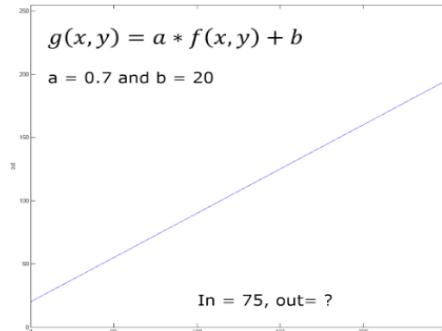
Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)

### Linear Transformation

$g(x, y) = a * f(x, y) + b$

$a = 0.7$  and  $b = 20$

In = 75, out= ?



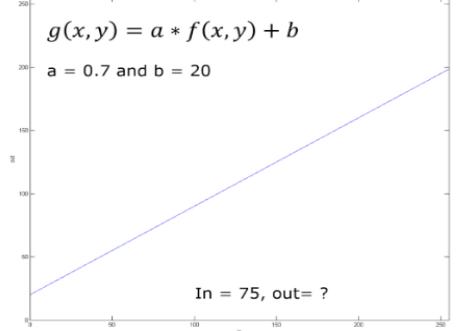
20	0
45	0
72	0
103	0
230	0

Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)



### Linear Transformation

$g(x, y) = a * f(x, y) + b$   
 $a = 0.7$  and  $b = 20$



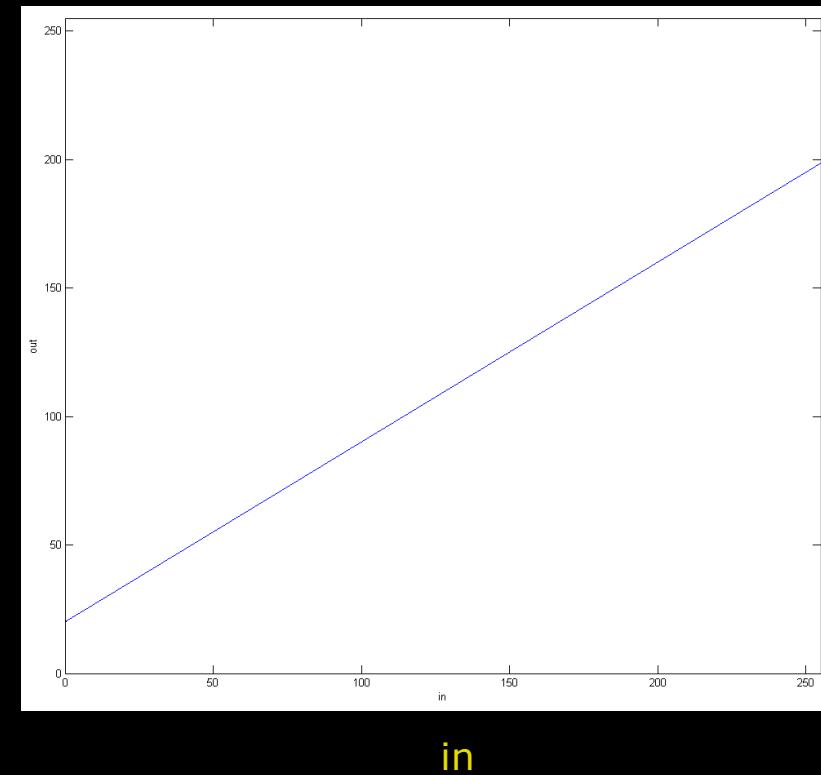
A graph showing a linear transformation. The x-axis and y-axis both range from 0 to 250. A blue line starts at approximately (0, 20) and passes through (250, 200). Text on the graph reads:  $g(x, y) = a * f(x, y) + b$ ,  $a = 0.7$  and  $b = 20$ , and  $In = 75, out = ?$ .

Value	0
20	20
45	45
72	72
103	103
230	230

Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)

# Combining brightness and contrast

- A straight line
- Called a *linear transformation*
- Here  $a = 0.7$  and  $b = 20$
- What will the visual result be on the output image?
  - More bright ( $b > 0$ )
  - Less contrast ( $a < 1$ )

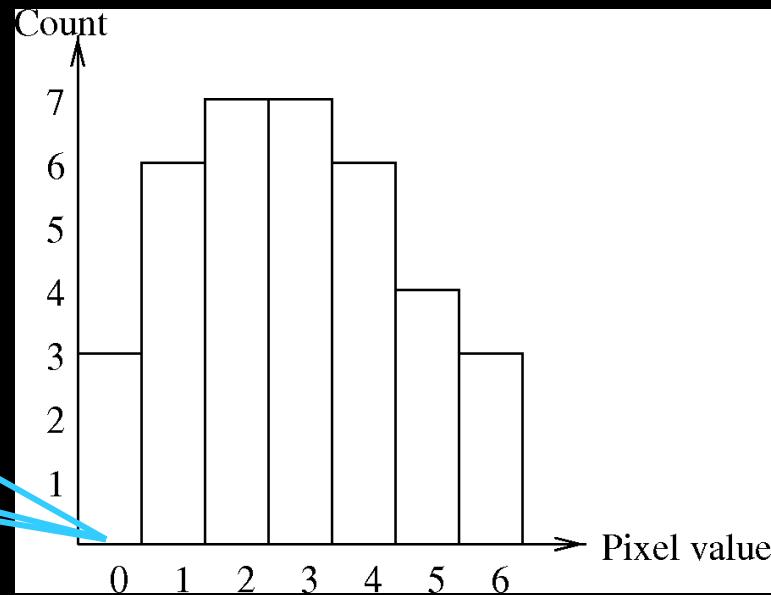


$$g(x, y) = a * f(x, y) + b$$

# Histogram Reminder

- A histogram normally contains the same number of “bins” as the possible pixel values
- A bin stores the number of pixel with that value

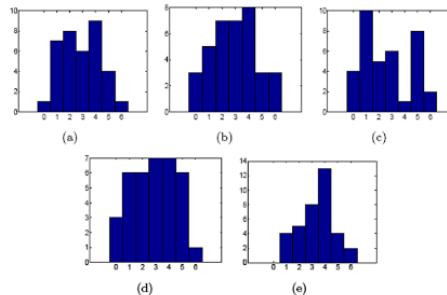
0	2	6	6	3	3
1	4	3	4	4	4
3	2	5	1	5	2
1	4	2	1	3	1
2	5	3	0	2	0
4	2	5	6	3	1



Choose the histogram that represents the image

0	5	3	5	2	1
3	5	5	3	3	1
1	1	1	3	2	3
6	2	2	1	0	0
0	2	1	5	1	5
5	5	1	4	1	6

Figur 6: Grayscale billede.



A

B

C

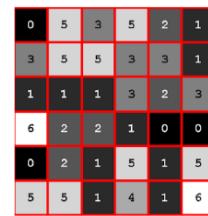
D

None of the above

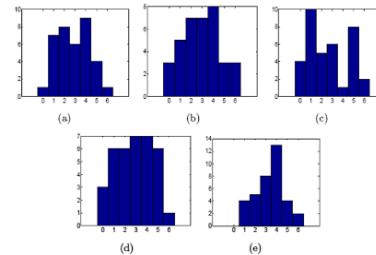
Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)



Choose the histogram that represents the image



Figur 6: Grayscale billede.



A

B

C

D

None of the above

0

0

0

0

0

Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)

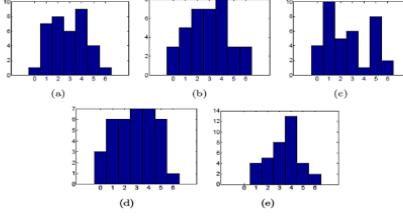
Choose the histogram that represents the image

Grayscale image:

0	5	3	5	2	1
3	5	5	3	3	1
1	1	1	3	2	3
6	2	2	1	0	0
0	2	1	5	1	5
5	5	1	4	1	6

Figur 6: Grayscale billedie.

Histograms:



(a) (b) (c)

(d) (e)

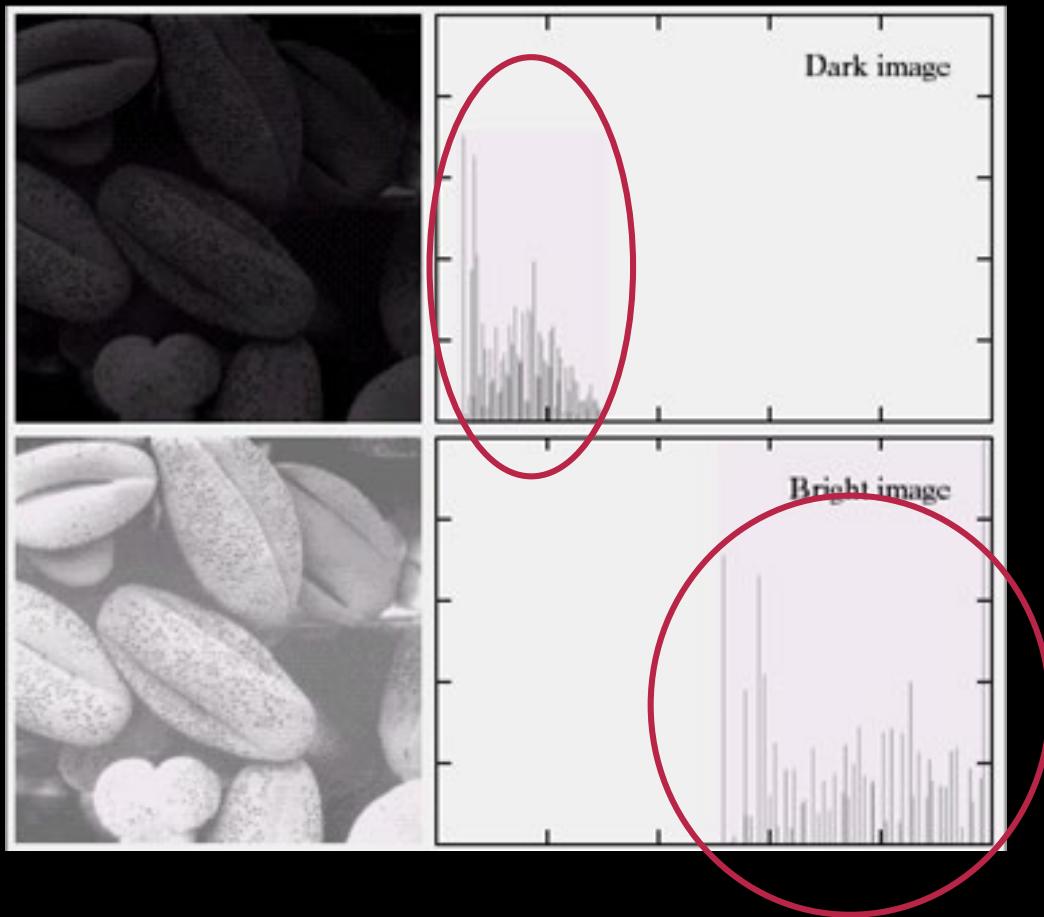
A  
B  
C  
D  
None of the above

Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)

# Back to the histogram

- The shape of the histogram tells us a lot!

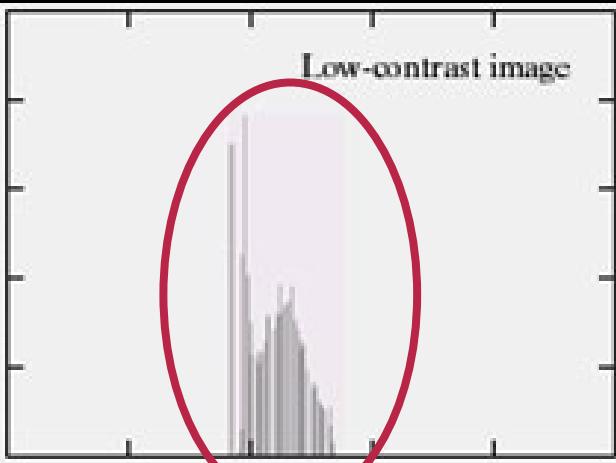
# Histogram inspection



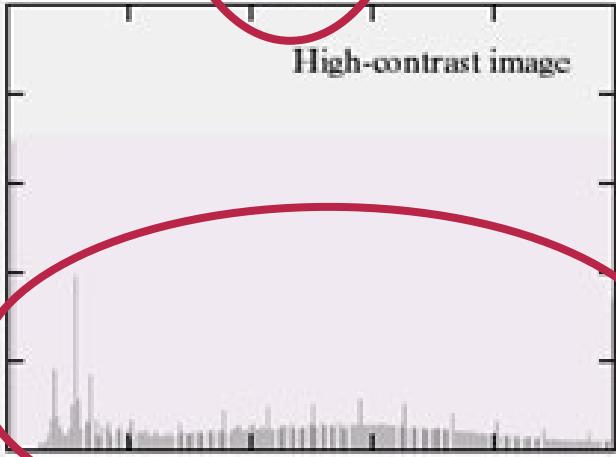
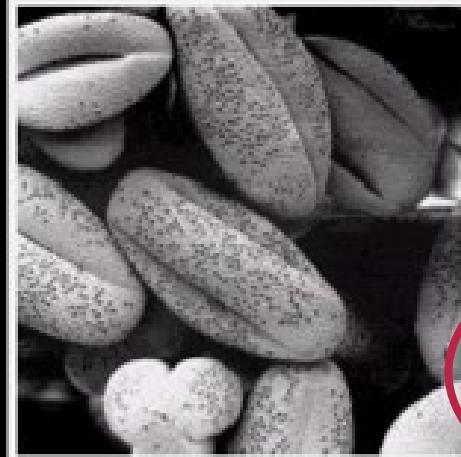
Dark image

Bright image

# Histogram inspection

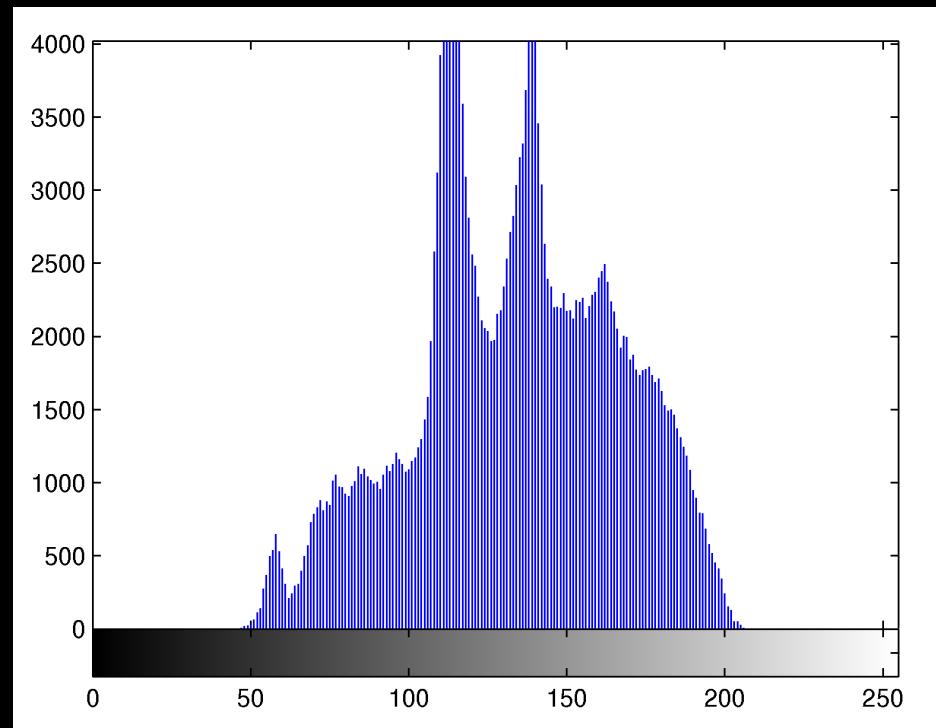


Low contrast



High contrast

# Histogram stretching



- How do we optimise the image using the histogram?
  - Minimum and maximum values?
  - Stretch it so new minimum = 0 and new maximum = 255

### Histogram stretching

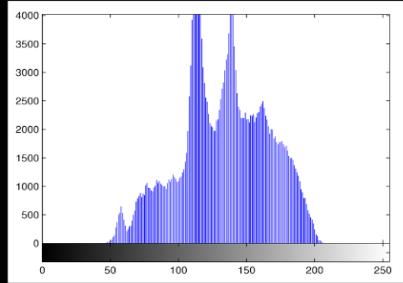
- We want
  - Min = 0
  - Max = 255
- We have
  - Min = 32
  - Max = 208

Using brightness

Using contrast

Using brightness and contrast

None of the above



Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)



### Histogram stretching

■ We want  
- Min = 0  
- Max = 255

■ We have  
- Min = 32  
- Max = 208

Using brightness 0%

Using contrast 0%

Using brightness and contrast 0%

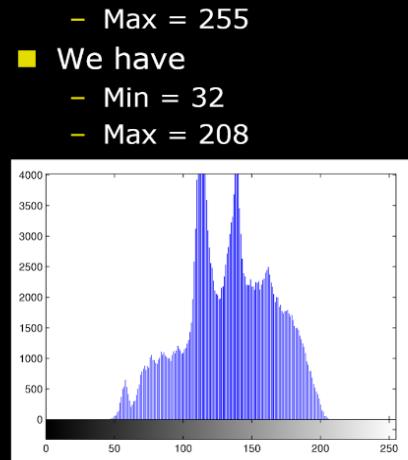
None of the above 0%

Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)

### Histogram stretching

■ We want  
– Min = 0  
– Max = 255

■ We have  
– Min = 32  
– Max = 208



Using brightness  0%

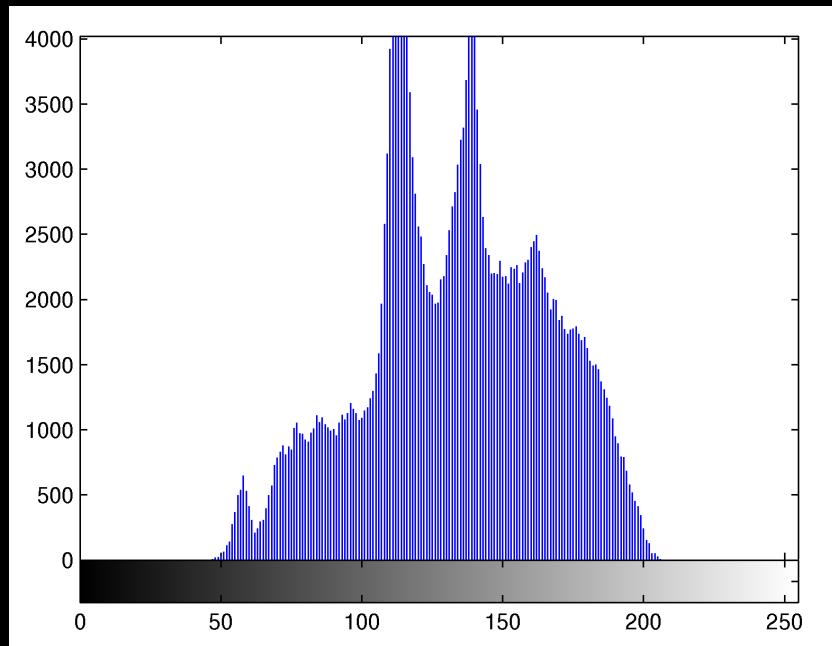
Using contrast  0%

Using brightness and contrast  0%

None of the above  0%

Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)

# Histogram stretching



- We want
  - Min = 0
  - Max = 255
- We have
  - Min = 32
  - Max = 208

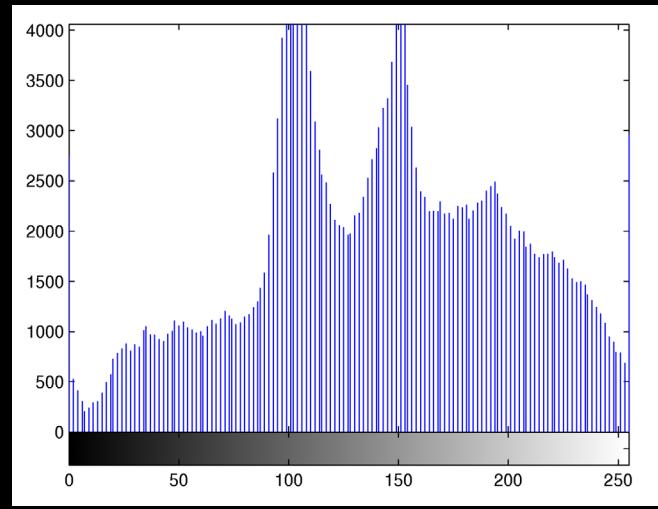
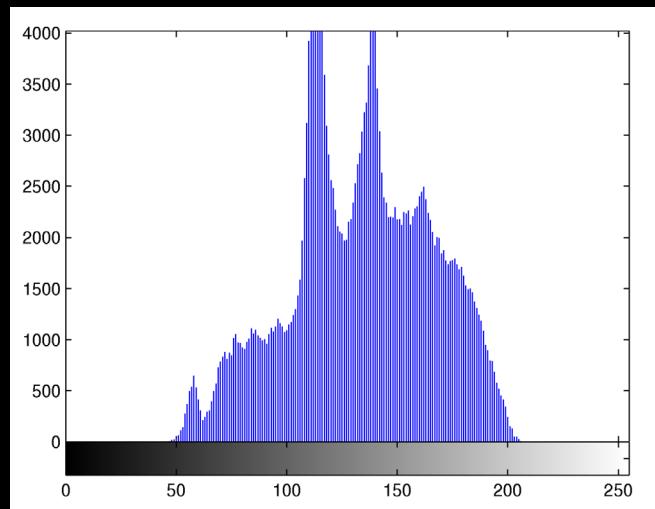
$$g(x, y) = \frac{v_{max,d} - v_{min,d}}{v_{max} - v_{min}} (f(x, y) - v_{min}) + v_{min,d}$$

# Histogram stretching formula

$$g(x, y) = \frac{v_{max,d} - v_{min,d}}{v_{max} - v_{min}} (f(x, y) - v_{min}) + v_{min,d}$$

- Desired min value       $v_{min,d} = 0$
- Desired max value       $v_{max,d} = 255$
- Current min value       $v_{min} = 32$
- Current max value       $v_{max} = 208$

# Histogram stretching

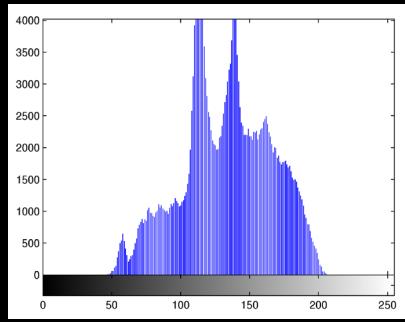


$$g(x, y) = \frac{255}{176} (f(x, y) - 32)$$

# Effect of histogram stretching



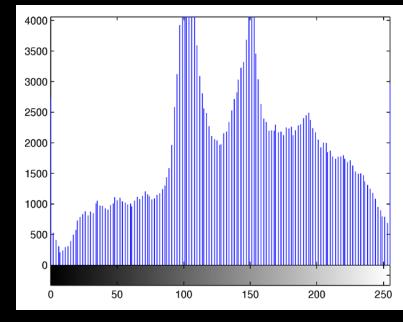
32



208



0



255

# Histogram stretching – weaknesses

- A single pixel value of 0 or 255 ruins it
- Sometimes you want
  - To stretch only the high pixel values
  - While “compressing” the low pixel values
  - Non-linear mapping



Linear mapping on an image

A linear mapping is performed on the image below. The mapping is performed so the mapped image has a maximum value of 255 and a minimum value of 0. What is the new value in the marked pixel?

208	25	40	36	167
231	71	23	108	18
32	139	244	234	217
233	244	124	202	238
161	245	204	245	173

Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)

95

111

98

119

101



Linear mapping on an image

A linear mapping is performed on the image below. The mapping is performed so the mapped image has a maximum value of 255 and a minimum value of 0. What is the new value in the marked pixel?

208	25	40	36	167
231	71	23	108	18
32	139	244	234	217
233	244	124	202	238
161	245	204	245	173

95      0

111      0

98      0

119      0

101      0

Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)

### Linear mapping on an image

A linear mapping is performed on the image below. The mapping is performed so the mapped image has a maximum value of 255 and a minimum value of 0. What is the new value in the marked pixel?

208	25	40	36	167
231	71	23	108	18
32	139	244	234	217
233	244	124	202	238
161	245	204	245	173

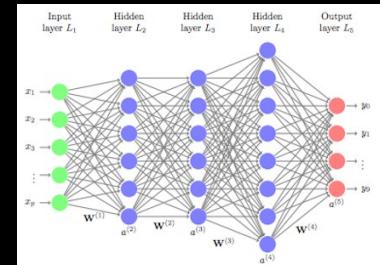
The value 108 is highlighted with a green circle.

95      0  
111      0  
98      0  
119      0  
101      0

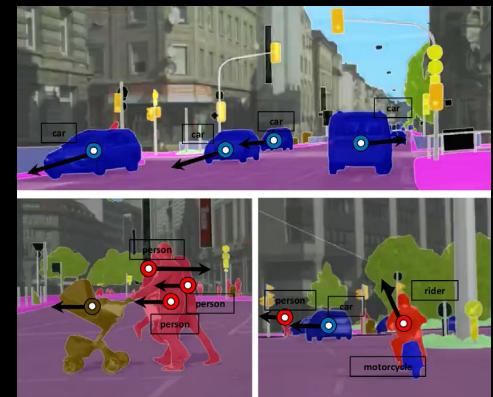
Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)

# Deep learning and color/gray scale transformations

- Deep learning needs training data
  - Input image
  - Ground truth labels or classes
  
- When you lack data you can *augment* your data
  - Create artificial versions
  - Adding variation
  - Changing gray / color levels in the image
  - Point wise operations



[http://uc-r.github.io/feedforward\\_DNN](http://uc-r.github.io/feedforward_DNN)



Luc, Pauline, et al. "Predicting deeper into the future of semantic segmentation." IEEE International Conference on Computer Vision (ICCV). Vol. 1. 2017.



<https://www.quora.com/What-does-the-term-semantic-segmentation-mean-in-the-context-of-Deep-Learning>

# Other mappings

- Non-linear mappings
- Not always nice to work with byte images
  - Better to work with image with values in  $[0,1]$



Byte image

Conversion to  $[0,1]$

Back to bytes

Non-linear transformation



Byte image

# Working with bytes and doubles

- A byte contains integer values [0,255]
  - A byte can not store 127.4232
- A value of type *double* can contain “all numbers”
- Why not use doubles always?
  - One double = 8 bytes in the memory
  - Images become very large!
  - Many things can be done with bytes

# Map pixels to [0,1]

- Simple conversion to [0,1]

$$g(x, y) = \frac{1}{255}f(x, y)$$

## Pixels back to bytes

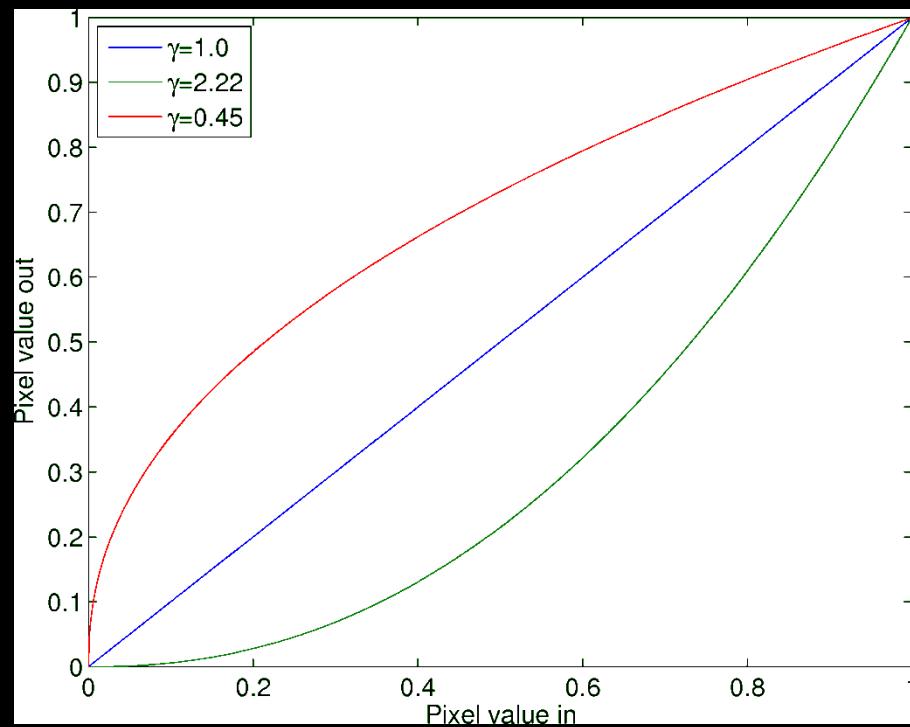
- Input pixels are [0,1]
- We want them to be [0,255]
- Simple linear transformation:

$$g(x, y) = 255 * f(x, y)$$

# Gamma mapping

- Gamma mapping is used in televisions and flat panels
- Can increase the contrast (dynamics) in more selected part of the histogram
- Many games have a possibility for a gamma correction

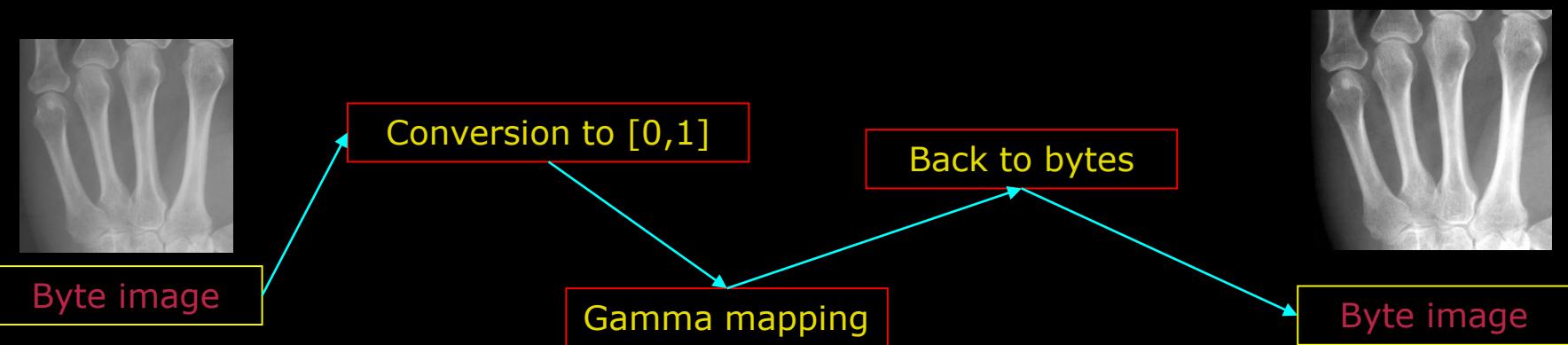
# Gamma curves



- Named after the Greek letter gamma
- What happens to the dark areas
  - With **0.45**?
  - With **2.22**?

$$g(x, y) = f(x, y)^\gamma$$

# Perform the gamma mapping

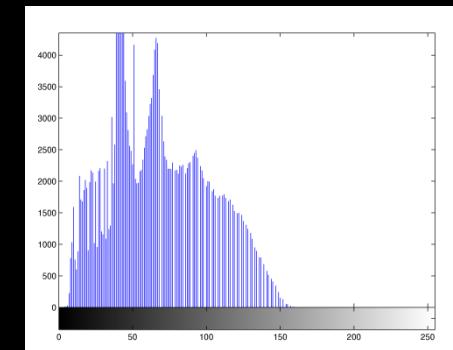
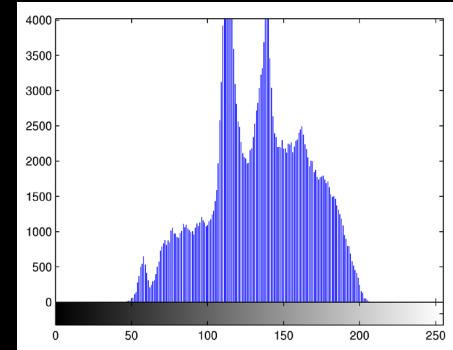
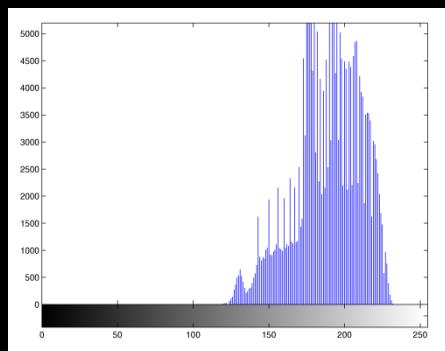


# Results of gamma mapping

0.45



2.22





Gamma mapping on an image

A gamma mapping is performed on the image below with  $\gamma = 1.3$ . What is the minimum and maximum value in the mapped image?

208	25	40	36	167
231	71	23	108	18
32	139	244	234	217
233	244	124	202	238
161	245	204	245	173

0, 255

25, 130

8, 242

15, 230

37, 219

Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)

Gamma mapping on an image

A gamma mapping is performed on the image below with  $\gamma = 1.3$ . What is the minimum and maximum value in the mapped image?

208	25	40	36	167
231	71	23	108	18
32	139	244	234	217
233	244	124	202	238
161	245	204	245	173

0, 255

25, 130

8, 242

15, 230

37, 219

Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)

Gamma mapping on an image

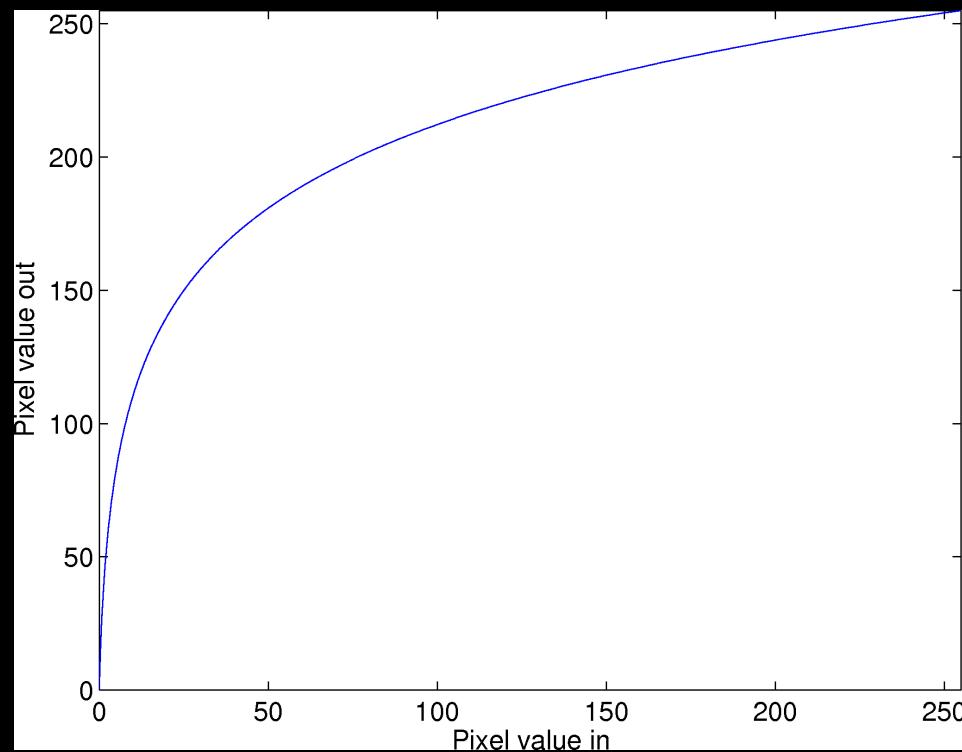
A gamma mapping is performed on the image below with  $\gamma = 1.3$ . What is the minimum and maximum value in the mapped image?

208	25	40	36	167
231	71	23	108	18
32	139	244	234	217
233	244	124	202	238
161	245	204	245	173

0, 255      25, 130      8, 242      15, 230      37, 219

Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)

# Logarithmic mapping



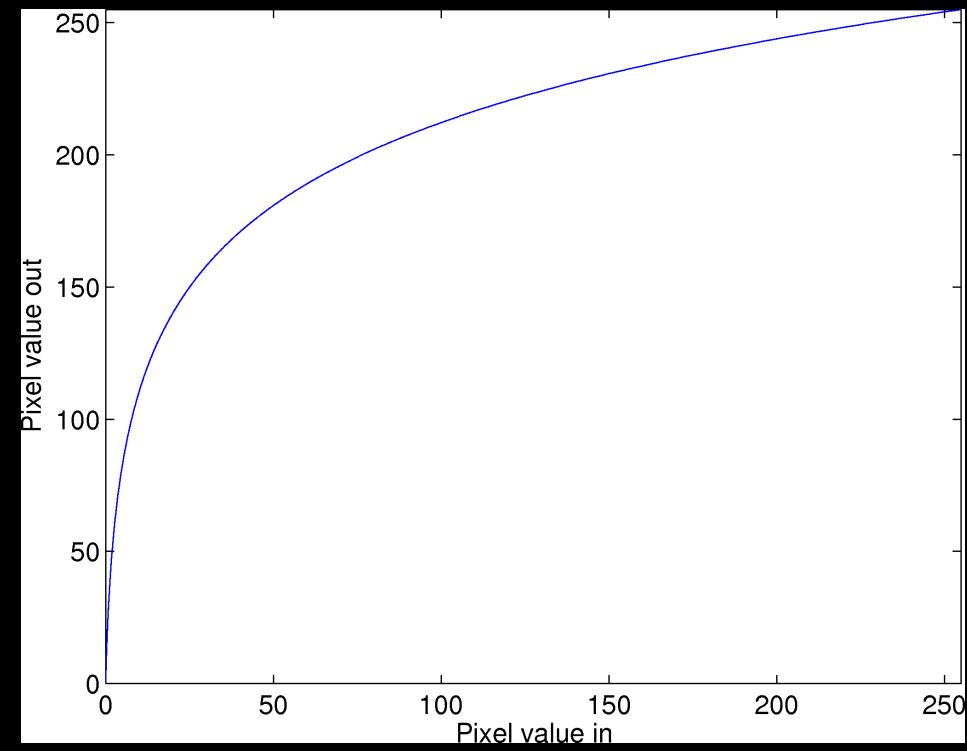
Maps from [0,255] to [0,255]

Why?  
$$g(x, y) = c \log(1 + f(x, y))$$

$$c = \frac{255}{\log(1 + v_{max})}$$

# Logarithmic mapping – when?

- For images with very bright spots
- Low intensity pixel values are enhanced

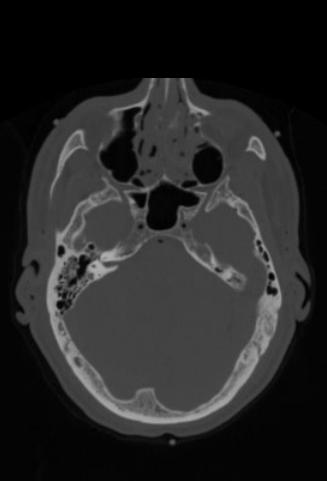


# What do we get out of pixel mappings

- Spreading out or compressing pixel values
  - Better for humans to see
  - New information – no!

# Now for something different

- Until now image processing
  - Input image transformed to output image
- Now for something more like image analysis
- Segmentation
  - Segment the image into regions
    - Background and objects for example



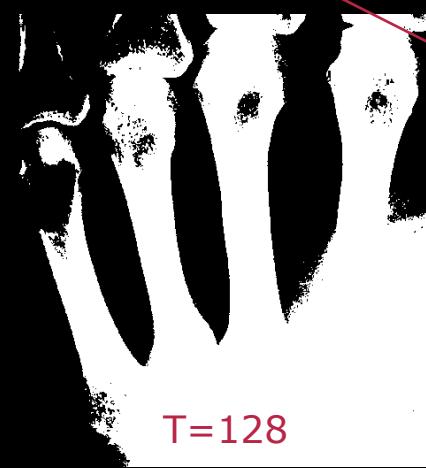
# Thresholding

- A threshold  $T$  is a value
  - Pixels below that value is set to 0 (background)
  - Pixels equal or above is set to 1 (object)
- One threshold value for the entire image
  - Difficult to choose!

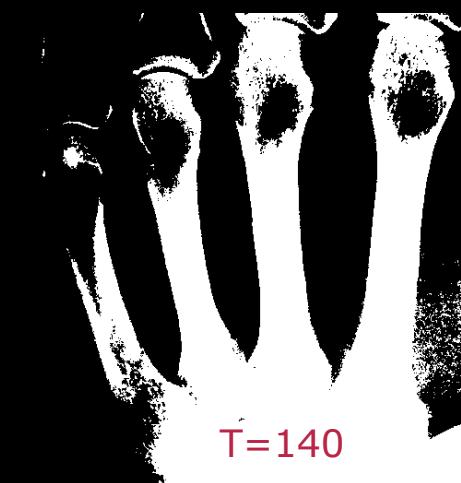
$\text{if } f(x, y) \leq T \text{ then } g(x, y) = 0$

$\text{if } f(x, y) > T \text{ then } g(x, y) = 255$

# Thresholding



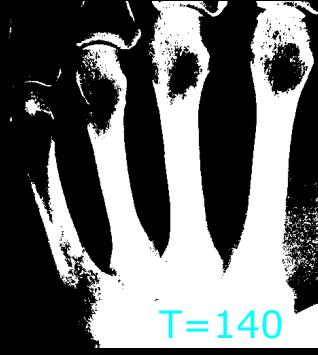
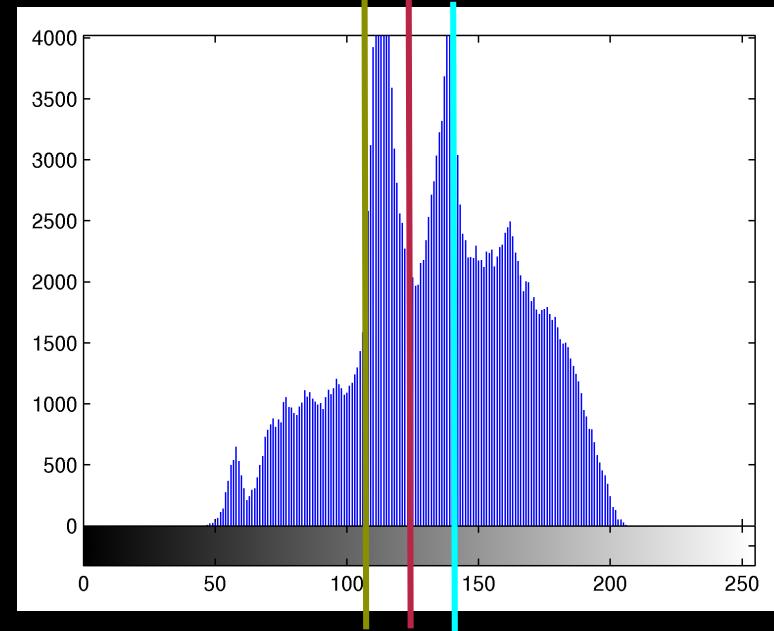
Background  
and bone  
have same  
value!



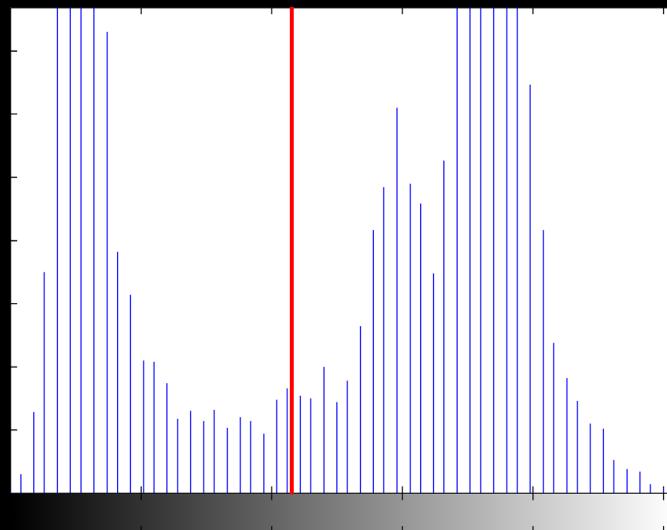
# Thresholding based on the histogram



The bones are visible in the histogram!  
But mixed with soft-tissue



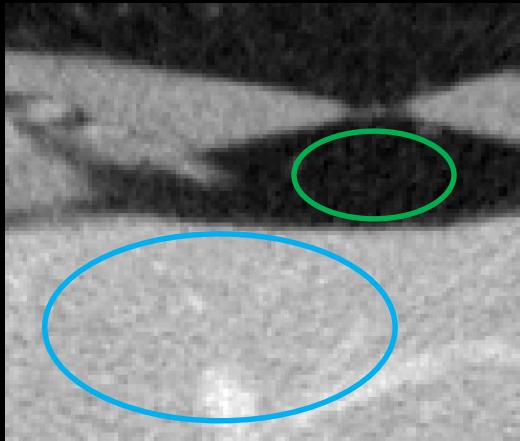
# Automatic Thresholding



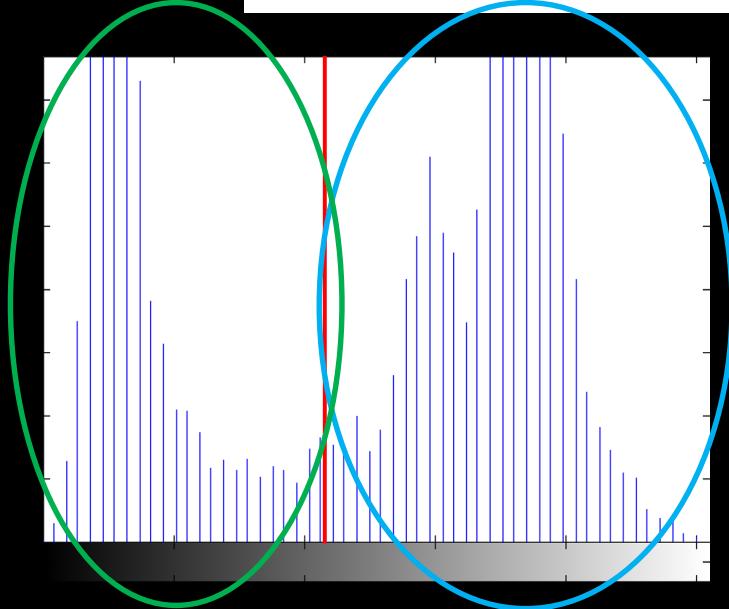


# Automatic Thresholding

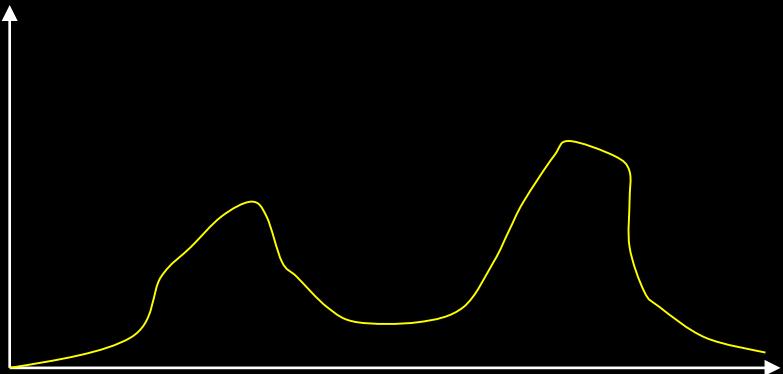
## Otsu's method



- Two classes: **background** and **object**
- $T$  divides pixels into object and background
- Compute pixel value variance in each class
- Find  $T$  that minimises combined variance



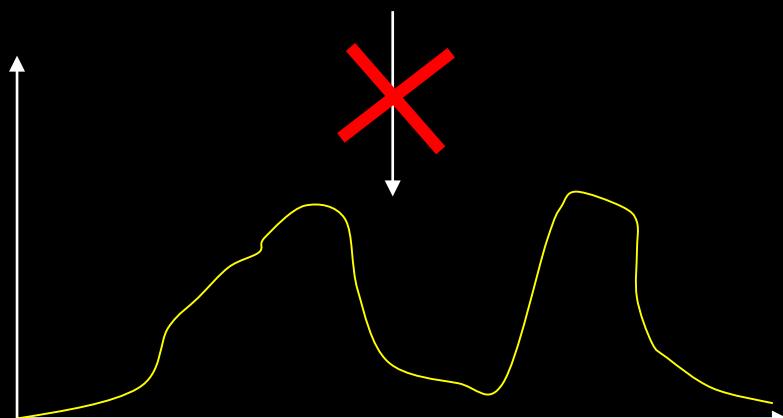
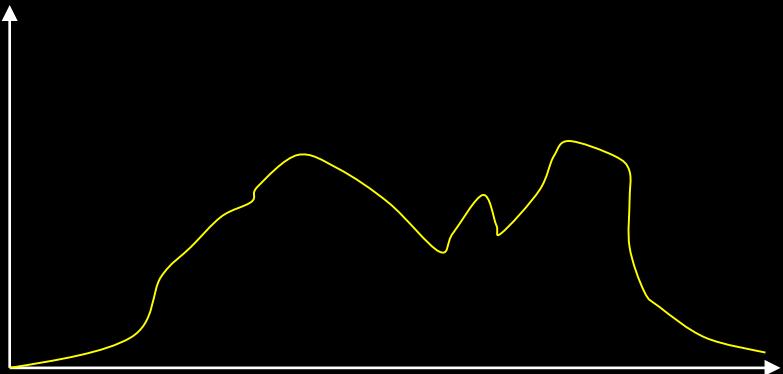
# Segmentation – histogram shaping



Ideal

- With a threshold you want a histogram with two peaks
  - *Bimodal*
- An ideal histogram has well separated peaks
- Obtaining a bi-modal histogram is very important in the image acquisition

# Histogram shaping



- It is not possible to “unmix” using gray level transformations

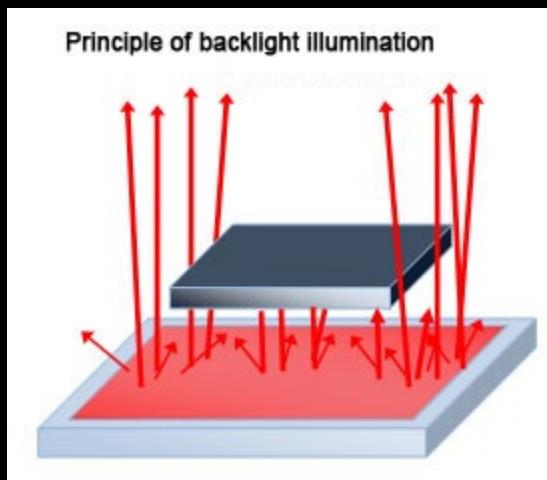
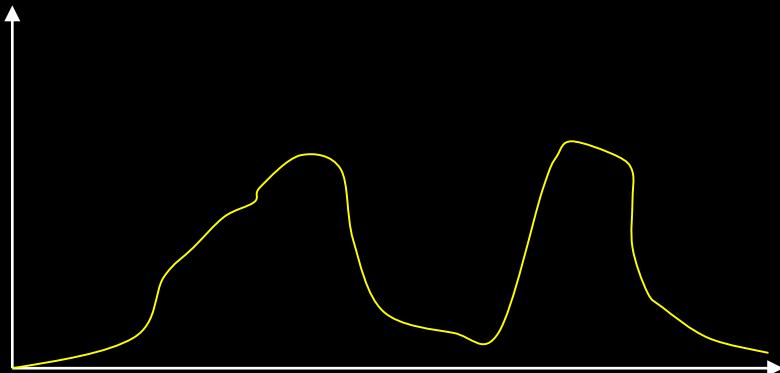


Should be  
higher

Should be  
lower

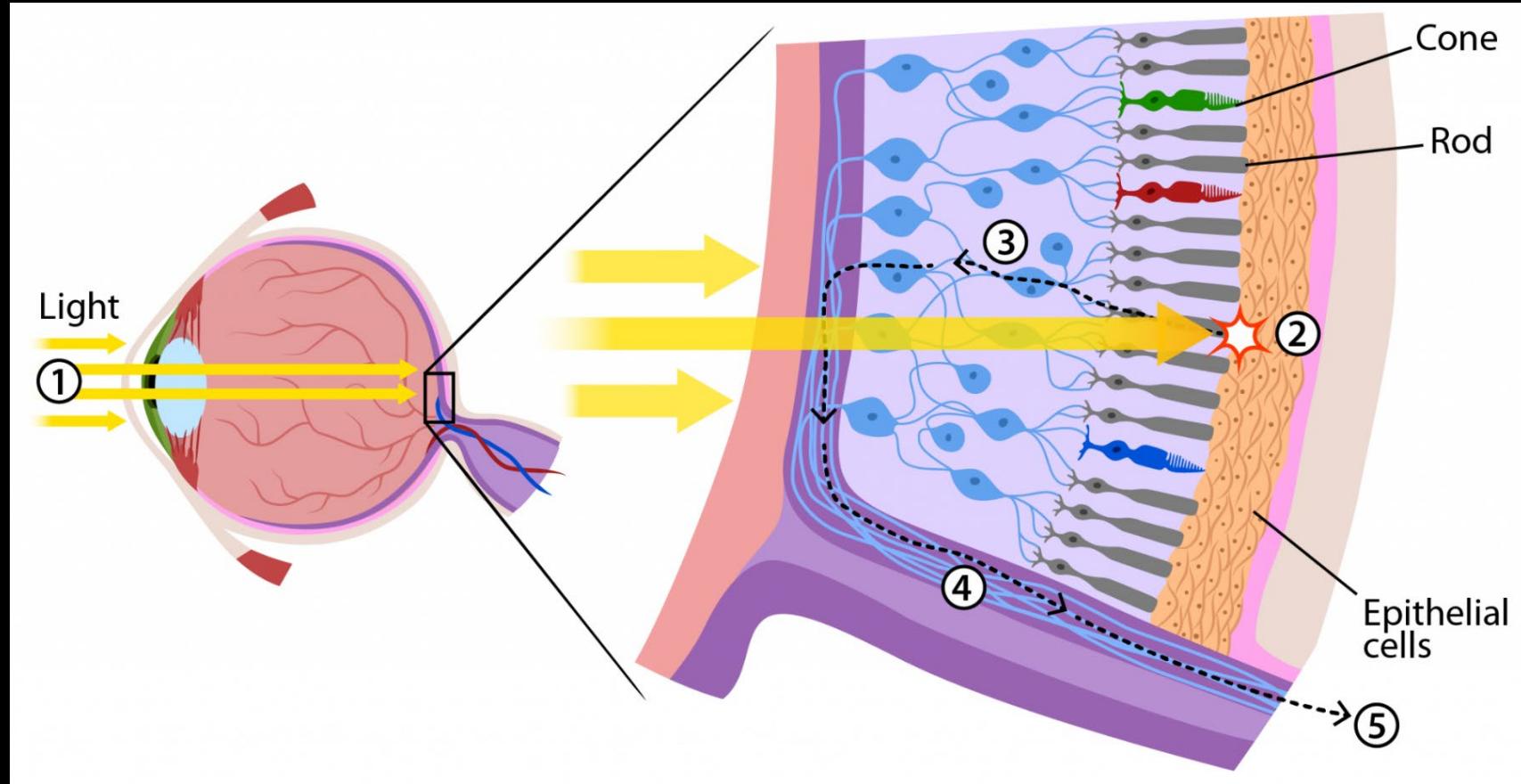
# How to obtain good histograms

- With cameras
  - Light
  - Setup
  - Camera
  - Lens
  - Backlight



# Colour images and colour perception

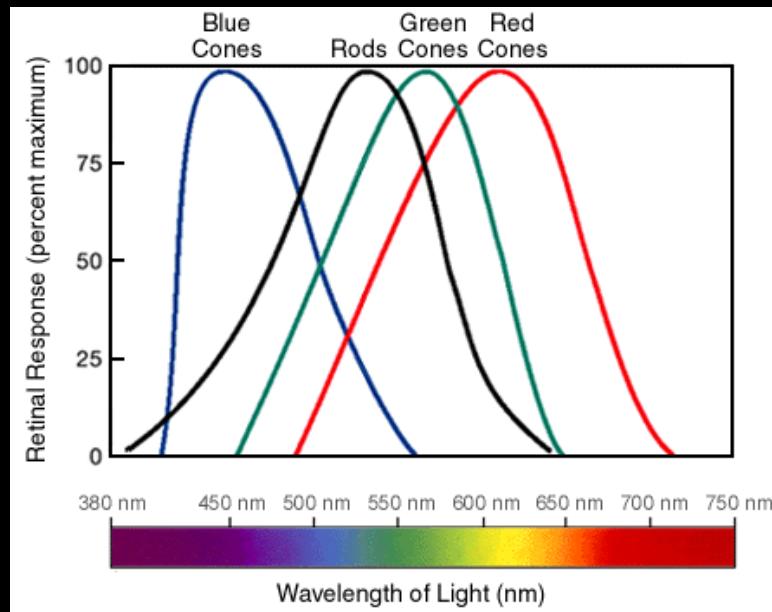
## The Human Eye



<https://askabiologist.asu.edu/rods-and-cones>

# Color sensitivity

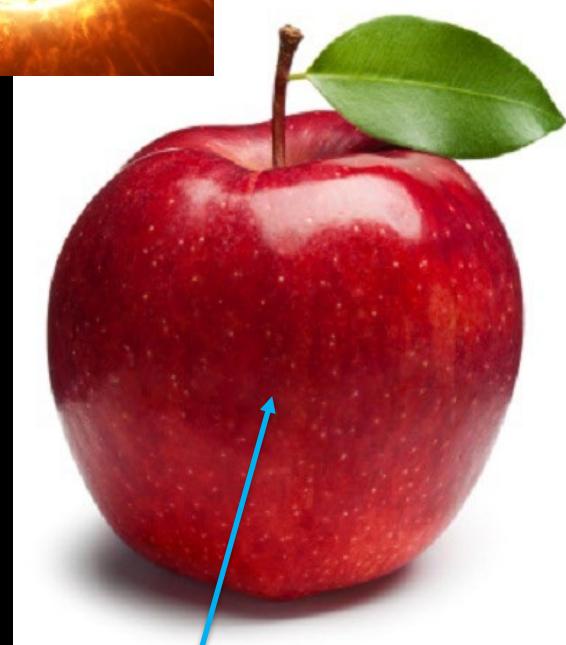
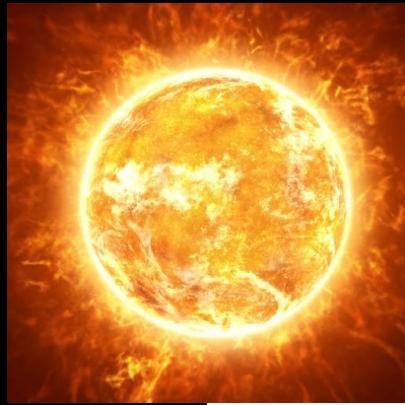
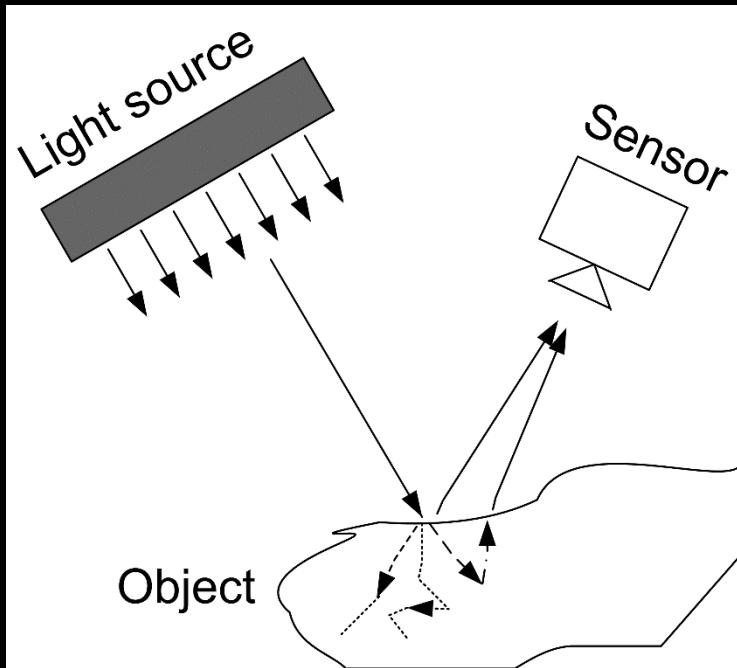
Photoreceptor cell	Wavelength in nanometers (nm)	Peak response in nanometer (nm)	Interpretation by the human brain
Cones (type L)	[400-680]	564	Red
Cones (type M)	[400-650]	534	Green
Cones (type S)	[370-530]	420	Blue
Rods	[400-600]	498	Shade of gray



<https://askabiologist.asu.edu/rods-and-cones>

# Object colors

## Subtractive colors



All other colors than red absorbed

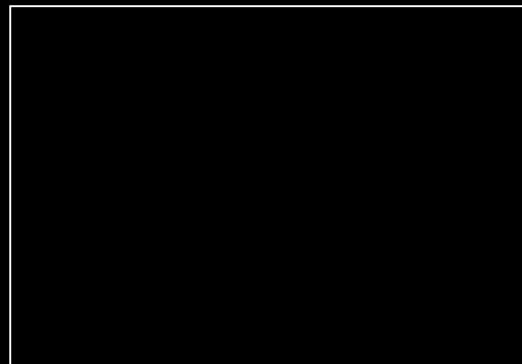
# Object colors

## Additive colors



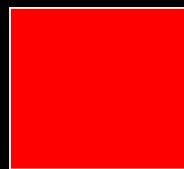
- Additive colours: Final colour is made by mixing red, green, and blue
- RGB = Red, Green, and Blue
- Television, computers, digital cameras use the “RGB color space”
- Typically the values of R, G, and B lie between 0 and 255

# RGB Colours

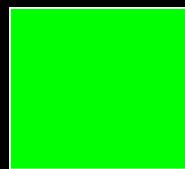


RGB = (0,0,0)

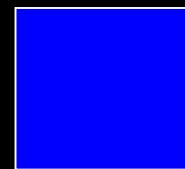
- When alle three “Lamps” are turned of we get black
- When all three “lamps” are on what do we get?



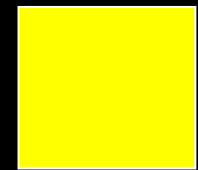
(255,0,0)



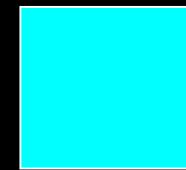
(0,255,0)



(0,0,255)



(255,255,0)



(0,255,255)

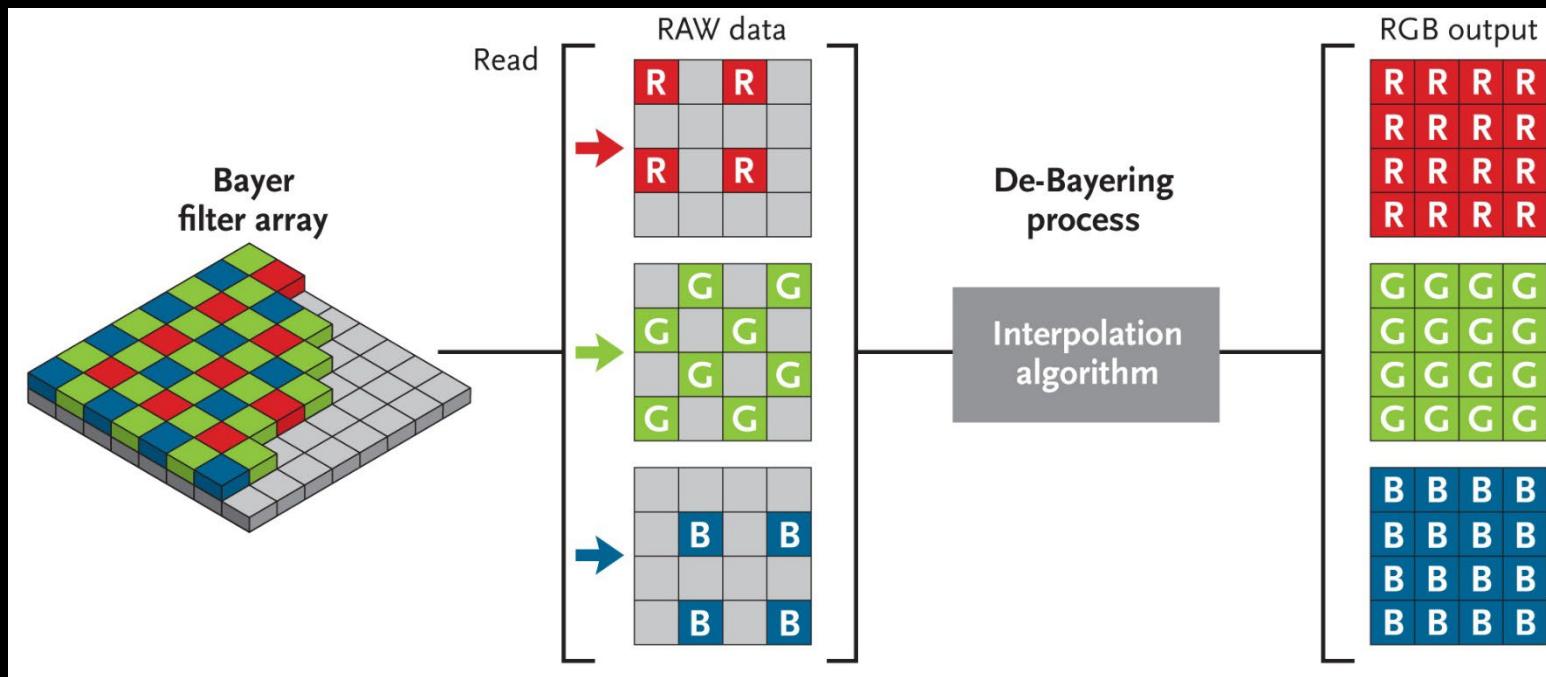


RGB = (255,255,255)



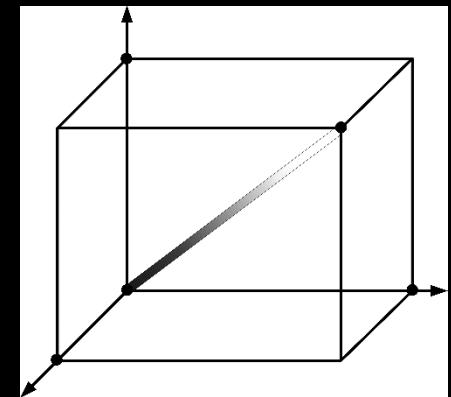
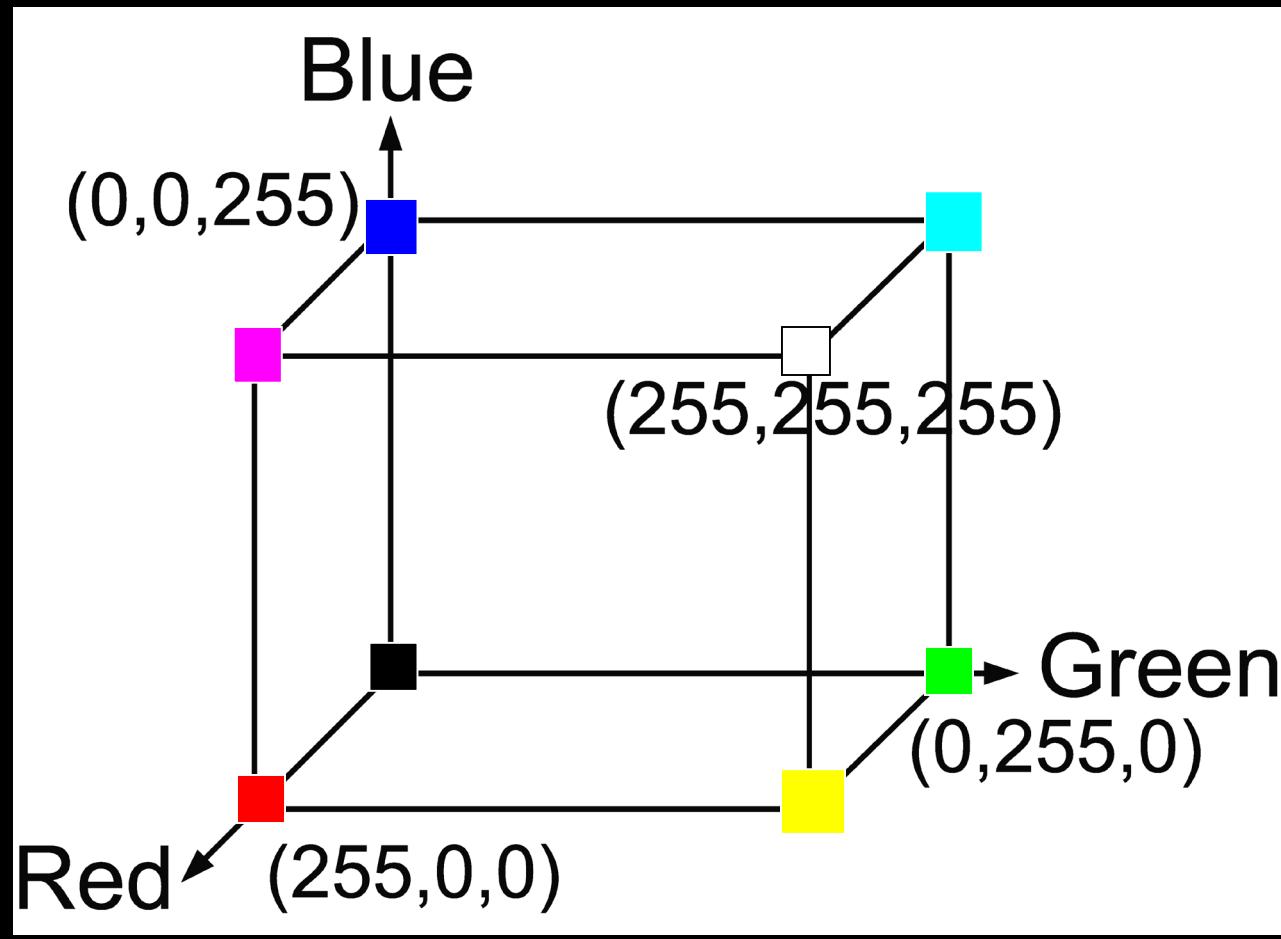
(255,0,255)

# Color camera with one sensor



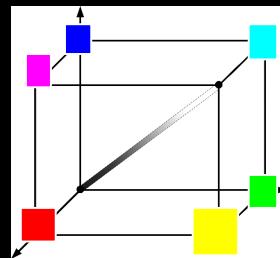
<http://www.skyandtelescope.com/astronomy-resources/astrophotography-tips/redeeming-color-planetary-cameras/>

# RGB color space



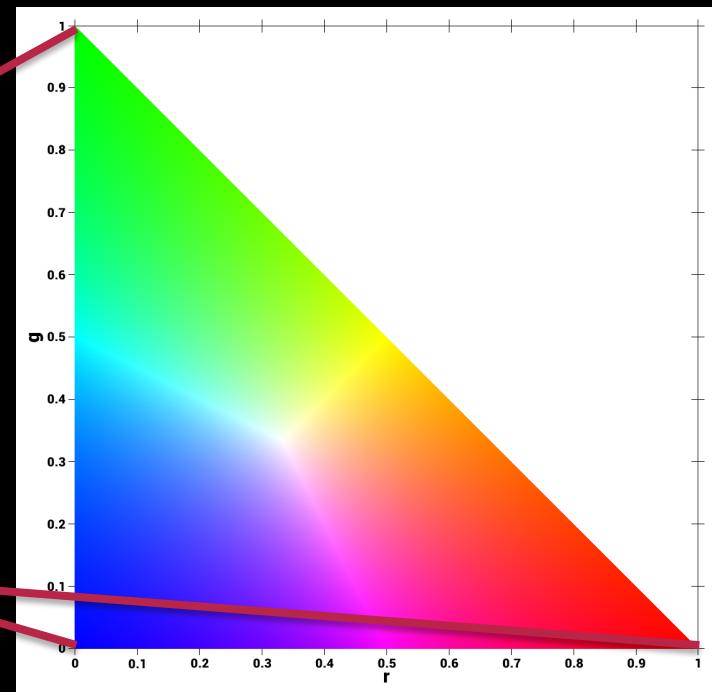
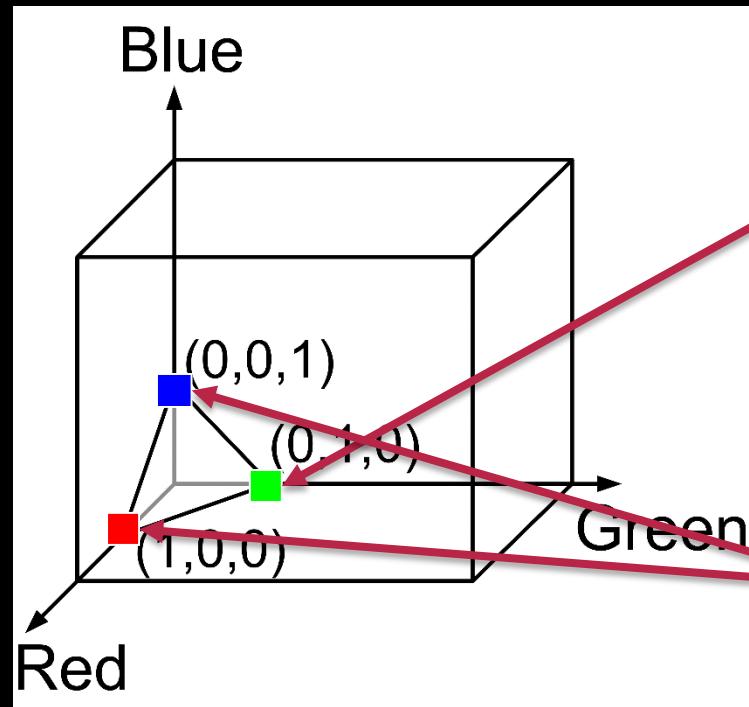
# Converting colour to grayscale

$$v = 0.2989 * R + 0.5870 * G + 0.1140 * B$$

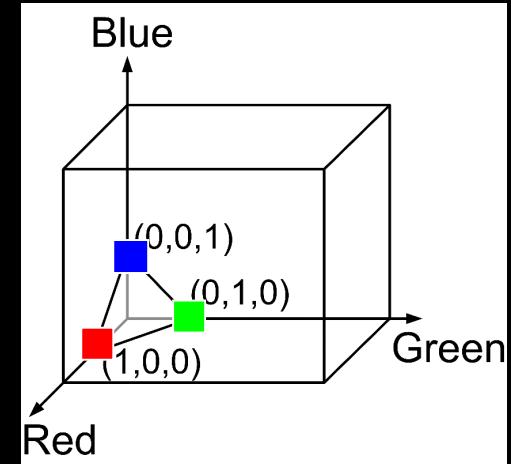
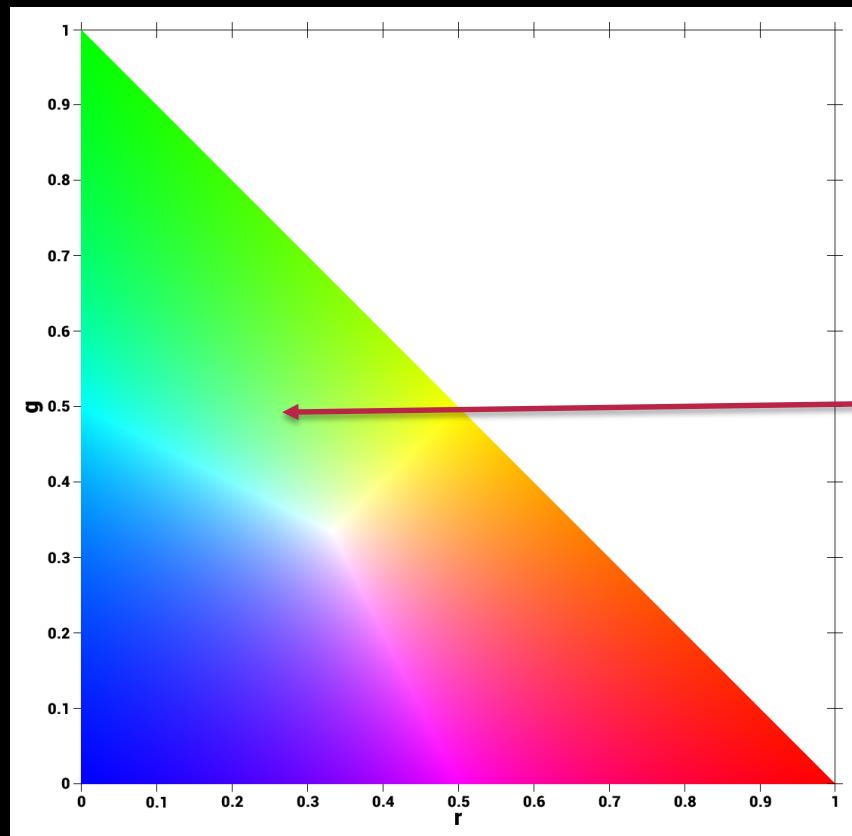


# Normalized RGB colors

$$(r, g, b) = \left( \frac{R}{R + G + B}, \frac{G}{R + G + B}, \frac{B}{R + G + B} \right)$$



# Another RGB representation

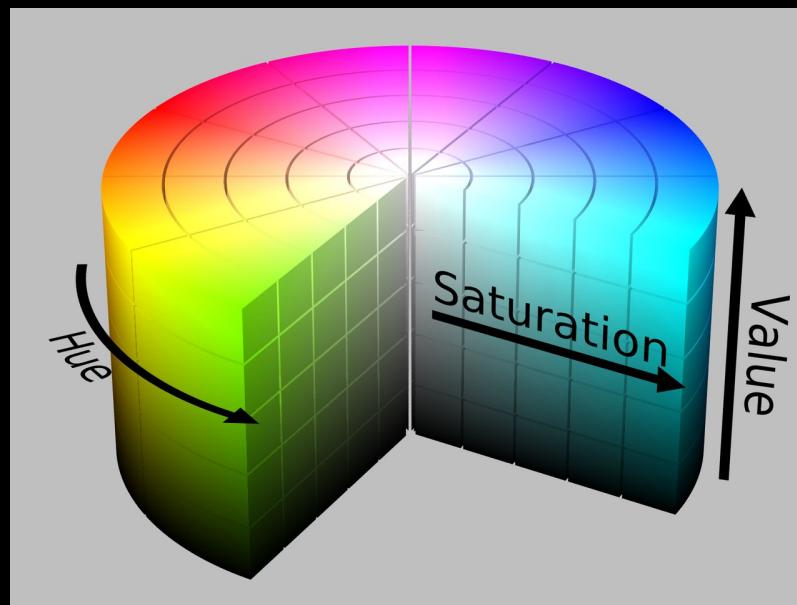


(r,g,I)

$$I = \frac{R+G+B}{3}.$$

# HSI Color Representation

- **Hue** – the dominant wave length in the perceived light (the pure color)
- **Saturation** – the purity of the color
- **Intensity** – the brightness of the color (sometimes called the value)



# Converting between RGB and HSI

- You have an RGB value
- You want the corresponding HSI value

$$H = \begin{cases} \cos^{-1} \left( 1/2 \cdot \frac{(R-G)+(R-B)}{\sqrt{(R-G)(R-G)+(R-B)(G-B)}} \right), & \text{if } G \geq B; \\ 360^\circ - \cos^{-1} \left( 1/2 \cdot \frac{(R-G)+(R-B)}{\sqrt{(R-G)(R-G)+(R-B)(G-B)}} \right), & \text{Otherwise.} \end{cases} \quad (8.8)$$

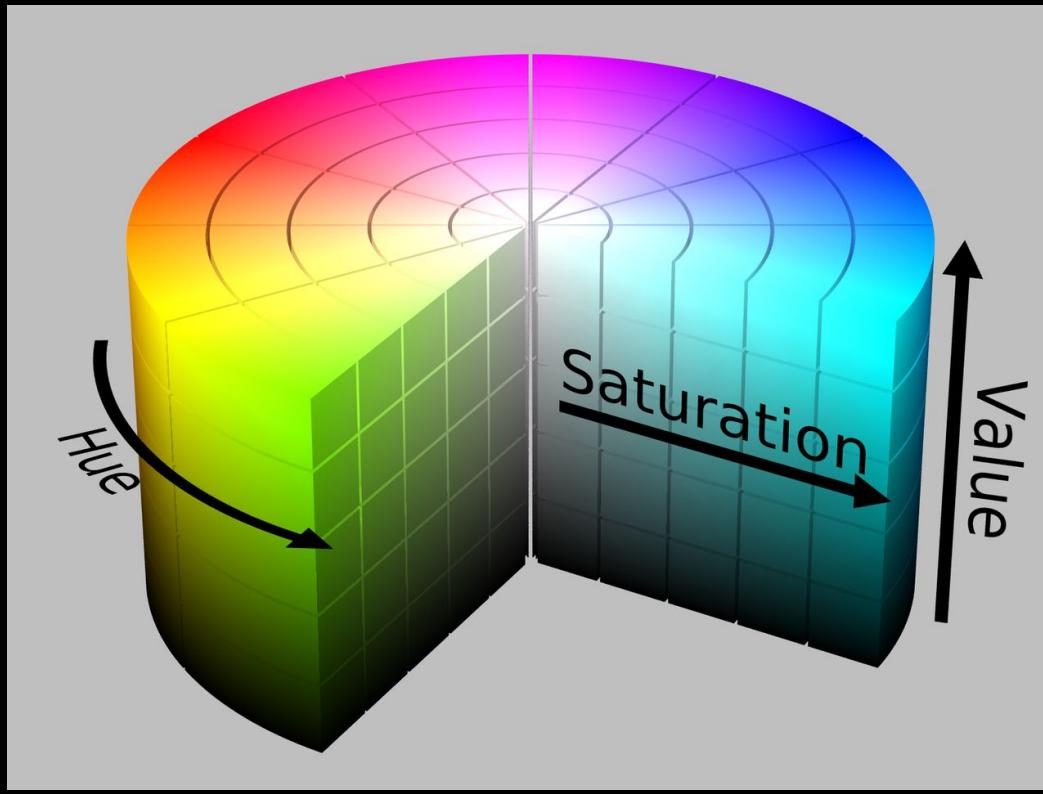
$H \in [0, 360[$

$$S = 1 - 3 \cdot \frac{\min\{R, G, B\}}{R + G + B} \quad S \in [0, 1] \quad (8.9)$$

$$I = \frac{R + G + B}{3} \quad I \in [0, 255] , \quad (8.10)$$

# Why other colorspaces

- Why should we use for example HSI ?

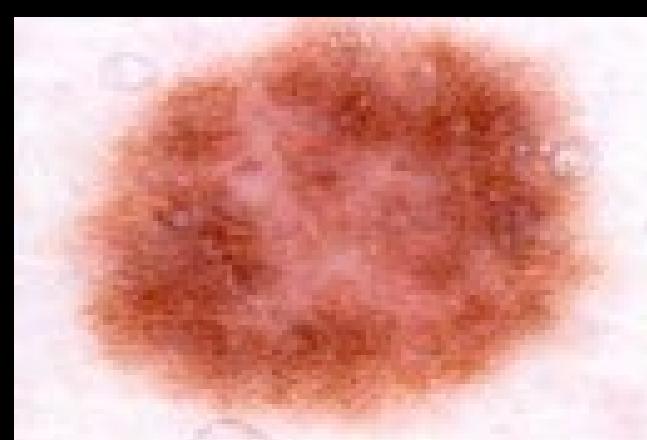
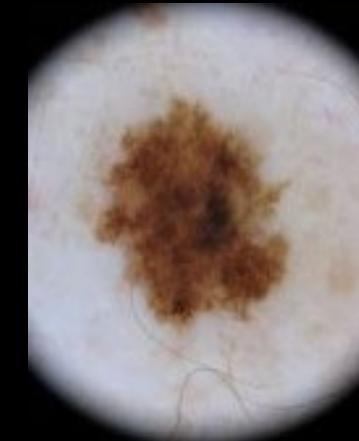
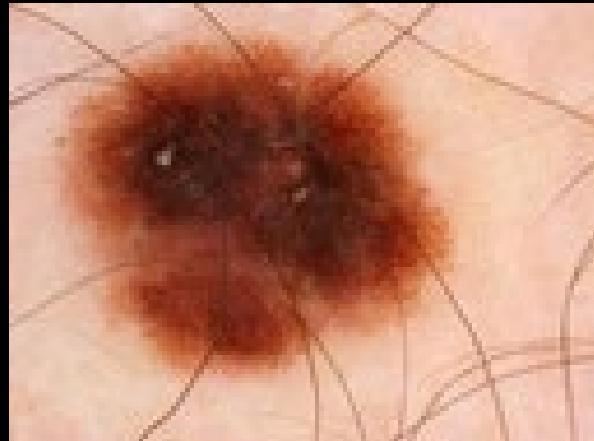


# Melanoma segmentation

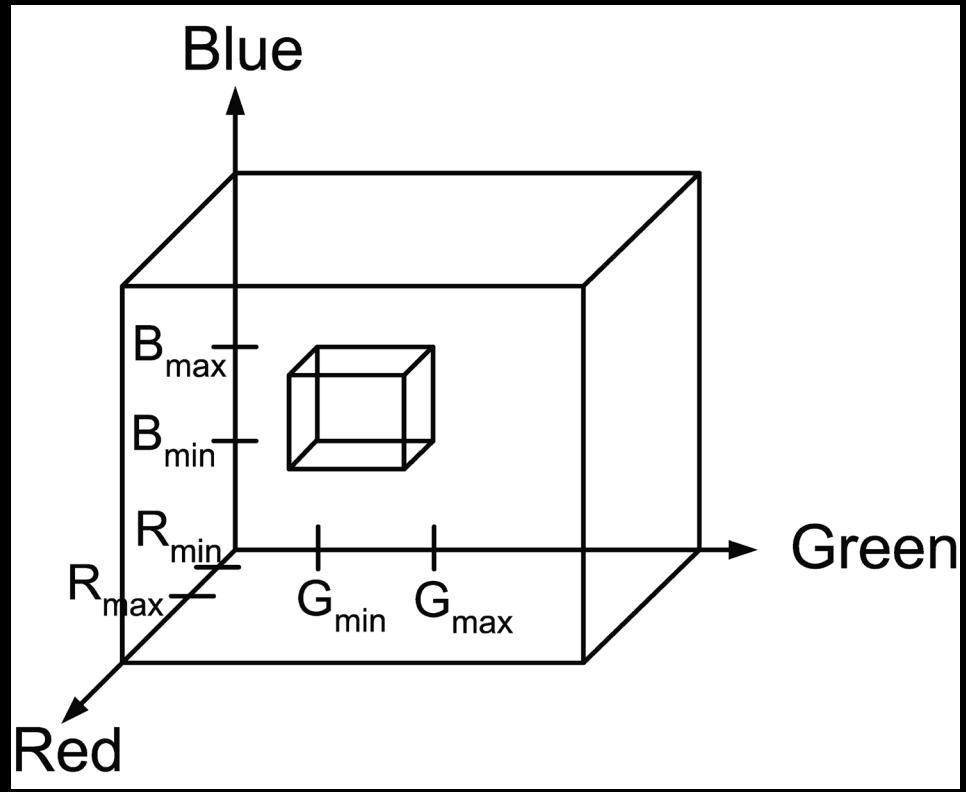


- An algorithm that can do pixelwise classification
  - Background / skin
  - Melanoma
  
- Use the colors

# Melanoma segmentation – color variation



# Color thresholding



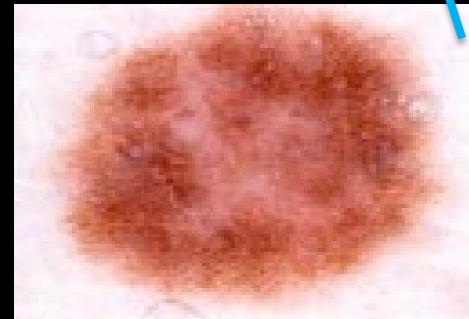
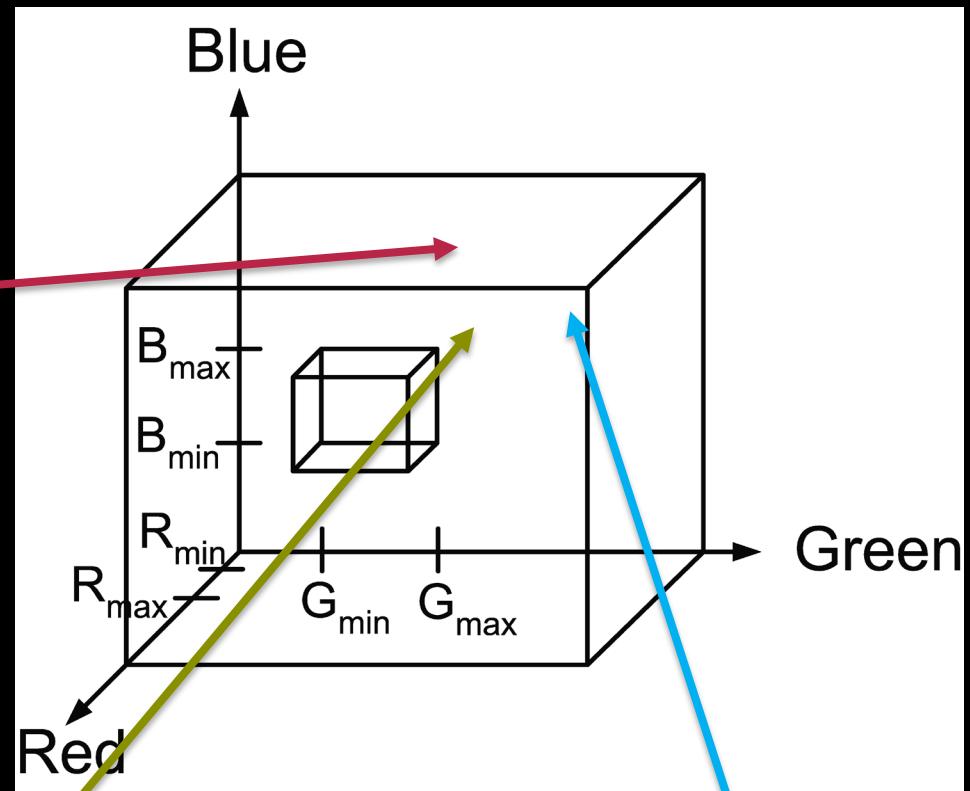
If

$R > R_{min}$  and  $R < R_{max}$  and  
 $G > G_{min}$  and  $G < G_{max}$  and  
 $B > B_{min}$  and  $B < B_{max}$

Then  $g(x, y) = 255$

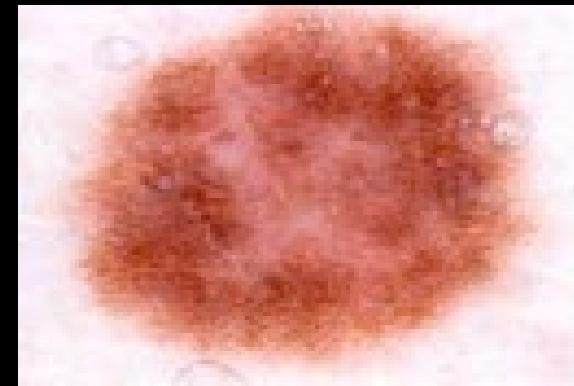
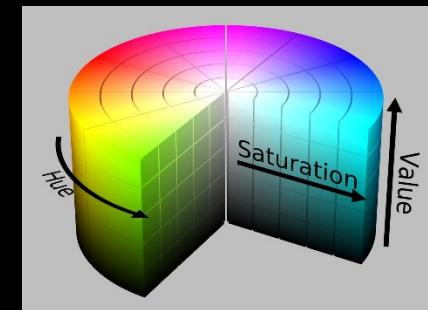
Else  $g(x, y) = 0$

# Color thresholding



# Color variation

- The major variation is in the brightness
  - This will spread out the values in RGB space
- The Hue is rather constant
- HSI Space
  - HUE and saturation rather stable
  - Only variation in intensity / value





**Level of the lectures**

Far too easy

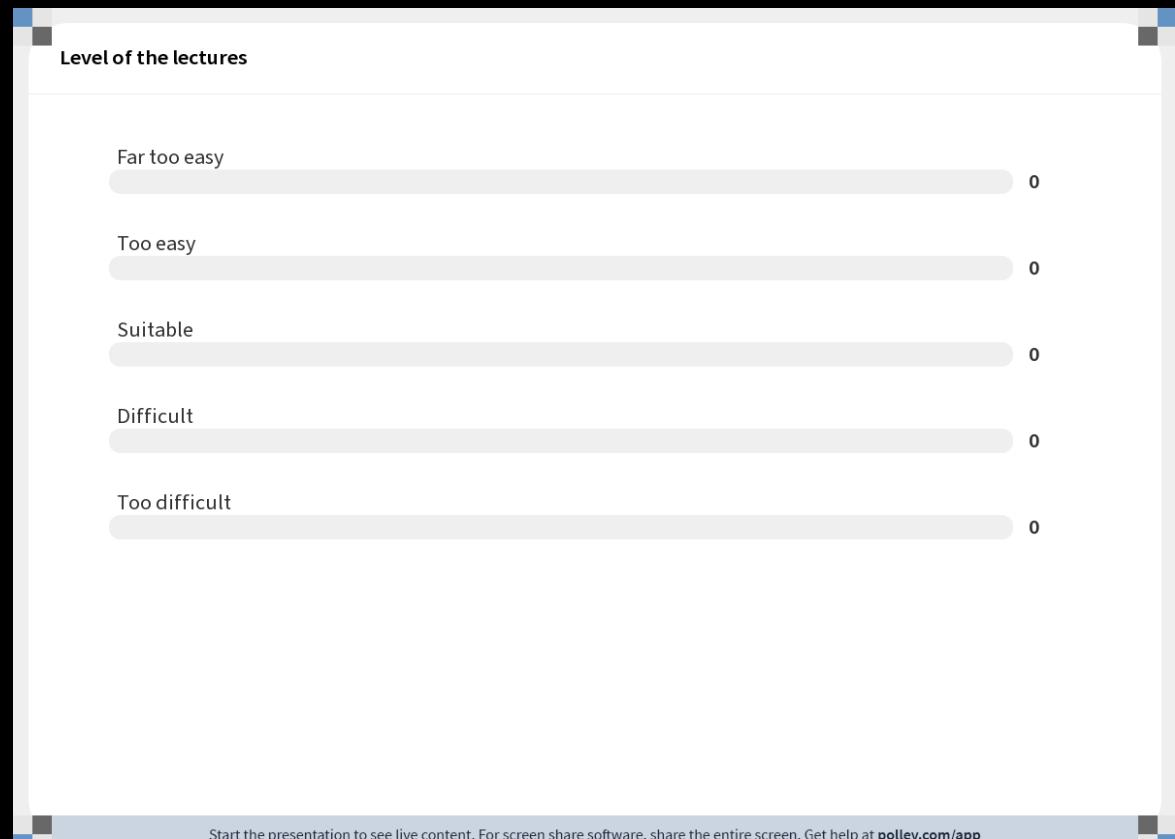
Too easy

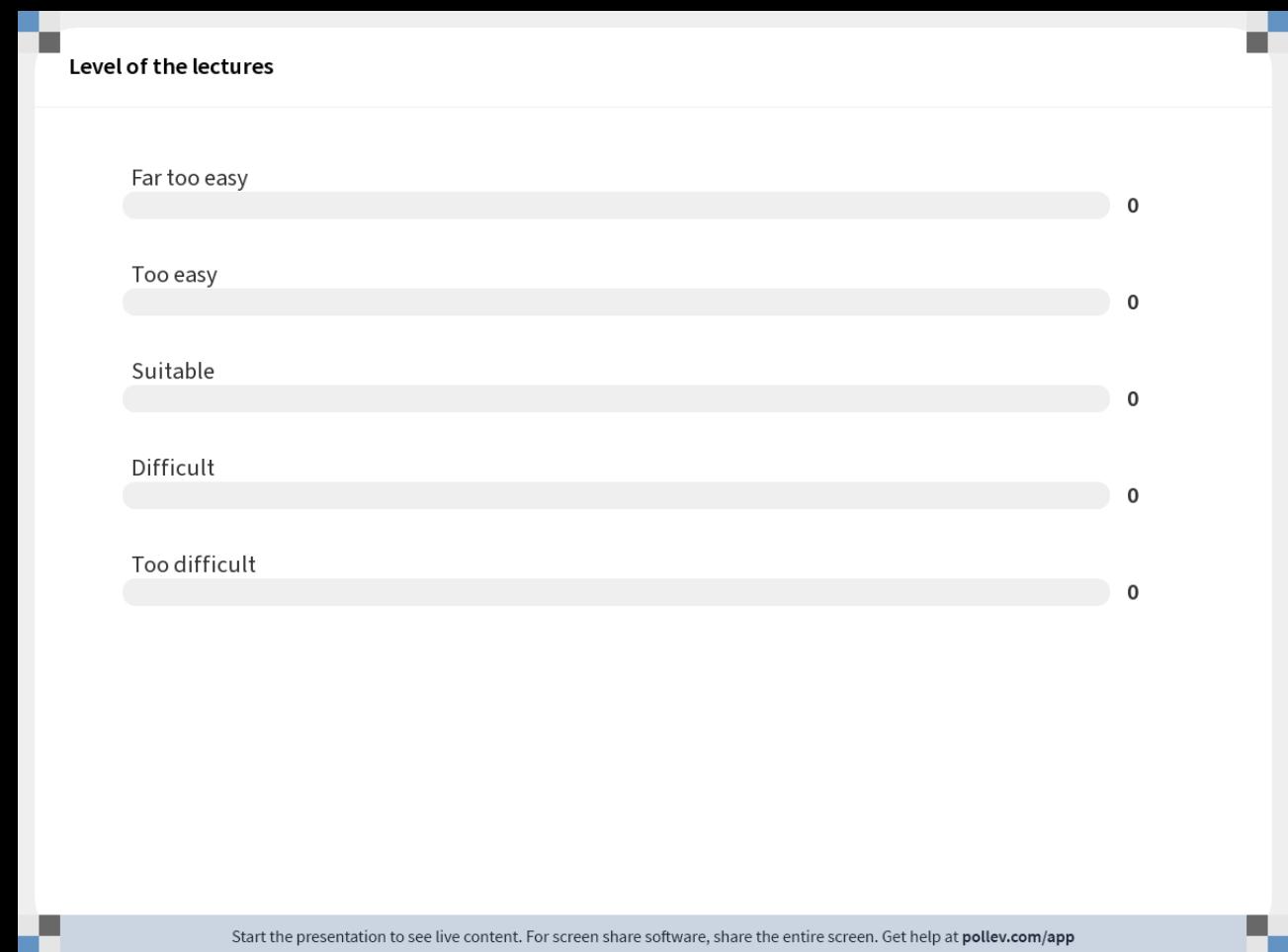
Suitable

Difficult

Too difficult

Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)







### Level of the exercises

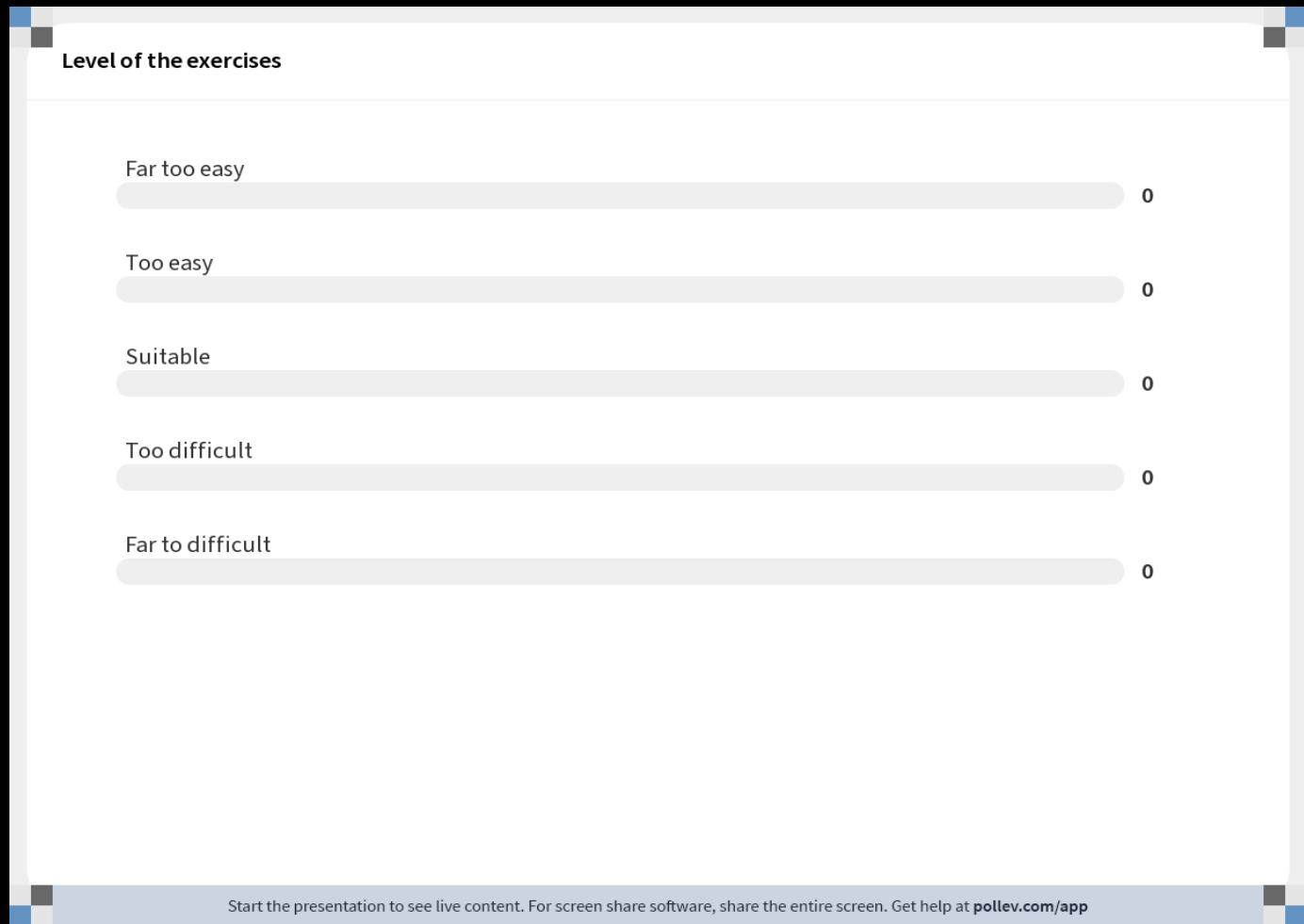
Far too easy

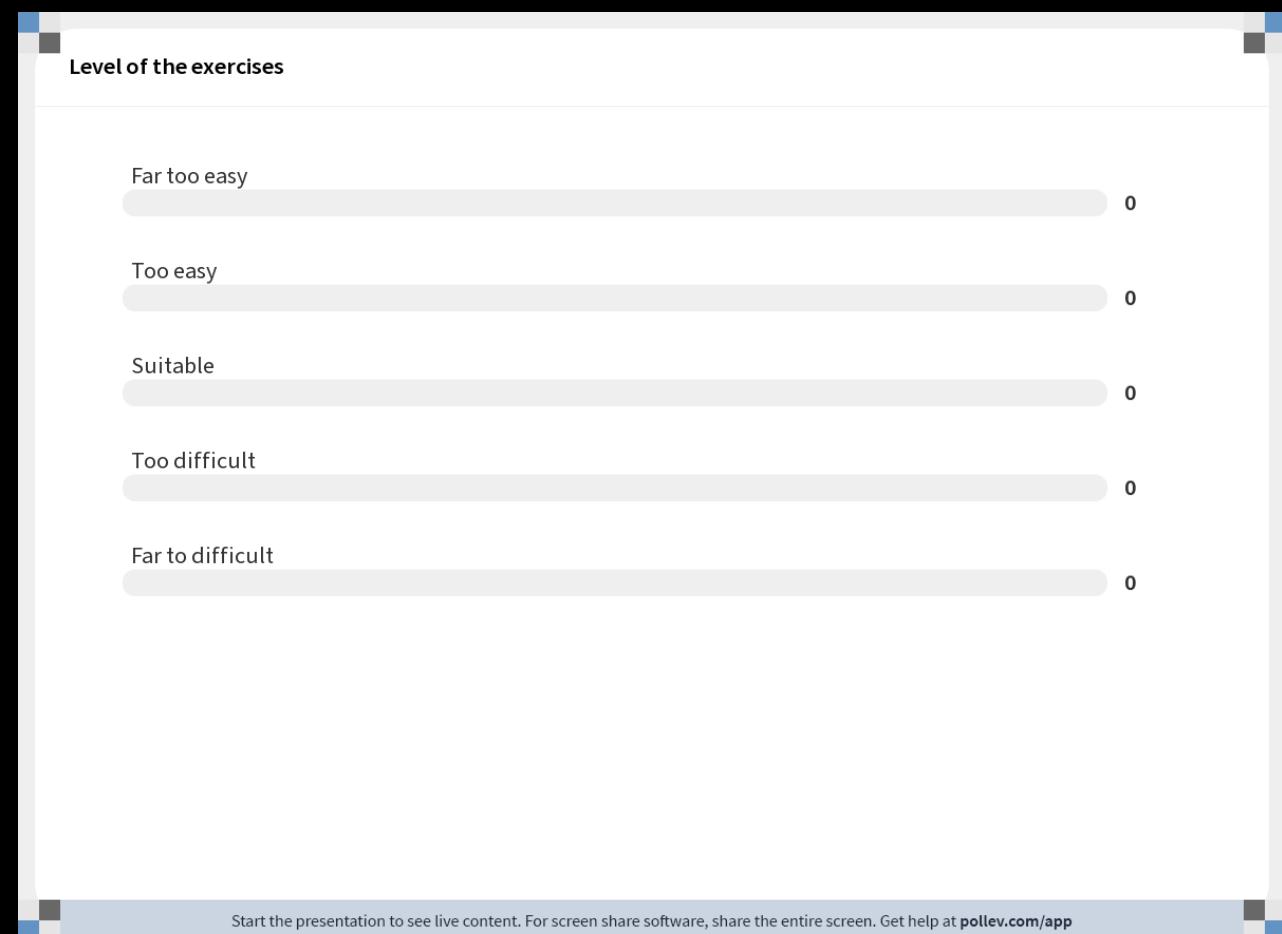
Too easy

Suitable

Too difficult

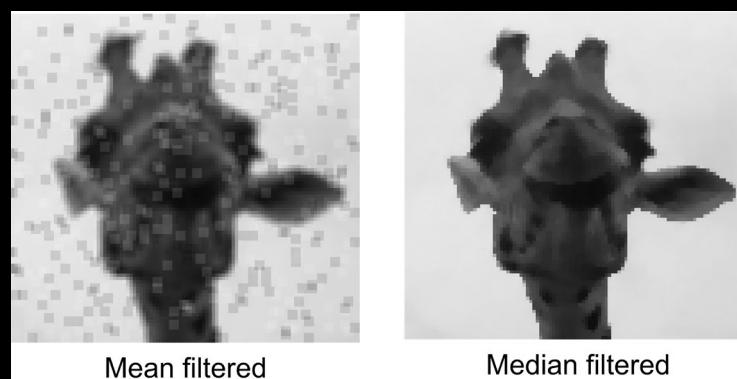
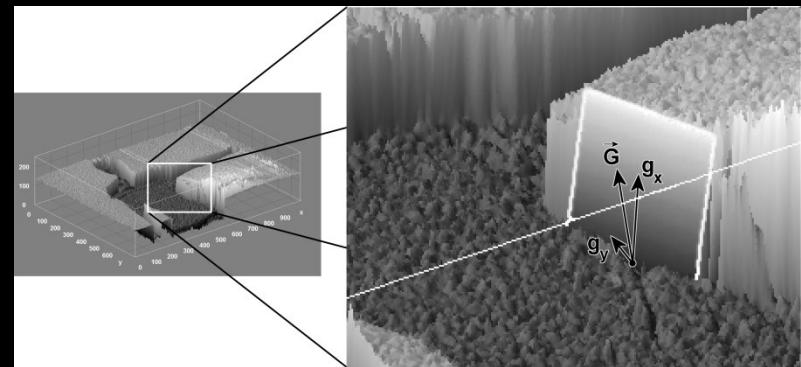
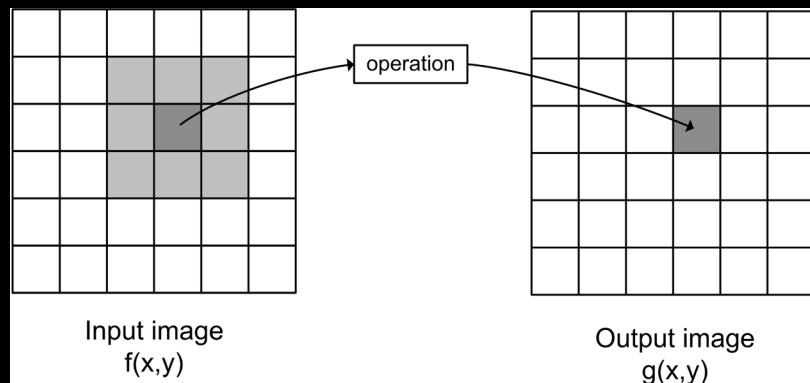
Far too difficult





# Next week

- Neighbourhood processing (Filtering)
- Morphology





# Principal component analysis on images

Rasmus R. Paulsen

DTU Compute

Based on

M. Turk and A. Pentland. *Face recognition using eigenfaces*. Computer Vision and Pattern Recognition, 1991.

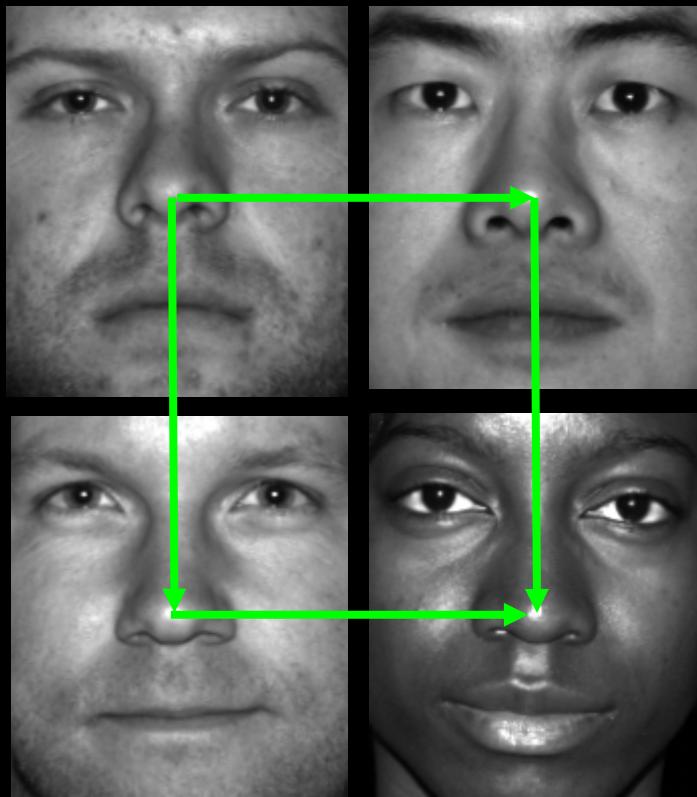
<http://compute.dtu.dk/courses/02502>

# Principal Component Analysis on images

## learning objectives

- Construct a column matrix from a single gray scale image
- Construct a data matrix from a set of gray scale images
- Compute and visualize an average image from a set of images
- Compute the principal components of a set of images
- Visualize the principal components computed from a set of images
- Synthesize an image by combining the average image and a linear combination of principal components

# Face data



- 38 face images
  - 168 x 192 grayscale
- Aligned
  - The anatomy is placed "in the same position in all image"
- Same illumination conditions on the images we use

The Extended Yale Face Database B

<http://vision.ucsd.edu/~leekc/ExtYaleDatabase/ExtYaleB.html>

# Principal component analysis on face images



- What is the main variation in face images?
  - The variation of appearance
  - Not the position in the image
  - Not the light conditions
  - Not the direction of the head

# Putting images into matrices

- An image can be made into a column matrix
  - Stack all image columns into one column



$$\Rightarrow \begin{bmatrix} p_1 \\ p_2 \\ \dots \\ p_m \end{bmatrix}$$

...

# Face images in matrix form

- One column is one face
- $n=38$  faces
- $m=168 \times 192 = 32256$  pixel values per image



$$\mathbf{X} = \begin{bmatrix} p_{1,1} & \cdots & p_{1,n} \\ \vdots & \ddots & \vdots \\ p_{m,1} & \cdots & p_{m,n} \end{bmatrix}$$

# The average face



$$\mathbf{X} = \begin{bmatrix} p_{1,1} & \cdots & p_{1,n} \\ \vdots & \ddots & \vdots \\ p_{m,1} & \cdots & p_{m,n} \end{bmatrix}$$



## ■ The average face

- Average of each row
- One column
- Put it back into image shape

## ■ Blurry around the eyes

- Not perfectly aligned

# Subtracting the mean face

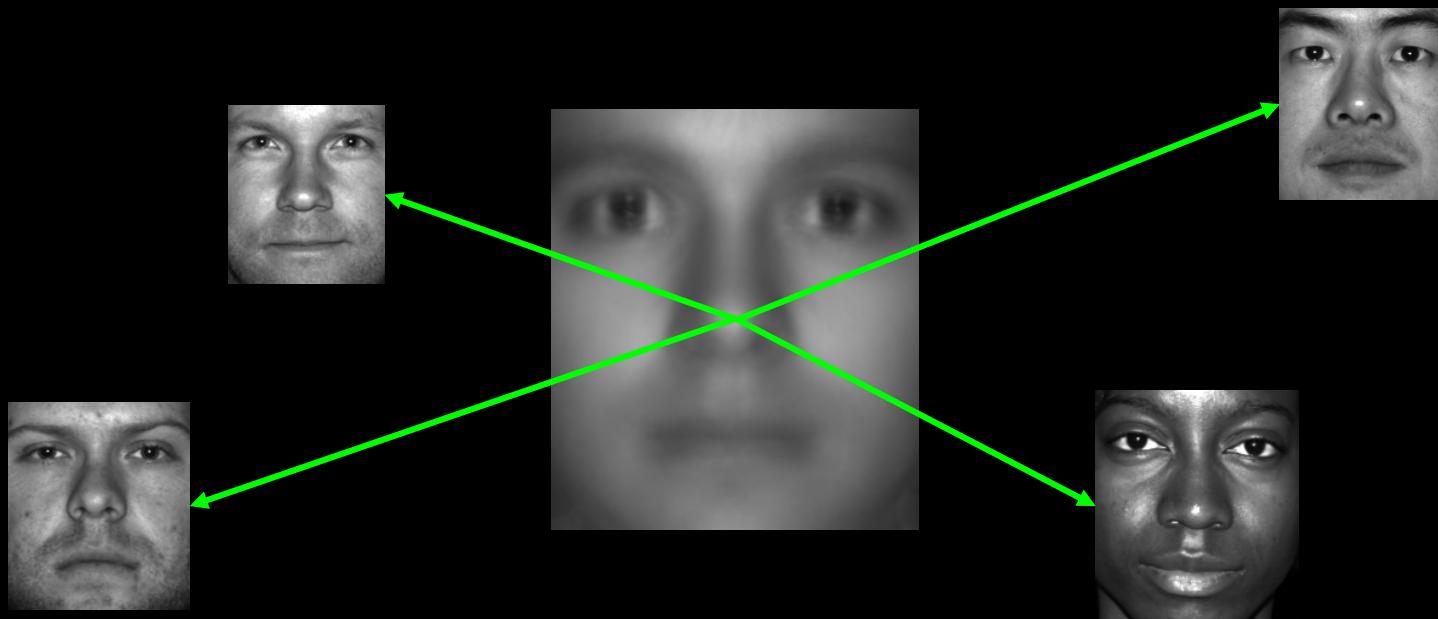
$$\mathbf{X}' = \begin{bmatrix} p_{1,1} & \cdots & p_{1,n} \\ \vdots & \ddots & \vdots \\ p_{m,1} & \cdots & p_{m,n} \end{bmatrix} - \bar{\mathbf{X}}$$

- We subtract the mean face from all faces



# Analyzing the deviation from the mean face

- We want to do the principal component analysis on the *deviations from the average face*

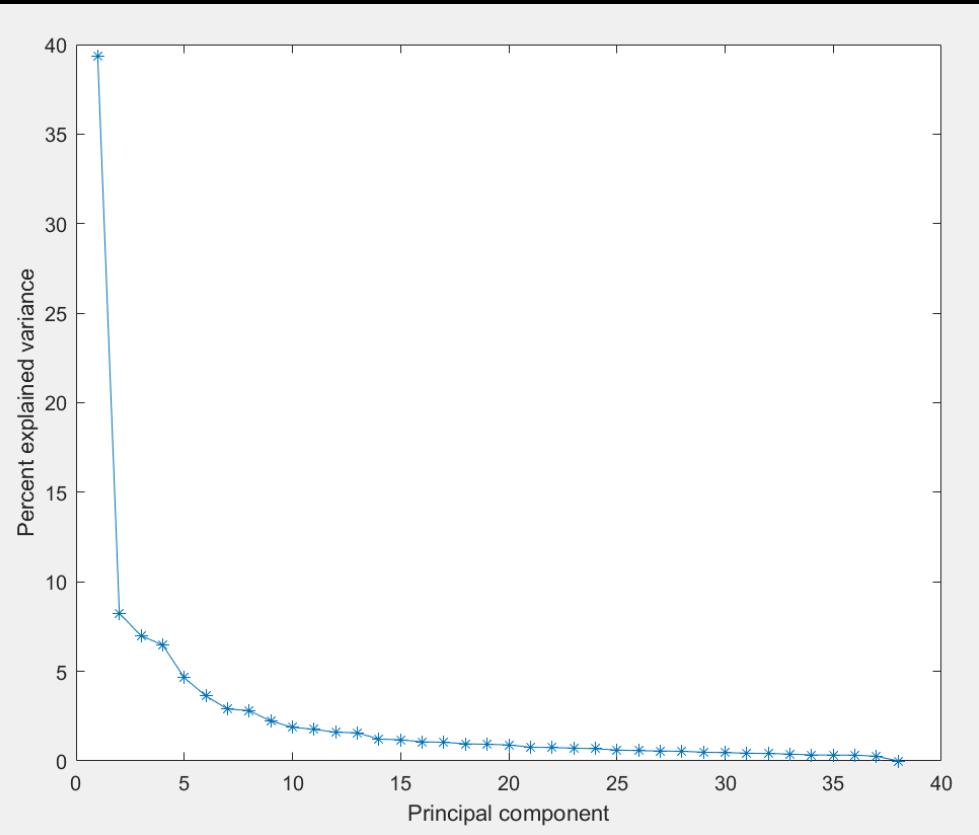


# PCA Analysis on face data

$$\mathbf{X}' = \begin{bmatrix} p_{1,1} & \cdots & p_{1,n} \\ \vdots & \ddots & \vdots \\ p_{m,1} & \cdots & p_{m,n} \end{bmatrix} - \bar{\mathbf{X}}$$

- We do the PCA analysis on the  $\mathbf{X}'$  matrix
- $\mathbf{X}'$  is  $32256 \times 38$
- Standard covariance matrix is  $32256 \times 32256$
- Turk and Pentland found a trick:
  - Compute the PCA on the  $38 \times 38$  matrix instead of the  $32256 \times 32256$  matrix
  - Details in the paper
    - Beyond the scope here

# PCA on faces



- First eigenvector explains 40% of variation
- Second eigenvector explains 8% of variation

# Visualizing the PCA faces

*Main deviations from the average face*



First PC – 40% of variation



Second PC – 8% of variation

-PC

Average face

+PC

A tool to see major variations –  
brow lifting

# Synthesizing faces

- A new face can be created by combining
  - Average face
  - Linear combination of principal components



Average face

+ 0.05



PC1

- 0.12



PC2

=



# Decomposing faces

- A given face can be reconstructed using
  - The average face
  - Linear combination of principal components
- Found by projecting the face on the principal components
- The weights can then be used for classification/identification



Average face

$+w_1$



PC1

$+w_2$



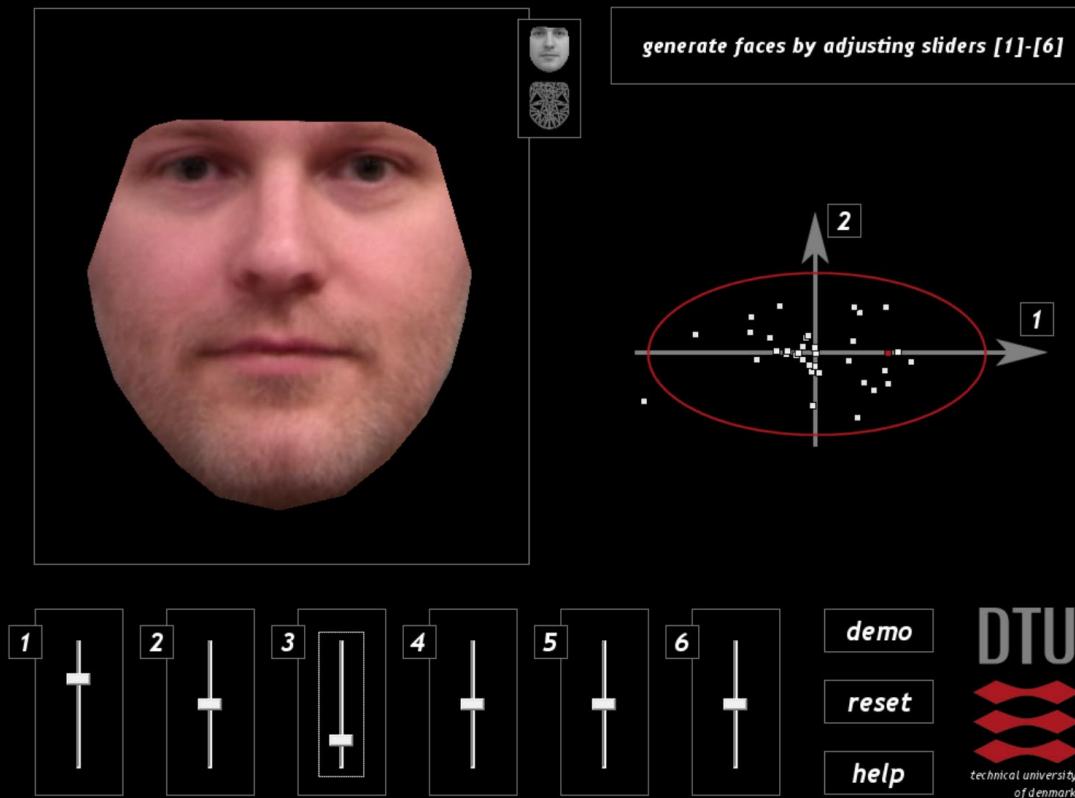
PC2

$\cong$



# Face analysis plus plus?

- More examples later in the course





# Image Analysis

Tim B. Dyrby  
Rasmus R. Paulsen  
DTU Compute

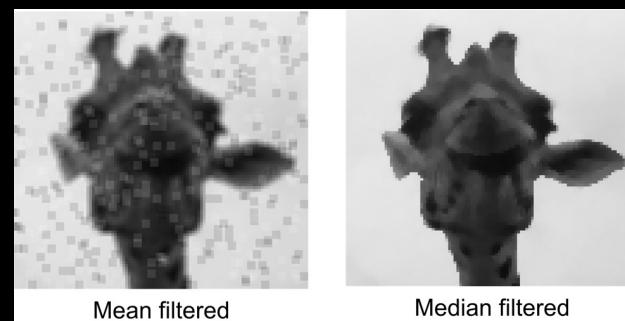
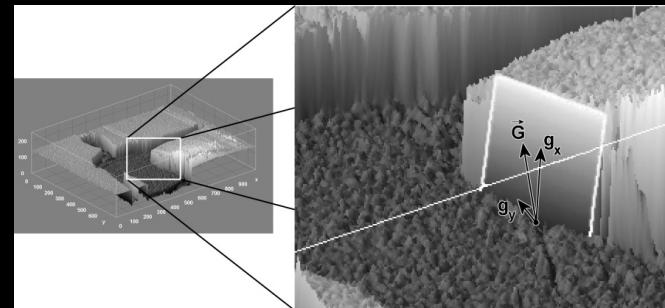
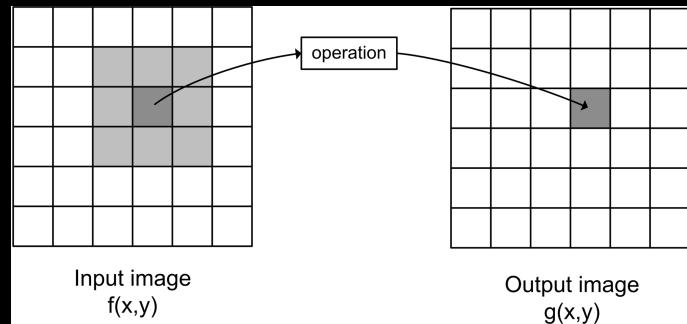
[rapa@dtu.dk](mailto:rapa@dtu.dk)

<http://www.compute.dtu.dk/courses/02502>

Plenty of slides adapted from Thomas Moeslunds lectures

# Lecture 4

## Neighbourhood Processing

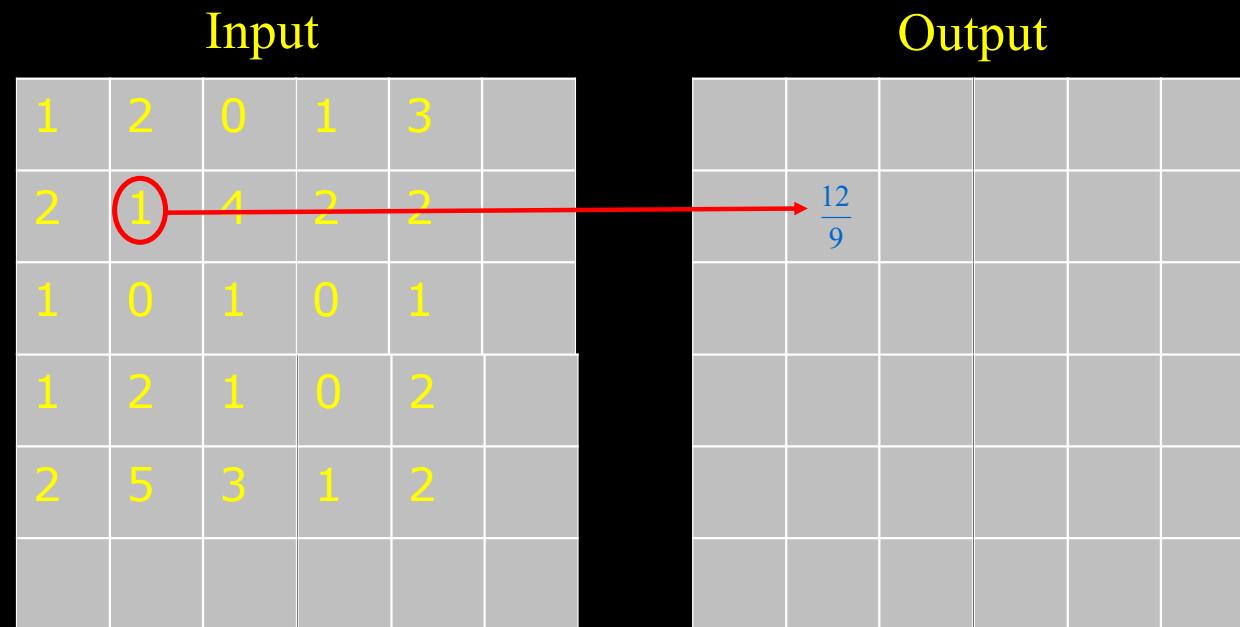




## What can you do after today?

- Describe the difference between point processing and neighbourhood processing
- Compute a rank filtered image using the min, max, median, and difference rank filters
- Compute a mean filtered image
- Decide if median or average filtering should be used for noise removal
- Choose the appropriate image border handling based on a given input image
- Implement and apply template matching
- Compute the normalised cross correlation and explain why it should be used
- Apply given image filter kernels to images
- Use edge filters on images
- Describe finite difference approximation of image gradients including the magnitude and the direction
- Compute the magnitude of the gradient
- Compute edge detection in images

# Point processing

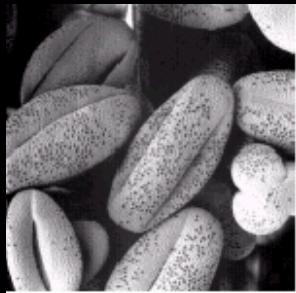


- The value of the output pixel is only dependent on the value of one input pixel
- A global operation – changes all pixels

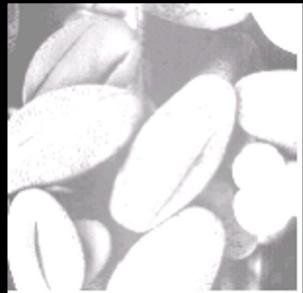
# Point processing

- Grey level enhancement
  - Process one pixel at a time independent of all other pixels
  - For example, used to correct Brightness and Contrast

Correct



Too high  
brightness



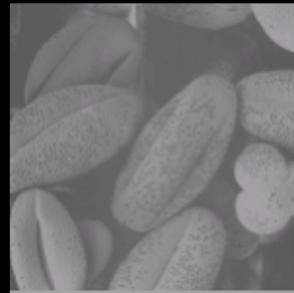
Too low  
brightness



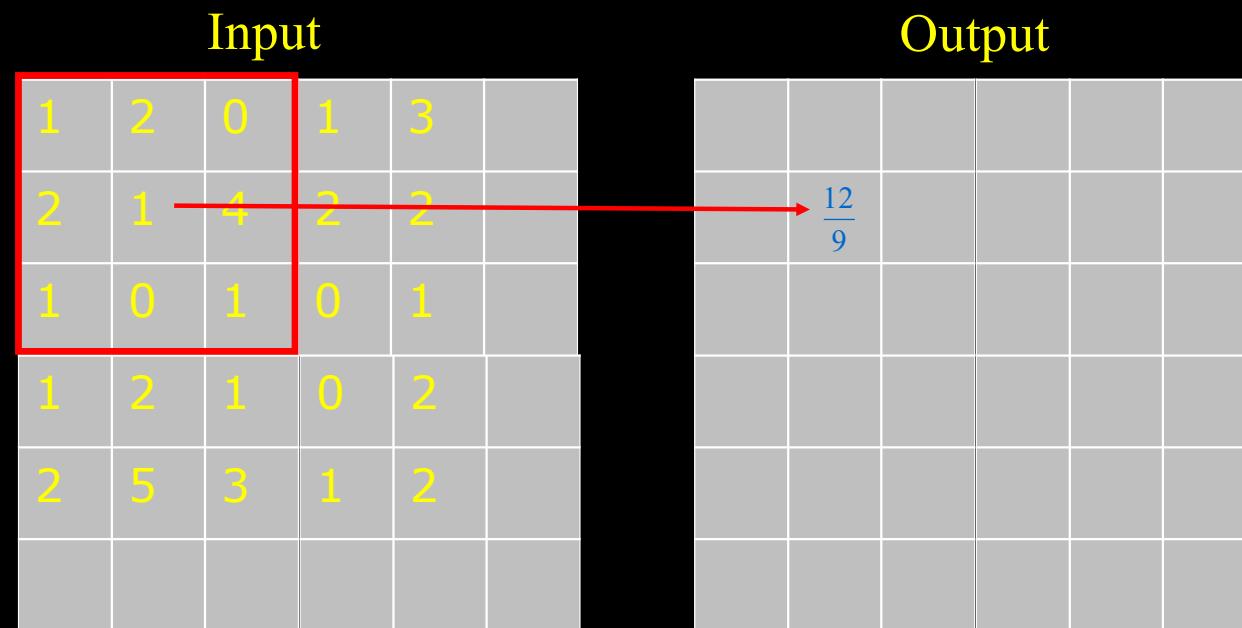
Too high  
contrast



Too low  
contrast



# Neighbourhood processing



- Several pixels in the input has an effect on the output

# Use of filtering



Noise removal



Enhance edges



Smoothing

- Image processing
- Typically done before actual image analysis

## Salt and pepper noise



- Pixel values that are very different from their neighbours
- Very bright or very dark spots
- Scratches in X-rays

What is that?

# Salt and pepper noise



- Fake example
  - Let us take a closer look at noise pixels

169	169	173	170	170	172	171	171	169
172	173	172	172	169	171	168	171	170
168	171	169	168	0	169	170	169	255
173	175	170	172	173	168	170	169	171
169	175	170	172	170	255	169	255	169
173	172	255	171	170	172	169	169	170
176	175	172	173	172	171	169	168	173
173	172	169	168	166	0	170	165	166
170	172	172	170	169	169	169	168	172
174	172	172	166	167	168	168	170	172

They are all 0 or 255

Should we just remove all the 0's and 255's from the image?

## What is so special about noise?

169	169	173	170	170	172	171	171	169
172	173	172	172	169	171	168	171	170
168	171	169	168	0	169	170	169	255
173	175	170	172	173	168	170	169	171
169	175	170	172	170	255	169	255	169
173	172	255	171	170	172	169	169	170
176	175	172	173	172	171	169	168	173
173	172	169	168	166	0	170	165	166
170	172	172	170	169	169	169	168	172
174	172	172	166	167	168	168	170	172

- What is the value of the pixel compared to the neighbours?
- Average of the neighbours
  - 170
- Can we compare to the average?
  - Difficult – should we remove all values bigger than average+1 ?
- It is difficult to detect noise!

172, 169, 171, 168, 0, 169, 172, 173, 168

# Noise – go away!



172, 169, 171, 168, 0, 169, 172, 173, 168

169, 168, 0, 170, 172, 173, 170, 172, 170

- We cannot tell what pixels are noise
- One solution
  - Set all pixels to the average or mean of the neighbours (and the pixel itself)
- Oh no!
  - Problems!
  - The noise “pollutes” the good pixels



Quiz 1: What is the median value of  
[169, 168, 0, 170, 172, 173, 170, 172, 170]?

- A) 170
- B) 173
- C) 169
- D) 171
- E) 172

## The median value

- The values are sorted from low to high
- The middle number is picked
  - The median value

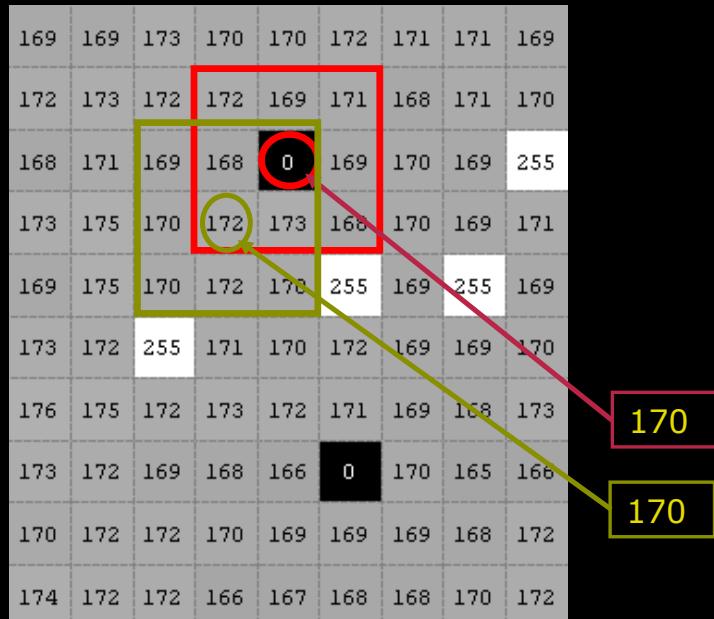
169, 168, 0, 170, 172, 173, 170, 172, 170

0, 168, 169, 170, 170, 170, 172, 172, 173

Median

Noise has no influence on the median!

## Noise away – the median filter



172, 169, 171, 168, 0, 169, 172, 173, 168

169, 168, 0, 170, 172, 173, 170, 172, 170

- All pixels are set to the median of its neighbourhood (including the pixel itself)
- Noise pixels do not pollute good pixels

## Noise removal – average filter



Scanned X-ray with salt and pepper noise



Average filter (3x3)

## Noise removal – median filter



Scanned X-ray with salt and pepper noise



Median filter (3x3)

# Image Filtering

169	169	173	170	170	172	171	171	169
172	173	172	172	169	171	168	171	170
168	171	169	168	0	169	170	169	255
173	175	170	172	173	168	170	169	171
169	175	170	172	170	255	169	255	169
173	172	255	171	170	172	169	169	170
176	175	172	173	172	171	169	168	173
173	172	169	168	166	0	170	165	166
170	172	172	170	169	169	169	168	172
174	172	172	166	167	168	168	170	172

- Creates a new *filtered* image
- Output pixel is computed based on a neighbourhood in the input image
- 3 x 3 neighbourhood
  - Filter size 3 x 3
  - Kernel size 3 x 3
  - Mask size 3 x 3
- Larger filters often used
  - Size
    - 7 x 7
  - Number of elements
    - 49

## Quiz 2: Median filter on image

- A) 25
- B) 90
- C) 198
- D) 86
- E) 103

Answer:

4, 25, 34, 86, 90, 103, 125, 209, 230

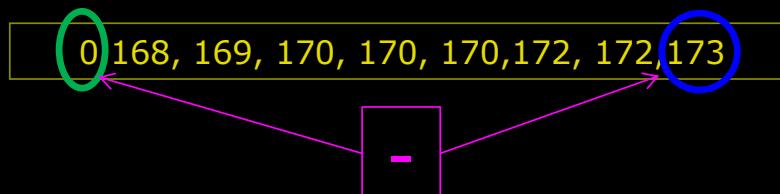
The image is filtered with a  $3 \times 3$  median filter. What is the result in the pixel marked with a circle?

66	222	102	230	199	147	166	175
204	148	19	241	99	15	187	47
110	140	61	125	62	60	165	94
232	37	31	125	103	90	115	160
46	218	47	86	25	209	139	199
67	159	61	230	34	4	76	21
37	89	106	94	240	11	190	237
35	131	13	28	244	43	48	198

## Rank filters

169	169	173	170	170	172	171	171	169
172	173	172	172	169	171	168	171	170
168	171	169	168	0	169	170	169	255
173	175	170	172	173	168	170	169	171
169	175	170	172	170	255	169	255	169
173	172	255	171	170	172	169	169	170
176	175	172	173	172	171	169	168	173
173	172	169	168	166	0	170	165	166
170	172	172	170	169	169	169	168	172
174	172	172	166	167	168	168	170	172

- Based on sorting the pixel values in the neighbouring region as the median filter
- Minimum rank filter
  - Darker image. Noise problems.
- Maximum rank filter
  - Lighter image. Noise problems.
- Difference filter
  - Enhances changes (edges)





## Quiz 3: Rank filters on image

- A) 3
- B) 84**
- C) 112
- D) 73
- E) 202

Answer:

medI: 

15	60	62	90	99	103	115	165	187

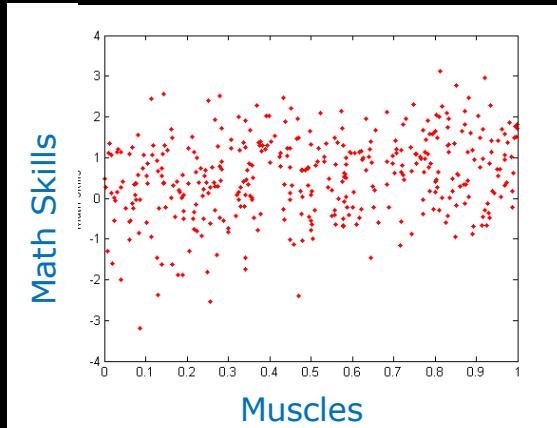
minI: 

15
----

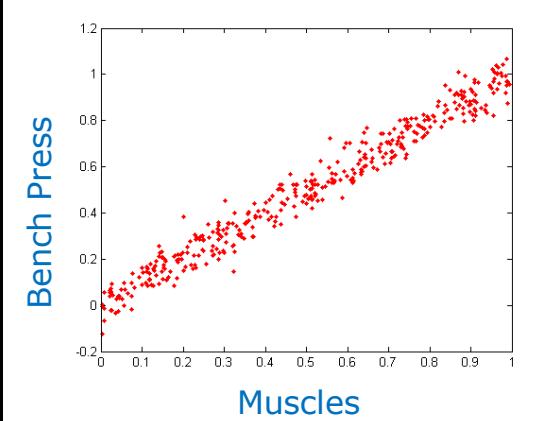
The image is filtered with a  $3 \times 3$  median filter (medI). The image (the original) is also filtered with a  $5 \times 5$  minimum rank filter (minI). The final image is made by subtracting minI from medI. What is the result in the marked pixel?

67	159	61	230	34	4	76	21
37	89	106	94	240	11	190	237
35	131	13	28	244	43	48	198
222	102	230	199	147	166	175	124
148	19	241	99	15	187	47	111
140	61	125	62	60	165	94	114
37	31	125	103	90	115	160	78
218	47	86	25	209	139	199	130

# Correlation



Low

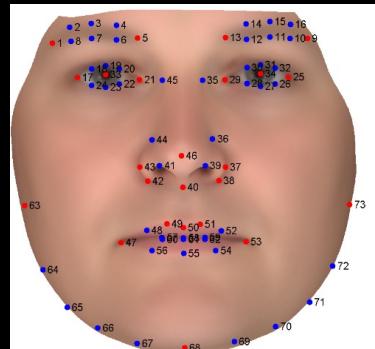


High

- What is it?
- Two measurements
  - Low correlation
  - High correlation
- High correlation means that there is a *relation* between the values
- They *look* the same
- Correlation is a *measure of similarity*

## Why do we need similarity?

- Image analysis is also about recognition of patterns
- Often an example pattern is used
  - Often with some kind of meta data to apply to the targets
- We need something to tell us if there is a high match between our pattern and a part of the image

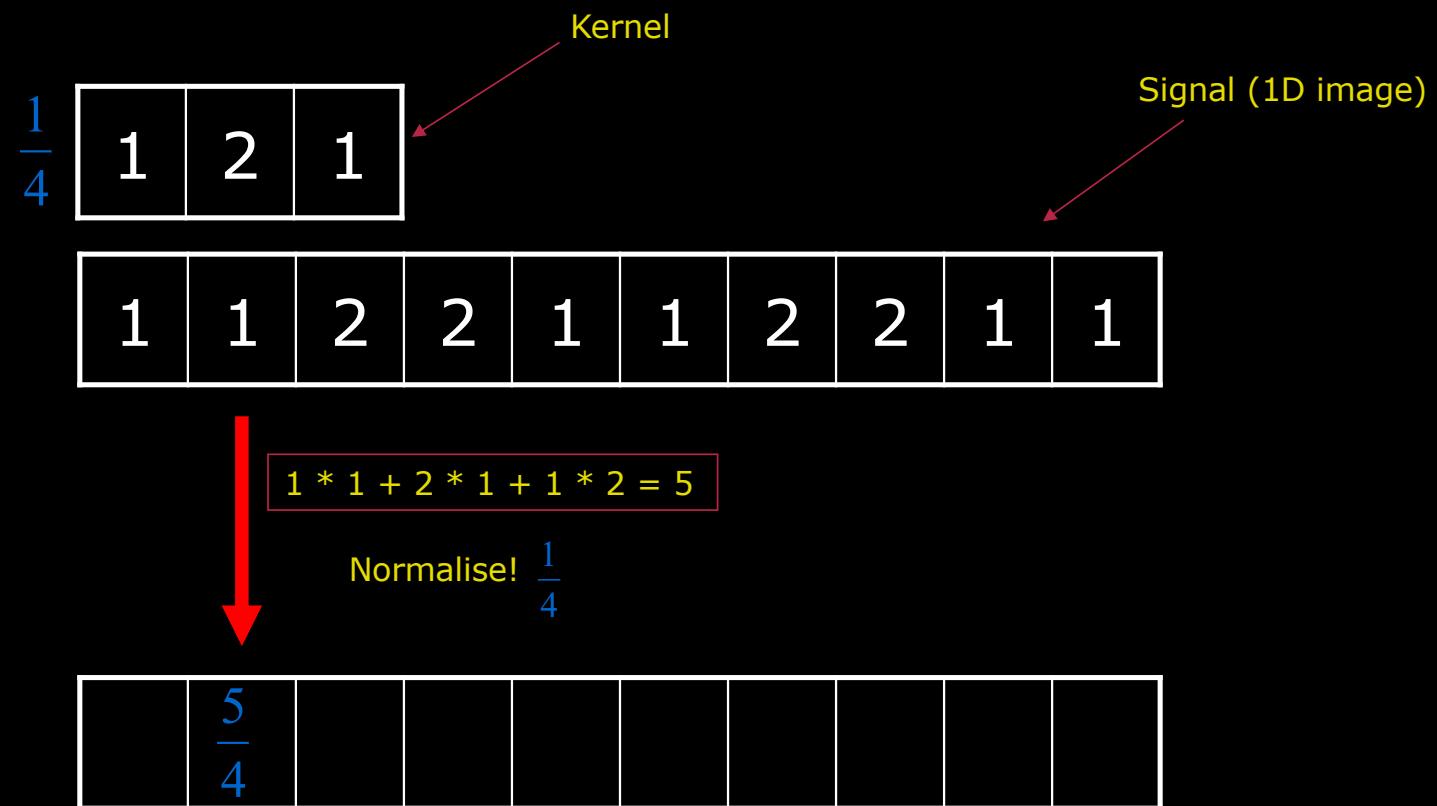


Example pattern  
With meta information

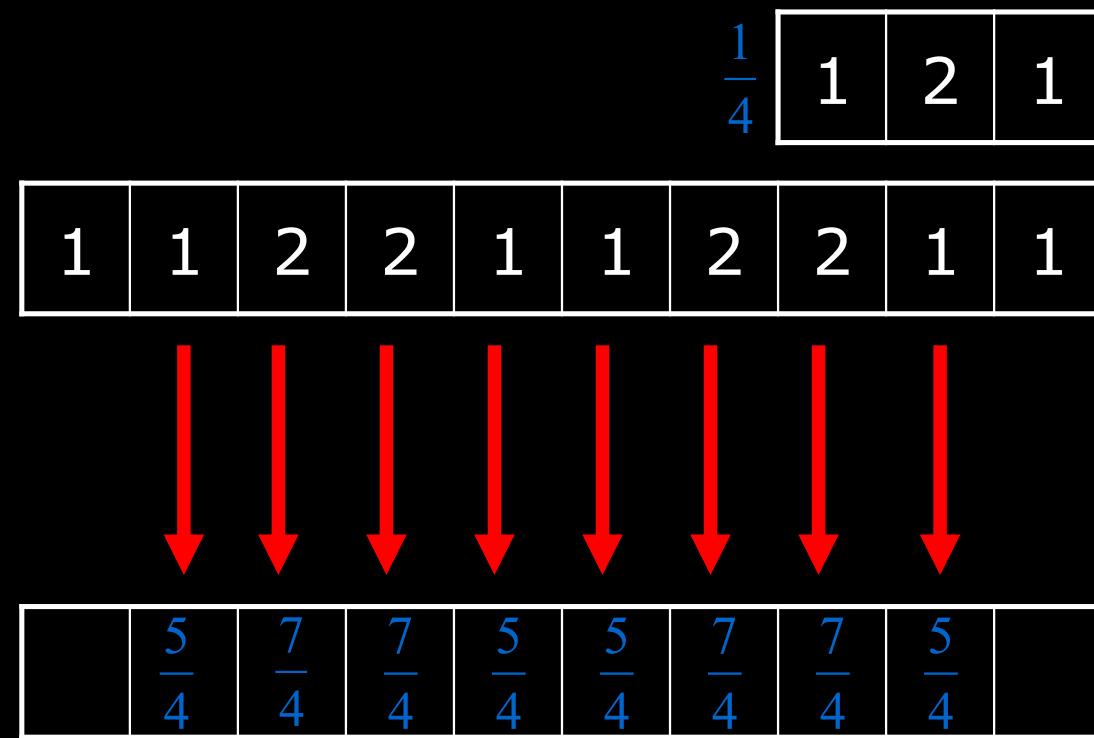
Find  
matches



# Correlation (1D)



## Correlation (1D)



# Normalisation

- The sum of the kernel elements is used
- Keep the values in the same range as the input image

$$\frac{1}{4} \begin{bmatrix} 1 & 2 & 1 \end{bmatrix}$$

Sum is 4

$$\begin{bmatrix} 1 & 1 & 2 & 2 & 1 & 1 & 2 & 2 & 1 & 1 \end{bmatrix}$$


$$1 * 1 + 2 * 1 + 1 * 2 = 5$$

Normalise!  $\frac{1}{4}$

$$\begin{bmatrix} & \frac{5}{4} & & & & & & & & \end{bmatrix}$$



# Normalisation

$$h(x) \begin{array}{|c|c|c|} \hline 1 & 2 & 1 \\ \hline \end{array}$$

- Normalisation factor
  - Sum of kernel coefficients

$$\sum_x h(x) = 1 + 2 + 1$$

## Correlation on images

- The filter is now 2D

Kernel coefficients

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

1	1	1
1	1	1
1	1	1

Kernel

Input

1	2	0	1	3	1
2	1	4	2	2	2
1	0	1	0	1	3
1	2	1	0	2	4
2	5	3	1	2	2
2	1	3	1	6	3

Output


$$\frac{12}{9}$$

# Correlation on images

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

# Input

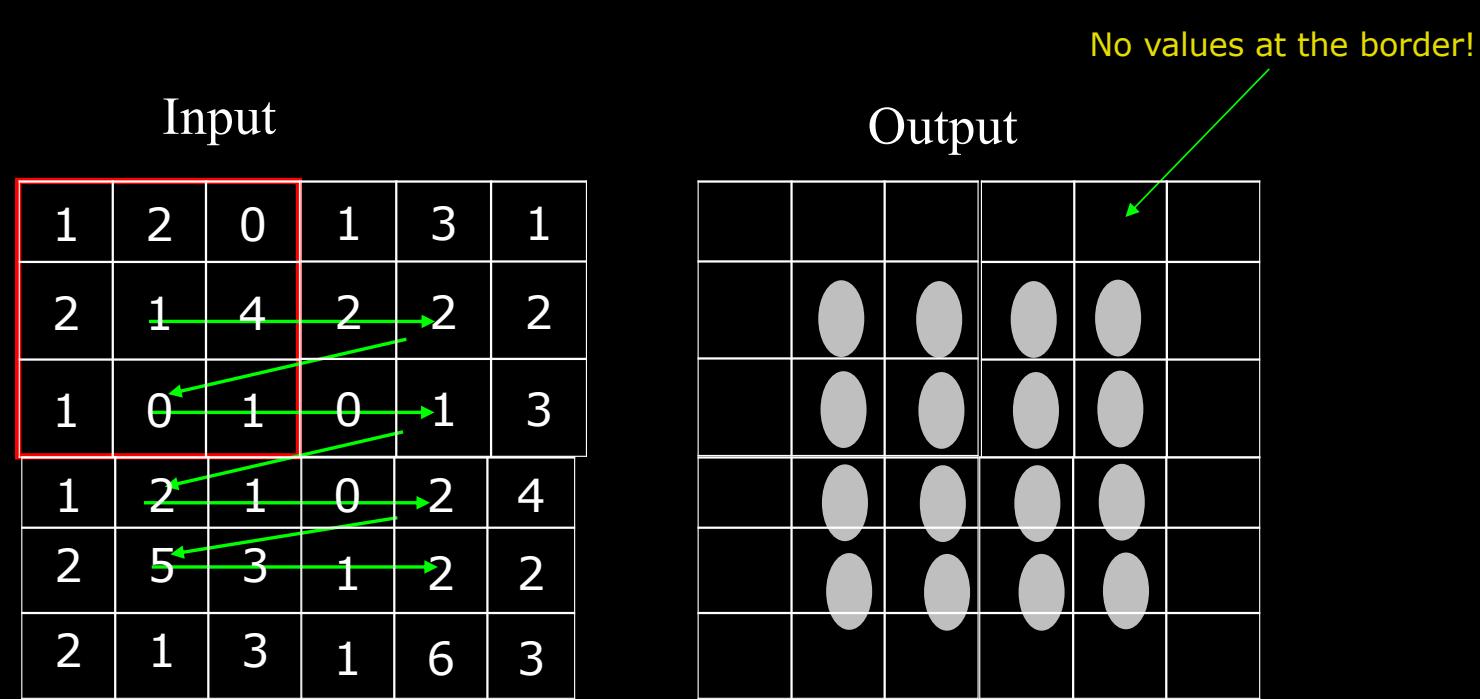
1	2	0	1	3	1
2	1	4	2	2	2
1	0	1	0	1	3
1	2	1	0	2	4
2	5	3	1	2	2
2	1	3	1	6	3

## Output

	$\frac{12}{9}$		$\frac{11}{9}$	

# Correlation on images

The mask is moved row by row



## Quiz 4: Correlation on image – no normalisation

- A) 4
- B) 7
- C) 16
- D) 23
- E) 25

1	2	1
1	3	1
1	2	1

1	2	0	1	3	1
2	1	4	2	2	2
1	0	1	0	1	3
1	2	1	0	2	4
2	5	3	1	2	2
2	1	3	1	6	3

## Quiz 5: Correlation on image 2 – no normalisation

- A) 0,10
- B) 3, 3
- C) 6, 2
- D) 10, 15
- E) 11, 32

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

1	2	0	1	3	1
2	1	4	2	2	2
1	0	1	0	1	3
1	2	1	0	2	4
2	5	3	1	2	2
2	1	3	1	6	3

## Mathematics of 2D Correlation

$$g(x, y) = f(x, y) \circ h(x, y)$$

Correlation operator

1	2	0	1	3	1
2	1	4	2	2	2
1	0	1	0	1	3
1	2	1	0	2	4
2	5	3	1	2	2
2	1	3	1	6	3

**f**

1	2	1
1	3	1
1	2	1

**h**

## Mathematics of 2D Correlation

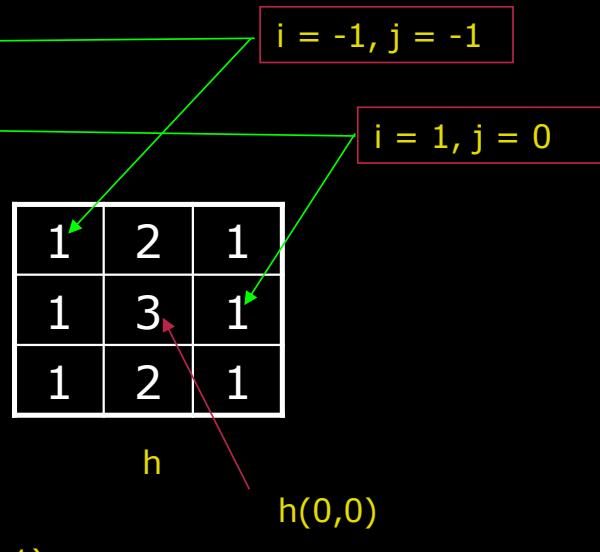
$$g(x, y) = \sum_{j=-R}^R \sum_{i=-R}^R h(i, j) \cdot f(x + i, y + j)$$

Example  $g(2,1)$

1	2	0	1	3	1
2	1	4	2	2	2
1	0	1	0	1	3
1	2	1	0	2	4
2	5	3	1	2	2
2	1	3	1	6	3

$f$

$f(2,1)$



## Mathematics of 2D Correlation

$$g(x, y) = \sum_{j=-R}^R \sum_{i=-R}^R h(i, j) \cdot f(x + i, y + j)$$

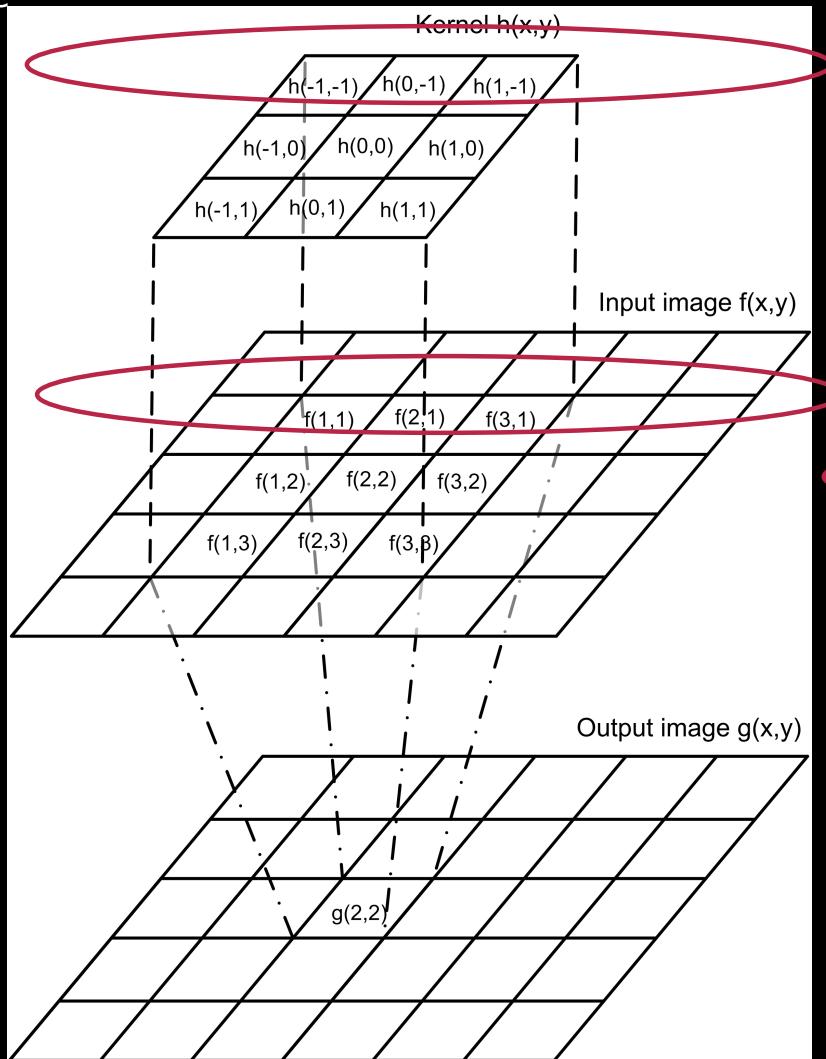
$$g(x, y) = 1 \cdot 2 + 2 \cdot 0 + 1 \cdot 1 + 1 \cdot 1 + 3 \cdot 4 + 1 \cdot 2 + 1 \cdot 0 + 2 \cdot 1 + 1 \cdot 0$$

1	2	0	1	3	1
2	1	4	2	2	2
1	0	1	0	1	3
1	2	1	0	2	4
2	5	3	1	2	2
2	1	3	1	6	3

$f$

1	2	1
1	3	1
1	2	1

$h$



$$g(x,y) = \sum_{j=-R}^R \sum_{i=-R}^R h(i,j) \cdot f(x+i, y+j)$$

$$g(2,2) =$$

$$h(-1,-1) \cdot f(1,1) + h(0,-1) \cdot f(2,1) + h(1,-1) \cdot f(3,1) +$$

$$h(-1,0) \cdot f(1,2) + h(0,0) \cdot f(2,2) + h(1,0) \cdot f(3,2) +$$

$$h(-1,1) \cdot f(1,3) + h(0,1) \cdot f(2,3) + h(1,1) \cdot f(3,3)$$

## Correlation is a dot-product between two vectors

$$g(x, y) = \sum_{j=-R}^R \sum_{i=-R}^R h(i, j) \cdot f(x + i, y + j)$$

1	2	0	1	3	1
2	1	4	2	2	2
1	0	1	0	1	3
1	2	1	0	2	4
2	5	3	1	2	2
2	1	3	1	6	3

1	2	1
1	3	1
1	2	1

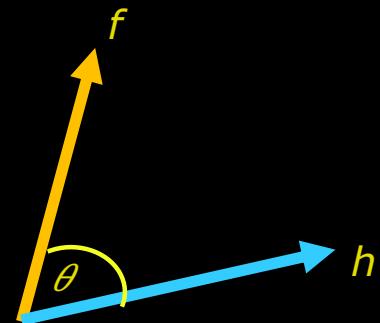
h

- In correlation we multiply  $h$  and  $f$
- Kernel  $h$  and 3x3 patch in figure  $f$  are two high dimensional vectors:

$$f(2,1) = [2, 0, 1, 1, 4, 2, 0, 1, 0]$$

$$h = [1, 2, 1, 1, 3, 1, 1, 2, 1]$$

- Multiplication is a dot product
  - $h \cdot f = \|h\| \|f\| \cos \theta$



## 2D Kernel Normalisation

Normalisation factor:

$$\sum_x \sum_y h(x, y)$$

$$1 + 2 + 1 + 1 + 3 + 1 + 1 + 2 + 1 = 13$$

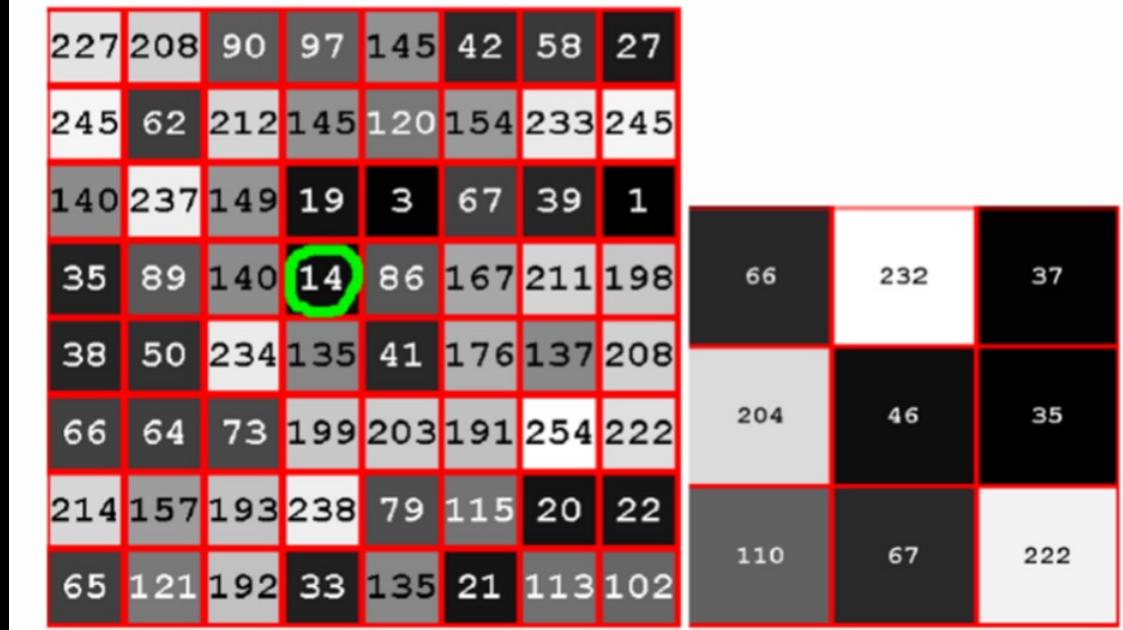
1	2	1
1	3	1
1	2	1

$h$

## Quiz 6: Template match on image

- A) 50122
- B) 123001
- C) 11233
- D) 2550
- E) 90454

A template match is done on the image to the left with the template seen to the right. To find the best match the correlation is computed. What is the correlation in the marked pixel?



# Smoothing filters

- Also know as
  - Smoothing kernel, Mean filter, Low pass filter, blurring
- The simplest filter:
  - *Spatial low pass filter*
  - Removes high frequencies
- Another mask:
  - Gaussian filter

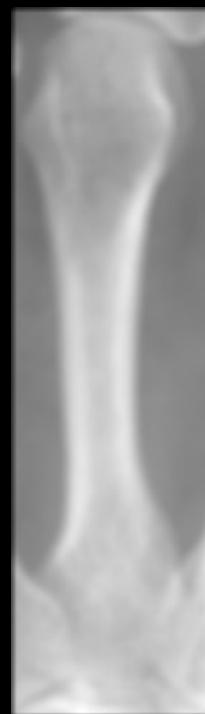
$$\frac{1}{9} \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$

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

## Use of smoothing



3x3



7x7

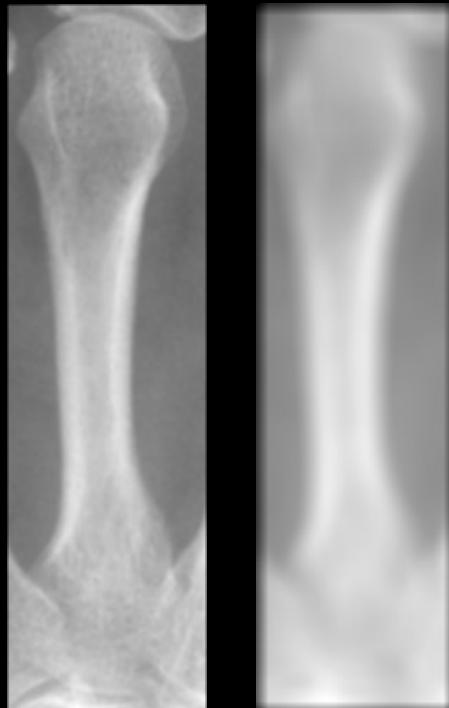


11x11



15x15

## Use of smoothing



- Large kernels smooth more
- Removes high frequency information
- Good at enhancing *big structures*

## Quiz 7: Mean filter on image

- A) 166
- B) 113
- C) 12
- D) 51
- E) 245

Answer:

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

$$(608+x*1)/9=86 \rightarrow x=166$$

A 3x3 mean filter is applied to the image. The result in the marked pixel is 86. What is the value of the pixel, where the value is missing?

227	208	90	97	145	42	58	27
245	62	212	145	120	154	233	245
140	237	149	19	3	67	39	1
35	89	140	14	86		211	198
38	50	234	135	41	176	137	208
66	64	73	199	203	191	254	222
214	157	193	238	79	115	20	22
65	121	192	33	135	21	113	102

# Border handling

Input

1	2	0	1	3	1
2	1	4	2	2	2
1	0	1	0	1	3
1	2	1	0	2	4
2	5	3	1	2	2
2	1	3	1	6	3

Output

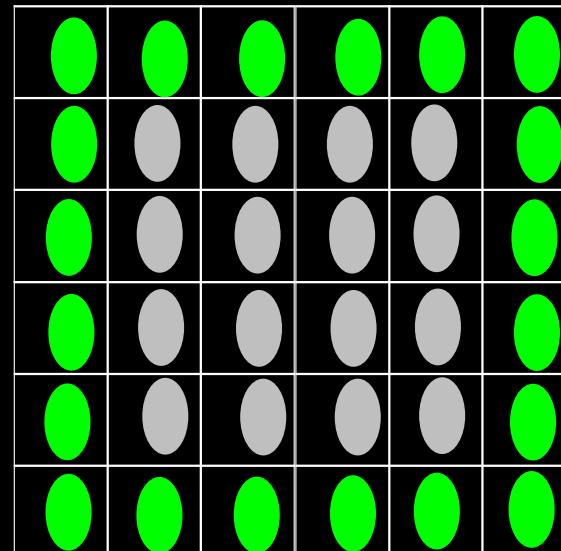

No values at the border!

## Border handling – extend the input

Input

0	0	0	0	0	0	0
0	1	2	0	1	3	1
0	2	1	4	2	2	2
0	1	0	1	0	1	3
0	1	2	1	0	2	4
2	5	3	1	2	2	2
2	1	3	1	6	3	

- Zero padding – what happens?
- Zero is black – creates dark border around the image



## Quiz 8: Correlation on image with zero padding

- A) 5, 8
- B) 12, 16
- C) 13, 3
- D) 15, 21
- E) 17, 12

1	2	1
1	3	1
1	2	1

1	2	0	1	3	1
2	1	4	2	2	2
1	0	1	0	1	3
1	2	1	0	2	4
2	5	3	1	2	2
2	1	3	1	6	3

## Quiz 8: Correlation on image with zero padding

- A) 5, 8
- B) 12, 16
- C) 13, 3
- D) 15, 21
- E) 17, 12

Answers:

1	2	1
1	3	1
1	2	1

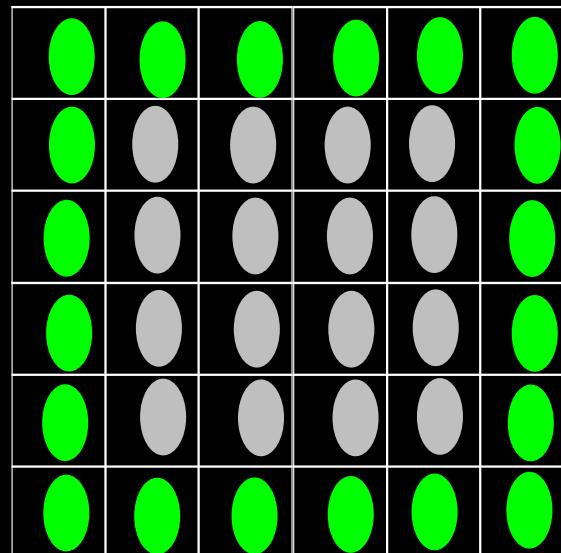
1	2	0	1	3	1
2	1	4	2	2	2
1	0	1	0	1	3
1	2	1	0	2	4
2	5	3	1	2	2
2	1	3	1	6	3

## Border handling – extend the input

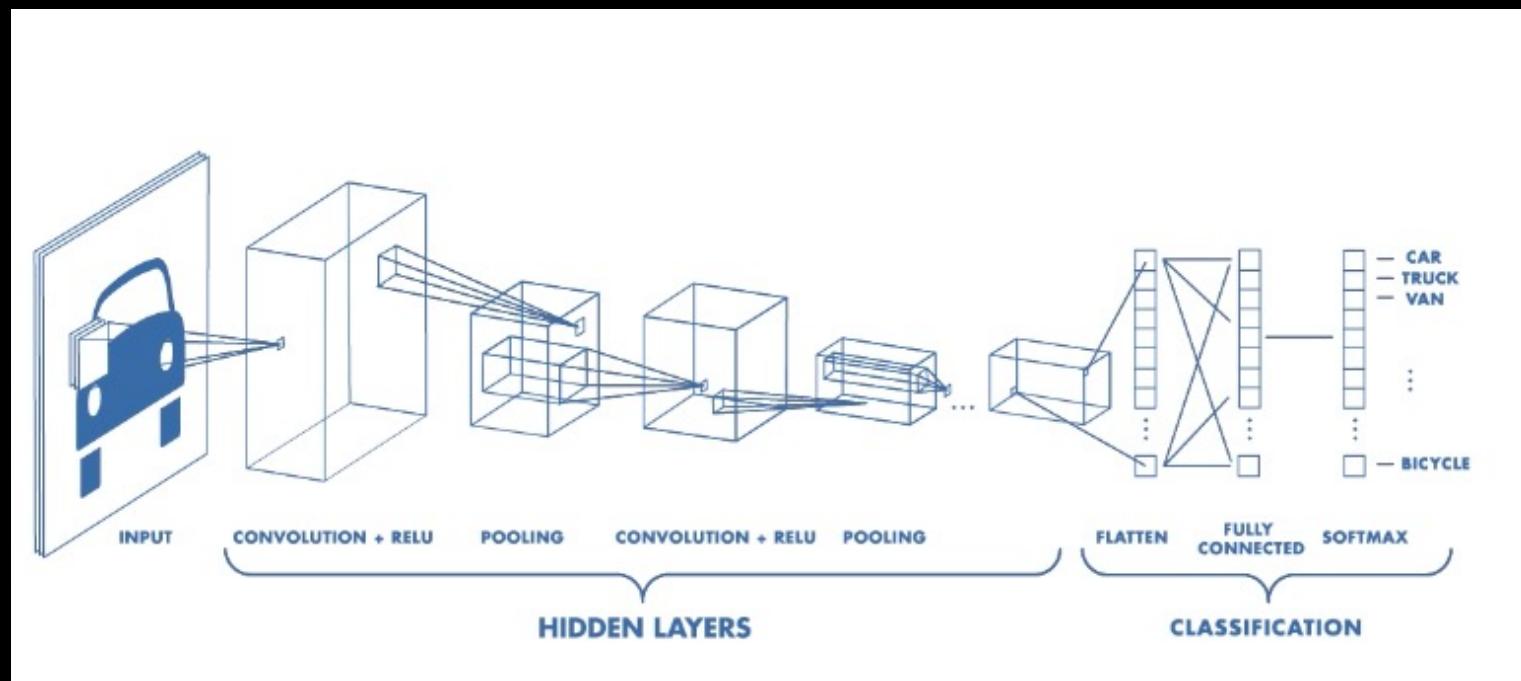
Input

1	1	2	0	1	3	
1	1	2	0	1	3	1
2	2	1	4	2	2	2
1	1	0	1	0	1	3
1	1	2	1	0	2	4
2	5	3	1	2	2	
2	1	3	1	6	3	

- Reflection
- Normally better than zero padding



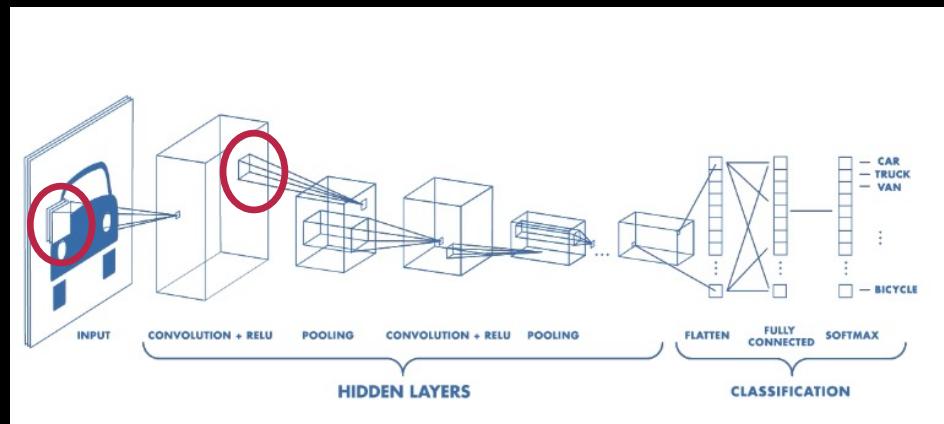
# What is the connection to deep learning?



<https://se.mathworks.com/videos/introduction-to-deep-learning-what-are-convolutional-neural-networks--1489512765771.html>

## Banks of filters

- The part of the network that touches the image consists of a bank of filters i.e., kernel coefficients
  - Organised in a multi-level hierarchy
- The weights of the filters are adapted to the problem



# Template Matching

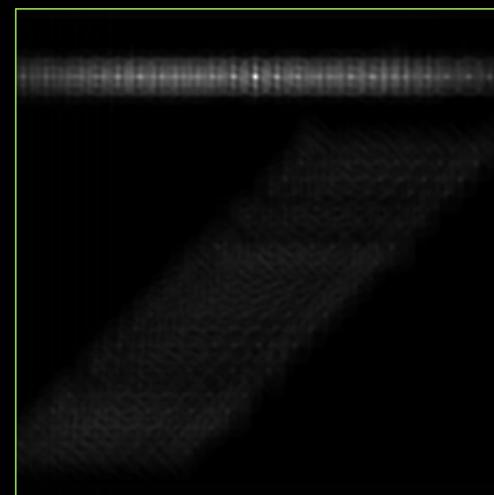
- Template
  - *Skabelon* på dansk
- Locates objects in images



Input

processing

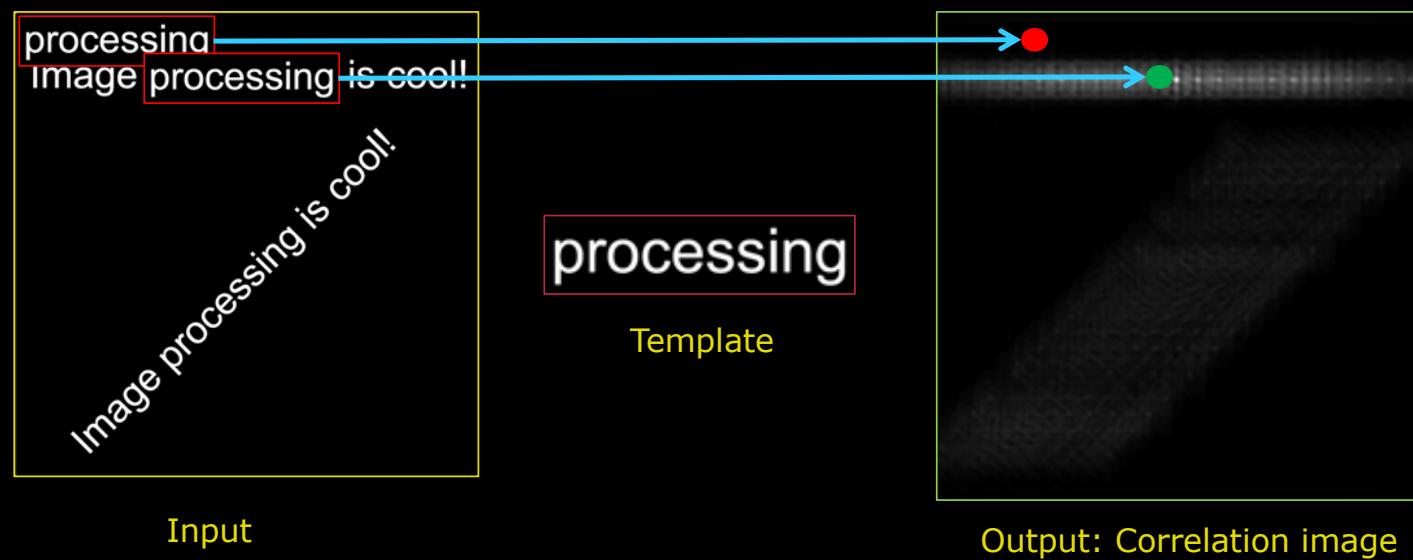
Template



Output: Correlation image

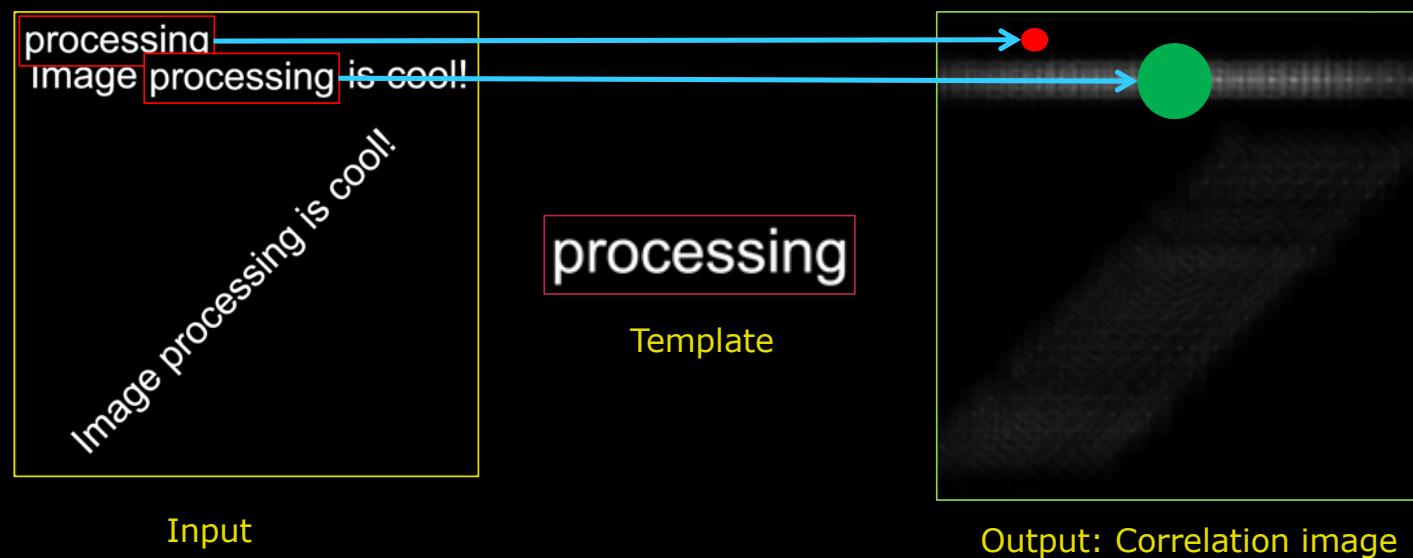
# Template Matching

- The correlation between the template and the input image is computed for each pixel



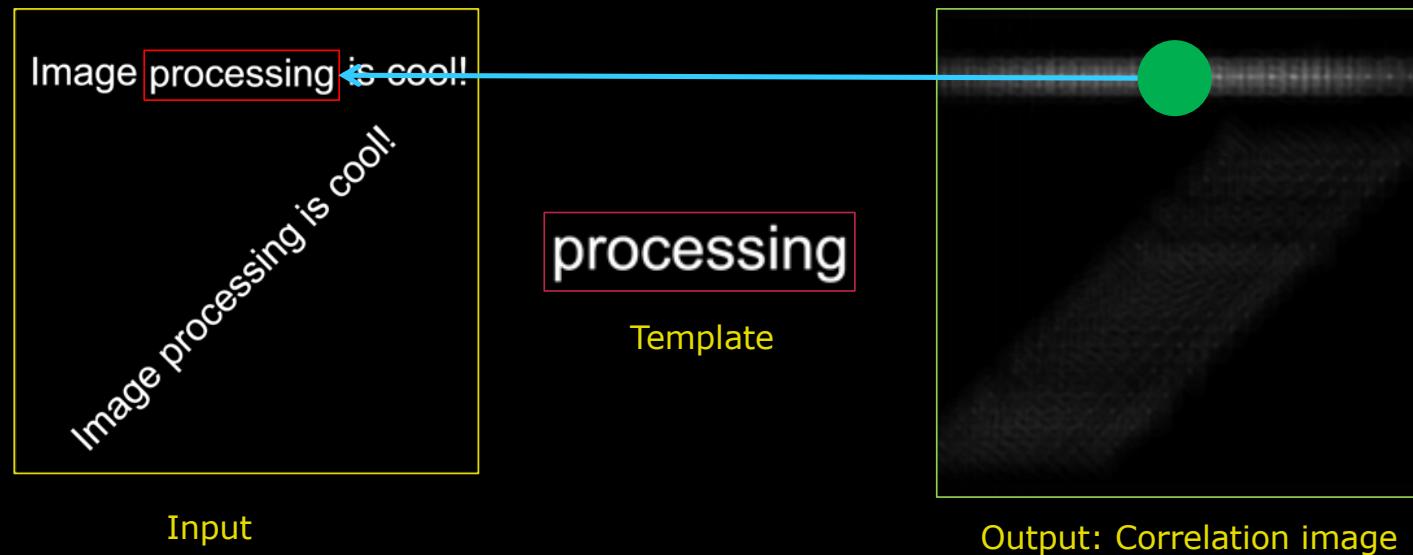
# Template Matching

- The pixel with the highest value is found in the output image
  - Here is the highest correlation



# Template Matching

- This corresponds to the found pattern in the input image

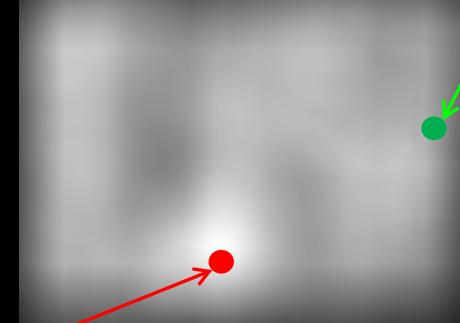


## Problematic Correlation

- Correlation matching has problem with light areas – why?

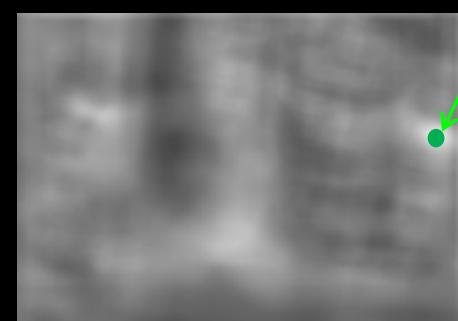
$$g(x, y) = \sum_{j=-R}^R \sum_{i=-R}^R h(i, j) \cdot f(x + i, y + j)$$

Very High!



# Normalised Cross Correlation

$$\text{NCC}(x, y) = \frac{\text{Correlation}}{\text{Length of image patch} \cdot \text{Length of template}}$$

Input ( $f$ )Template ( $h$ )

Output: Correlation image

Real max

## Length of template

- Vector length (or magnitude)
  - Put all pixel values into a vector
  - Compute the length of this vector
- Describes the intensity of the template
  - Bright template has a large length
  - Dark template has a small length

$$\text{Length of template} = \sqrt{\sum_{j=-R}^R \sum_{i=-R}^R h(i,j) \cdot h(i,j)}$$



Template ( $h$ )

## Length of image patch

- Vector length based on pixel values in image patch
- Describes the intensity of the image patch



Input (f) with patch



Template (h)



## Normalised Cross Correlation

- The length of the image patch ( $\|f\|$ ) and the length of template ( $\|h\|$ ) normalises the correlation
- Correlation is a dot-product i.e., correlation =  $h \cdot f = \|h\| \|f\| \cos \theta$
- Normalised dot product is the angle between vector  $f$  and  $h$

$$\text{NCC}(x,y) = \frac{\text{Correlation}}{\text{Length of image patch} \cdot \text{Length of template}}$$

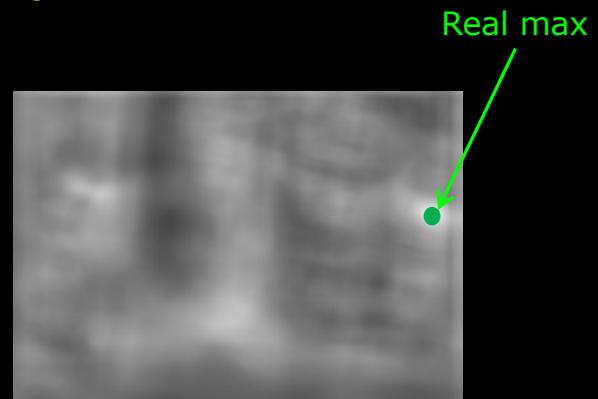
$$\text{NCC}(x,y) = \frac{\|h\| \|f\| \cos \theta}{\|h\| \|f\|}$$

- Normalised dot product:  $\text{NCC}(x,y) = \cos \theta$

# Normalised Cross Correlation

- NCC will be between
  - **0** : No similarity between template and image patch
  - **1** : Template and image patch are identical

$$\text{NCC}(x, y) = \frac{\text{Correlation}}{\text{Length of image patch} \cdot \text{Length of template}} = \cos \theta$$

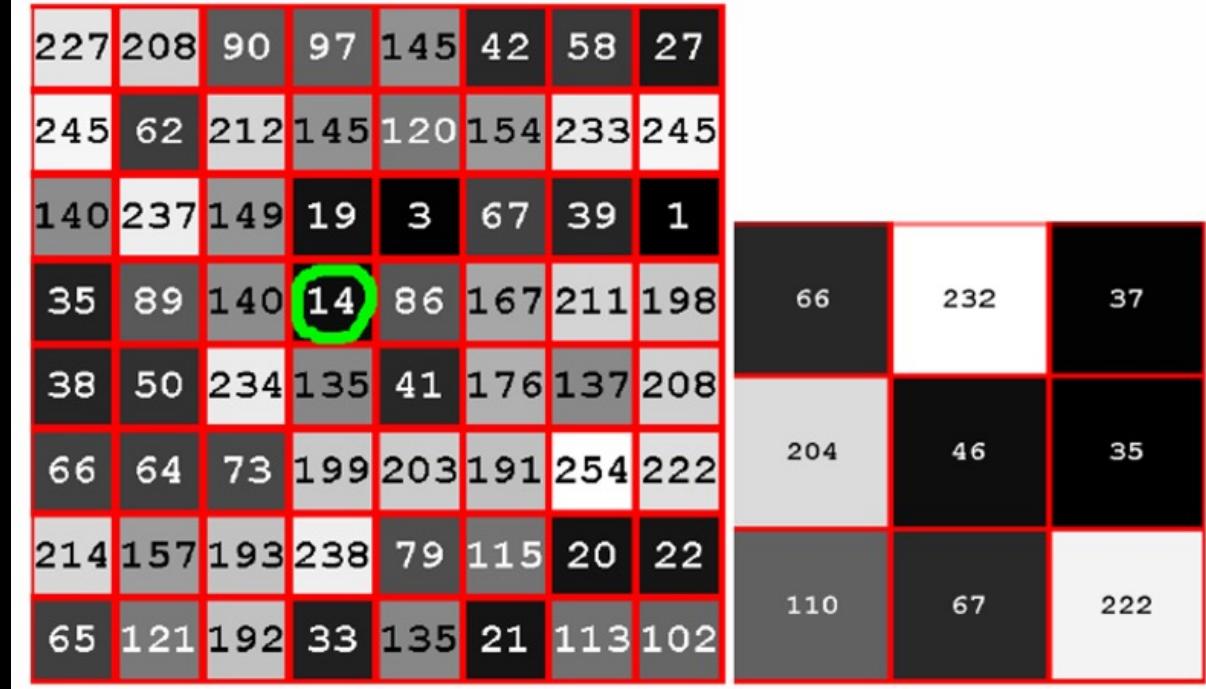
Input ( $f$ )Template ( $h$ )

Output: Correlation image

## Quiz 9: Normalised cross correlation on image

A template match using normalised cross correlation is performed. What is the resulting value in the marked pixel?

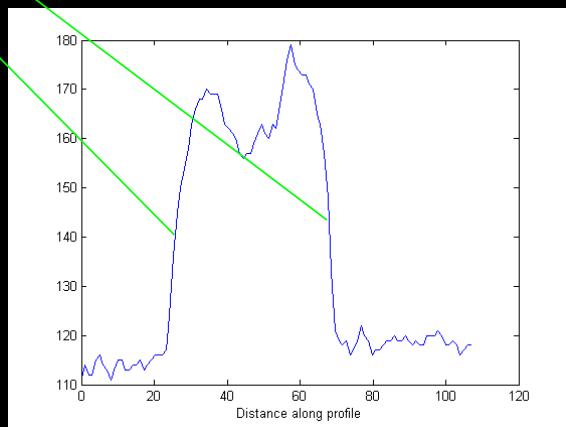
- A) 0.10
- B) 0.33
- C) 0.83
- D) 0.62
- E) 0.98



# Edges



Gray level profile

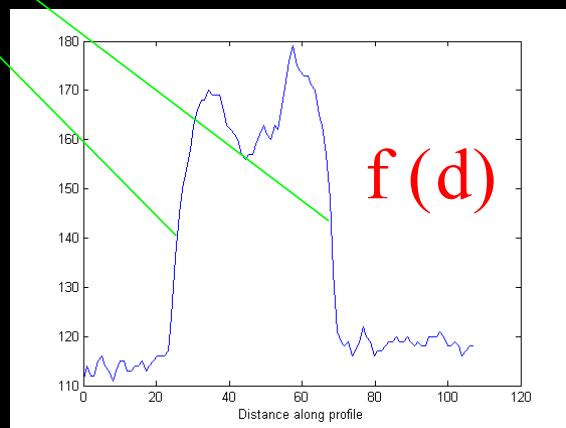


- An edge is where there is a high change in gray level values
- Objects are often separated from the background by edges

# Edges



$$f'(d)$$



- The profile as a function  $f(d)$
- What value is high when there is an edge?
  - The slope of  $f$
  - The slope of the tangent at  $d$



# Finite Difference

## ■ Definition of slope

$$f'(d) = \lim_{h \rightarrow 0} \frac{f(d + h) - f(d)}{h}$$

## ■ Approximation

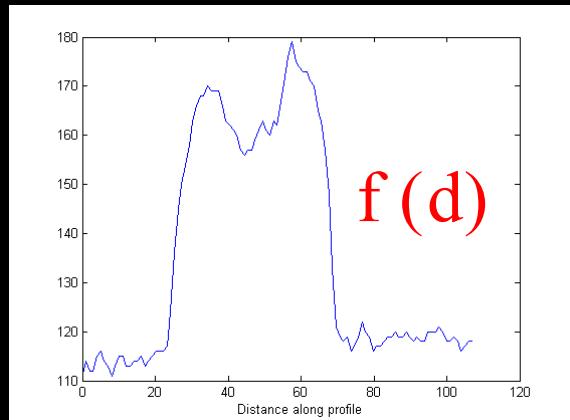
$$f'(d) \approx \frac{f(d + h) - f(d)}{h}$$

## ■ Simpler approximation

$$f'(d) \approx f(d + 1) - f(d)$$

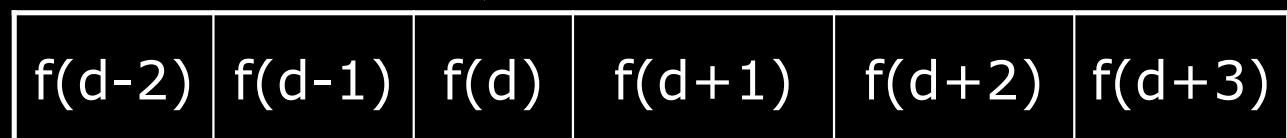
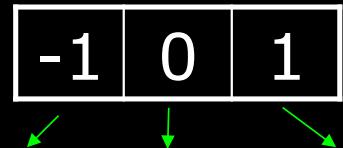
$$h = 1$$

# Edges

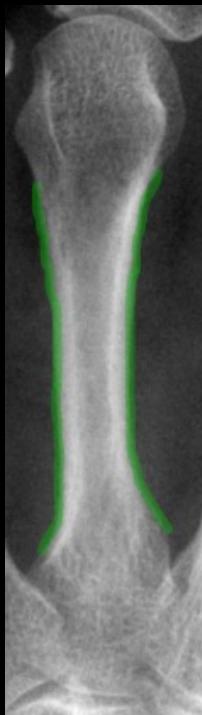


- Discrete approximation of  $f'(d)$
- Central differences is used:  $d=+/-1$
- Can be implemented as a filter

$$f'(d) \approx f(d \pm 1) - f(d)$$



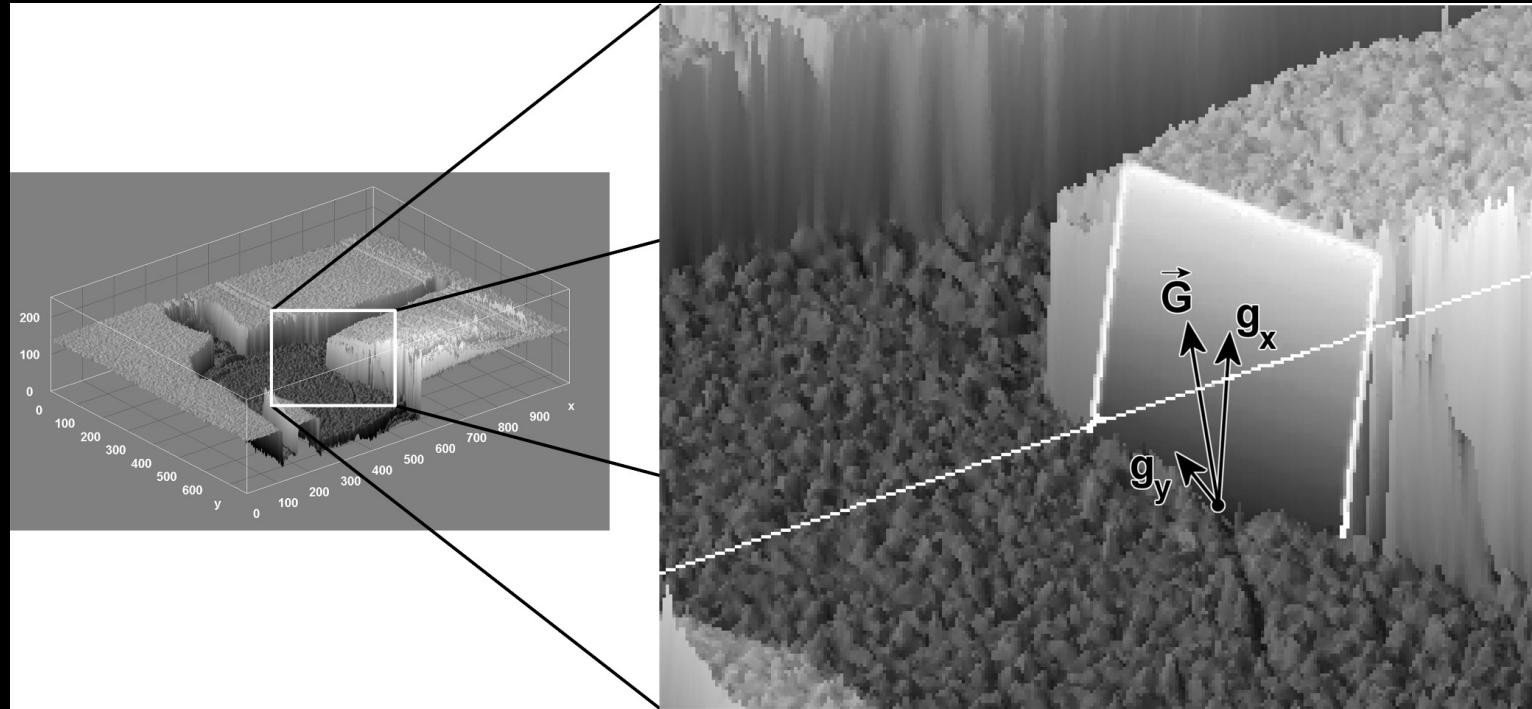
# Edges in 2D



- Changes in gray level values
  - Image gradient
  - Gradient is the 2D derivative of a 2D function  $f(x,y)$
  - Equal to the *slope* of the image
  - A steep slope is equal to an edge

$$\nabla f(x, y) = \vec{G}(g_x, g_y)$$

## 2D Gradient



$$\text{magnitude} = \sqrt{g_x^2 + g_y^2}$$

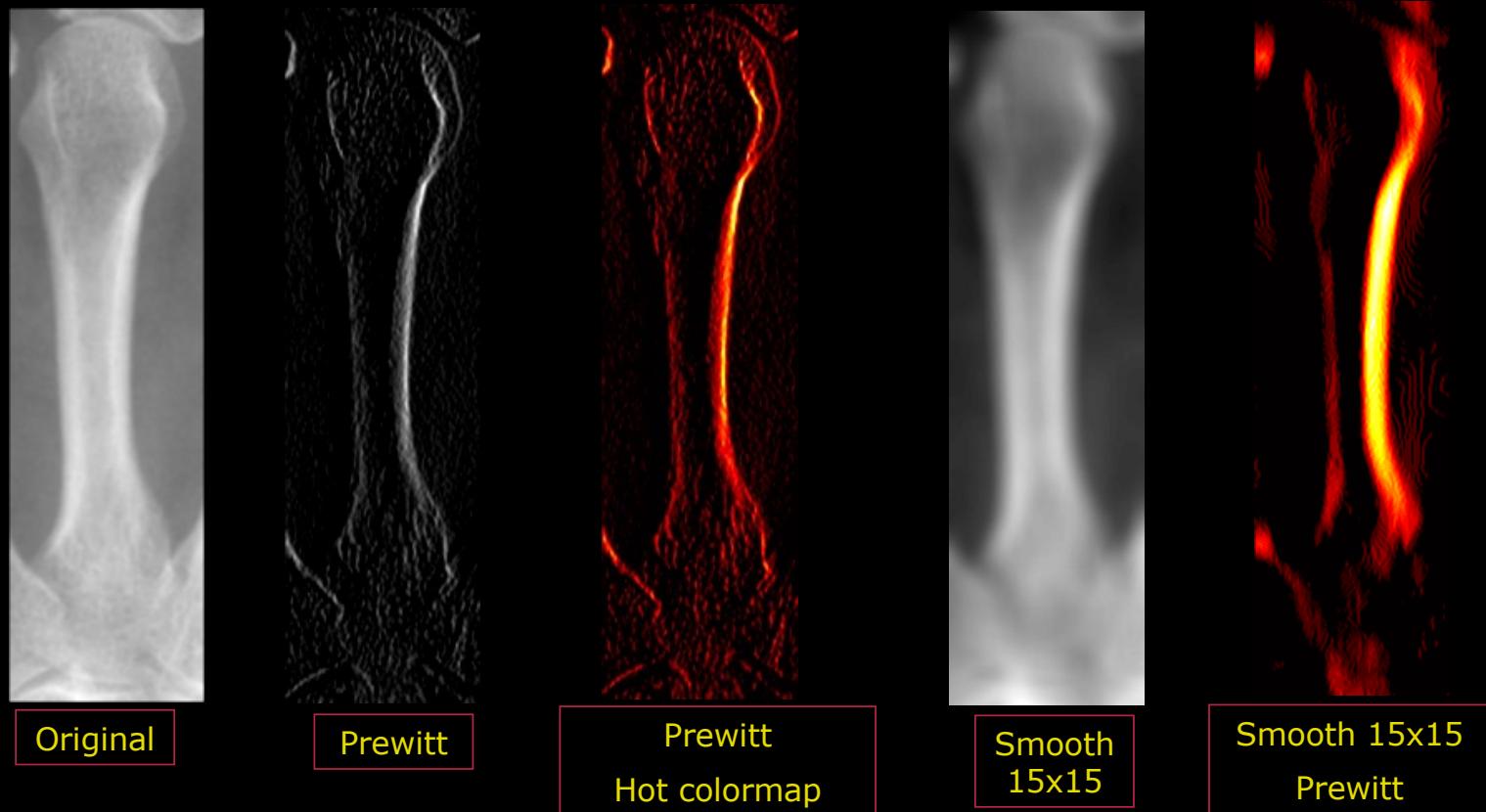
## Edge filter kernel

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

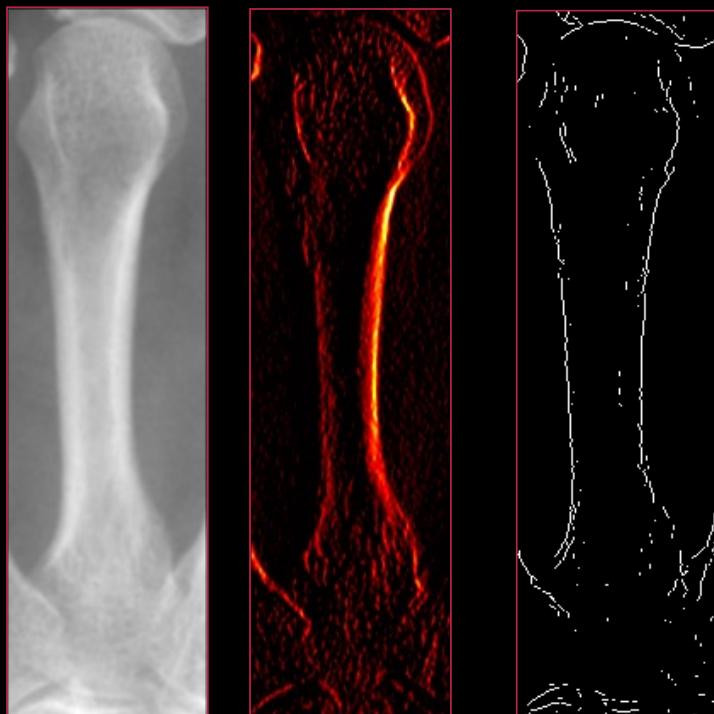
Vertical Prewitt filter  
(gradient:  $g(x)$ )

- The Prewitt filter is a typical edge filter
- Output image has high values where there are edges

## Prewitt filter (vertical)

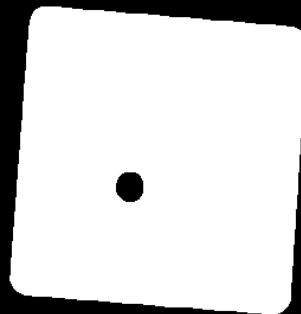
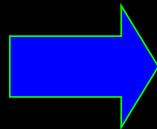
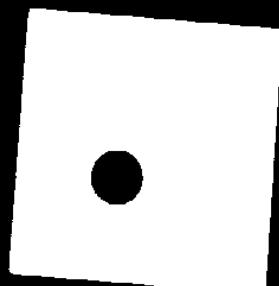


# Edge detection



- Edge filter
  - Prewitt for example
- Find magnitude of 2D gradient
- Thresholding
  - Separate edges from non-edges
- Output is binary image
  - Edges are white

## Lecture 4b – Morphology



0	0	1	1	1	0	0
0	1	1	1	1	1	0
1	1	1	1	1	1	1
1	1	1	1	1	1	1
1	1	1	1	1	1	1
0	1	1	1	1	1	0
0	0	1	1	1	0	0

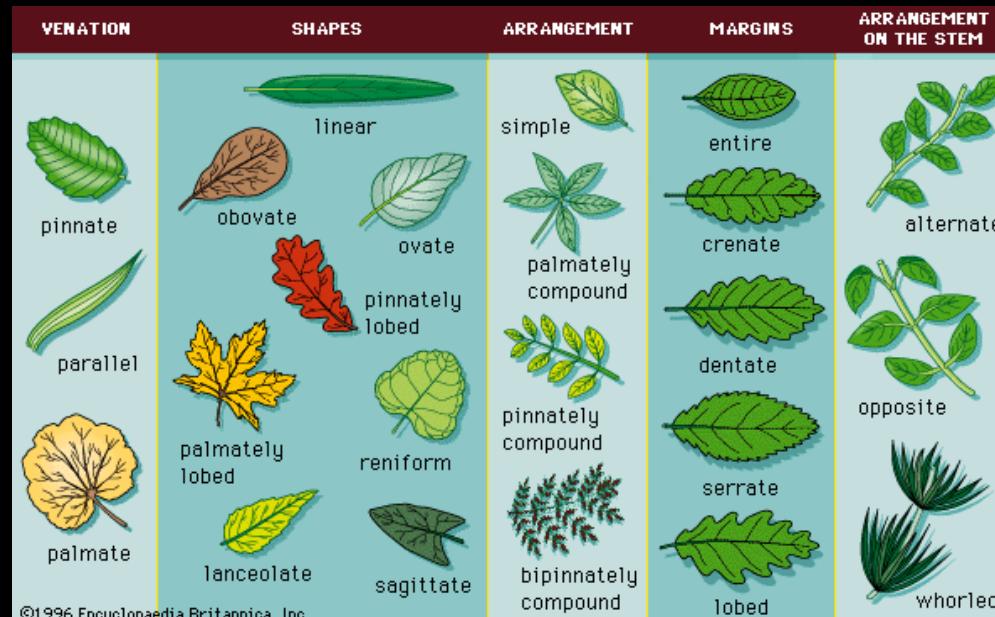


## What can you *also* do after today?

- Describe the similarity between filtering and morphology
- Describe a structuring element
- Compute the dilation of a binary image
- Compute the erosion of a binary image
- Compute the opening of a binary image
- Compute the closing of a binary image
- Apply compound morphological operations to binary images
- Describe typical examples where morphology is suitable
- Remove unwanted elements from binary images using morphology
- Choose appropriate structuring elements and morphological operations based on image content

# Morphology

- The science of *form, shape* and *structure*
- In biology: The form and structure of animals and plants



Common leaf morphologies



# Mathematical morphology

Theorem 4.10

$$\left\{ \begin{array}{l} \psi_m = \tilde{\varphi} \tilde{\gamma} = \tilde{\gamma} \tilde{\varphi} \tilde{\gamma} = \psi \tilde{\gamma} \\ \psi_M = \tilde{\gamma} \tilde{\varphi} = \tilde{\varphi} \tilde{\gamma} \tilde{\varphi} = \psi \tilde{\varphi} \\ \psi = \tilde{\gamma} \psi = \tilde{\varphi} \psi, \\ \tilde{\gamma} \leq \psi_m \leq \psi \leq \psi_M \leq \tilde{\varphi} \end{array} \right. ,$$

The same theorem may be restated in another way. If  $\mathcal{Jd}(\mathcal{B}) \neq \emptyset$  then let  $B_i$  be a family of elements of  $\mathcal{B}$ . We have  $\vee B_i \in \sim B$ , and thus  $\tilde{\gamma}(\vee B_i) = \vee B_i$ . From the first relation above, it follows for any  $\psi \in \mathcal{Jd}(\mathcal{B})$ , that

$$\psi(\vee B_i) = \psi \tilde{\gamma}(\vee B_i) = \tilde{\varphi} \tilde{\gamma}(\vee B_i).$$

But  $\tilde{\gamma}(\vee B_i) = \vee B_i$ , so that

$$\tilde{\varphi}(\vee B_i) = \psi(\vee B_i) \in \mathcal{B}.$$

In the same way, we also obtain

$$\tilde{\gamma} \tilde{\varphi}(\wedge B_i) = \tilde{\gamma}(\wedge B_i) = \psi(\wedge B_i) \in \mathcal{B}.$$

In other words,  $\mathcal{B}$  is a *complete lattice* with respect to the ordering on  $\mathcal{B}$  induced by  $\leq$ , i.e. any family  $B_i$  in  $\mathcal{B}$  has a smallest upper bound  $\tilde{\varphi}(\vee B_i)$  and a greatest lower bound  $\tilde{\gamma}(\wedge B_i) \in \mathcal{B}$ .

Conversely, let us assume that  $\mathcal{B}$  is a complete lattice. Thus, for any  $A \in \mathcal{L}$ , the family  $\{B : B \in \mathcal{B}, B \geq A\}$  has in  $\mathcal{B}$  a greatest lower bound, which is

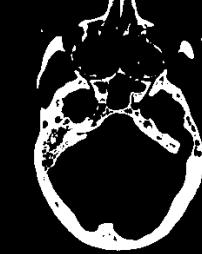
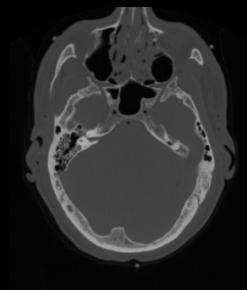
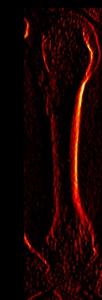
$$\tilde{\gamma}(\wedge \{B : B \in \mathcal{B}, B \geq A\}) = \tilde{\gamma} \tilde{\varphi}(A) \in \mathcal{B}.$$

But this implies  $\mathcal{B}_{\psi_M} \subseteq \mathcal{B}$  for the filter  $\psi_M = \tilde{\gamma} \tilde{\varphi}$ . Conversely, for any

- Developed in 1964
- Theoretical work done in Paris
- Used for classification of minerals in cut stone
- Initially used for binary images

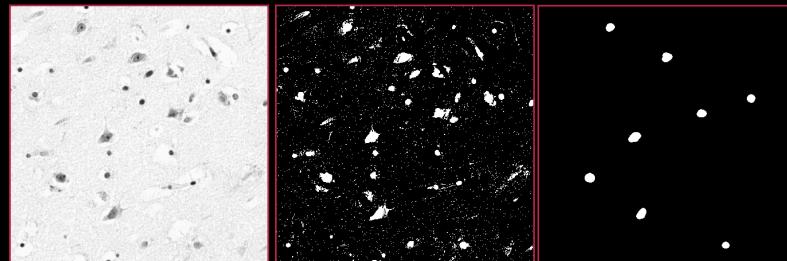
Do not worry! We use a much less theoretical approach!

# Relevance?

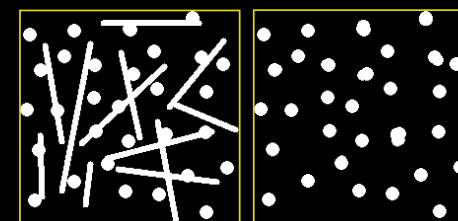
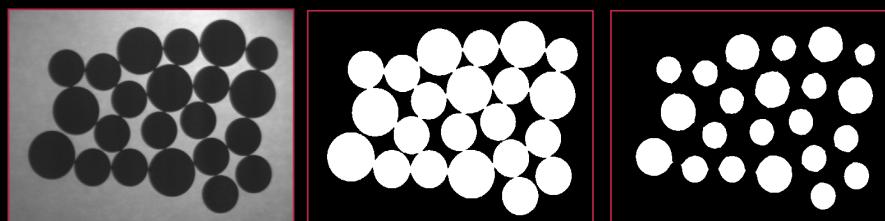


- Point wise operations
- Filtering
- Thresholding
  - Gives us objects that are separated by the background
- Morphology
  - Manipulate and enhance binary objects

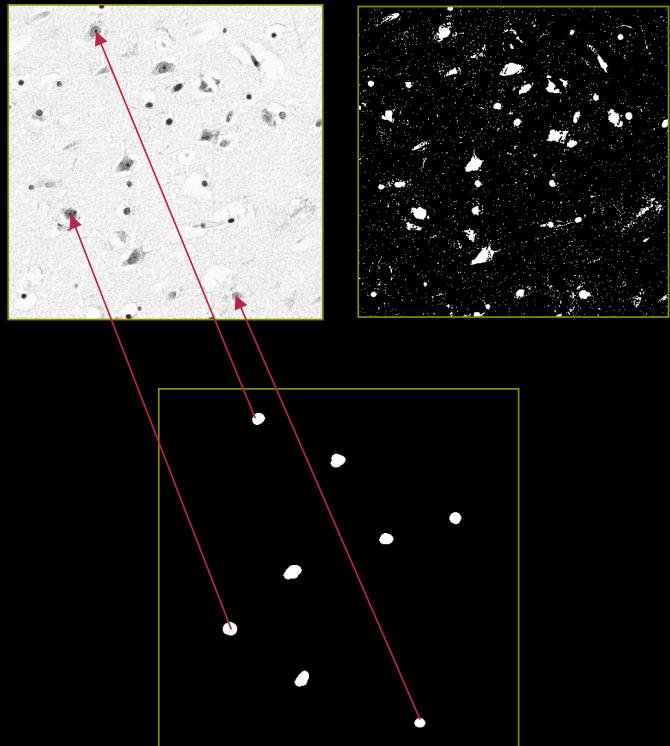
## What can it be used for?



- Remove noise
  - Small objects
  - Fill holes
- Isolate objects
- Customized to specific shapes



# How does it work?



- Grayscale image
- Preprocessing
  - Inversion
- Threshold => Binary image
- Morphology

# Filtering and morphology

## ■ Filtering

- Gray level images
- Kernel
- Moves it over the input image
- Creates a new output image

1	2	0	1	3	1
2	1	4	2	2	2
1	0	1	0	1	3
1	2	1	0	2	4
2	5	3	1	2	2
2	1	3	1	6	3

# Filtering and morphology

0	1	0
1	1	1
0	1	0

Disk

1	1	1
1	1	1
1	1	1

Box

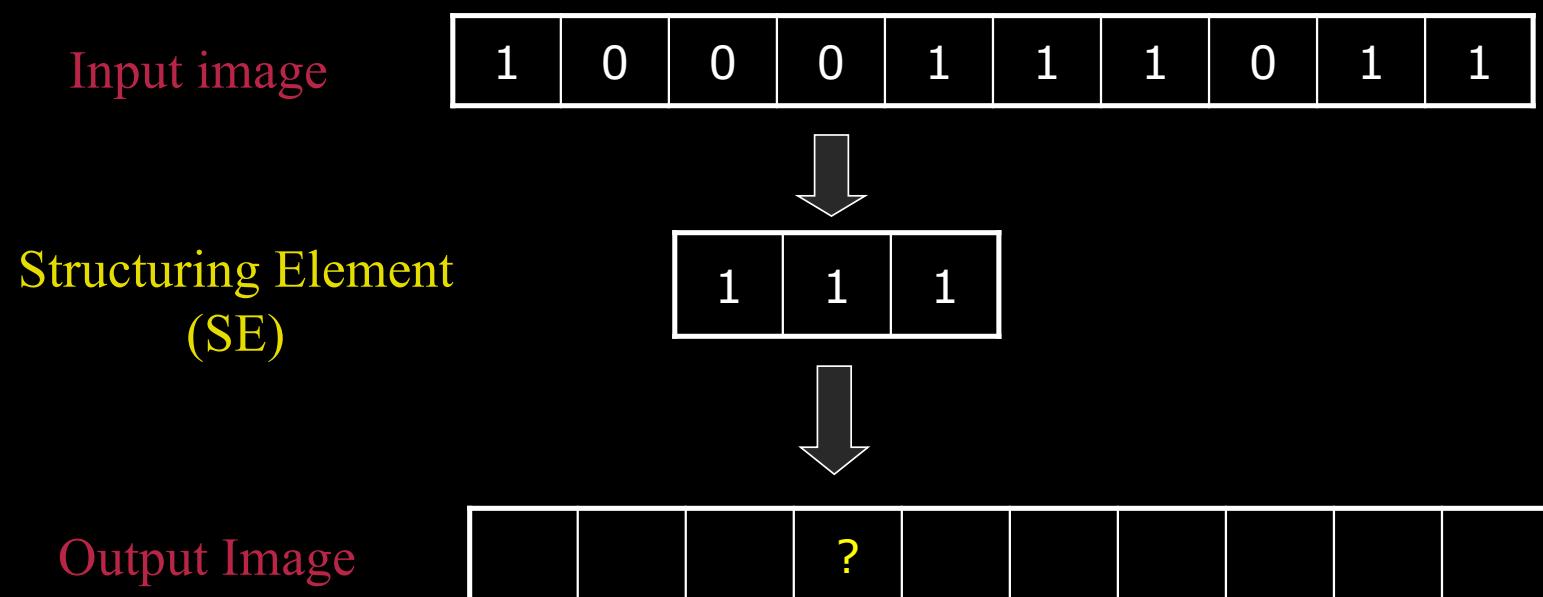
## ■ Filtering

- Gray level images
- Kernel
- Moves it over the input image
- Creates a new output image

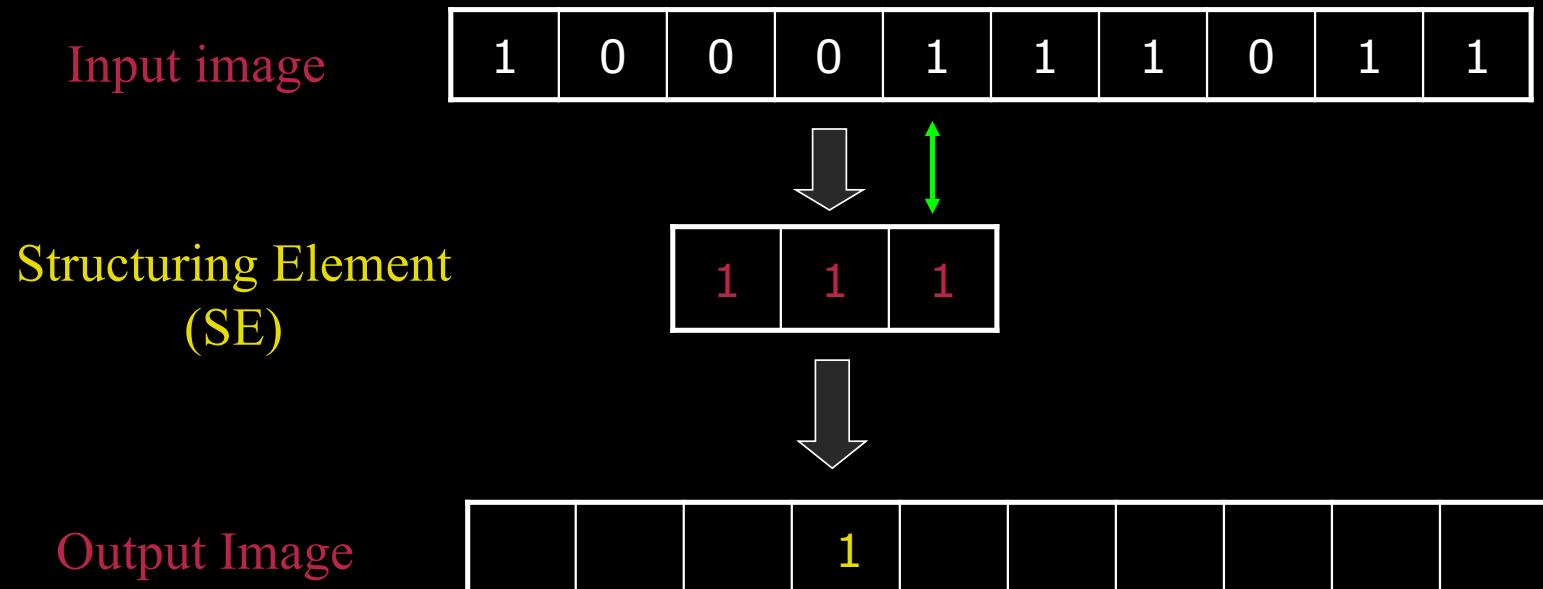
## ■ Morphology

- Binary images
- Structuring element (SE)
- Moves the SE over the input image
- Creates a new binary output image

# 1D Morphology

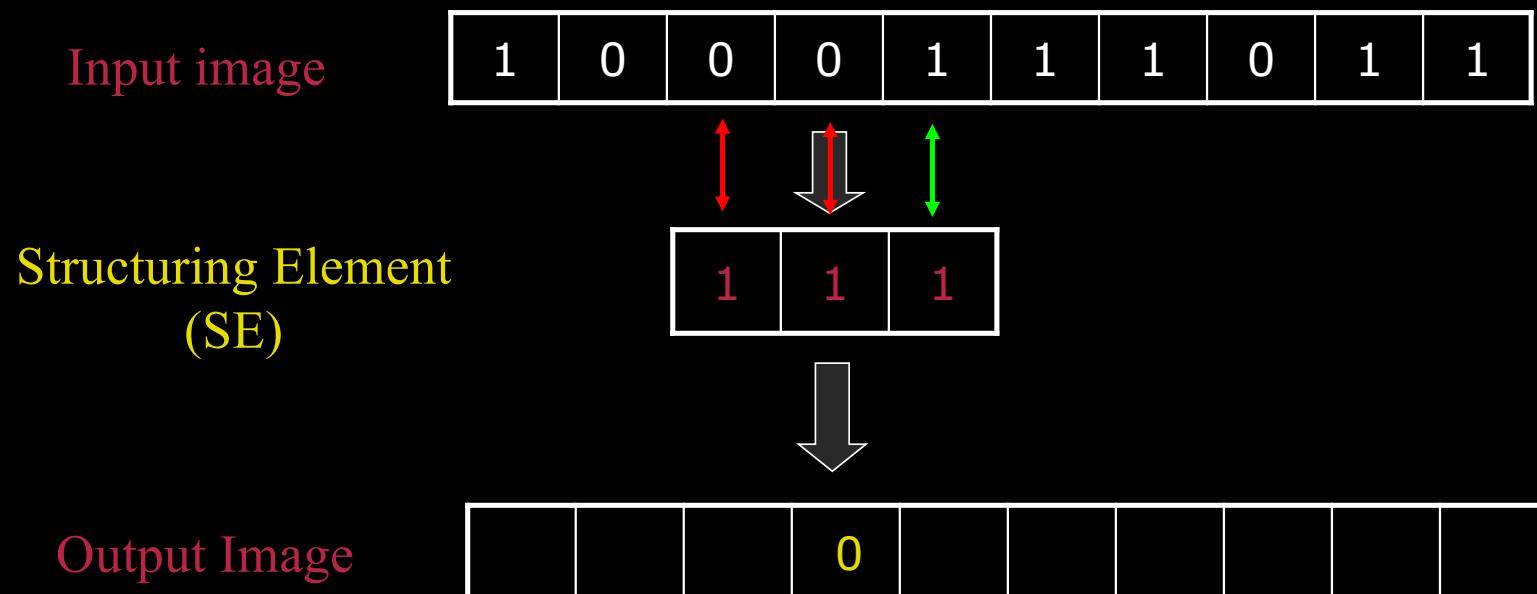


## 1D Morphology : The hit operation



- If just one 1 in the SE match with the input
  - output 1
- else
  - output 0

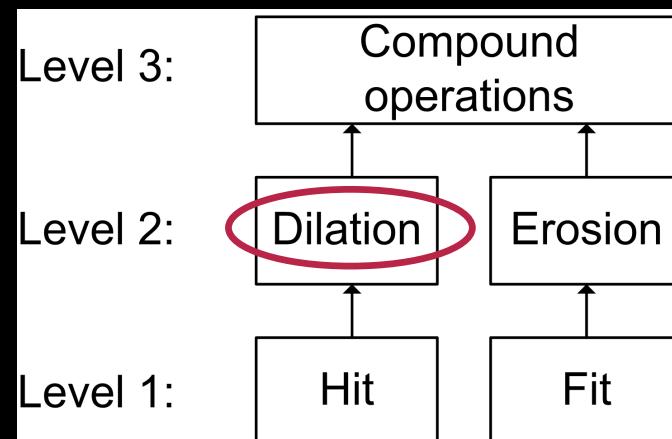
# 1D Morphology : The fit operation



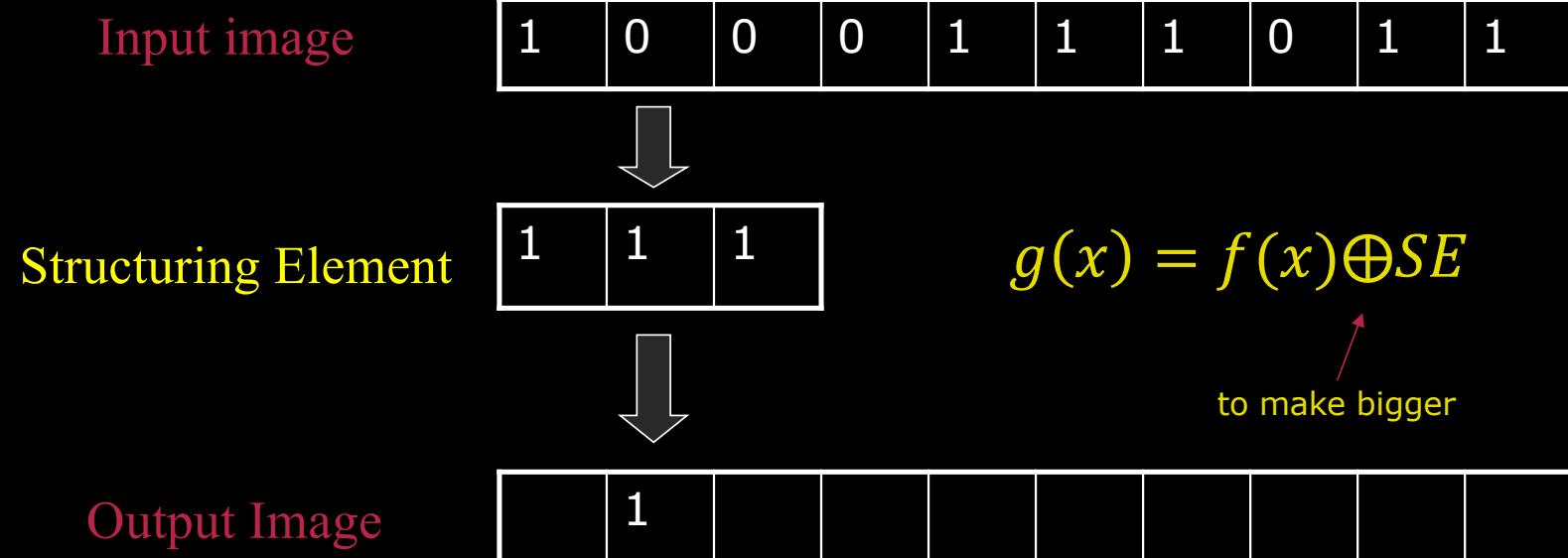
- If all 1 in the SE match with the input
  - output 1
- else
  - output 0

# 1D Morphology : Dilation

- Dilate : To make wider or larger
  - Dansk : udvide
- Based on the *hit* operation



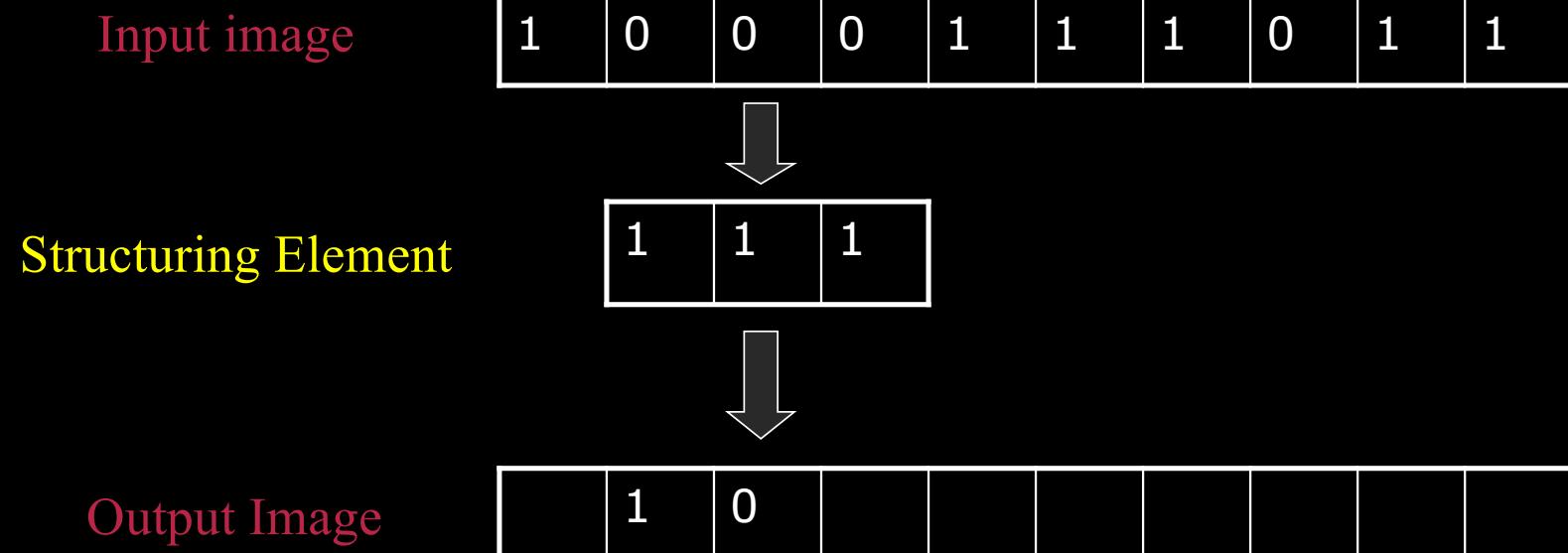
# 1D Dilation example



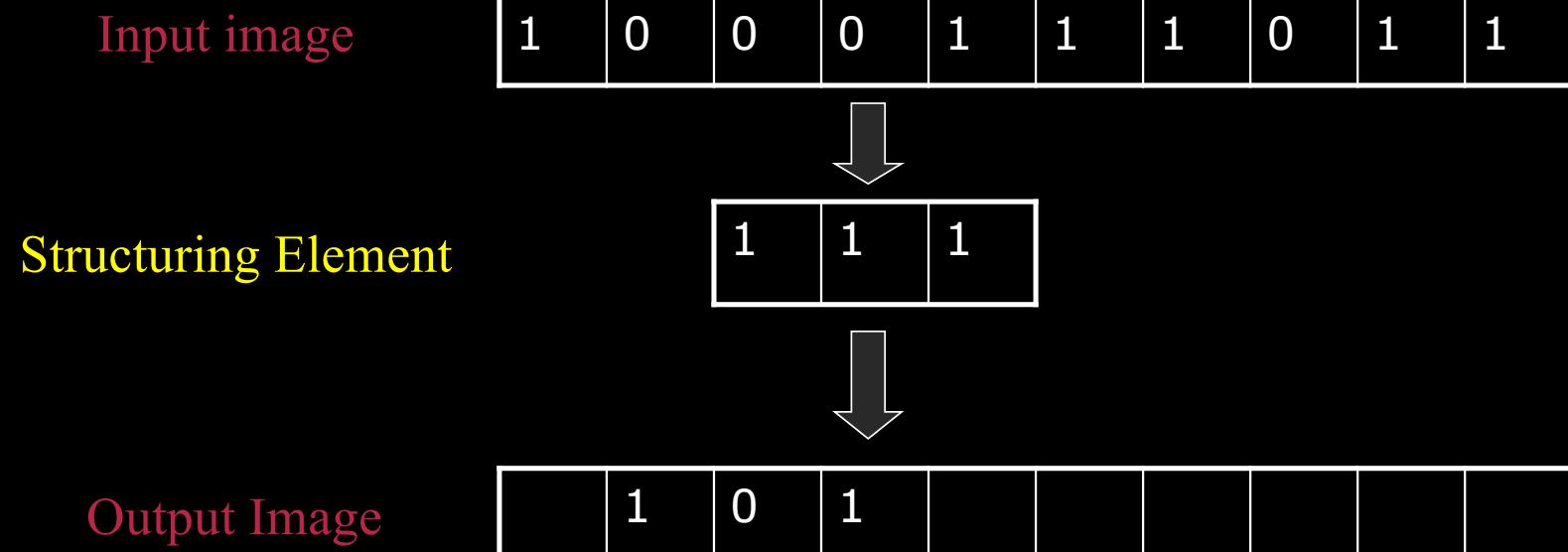
Hit

- If just one 1 in the SE match with the input
  - output 1
- else
  - output 0

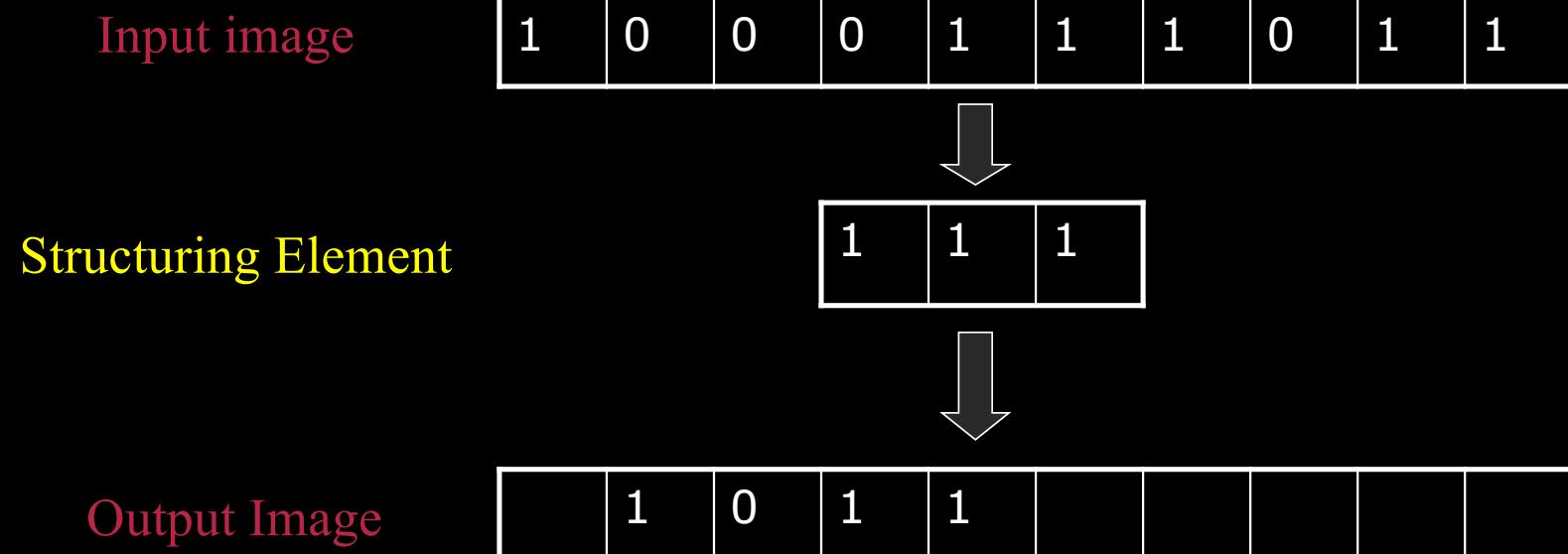
## Example for Dilation



## Example for Dilation



## Example for Dilation



## Example for Dilation

Input image

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



1	1	1
---	---	---



Structuring Element

Output Image

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

## Example for Dilation

Input image

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



Structuring Element

1	1	1
---	---	---



Output Image

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

## Example for Dilation

Input image

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



1	1	1
---	---	---

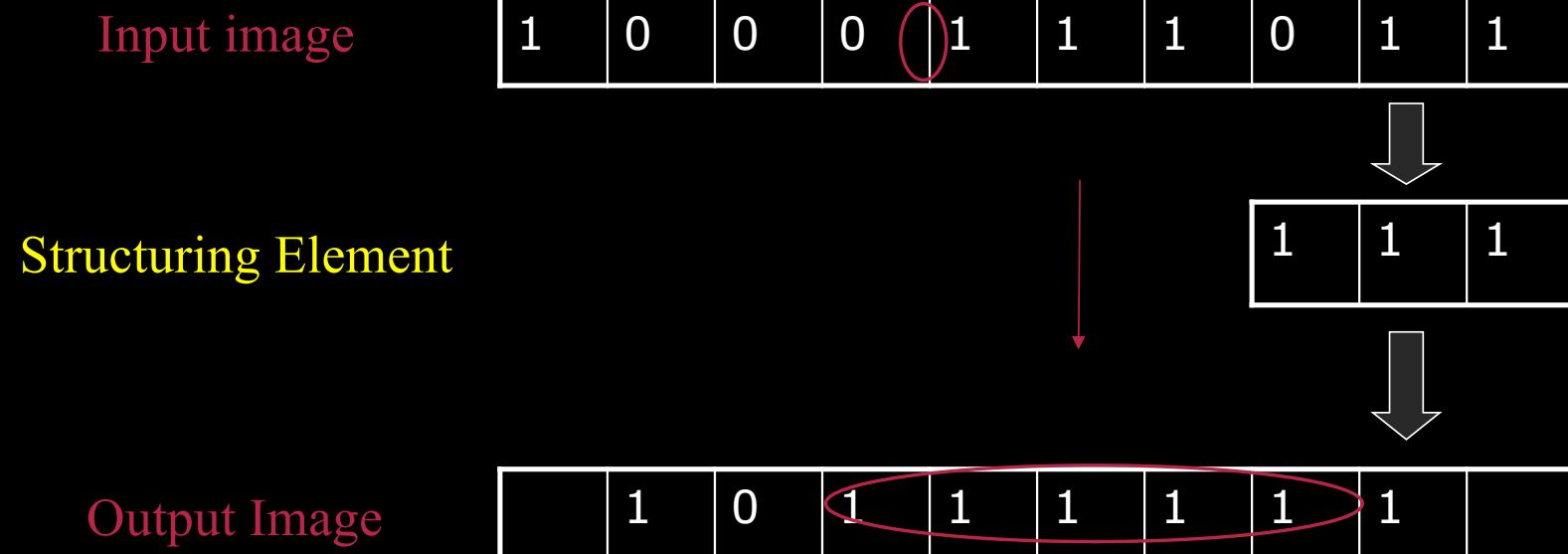


Structuring Element

Output Image

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

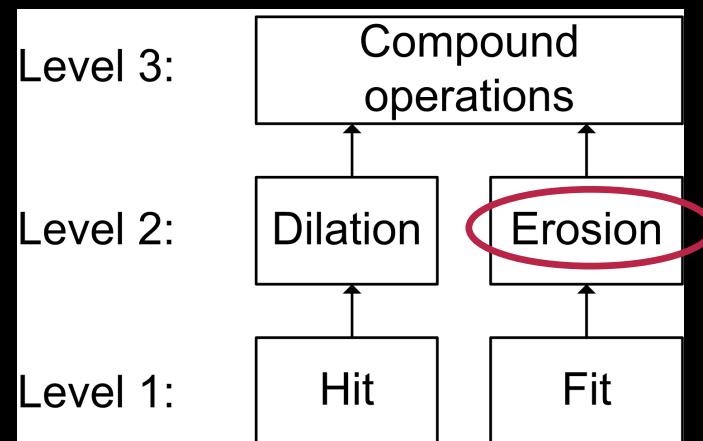
## Example for Dilation



The object gets bigger, and holes are filled!

## 1D Morphology : Erosion

- Erode : To wear down (*Waves eroded the shore*)
  - Dansk : tære, gnave
- Based on the *fit* operation



## Example for Erosion

Input image

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



1	1	1
---	---	---

$$g(x) = f(x) \ominus SE$$

to make smaller

Structuring Element

Output Image

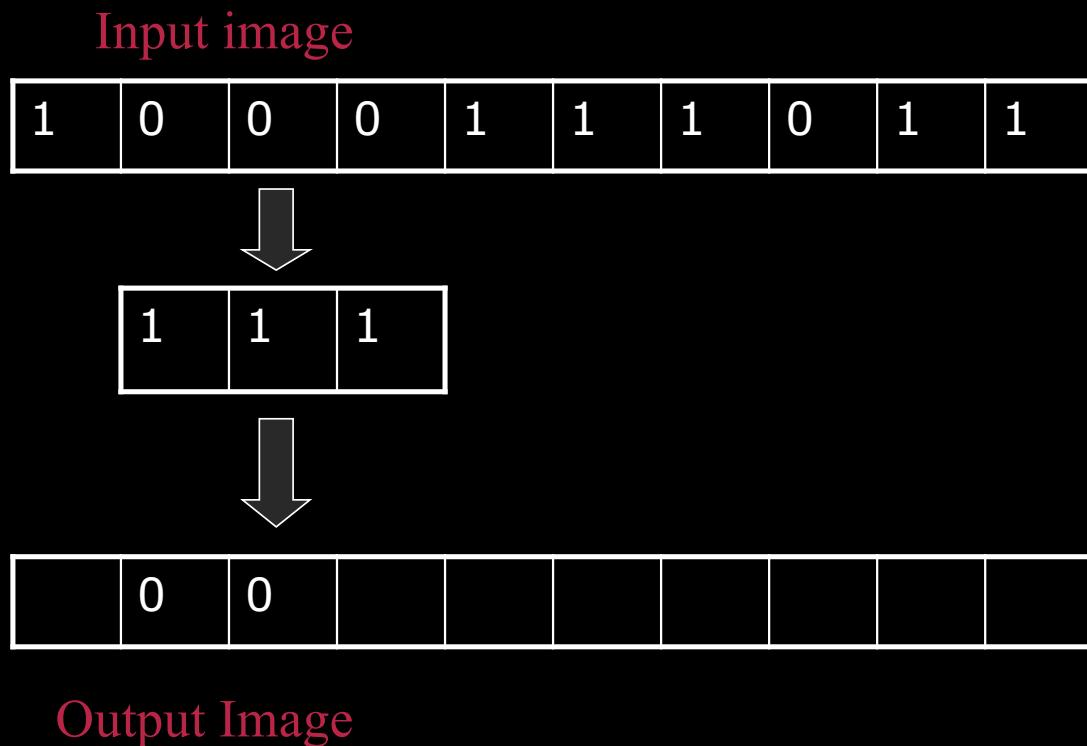
	0								
--	---	--	--	--	--	--	--	--	--

Fit

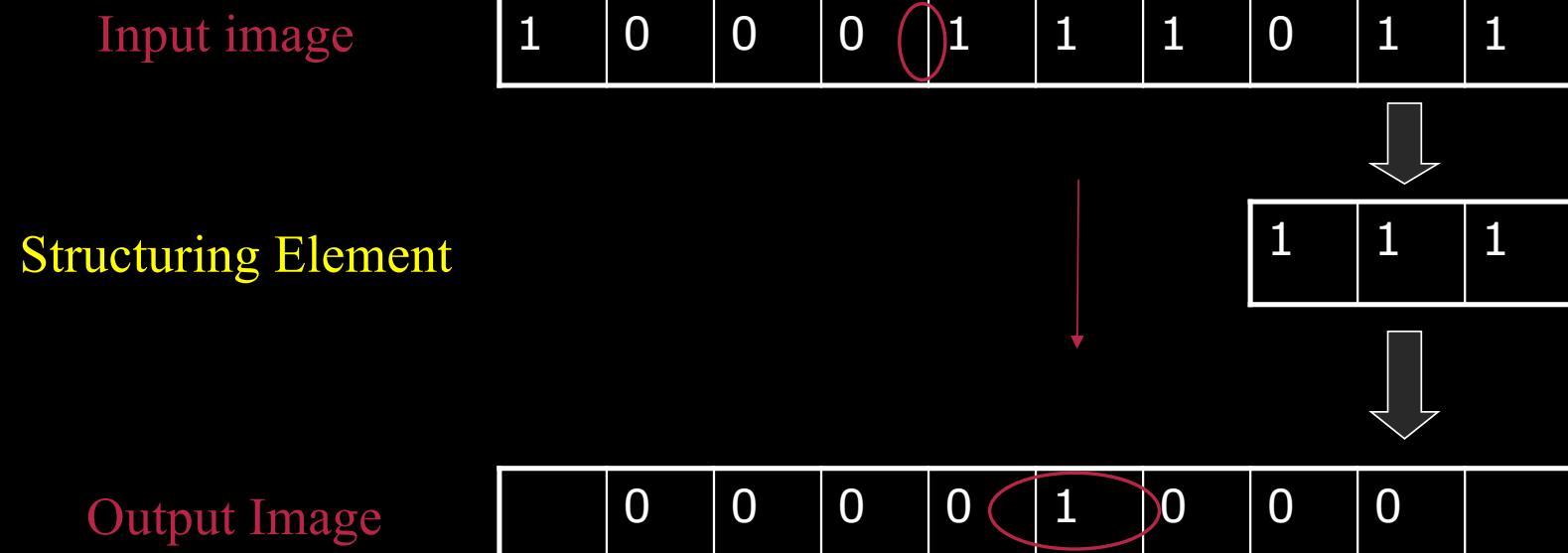
- If all 1 in the SE match with the input
  - output 1
- else
  - output 0

## Quiz 10: 1D Erosion

- A) 0 1 0 0 1 1 0 0
- B) 0 0 1 0 1 0 0 0
- C) 0 0 0 0 1 0 0 0
- D) 0 0 1 0 0 0 0 1
- E) 0 1 0 0 0 1 0 0



## Example for Erosion



The object gets smaller

# Structuring Element (Kernel)

0	1	0
1	1	1
0	1	0

Disk

1	1	1
1	1	1
1	1	1

Box

- Structuring Elements can have varying sizes
- Usually, element values are 0 or 1, but other values are possible (including none!)
- Structural Elements have an origin
- Empty spots in the Structuring Elements are *don't cares*!

		1	1	1		
1	1	1	1	1	1	
1	1	1	1	1	1	1
1	1	1	1	1	1	1
1	1	1	1	1	1	1
		1	1	1		

## Structuring Element Origin

0	1	0
1	1	1
0	1	0

- The origin is not always the center of the SE

1	1	1
1	1	1
1	1	1

# Special structuring elements

- Structuring elements can be customized to a specific problem

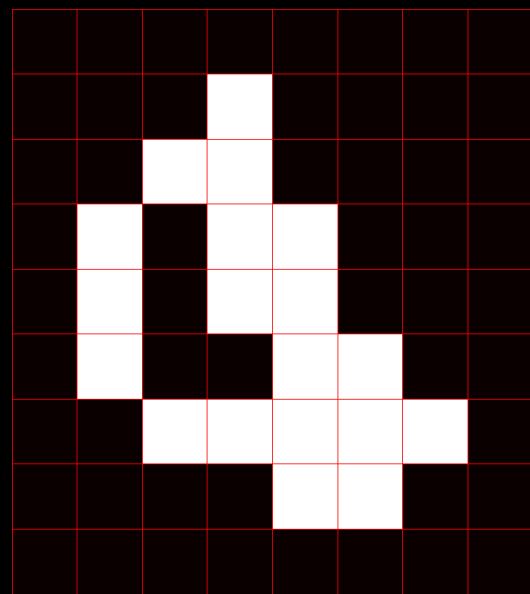
0	0	0	1	0	0	0
0	0	1	1	1	0	0
0	1	1	1	1	1	0
1	1	1	(1)	1	1	1
0	1	1	1	1	1	0
0	0	1	1	1	0	0
0	0	0	1	0	0	0

Diamond

0	0	0	0	0	1	1
0	0	1	(1)	1	0	0
1	1	0	0	0	0	0

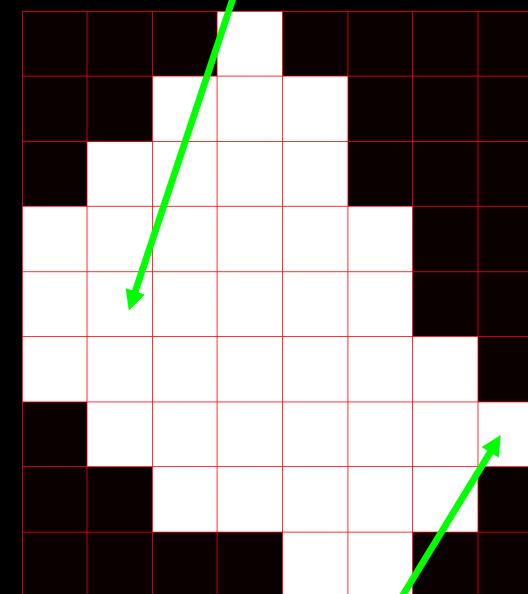
Line

## Dilation on images - disk



0	1	0
1	1	1
0	1	0

SE



Holes are closed

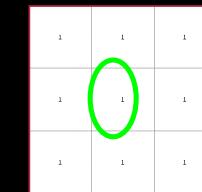
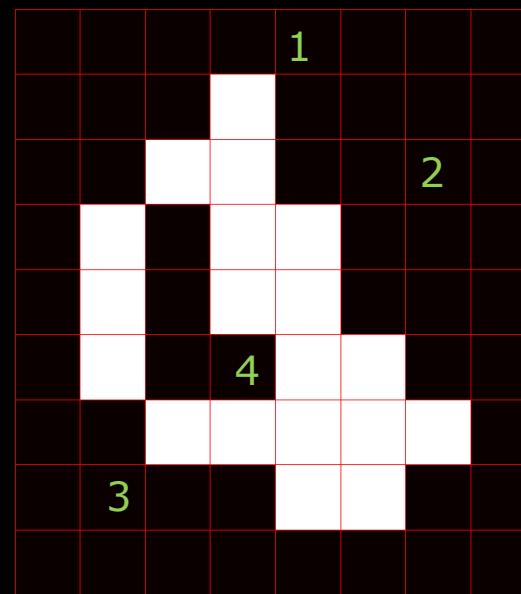
Object is bigger

$$g(x, y) = f(x, y) \oplus SE$$

## Quiz 11: Dilation on image - box

1 2 3 4

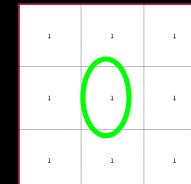
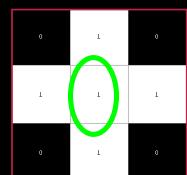
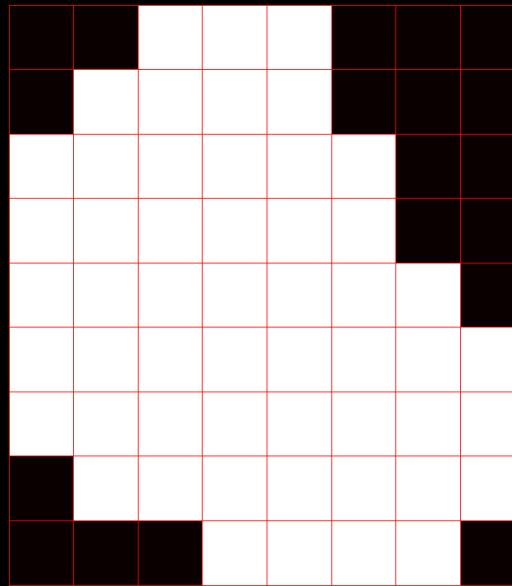
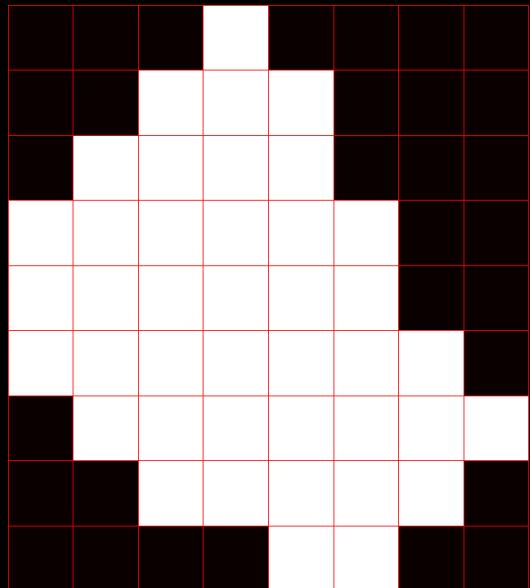
- A) 1 0 1 1
- B) 0 1 0 0
- C) 0 1 1 1
- D) 0 1 0 1
- E) 1 1 0 1



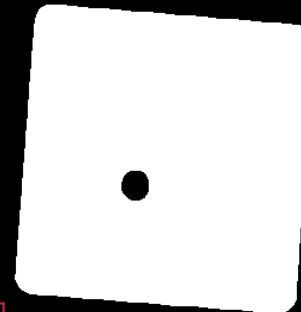
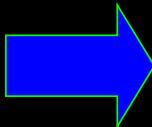
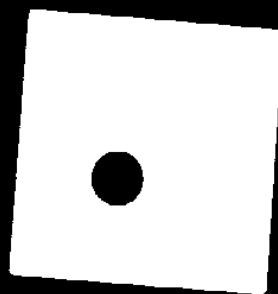
SE

$$g(x, y) = f(x, y) \oplus SE$$

## Dilation – the effect of the SE



## Dilation Example



- Round structuring element (disk)
- Creates round corners

0	0	1	1	1	0	0
0	1	1	1	1	1	0
1	1	1	1	1	1	1
1	1	1	1	1	1	1
1	1	1	1	1	1	1
0	1	1	1	1	1	0
0	0	1	1	1	0	0

## Quiz 12: Threshold and dilation

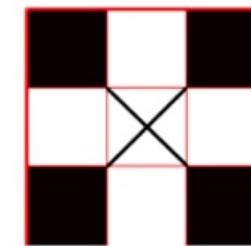
- A) 14
- B) 17
- C) 6
- D) 3
- E) 12

Answer:

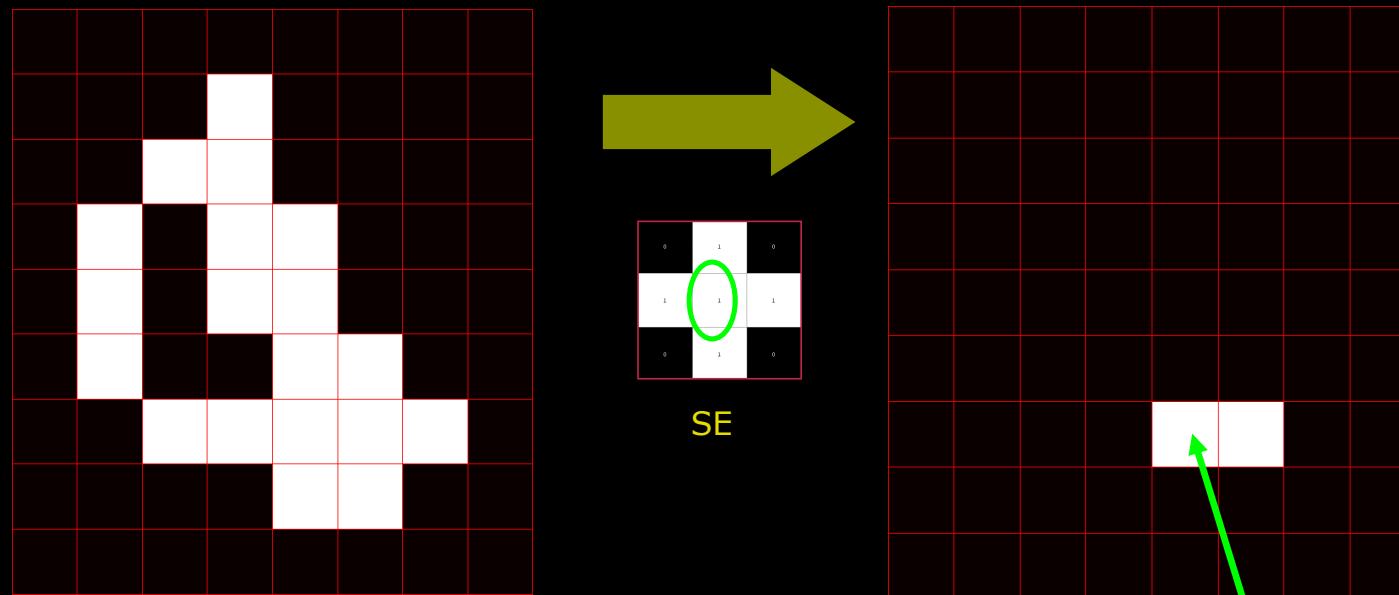
145	56	86	42	191
19	33	41	255	115
14	240	203	234	21
135	120	209	167	58
199	3	135	176	116

A threshold of 200 is applied to the image and the result is a binary image. Now a dilation is performed with the structuring element below. How many foreground pixels are there in the resulting image?

145	56	86	42	191
19	33	41	255	115
14	240	203	234	21
135	120	209	167	58
199	3	135	176	116



## Erosion on images - disk



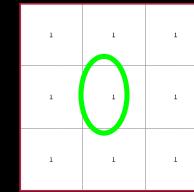
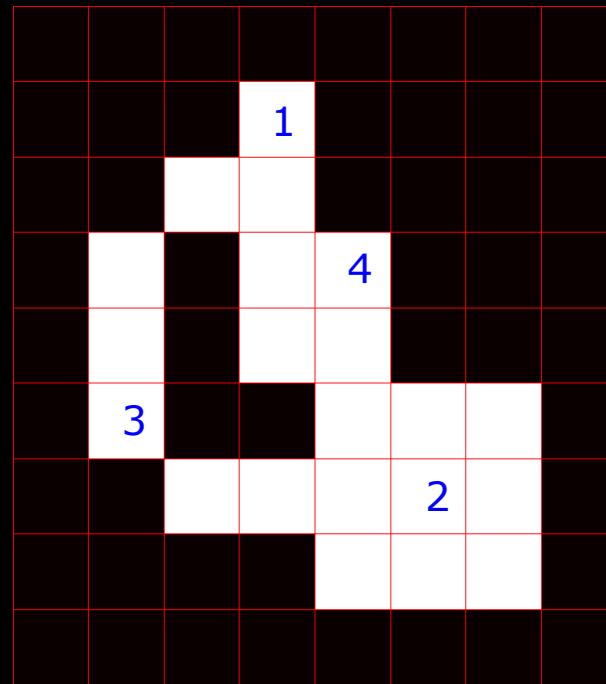
$$g(x, y) = f(x, y) \ominus SE$$

Object is smaller

## Quiz 13: Erosion on images - box

1 2 3 4

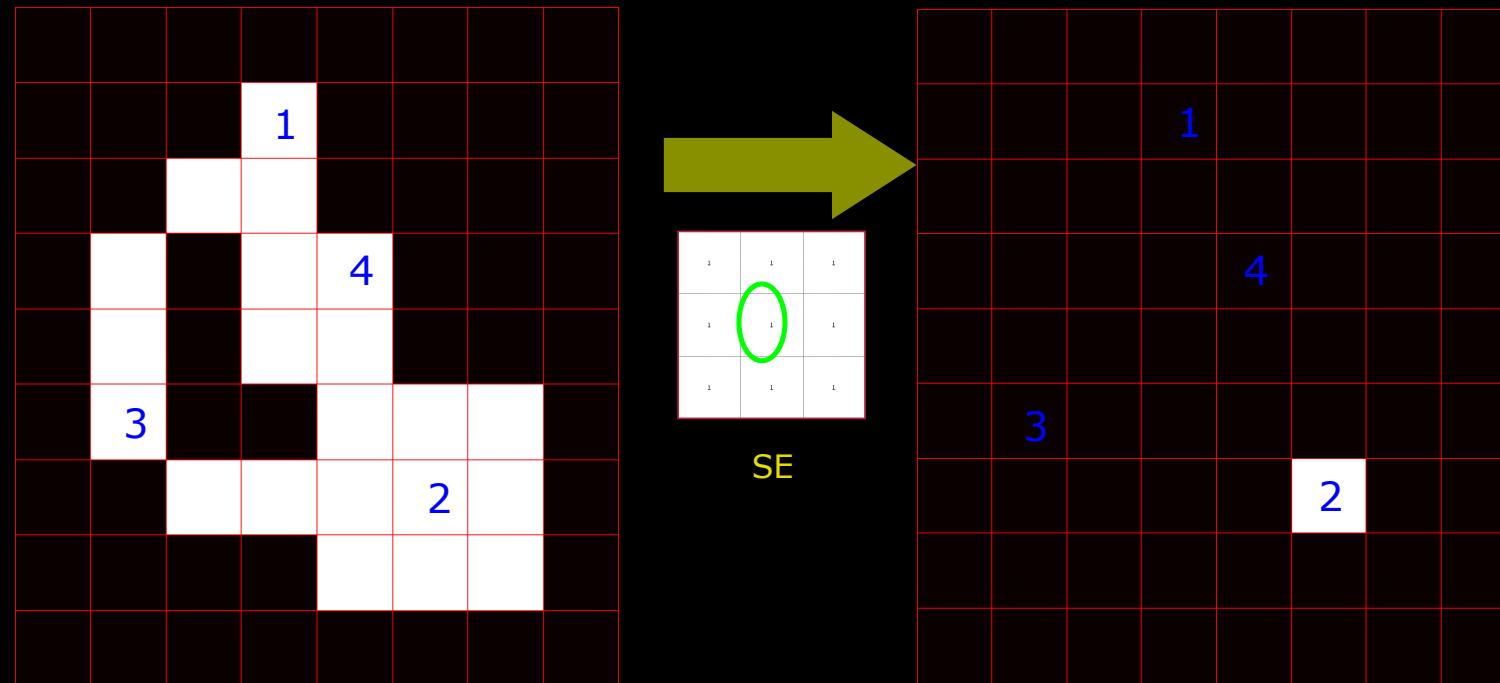
- A) 0 0 1 0
- B) 1 0 1 0
- C) 0 1 1 0
- D) 0 1 0 0
- E) 1 0 0 0



SE

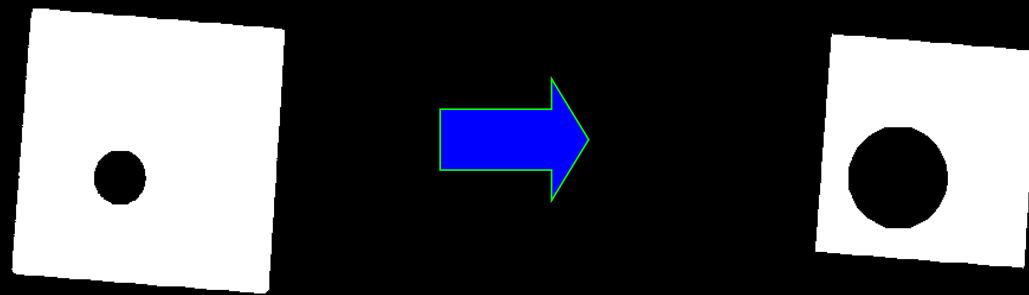
$$g(x, y) = f(x, y) \ominus SE$$

## Erosion on images – box (square)



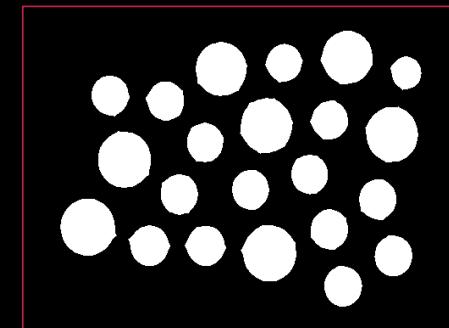
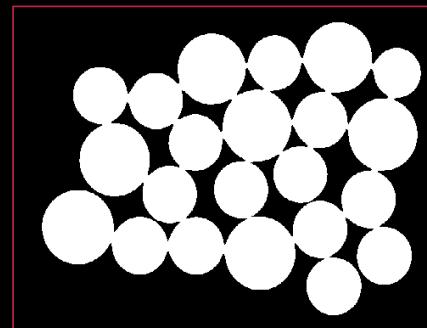
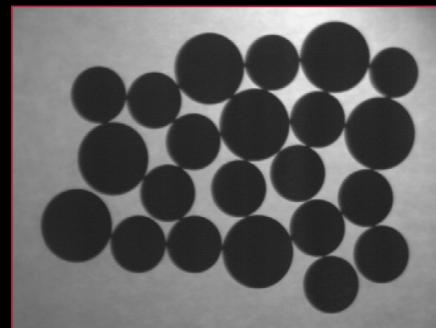
$$g(x, y) = f(x, y) \ominus SE$$

## Erosion example

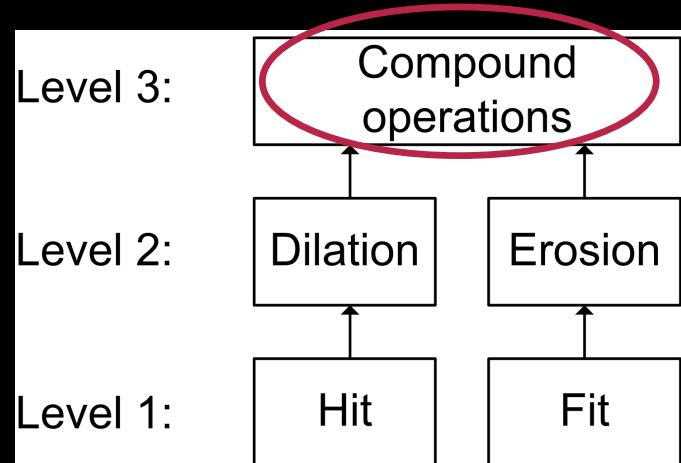


## Counting Coins

- Counting these coins is difficult because they touch each other!
- **Solution:** Threshold and Erosion separates them!
- More on counting next time!



# Compound operations

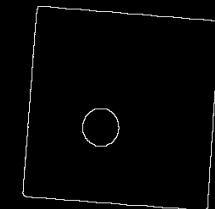
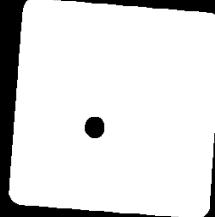
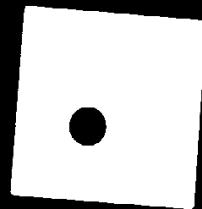


- Compound
  - *made of two or more separate parts or elements*
- Combining Erosion and Dilation into more advanced operations
  - Finding the outline
  - Opening
    - Isolate objects and remove small objects (better than Erosion)
  - Closing
    - Fill holes (better than Dilation)

## Finding the outline

1. Dilate input image (object gets bigger)
2. Subtract input image from dilated image
3. The outline remains!

$$g(x, y) = (f(x, y) \oplus SE) - f(x, y)$$





# Opening

- Motivation: Remove small objects BUT keep original size (and shape)
- Opening = Erosion + Dilation
  - Use the same structuring element!
  - Similar to erosion but less destructive
- Math:

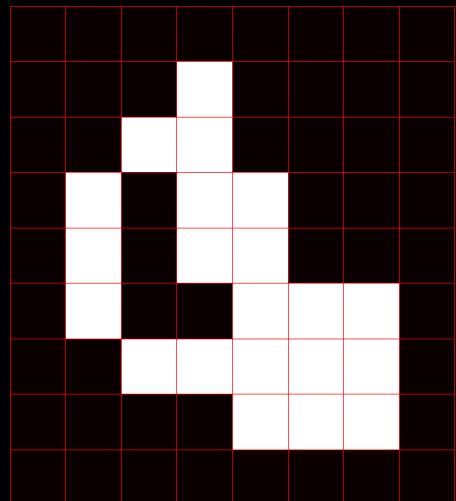
$$g(x, y) = f(x, y) \circ SE = (f(x, y) \ominus SE) \oplus SE$$

- Opening is **idempotent**: Repeated operations have no further effects!

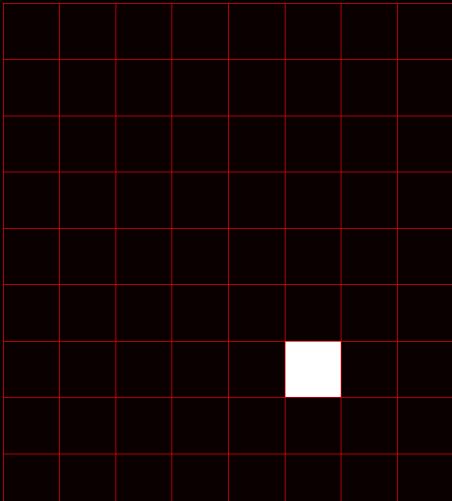
$$f(x, y) \circ SE = (f(x, y) \circ SE) \circ SE$$

## Opening

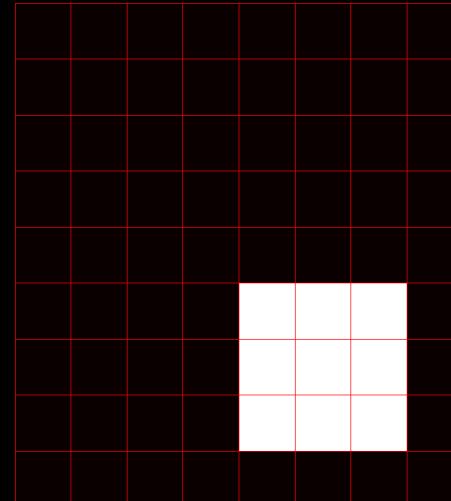
$$g(x, y) = (f(x, y) \ominus SE) \oplus SE$$



Original



Eroded



Dilated

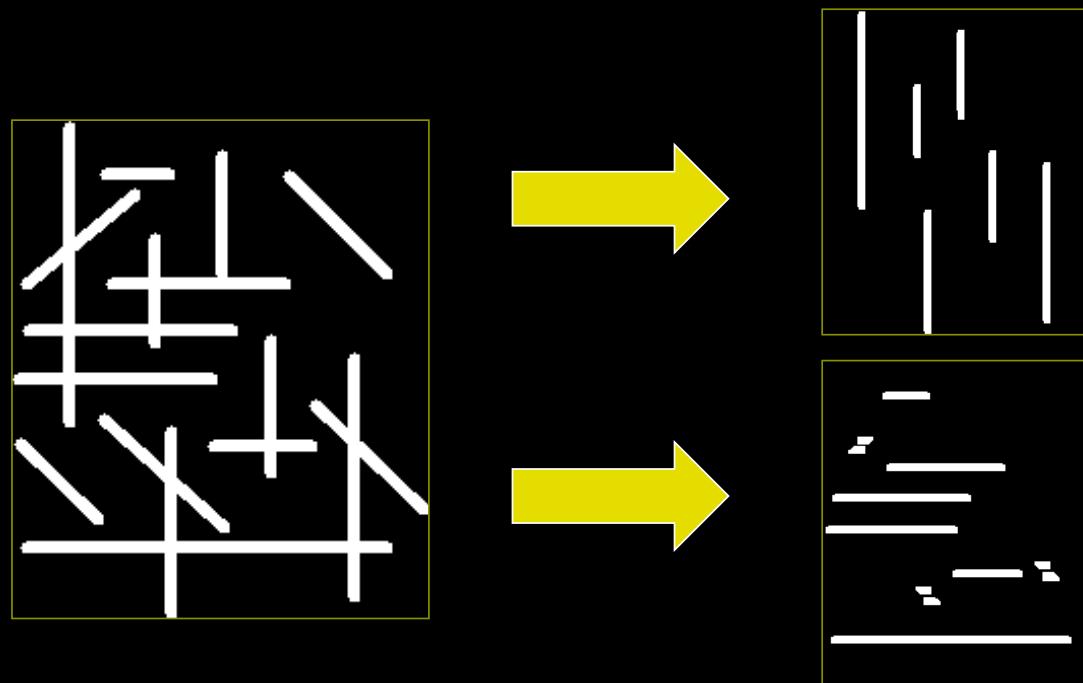
Opening = erosion+dilation



SE

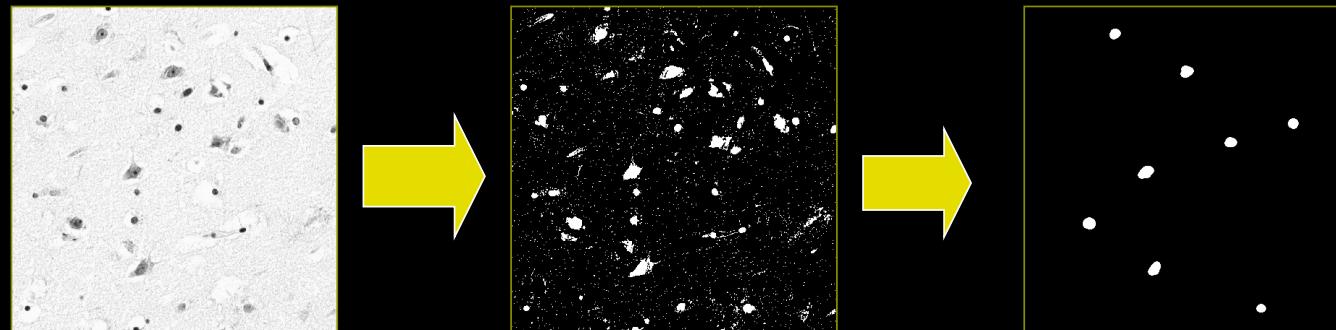
# Opening Example

- 9x3 and 3x9 Structuring Elements



## Opening example

- Size of structuring element should fit into the smallest object to keep
- Structuring Element: 11 pixels disc

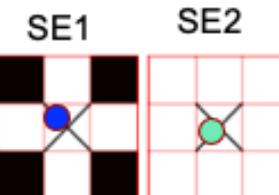
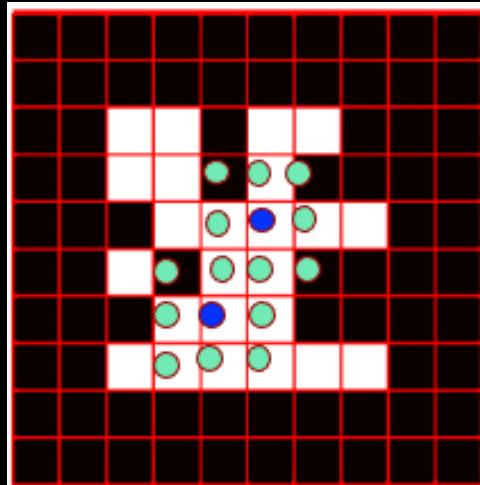




## Quiz 14: Compound operations on image

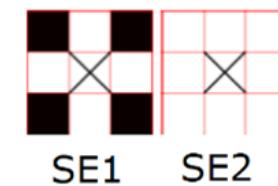
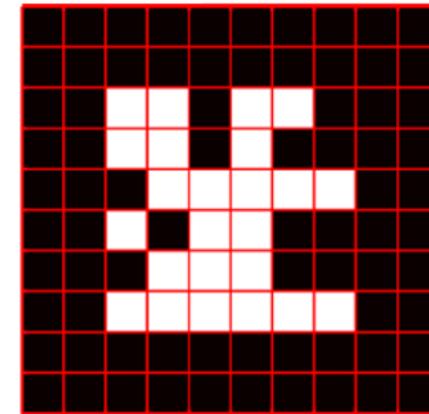
- A) 3
- B) 23
- C) 11
- D) 36
- E) 16

Answer:



The compound morphological operation seen below is applied to the image. How many foreground pixels are there in the resulting image?

$$(I \ominus SE1) \oplus SE2,$$





# Closing

- Motivation: Fill holes BUT keep original size (and shape)
- Closing = Dilation + Erosion
  - Use the same structuring element!
  - Similar to Dilation but less destructive
- Math:

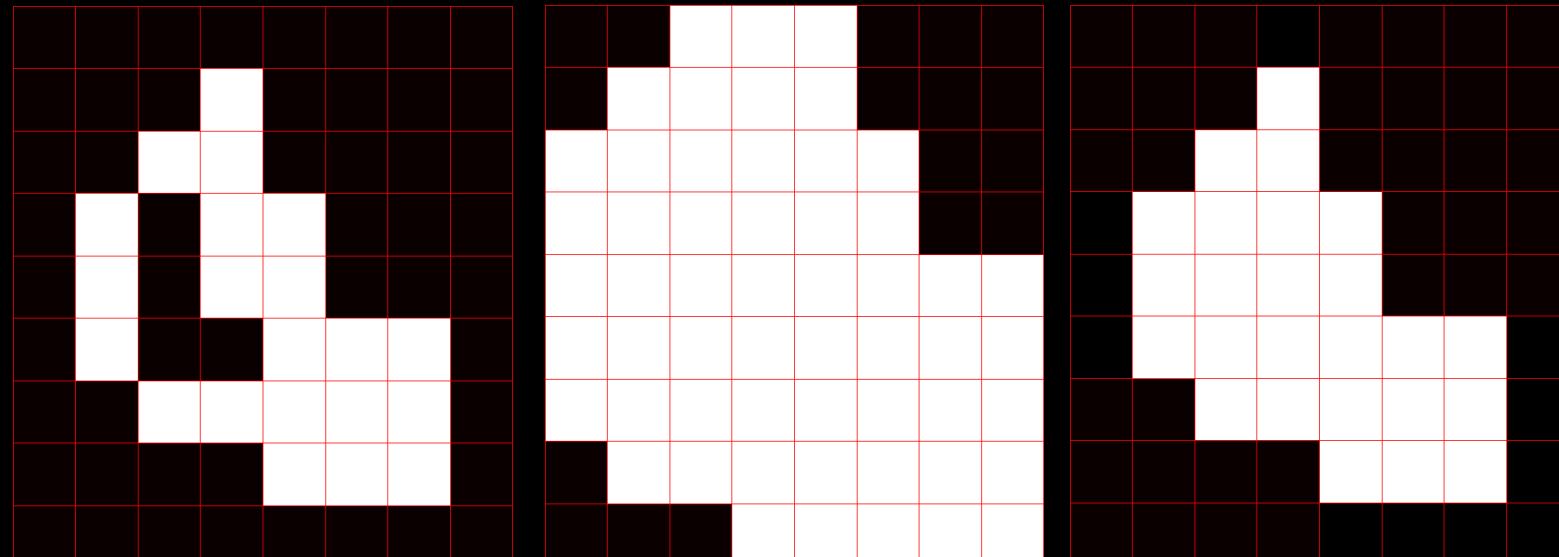
$$g(x, y) = f(x, y) \bullet SE = (f(x, y) \oplus SE) \ominus SE$$

- Closing is **idempotent**: Repeated operations have no further effects!

$$f(x, y) \circ SE = (f(x, y) \circ SE) \circ SE$$

# Closing

$$g(x, y) = (f(x, y) \oplus SE) \ominus SE$$

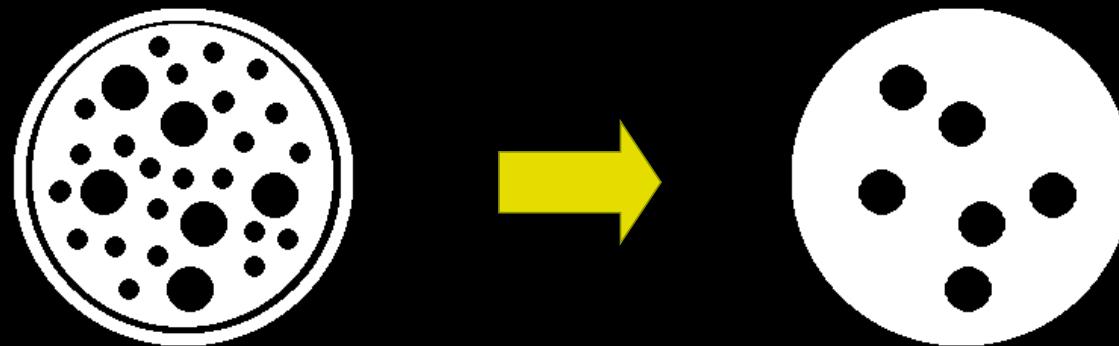


Closing = dilation + erosion



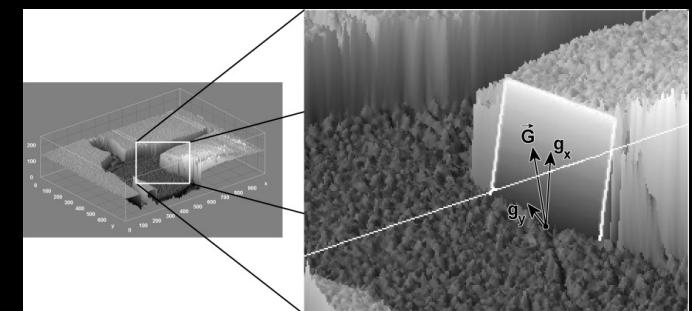
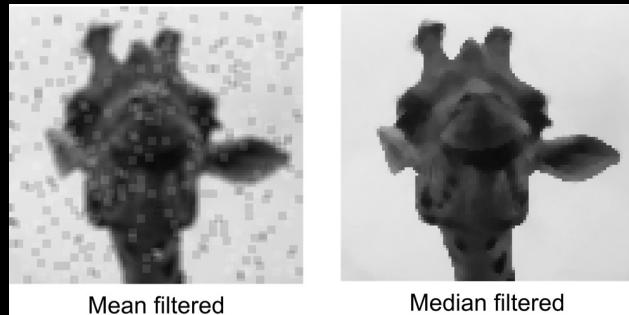
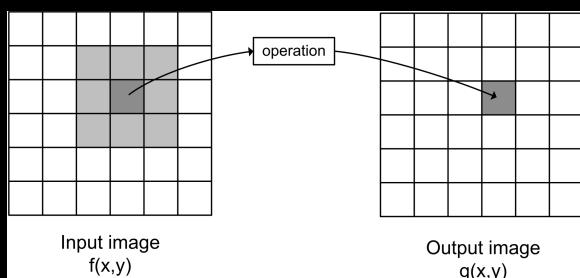
## Closing Example

- Closing operation with a 22 pixels disc
- Closes small holes

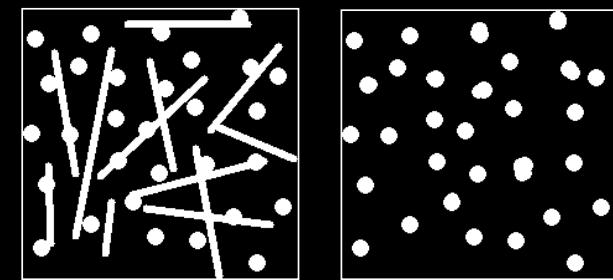
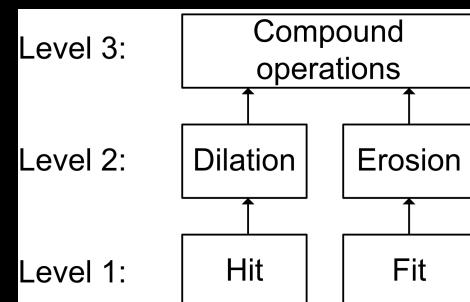
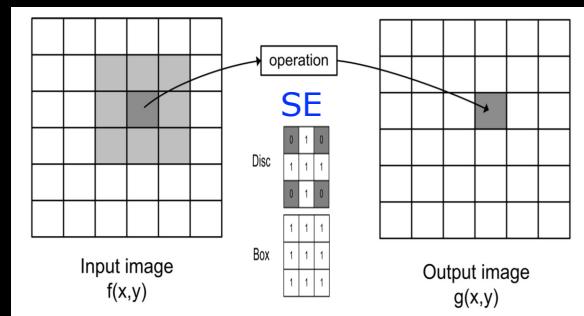


# What did we learn today

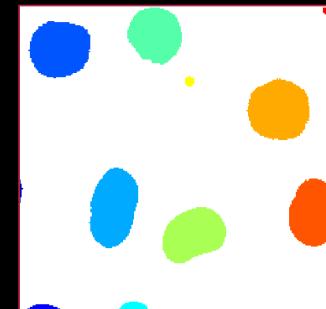
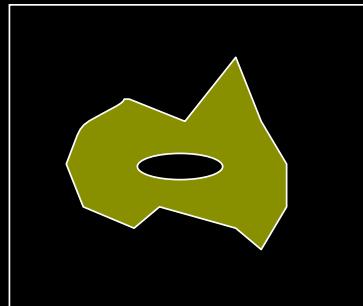
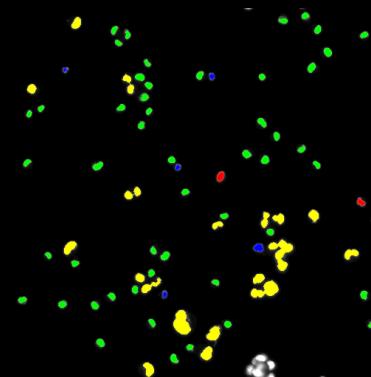
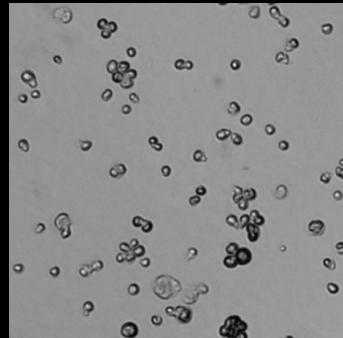
## Neighbourhood Processing



## Morphology of binary images



# Next week: Blob Analysis and object classification





# Image Analysis

Rasmus R. Paulsen

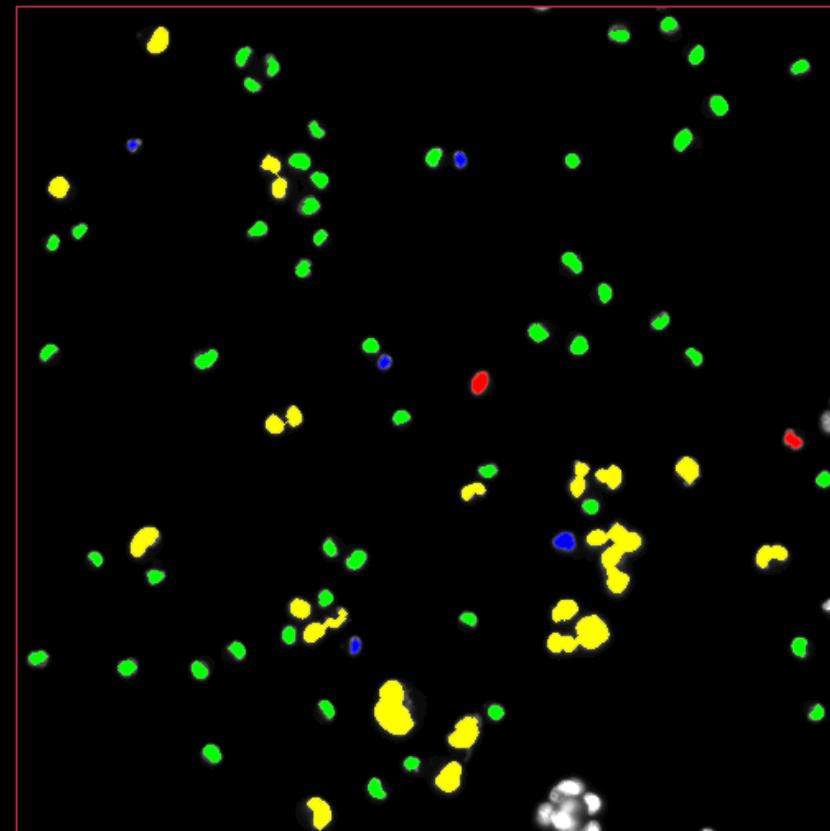
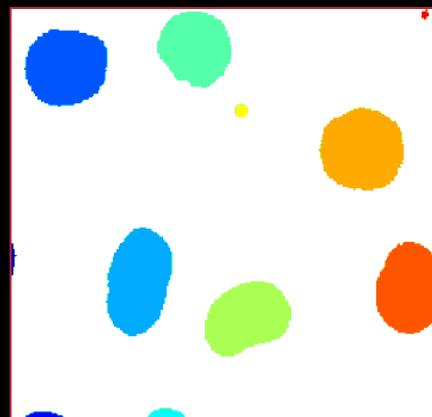
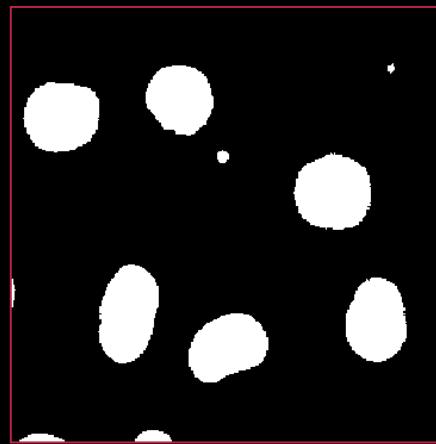
Tim B. Dyrby

DTU Compute

[rapa@dtu.dk](mailto:rapa@dtu.dk)

<http://www.compute.dtu.dk/courses/02502>

# Lecture 5 – BLOB analysis and feature based classification

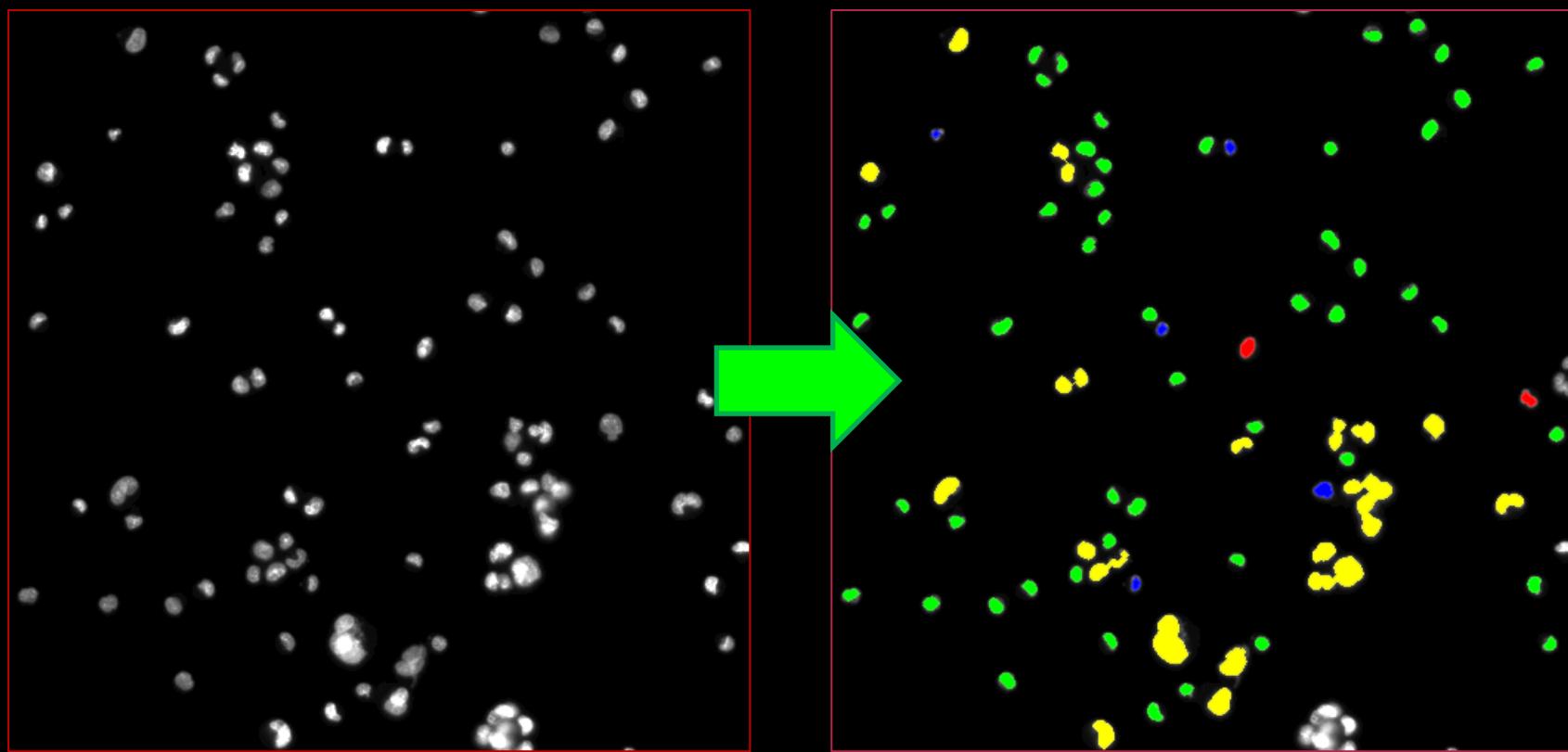


# What can you do after today?

- Calculate the connected components of a binary image. Both using 4-connected and 8-connected neighbours
- Compute BLOB features including area, bounding box ratio, perimeter, center of mass, circularity, and compactness
- Describe a feature space
- Compute blob feature distances in feature space
- Classify binary objects based on their blob features
- Estimate feature value ranges using annotated training data
- Compute a confusion matrix
- Compute rates from a confusion matrix including sensitivity, specificity and accuracy
- Determine and discuss what is the importance of sensitivity and specificity given an image analysis problem

# Object recognition

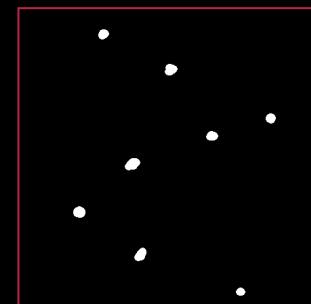
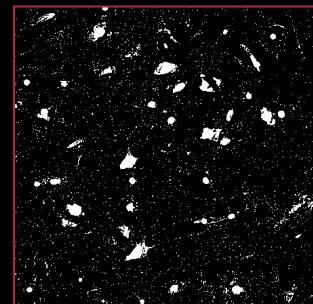
- Recognise objects in images
- Put them into different classes



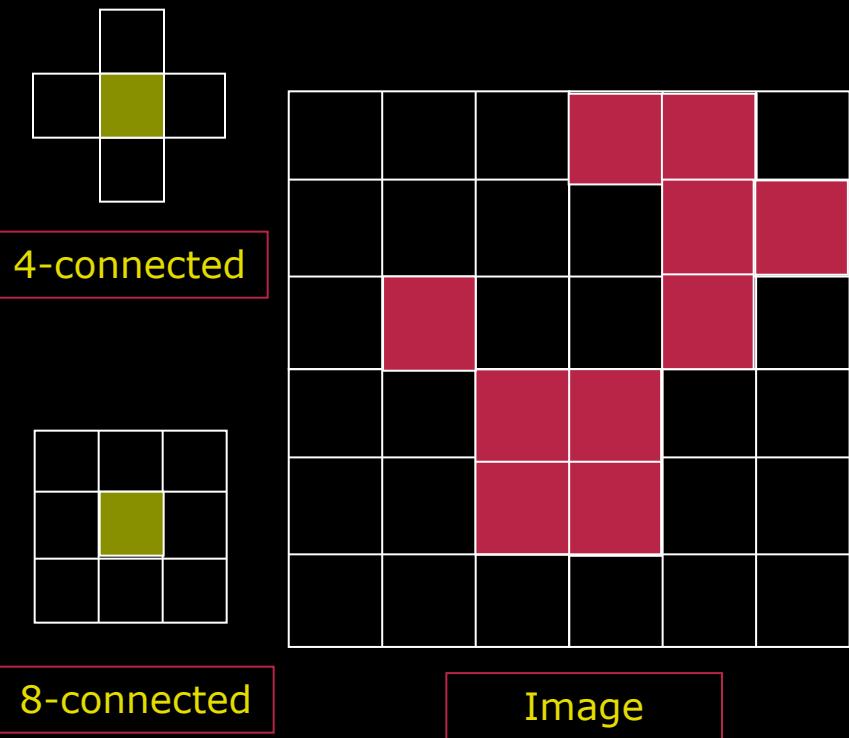
# BLOB – what is it?



- BLOB = Binary Large Object
  - Group of connected pixels
- BLOB Analysis
  - *Connected component analysis*
  - *Object labelling*



# Isolating a BLOB



## ■ What we want:

- For each object in the image, a list with its pixels

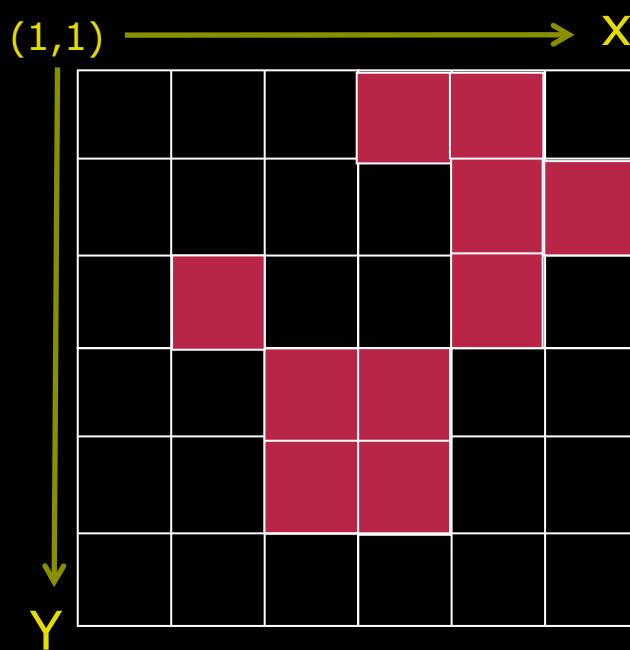
## ■ How do we get that?

- Connected component analysis

## ■ Connectivity

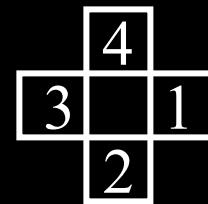
- Who are my neighbors?
- 4-connected
- 8-connected

# Connected component analysis



- Binary image
- Seed point: where do we start?
- *Grassfire* concept
  - Delete (burn) the pixels we visit
  - Visit all *connected* (4 or 8) neighbors

4-connected

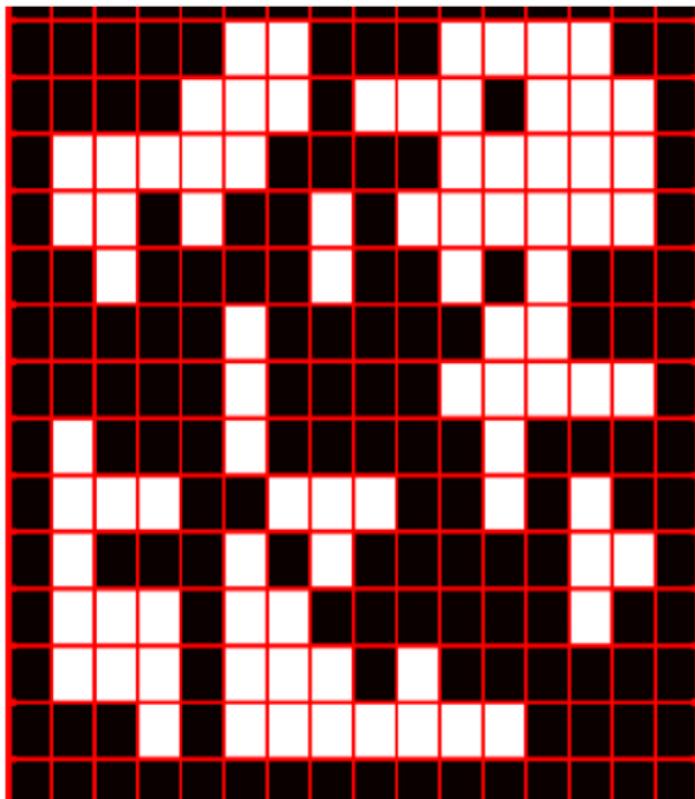




Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)

## BLOBs with 4- and 8- connectivity

A BLOB analysis is performed using both 4- and 8- connectivity. How many BLOBS are found using the two different connectivities?



3 and 7

9 and 5

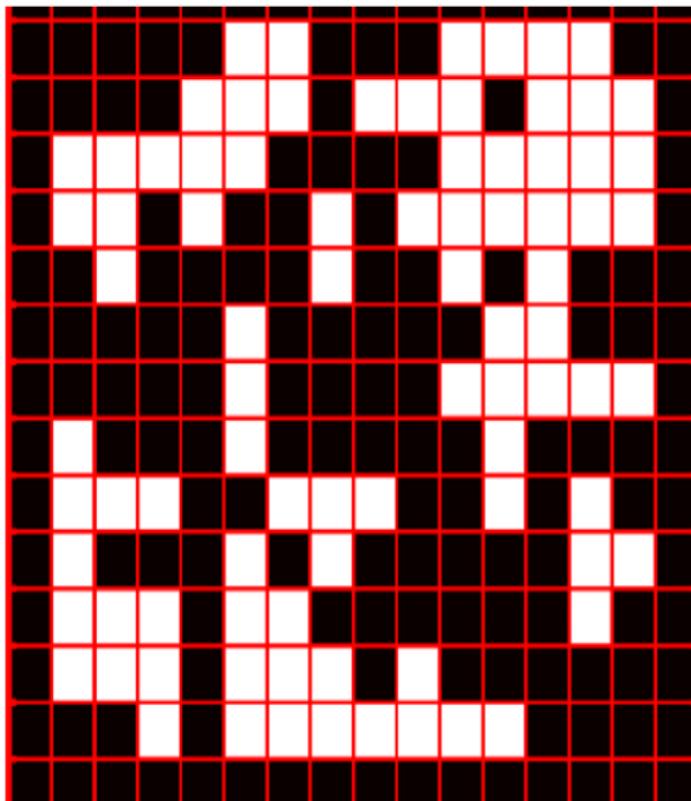
8 and 6

7 and 5

4 and 5

## BLOBs with 4- and 8- connectivity

A BLOB analysis is performed using both 4- and 8- connectivity. How many BLOBS are found using the two different connectivities?



3 and 7

0%

9 and 5

0%

8 and 6 ✓

100%

7 and 5

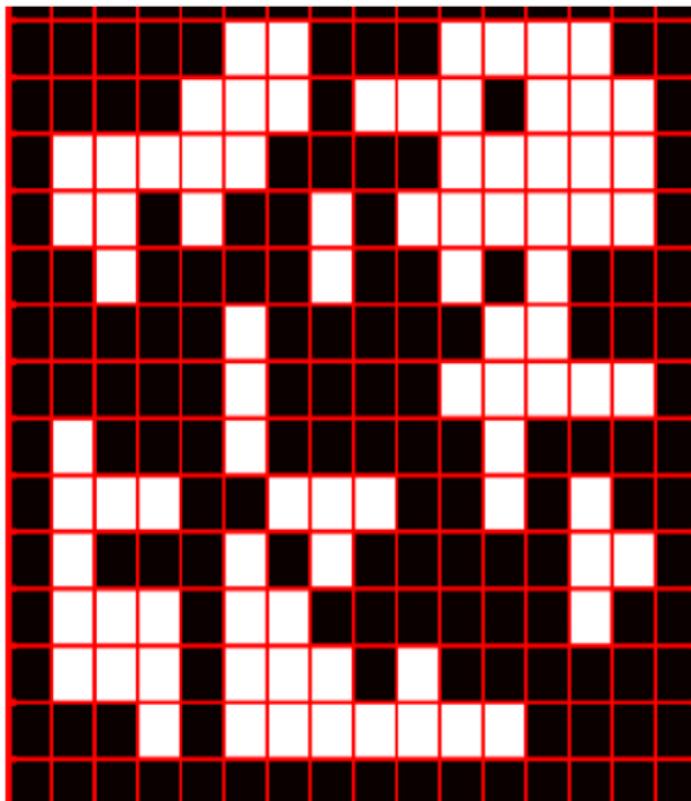
0%

4 and 5

0%

## BLOBs with 4- and 8- connectivity

A BLOB analysis is performed using both 4- and 8- connectivity. How many BLOBS are found using the two different connectivities?



3 and 7

0%

9 and 5

0%

8 and 6 ✓

100%

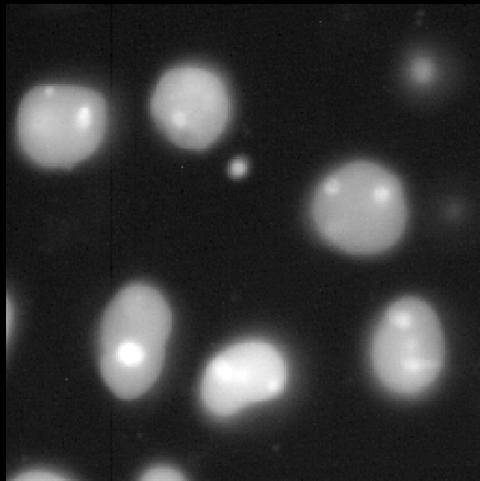
7 and 5

0%

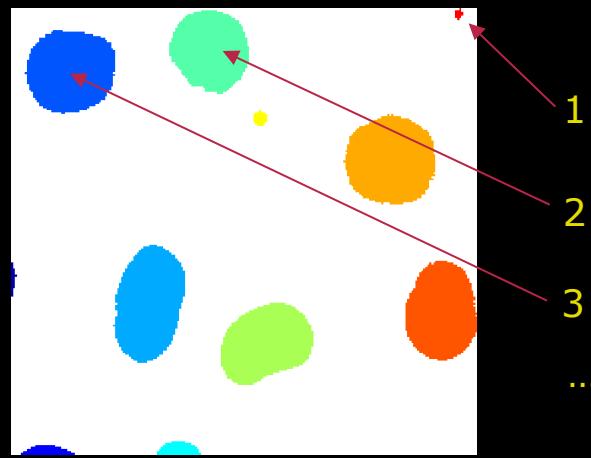
4 and 5

0%

# The result of connected component analysis



- An image where each BLOB (component) is labelled
- Each blob now has a unique ID number
- What do we do with these blobs?



# Features



- Feature
  - A prominent or distinctive aspect, quality, or characteristic
  - *This radio has many good features*
- Car (Ford-T) features
  - 4 wheels
  - 2 doors
  - 540 kg
  - 20 hp

# Feature vector



$f=[4, 2, 540, 20]$

- Feature vector
  - Vector with all the features for one object
- Ford-T features
  - 4 wheels
  - 2 doors
  - 540 kg
  - 20 hp
- Ford Fiesta features
  - 4 wheels
  - 3 doors
  - 1100 kg
  - 90 hp



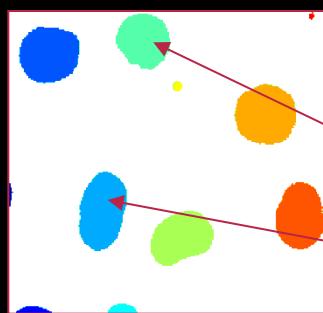
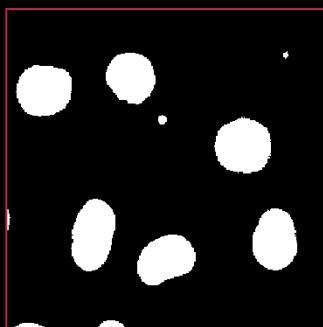
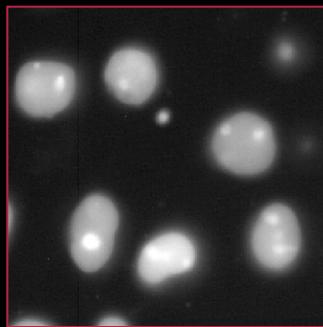
$f=[4, 3, 1100, 90]$

## Visual features to determine car type



design surfaces  
windshield  
seatbelts angles drag  
outline lights  
plate shape wheels  
car size tire  
**windows**  
side roof  
light headlights above  
bumper height  
chasis front wheel  
smooth door license noise  
rims open help  
has windshield angularity

# Feature extractions



- Compute features for each BLOB that can be used to identify it
  - Size
  - Shape
  - Position
- From image operations to mathematical operations
  - **Input:** a list of pixel positions
  - **Output:** Feature vector
- First step: remove invalid BLOBS
  - too small or big- using morphological operations for example
  - border BLOBS

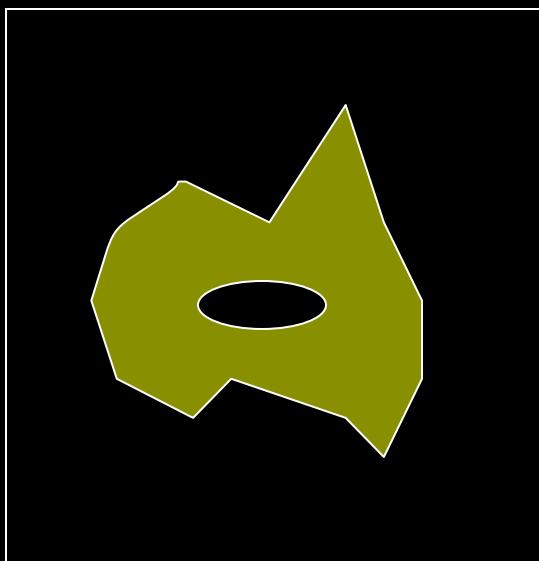
Feature vector = [2,1,...,3]

Feature vector = [4,7,...,0]

# BLOB Features

## ■ Area

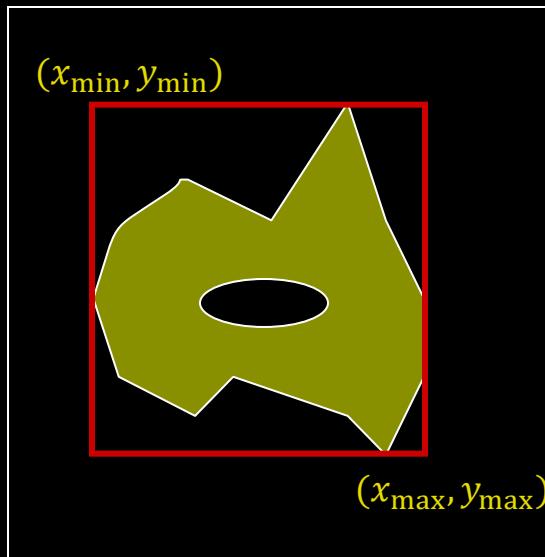
- number of pixels in the BLOB
- Can be used to remove noise (small BLOBS)



One BLOB

# BLOB Features

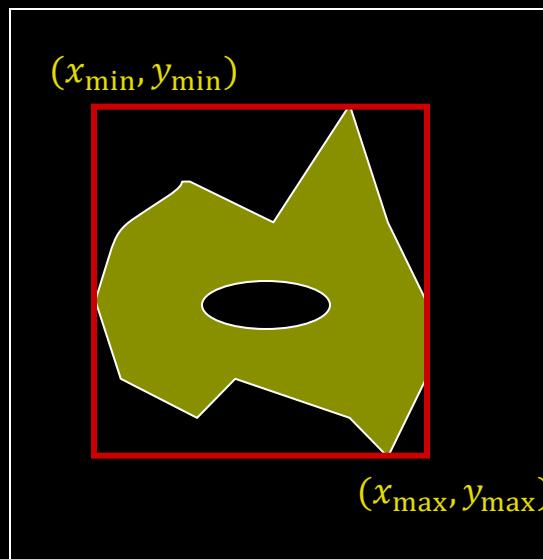
- Bounding box
  - Minimum rectangle that contains the BLOB
  - Height:  $y_{\max} - y_{\min}$
  - Width:  $x_{\max} - x_{\min}$
  - Bounding box ratio:  
$$\frac{y_{\max} - y_{\min}}{x_{\max} - x_{\min}}$$
  - tells if the BLOB is elongated



One BLOB

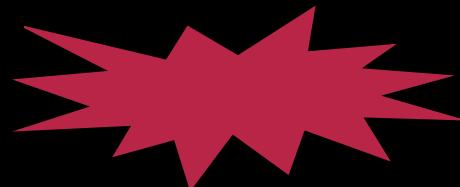
# BLOB Features

- Bounding box
  - Bounding box area:



One BLOB

$$\text{Compactness} = \frac{\text{BLOB Area}}{(y_{\max} - y_{\min}) \cdot (x_{\max} - x_{\min})}$$



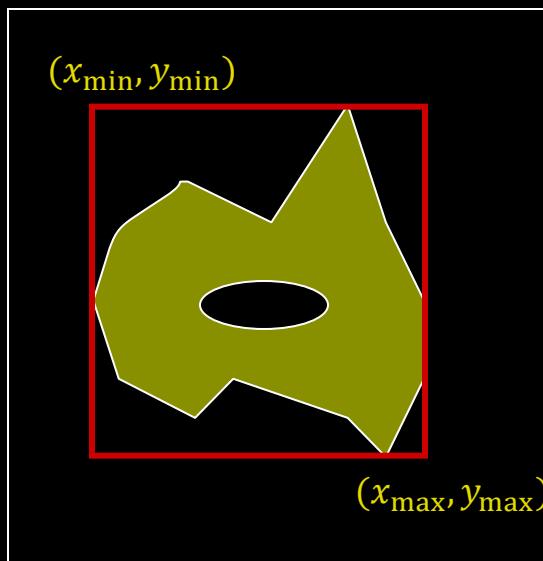
Not compact



Compact

# BLOB Features

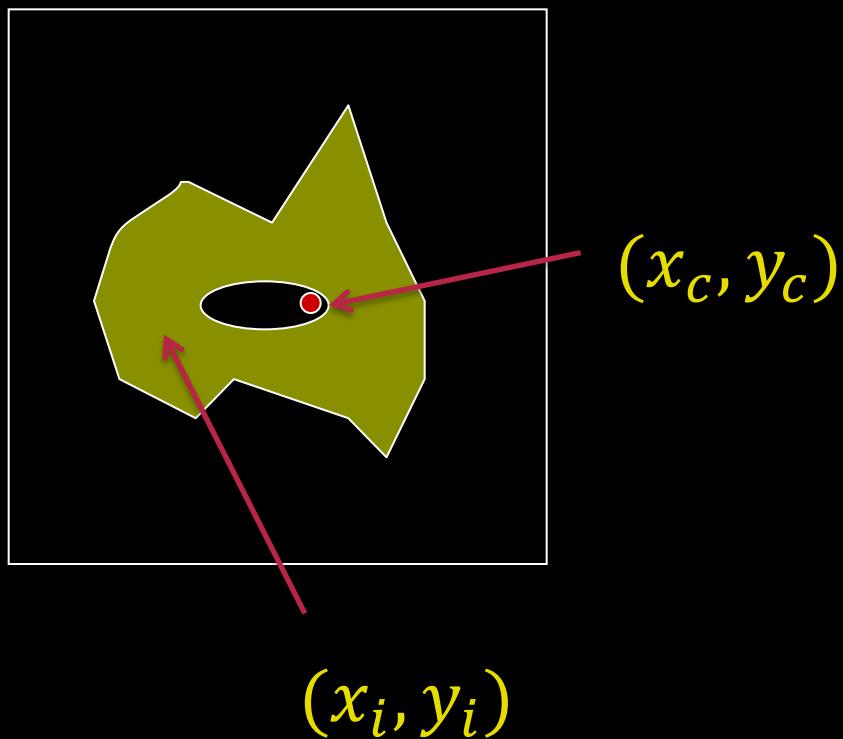
- Bounding box ratio
  - Bounding box height divided by the width



One BLOB

# BLOB Features

- Center of mass  $(x_c, y_c)$

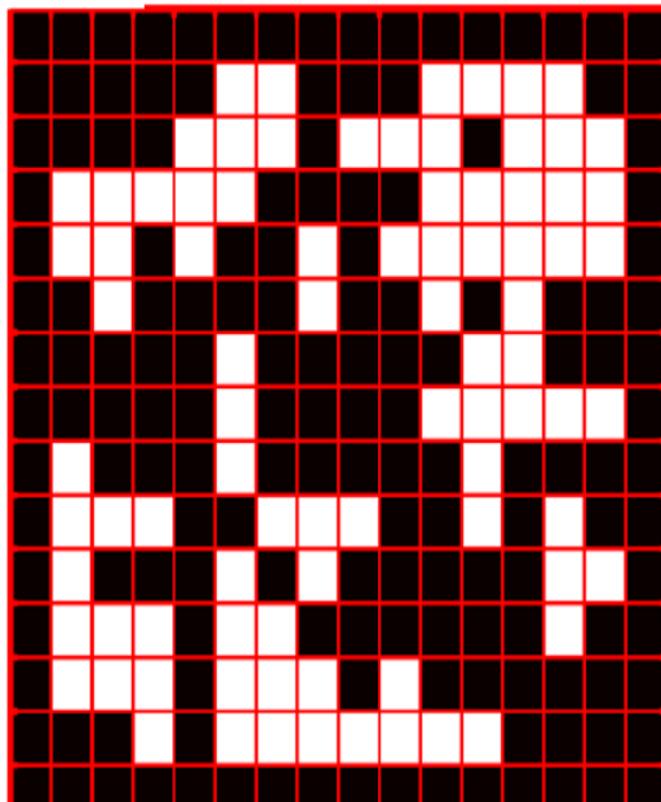


$$x_c = \frac{1}{N} \sum_{i=1}^N x_i$$

$$y_c = \frac{1}{N} \sum_{i=1}^N y_i$$

## BLOB Center of Mass

The smallest BLOB is found using 4-connectivity. What is the center of mass of this BLOB. The image has origin (0,0) and uses a (x,y) coordinate system.



(12, 1.5)

(5, 8.5)

(6.5, 3.5)

(4.5, 0.5)

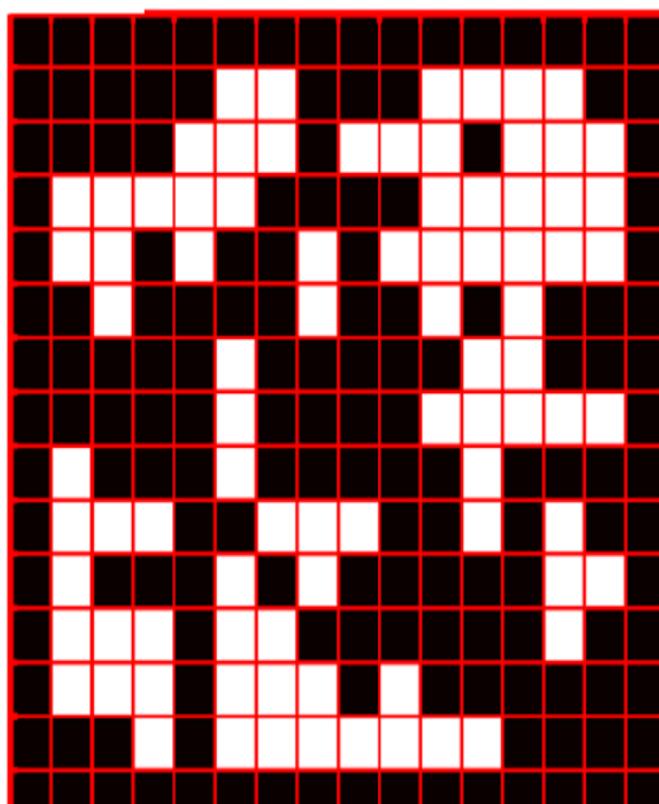
(7, 4.5)



Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)

## BLOB Center of Mass

The smallest BLOB is found using 4-connectivity. What is the center of mass of this BLOB. The image has origin (0,0) and uses a (x,y) coordinate system.



(12, 1.5)

0%

(5, 8.5)

3%

(6.5, 3.5)

6%

(4.5, 0.5)

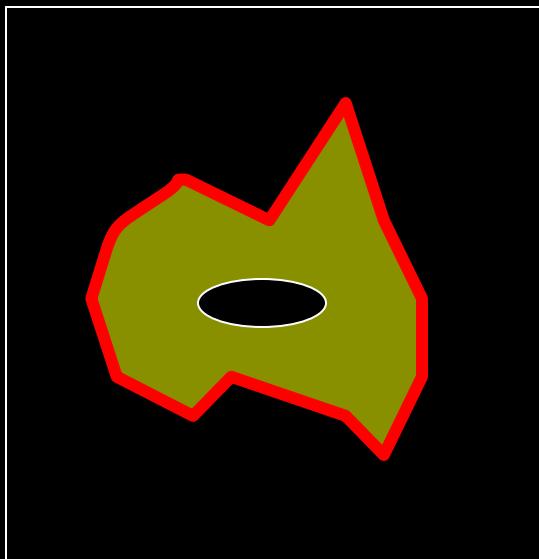
0%

(7, 4.5)

90%

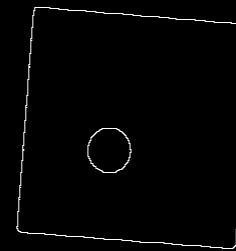
# BLOB Features

- Perimeter
  - Length of perimeter
  - How can we compute that?
- In practice, it is computed differently and more accurately

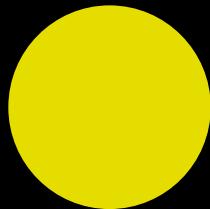


One BLOB

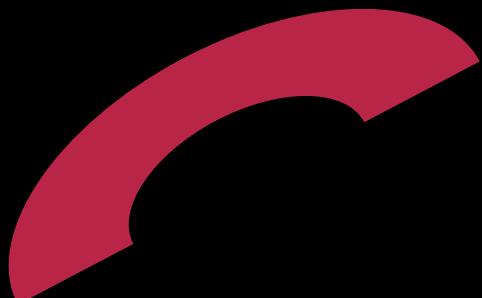
$$\sum((f(x,y) \oplus SE) - f(x,y))$$



# BLOB Features - circularity



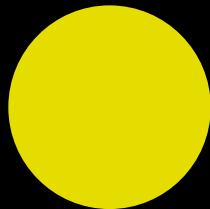
Circle like



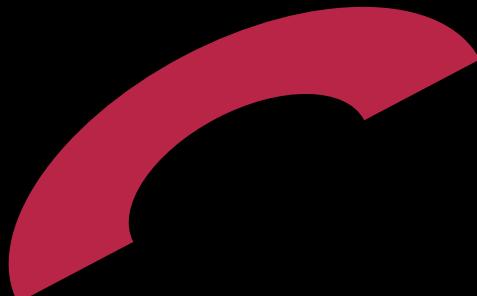
Not circle like

- How much does it look like a circle?
- Circle
  - Area  $A = \pi r^2$
  - Perimeter  $P = 2\pi r$
- New object assumed to be a circle
  - Measured perimeter  $P_m$
  - Measured area  $A_m$
- Estimate perimeter from (measured) area
  - Estimated perimeter  $P_e = 2\sqrt{\pi A_m}$

# BLOB Features - circularity



Circle like



Not circle like

- Compare the perimeters
  - Measured perimeter  $P_m$
  - Estimated perimeter  $P_e = 2\sqrt{\pi A_m}$
- Circularity 1:

$$\text{Circularity} = \frac{P_m}{P_e} = \frac{P_m}{2\sqrt{\pi A_m}}$$

## Circularity math



$$P_m < P_e$$

$$P_m = P_e$$

$$P_m > P_e$$

$$\text{Circularity} = \frac{P_m}{P_e} = \frac{P_m}{2\sqrt{\pi A_m}}$$

# Circularity math

$P_B < P_C$

20%



3%

$P_B > P_C$



77%

## Circularity math



$P_m < P_e$



$P_m = P_e$

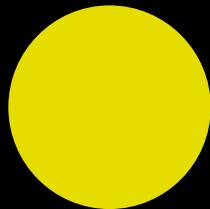


$$\text{Circularity} = \frac{P_m}{P_e} = \frac{P_m}{2\sqrt{\pi A_{n_i}}}$$

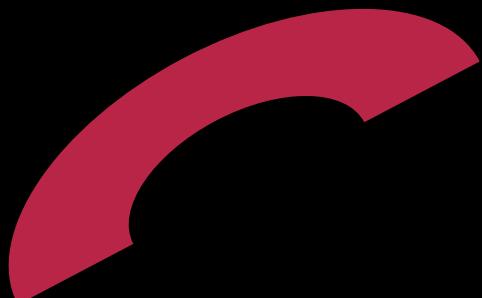
$P_m > P_e$



# BLOB Features - circularity



Circle like



Not circle like

- Compare the perimeters

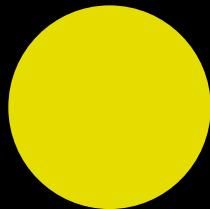
- Measured perimeter  $P_m$
- Estimated perimeter  $P_e = 2\sqrt{\pi A_m}$

- Circularity:

$$\text{Circularity} = \frac{P_m}{P_e} = \frac{P_m}{2\sqrt{\pi A_m}}$$

- This measure will normally be  $\geq 1$

# BLOB Features – circularity inverse



Circle like



Not circle like

- Compare the perimeters
  - Measured perimeter  $P_m$
  - Estimated perimeter  $P_e = 2\sqrt{\pi A_m}$

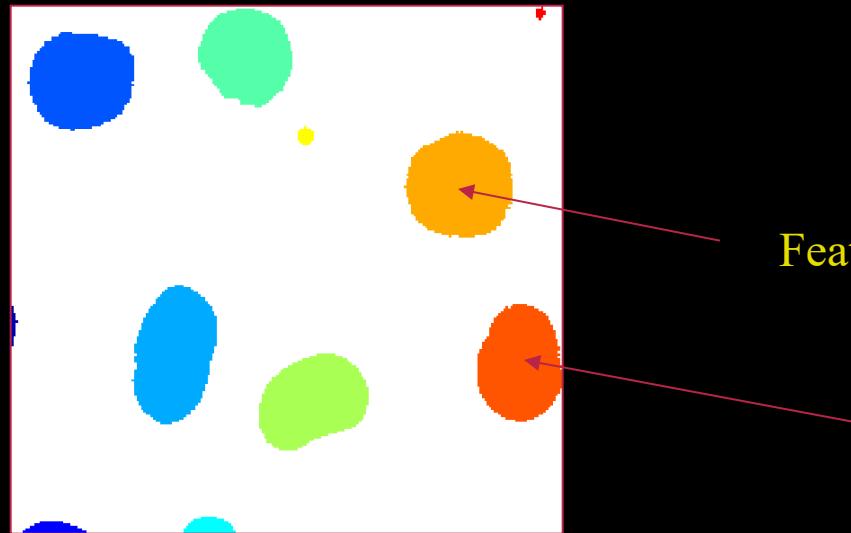
- Circularity (inverse):

$$\text{Circularity inverse} = \frac{P_e}{P_m} = \frac{2\sqrt{\pi A_m}}{P_m}$$

- This measure will normally be  $\leq 1$

# After feature extraction

Area, compactness, circularity etc calculated for all BLOB



Feature vector = [2,1,...,3]

Feature vector = [4,7,...,0]

One feature vector per blob

# BLOB Classification

## ■ Classification

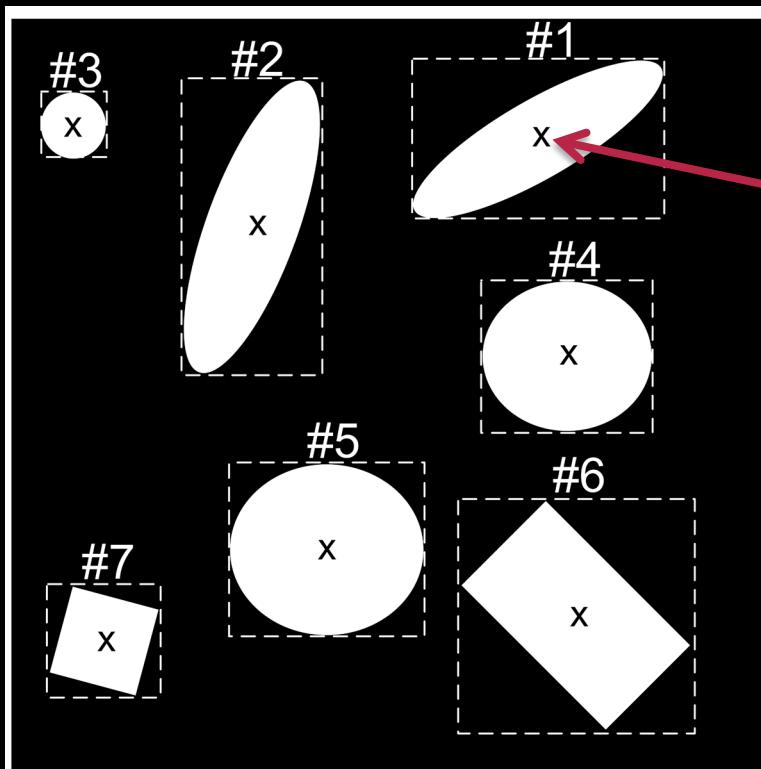
- Put a BLOB into a *class*

## ■ *Classes* are normally pre-defined

- *Car*
- *Bus*
- *Motorcycle*
- *Scooter*

## ■ Object recognition

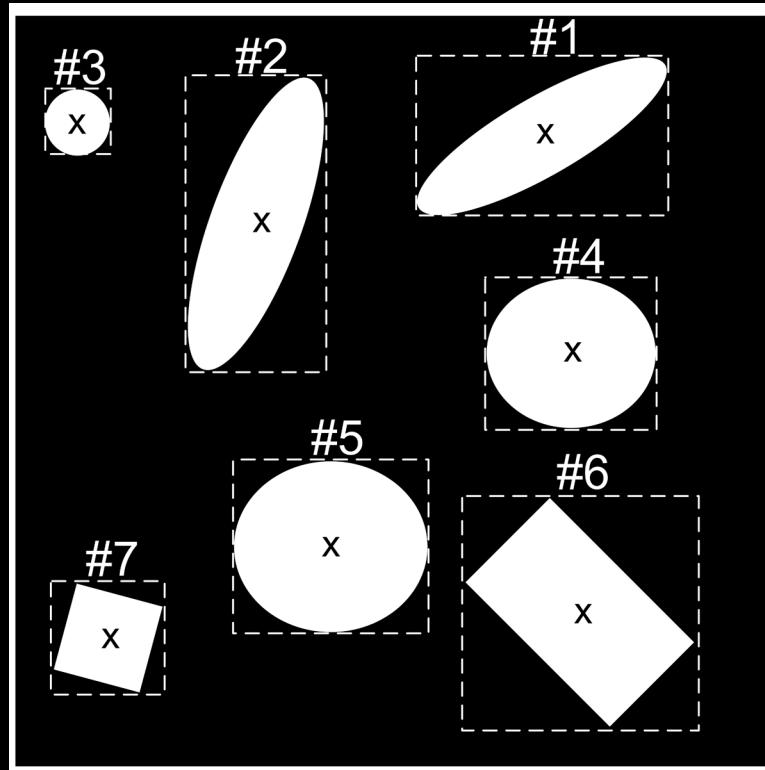
# Object recognition: Circle example



BLOB number	Circu-larity	Area (pixels)
1	0.31	6561
2	0.40	6544
3	0.98	890
4	0.97	6607
5	0.99	6730
6	0.52	6611
7	0.75	2073

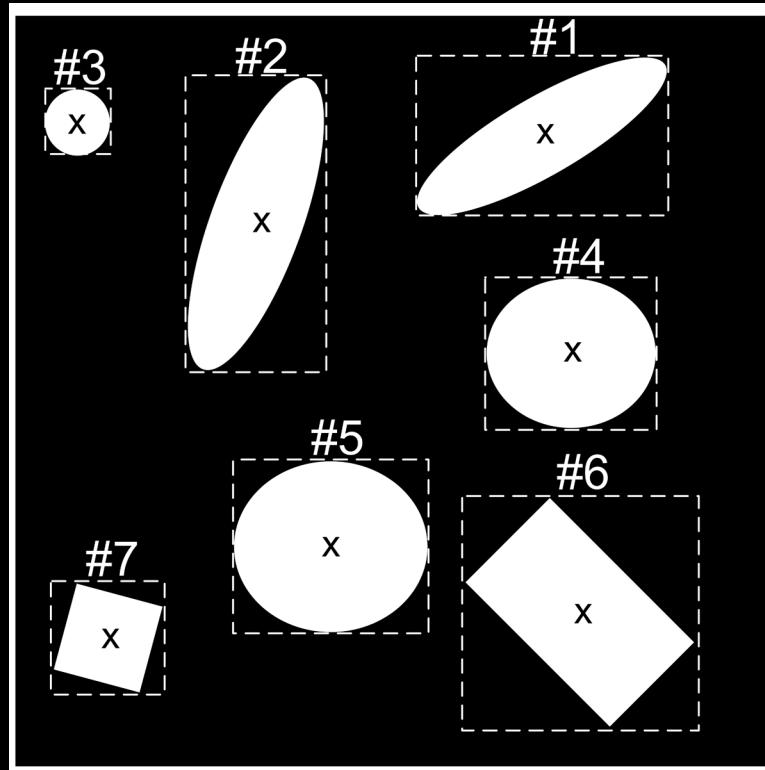
Which objects are circles?

# Circle classification



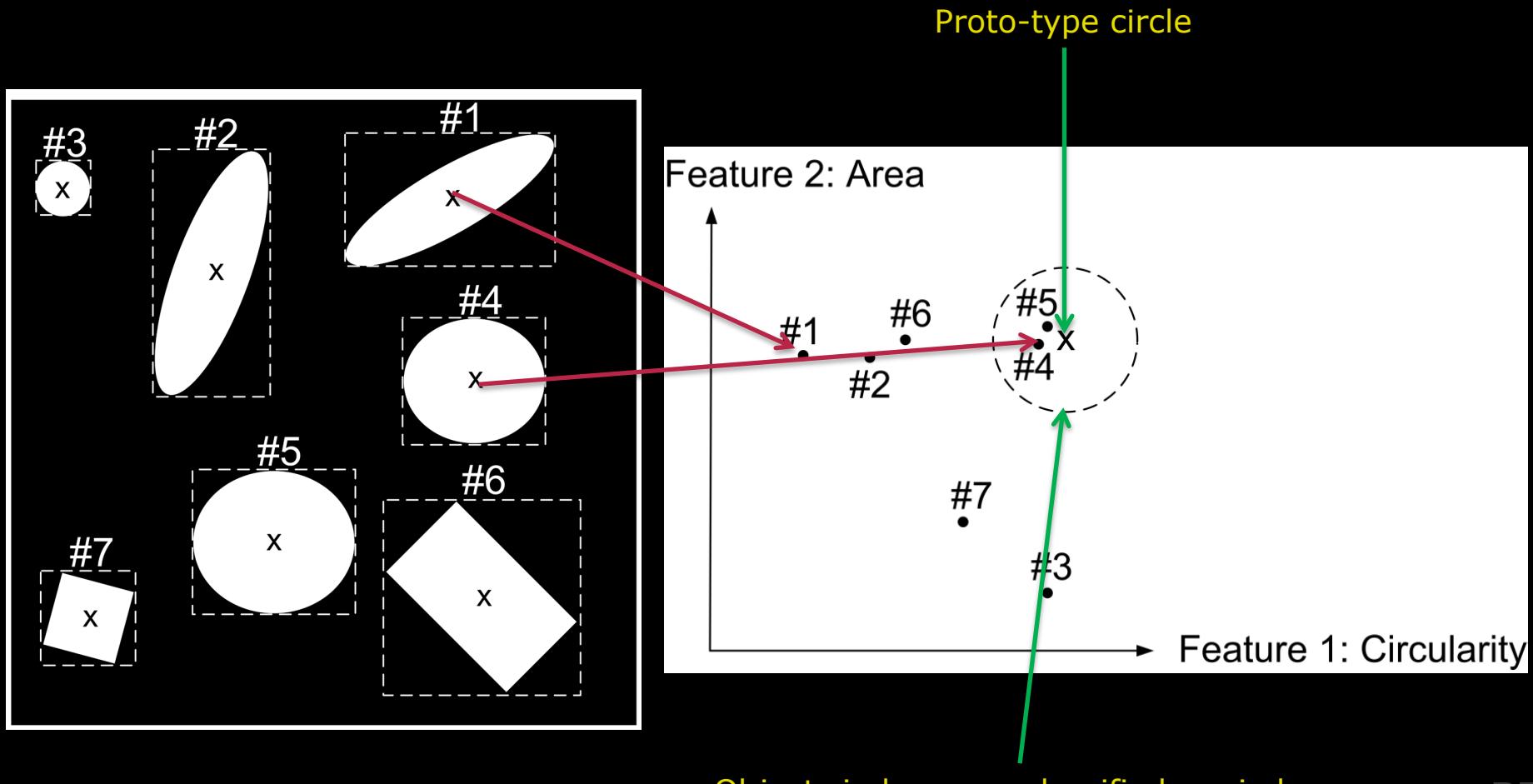
- Two classes:
  - Circle
  - Not-circle
- Lets make a model of a *proto-type* circle

# Circle classification

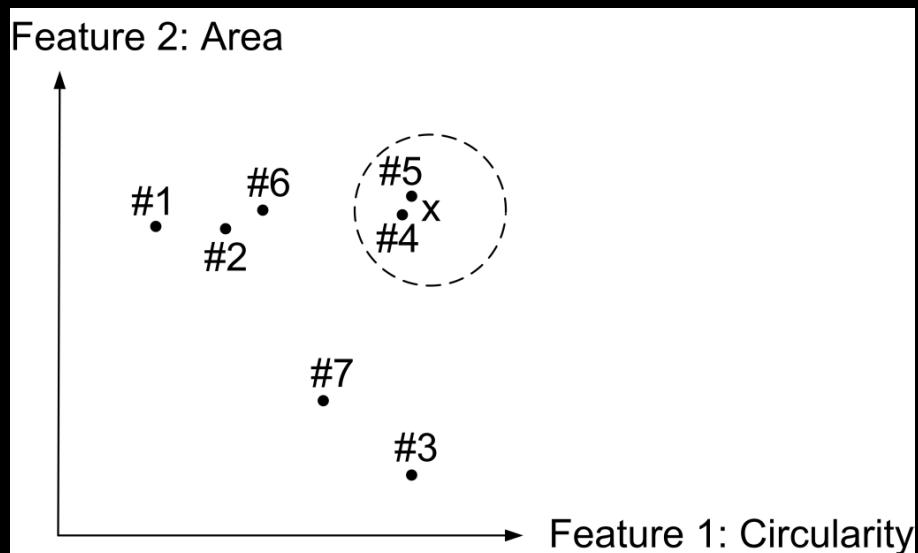


- Proto-type circle
  - Circularity : 1
  - Area: 6700

# Feature Space

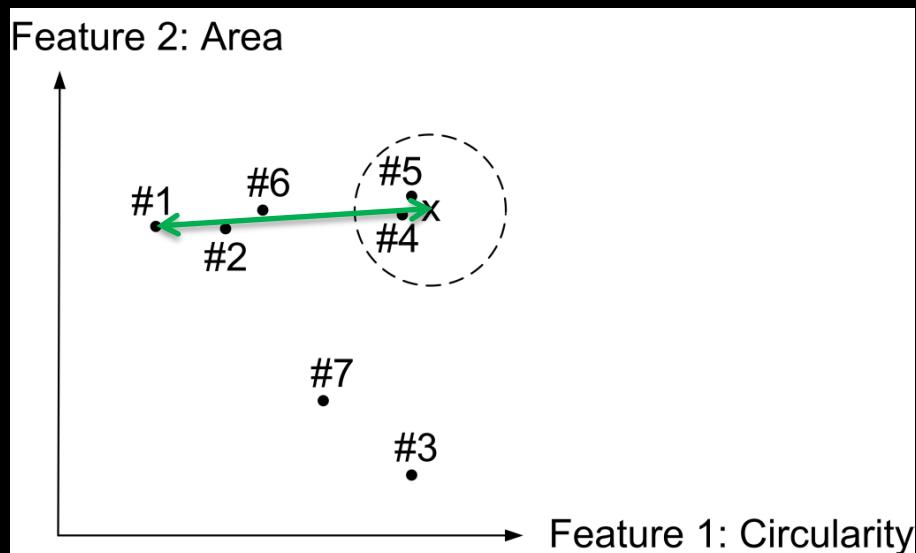


# Feature space



- Proto-type circle
  - Circularity : 1
  - Area: 6700
- Some slack is added to allow non-perfect circles
  - Circularity: 1 +/- 0.15

# Feature space - distances



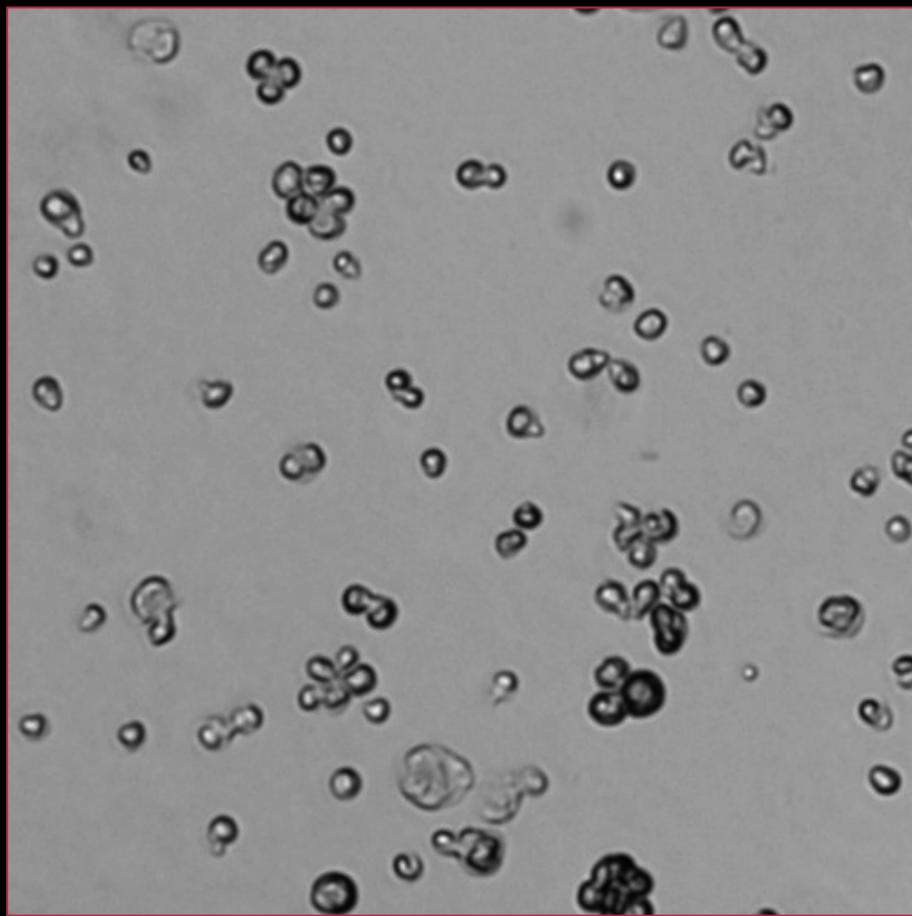
- How do we decide if an object is inside the circle?
- Feature space distance
- Euclidean distance in features space

Blob 1: circularity: 0.31, Area : 6561

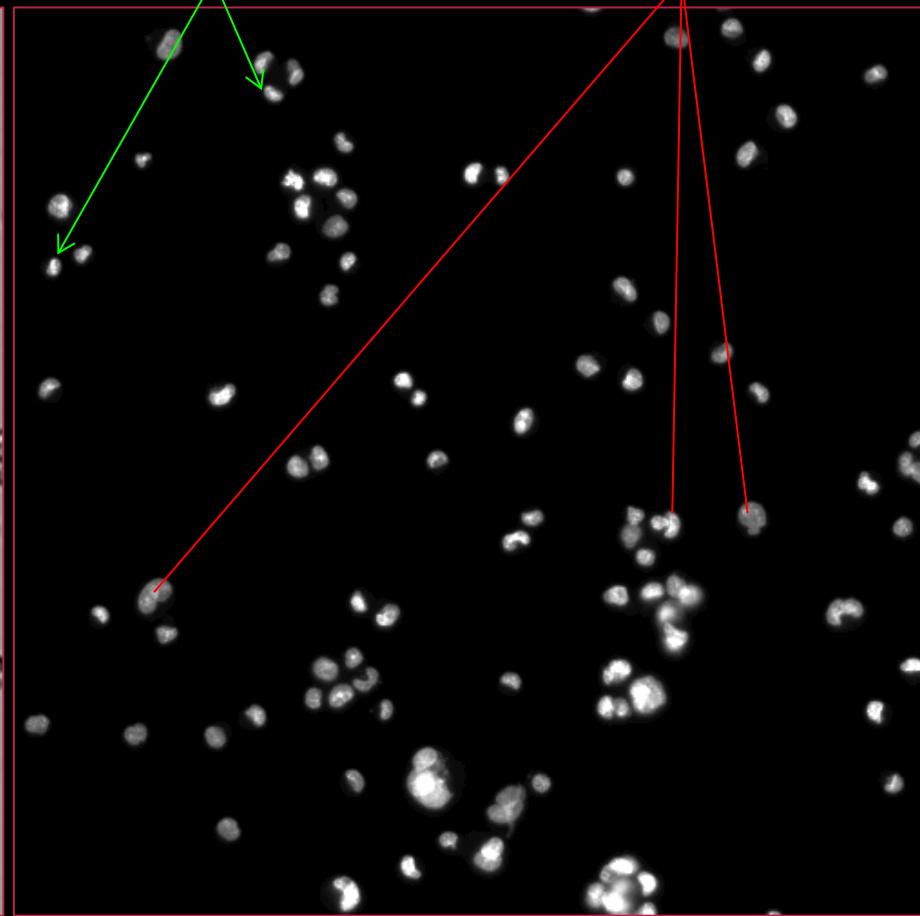
$$D = \sqrt{(0.31 - 1)^2 + (6561 - 6700)^2}$$

Dominoes all! – normalisation needed

# Cell classification



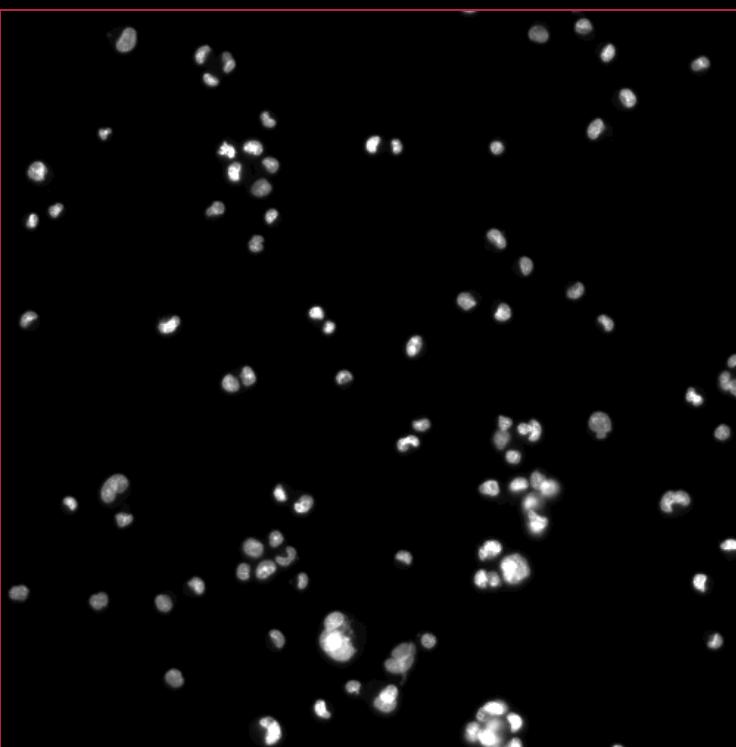
UV Microscopy



Fluorescence Microscopy (DAPI)

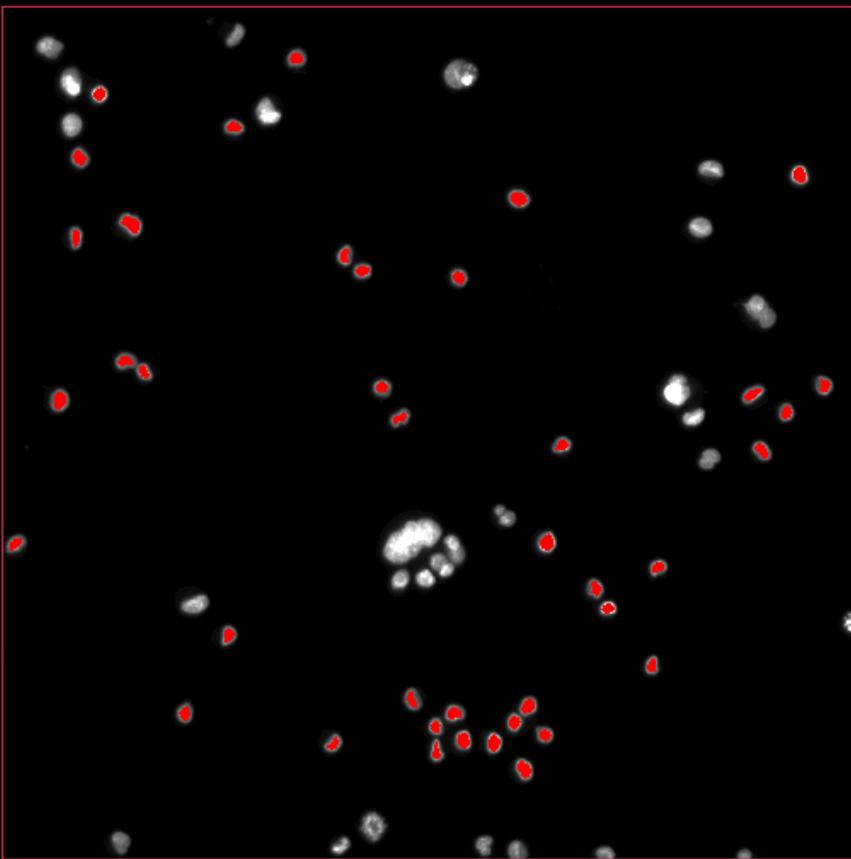
Images from ChemoMetec A/S

# Nuclei classification



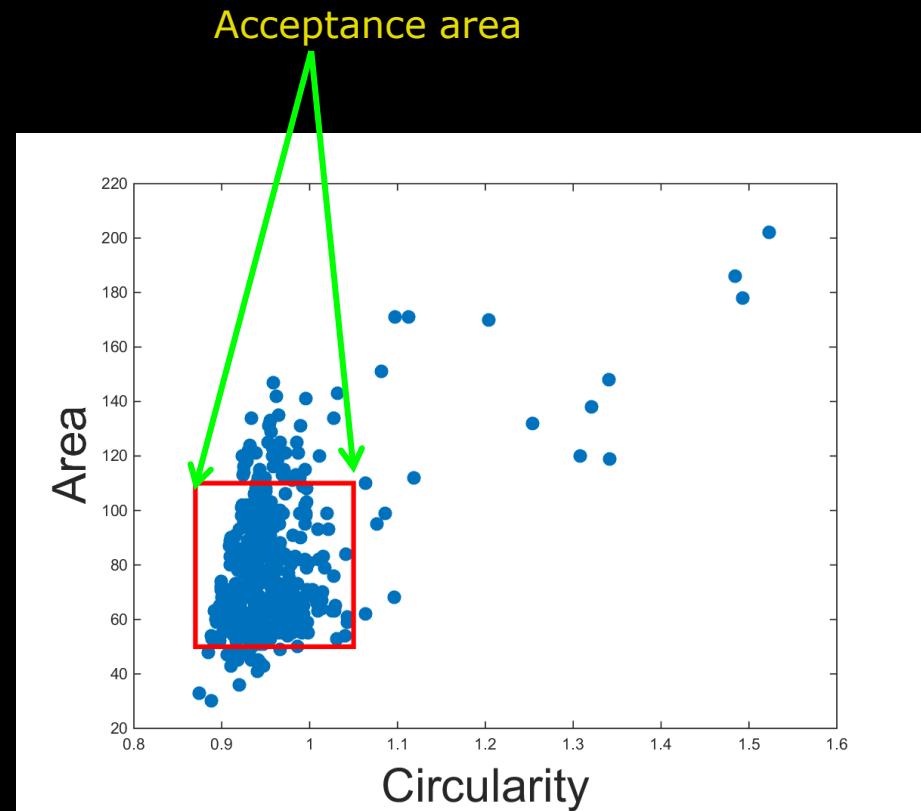
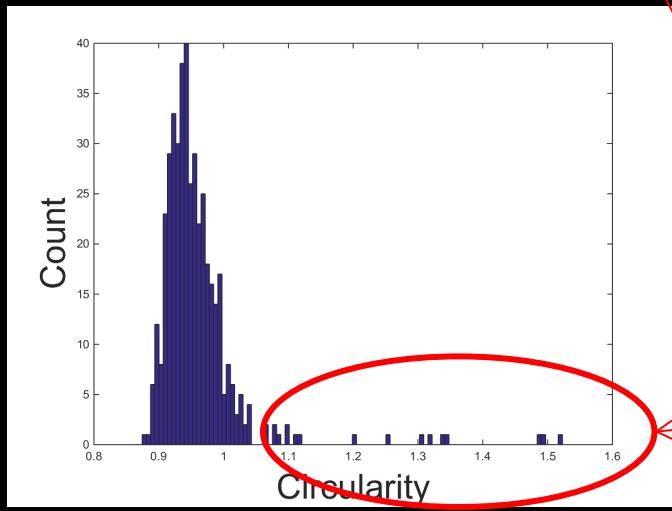
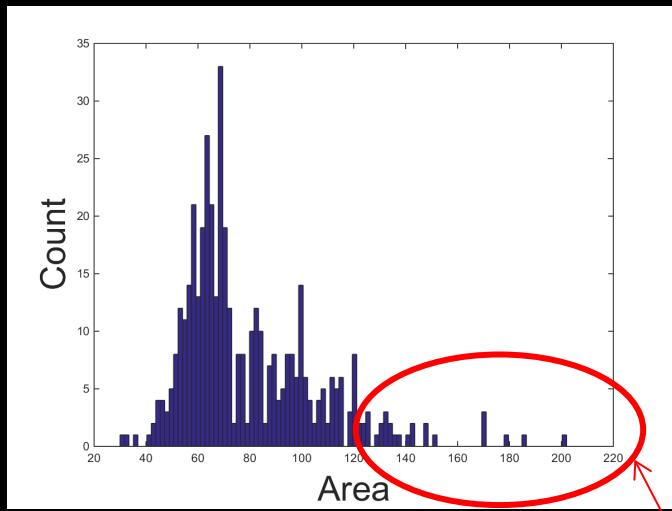
- DAPI image
- Two classes
  - Single nuclei
  - Noise
    - Multiple nuclei together
    - Debris
    - Other noise

# Training and annotation



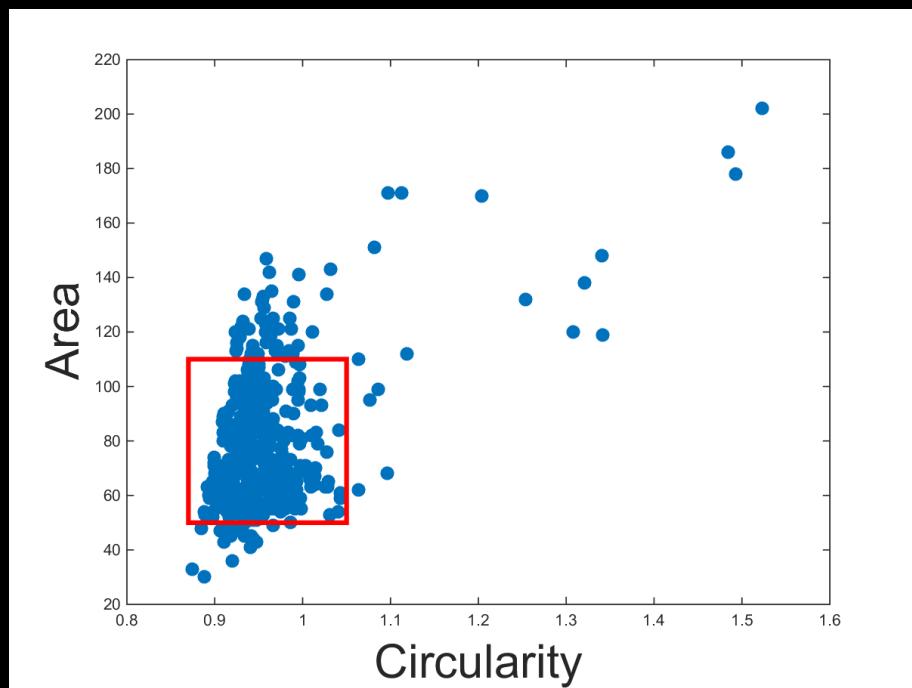
- Selection of true single nuclei marked
  
- Thresholding
- BLOB Analysis
  - Circularity
  - Area

# Training data - analysis



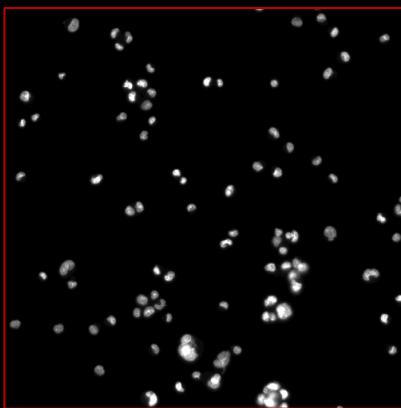
Probably outliers

# Feature ranges



Feature	Min	Max
Area	50	110
Circularity	0.87	1.05

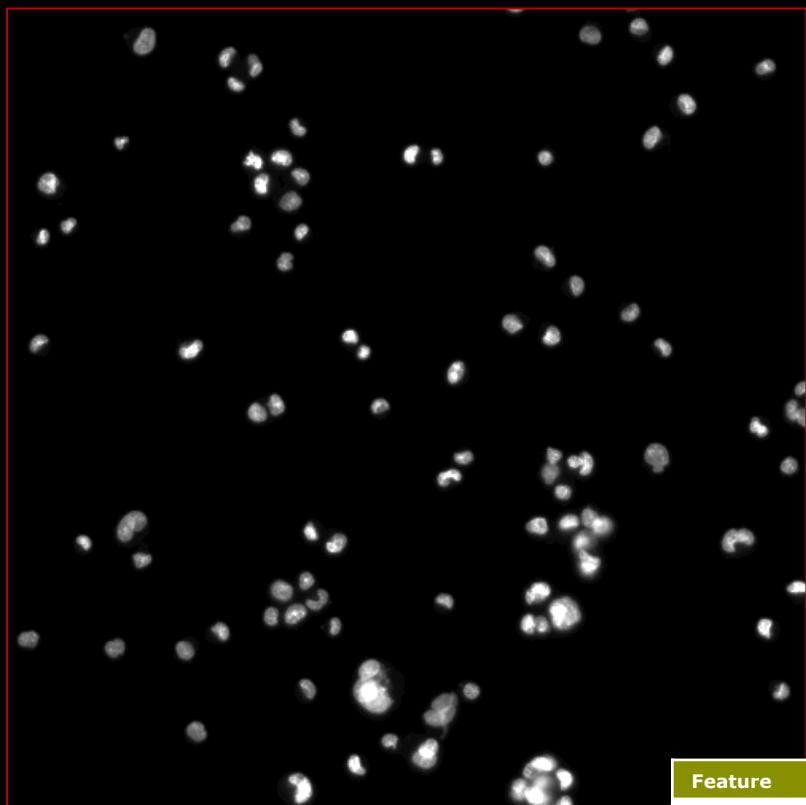
# Using the classifier



DAPI input image

- Threshold input image
- Morphological opening (SE 5x5)
- Morphological closing (SE 5x5)
- BLOBs found using 8-neighbours
- Border BLOBS removed
- BLOB features computed
  - Area + circularity
- BLOBs with features inside the acceptance range are **single-nuclei**

# Using the classifier



DAPI input image



Found single nuclei

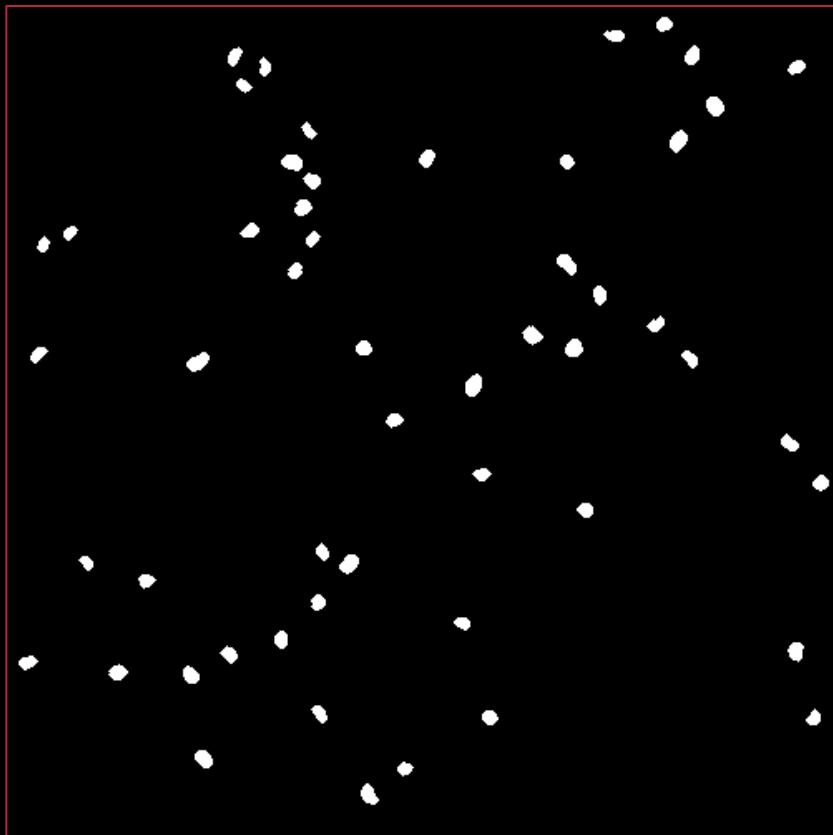
Feature	Min	Max
Area	50	110
Circularity	0.87	1.05

# How well does it work?

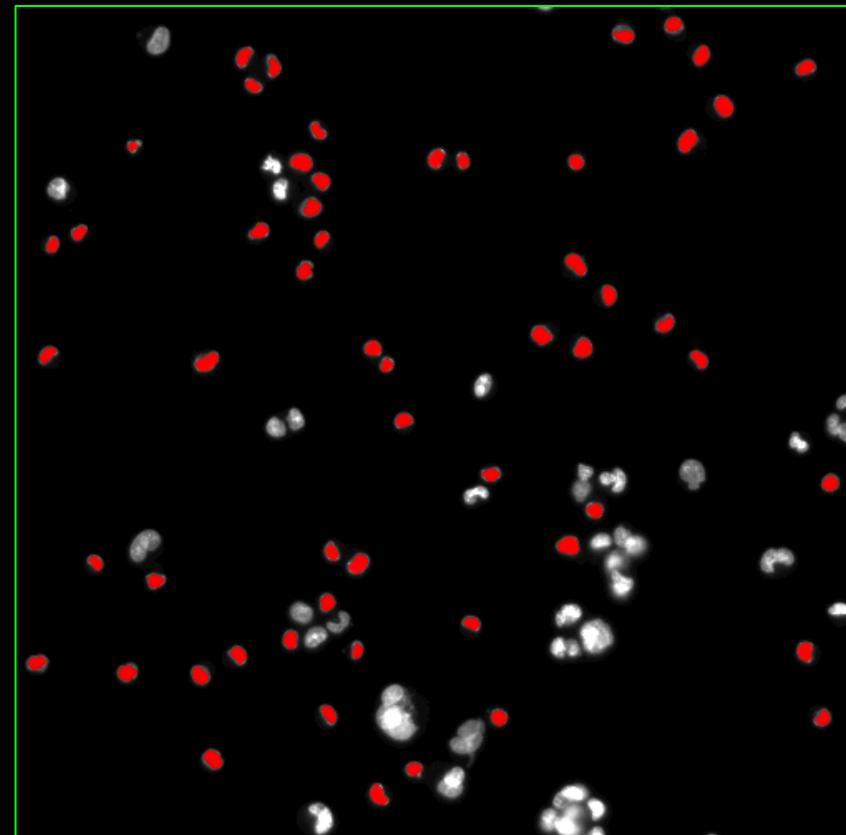


- We say we have a **great** algorithm!
- Strangely the doctor/biochemist do not trust this statement!
  - They need numbers!
- How do we report the performance?

# Creating ground truth – expert annotations



Found single nuclei



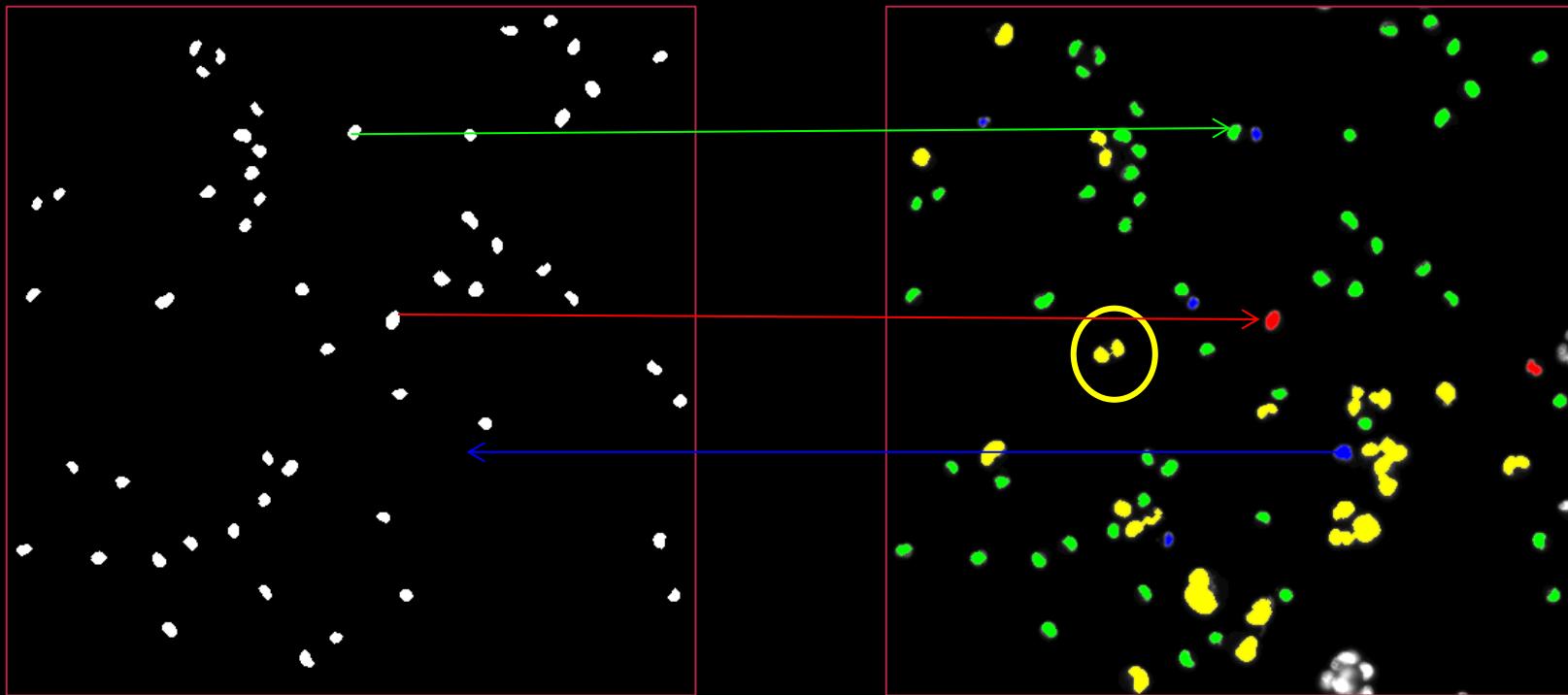
Expert opinion on true single nuclei

Red markings: Single nuclei

Not marked: Noise

# Four cases

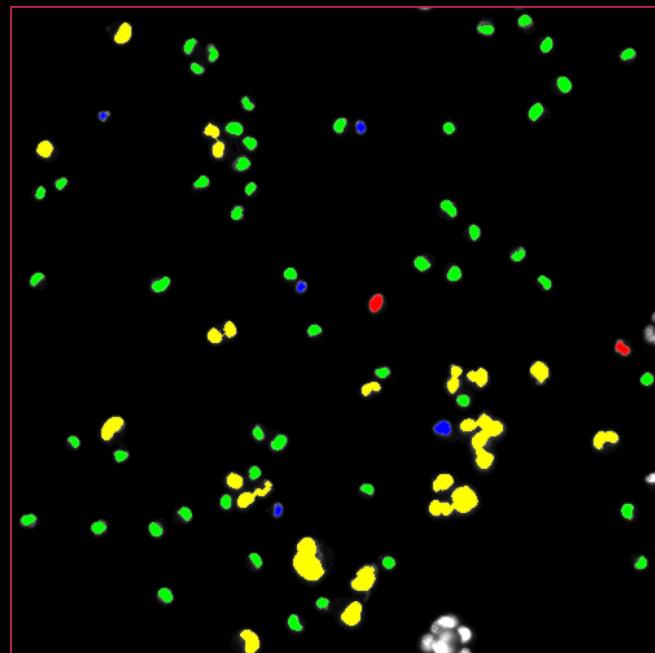
- True Positive (TP): A nuclei is classified as a nuclei
- True Negative (TN): A noise object is classified as noise object
- False Positive (FP): A noise object is classified as a nuclei
- False Negative (FN): A nuclei is classified as a noise object



Found single nuclei

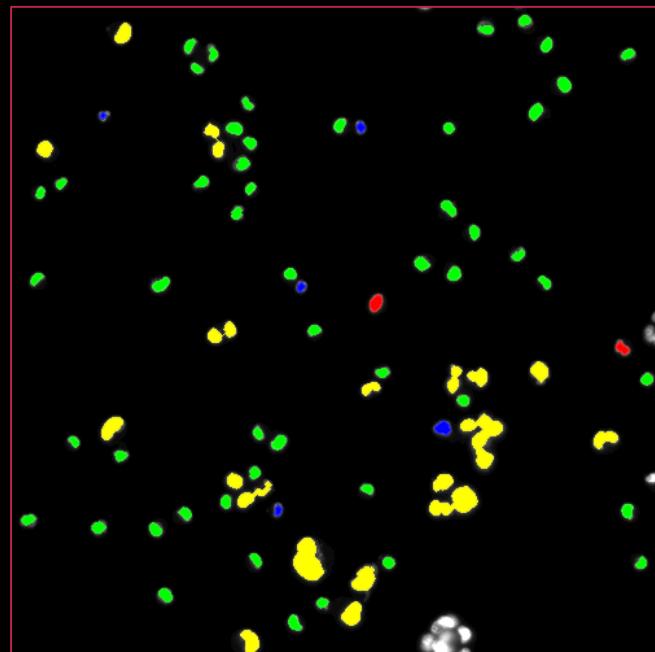
# Confusion matrix

	Predicted as noise	Predicted as single-nuclei
Actual noise		
Actual single-nuclei		



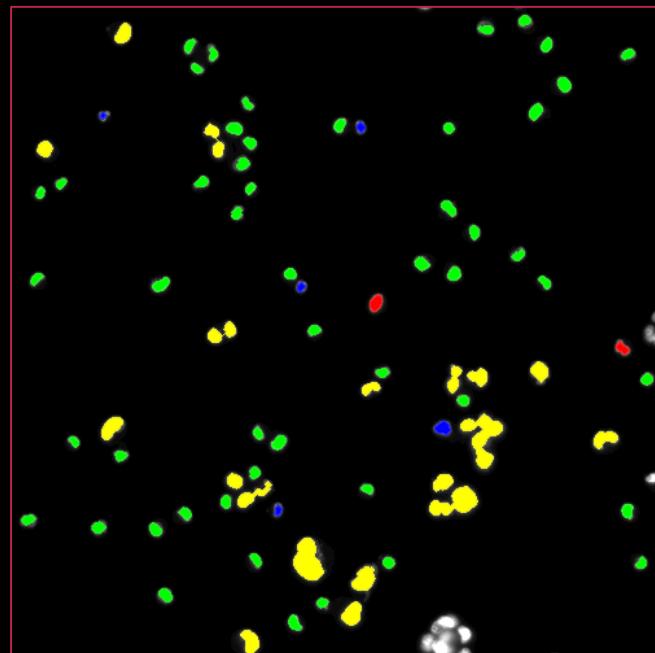
# Confusion matrix

	<b>Predicted as noise</b>	<b>Predicted as single-nuclei</b>
<b>Actual noise</b>	TN=19	
<b>Actual single-nuclei</b>		



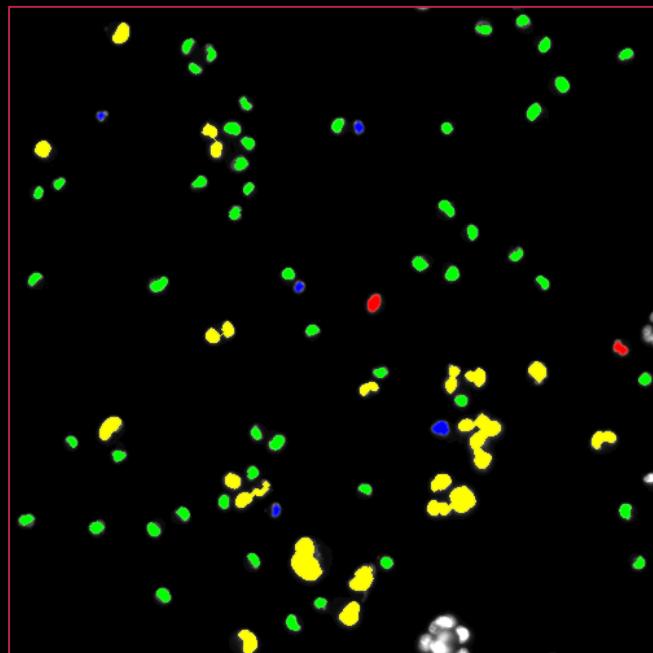
# Confusion matrix

	<b>Predicted as noise</b>	<b>Predicted as single-nuclei</b>
<b>Actual noise</b>	TN=19	
<b>Actual single-nuclei</b>		TP=51



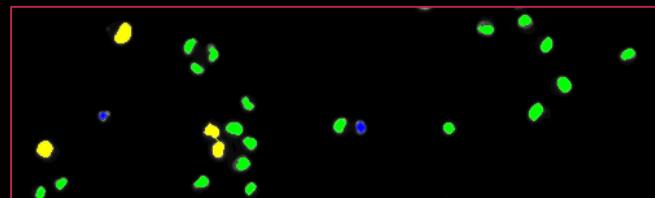
# Confusion matrix

	<b>Predicted as noise</b>	<b>Predicted as single-nuclei</b>
Actual noise	TN=19	FP=2
Actual single-nuclei		TP=51

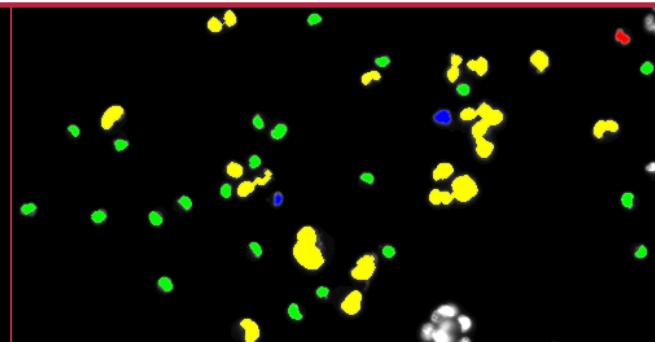


# Confusion matrix

	<b>Predicted as noise</b>	<b>Predicted as single-nuclei</b>
Actual noise	TN=19	FP=2
Actual single-nuclei	FN=5	TP=51



Something simpler?



# Accuracy

- Tells how often the classifier is correct

$$\text{Accuracy} = \frac{TP + TN}{N}$$

- N is the total number of annotated objects

$$N = TN + TP + FP + FN$$

## Accuracy from Confusion Matrix

	Predicted as noise	Predicted as single-nuclei
Actual noise	TN=19	FP=2
Actual single-nuclei	FN=5	TP=51

42%

65%

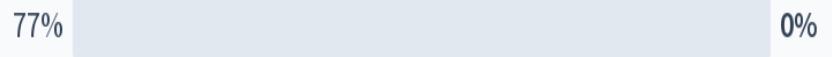
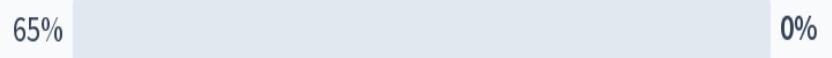
77%

91%

97%

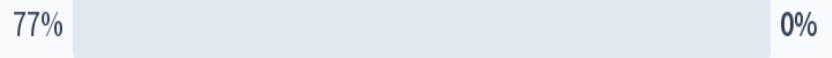
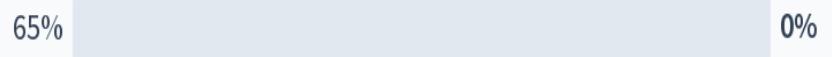
## Accuracy from Confusion Matrix

	Predicted as noise	Predicted as single-nuclei
Actual noise	TN=19	FP=2
Actual single-nuclei	FN=5	TP=51



## Accuracy from Confusion Matrix

	Predicted as noise	Predicted as single-nuclei
Actual noise	TN=19	FP=2
Actual single-nuclei	FN=5	TP=51

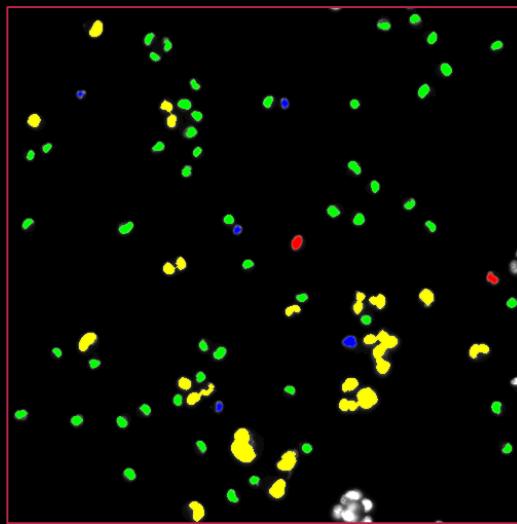


# True positive rate (sensitivity)

- How often is a positive predicted when it actually is positive

$$\text{Sensitivity} = \frac{TP}{FN+TP}$$

All the experts true single-nuclei

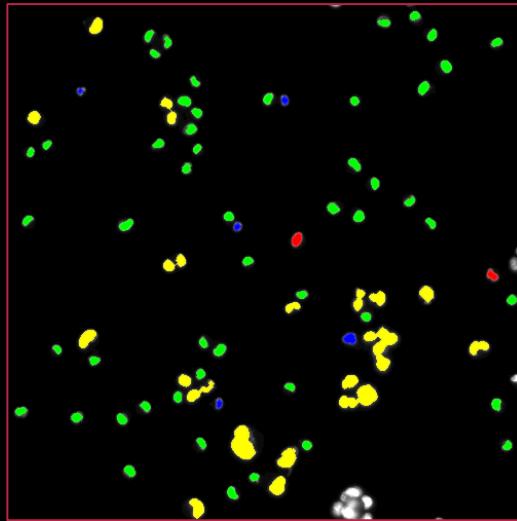


# Specificity

- How often is a negative predicted when it actually is negative

$$\text{Specificity} = \frac{TN}{TN + FP}$$

All the experts true noise objects



## True positive rate

77%

92%

81%

55%

67%

## True positive rate

You have made an algorithm that can locate neon fish in an aquarium. An expert has marked all neon fish in an image as seen in Figure 1 (left). The result of your algorithm is seen in Figure 1 (right). What is the true positive rate of your algorithm?

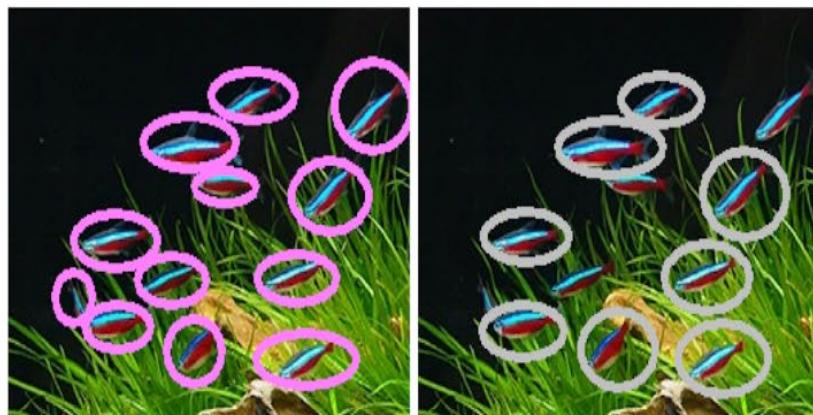


Figure 1: Image of aquarium with neon fish. Left: Expert markings are shown as ellipses. Right: Algorithm markings are shown as ellipses.

77%

0%

92%

0%

81%

5%

55%

0%

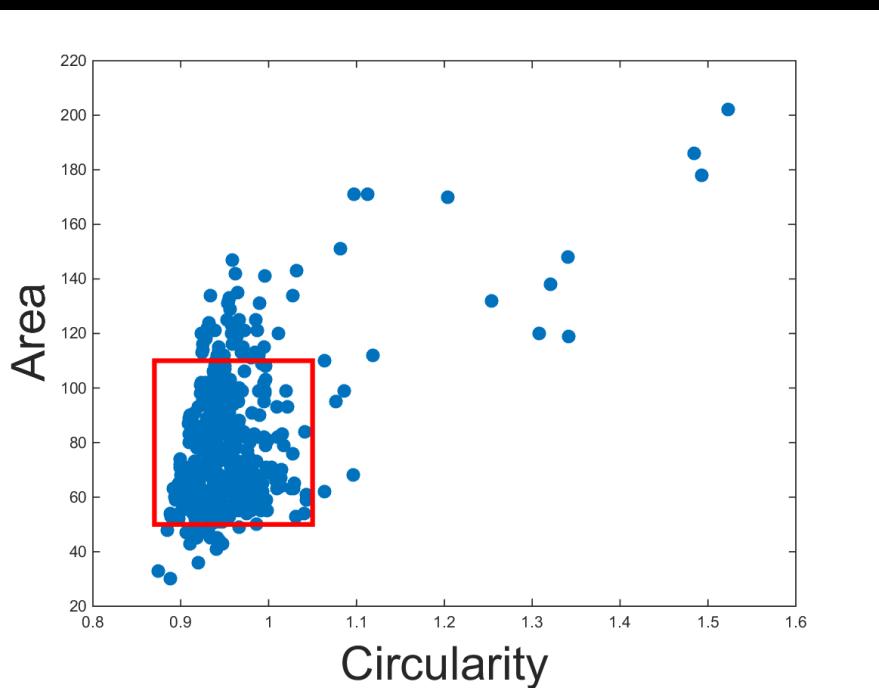
67% ✓

95%

## True positive rate



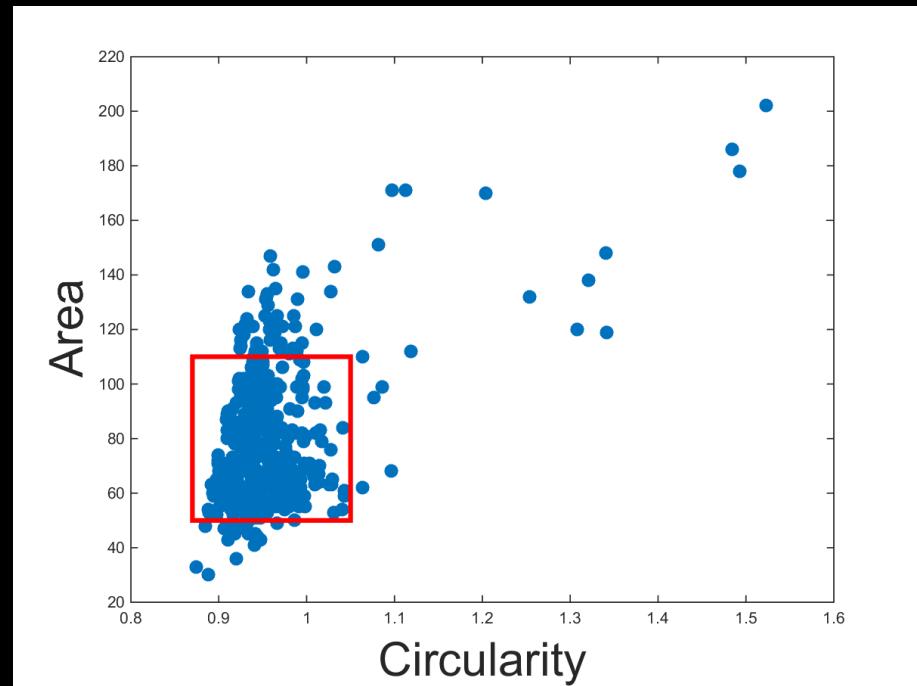
# Optimising the classification



- Changing the classification limits
- The rates will be changed:
  - Accuracy
  - Sensitivity
  - Specificity
  - ...
- Very dependent on the task what is optimal

# Dependencies

- Increasing **true positive rate**
  - Increased **false positive rate**
  - Decreased **precision**



## Example – cell analysis

- We want **only** single-nuclei cells
  - For further analysis
- We **do not** want to do an analysis of a noise object
- We are **not** interested in the true number of single nuclei

## What measure is the most important?

- We want **only** single-nuclei cells
  - For further analysis
- We **do not** want to do an analysis of noise objects
- We are **not** interested in the true number of single nuclei

Low false positives

High true positives

High true negatives

Low false negatives

## What measure is the most important?

Low false positives ✓ 82%

■ We want **only** single-nuclei cells

- For further analysis

High true positives 7%

■ We **do not** want to do an analysis  
of noise objects

■ We are **not** interested in the true  
number of single nuclei

High true negatives 4%

Low false negatives 7%

## What measure is the most important?

Low false positives ✓ 82%

■ We want **only** single-nuclei cells

- For further analysis

High true positives 7%

■ We **do not** want to do an analysis  
of noise objects

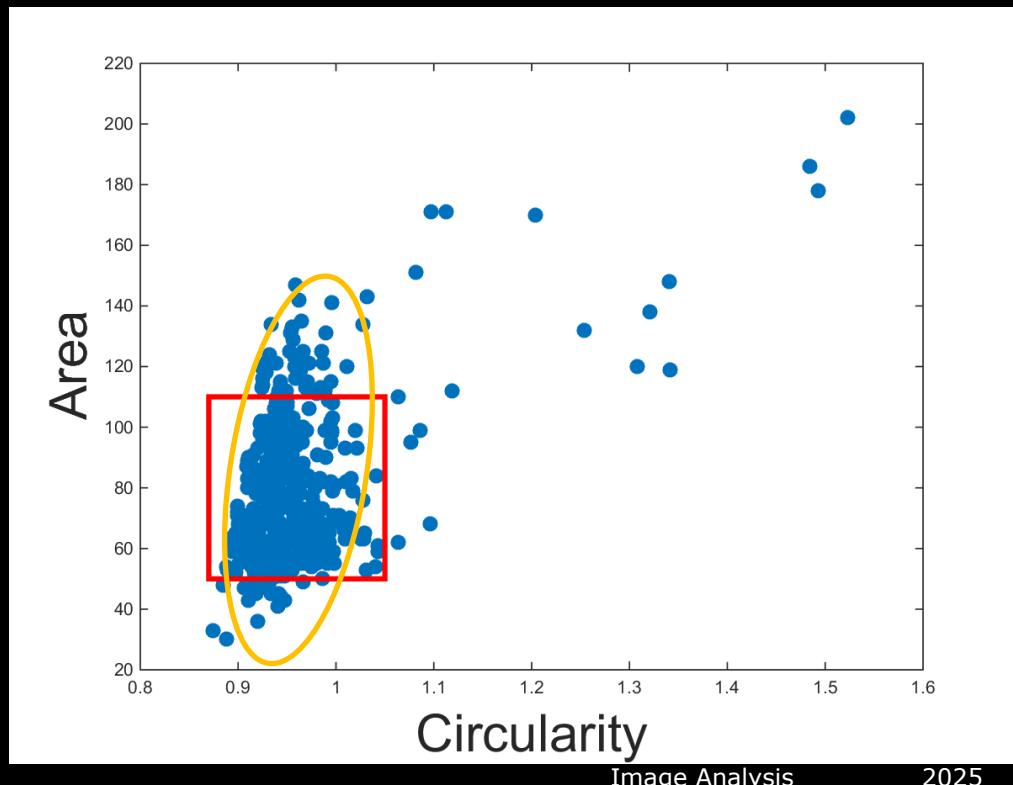
■ We are **not** interested in the true  
number of single nuclei

High true negatives 4%

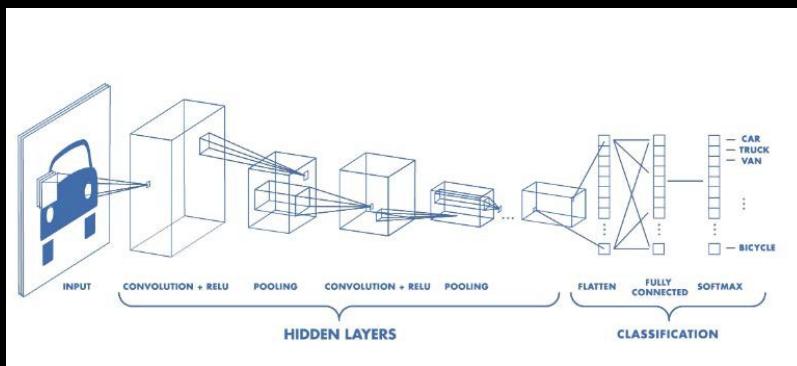
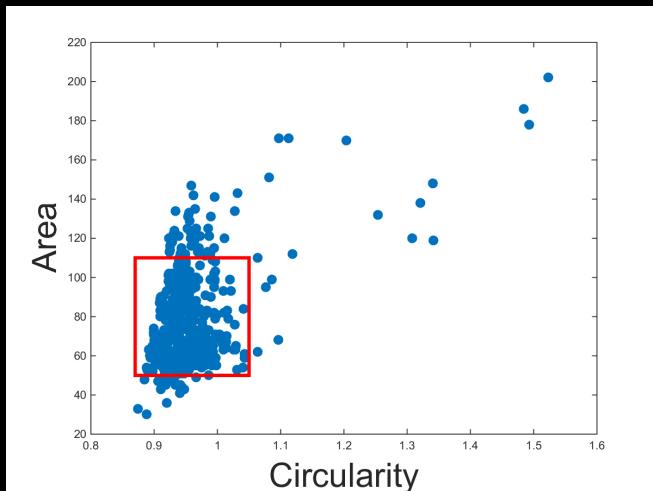
Low false negatives 7%

# Advanced classification

- Fitting more advanced functions to the samples
- Multivariate Gaussians
- Mahalanobis distances

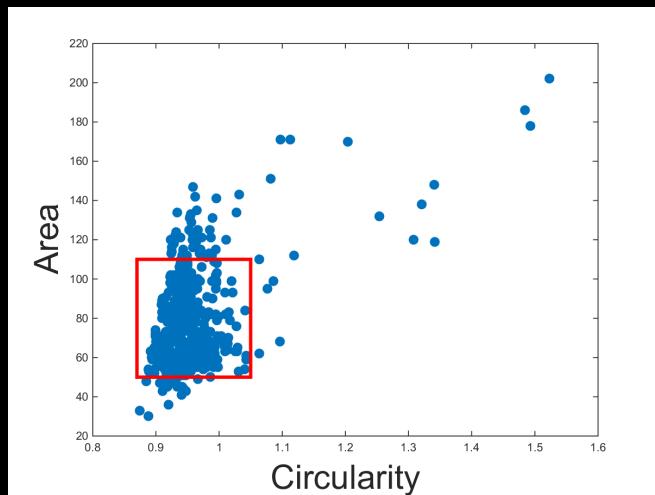


# Feature Engineering vs. Deep learning



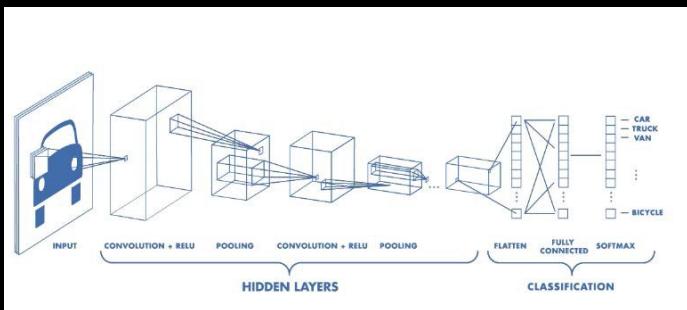
- Until around 5-7 years ago **feature engineering** was the way to go
- Now **deep learning beats** everything
- However – feature engineering is still important

# Feature engineering



- Given a classification problem
  - Cars vs. Pedestrians
- Use background knowledge to select relevant features
  - Area
  - Shape
  - Appearance
  - ...
- Use multivariate statistics to classify
- Depending on the selected features

# Deep learning

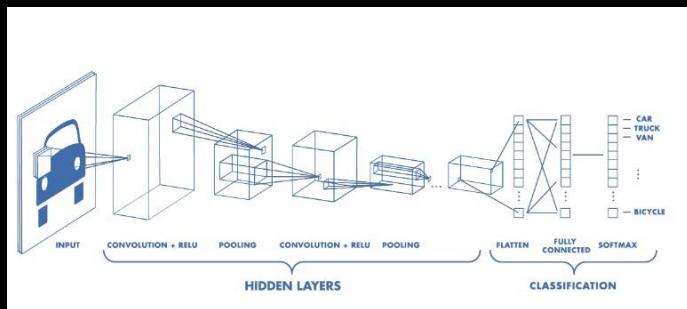


- You start with a dummy classifier
- Feed it with lots and lots of data with given labels
- The network learns the optimal features
- Layer/network engineering

# Feature Engineering vs. Deep learning

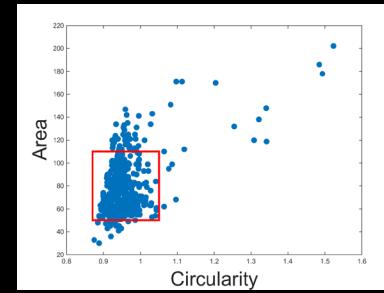
## Deep Learning

- When you have lot of annotated data
- Where it is not clear what features work

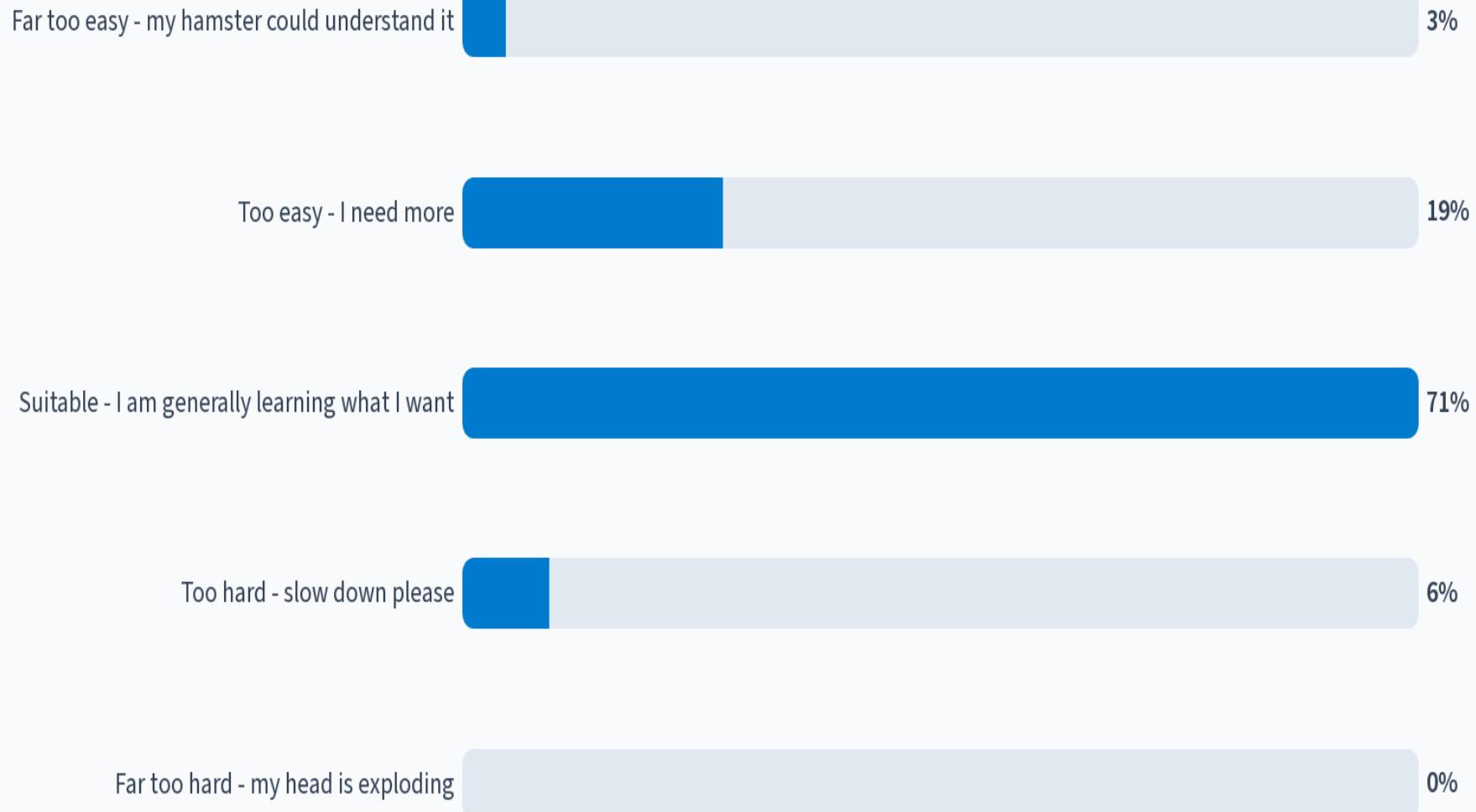


## Manual features

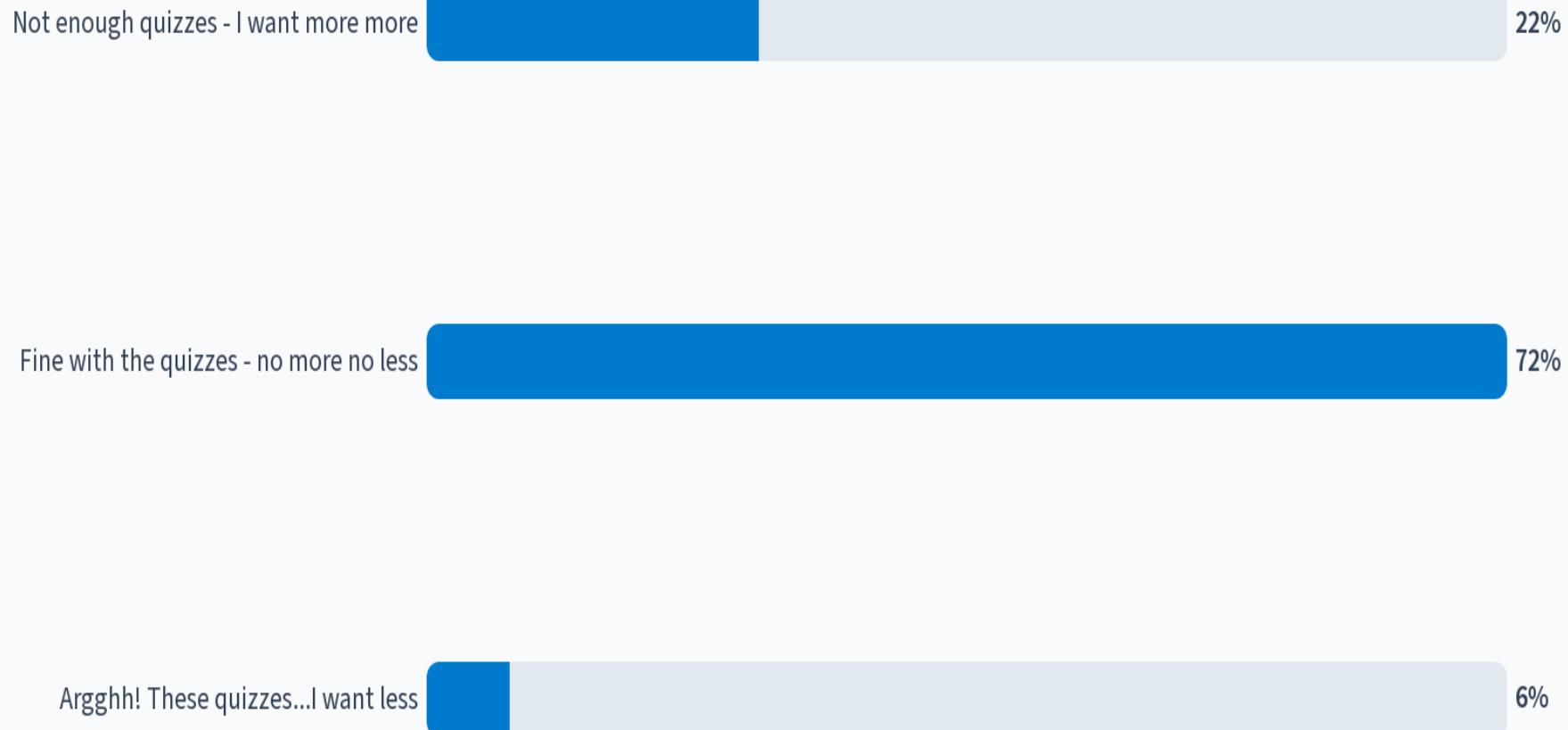
- When you have limited data
- When it is rather obvious what features can discriminate



## The level of the lecture



## The quizzes



# Next week

- Pixel classification
- Advanced classification



# Image Analysis

Tim B. Dyrby

Rasmus R. Paulsen

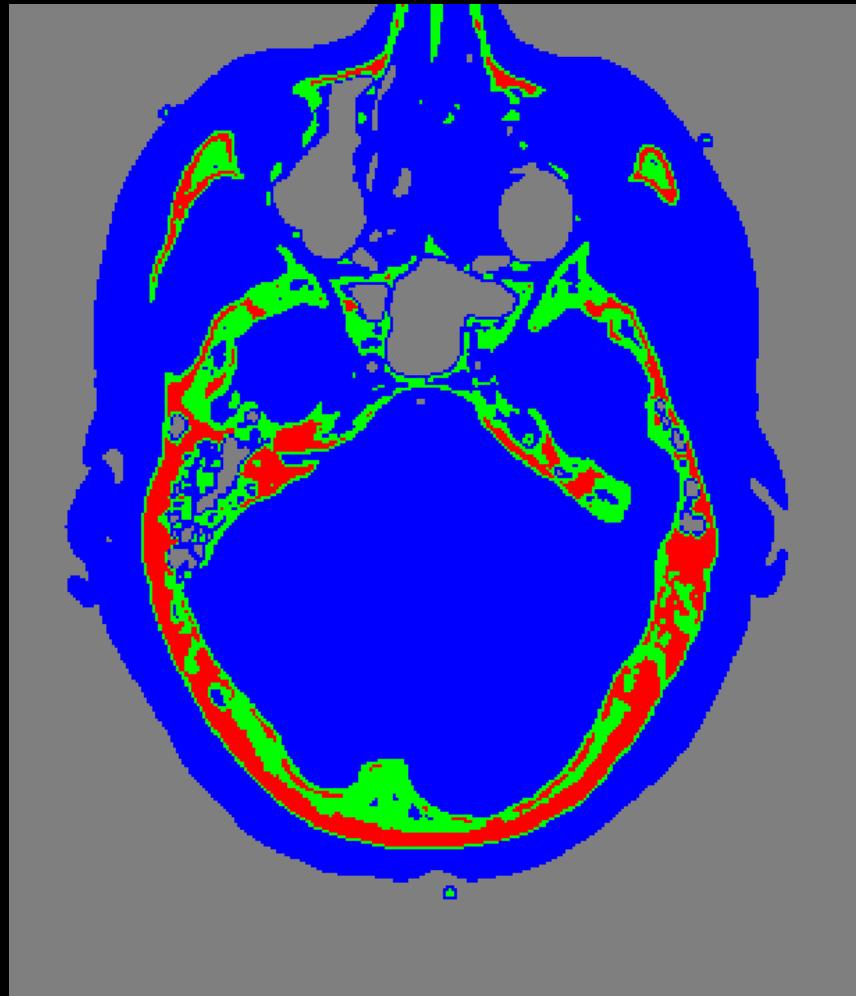
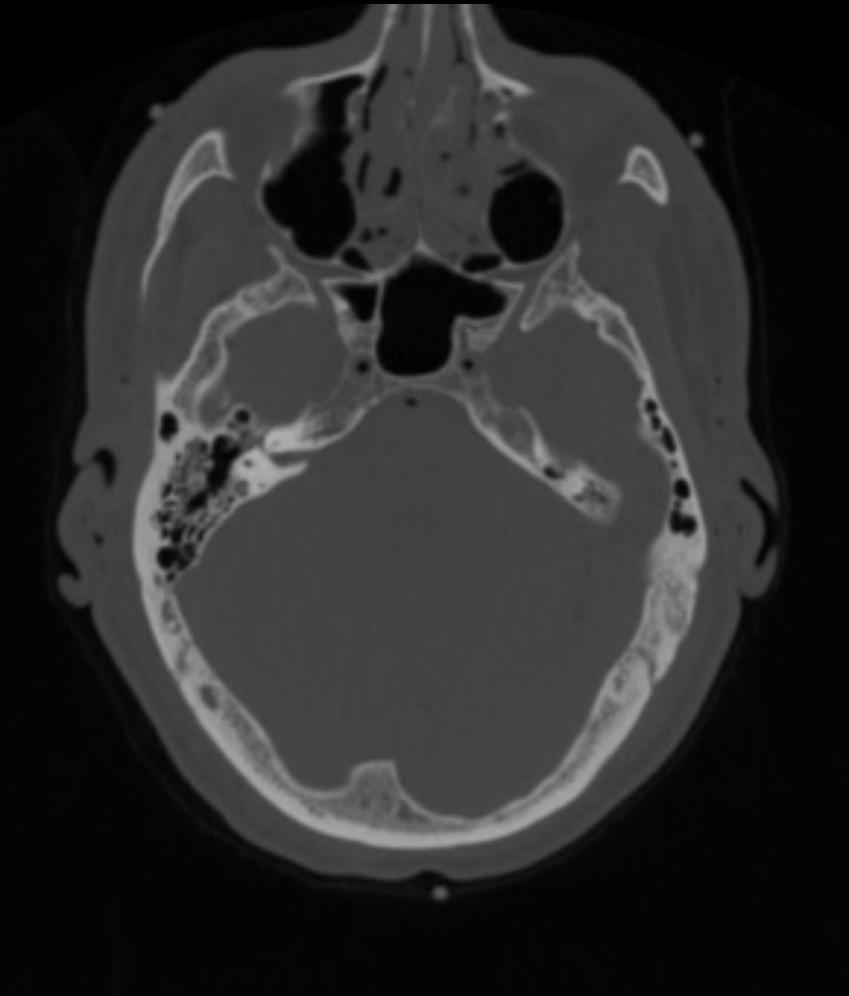
DTU Compute

[tbdy@dtu.dk](mailto:tbdy@dtu.dk)

<http://www.compute.dtu.dk/courses/02502>



# Lecture 6 – Pixel Classification and advanced segmentation





# What can you do after today?

- Describe the concept of pixel classification
- Compute the pixel value ranges in a minimum distance classifier
- Implement and use a minimum distance classifier
- Approximate a pixel value histogram using a Gaussian distribution
- Implement and use a parametric classifier
- Decide if a minimum distance or a parametric classifier is appropriate based on the training data
- Explain the concept of Bayesian classification
- Implement and use the linear discriminant analysis (LDA) classifier
- Decide where to place a decision boundary
- Understand the use of linear vs non-line hyperplanes for segmentation



Go to [www.menti.com](http://www.menti.com) and use the code 59 42 89 7

## Quiz 0: What is advanced segmentation?

0	0	0	0
To separate colours?	Use methods that mimics the human brain?	It just some vectors pointing in a space?	To draw linear and non-linear hyper plans in space

# Classification

- Take a measurement and put it into a class

Measurement



Wheels: 2

HP: 50

Weight: 200

Classifier

Classes

- Bike
- Truck
- Car
- Motorbike
- Train
- Bus

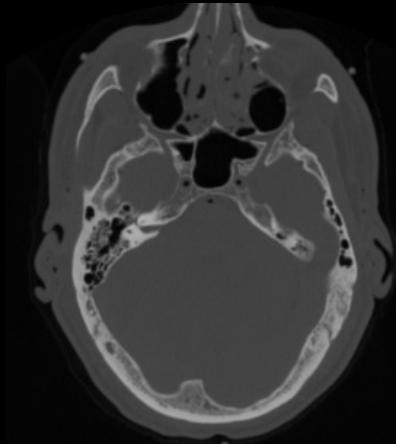


# General Classification

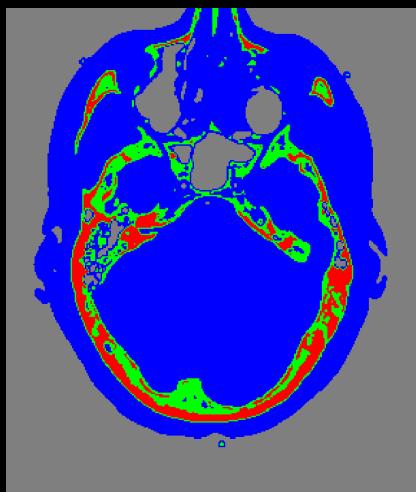
- Multi-dimensional measurement
- Pre-defined classes
  - Can also be found automatically – can be very difficult!

# Pixel Classification

CT scan of human head



Pixel wise segmentation



Four Class labels  
Background  
Soft-Tissue  
Trabecular Bone  
Hard Bone

- Classify each pixel
  - Independent of neighbours
- Also called labelling
  - Put a label on each pixel
- We look at the pixel value and assign them a label
- Labels already defined



# Quiz 1: Two class pixel classification?

Background and object

- A) Median filter
- B) Threshold
- C) Brightness
- D) Morphological Erosion
- E) BLOB analysis



# Pixel Classification – formal definition

Pixel value (the measurement)  $v \in R$

k classes

$$\mathcal{C} = c_1, \dots, c_k$$

Classification rule

$$c: R \rightarrow \{c_1, \dots, c_k\}$$



# Pixel Classification – example

Pixel value

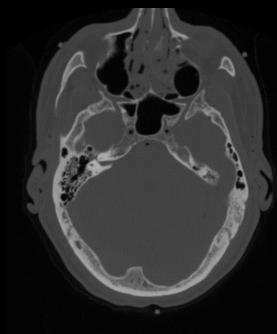
$$\nu \in [0,255]$$

Set of 4 classes

$$C = \{\text{background, soft-tissue, trabeculae, bone}\}$$

Classification rule

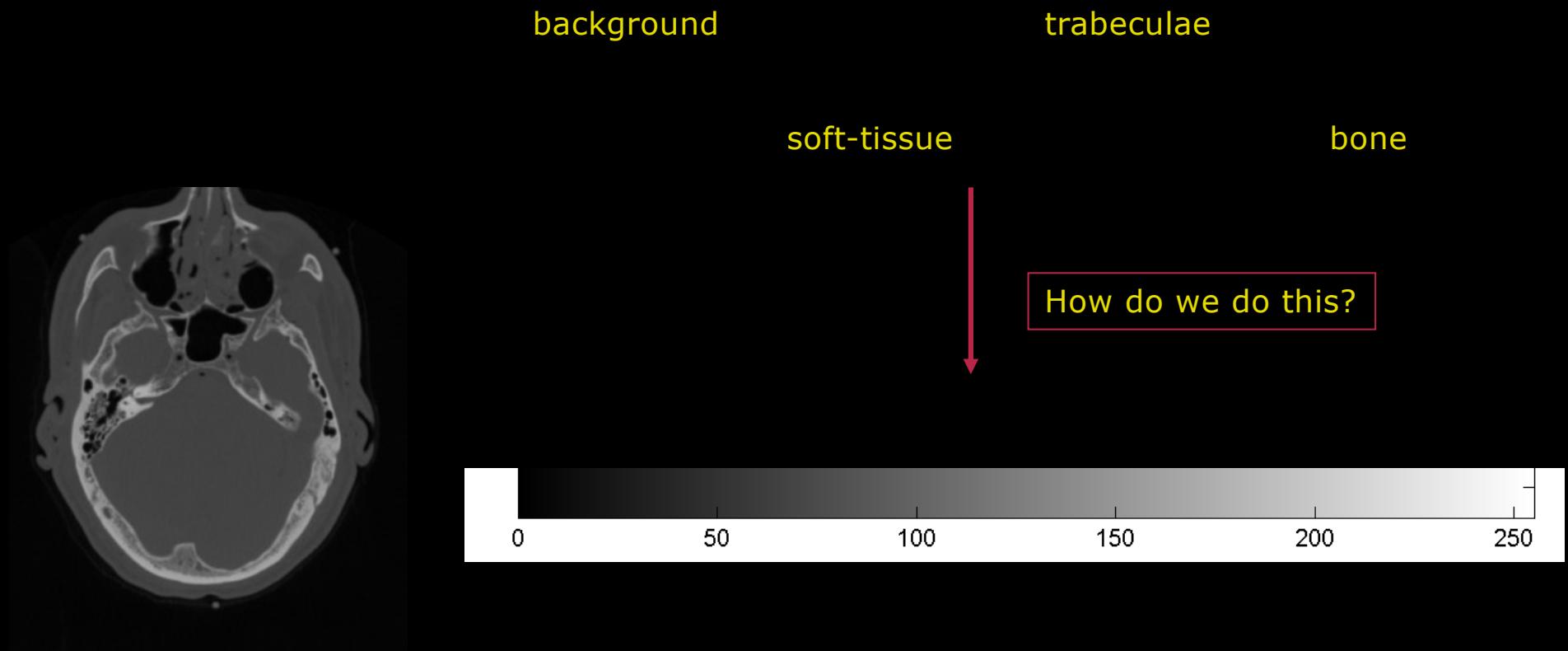
$$c: \nu \rightarrow \{\text{background, soft-tissue, trabeculae, bone}\}$$



How do we construct a classification rule?

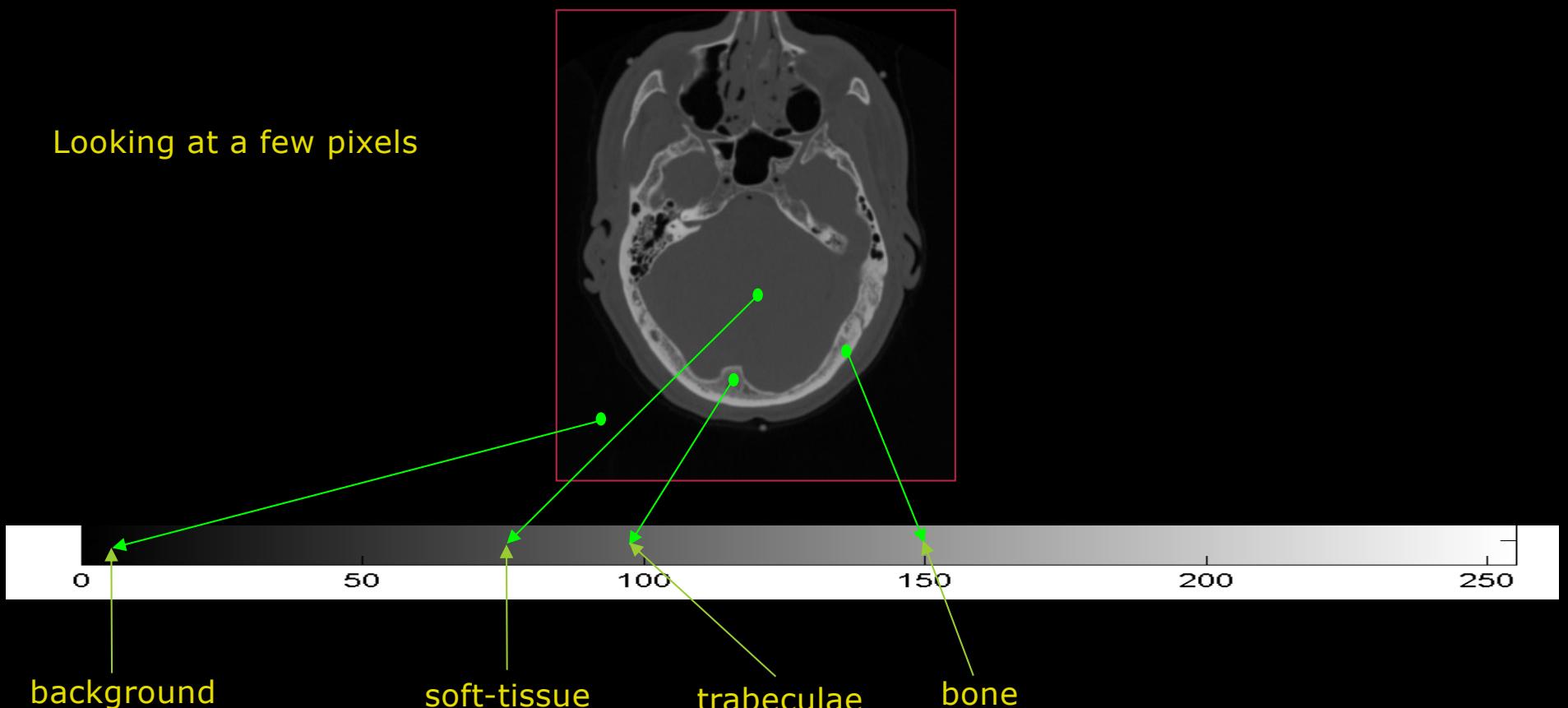
# Pixel classification rule

$c: v \rightarrow \{\text{background, soft-tissue, trabeculae, bone}\}$



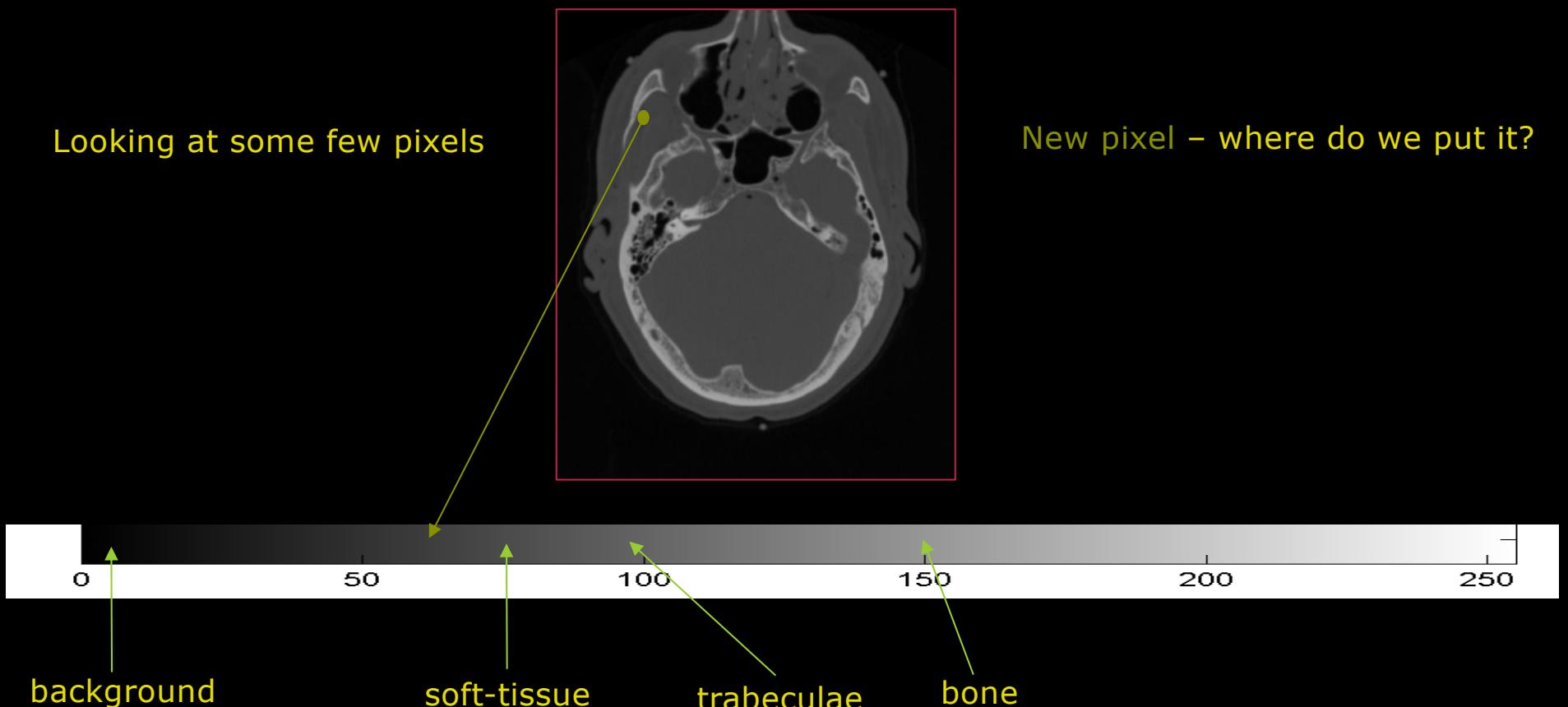
# Pixel classification rule – manual inspection

$c: v \rightarrow \{\text{background}, \text{soft-tissue}, \text{trabeculae}, \text{bone}\}$



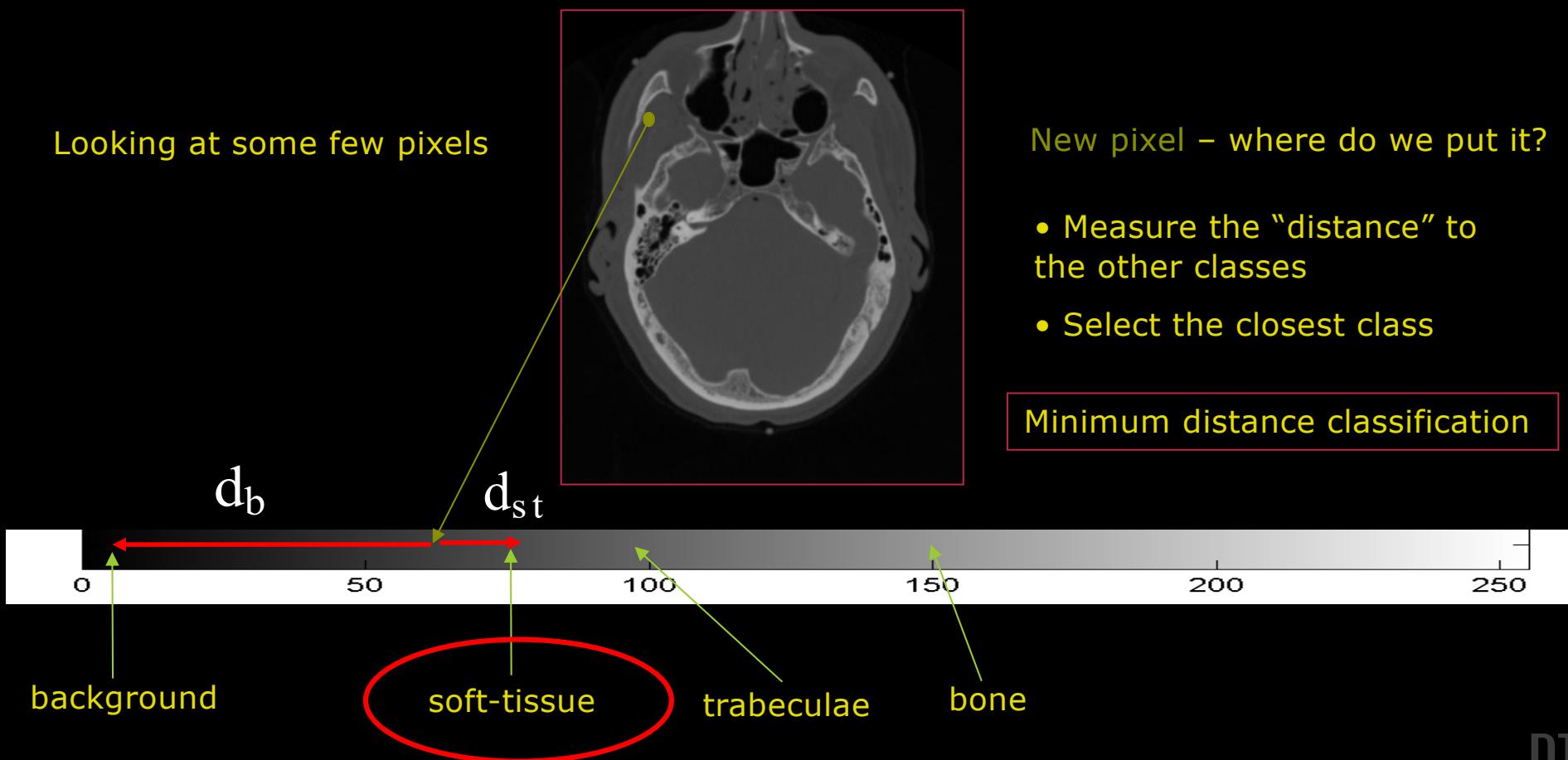
# Pixel classification rule – manual inspection

$c: v \rightarrow \{\text{background, soft-tissue, trabeculae, bone}\}$



# Pixel classification rule – manual inspection

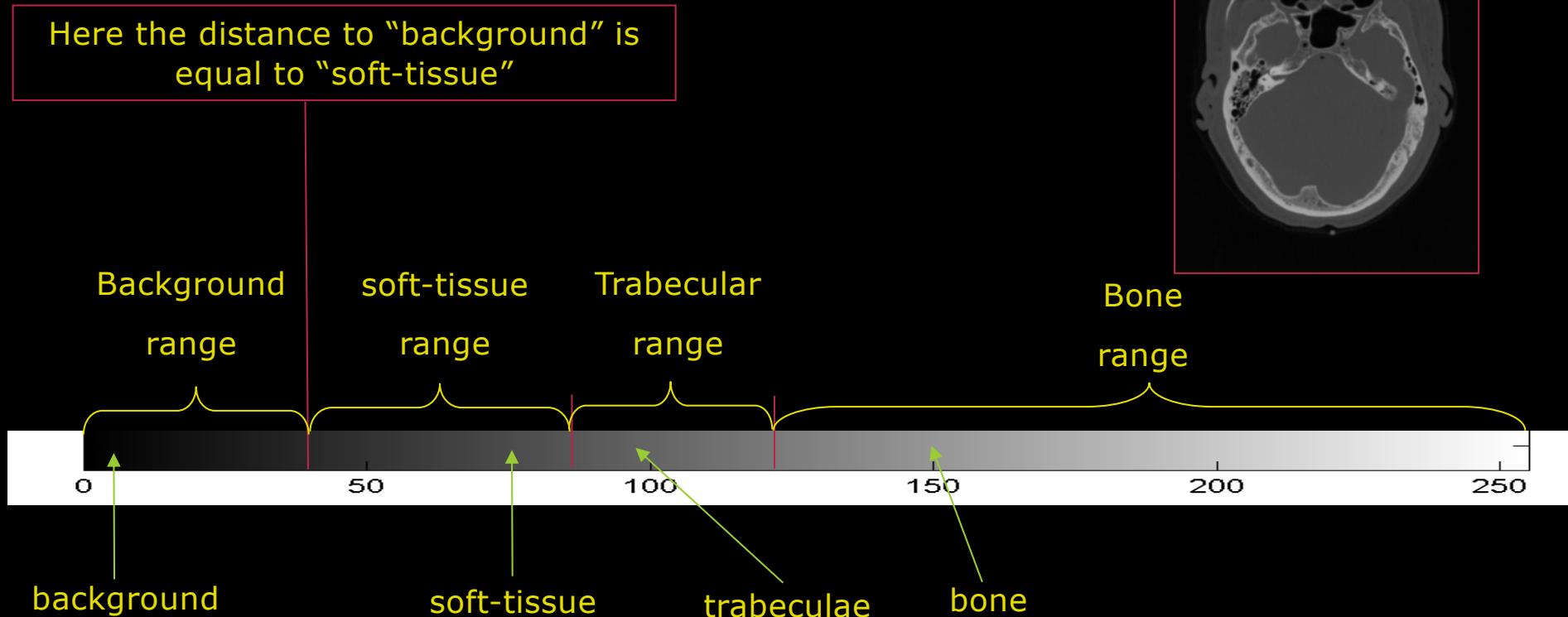
$c: v \rightarrow \{\text{background}, \text{soft-tissue}, \text{trabeculae}, \text{bone}\}$



# Pixel classification rule

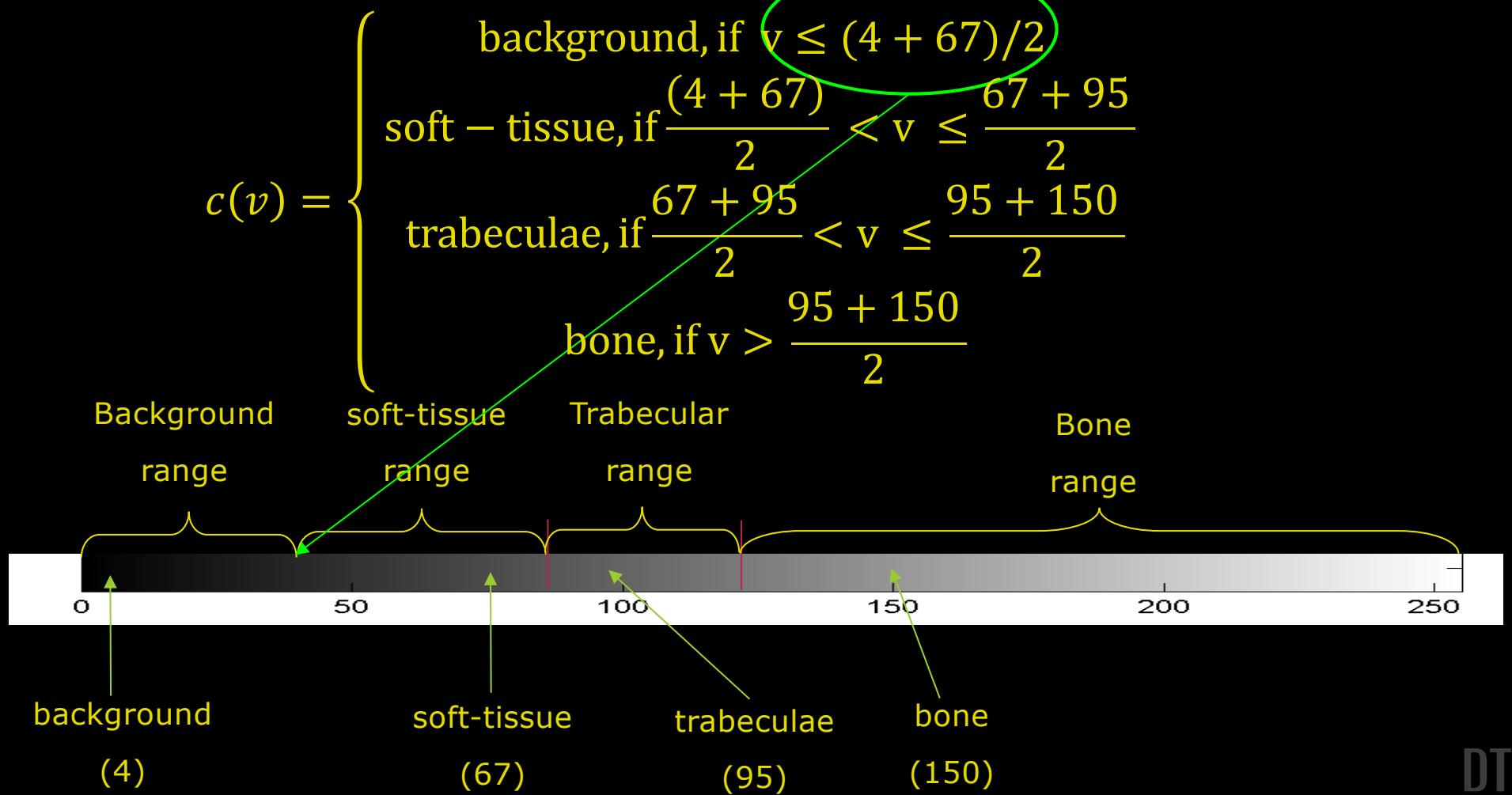
## Minimum Distance Classification

The possible pixel values are divided into ranges



# Pixel classification rule

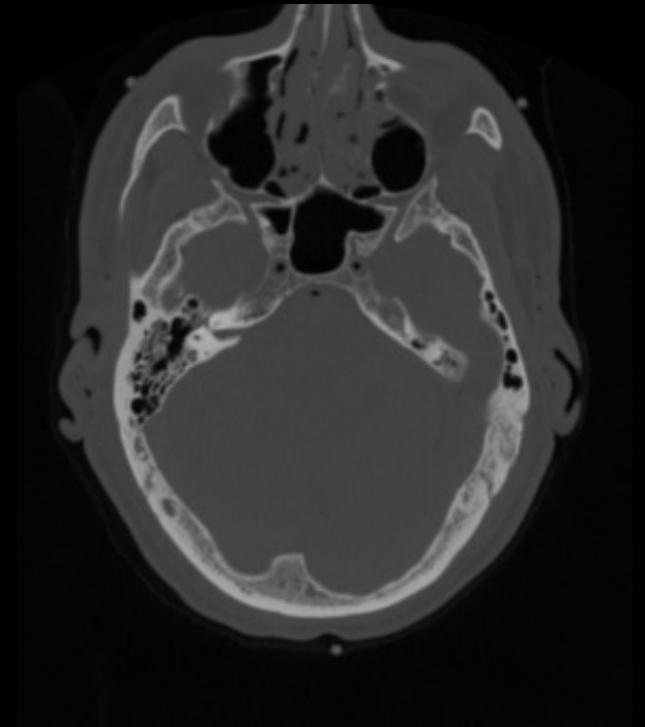
## Minimum Distance Classification



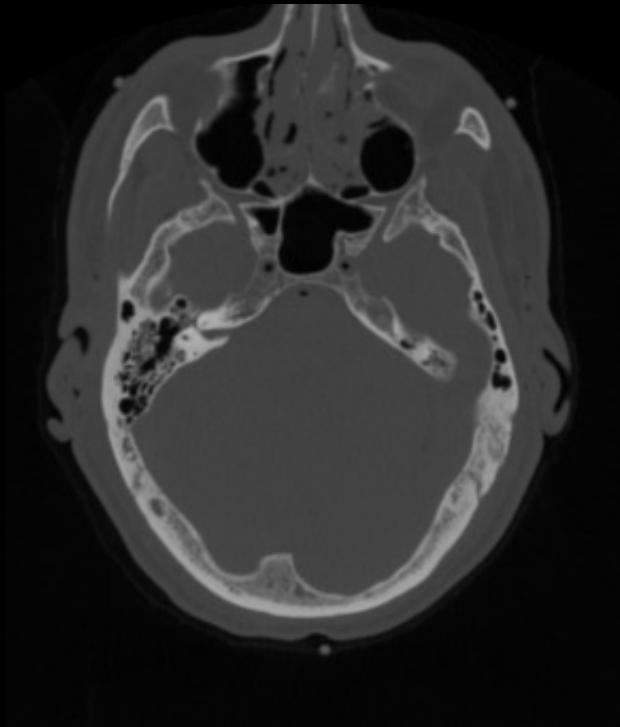
# Pixel classification rule

- For all pixel in the image do

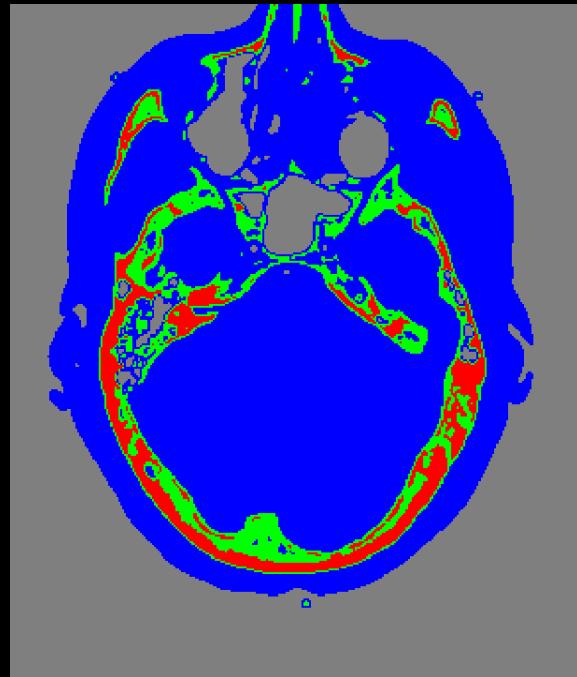
$$c(v) = \begin{cases} \text{background, if } v \leq (4 + 67)/2 \\ \text{soft - tissue, if } \frac{(4 + 67)}{2} < v \leq \frac{67 + 95}{2} \\ \text{trabeculae, if } \frac{67 + 95}{2} < v \leq \frac{95 + 150}{2} \\ \text{bone, if } v > \frac{95 + 150}{2} \end{cases}$$



# Pixel Classification example

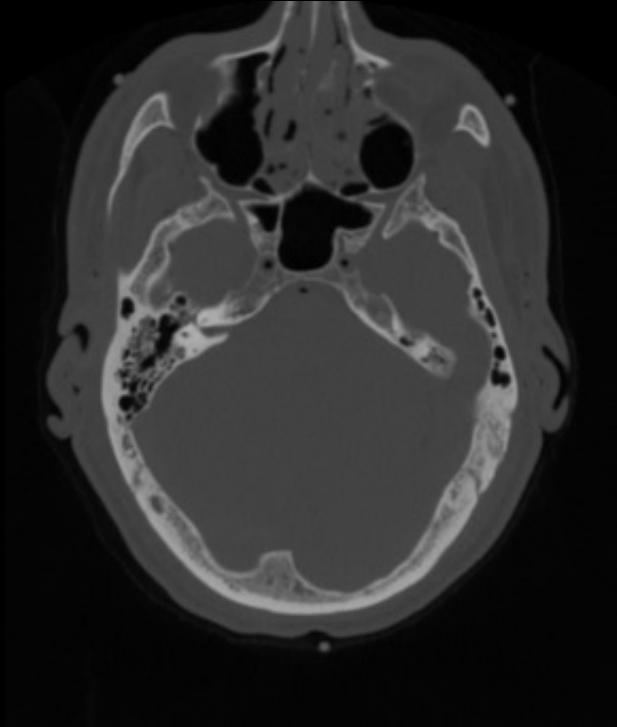


CT scan of human head



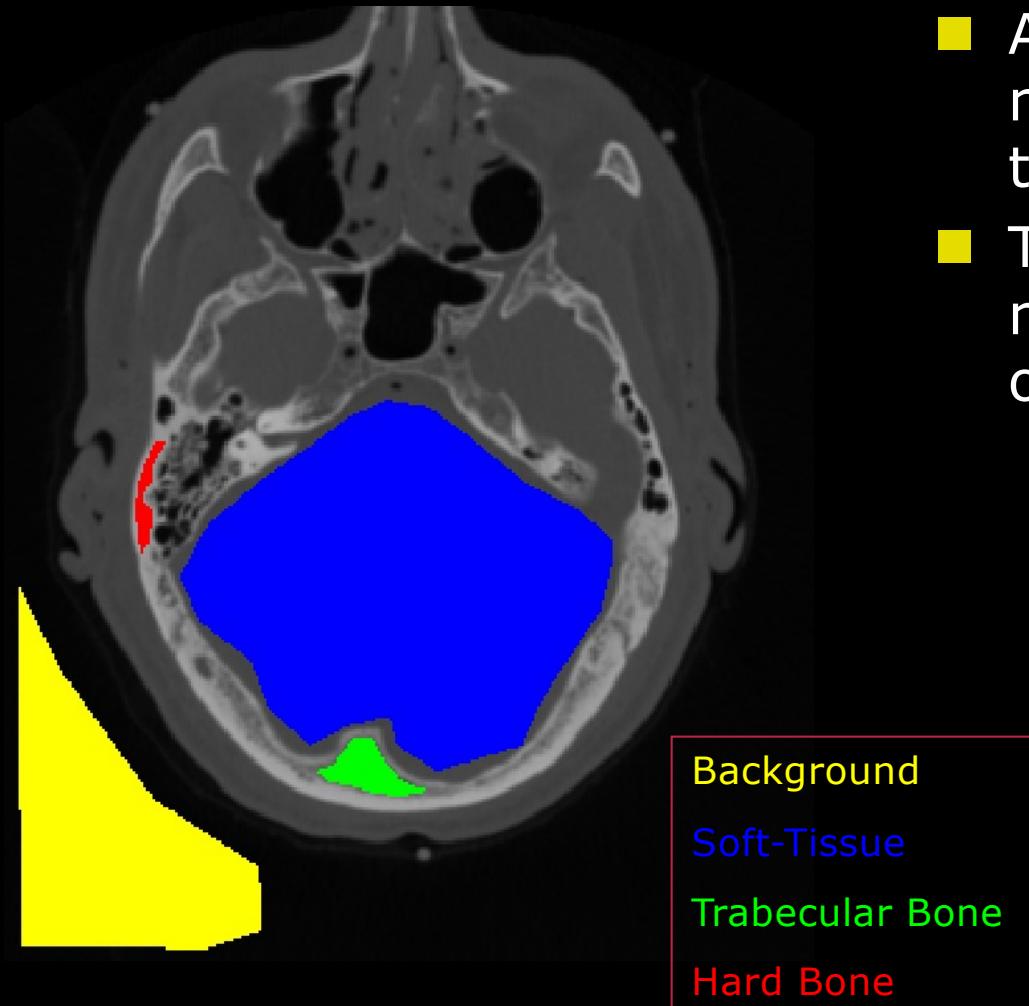
Background  
Soft-Tissue  
Trabecular Bone  
Hard Bone

# Better range selection



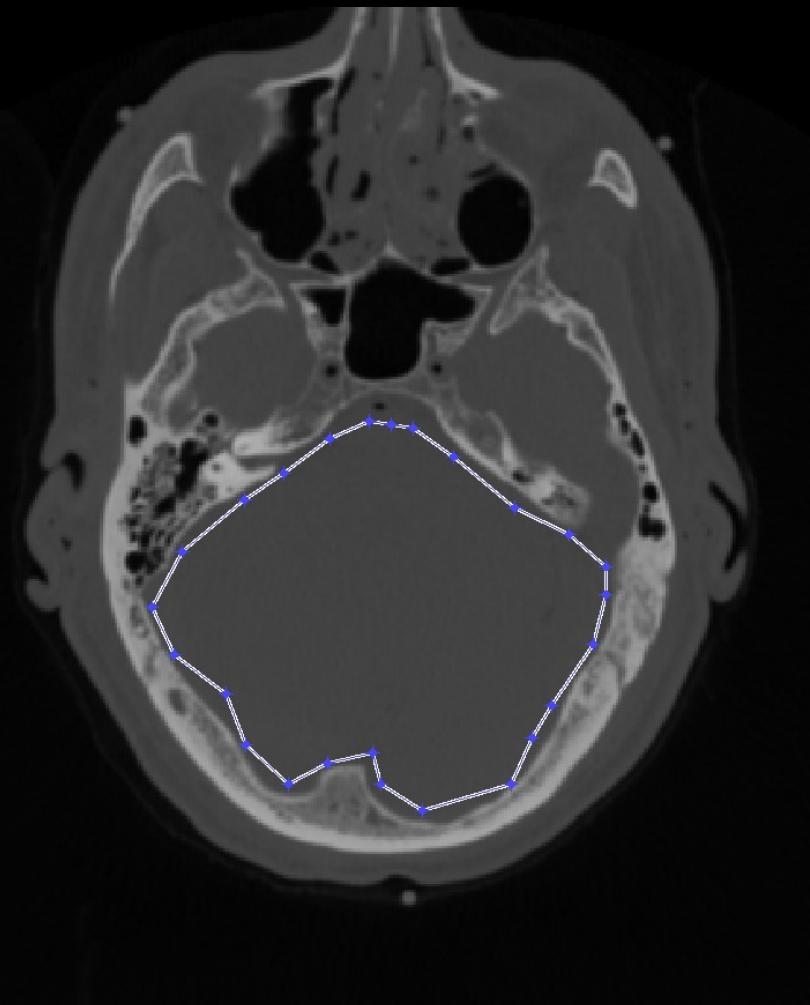
- Guessing range values is not a good idea
- Better to use “training data”
- Start by selecting representative regions from an image
- *Annotation*
  - To mark points, regions, lines or other significant structures

# Classifier training - annotation



- An “expert” is asked how many different tissue types that are possible
- Then the expert is asked to mark representative regions of the selected tissue types

# Classifier training – region selection

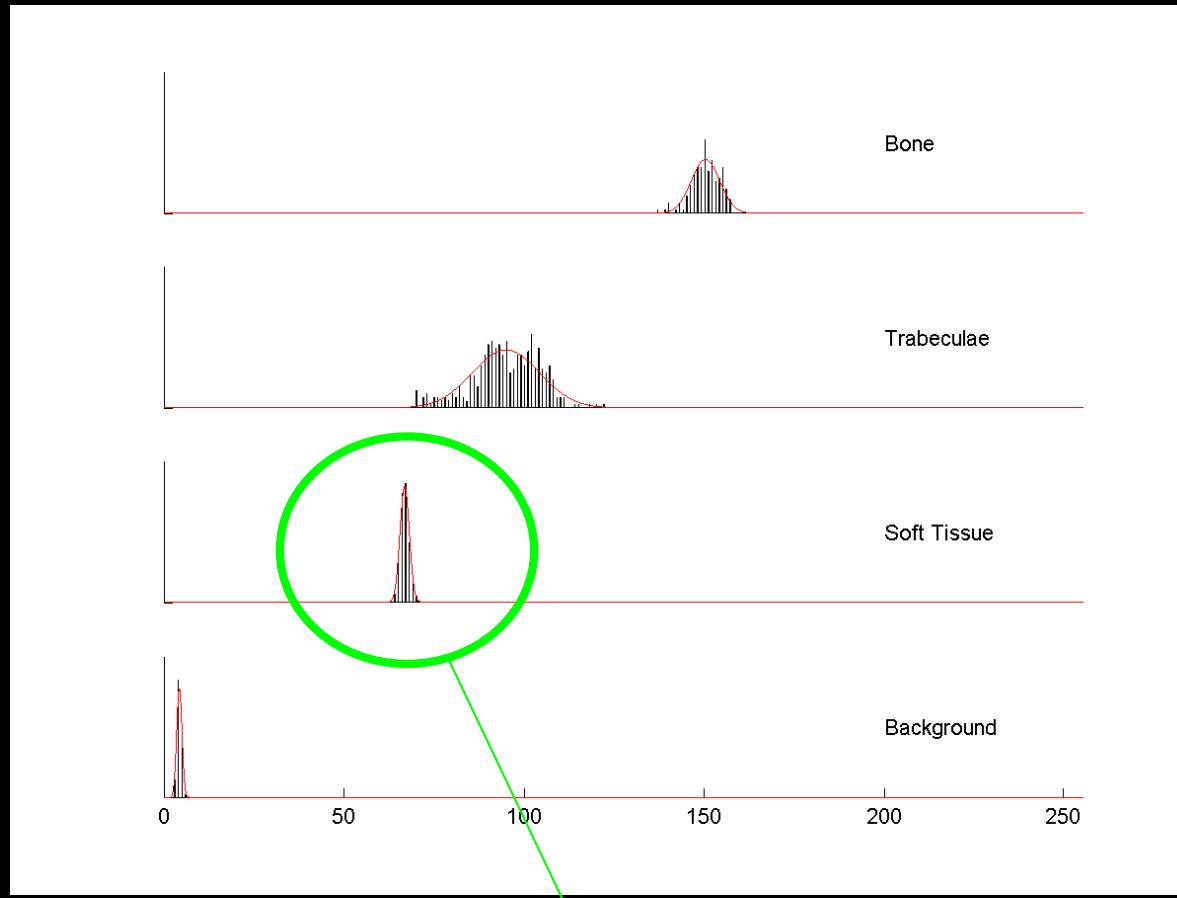


- Many tools exist
- Python module `roipoly`
  - Select closed regions using a piecewise polygon

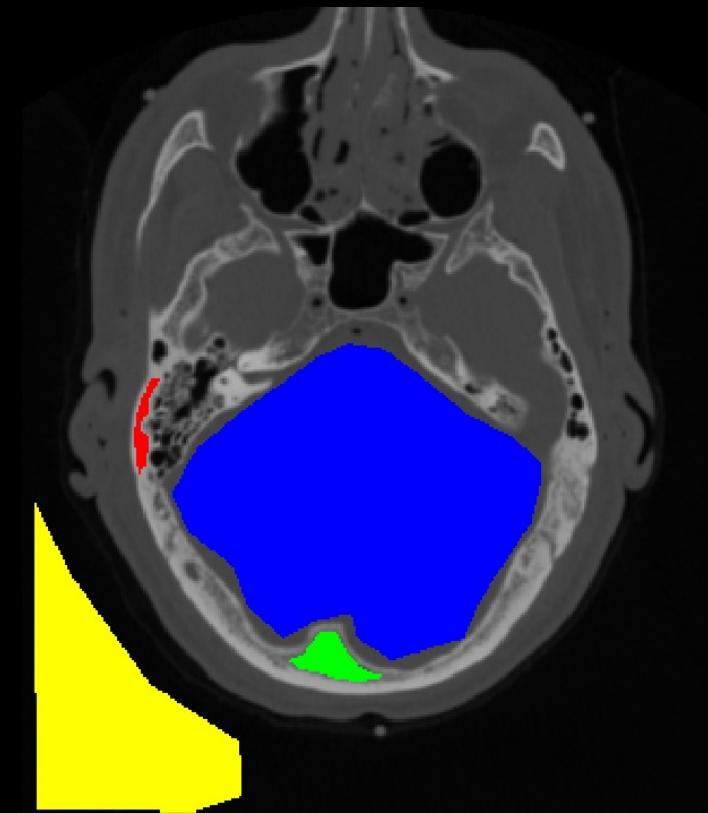
Training is only done once!

Optimally, the training can be used on many pictures that contains the same tissue types

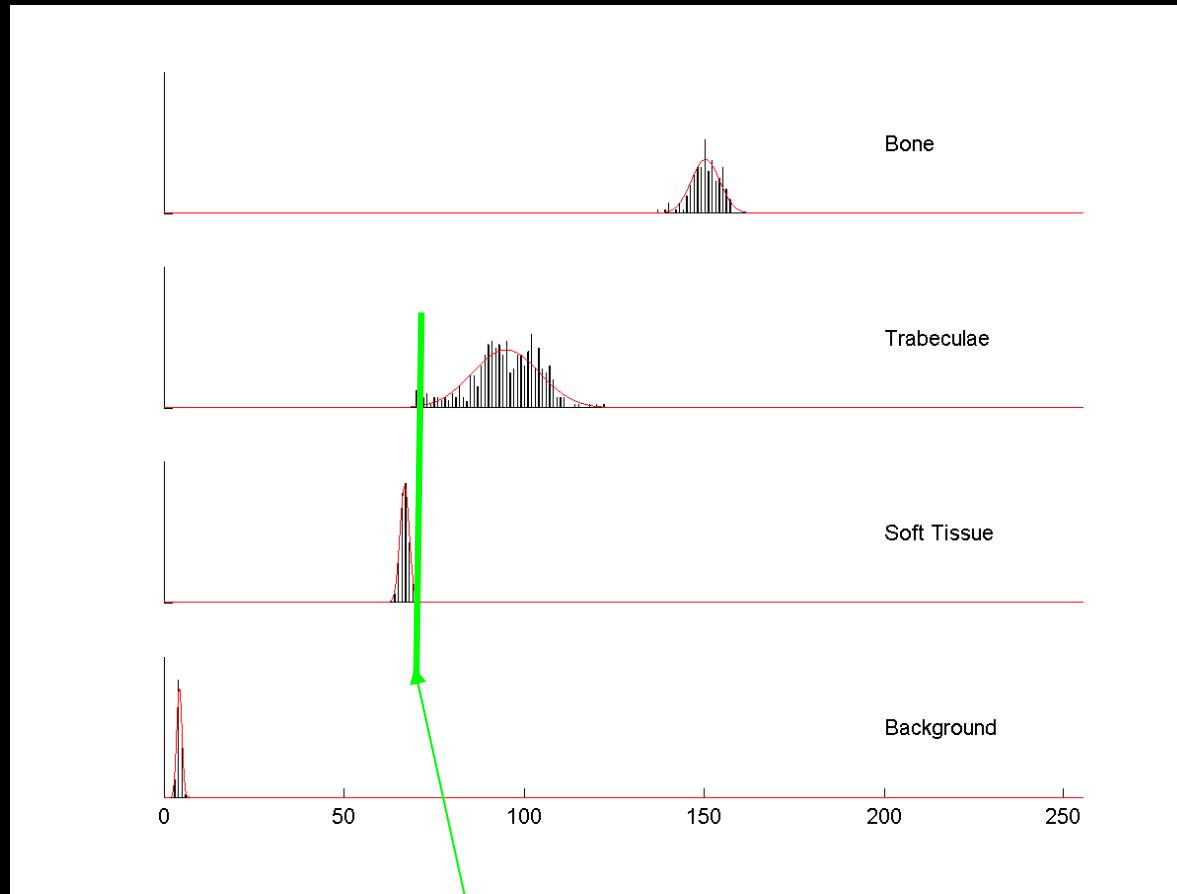
# Initial analysis - histograms



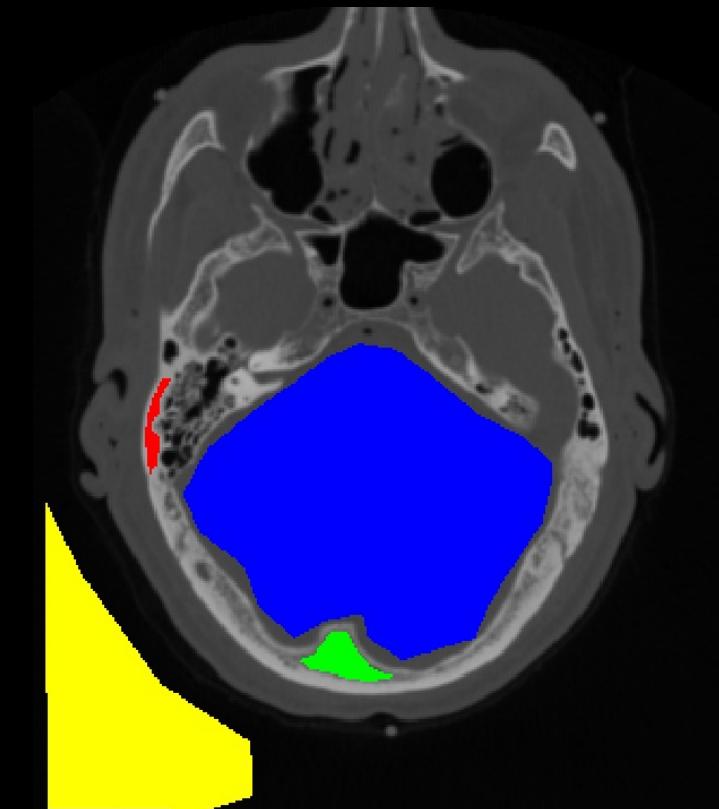
Gaussian



# Initial analysis - histograms

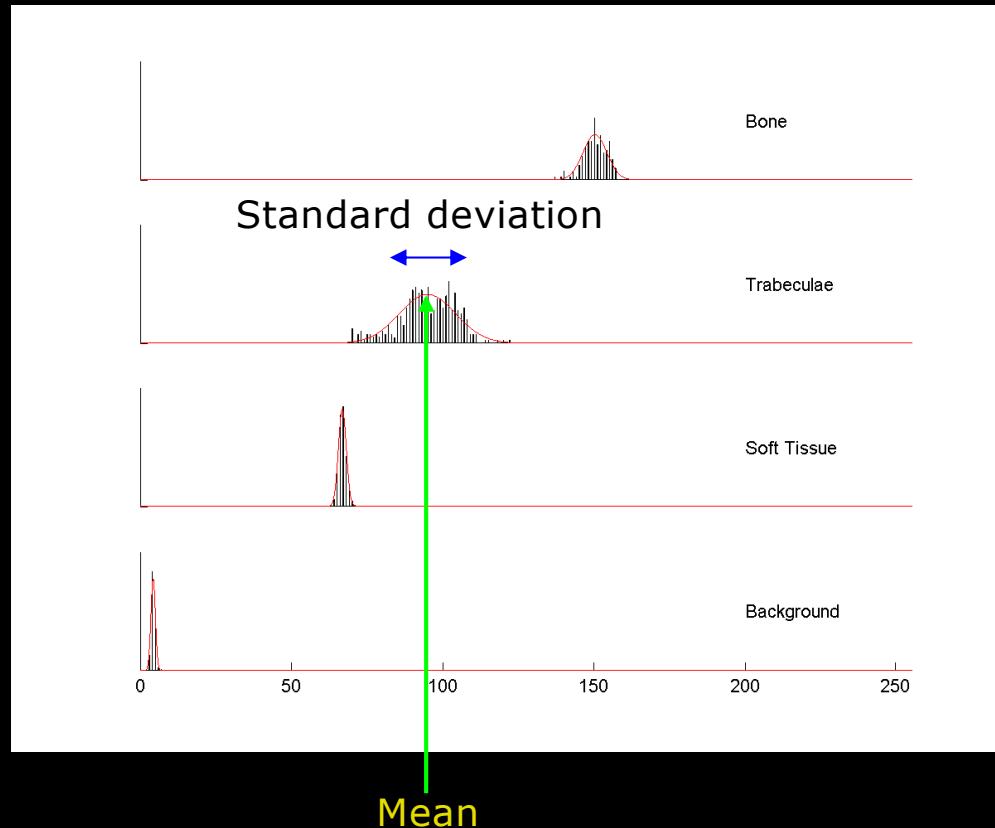


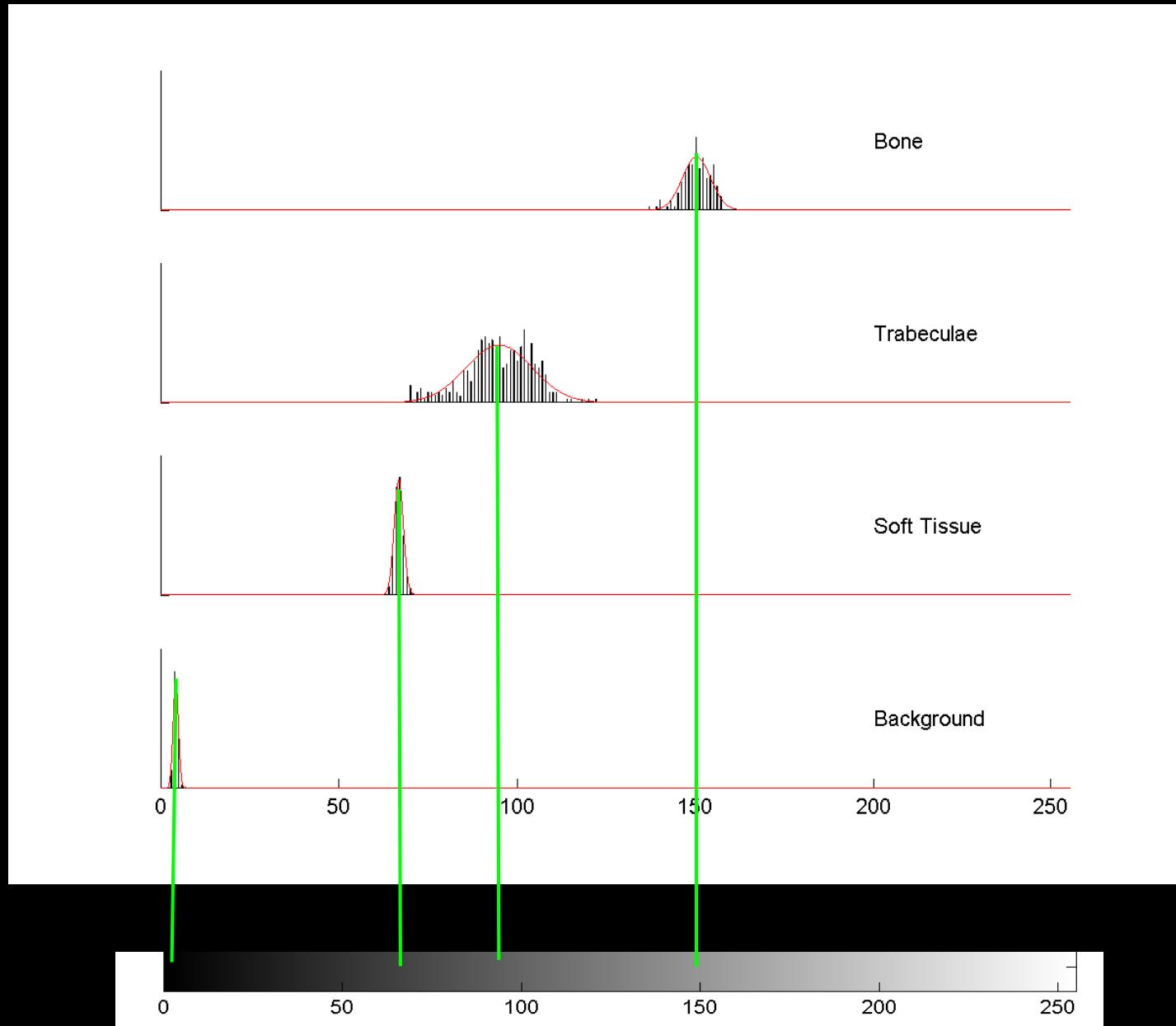
Class separation



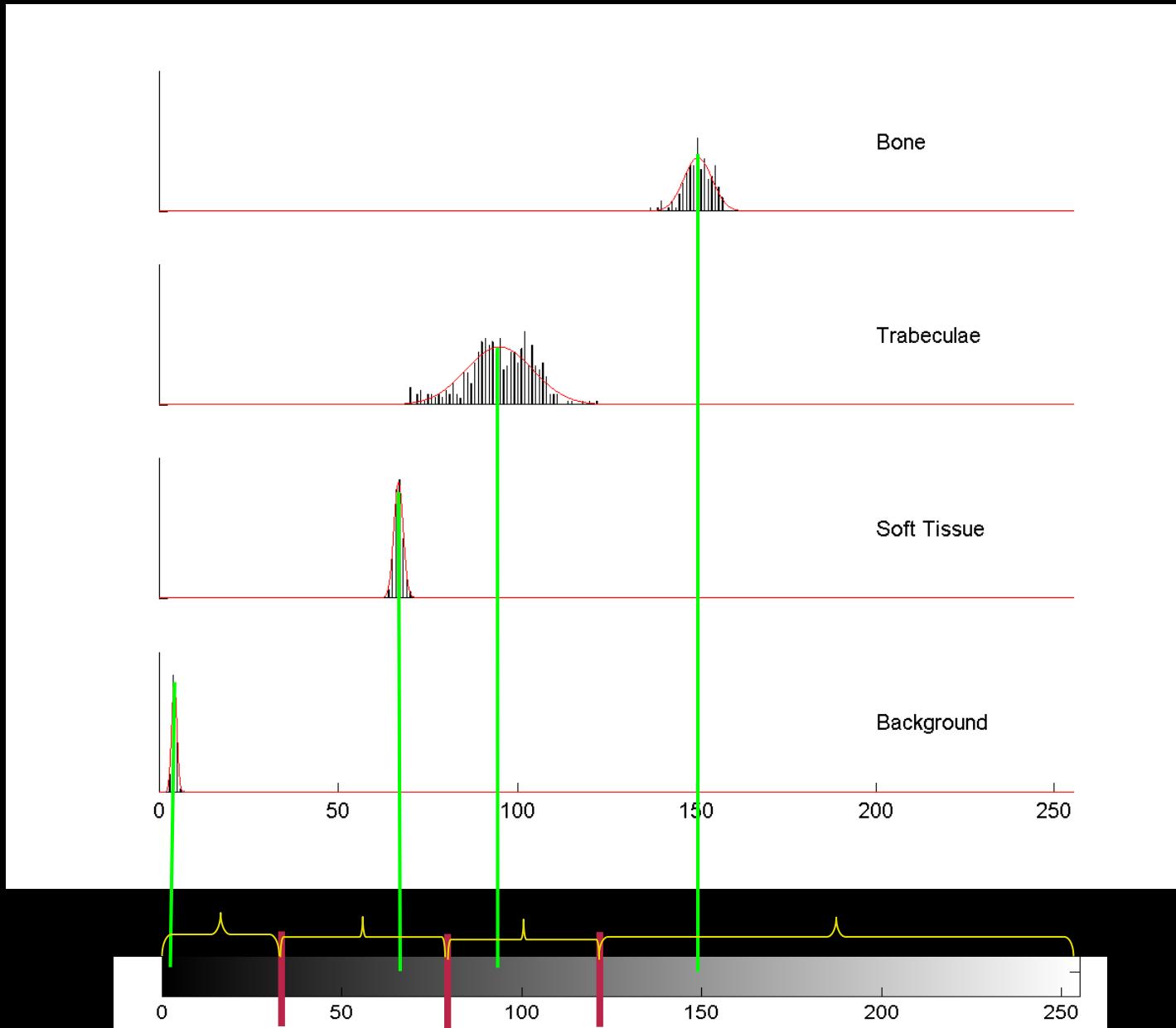
# Simple pixel statistics

- Calculate the mean and the standard deviation of each class





## Minimum distance classification



Any objections?

The pixel value ranges are not always in good correspondence with the histograms?

# Quiz 2:

## Minimum distance classification

- A) Background
- B) Soft tissue
- C)** Fat
- D) Bone
- E) None of the above

Solution:

$$\text{Green: } (6+4+9+5)/4=6$$

$$\text{Blue: } (132+130+134+133)/4= 132,25$$

$$\text{Yellow: } (178+175+176+174)/4=175,75$$

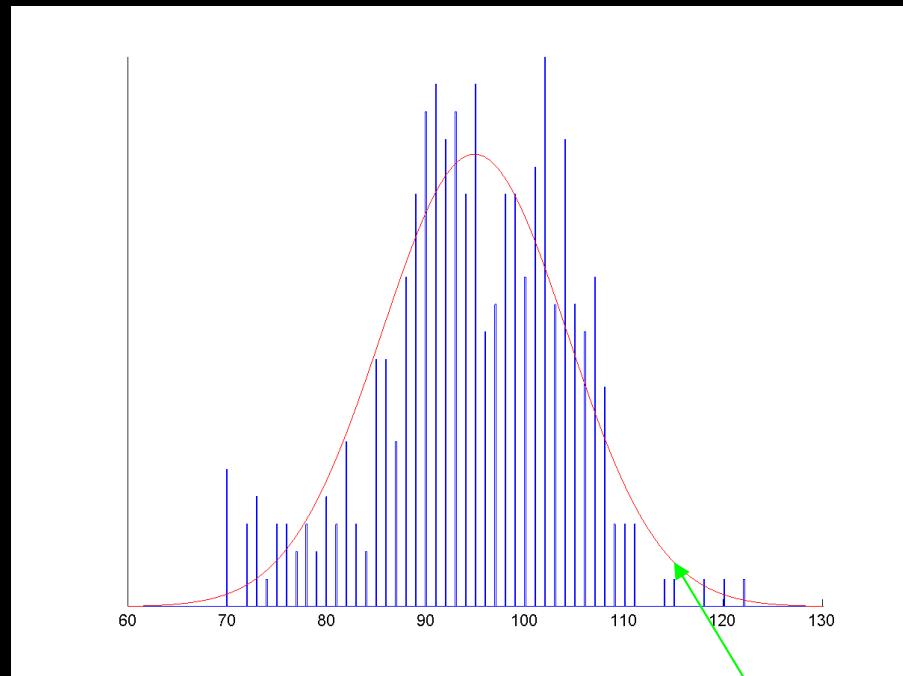
$$\text{Purple: } (222+220+219+221)/4=220$$

Blue: 158 is closer to 175,75 (yellow)= fat

To make a pixel classification an expert has selected representative regions in the image. They contain background (green), soft tissue (blue), fat (yellow), and bone (purple). The goal is to classify the pixel marked with a light blue circle. Using a minimum distance classifier it is classified as?

5	6	5	81	180	182	222	220
8	9	4	108	181	175	219	221
7	8	132	130	148	182	174	223
58	231	134	133	61	173	178	175
44	250	181	130	117	101	176	174
5	6	7	204	246	94	86	175
156	158	6	7	7	252	173	230
157	161	7	6	6	10	35	227

# Parametric classification



Trabecular bone

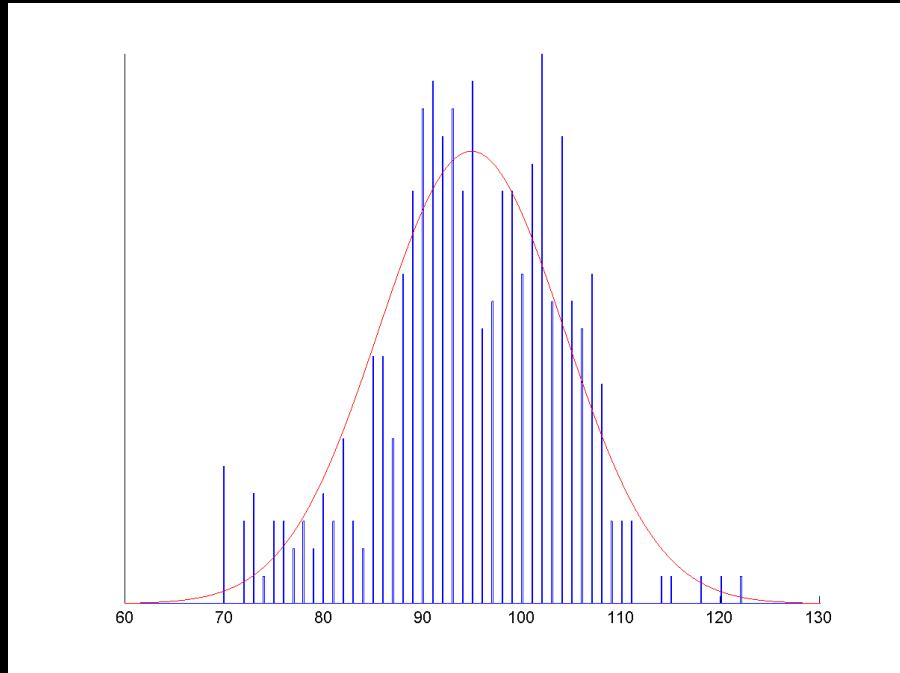
Only two values needed

- Describe the histogram using a few parameters
- Assume a “model” describing the signal values
- Model: Gaussian/Normal distribution

- The mean  $\mu$
- Standard deviation  $\sigma$
- $\mathcal{N}(\mu, \sigma)$

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x - \mu)^2}{2\sigma^2}\right)$$

# Parametric classification



Trabecular bone

Training pixel values  
(Belonging to one class)

Estimated mean

$$v_1, v_2, \dots, v_n ,$$

Estimated  
standard  
deviation

$$\hat{\mu} = \frac{1}{n} \sum_{i=1}^n v_i$$

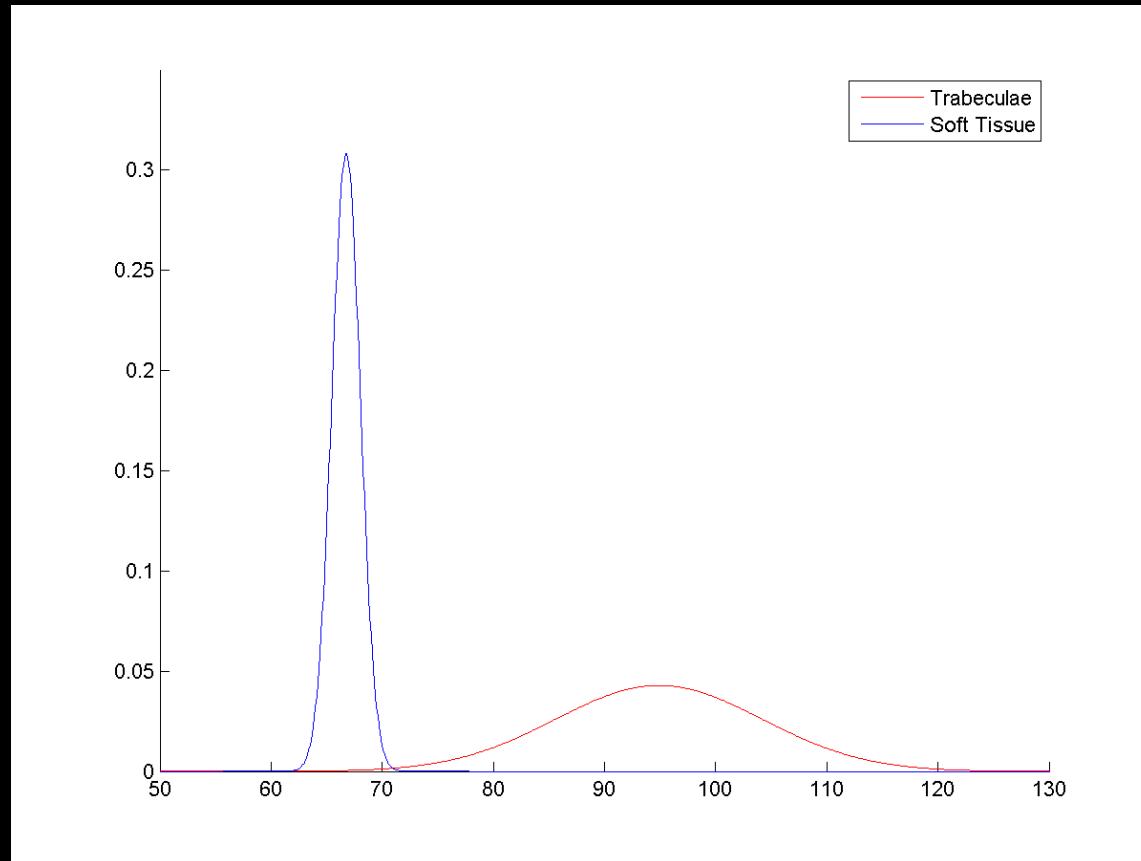
$$\hat{\sigma} = \sqrt{\frac{1}{n-1} \sum_{i=1}^n (v_i - \hat{\mu})^2}$$

The “signal model” is a Gaussian distribution

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x - \mu)^2}{2\sigma^2}\right)$$

# Parametric classification

- Fit a Gaussian to the training pixels for all classes



What do we see here?

What is the difference between the two classes?

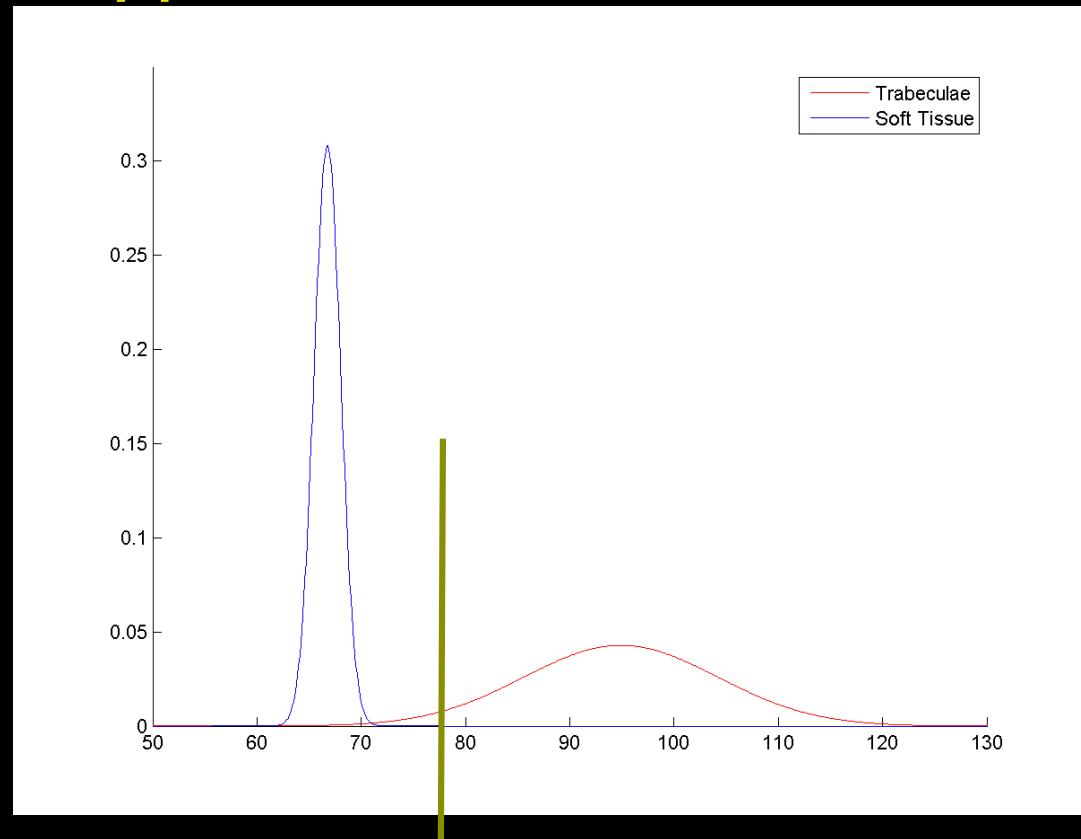
Trabeculae has much higher variation in the pixel values

# Quiz 3: Two tissue types – minimum distance

$v = 78$

Which tissue class?

- A) Trabeculae
- B) Soft-tissue



Solution: Minimum distance classifier

$v = 78$

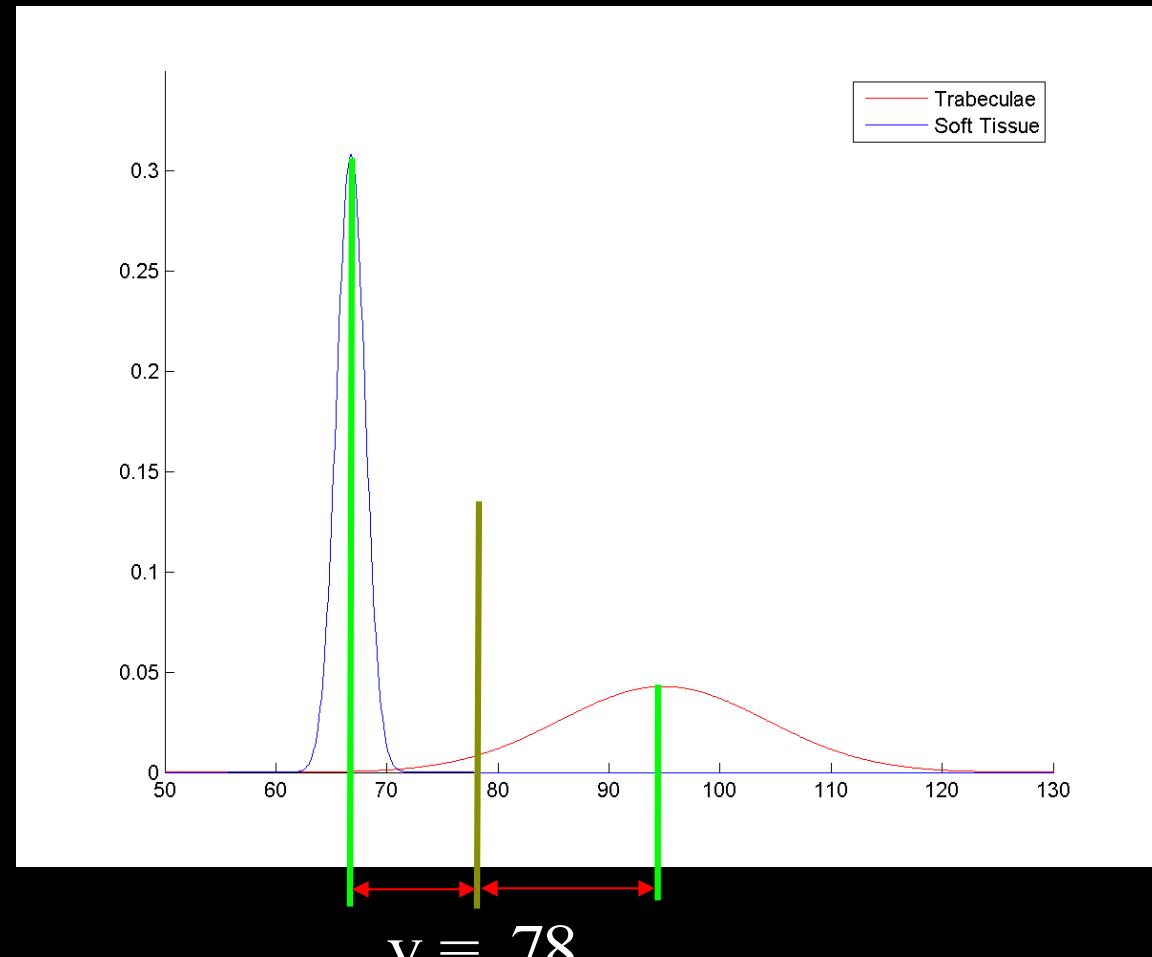
First we find the threshold,  $T$ :

B: mean(Soft Tissue)=68 and A: mean(Trabeculae)= 95

$$T = (95+68)/2 = 81,5$$

Then we classify/segment  $v=78$ : A if  $v>81,5$  or B if  $v<81,5$

# Parametric classification



- New pixel with value 78
  - Is it soft-tissue or trabecular bone?
- Minimum distance classifier?
  - Soft-tissue
- Is that fair?
  - Soft-tissue Gaussian says “Extremely low probability that this pixel is soft-tissue”

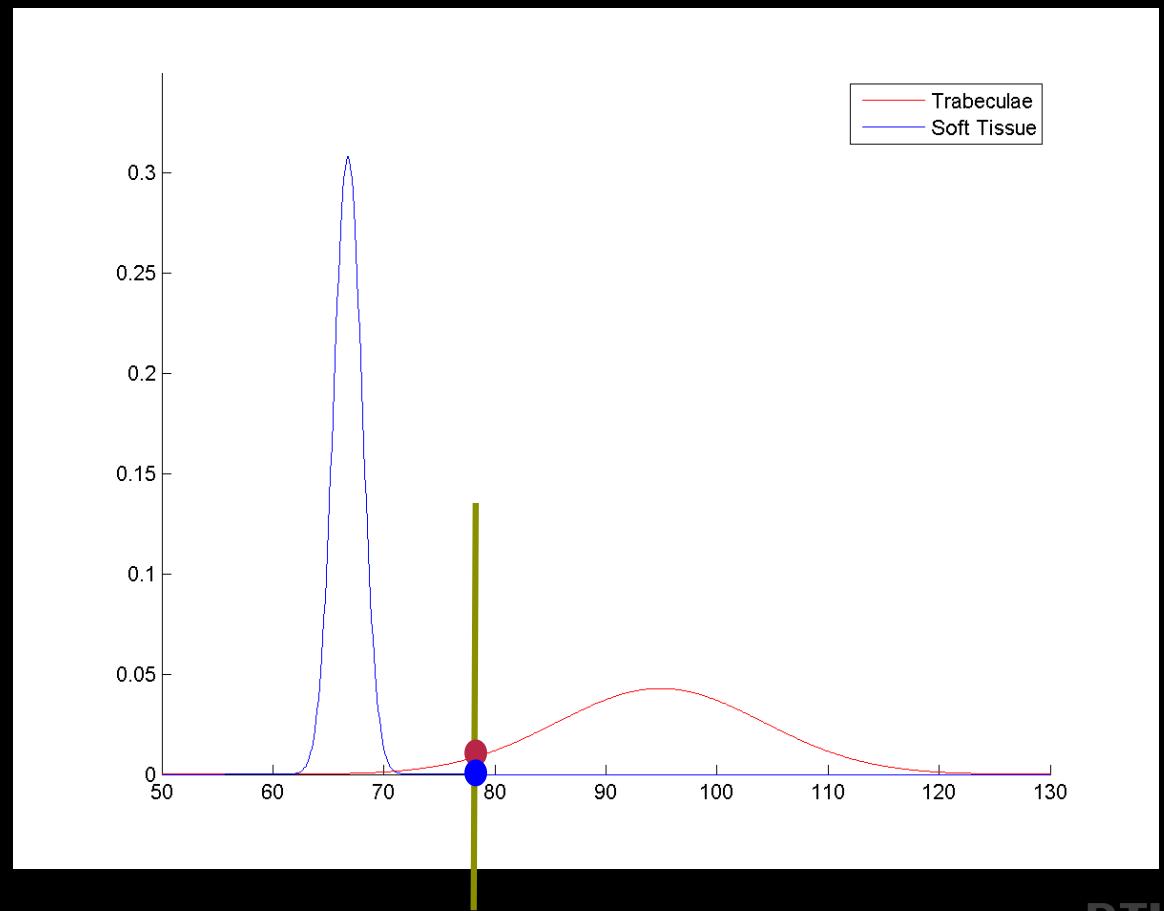
# Quiz 4: Two tissue types - parametric classification

Which tissue class?

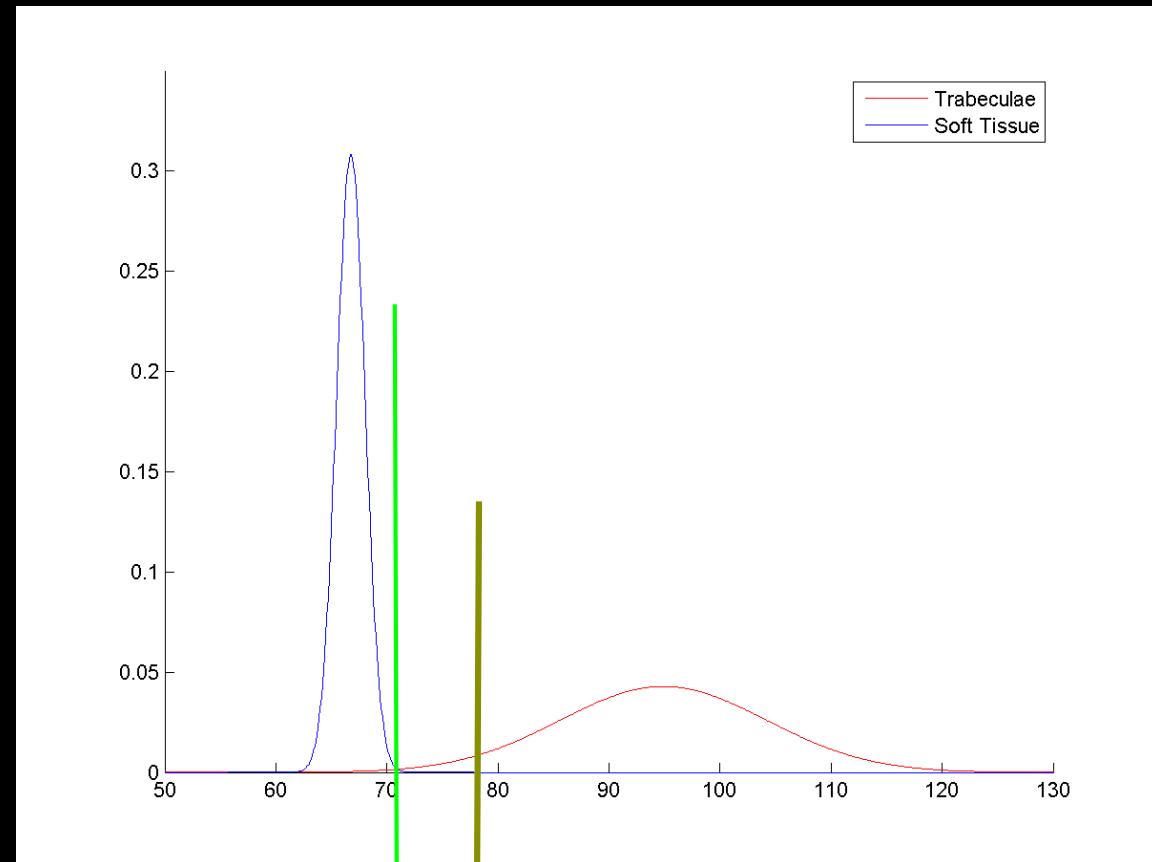
- A) Trabeculae
- B) Soft-tissue

Solution:

The A distribution (red) is higher than B (blue) at  $v=78$



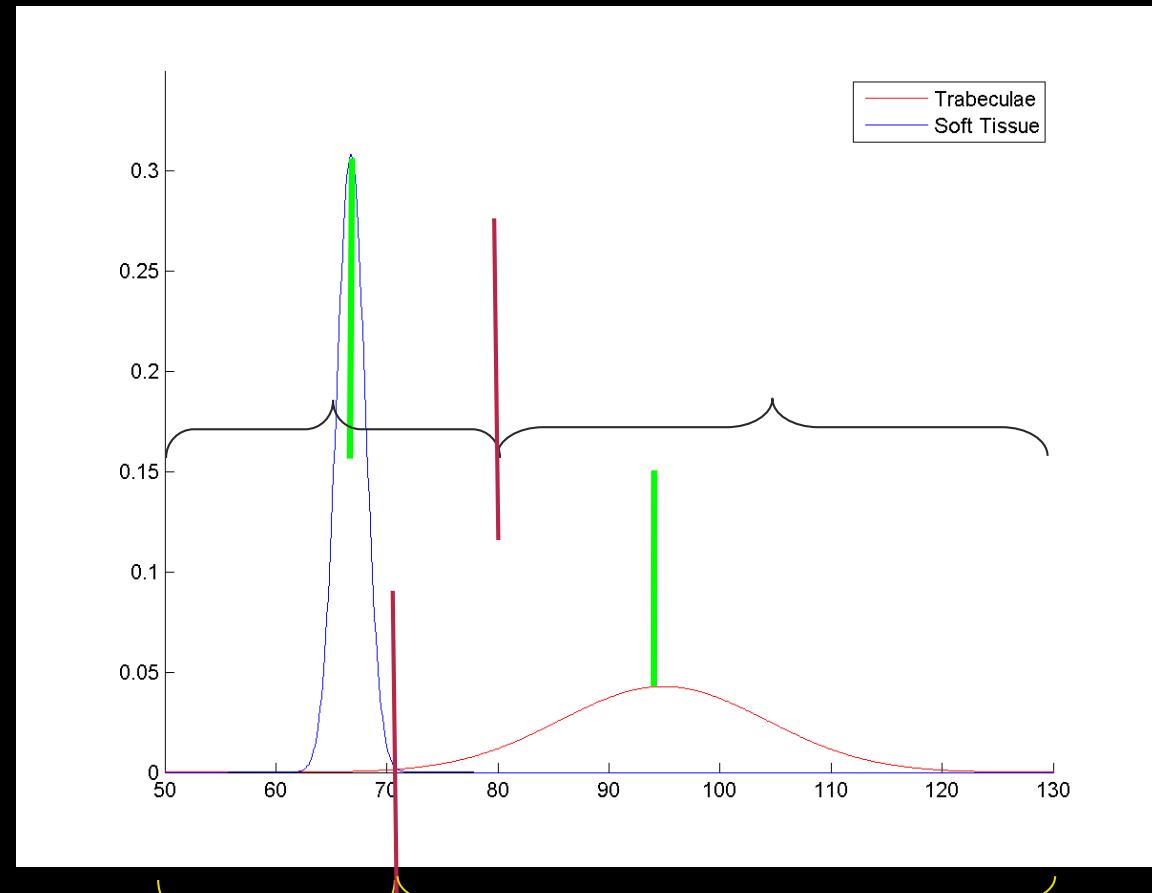
# Parametric classification – repeat the question



$v = 78$

- New pixel with value 78
  - Is it soft-tissue or trabecular bone?
  - Most probably trabecular bone
- Where should we set the limit?
  - Where the two Gaussians cross!

# Parametric classification – ranges



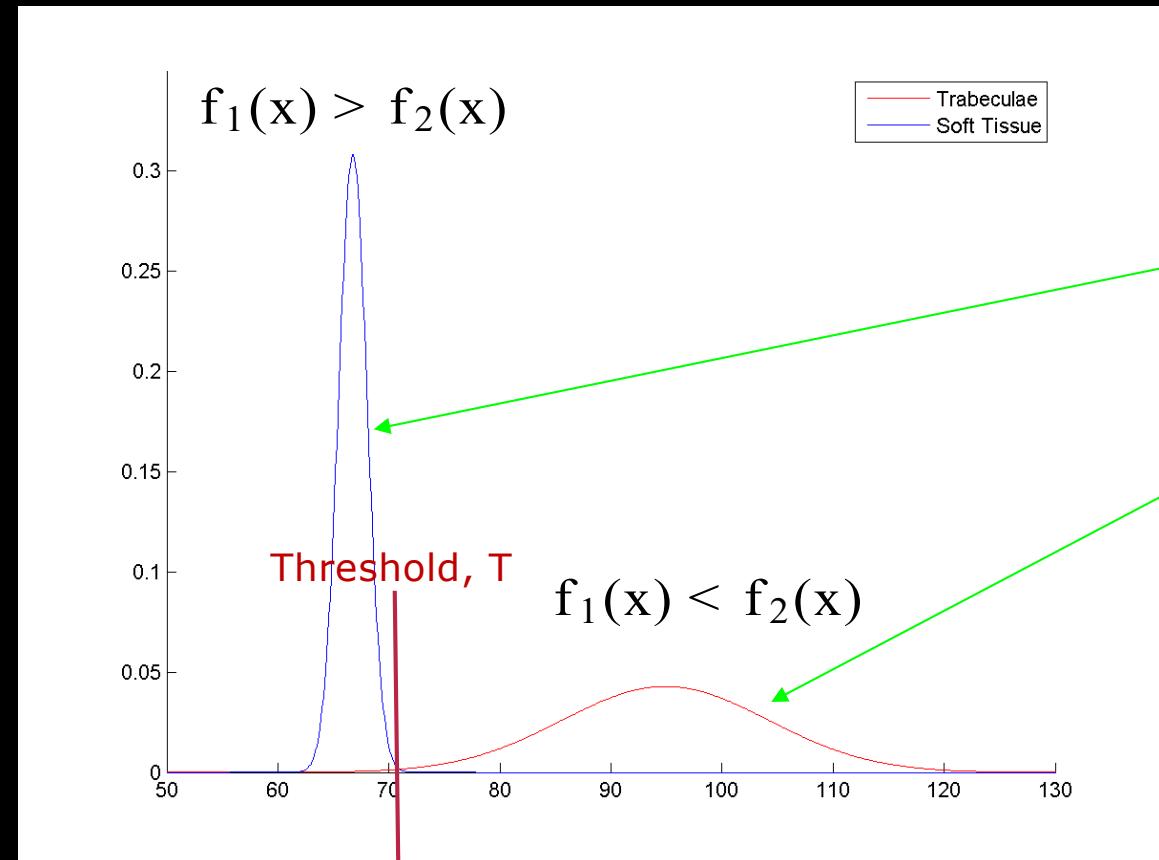
- The pixel value ranges depends on
  - The mean
  - The standard deviation
- Compared to the minimum distance classifier
  - Only the average



# Parametric classification – how to

- Select training pixels for each class
- Fit Gaussians ( $\mathcal{N}(\mu_i, \sigma_i)$ ) to each class
- Use Gaussians to determine pixel value ranges

# Parametric classifier - ranges



- We want to compute where they cross

$$f_1(x) = \frac{1}{\sigma_1 \sqrt{2\pi}} \exp\left(-\frac{(x - \mu_1)^2}{2\sigma_1^2}\right)$$

$$f_2(x) = \frac{1}{\sigma_2 \sqrt{2\pi}} \exp\left(-\frac{(x - \mu_2)^2}{2\sigma_2^2}\right)$$

Create a lookup table:

- Run through all 256 possible pixel values
- Check which Gaussian is the highest
- Store the [value, class] in the table



# Alternatively – analytic solution

The two Gaussians

$$\frac{1}{\sigma_1 \sqrt{2\pi}} \exp\left(-\frac{(v - \mu_1)^2}{2\sigma_1^2}\right) = \frac{1}{\sigma_2 \sqrt{2\pi}} \exp\left(-\frac{(v - \mu_2)^2}{2\sigma_2^2}\right)$$

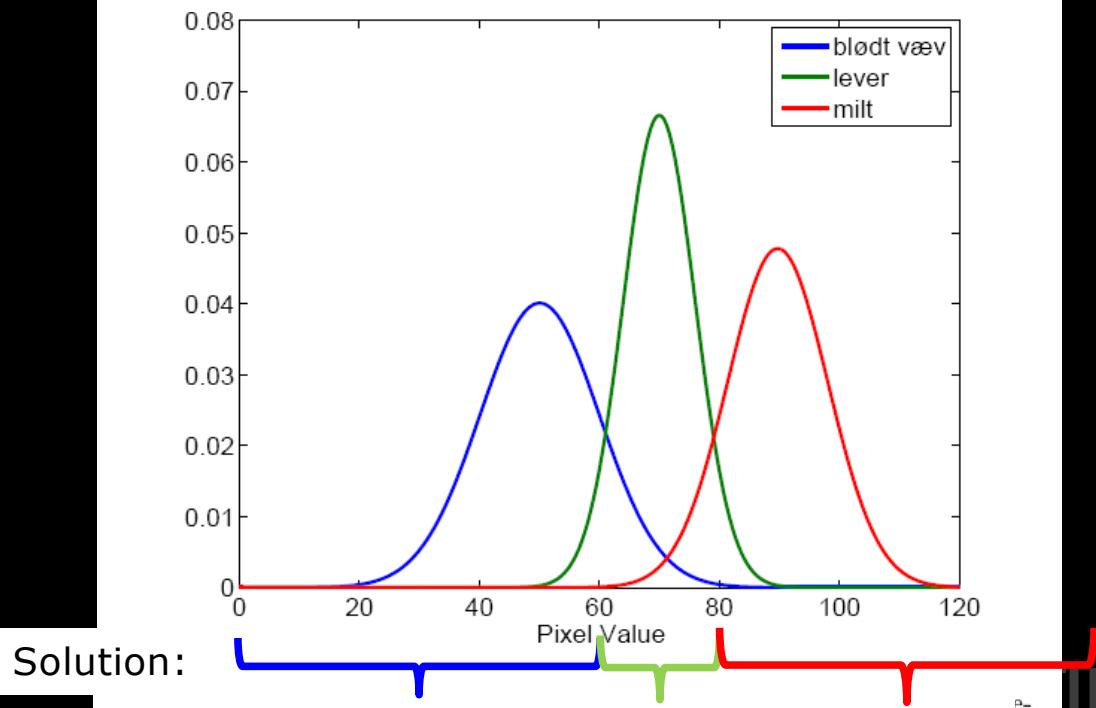
Intercept at

$$v = \frac{\sigma_1^2 \mu_2 - \sigma_2^2 \mu_1 \pm \sqrt{-\sigma_1^2 \sigma_2^2 \left(2 \mu_2 \mu_1 - \mu_2^2 - 2 \sigma_2^2 \ln\left(\frac{\sigma_2}{\sigma_1}\right) - \mu_1^2 + 2 \sigma_1^2 \ln\left(\frac{\sigma_2}{\sigma_1}\right)\right)}}{-\sigma_2^2 + \sigma_1^2}$$

## Quiz 5: Class ranges

- A) [0,45], ]45, 75], ]75,255]
- B) [40,60], ]60,100],]100,140]
- C) [0, 60],]60,80],]80,255]
- D) [0,60],]60,100],]100,255]
- E) [0,75],[75,100],]100,255]

An expert have chosen representative regions in an image that contains soft tissue, liver and spleen. The image pixel minimum and maximum values are 0 and 255. To make a parametric classification, the histograms are parameterized using Gaussian distributions as seen in the image. What are the class ranges?





# Thomas Bayes



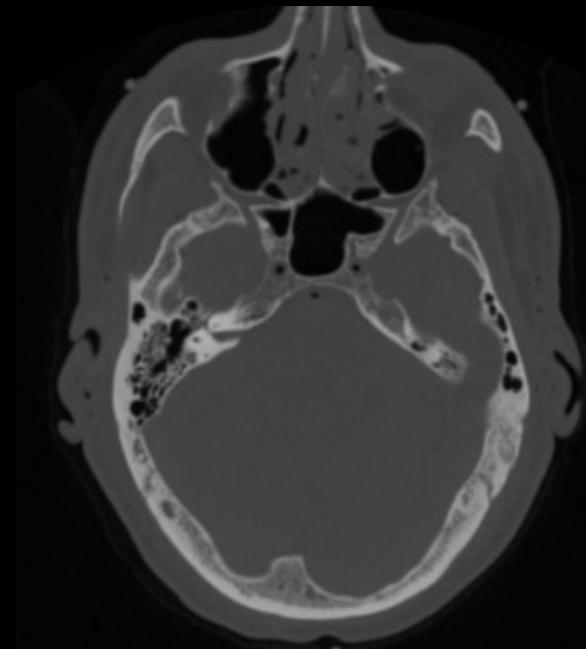
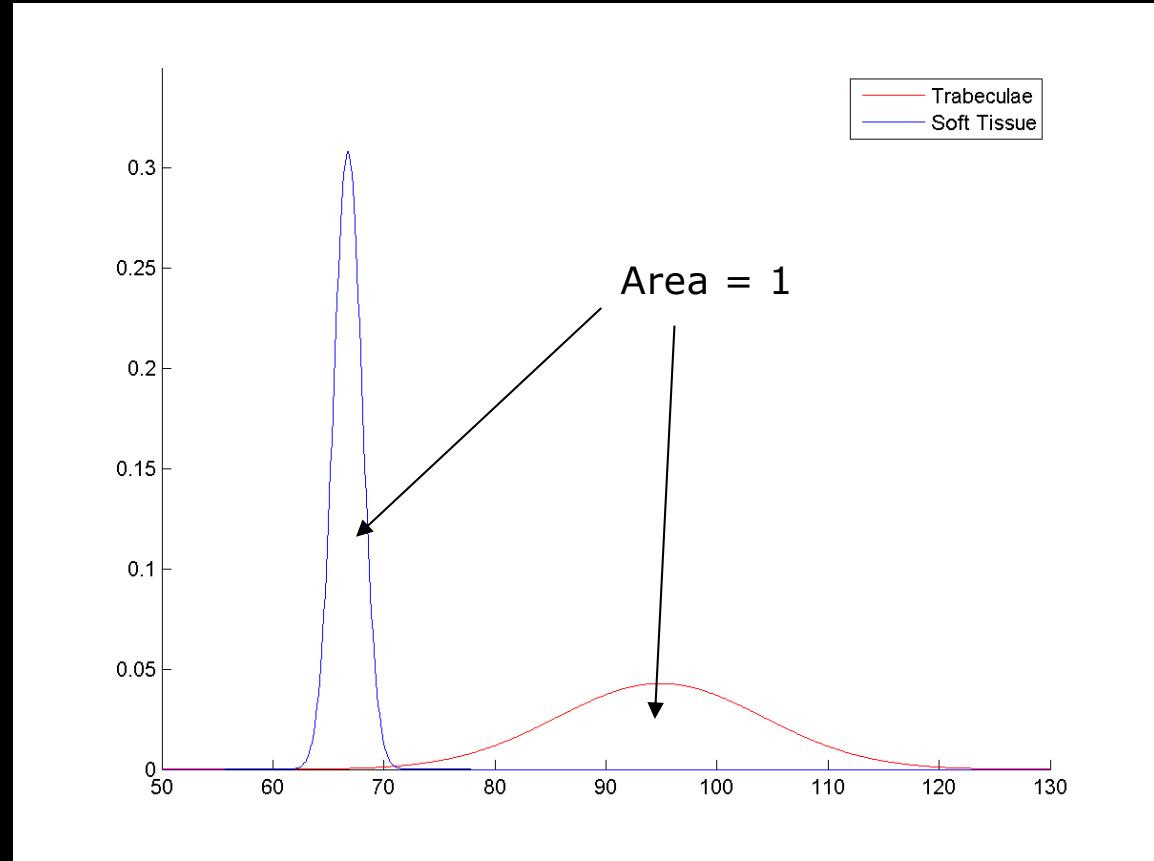
Wikipedia

- 1702-1761
- English mathematician and Presbyterian minister
- Bayes' theorem

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

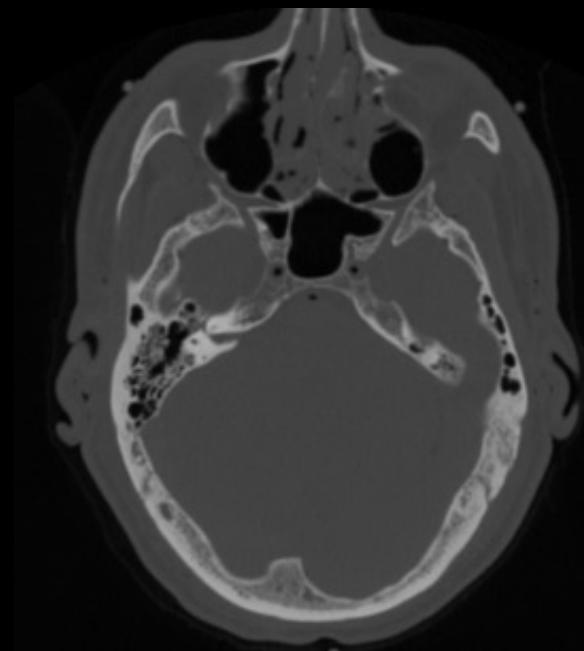
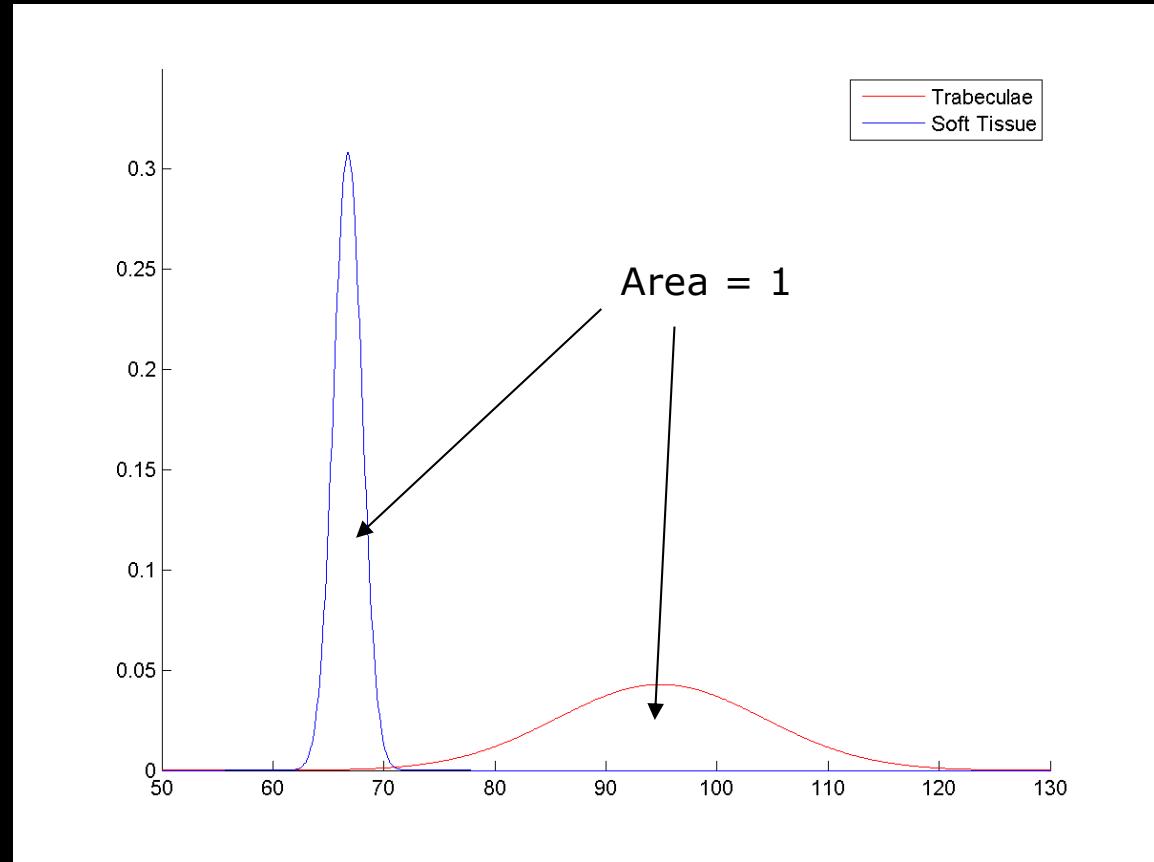
# Bayesian Classification

Pure parametric classifier  
assumes **equal amount** of  
different tissue types



# Bayesian Classification

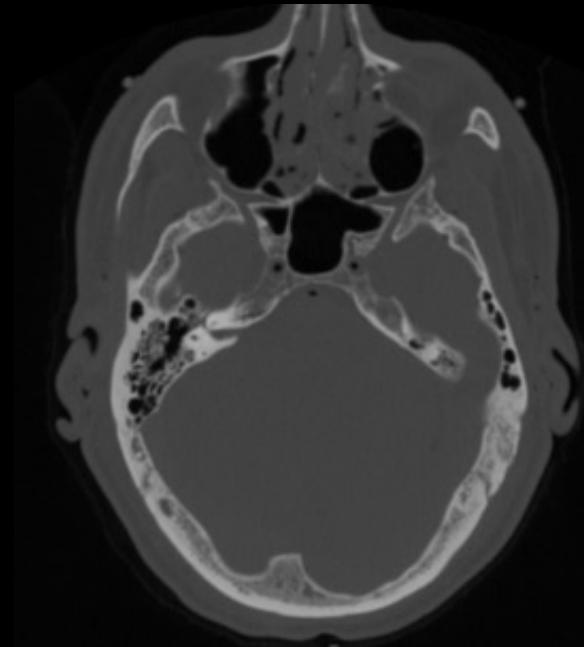
But much more soft-tissue than trabecular bone



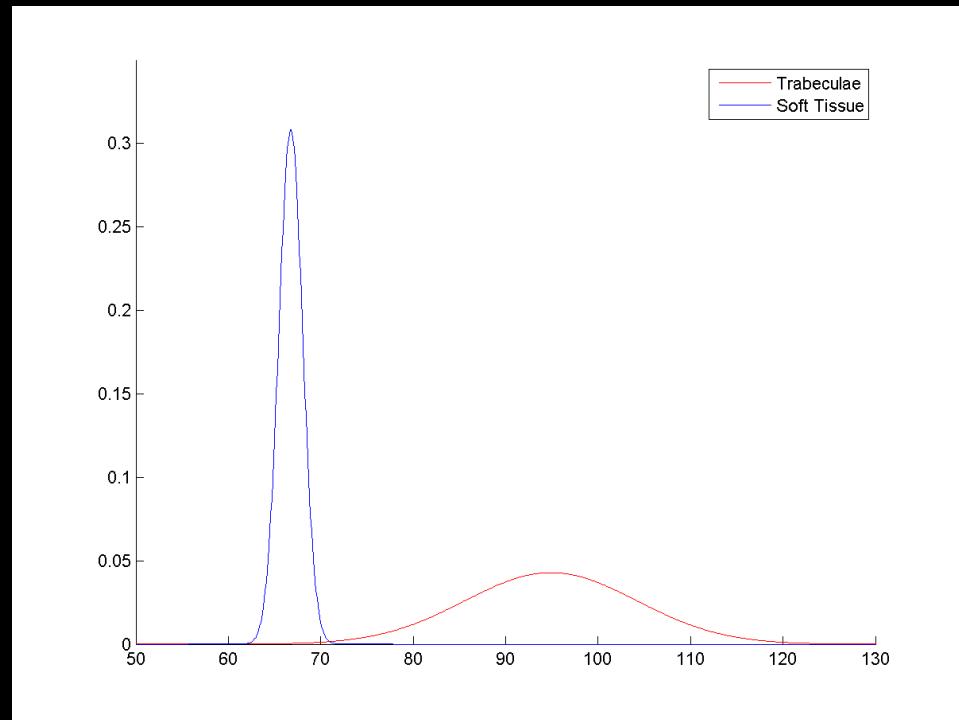
How do we handle that?

# Bayesian Classification

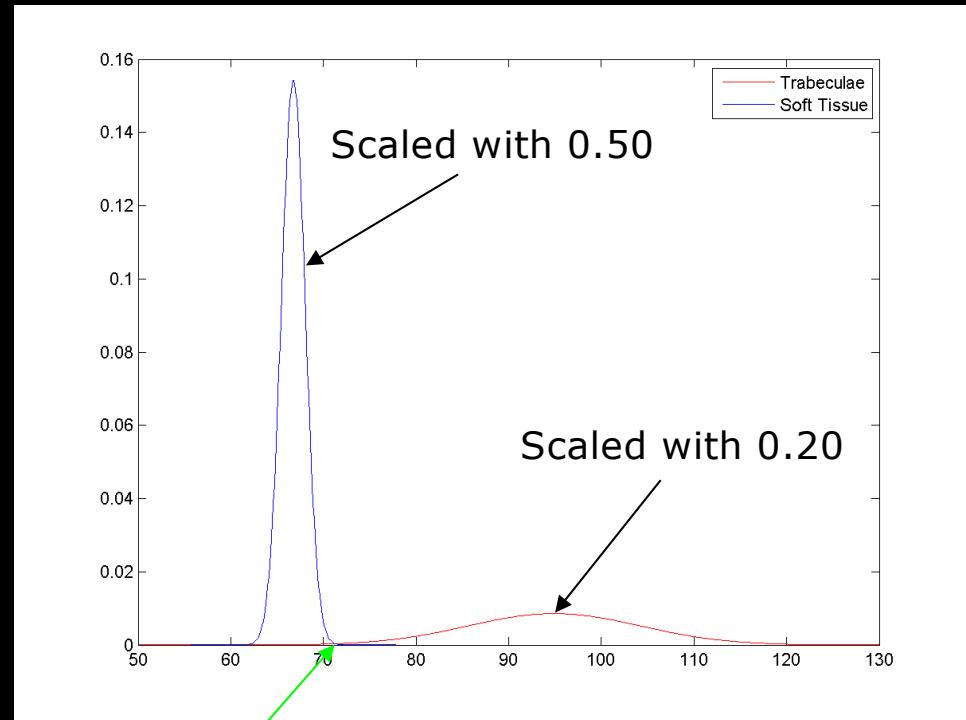
- An expert tells us that a CT scan of a head contains
  - 20% Trabecular bone
  - 50% Soft-tissue
- Picking a random pixel in the image
  - 20% Chance that it is trabecular bone
  - 50% Chance that it is soft-tissue
- How to use that?



# Bayesian Classification – histogram scaling



Parametric classifier



Bayesian classifier

Little change in class border  
(sometimes significant changes)



# Formal definition

- The *posterior probability*
- Given a pixel value  $v$ 
  - What is the probability that the pixel belongs to class  $C_i$

**Example:** If the pixel value is 78, what is the probability that the pixel is bone

$$P(c_i|v) = \frac{P(v|c_i)P(c_i)}{P(v)}$$

# Formal definition

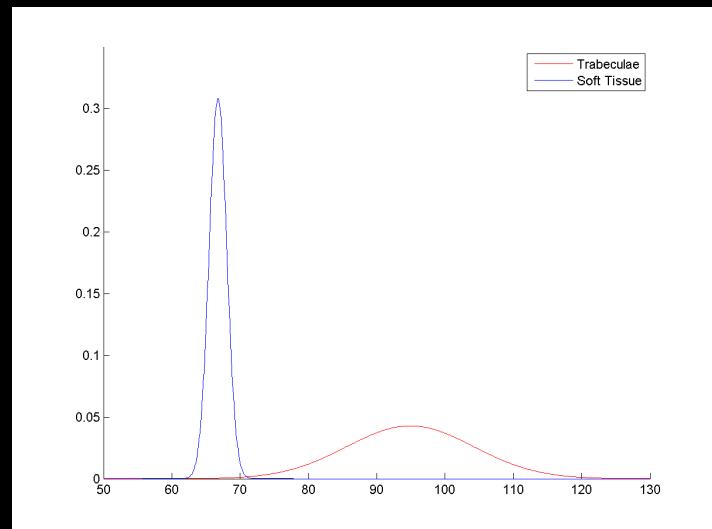
- The *a priori probability* (what is known from before)

**Example:** From general biology it is known that 20% of a brain CT scan is trabecular bone. Therefore  $P(\text{trabecular}) = 0.20$

$$P(c_i|v) = \frac{P(v|c_i)P(c_i)}{P(v)}$$

# Formal definition

- The *class conditional probability* also called the *likelihood*
- Given a class, what is the probability of a pixel with value  $v$ ?



**Example:** If we consider class = soft-tissue.  
What is the probability that the pixel value is 78?

Very low

$$P(c_i|v) = \frac{P(v|c_i)P(c_i)}{P(v)}$$



# Formal definition

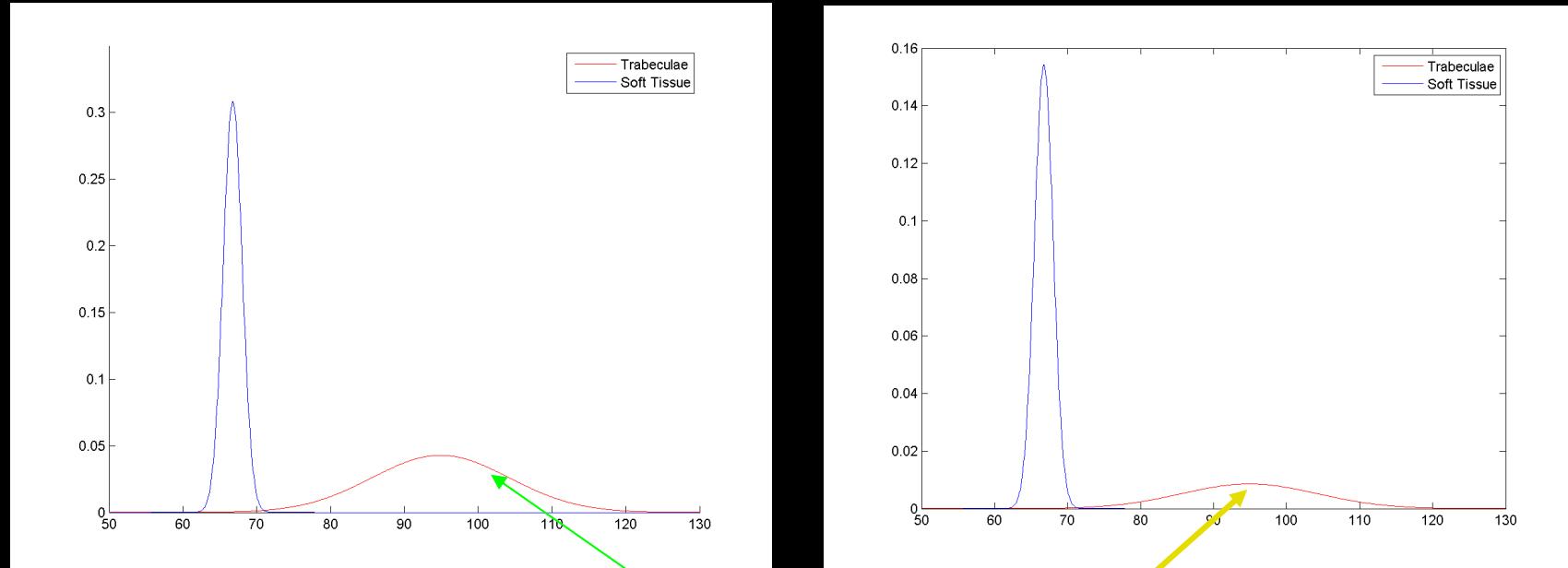
- The *model evidence* or *marginal probability*
- It is basically a normalisation factor:  $P(v) = \sum_i P(v|c_i)P(c_i)$

Constant – ignored from now on

$$P(c_i|v) = \frac{P(v|c_i)P(c_i)}{P(v)}$$



# Formal definition – sum up



$$P(c_i|v) = \frac{P(v|c_i)P(c_i)}{P(v)}$$

;  $C_i$ =trabeculae



# Bayesian classification – how to

- Select training pixels for each class
- Fit Gaussians to each class
- Ask an expert for the prior probabilities (how much there normally is in total of each type)
- For each pixel in the image
  - Compute  $P(c_i|\nu)$  for each class (the *a posterior probability*)
  - Select the class with the highest  $P(c_i|\nu)$

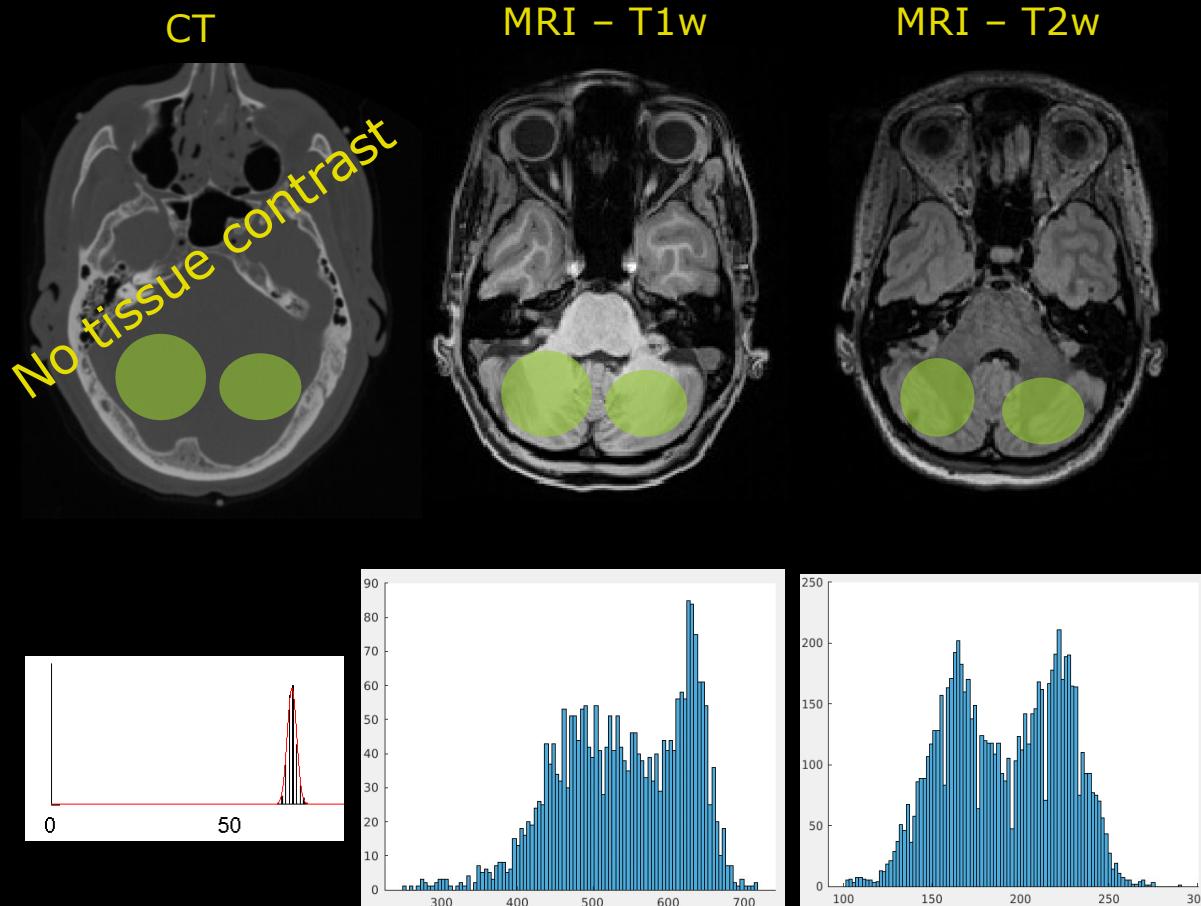
$$P(c_i|\nu) = \frac{P(\nu|c_i)P(c_i)}{P(\nu)}$$



## When to use Bayesian classification

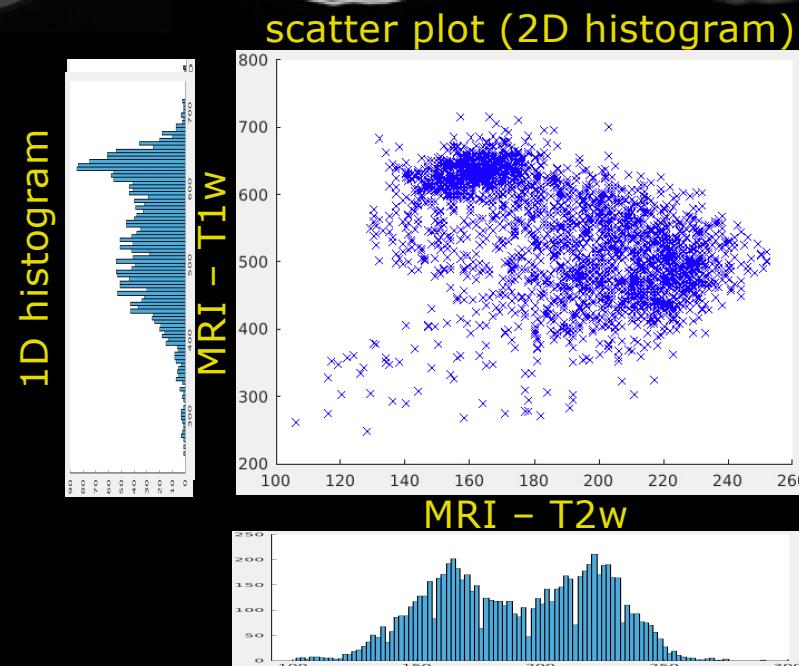
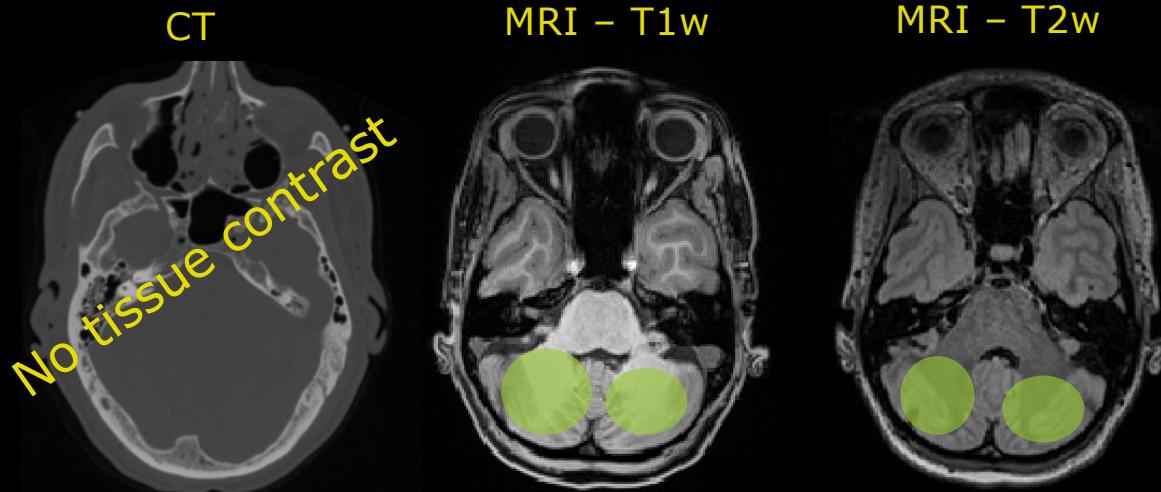
- The *parametric classifier* is good when there are approximately the same amount of all type of tissues
- Use *Bayesian classification* if there are very little or very much of some types
- A more general formulation for segmentation
  - especially when going to a higher dimensional feature space

# High dimensional feature space



- Combine different feature inputs to **improve** segmentation
  - Different image modalities e.g. CT vs MRI
  - Subject groups
    - Healthy vs disease
  - Different angles of object e.g. cars

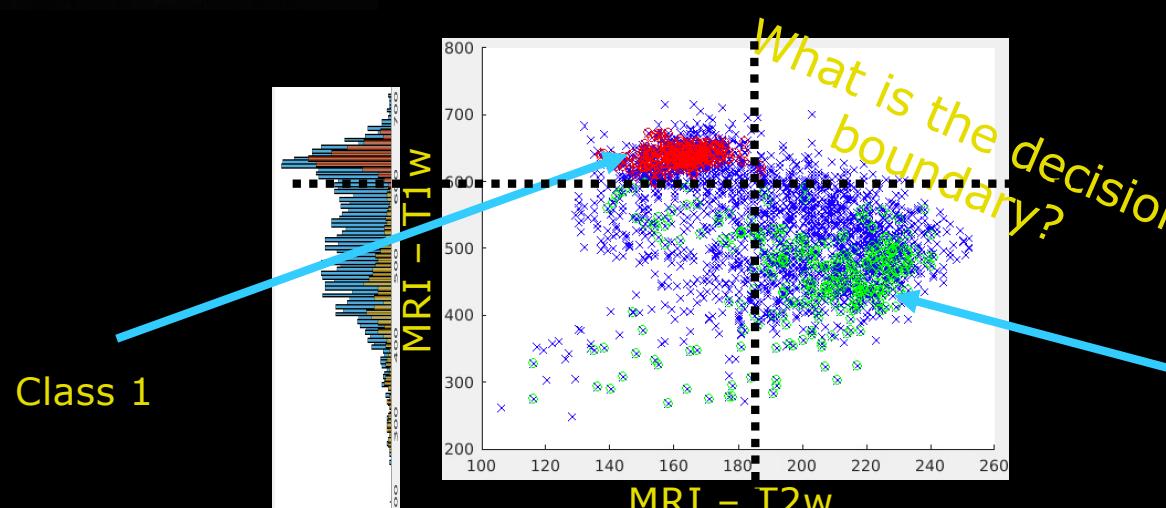
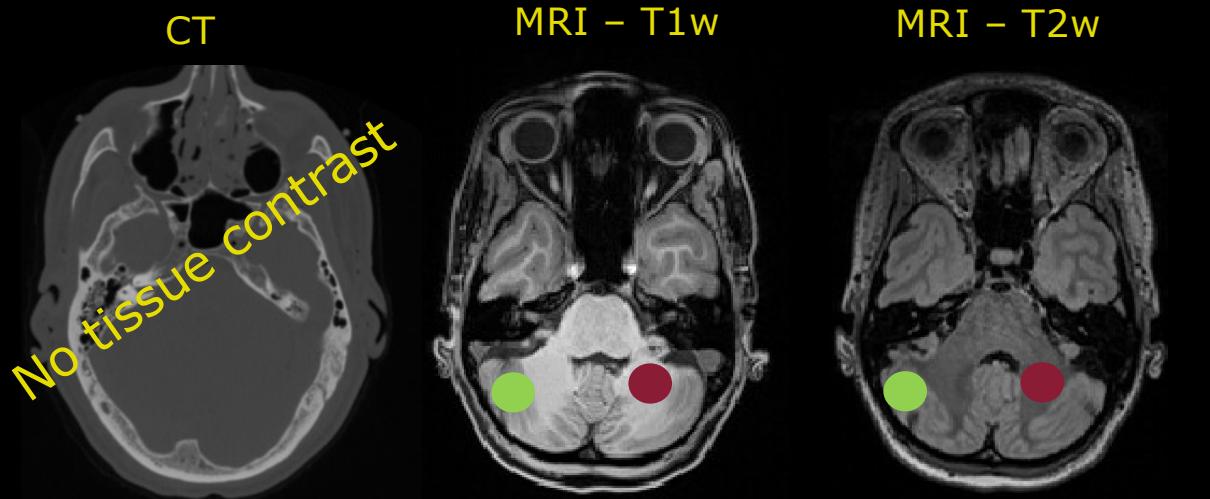
# High dimensional feature space



## ■ Feature space:

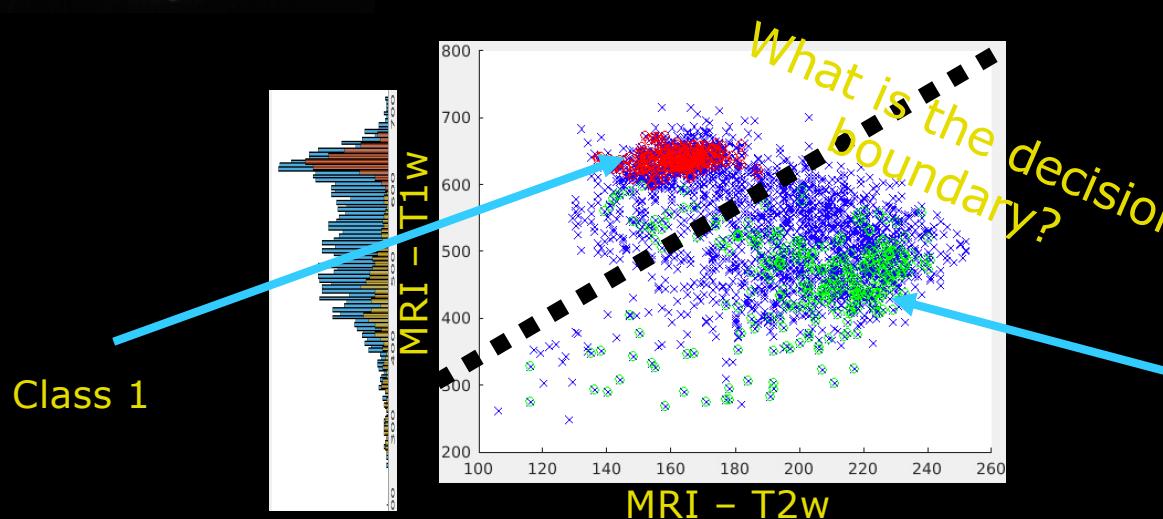
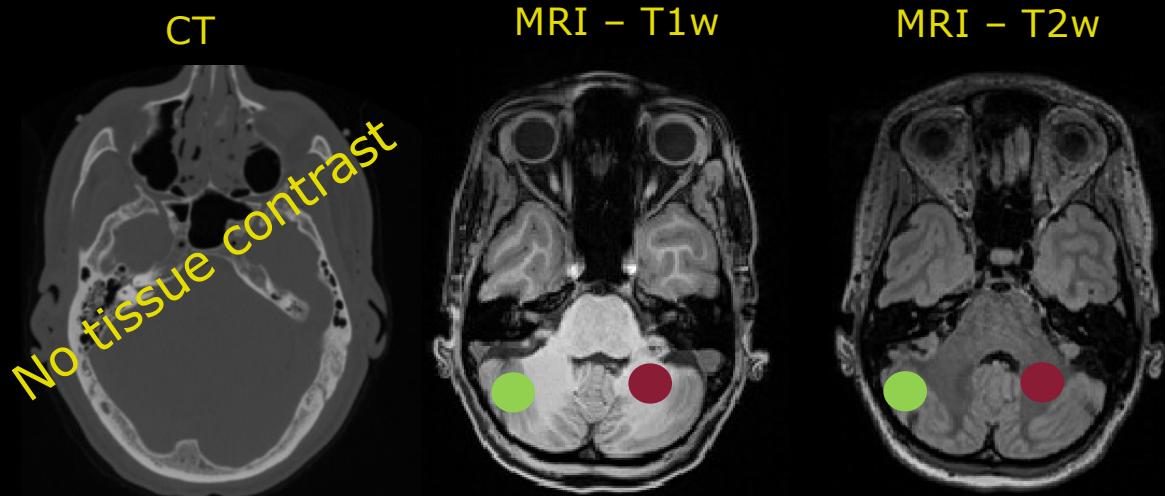
- 1D is a histogram
- 2D is a scatterplot i.e. 2D histogram
- >2D is bit more complicated to show

# High dimensional feature space



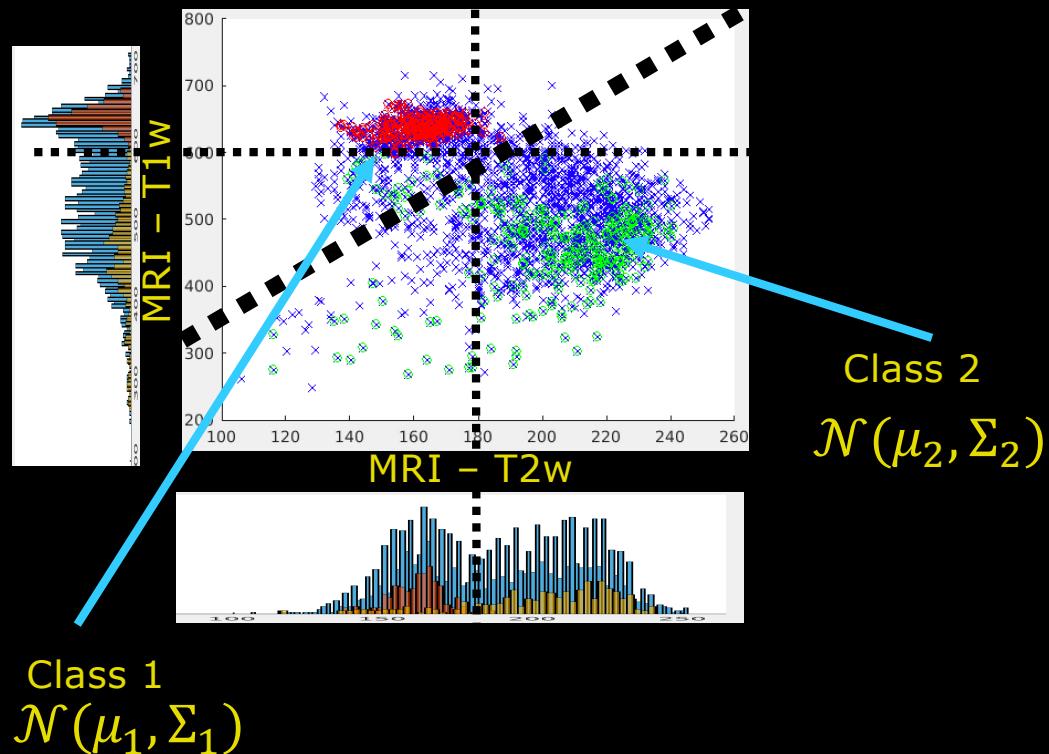
- Segmentation with more feature inputs
- To train our **classifier** model with class examples
  - Draw tissue specific regions for each class
  - **Class 1** and **Class 2**
  - Tissue **type 1** and **type 2**
- Segmentation:
  - Define the threshold for the decision boundary?
  - 1D vs 2D

# High dimensional feature space



- Segmentation with more feature inputs
- To train our **classifier** model with class examples
  - Draw tissue specific regions for each class
  - **Class 1** and **Class 2**
  - Tissue **type 1** and **type 2**
- Segmentation:
  - Define the threshold for the decision boundary?
  - 1D vs 2D

# Decision boundary: Define a model



- 2D feature space
  - Better class separation vs 1D?
- Model assumption
  - Type of distribution?
- Intensity histograms looks Gaussian-like, or?
  - We assume Gaussian distributions:  $\mathcal{N}(\mu_i, \Sigma_i)$
- Use Bayes theorem
  - Probability of belonging to C2:
$$\frac{P(C2|x)}{P(C1|x)} > T$$
- Desicion boundary
  - A hyperplane for  $T=1$ :
  - $P(C2|x)=P(C1|x)$

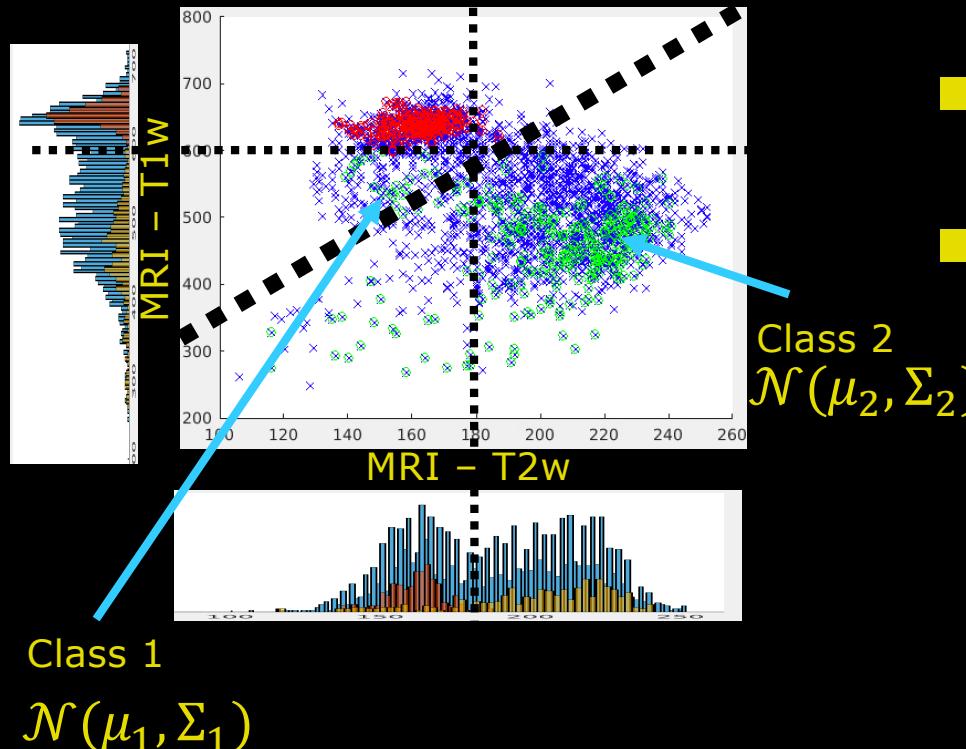
# Decision boundary: Train a model

- We wish to use Bayes:

$$\frac{P(C2|x)}{P(C1|x)} > T$$

- The posterior probability
  - $P(Ci|x) = P(x|\mu_i, \Sigma_i)P_{Ci}$
- The likelihood: A Gaussian model  

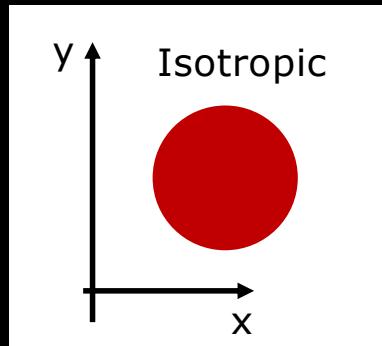
$$P(x|\mu_i, \Sigma_i) = K_i \exp((x - \mu_i)^T \Sigma_i^{-1} (x - \mu_i))$$



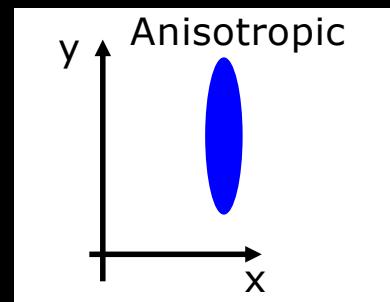
- What about the prior probability  $P(C_i)$ ?

- Data points:
  - $x_i = [x_1, x_2]^T$
- Training set:
  - $t_{x \in C1} = 0$  and  $t_{x \in C2} = 1$
- The class mean-parameter
  - $\mu_i = \frac{1}{N} \sum_{n \in Ci} x_n$
- The covariance matrix-parameter
  - $\Sigma_i = (x - \mu_i)^T (x - \mu_i)$

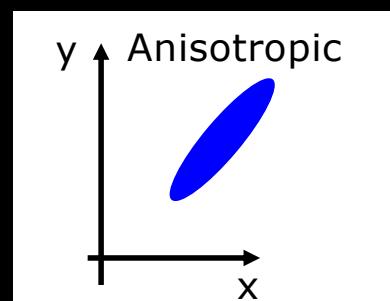
# Gaussian in 2D: The covariance matrix



Rotational invariant

$$\Sigma = \begin{bmatrix} \sigma_{xx}^2 & 0 \\ 0 & \sigma_{yy}^2 \end{bmatrix}$$
$$\sigma_{xx} = \sigma_{yy}$$


Aligned with coordinate system

$$\Sigma = \begin{bmatrix} \sigma_{xx}^2 & 0 \\ 0 & \sigma_{yy}^2 \end{bmatrix}$$
$$\sigma_{xx}^2 \neq \sigma_{yy}^2$$


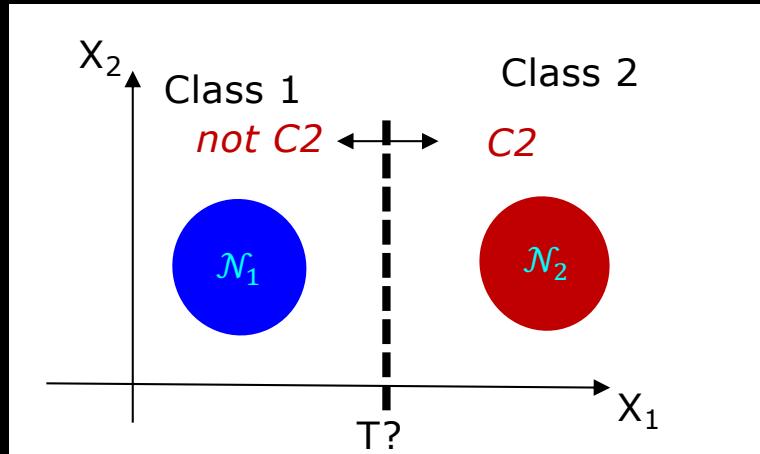
Not aligned with coordinate system

$$\Sigma = \begin{bmatrix} \sigma_{xx}^2 & \sigma_{xy}^2 \\ \sigma_{yx}^2 & \sigma_{yy}^2 \end{bmatrix}$$

## QUICK REFRESH:

- The covariance matrix:  
 $\Sigma_i = (x - \mu_i)^T(x - \mu_i)$
- Expresses the orientation of anisotropic variance in relation to coordinate system

# The linear discriminant classifier



- Classifier: If  $\mathbf{x}$  belongs to  $C_2$ :

$$\frac{P(C_2|\mathbf{x})}{P(C_1|\mathbf{x})) > T}$$

- Take the logarithm

$$\ln(P(C_2|\mathbf{x})) - \ln(P(C_1|\mathbf{x})) > \ln(T)$$

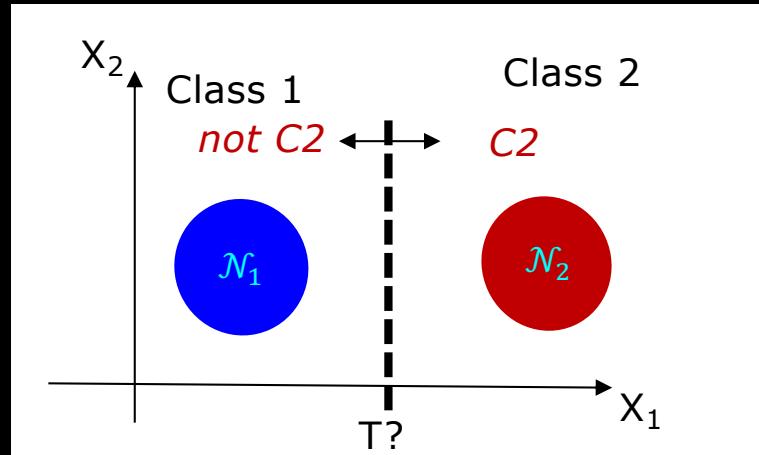
$$\mathcal{N}_1(\mu_1, \Sigma_1) \quad \mathcal{N}_2(\mu_2, \Sigma_2)$$

Inspiration derive:

[https://en.wikipedia.org/wiki/Linear\\_discriminant\\_analysis](https://en.wikipedia.org/wiki/Linear_discriminant_analysis)

<https://people.revoledu.com/kardi/tutorial/LDA/LDA%20Formula.htm>

# The linear discriminant classifier



$$\mathcal{N}_1(\mu_1, \Sigma_1) \quad \mathcal{N}_2(\mu_2, \Sigma_2)$$

- Classifier: If  $\mathbf{x}$  belongs to  $C_2$ :

$$\frac{P(C_2|\mathbf{x})}{P(C_1|\mathbf{x})) > T}$$

- Take the logarithm

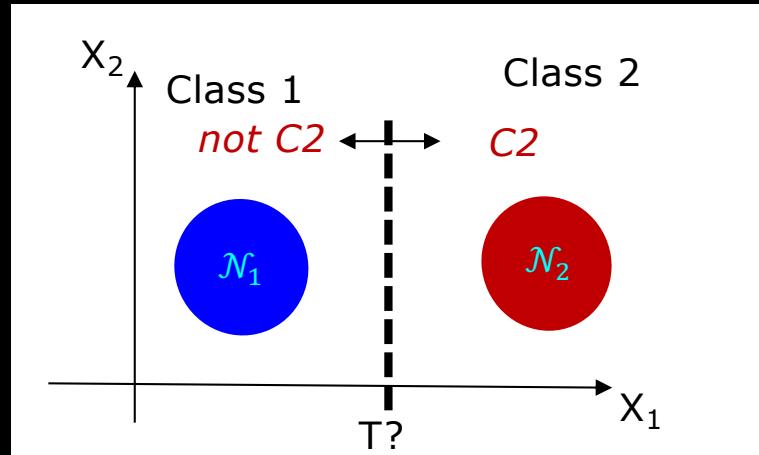
$$\ln(P(C_2|\mathbf{x})) - \ln(P(C_1|\mathbf{x})) > \ln(T)$$

- Where the log-posterior probability for  $C_i$ :

$$\ln(P(C_i|\mathbf{x})) = \frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1} (\mathbf{x} - \boldsymbol{\mu}_i) + \ln(K_i) + \ln(P_i)$$

- $P_i$  is the prior probability for class  $C_i$

# The linear discriminant classifier


 $\mathcal{N}_1(\mu_1, \Sigma_1)$ 
 $\mathcal{N}_2(\mu_2, \Sigma_2)$ 

- Classifier: If  $\mathbf{x}$  belongs to  $C_2$ :

$$\frac{P(C_2|\mathbf{x})}{P(C_1|\mathbf{x})) > T}$$

- Take the logarithm

$$\ln(P(C_2|\mathbf{x})) - \ln(P(C_1|\mathbf{x})) > \ln(T)$$

- Where the log-posterior probability for  $C_i$ :

$$\ln(P(C_i|\mathbf{x})) = \frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1} (\mathbf{x} - \boldsymbol{\mu}_i) + \ln(K_i) + \ln(P_i)$$

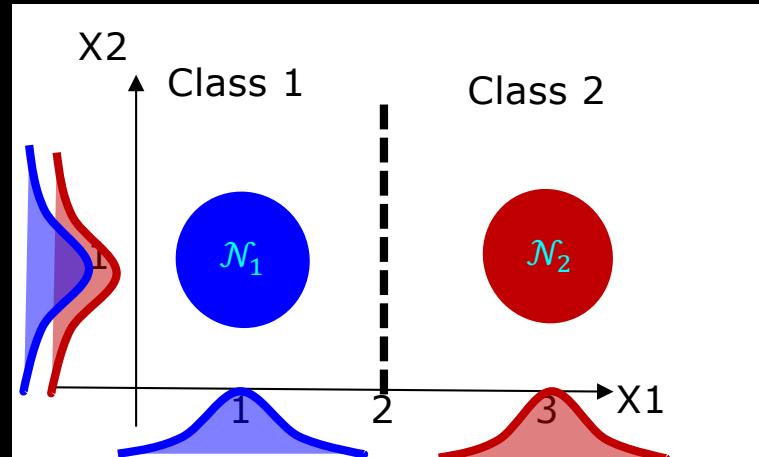
- $P_i$  is the prior probability for class  $C_i$

- Assuming homoscedasticity ( $\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2 = \boldsymbol{\Sigma}_0$ ) and isotropic covariance matrix we have **the Linear Discriminant Analysis (LDA) classifier model:**

$$\ln \frac{P_2}{P_1} - \frac{1}{2}(\boldsymbol{\mu}_2 + \boldsymbol{\mu}_1)^T \boldsymbol{\Sigma}_0^{-1} (\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1) + \mathbf{x}^T \boldsymbol{\Sigma}_0^{-1} (\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1) > \ln(T)$$

- We train the classifier with examples obtained from the two distributions N1 and N2

# Quiz 6 - The LDA classifier



## ■ Linear Discriminat Analysis (LDA):

$$\ln \frac{P_2}{P_1} - \frac{1}{2}(\boldsymbol{\mu}_2 + \boldsymbol{\mu}_1)^T \boldsymbol{\Sigma}_0^{-1} (\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1) + \mathbf{x}^T \boldsymbol{\Sigma}_0^{-1} (\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1) > \ln(T)$$

Where:

$$\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2 = \boldsymbol{\Sigma}_0 = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$$

Prior probabilities:  $P_1=P_2=0,5$

Which data points are placed on the hyperplane for  $P(C2|x)=P(C1|x)$ ?

- A)  $[0,5]^T$
- B)  $[1,7]^T$
- C)  $[3,3]^T$
- D)  $[2,0]^T$
- E)  $[0,7]^T$

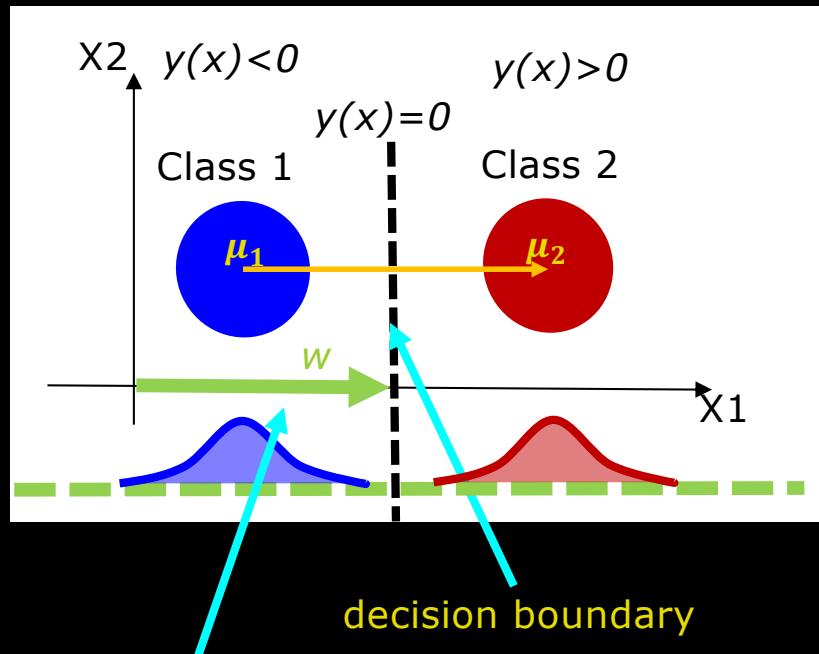
**Solution** – We see that when  $T=1=>\ln(1)=0$  is the decision boundary which is placed only along  $X_1$  i.e. a solution in 1D:

$$\ln \frac{P_2}{P_1} - \frac{1}{2}(\boldsymbol{\mu}_2 + \boldsymbol{\mu}_1) \frac{(\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1)}{\sigma_0} = -x_1 \frac{(\boldsymbol{\mu}_2 - \boldsymbol{\mu}_1)}{\sigma_0}$$

$$-\ln \frac{0,5}{0,5} + \frac{1}{2}(3+1) \frac{(3-1)}{2} = x_1 \frac{(3-1)}{2}$$

$$x_1=2 \quad \& \quad x_2= \text{all values}$$

# Projections in the feature space



- $w$  projects the class mean direction i.e. the weight vector
- $w$  is normal to the hyperplane of the decision boundary for  $y_i(x)=0$
- $x^T w$  is a dot product i.e.  $x$  and  $c$  are projected onto  $w$  ( $a^T b = \|a\| \|b\| \cos(\theta)$ )

- General formulation of a classifier
  - A projection of data points in relation to the decision boundary

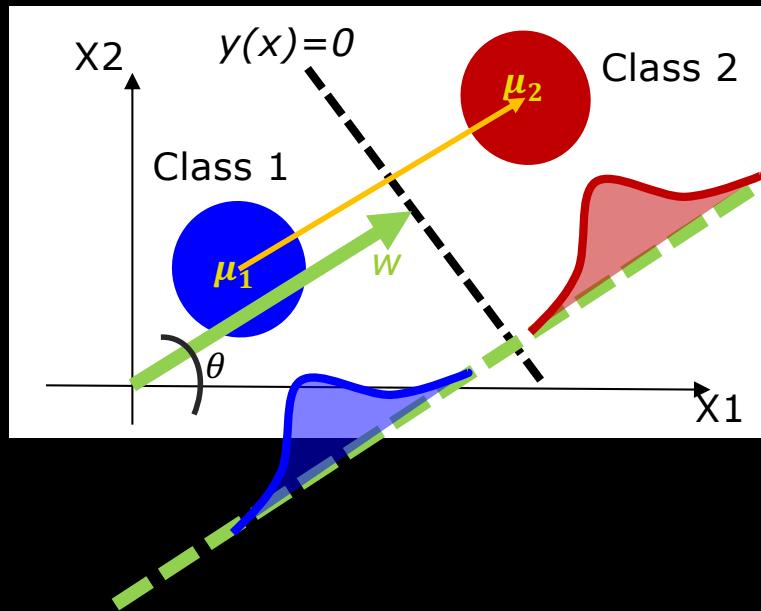
- The LDA function for  $C_2$ :

$$\ln \frac{P_1}{P_2} - \frac{1}{2} (\mu_2 + \mu_1)^T \Sigma_0^{-1} (\mu_2 - \mu_1) + x^T \Sigma_0^{-1} (\mu_2 - \mu_1) > \ln T$$

P1 1 (μ<sub>2</sub> + μ<sub>1</sub>)<sup>T</sup> Σ<sub>0</sub><sup>-1</sup> (μ<sub>2</sub> - μ<sub>1</sub>) x<sup>T</sup> Σ<sub>0</sub><sup>-1</sup> (μ<sub>2</sub> - μ<sub>1</sub>) > lnT  
C W W  
w<sub>0</sub>

- The linear discriminant function  
 $y_{C \in 2}(x) = x^T w + w_0$   
 -where  $w_0$  is the threshold
- $x$  is assigned to  $C2$  if  $y_{C \in 2}(x) > 0$

# Projections in the feature space



- $w$  projects the class mean direction i.e. the weight vector
- $w$  is normal to the hyperplane of the decision boundary  $y_i(x)=0$
- $x^T w$  is a dot product i.e.  $x$  and  $c$  are projected onto  $w$  ( $a^T b = \|a\| \|b\| \cos(\theta)$ )

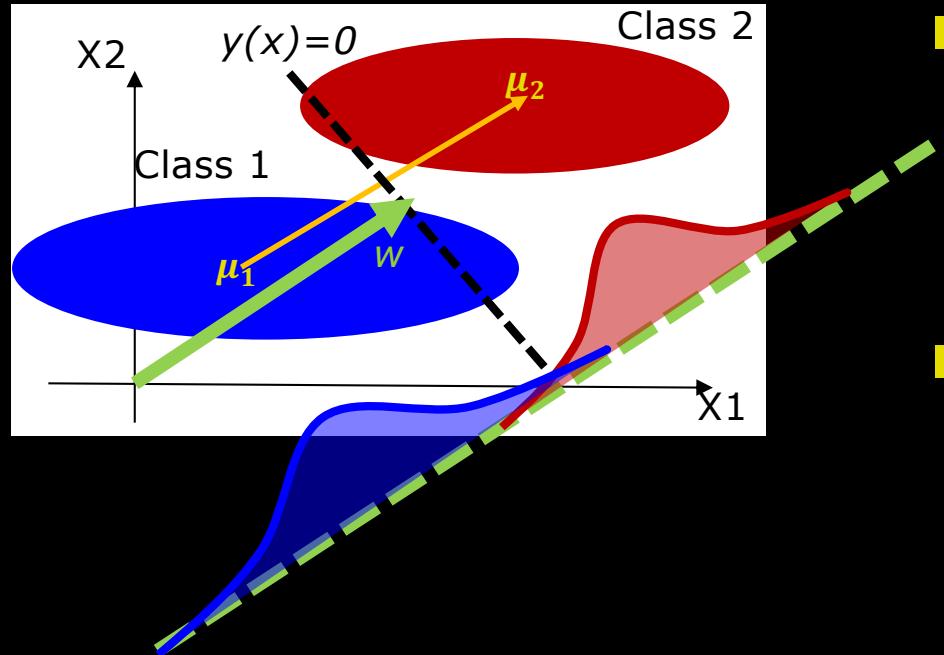
- General formulation of a classifier
  - A projection of data points in relation to the decision boundary
- The LDA function for  $C_2$ :

$$\ln \frac{P_1}{P_2} - \frac{1}{2} (\mu_2 + \mu_1)^T \Sigma_0^{-1} (\mu_2 - \mu_1) + x^T \Sigma_0^{-1} (\mu_2 - \mu_1) > \ln T$$

c      w      w  
 $w_0$

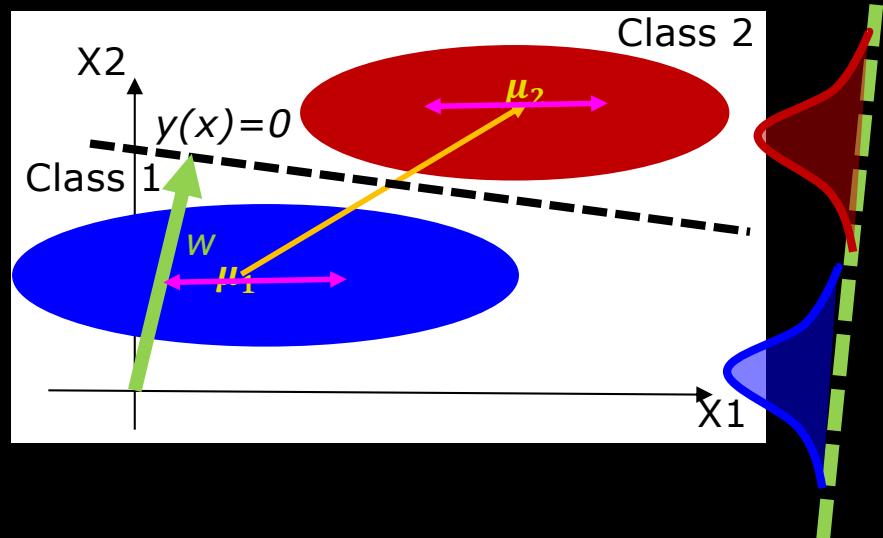
- The linear discriminant function  
 $y_{C \in 2}(x) = x^T w + w_0$   
 -where  $W_0$  is the threshold
- $x$  is assigned to  $C2$  if  $y_{C \in 2}(x) > 0$

# Projections in the feature space



- If the covariance is *anisotropic* and have different class variances
  - The LDA classifier does not ensure an optimal class separation!
  - LDA only separate the class means
- To improve the separation
  - We need to change the model hence the **weight vector,  $\mathbf{W}$**

# Projections in the feature space



Optimal class separation:

- The *weight vector*,  $w$ , now accounts for both class means and variances

## ■ Fisher's LDA:

- Uses: *between-class (means) covariance*:  

$$S_B = (\mu_2 - \mu_1)^T(\mu_2 - \mu_1)$$
- and: optimise *(total) within-class covariance*  

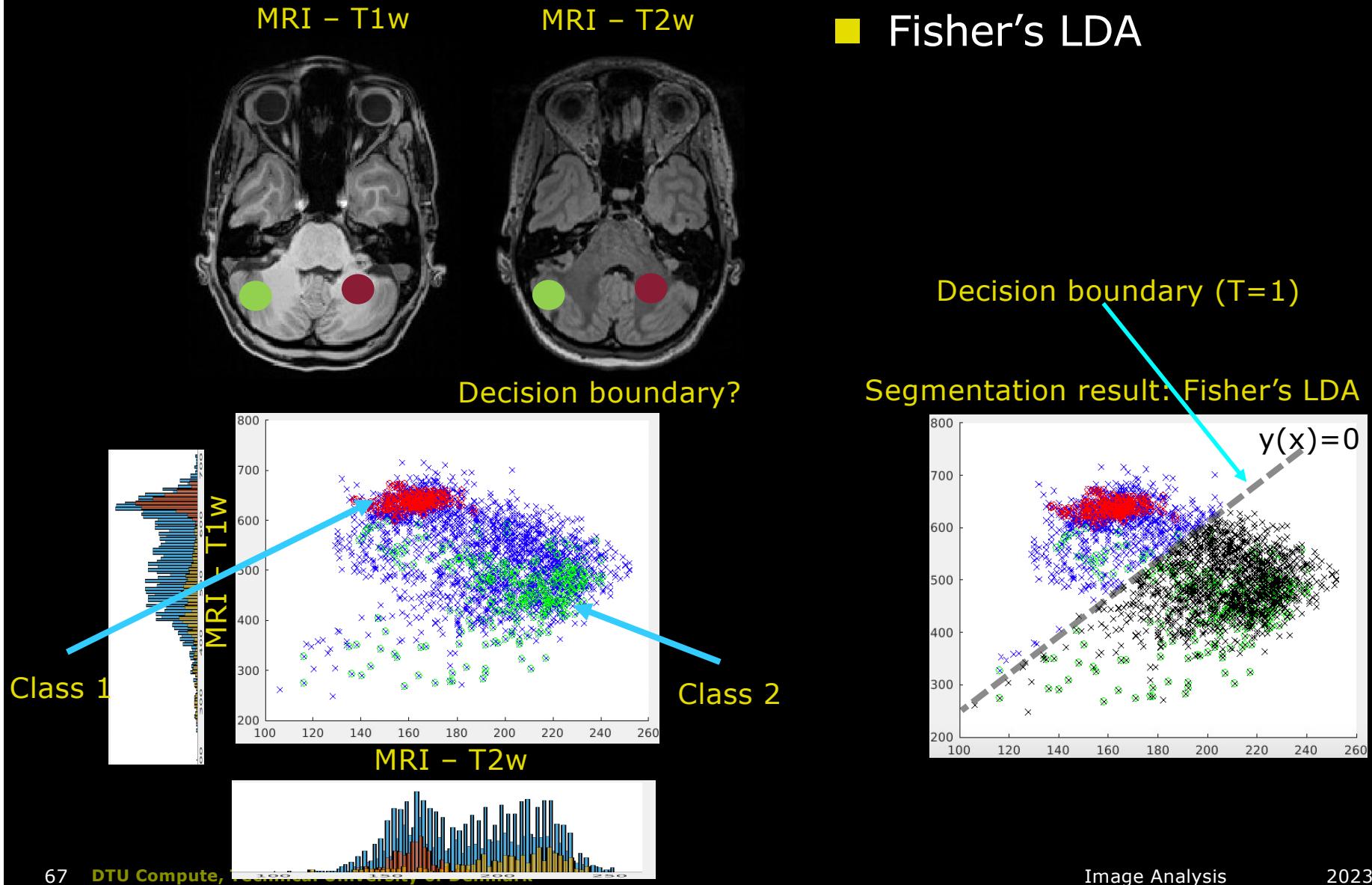
$$S_W = \Sigma_1 + \Sigma_2$$

## ■ Find projection $w$ using a cost function:

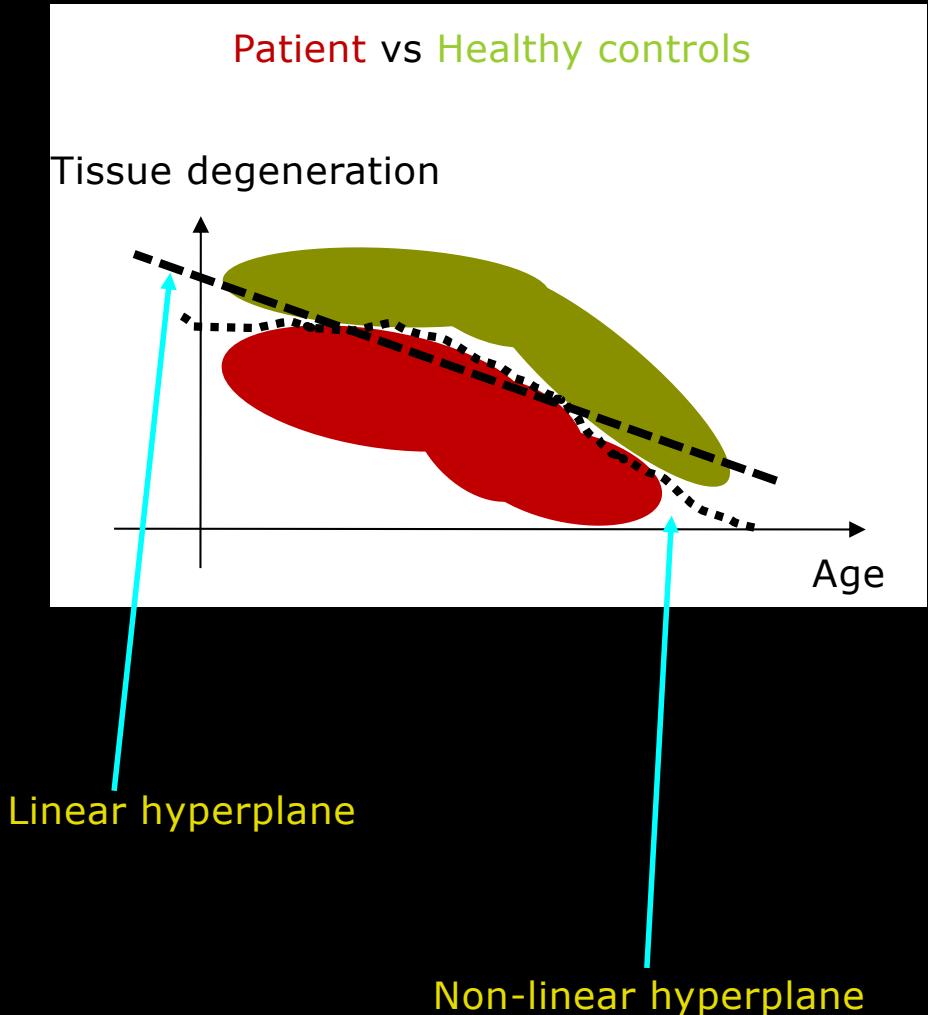
- $$J(w) = \frac{w^T S_B w}{w^T S_W w}$$
- differentiate:  $\frac{\partial J(w)}{\partial w} = 0$
- which gives (simple solution):  

$$w \propto S_W^{-1}(\mu_2 - \mu_1)$$

# Segmentation of brain data using LDA

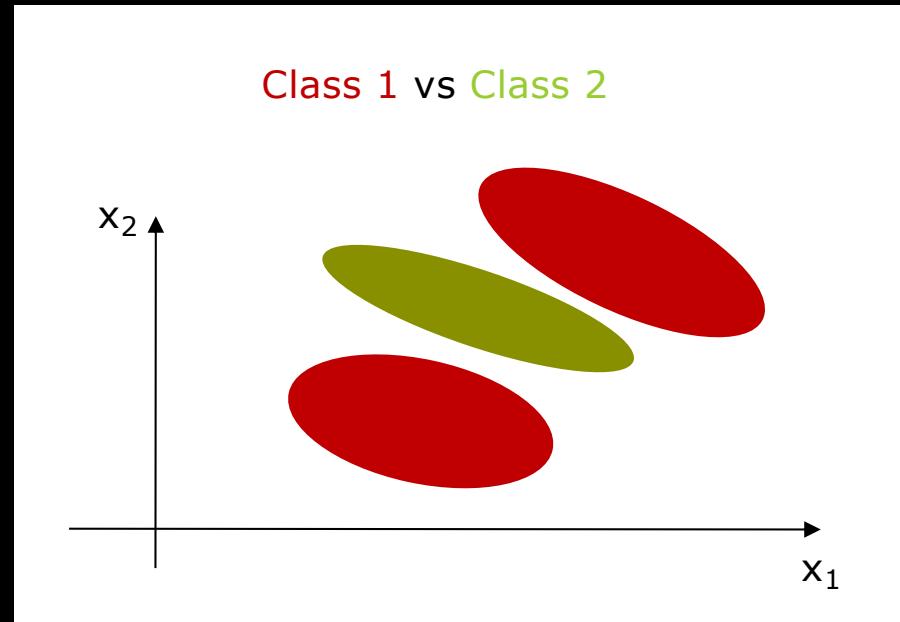


# Limitations of LDA



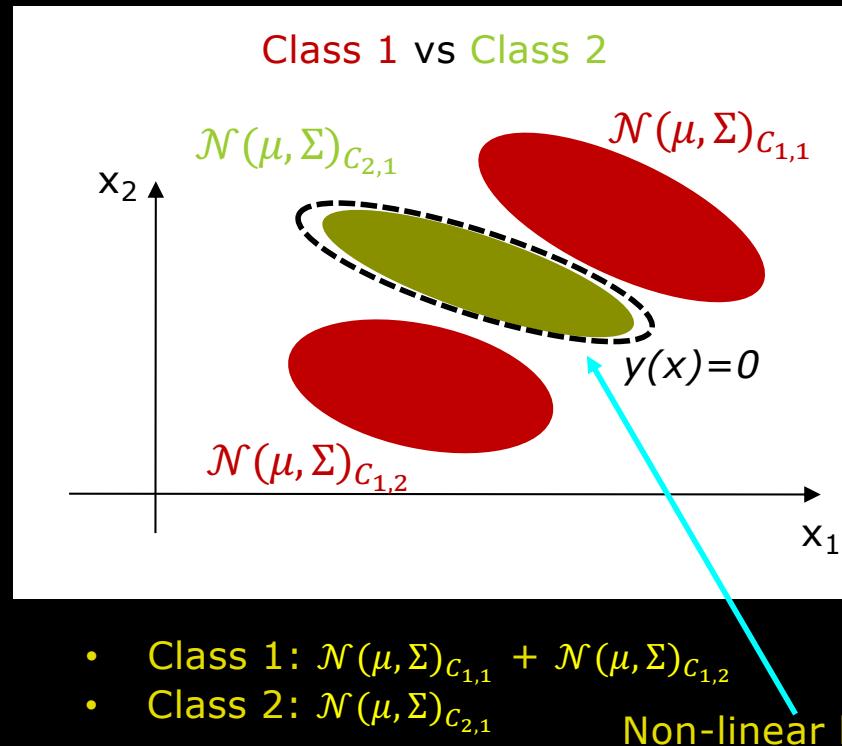
- Linear discriminant analysis (LDA)
  - Only linear hyperplanes
- Non-linear hyperplanes?
- Example:
  - I wish to make a classifier
  - Features (2D):
    - Age vs. Tissue degeneration
- Classes
  - Healthy controls vs Patient

# Limitations of LDA



- One class can be separated
  - A non-linear problem

# Non-linear Hyperplanes

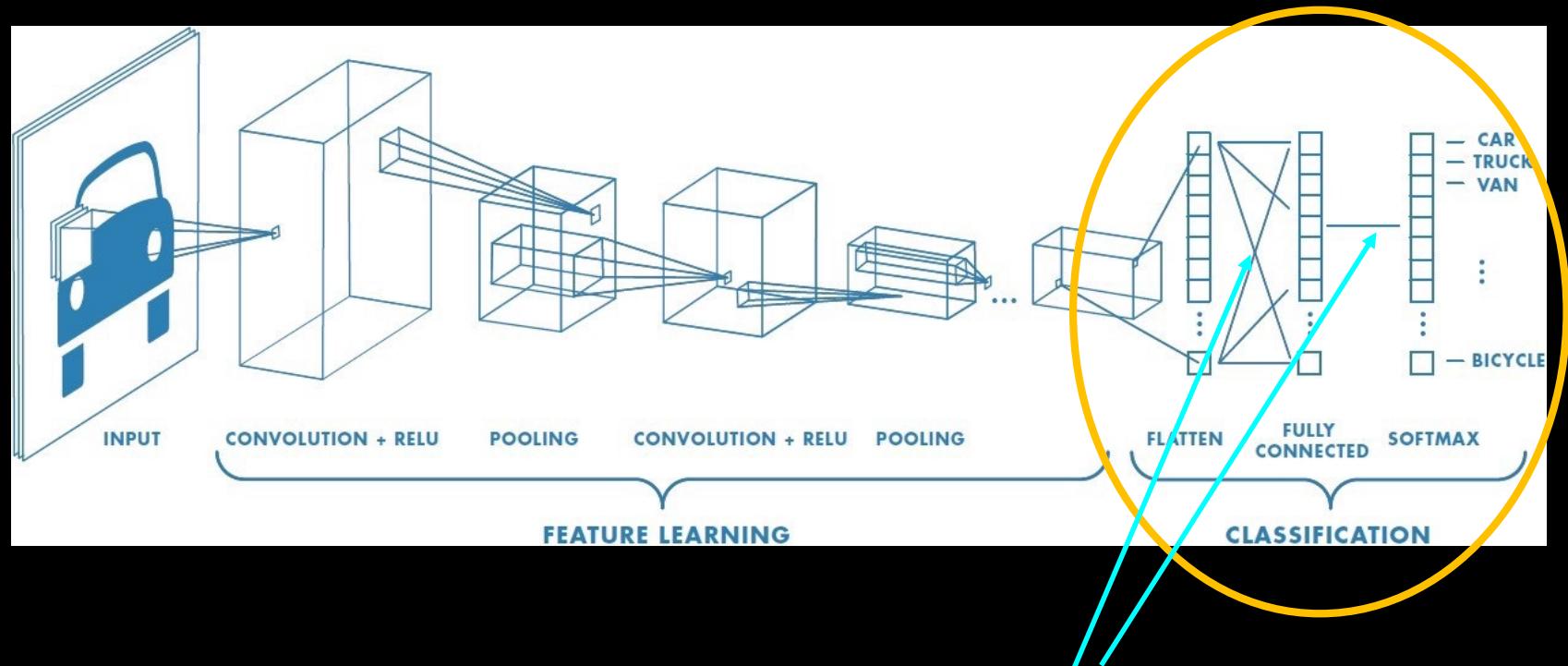


Non-linear classifiers  
(Machine learning):  
Example:

- Gaussian Mixture Model
  - Each class is modelled using a number of Gauss distributions e.g. class 1
- Again use Bayes theorem also for Gaussian Mixture Model
- Optimisation:
  - We derive  $\frac{\partial J(\mathbf{w})}{\partial \mathbf{w}}=0$  for a Gaussian mixture model
  - Iterative optimisation algorithm is used to find  $\mathbf{w}$

# Segmentation - Non-linear Hyperplanes

- Convolutional neural network and classification



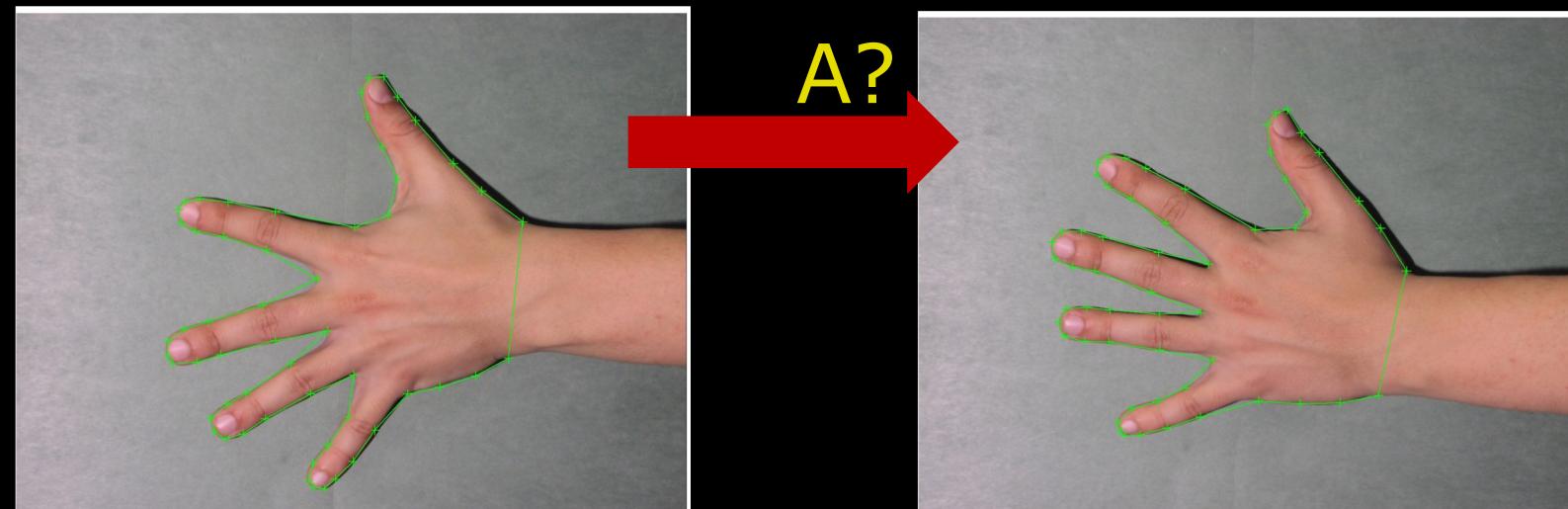
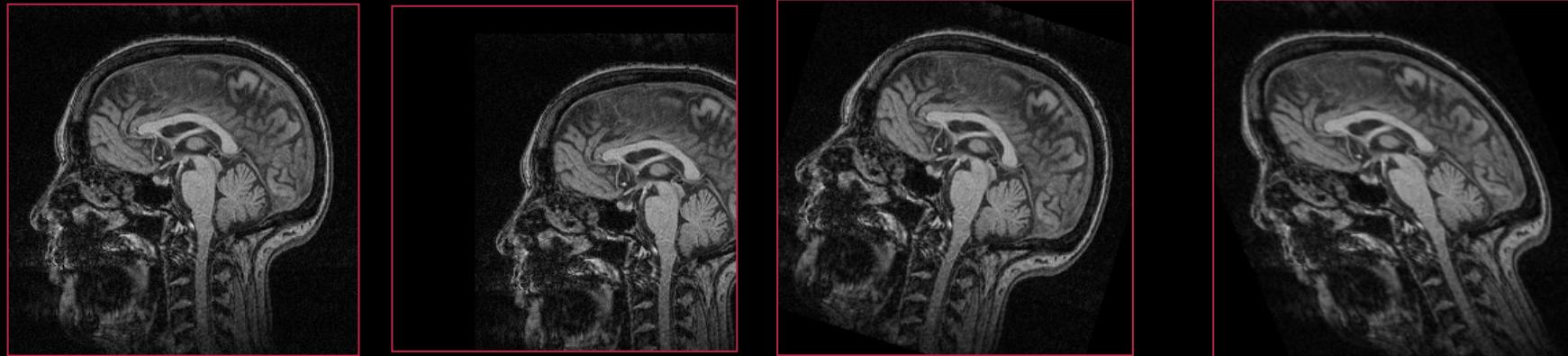
Weights can be non-linear sigmoid functions:  $y_k = \phi(x, w, w_0)$



# What did you learn today?

- Describe the concept of pixel classification
- Compute the pixel value ranges in a minimum distance classifier
- Implement and use a minimum distance classifier
- Approximate a pixel value histogram using a Gaussian distribution
- Implement and use a parametric classifier
- Decide if a minimum distance or a parametric classifier is appropriate based on the training data
- Explain the concept of Bayesian classification
- Implement and use the linear discriminant analysis (LDA) classifier
- Decide where to place a decision boundary
- Understand the use of linear vs non-line hyperplanes for segmentation

# Lecture 7 – Geometric Transformation and image registration





# Image Analysis

Tim B. Dyrby

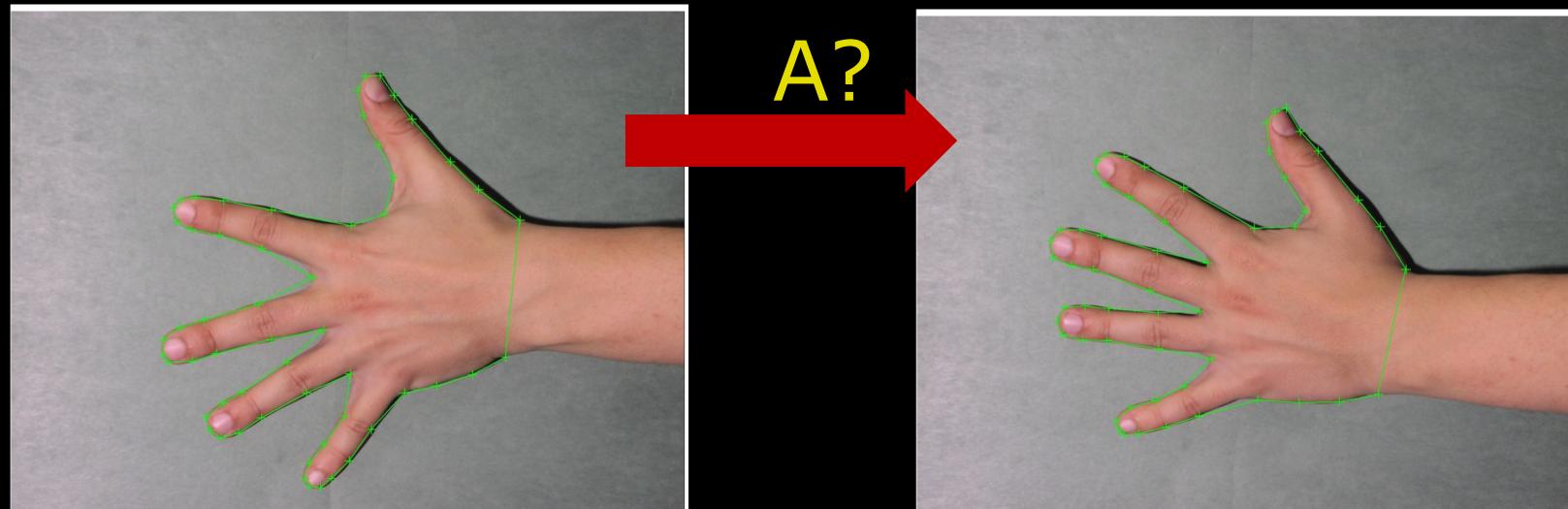
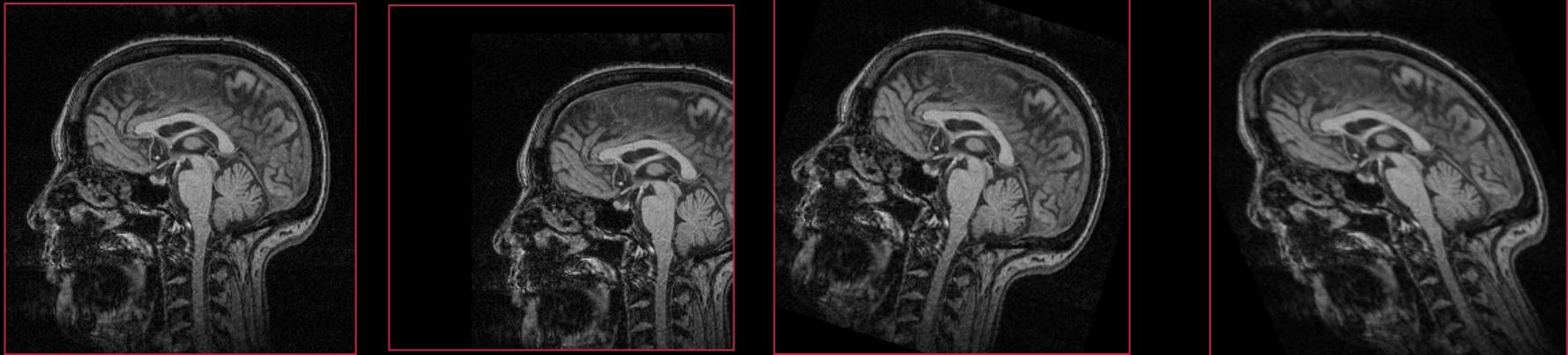
Rasmus R. Paulsen

DTU Compute

[tbdy@dtu.dk](mailto:tbdy@dtu.dk)

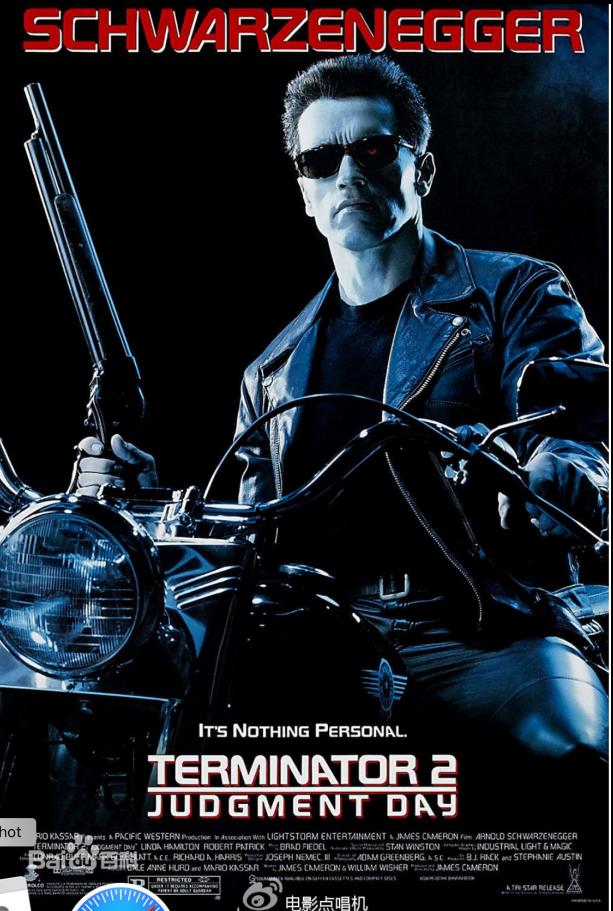
<http://www.compute.dtu.dk/courses/02502>

# Lecture 8 – Geometric Transformation and image registration



# What can you do after today?

- Construct a translation, rotation, scaling, and shearing transformation matrix of a point
  - Use transformation matrices to perform point transformations
  - Describe the difference between forward and backward mapping
  - Transform an image using backward mapping and bilinear interpolation
- 
- Describe the image registration
  - Describe the different types of landmarks
  - Annotate a set of image using anatomical landmarks
  - Describe the objective function used for landmark and joint histogram-based registration
  - Compute the optimal translation between two sets of landmarks
  - Use the rigid body transformation for image registration
  - Describe the general "pipeline" for image registration



From 1991

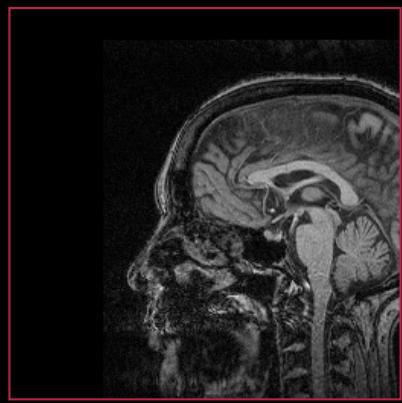
Go to [www.menti.com](http://www.menti.com) and use the code **8400 2182**  
Quiz testing: What is it that the Terminator II movie is famous for?



- 1) Arnold Schwarzenegger
- 2) Fancy new robots
- 3) Computer graphics
- 4) Time travel

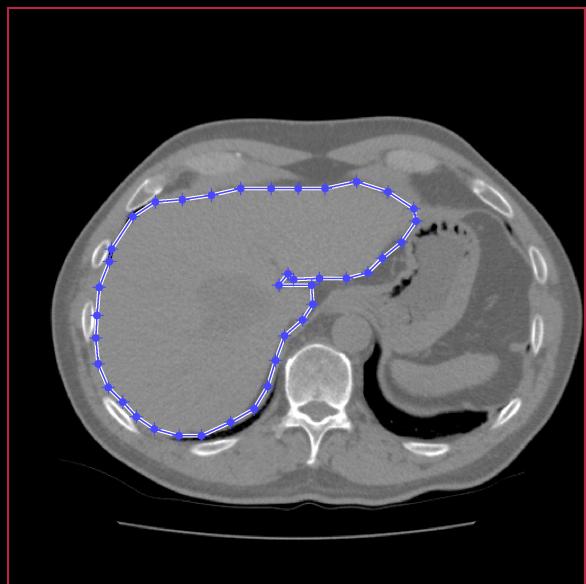
# Geometric transformation

- Moving and changing the dimensions of images
- Why do we need it?



# Change detection

- Patient imaged before and after surgery
- What are the changes in the operated organ?
- Patient cannot be placed in the exact same position in the scanner



Before surgery

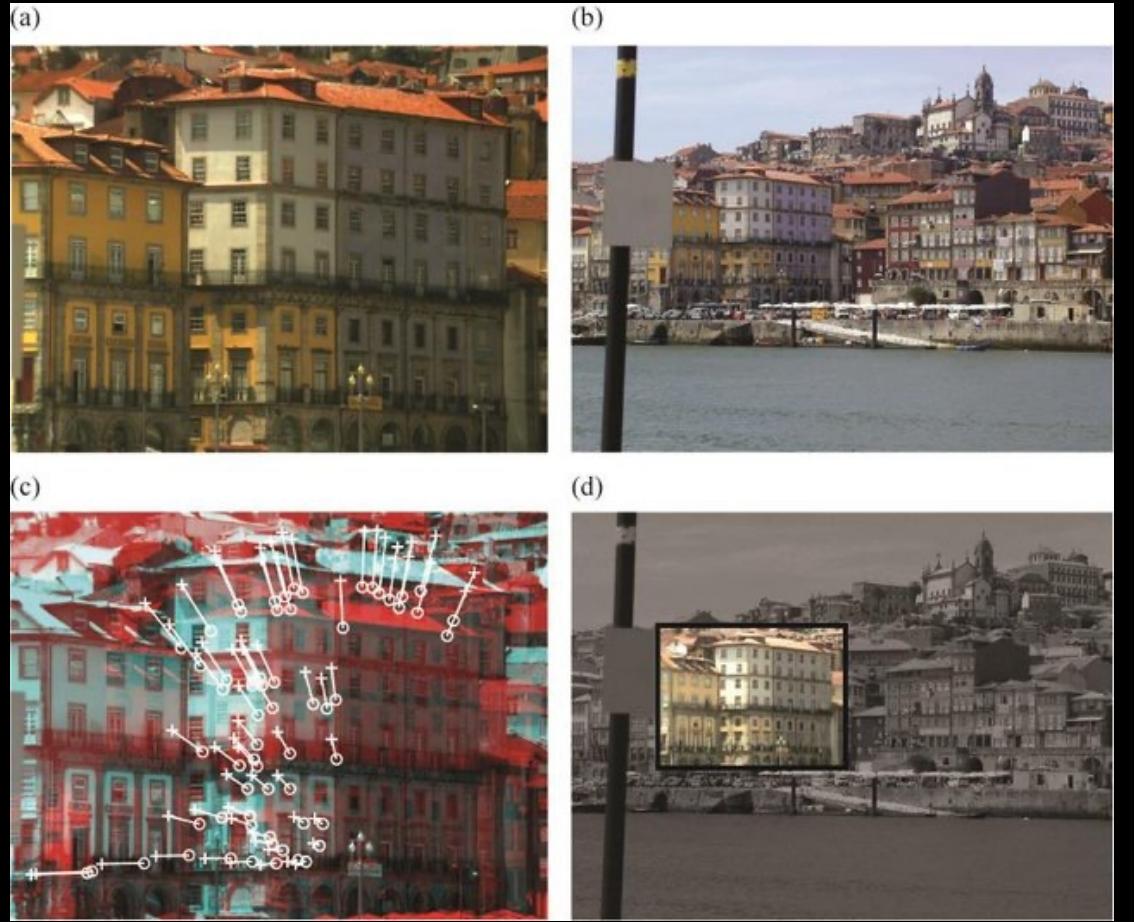


After surgery

Bachelor project: Image Guided Surgery Planning

# Similarity transform

- Objects at different distances



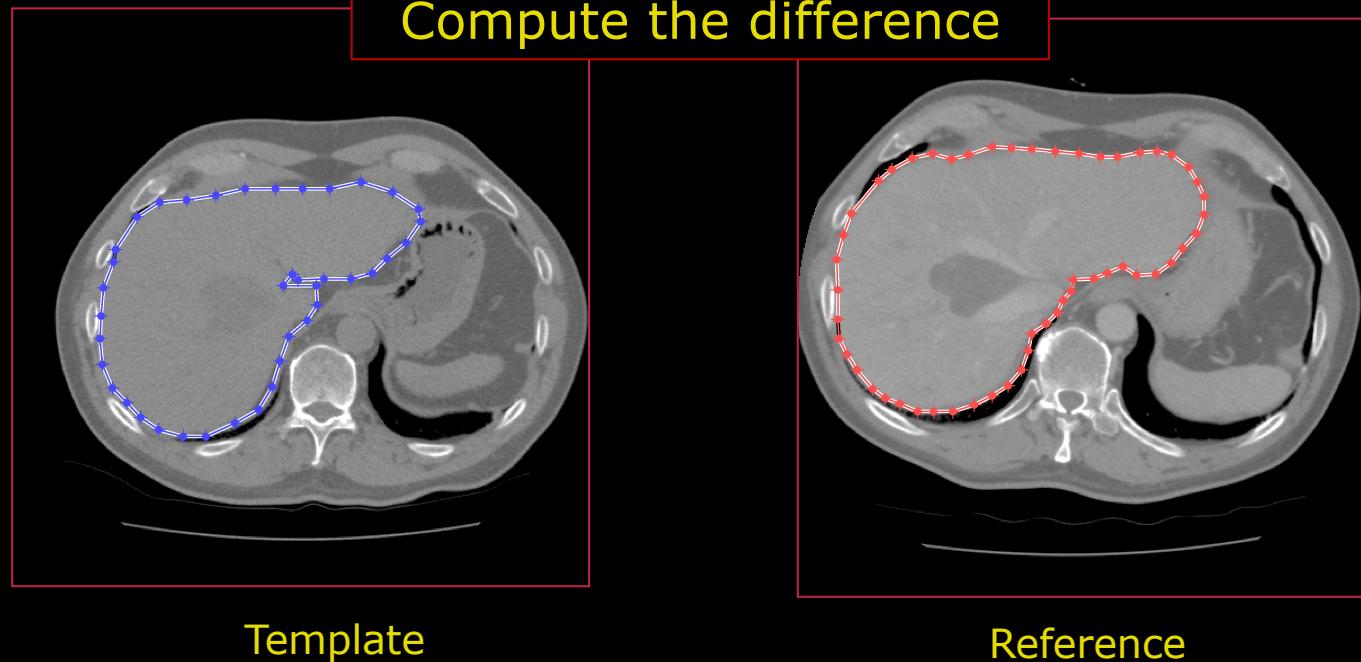
Amano et al 2016, DOI: 10.1051/matecconf/20166600024

DTU

# Image Registration

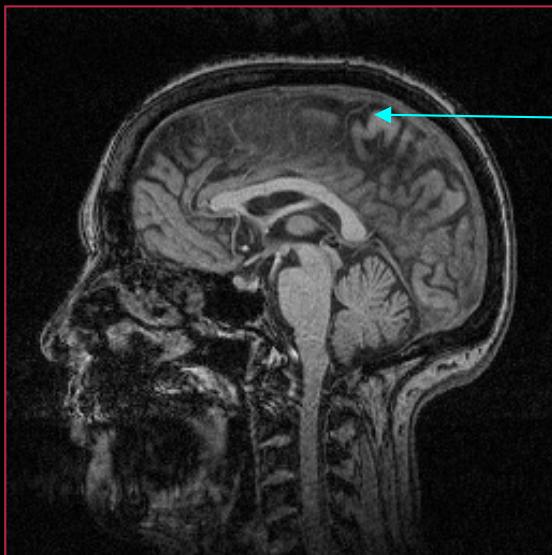
- Change one of the images so it fits with the other
- Formally
  - Template image
  - Reference image
  - Template is moved to fit the reference

Compute the difference



# Geometric Transform

- The pixel intensities are not changed
- The “pixel values” just change positions



Same value  
Different place



# Different transformations

- Translation
- Rotation
- Scaling
- Shearing



## ■ Advanced transformations

From Terminator 2 movie: Non-linear image transformation



# Translation

- The image is shifted – both vertically and horizontally

$$\begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} 60 \\ 20 \end{bmatrix}$$



# Rotation

- The image is rotated around the centre or the upper left corner
- Remember to use degrees and radians correctly
  - Python uses radians
  - Degrees easier for us humans

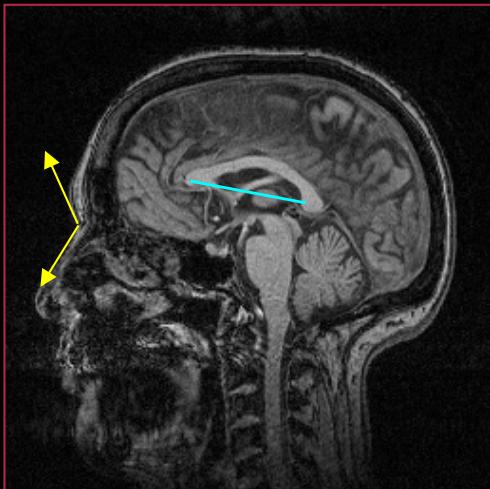


$$\theta = 15^\circ$$



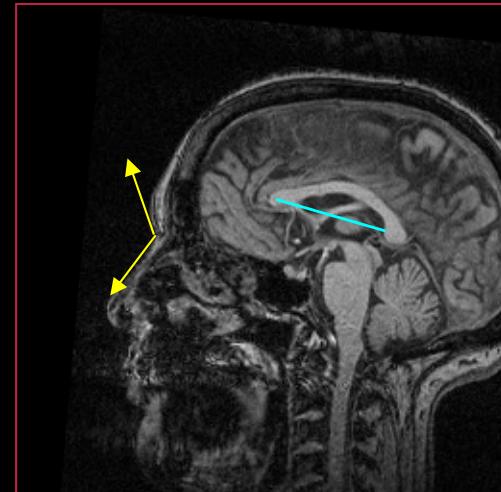
# Rigid body transformation

- Translation and rotation
- Rigid body
- Angles and *distances* are kept



$$\begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} 60 \\ 20 \end{bmatrix}$$

$$\theta = 5^\circ$$

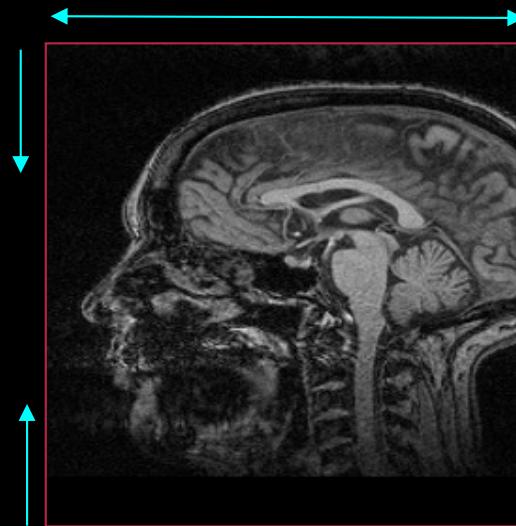


# Scaling

- The size of the image is changed
- Scale factors
  - X-scale factor  $S_x$
  - Y-scale factor  $S_y$
- Uniform scaling:  $S_x = S_y$

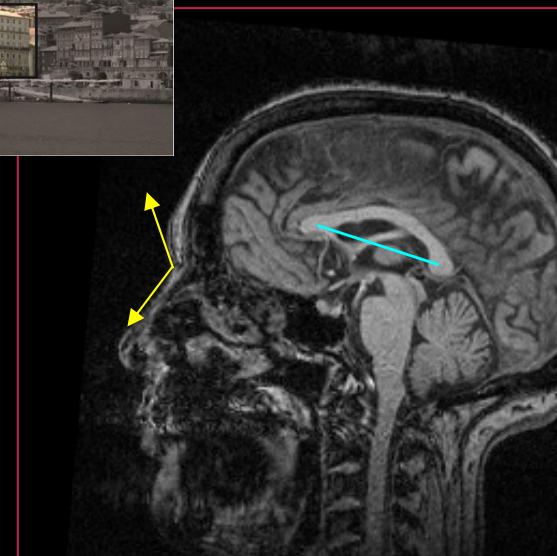
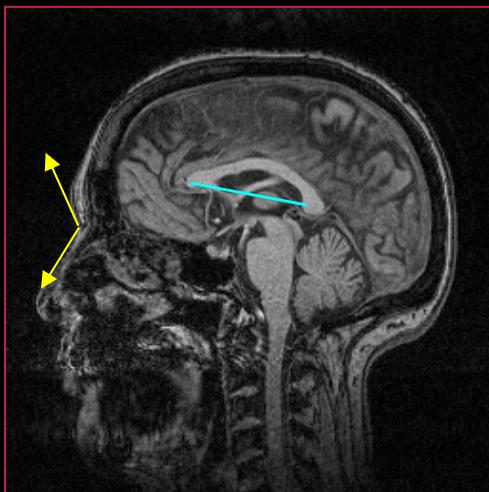
$$S_x = 1.2$$

$$S_y = 0.9$$



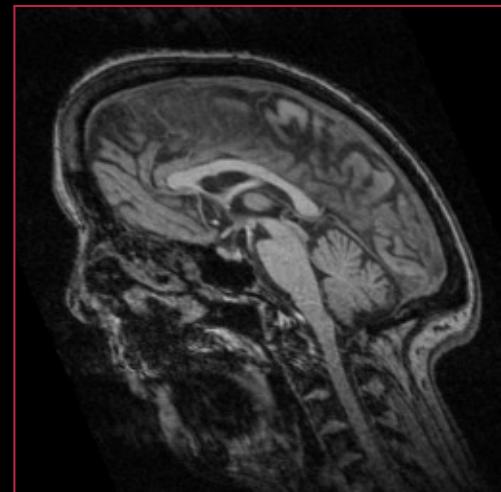
# Similarity transformation

- Translation, and uniform scaling
- Angles are kept
- Distances change

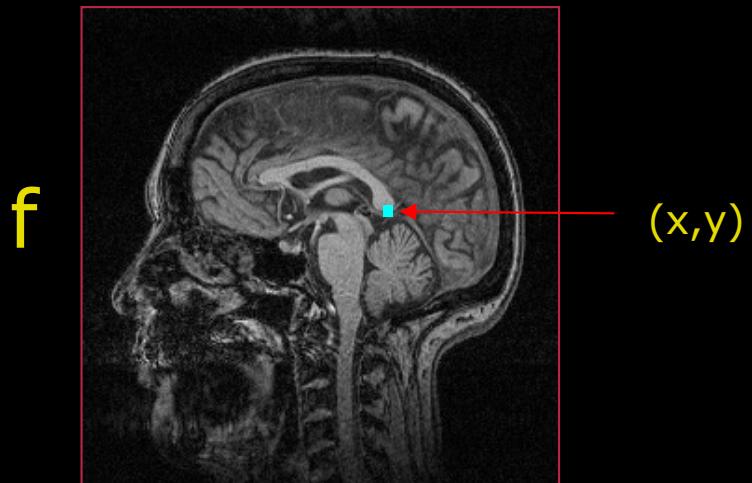


# Shearing

- Pixel shifted horizontally or/and vertically
- Shearing factors
  - X-shear factor  $B_x$
  - Y-shear factor  $B_y$
- Is less used than translation, rotation, and scaling



# Transformation Mathematics



- Transformation of *positions*
- Structure found at position  $(x, y)$  in the input image  $f$
- Now at position  $(x', y')$  in output image  $g$
- A *mapping function* is needed

$$x' = A_x(x, y)$$
$$y' = A_y(x, y)$$

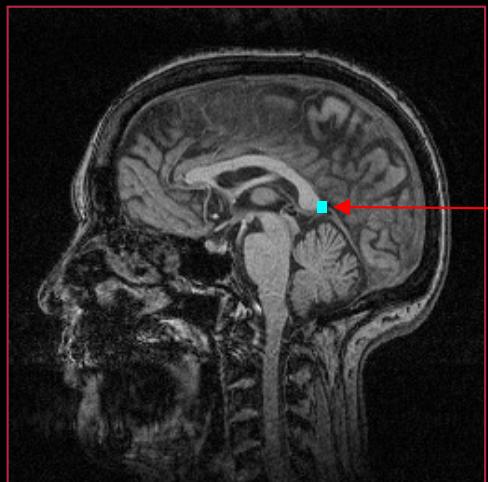
Depends on both  $x$  and  $y$ !

# Translation mathematics

- The image is shifted – both vertically and horizontally

$$x' = x + \Delta x$$

$$y' = y + \Delta y$$



$(x, y)$

$$\Delta x = 60$$

$$\Delta y = 20$$



$(x', y')$



# Matrix notation

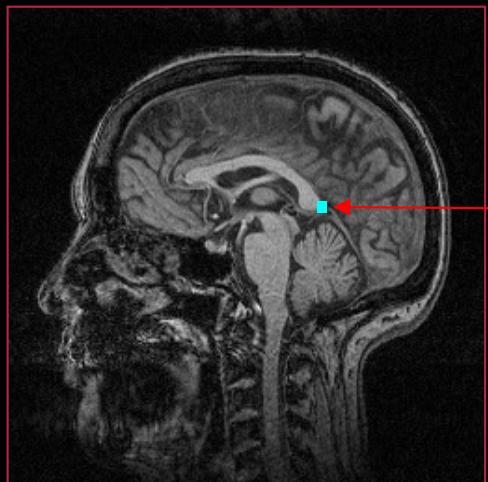
- Coordinates in column matrix format

$$\begin{bmatrix} x \\ y \end{bmatrix}$$

# Translation mathematics in matrix notation

- The image is shifted – both vertically and horizontally

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}$$



$$\begin{bmatrix} x \\ y \end{bmatrix}$$

$$\Delta x = 60$$

$$\Delta y = 20$$



$$\begin{bmatrix} x' \\ y' \end{bmatrix}$$

# Scaling

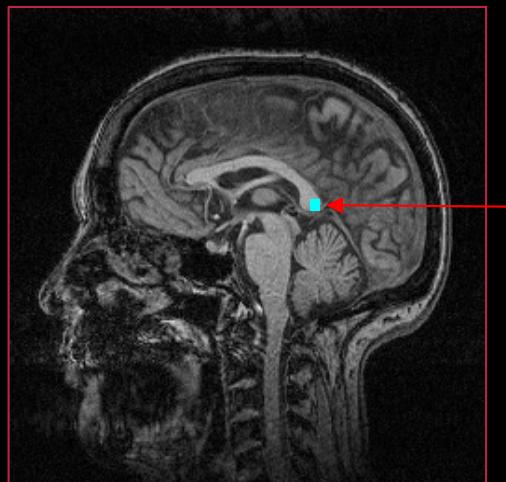
- The size of the image is changed

- Scale factors

- X-scale factor  $S_x$
- Y-scale factor  $S_y$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix}$$

- Uniform scaling:  $S_x = S_y$



$$\begin{bmatrix} x \\ y \end{bmatrix} \quad S_y = 0.9$$



$$S_x = 1.2$$



# Matrix multiplication details

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix}$$

Is equal to:

$$x' = x \cdot S_x$$

$$y' = y \cdot S_y$$



# Transformation matrix

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix}$$

Can be written as

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \mathbf{A} \cdot \begin{bmatrix} x \\ y \end{bmatrix}$$

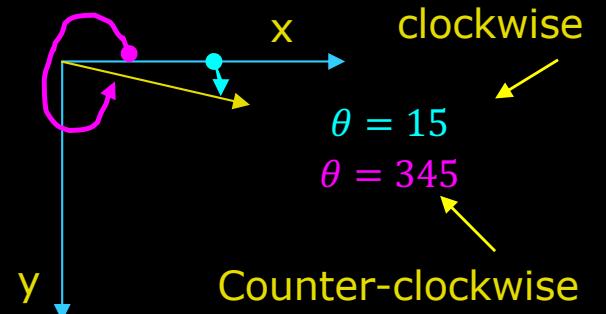
Where

$$\mathbf{A} = \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix}$$

is a *transformation matrix*

# Rotation

- A rotation matrix is used
- Is it a clockwise rotation or counter-clockwise?

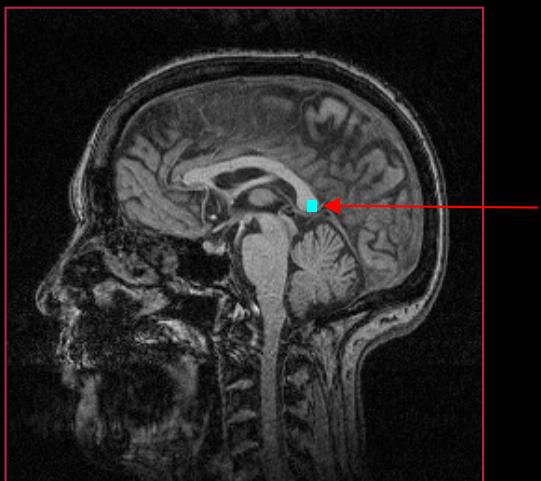


Clockwise

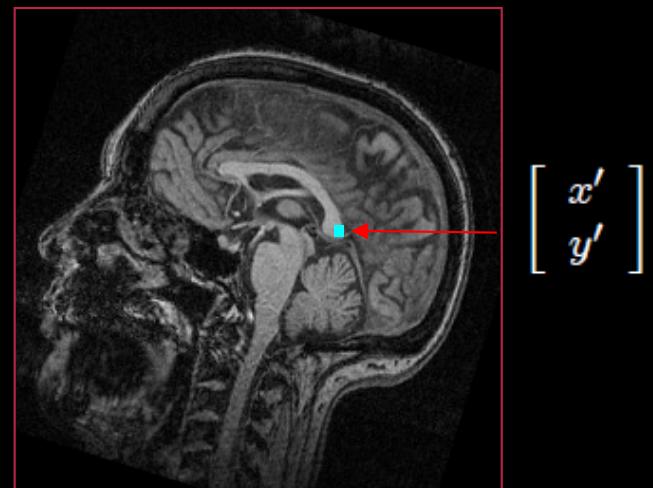
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos \theta & \sin \theta \\ -\sin \theta & \cos \theta \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix}$$

Counter-clockwise

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



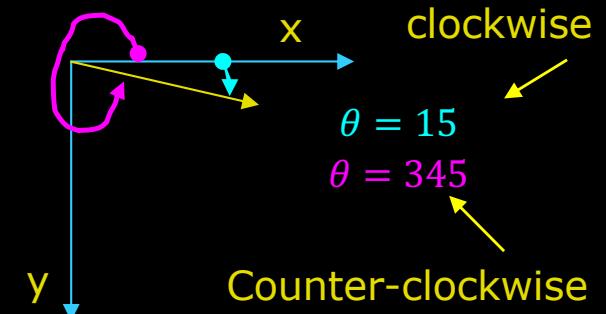
$$\begin{bmatrix} x \\ y \end{bmatrix}$$



$$\begin{bmatrix} x' \\ y' \end{bmatrix}$$

# Rotation

- A rotation matrix is used
- Is it a clockwise rotation or counter-clockwise?
  - In this course we use a counter-clockwise rotation

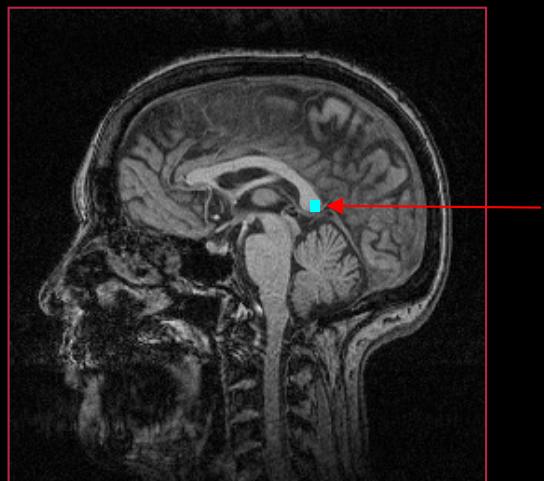


$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos \theta & \sin \theta \\ -\sin \theta & \cos \theta \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix}$$

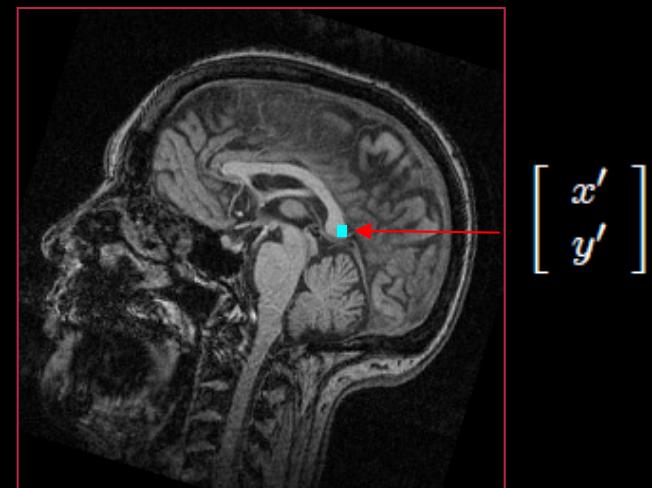
**Clockwise**

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

**Counter-clockwise**



$$\theta = 345^\circ$$

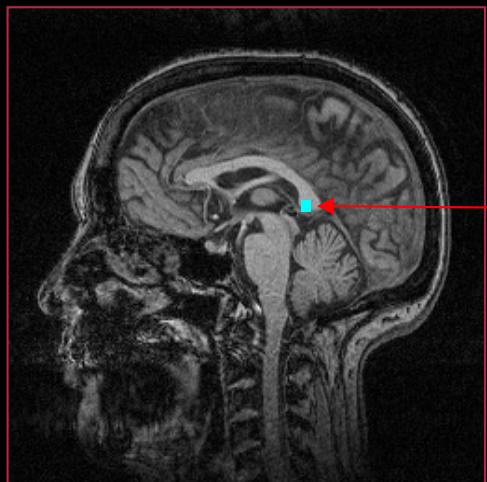


# Shearing

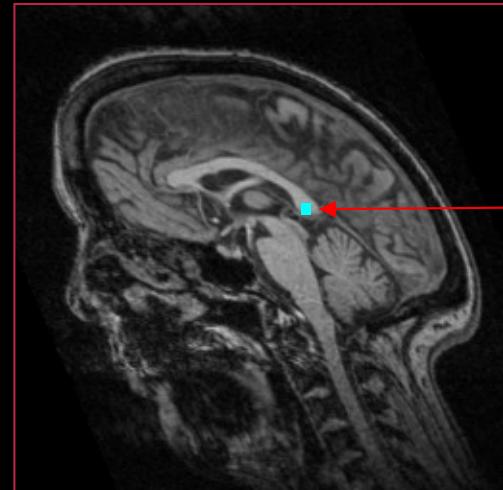
- Pixel shifted horizontally or/and vertically

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1 & B_x \\ B_Y & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix}$$

New x value  
depends on x  
and y



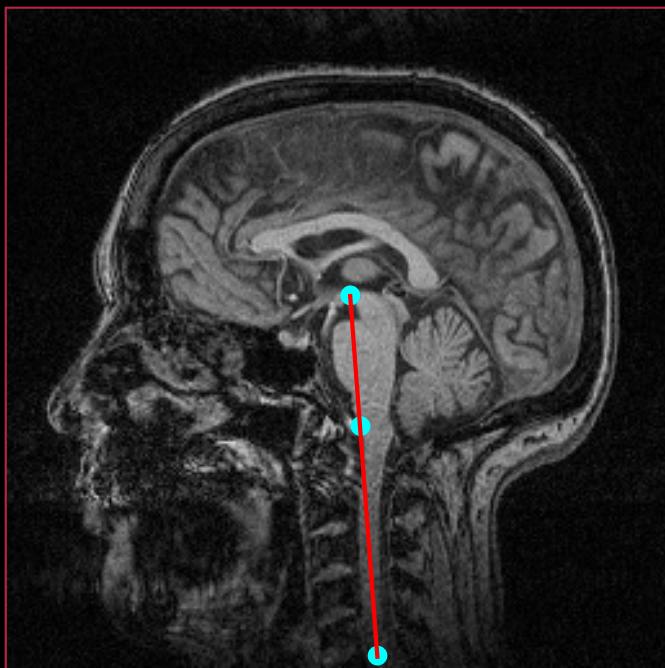
$$\cdot \begin{bmatrix} x \\ y \end{bmatrix}$$



$$\begin{bmatrix} x' \\ y' \end{bmatrix}$$

# Affine transformation

- The collinearity relation between points, i.e., three points which lie on a line continue to be collinear after the transformation



# Combining transformations

Scaling  $S_x = S_y = 1.10$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix}$$

Rotation  $\theta = 5^\circ$

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

- Suppose you first want to rotate by 5 degrees and then scale by 10%

How do we combine  
the transformations?

# Combining transformations

- Combination is done by matrix multiplication

Scaling

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix}$$

Rotation

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

Combined

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix} \cdot \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix}$$

# Combining transformations

## ■ Compact notation

Scaling 
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix} = \mathbf{A}_S \cdot \begin{bmatrix} x \\ y \end{bmatrix}$$

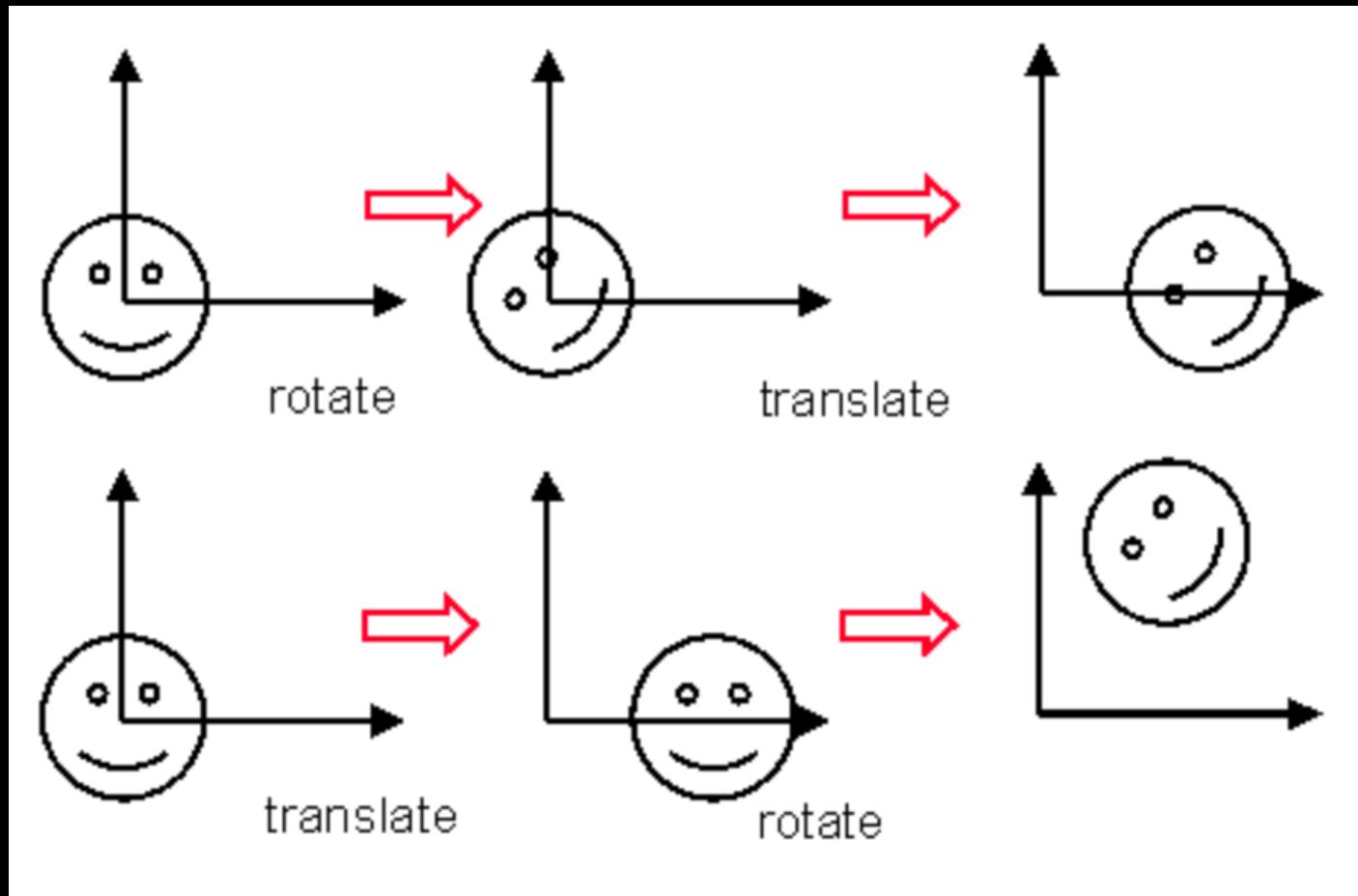
Rotation 
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix} = \mathbf{A}_R \cdot \begin{bmatrix} x \\ y \end{bmatrix}$$

Combined 
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \mathbf{A}_S \cdot \mathbf{A}_R \cdot \begin{bmatrix} x \\ y \end{bmatrix}$$

Remember: The order of matrix multiplications matters!

# Compositions of transformations

- Rigid body transformation: translation and rotation
- Composition of transformations matters = product of metrics
- Remember: order matters!





# Quiz 1: Combining transforms

The point  $(x,y)=(5,6)$  is transformed. First with:

$$\begin{bmatrix} 2 & 0 \\ 0 & 3 \end{bmatrix} \quad (2)$$

and then with:

$$\begin{bmatrix} 0.9239 & 0.3827 \\ -0.3827 & 0.9239 \end{bmatrix} \quad (3)$$

The result is:

1. (16.12, 12.8)
2. (2.35, 20.46)
3. (11.3, 1.21)
4. (-1.2, 3.13)
5. (-30.8, 24.21)

Solution:

$$\begin{bmatrix} 2 & 0 \\ 0 & 3 \end{bmatrix} \begin{bmatrix} 5 \\ 6 \end{bmatrix} = \begin{bmatrix} 2 * 5 + 0 * 6 \\ 0 * 5 + 3 * 6 \end{bmatrix} = \begin{bmatrix} 10 \\ 18 \end{bmatrix}$$

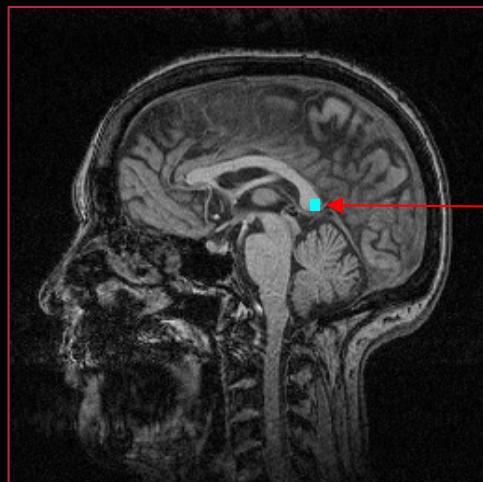
$$\begin{bmatrix} 0.9239 & 0.3827 \\ -0.3827 & 0.9239 \end{bmatrix} \begin{bmatrix} 10 \\ 18 \end{bmatrix} = \begin{bmatrix} 0.9239 * 10 + 0.3827 * 18 \\ -0.3827 * 10 + 0.9239 * 18 \end{bmatrix} = \begin{bmatrix} 16.12 \\ 12.8 \end{bmatrix}$$

# What do we have now?

- We can pick a position in the input image  $f$  and find it in the output image  $g$

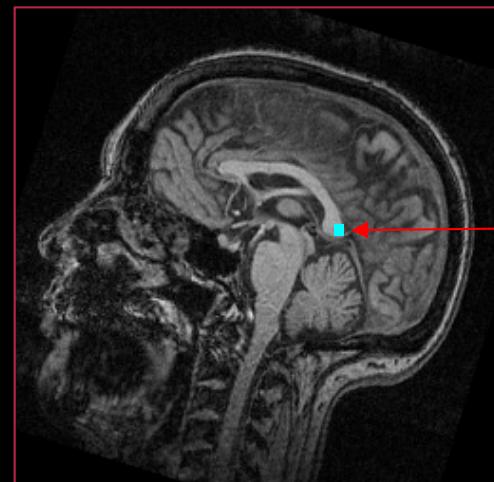
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \mathbf{A} \cdot \begin{bmatrix} x \\ y \end{bmatrix}$$

We can transfer one pixel – what about the whole image?



$f$

$$\begin{bmatrix} x \\ y \end{bmatrix}$$



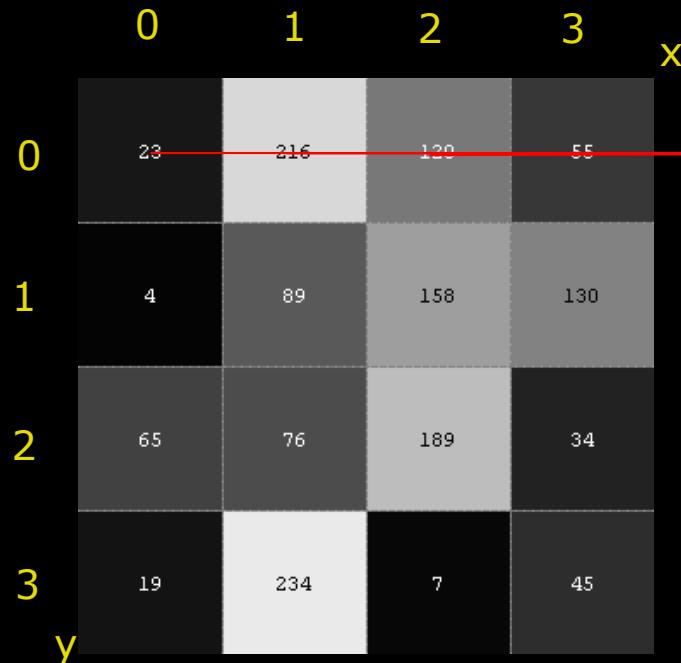
$g$

$$\begin{bmatrix} x' \\ y' \end{bmatrix}$$

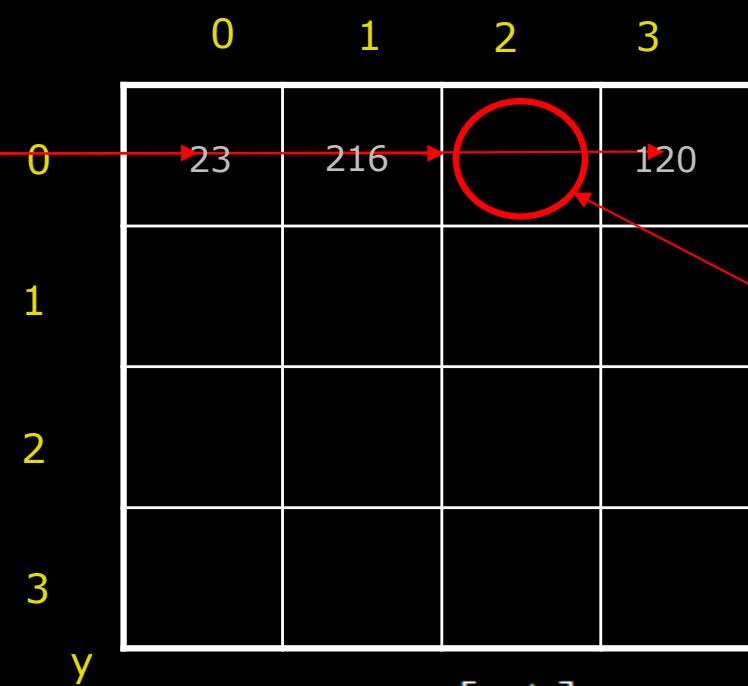
# Solution 1 : Input-to-output

- Run through all pixel in input image
- Find position in output image and set output pixel value

Scaling example  $\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1.5 & 0 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix}$



$$f \begin{bmatrix} x \\ y \end{bmatrix}$$



$$g \begin{bmatrix} x' \\ y' \end{bmatrix}$$

Missing  
value!



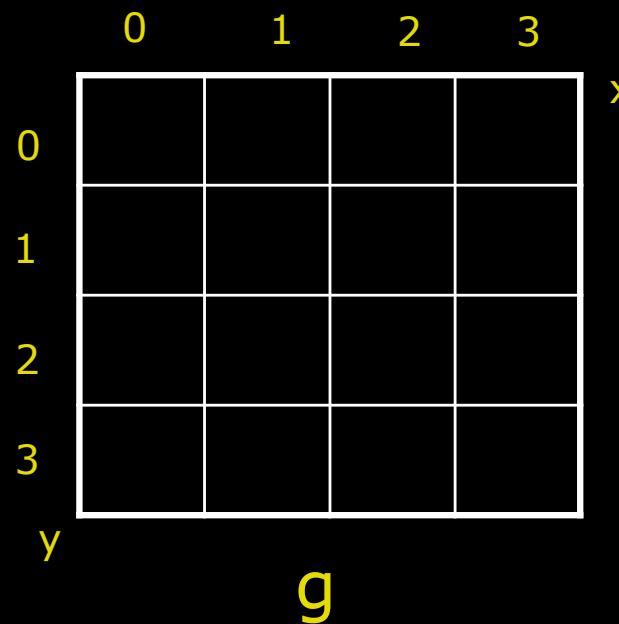
# Input-to-Output

- The input to output transform is not good!
- It creates holes and other nasty looking stuff
- What do we do now?

# Some observations

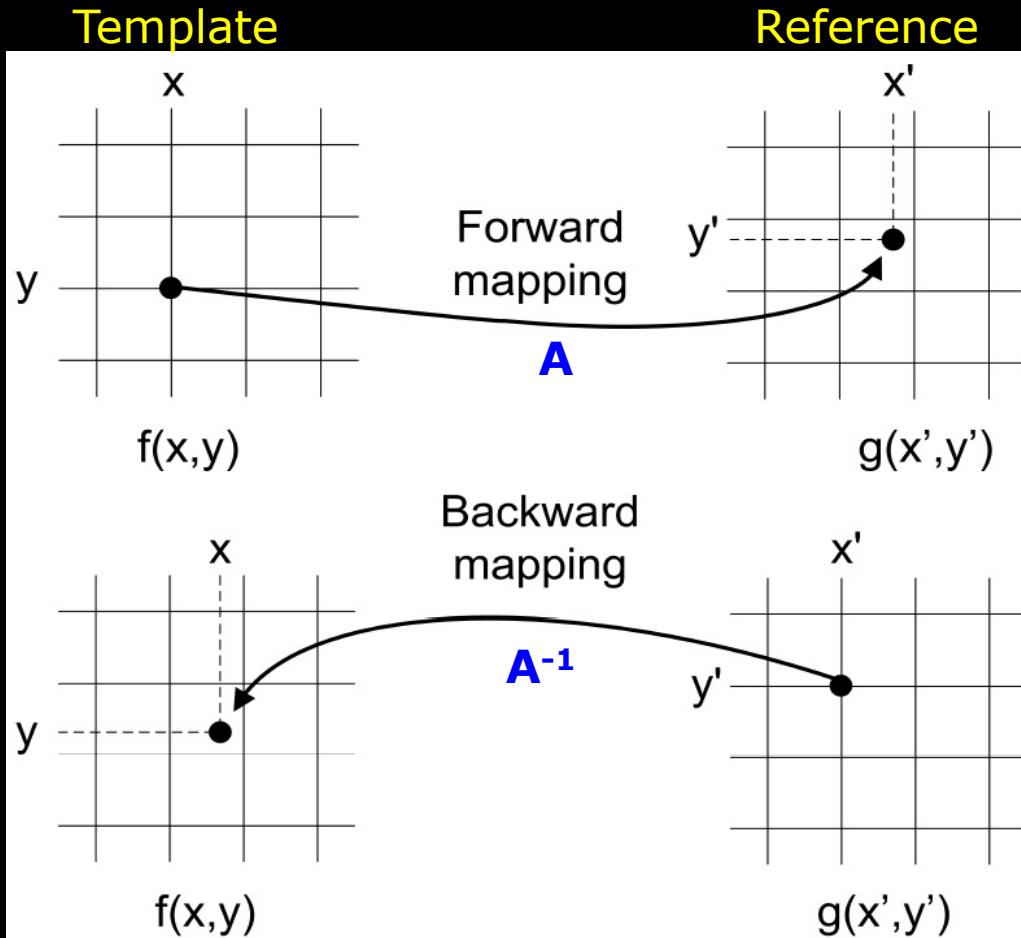
- We want to fill all the pixels in the output image
  - Not just the pixels that are “hit” by the pixels in the input image
- Run through all pixels in the output image?
  - Pick the relevant pixels in the input image?

We need to go “backwards”  
From the output to the input



# Forward vs. Backward mapping

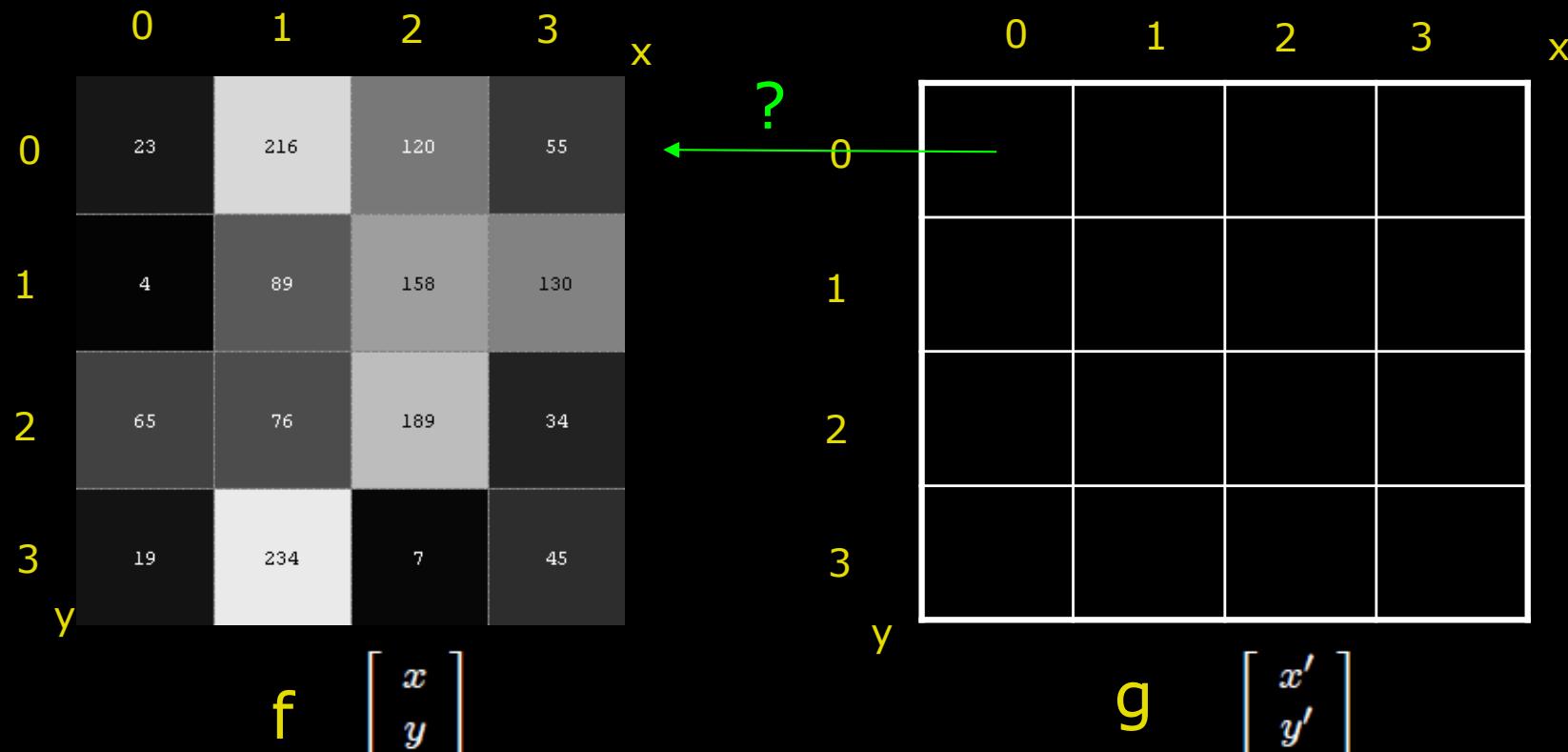
- In a nutshell
  - Going backward we need to invert the transformation



# Inverse transformation

- We want to go from the output to the input

Scaling example  $\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1.5 & 0 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix}$   $\xrightarrow{\text{inverse}}$   $\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 1/1.5 & 0 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x' \\ y' \end{bmatrix}$

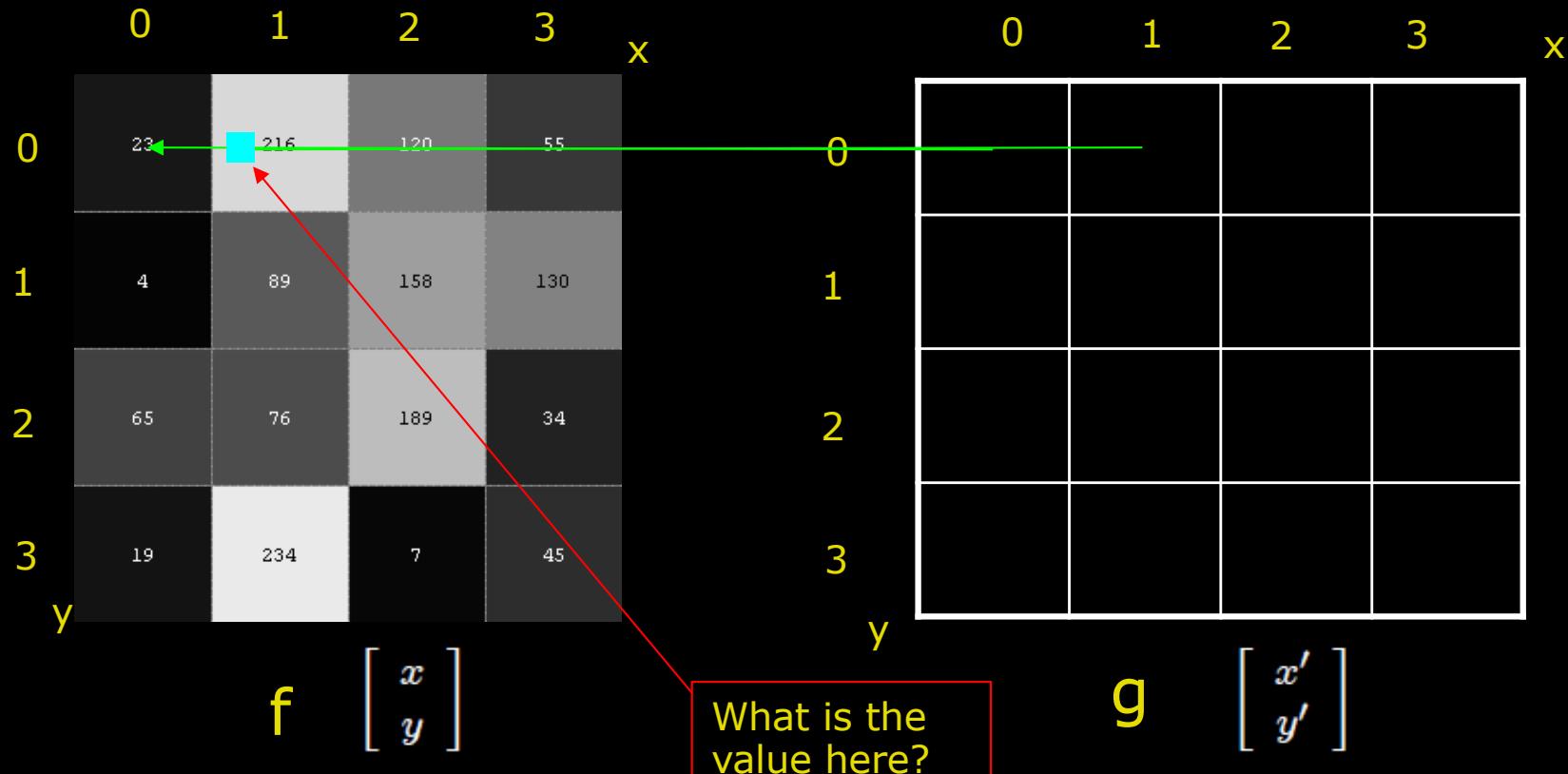


# Output-to-input transformation

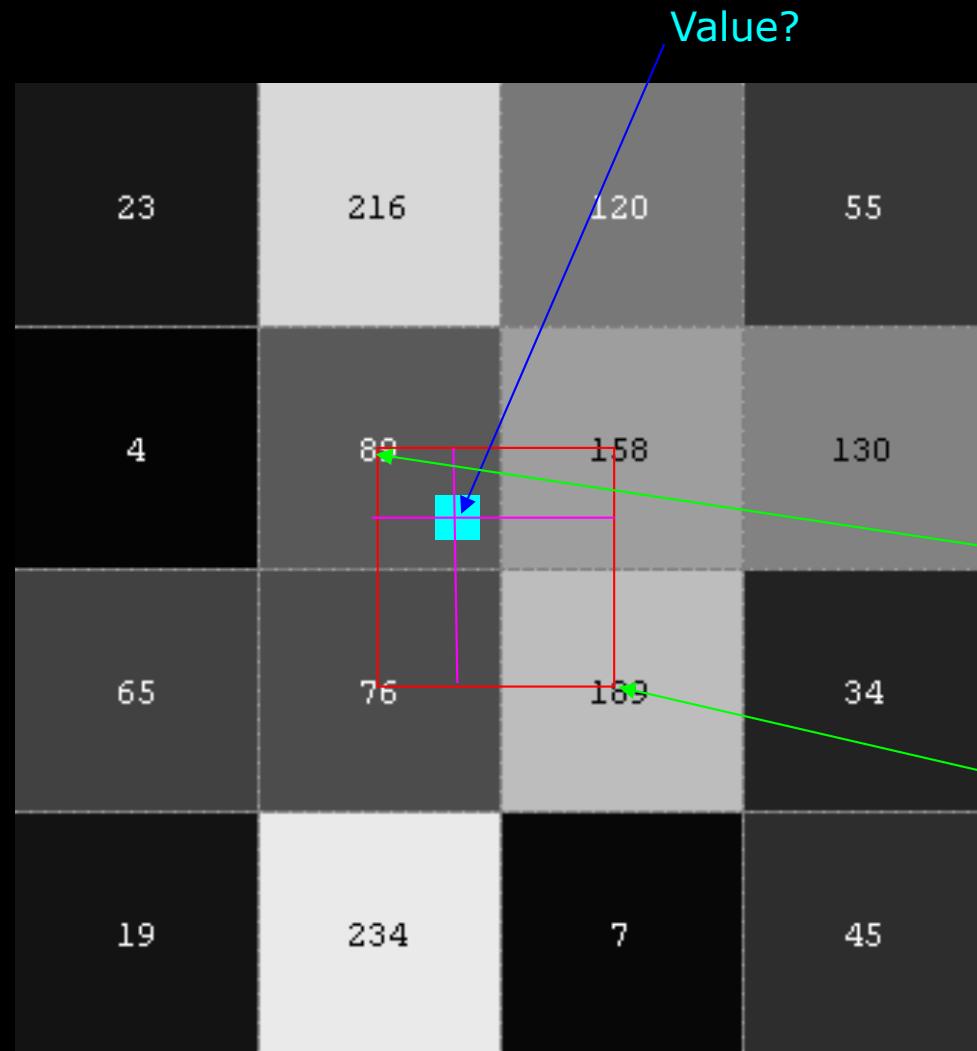
## *Backward mapping*

- Run through all pixel in output image
- Find position in input image and *get the value*

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 1/1.5 & 0 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x' \\ y' \end{bmatrix}$$



# Bilinear Interpolation



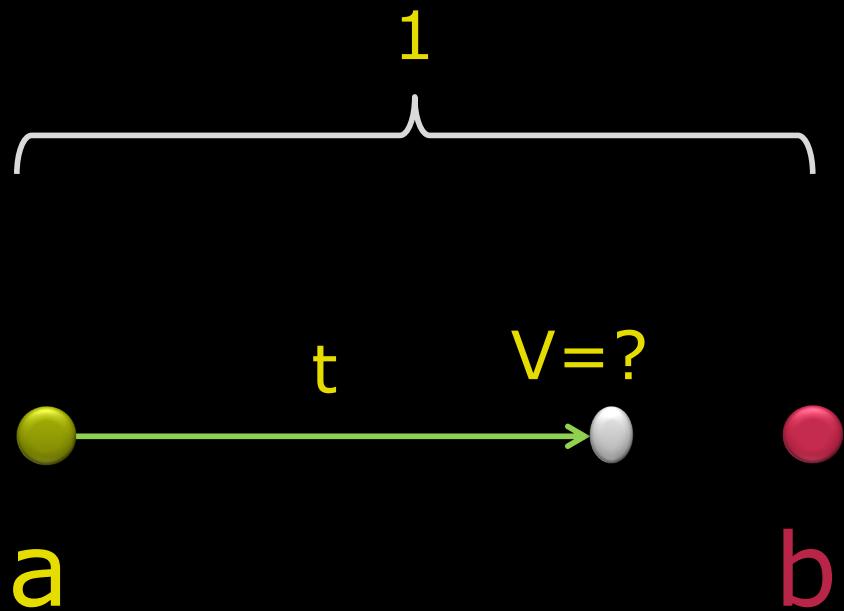
- The value is calculated from 4 neighbours
- The value is based on the distance to the neighbours

Lots of 89!

Not so much 189

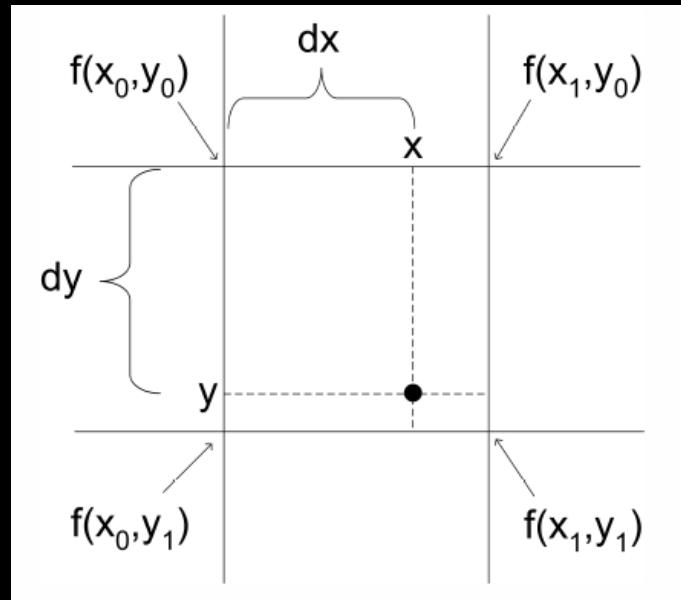
# Linear Interpolation (1D)

$$v = tb + (1 - t)a$$



# Bilinear interpolation (2D)

$$\begin{aligned} g(x', y') = & f(x_0, y_0) \cdot (1 - dx)(1 - dy) + \\ & f(x_1, y_0) \cdot (dx)(1 - dy) + \\ & f(x_0, y_1) \cdot (1 - dx)(dy) + \\ & f(x_1, y_1) \cdot (dx \cdot dy) , \end{aligned}$$





# Quiz 2: Bilinear interpolation

Solution:

Distance between grid points is 1

hence:  $dx=0.1$  and  $dy=0.8$

Do the interpolation (see previous slide)

$$g(173.1, 57.8) =$$

$$110 * (1-0.1) * (1-0.8) +$$

$$140 * (0.1) * (1-0.8) +$$

$$156 * (1-0.1) * (0.8) +$$

$$101 * (0.1) * (0.8)$$

$$= 143$$

Bilinear interpolation is used to create a line profile from an image. In a given point  $(x,y) = (173.1, 57.8)$ , the four nearest pixels are:

x	y	værdi
173	57	110
174	57	140
173	58	156
174	58	101

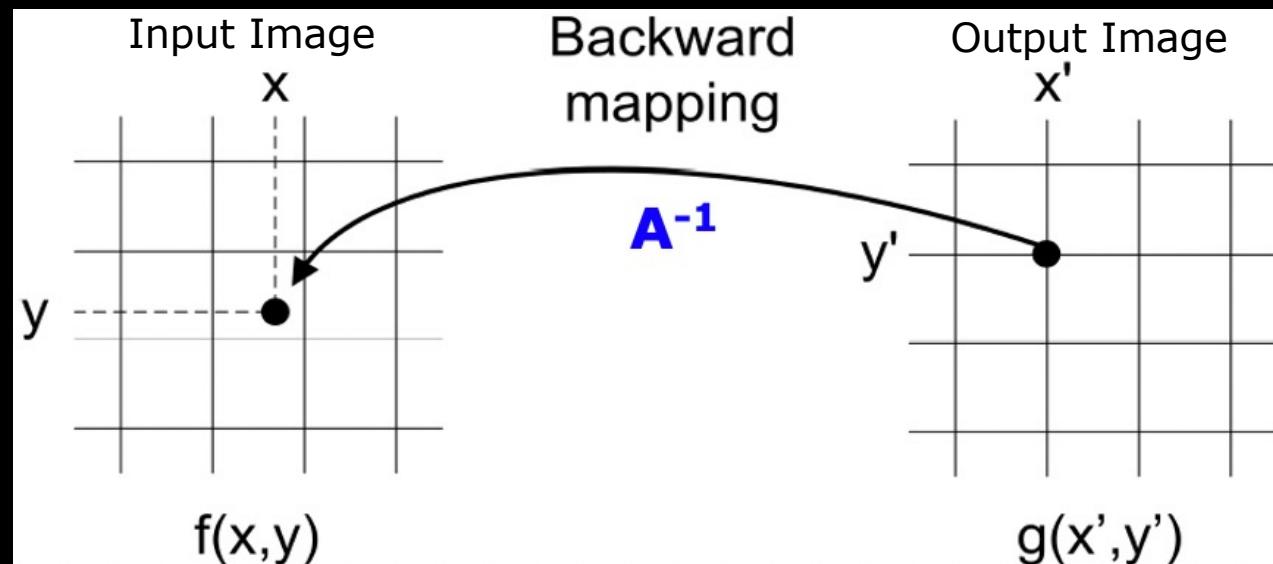
What is the interpolated value in the point:

1. 131
2. 143
3. 128
4. 151
5. 139

# Output-to-input transformation

## *Backward mapping*

- Run through all the pixel in the output image
- Use the inverse transformation to find the position in the input image
- Use bilinear interpolation to calculate the value
- Put the value in the output image



# Inverse transformation

Scaling

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 1.5 & 0 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix}$$

- We can calculate the inverse transformation for the scaling
- What about the others?

Inverse

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 1/1.5 & 0 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x' \\ y' \end{bmatrix}$$



# General inverse transformation

Affine transformation

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \mathbf{A} \cdot \begin{bmatrix} x \\ y \end{bmatrix}$$

- The transformation is expressed as a transformation matrix  $\mathbf{A}$
- The *matrix inverse* of  $\mathbf{A}$  gives the inverse transformation

Inverse transformation

$$\begin{bmatrix} x \\ y \end{bmatrix} = \mathbf{A}^{-1} \cdot \begin{bmatrix} x' \\ y' \end{bmatrix}$$

Where

$$\mathbf{A}^{-1} \cdot \mathbf{A} = \mathbf{I}$$



# Quiz 3: Transformation

The point  $(x,y) = (45, 23)$  is transformed using:

$$\begin{bmatrix} 0.5 & 2 \\ 2 & 0.8 \end{bmatrix} \quad (1)$$

And the result is translated with  $(-15, 20)$ . The result is:

Solution:

$$(x',y') = \begin{bmatrix} -15 \\ 20 \end{bmatrix} + \begin{bmatrix} 0.5 & 2 \\ 2 & 0.8 \end{bmatrix} \begin{bmatrix} 45 \\ 23 \end{bmatrix}$$

$$= \begin{bmatrix} -15 \\ 20 \end{bmatrix} + \begin{bmatrix} 68.5 \\ 108.4 \end{bmatrix}$$

$$= \begin{bmatrix} 53.5 \\ 128.4 \end{bmatrix}$$

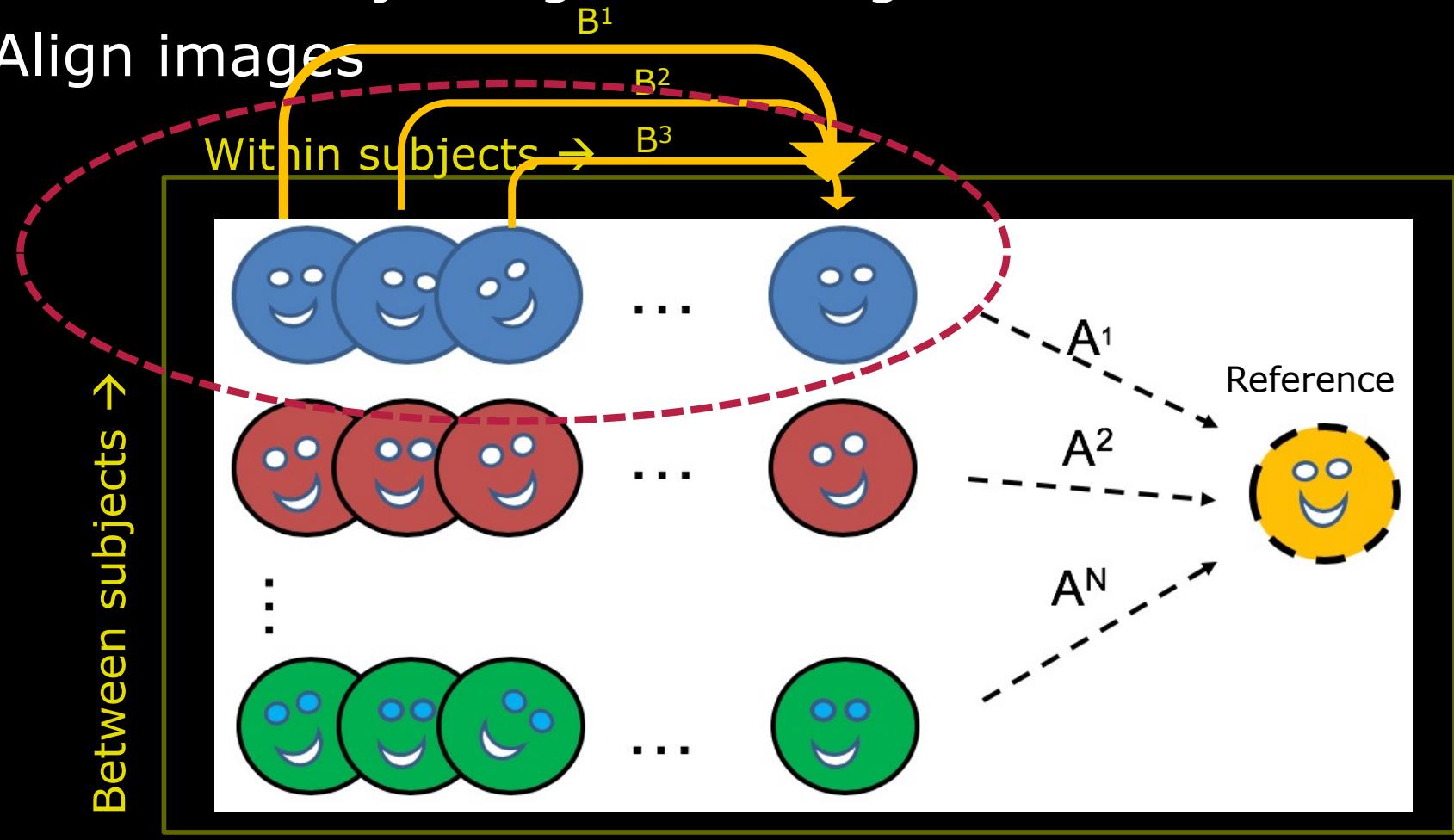
1. (53.5, 128.4)
2. (3.4, -10.3)
3. (45.3, 80.2)
4. (150.8, 32.4)
5. (-20.5, 22.6)



# Image Registration

# Image Registration

- The act of adjusting something to match a standard
- Align images



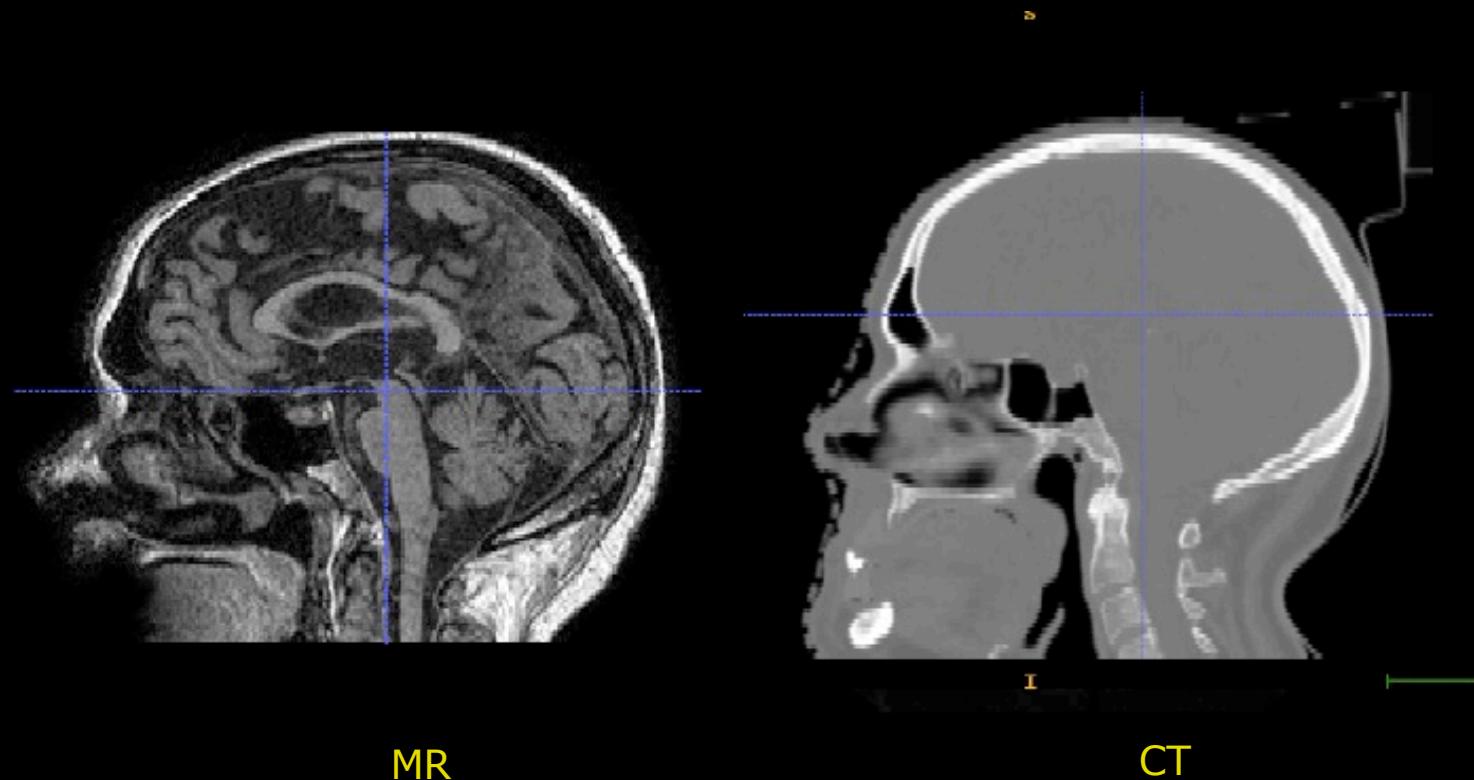


# Image registration

- Monitoring of change in the individual
- Fusion of information from different sources in a meaningful way
- Comparison of one subject with others
- Comparison of groups with others
- Comparing with an atlas

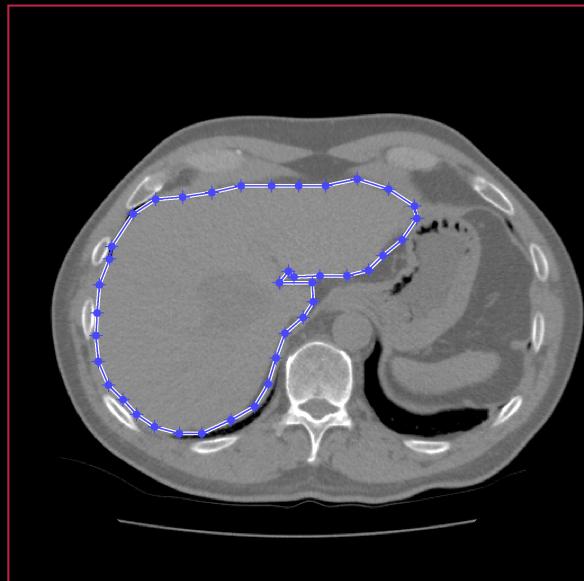
# Data fusion

Same patient – two scans

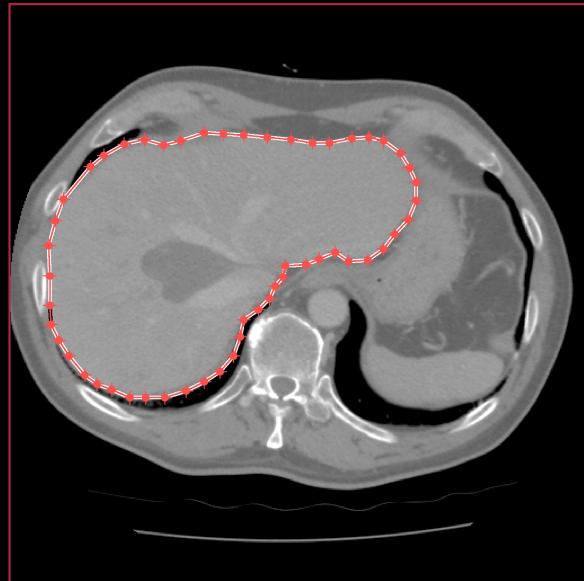


# Change detection

- Patient image before and after operation
- What has changed?
- Images need to be aligned before comparison

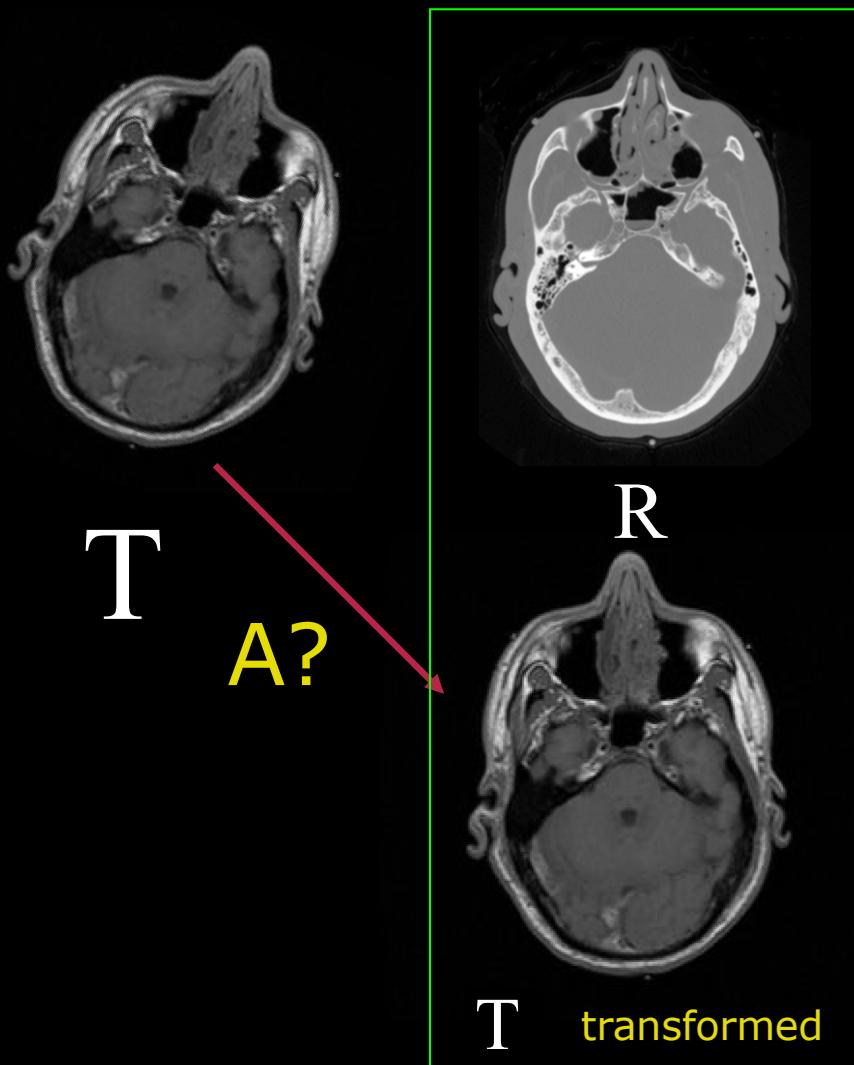


Before operation



After operation

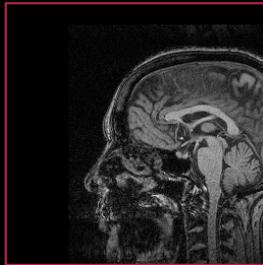
# Reference and template image



- The reference image R
- Template image T
- Transform the template so it fits the reference
- Combine geometrical transformations
- Find the transformation matrix, A for the best match

# The transformations

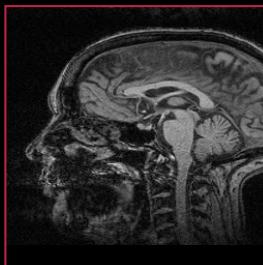
■ Translation



■ Rotation



■ Scaling

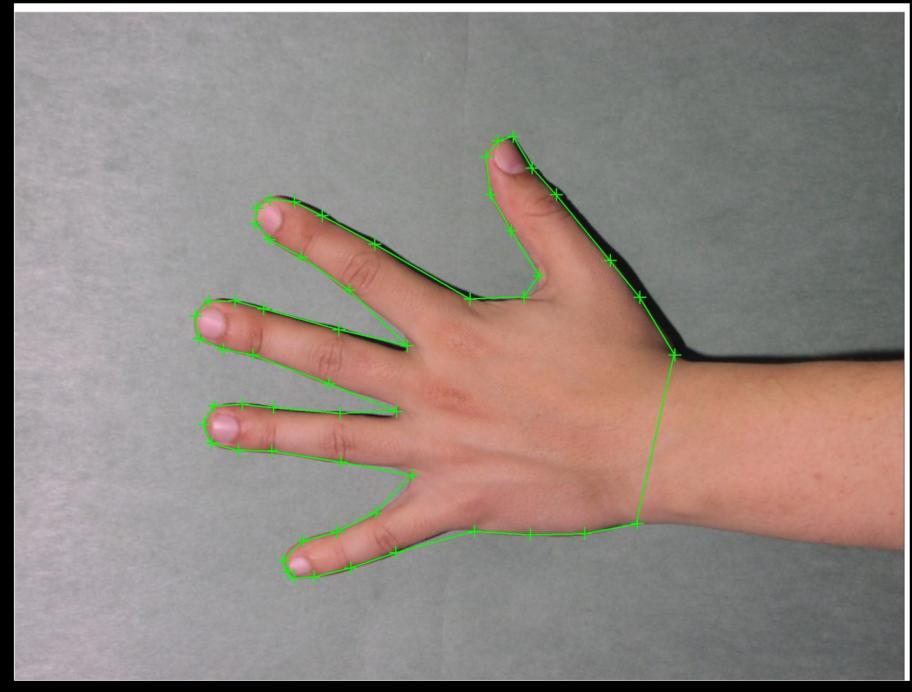
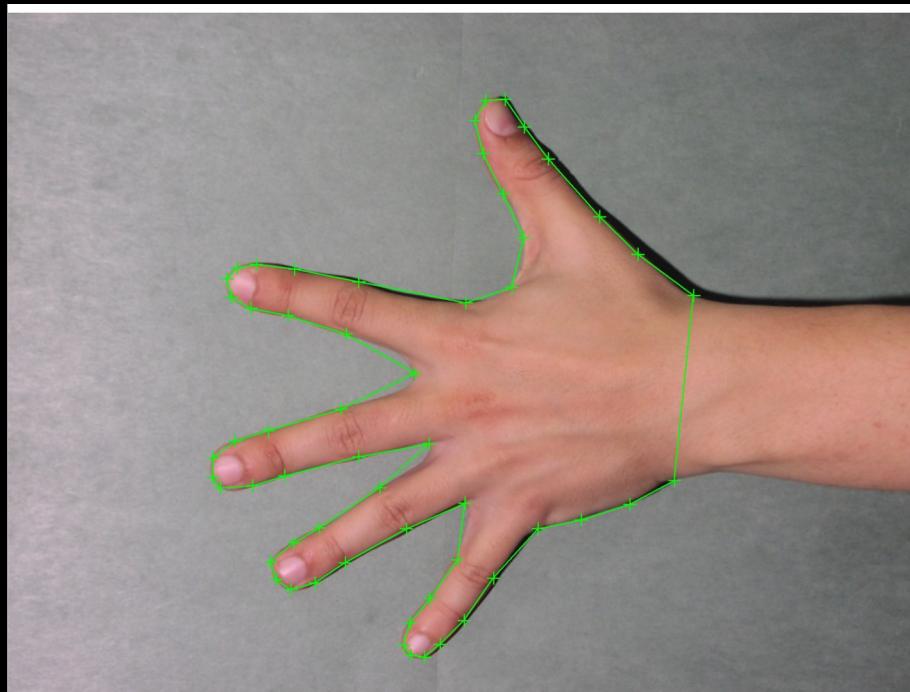


# Similarity measures

- The aim is to transform the template, so it *looks like* the reference
- Looks like = Similarity measure
- Image similarity
  - Subtract the two images and see “what is left”
- Landmark similarity
  - Landmarks from the two images should be “close together”

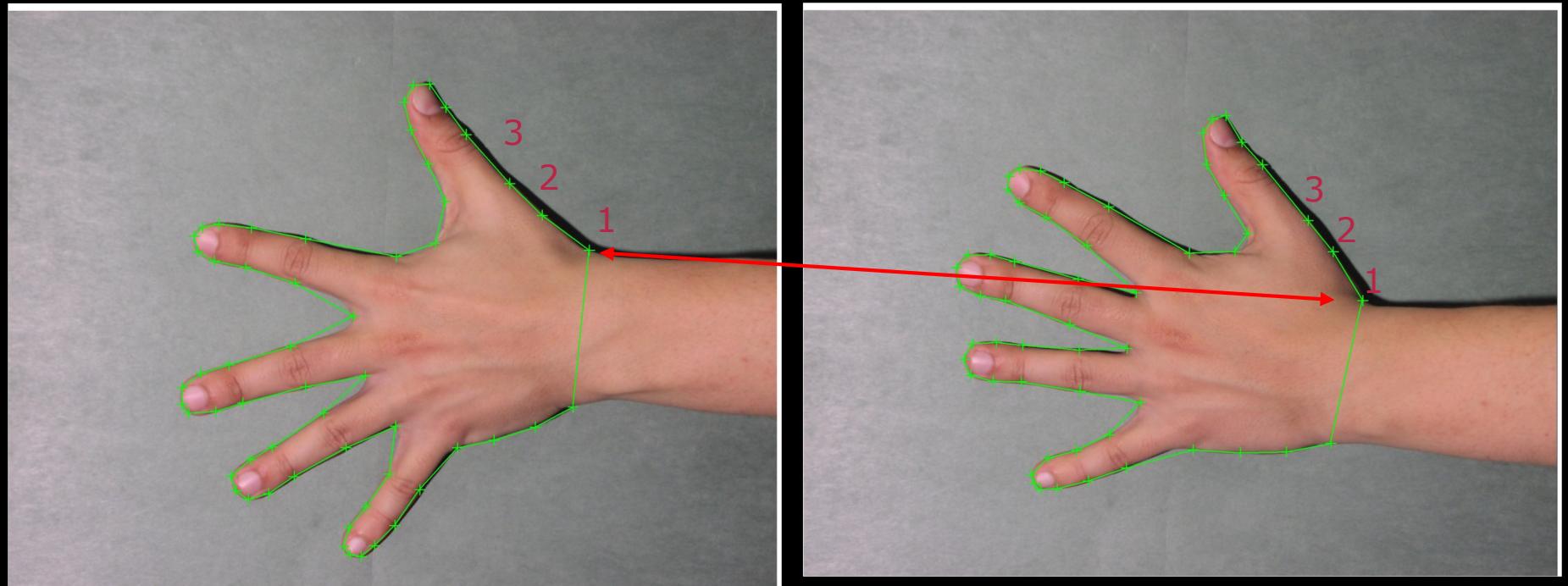
# Landmark Based Registration

- Landmarks placed on both reference and template image
- The landmark should have *correspondence*

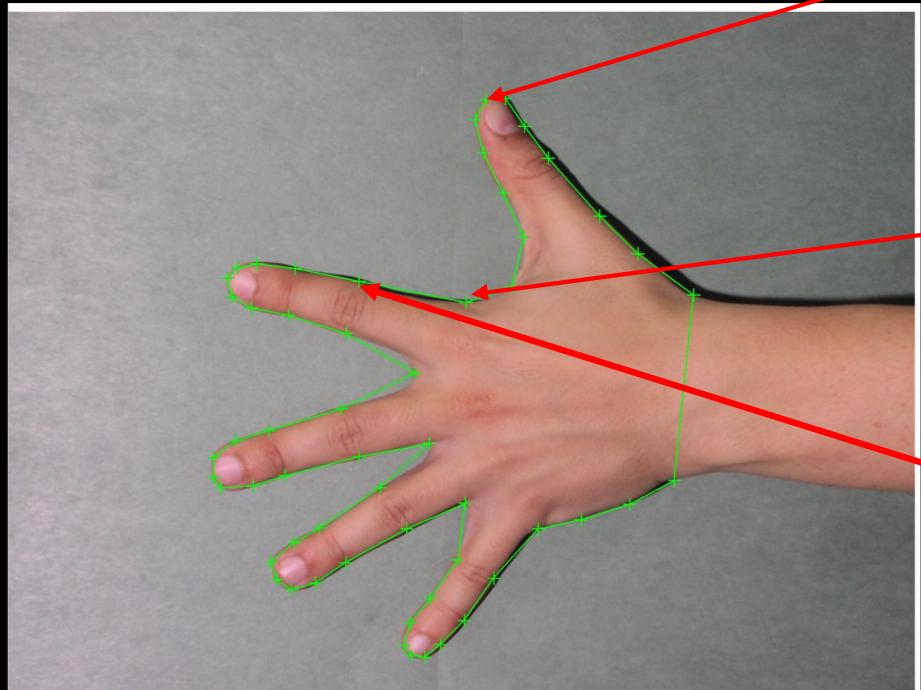


# Point correspondence

- Landmarks are numbered
- Each landmark should be placed the same place on both images



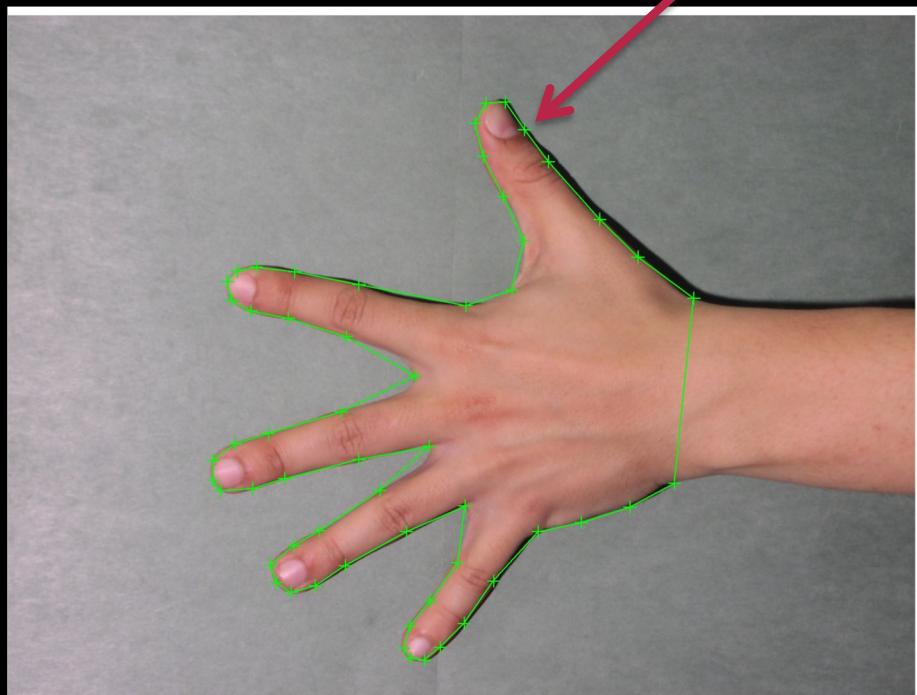
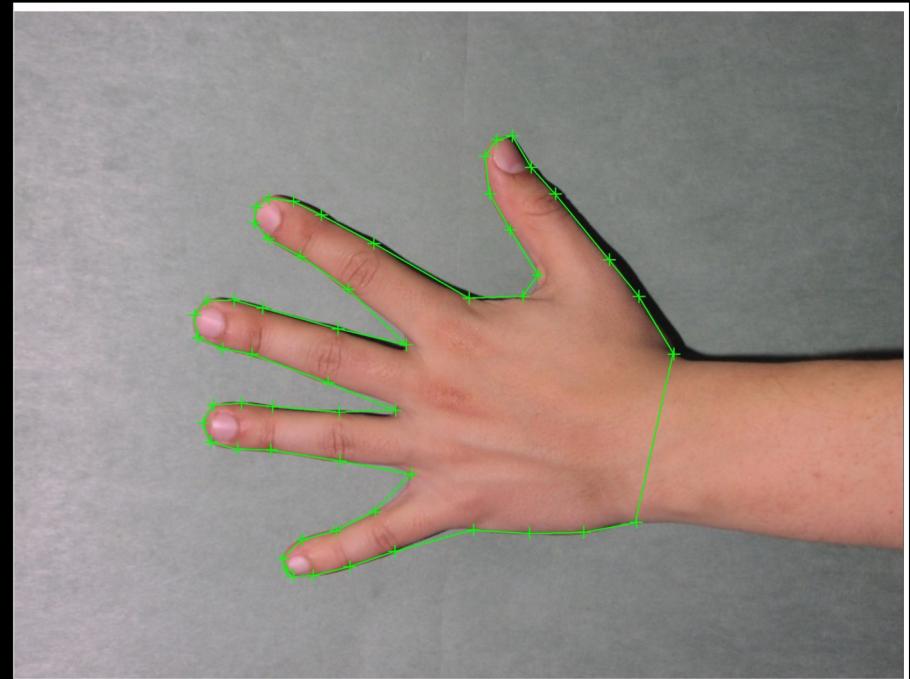
# Landmark types



- Anatomical landmark
  - a mark assigned by an expert that corresponds between objects in a biologically meaningful way
- Mathematical landmark
  - a mark that is located on a curve according to some mathematical or geometrical property
- Pseudo landmark
  - a mark that is constructed on a curve based on anatomical or mathematical landmarks

# Landmarks

$$a_5 = (412, 55)$$

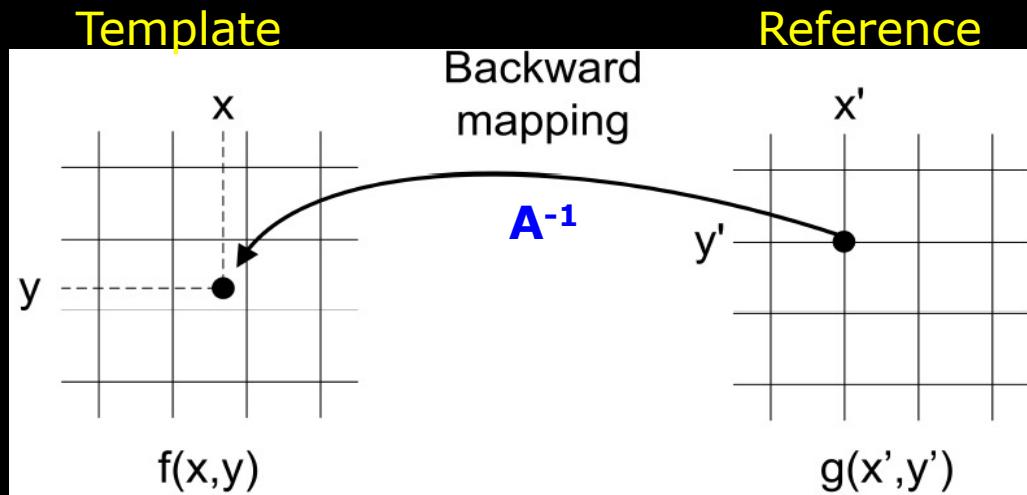
 $a_i$ Reference image  $R$  $b_i$ Template image  $T$

# The aim of registration

- We have selected Landmark points
- Find a transformation that maps the coordinates of the reference to the coordinates of the template
  - Why not the template to the reference?

Sampling the template image:

Backward mapping -> inverse transform



# The Transformation

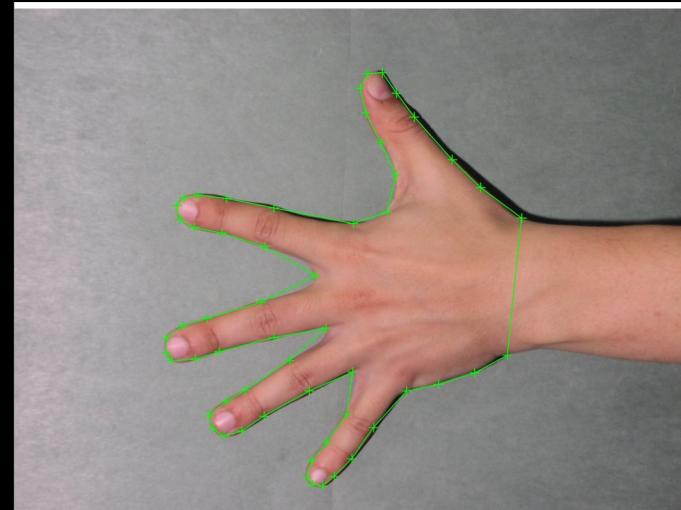
$$p' = T(p)$$

- Transforms point  $p$
- Into point  $p'$
- $T$  is for example geometrical transformations eg. a
  - Translation
  - Rotation
  - Rigid body transform
  - Similarity transform

# The Transformation

- Transforms points from the reference

$$a'_i = T(a_i)$$



$$a_i$$

# The parameters

$$w \in R^p$$

parameters

- The parameters is a vector with p elements
  - The type of transformation determines the number of parameters
  - Translation p = 2
  - Rotation p = 1
  - Scaling p = 1



# Quiz 4: Rigid body transform

## How many parameters?

$$w \in R^p$$

- A) 1
- B) 2
- C) 3
- D) 4
- E) 5

Solution:

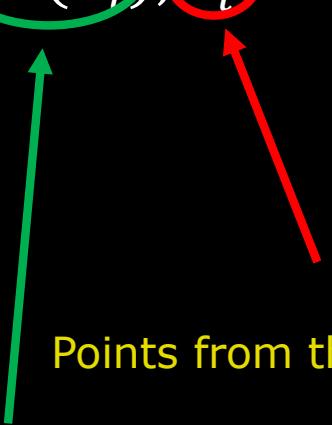
We have:

- Translation in x and y axis p= 2
- Rotation P= 1

In total 3 parameters for rigid transformation

$$w = (\Delta x, \Delta y, \theta)$$

# Objective function

$$F = \sum_{i=1}^N D(T(a_i), b_i)^2$$


Points from the template image

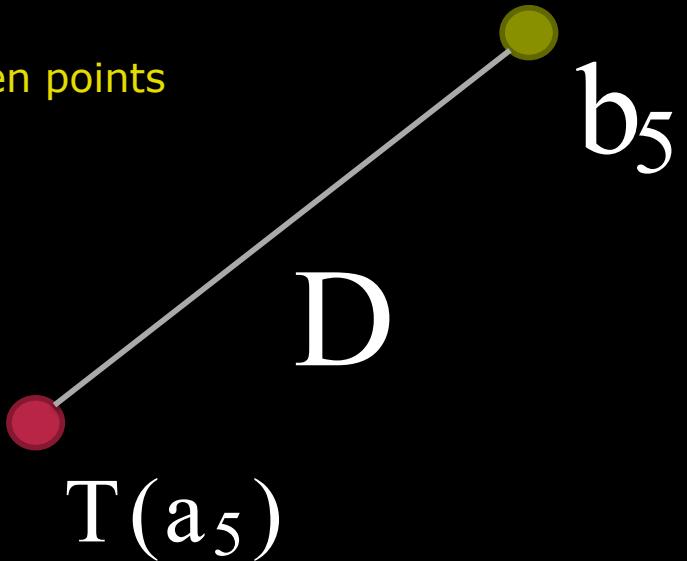
Transformed points from the reference image

- The *objective function* measures how well two point sets match
- It uses a *cost function* that *describe how to evaluate the match*
- Here the cost function is a *sum-of-squares distance function*
- Point sets could be **landmarks**

# Objective function

$$F = \sum_{i=1}^N D(T(a_i), b_i)^2$$

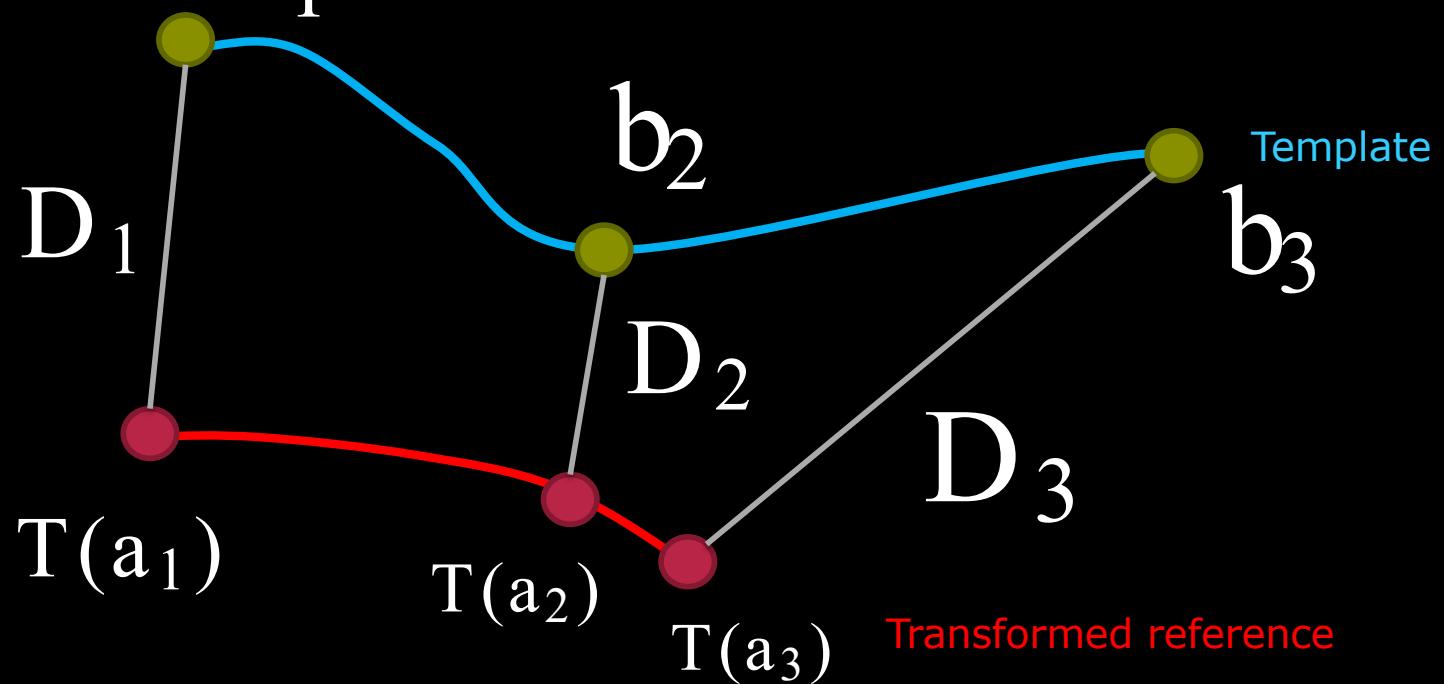
Distance between points



- The *objective function* measures how well two point sets match

# Objective function

$$F = \sum_{i=1}^3 D(T(a_i), b_i)^2 = D_1^2 + D_2^2 + D_3^2$$

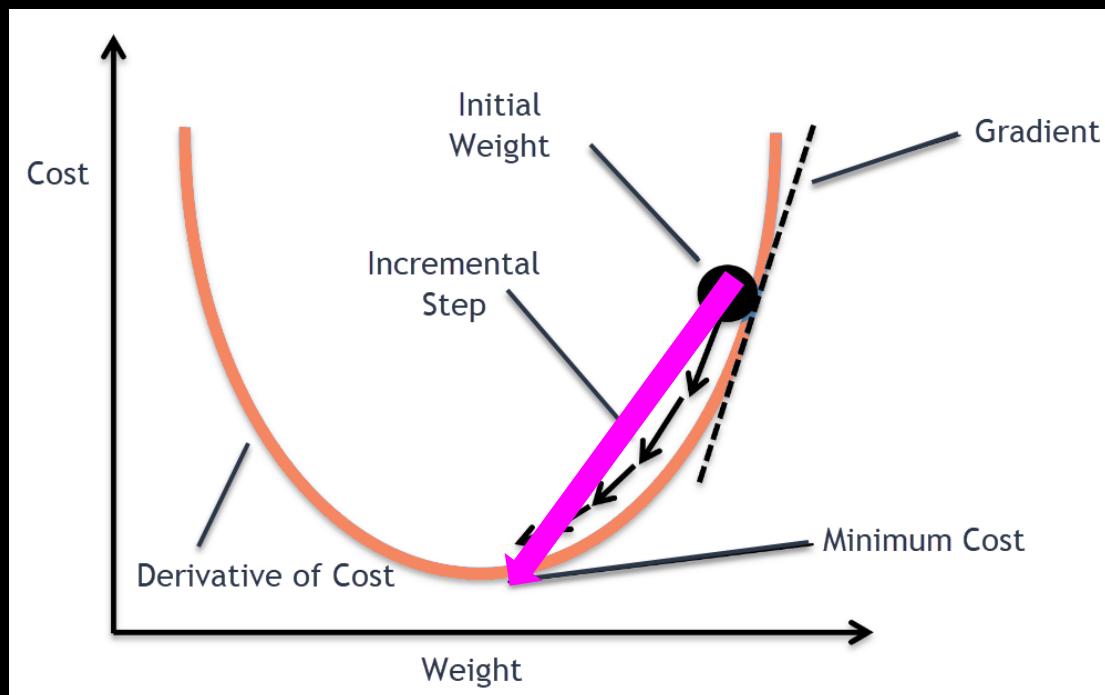


# Minimization / Optimization

$$F = \sum_{i=1}^N D(T(a_i), b_i)^2$$

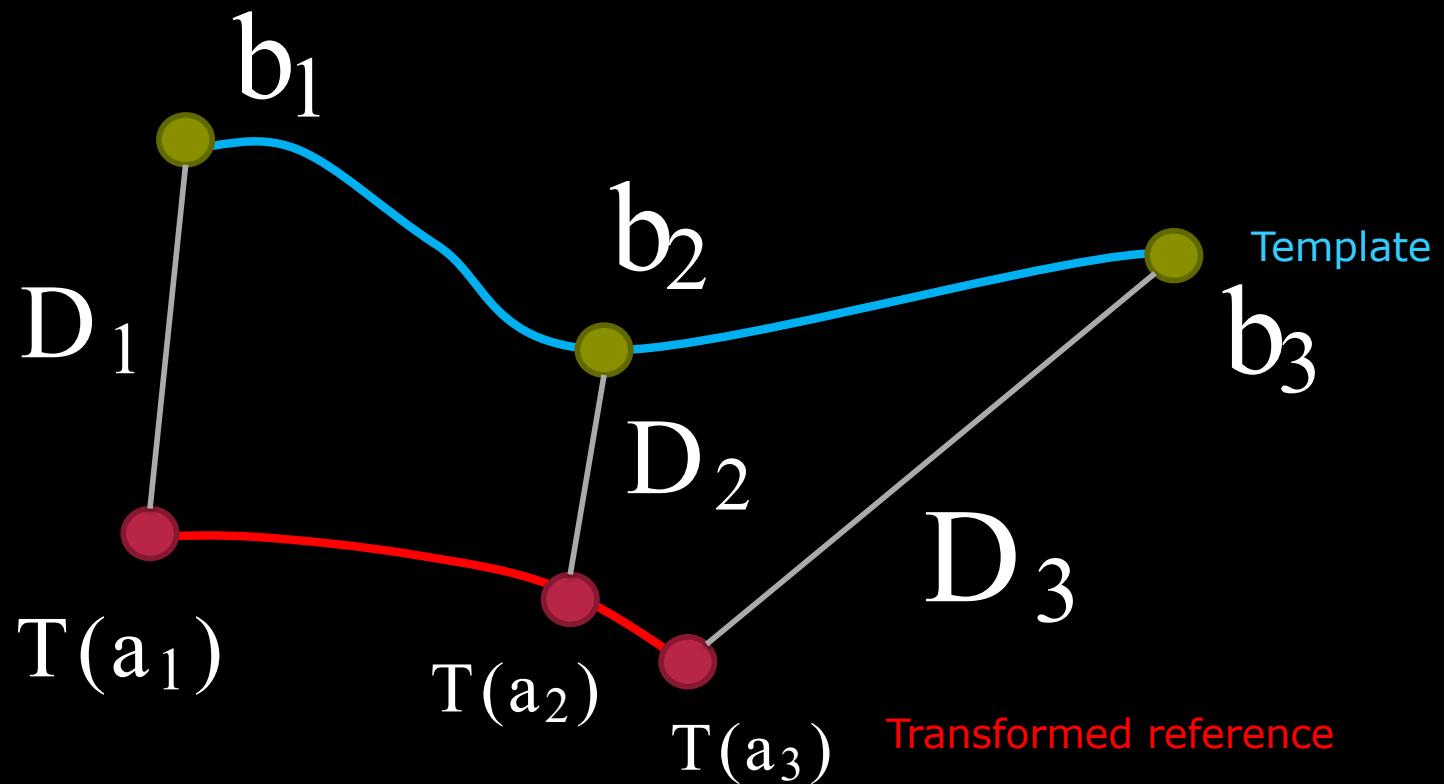
- Find the set of parameters that minimizes the objective function
- Optimisation strategy: **Analytic** (exact solution) vs Numerical?

$$\hat{w} = \arg \min_w F$$



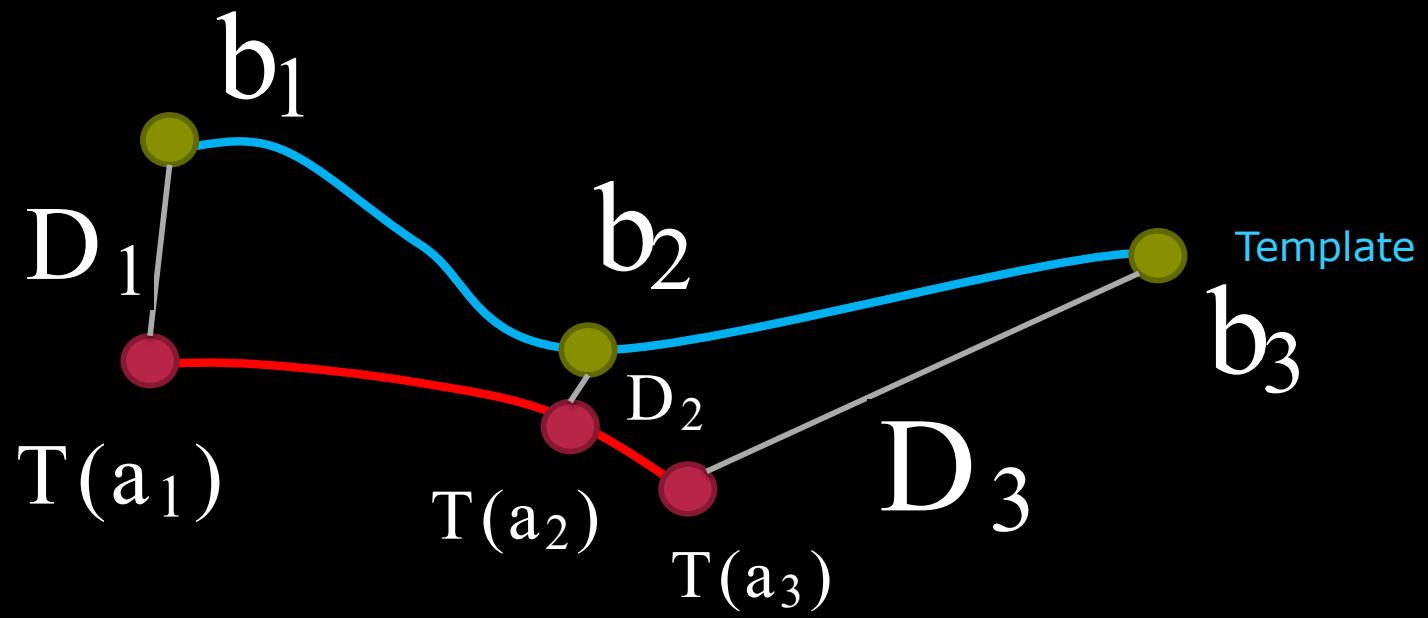
# Minimization – pure translation

$$F = D_1^2 + D_2^2 + D_3^2$$



## Minimization – pure translation

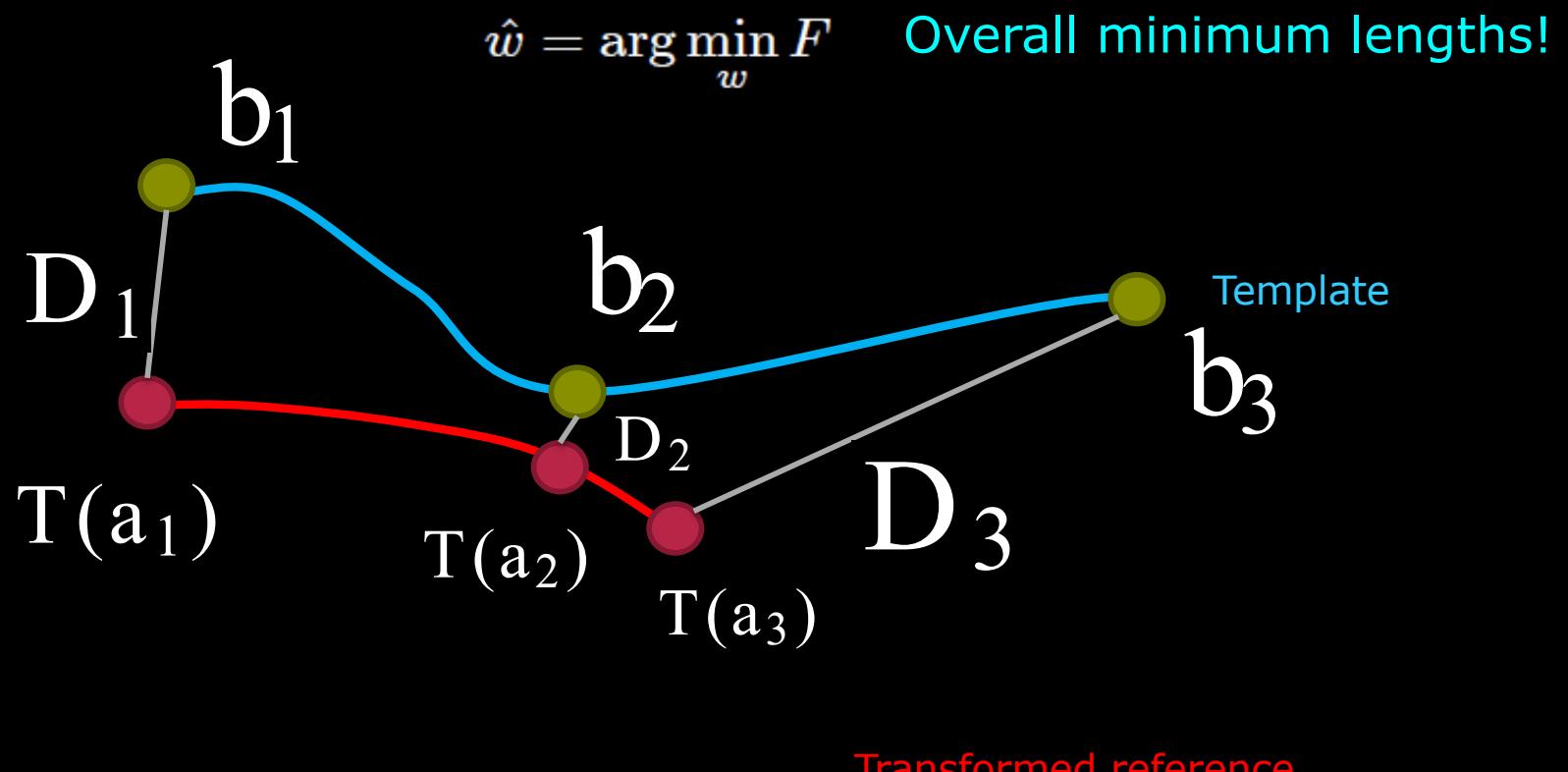
$$F = D_1^2 + D_2^2 + D_3^2 \text{ Decreased!}$$



Transformed reference

## Minimization – pure translation

$$F = D_1^2 + D_2^2 + D_3^2 \text{ Decreased!}$$



# Quiz 5: Objective function

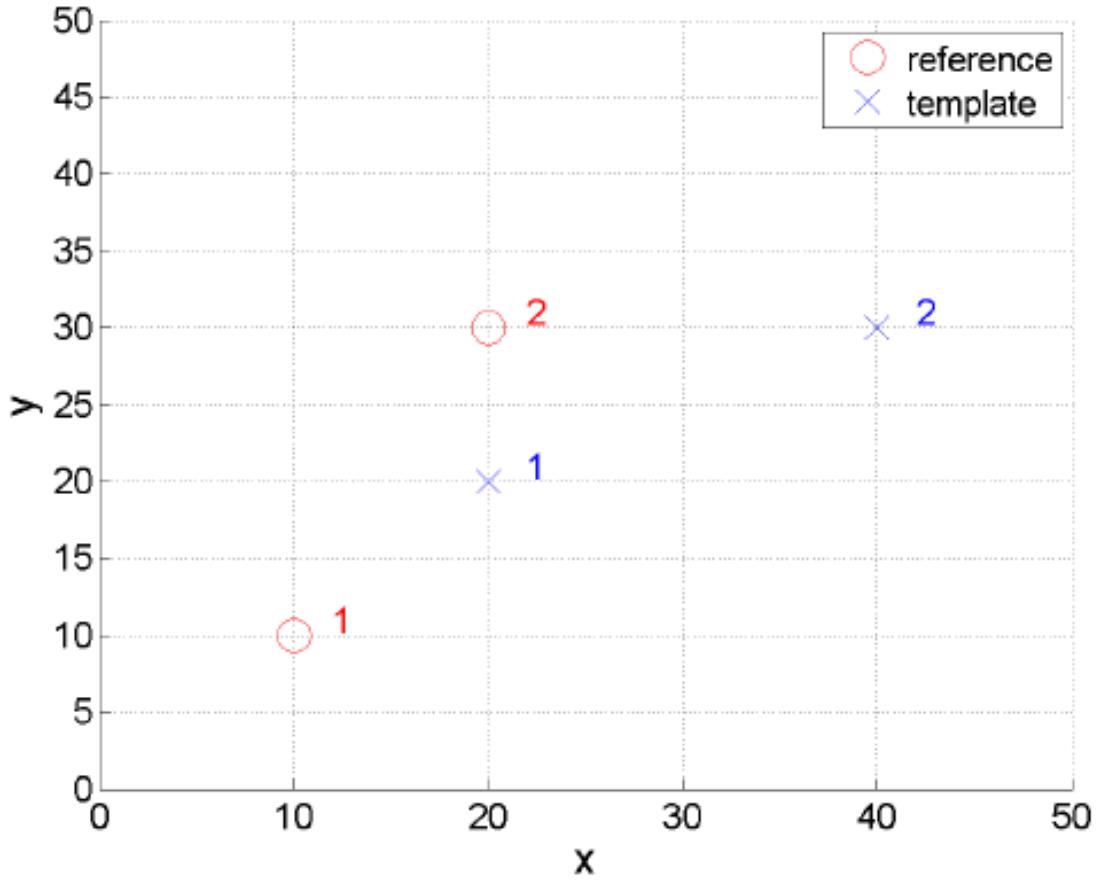
- A) 600
- B) 50
- C) 100
- D) 900
- E) 300

Solution:

$$D1^2 = \left\| \begin{bmatrix} 10 \\ 10 \end{bmatrix} - \begin{bmatrix} 20 \\ 20 \end{bmatrix} \right\|^2 = \left\| \begin{bmatrix} 10 \\ 10 \end{bmatrix} \right\|^2 = 200$$

$$D2^2 = \left\| \begin{bmatrix} 20 \\ 30 \end{bmatrix} - \begin{bmatrix} 40 \\ 30 \end{bmatrix} \right\|^2 = \left\| \begin{bmatrix} 20 \\ 0 \end{bmatrix} \right\|^2 = 400$$

An expert has placed two sets of landmark in the image below. We want to find the optimal translation. First we compute the objective function  $F$  and it is:

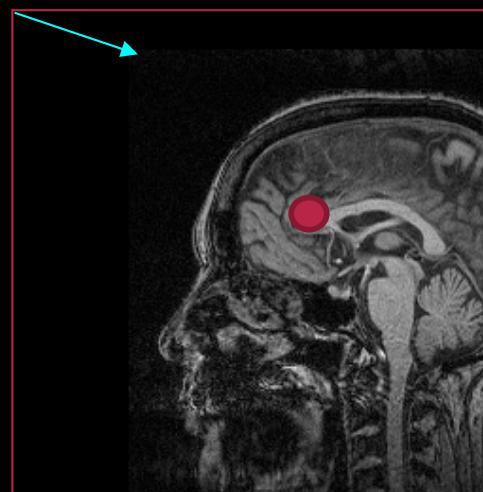
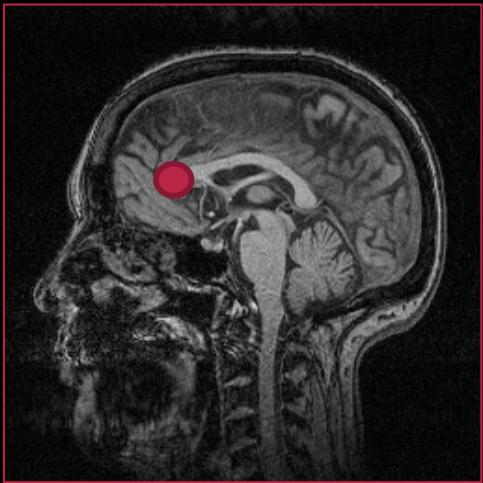


# Translation

- Simple shift of coordinates

$$T \begin{pmatrix} x \\ y \end{pmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = (x, y) + t$$

parameters  $w = (\Delta x, \Delta y)$





# Objective function for translation

Objective function

$$F = \sum_{i=1}^N D(T(a_i), b_i)^2$$

Translation

$$a'_i = a_i + t$$

Objective function for translation

$$F = \sum_{i=1}^N \|(a_i + t) - b_i\|^2$$



# Optimal function value

$$F = \sum_{i=1}^N D(T(a_i), b_i)^2$$

To find:  $\hat{w} = \arg \min_w F$

We simply differentiate w.r.t.  $w$ :

$$\frac{\partial F}{\partial w} = 0$$



# Optimal translation

Objective function

$$F = \sum_{i=1}^N \|(a_i + t) - b_i\|^2$$

Parameters

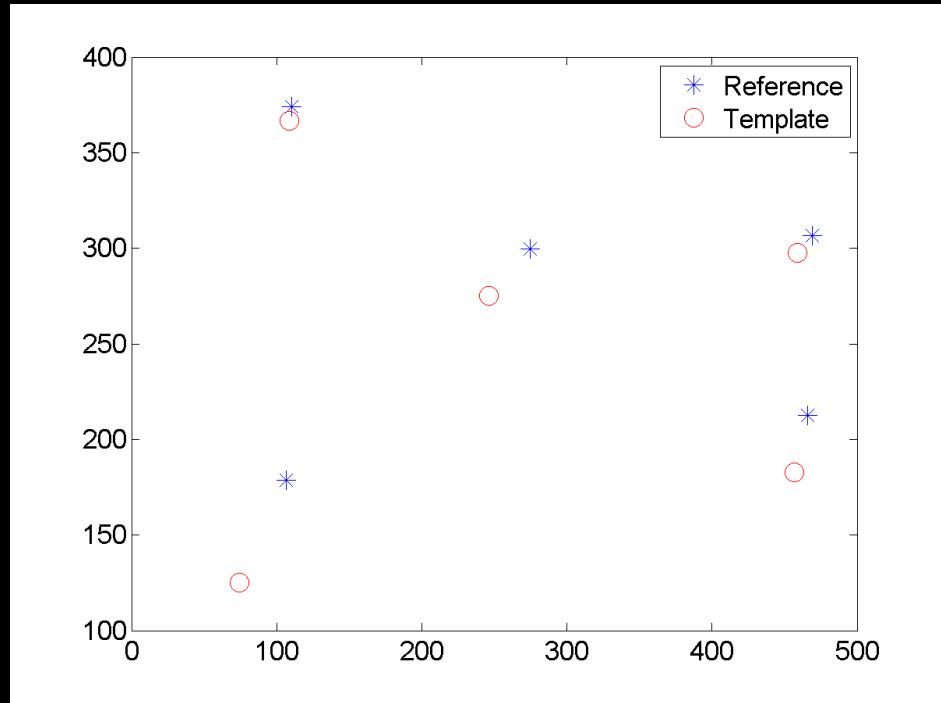
$$w = (\Delta x, \Delta y) = t$$

Optimal translation

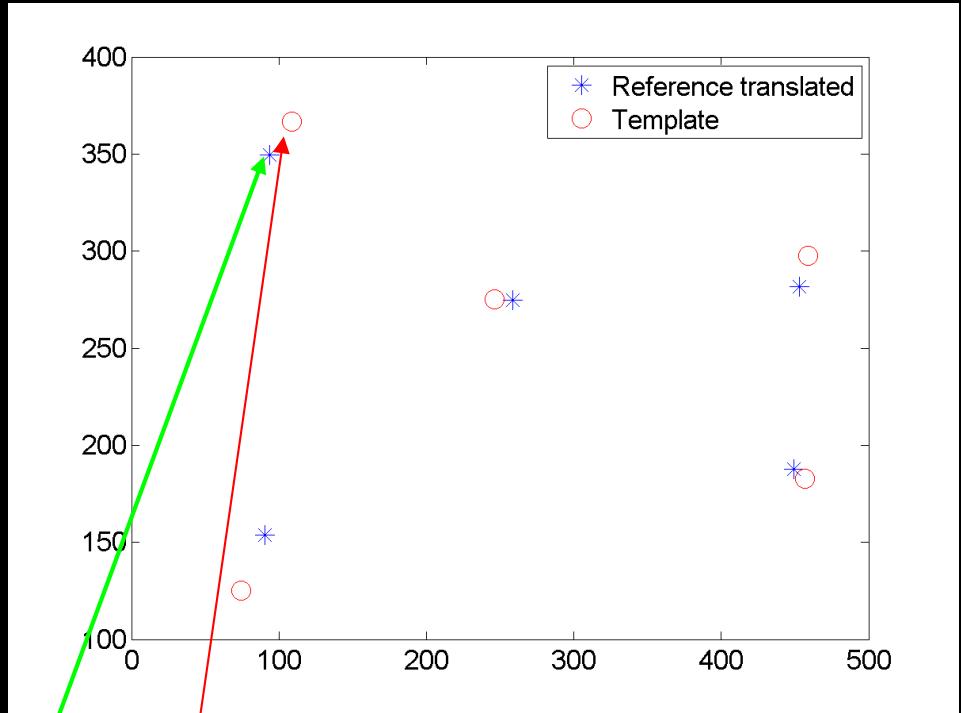
$$\hat{t} = \bar{b} - \bar{a} \qquad \bar{a} = \frac{1}{N} \sum_{i=1}^N a_i$$

Average point = centre of mass

# Optimal translation



Original landmarks



Reference points translated

$$F = \sum_{i=1}^N \| (a_i + t) - b_i \|^2$$

# Quiz 6:

## Optimal translation

- A) (-10, 10)
- B) (20, 5)
- C) (20, -5)
- D) (15, -5)

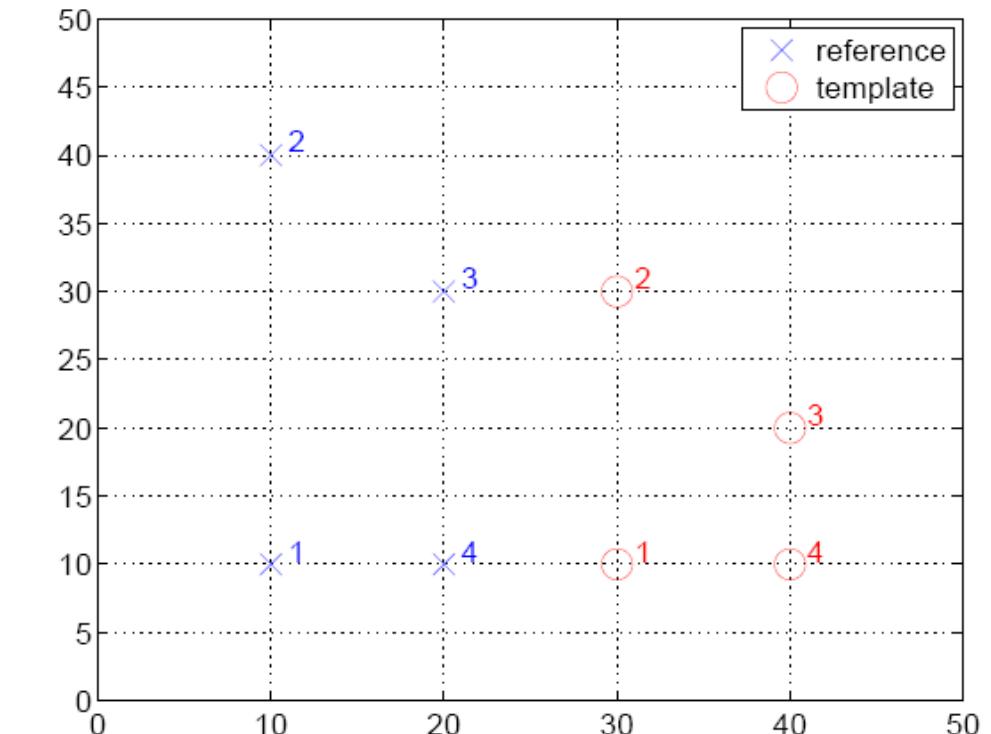
Solution:

$$\hat{t} = \bar{b} - \bar{a}$$

$$\bar{a} = \frac{1}{4} \left( \begin{bmatrix} 10 \\ 10 \end{bmatrix} + \begin{bmatrix} 10 \\ 40 \end{bmatrix} + \begin{bmatrix} 20 \\ 30 \end{bmatrix} + \begin{bmatrix} 20 \\ 10 \end{bmatrix} \right) = \begin{bmatrix} 15 \\ 22,5 \end{bmatrix}$$

$$\bar{b} = \frac{1}{4} \left( \begin{bmatrix} 30 \\ 10 \end{bmatrix} + \begin{bmatrix} 30 \\ 40 \end{bmatrix} + \begin{bmatrix} 40 \\ 20 \end{bmatrix} + \begin{bmatrix} 40 \\ 10 \end{bmatrix} \right) = \begin{bmatrix} 35 \\ 17,5 \end{bmatrix}$$

An expert has placed four landmarks in two images. The optimal translation that brings the landmarks from the reference image over in the landmarks from the template image. What is this translations?



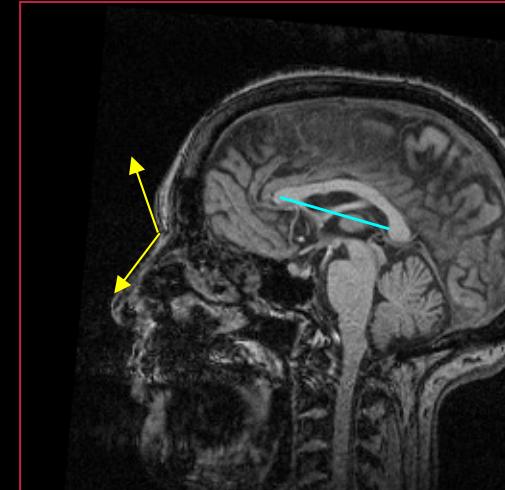
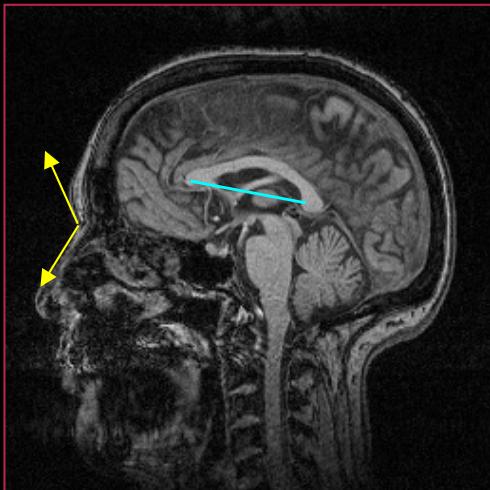
# Rigid body transformation

- Translation and rotation
- Rigid body
- Angles and *distances* are kept

$$R = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix}$$

$$a'_i = Ra_i + t$$

$$w = (\Delta x, \Delta y, \theta)$$





# Rigid body transformation

Transformation

$$a'_i = Ra_i + t$$

Rotation matrix

$$R = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix}$$

Objective function

$$F = \sum_{i=1}^N \| (Ra_i + t) - b_i \|^2$$

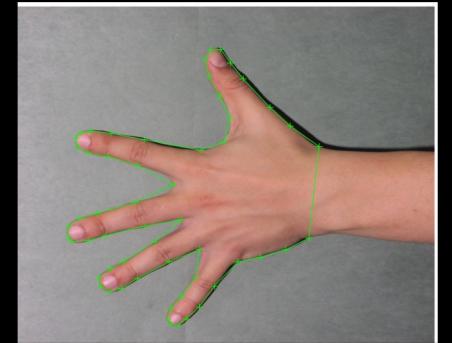
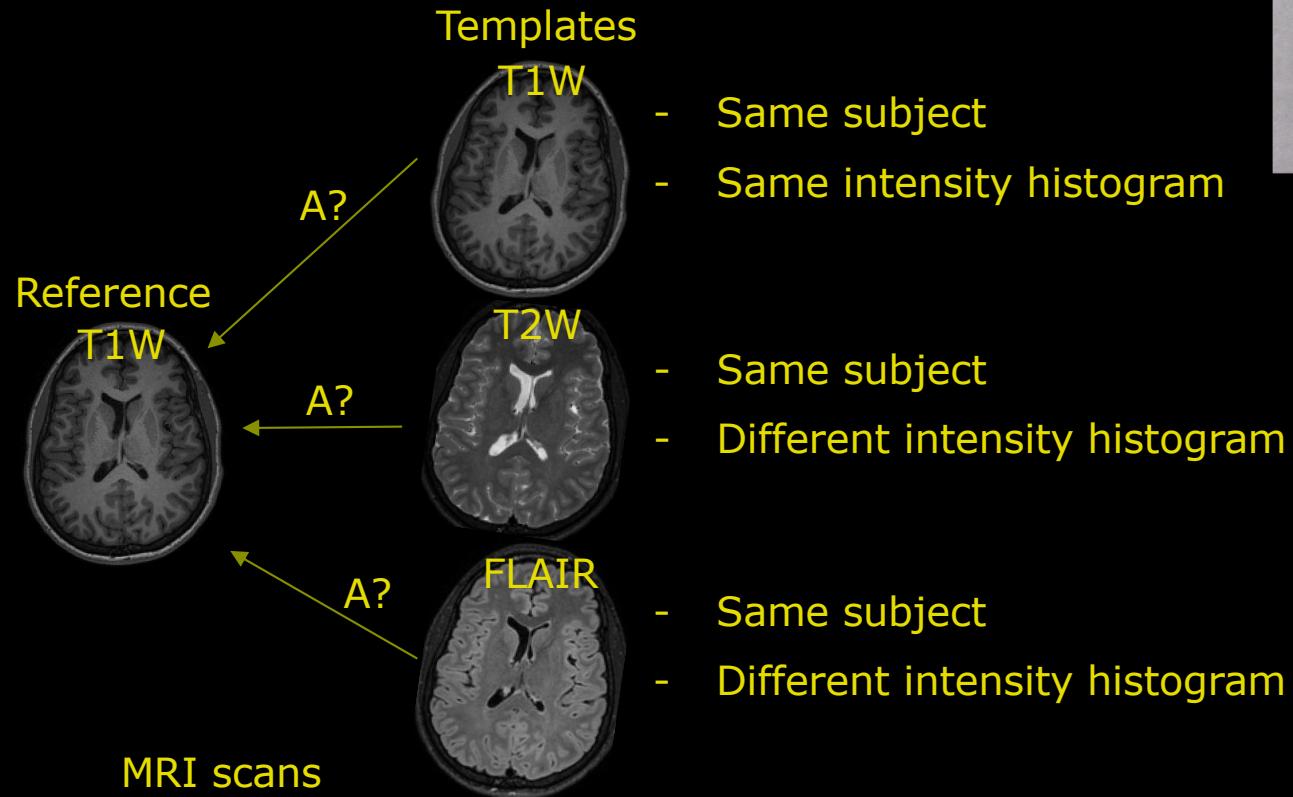


# Optimal rigid body transformation

- The minimum of the objective function can be found in several ways
- The rotation can be found analytically by *singular value decomposition*

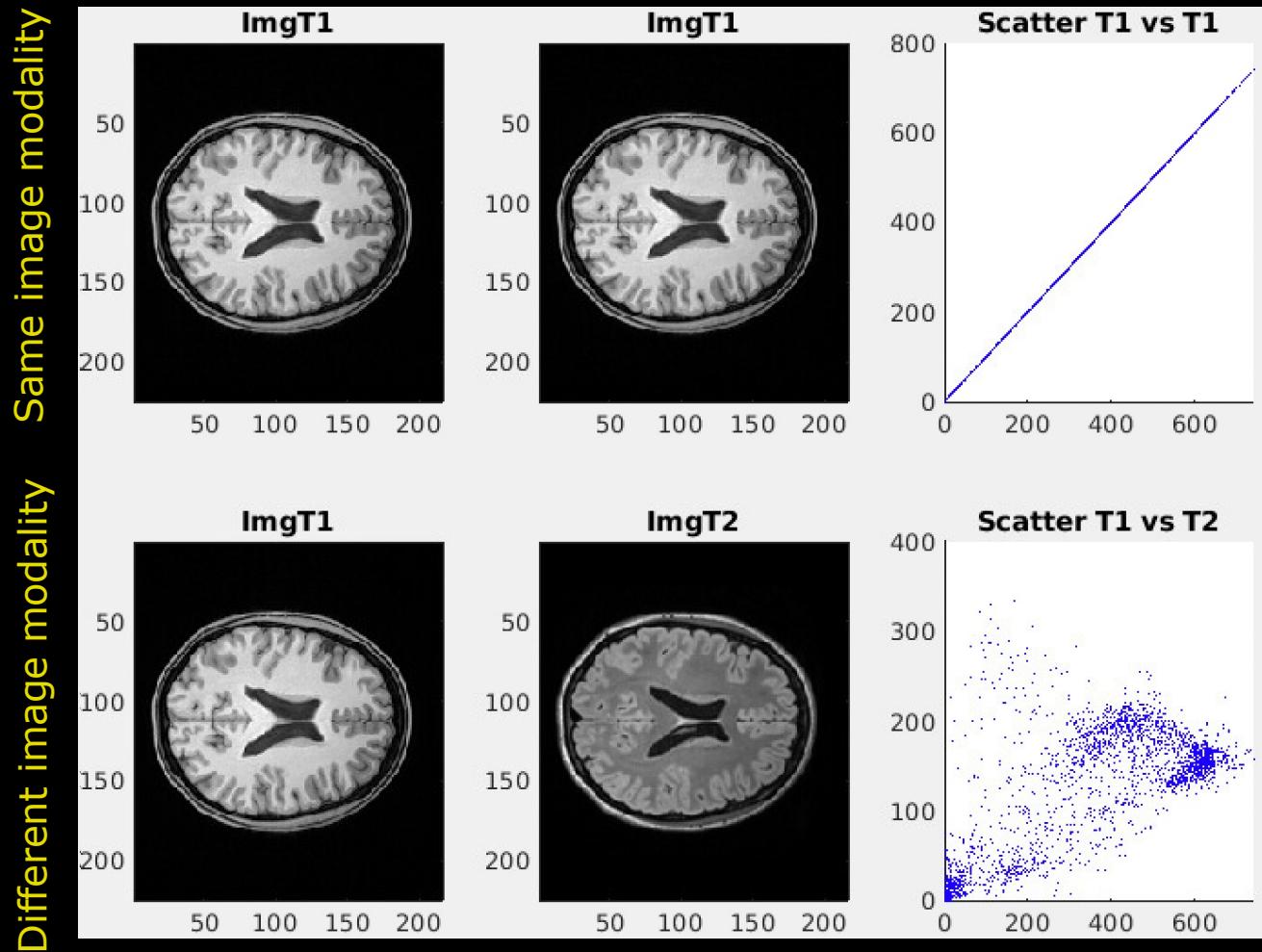
# Similarity measures

- Landmarks - time consuming to obtain
- Alternative: joint intensity histograms?



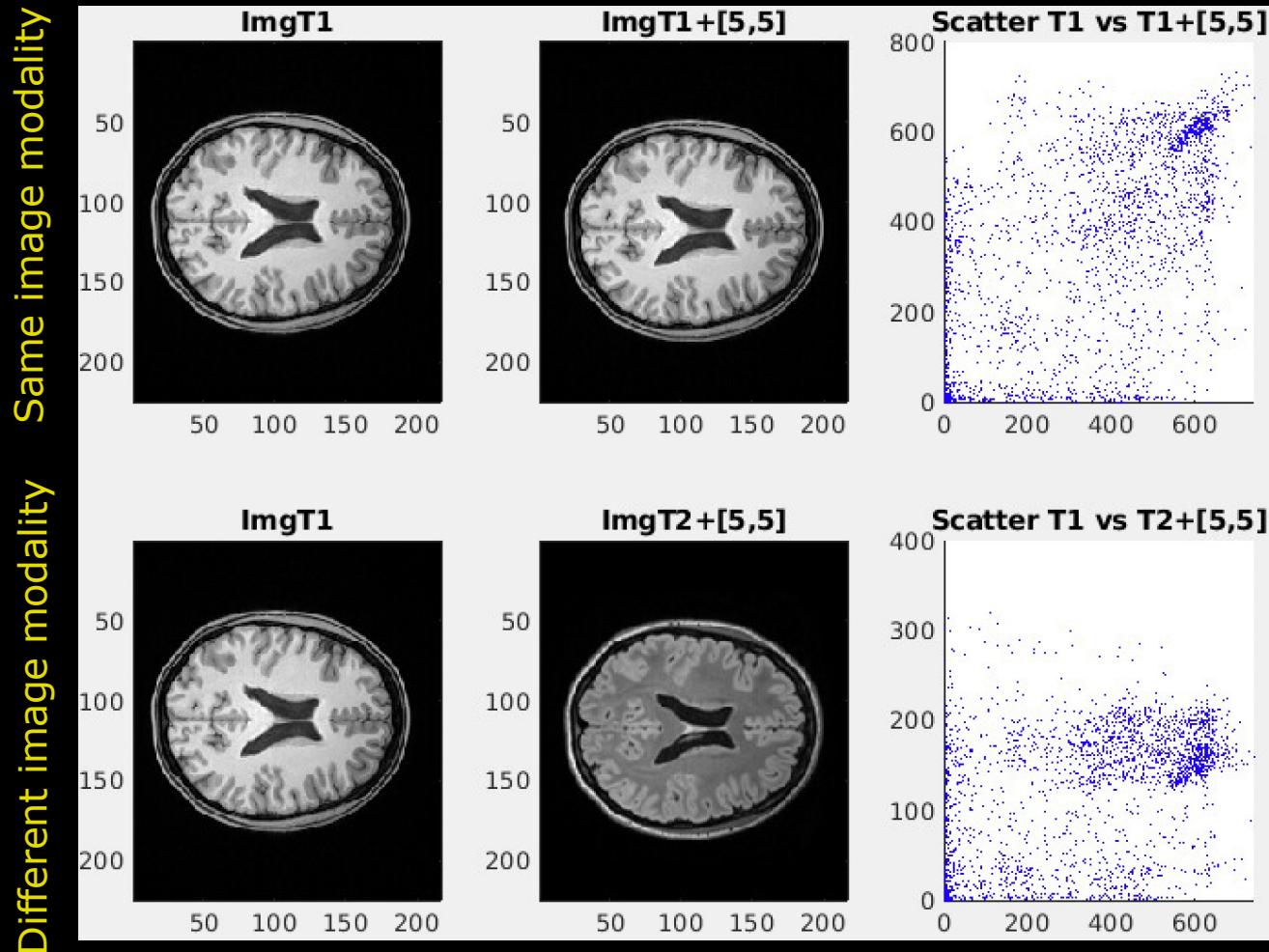
# Joint intensity histograms

- Perfect registered: Optimal joint intensity agreement



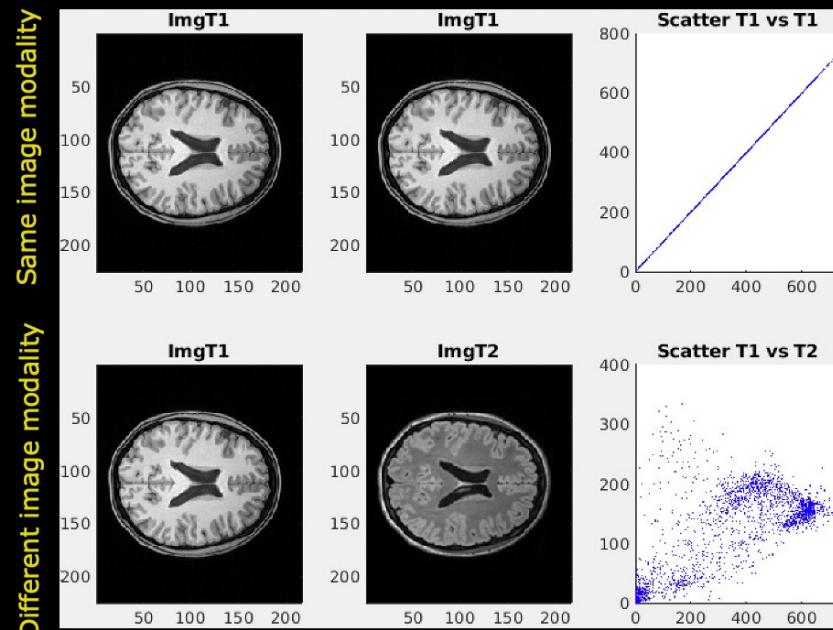
# Joint intensity histograms

- Small translation difference: Lower joint intensity agreement



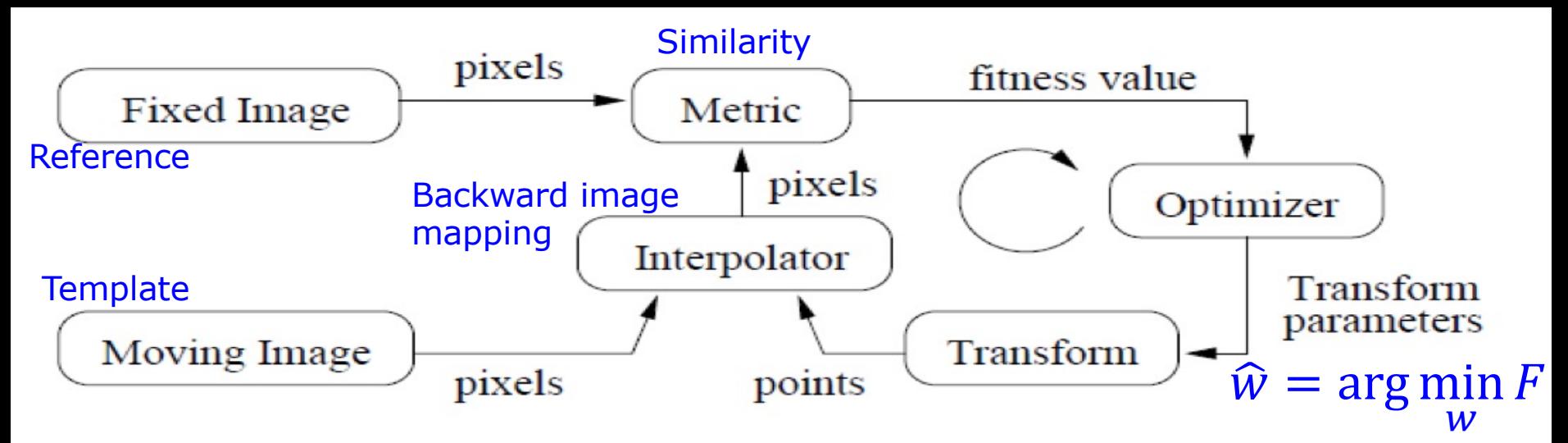
# Joint intensity histograms

- Objective function i.e. a **similarity measure** to find the optimal transformation
- Many methods exist but two types dominate:
  - Cross-correlation based
    - Fast to estimate, not optimal choice if different image modalities (histograms)
  - Joint entropy based also known as **Mutual Information (MI)**
    - Slow to estimate, robust when image modalities (i.e., histograms) are different



# The image registration “pipeline”

- Register *Template image* to *Reference image* via geometrical transformations
- Select a similarity measure to map coordinates from template
- Objective function - Find optimal parameters:  $\hat{w} = \arg \min_w F$
- The solution is often found by numerical optimisation (optimizer)

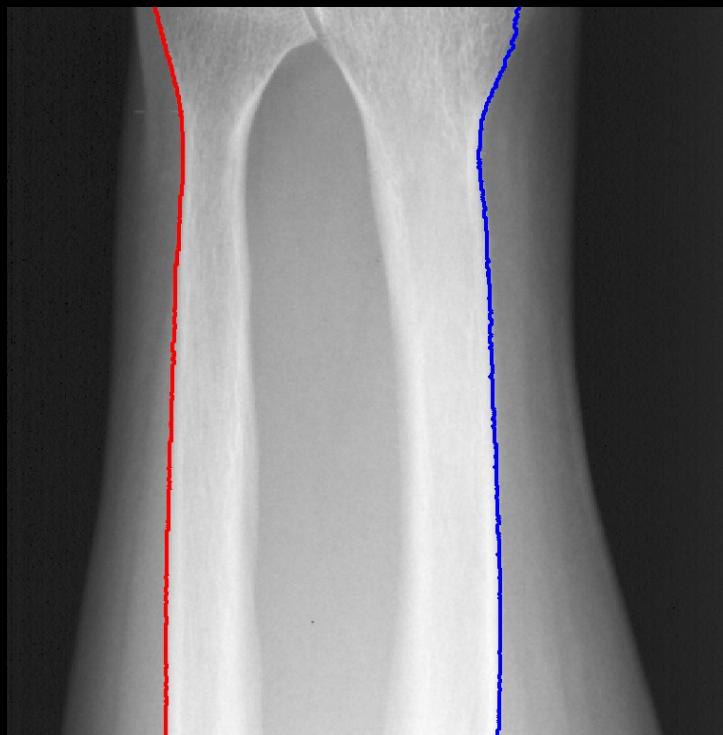
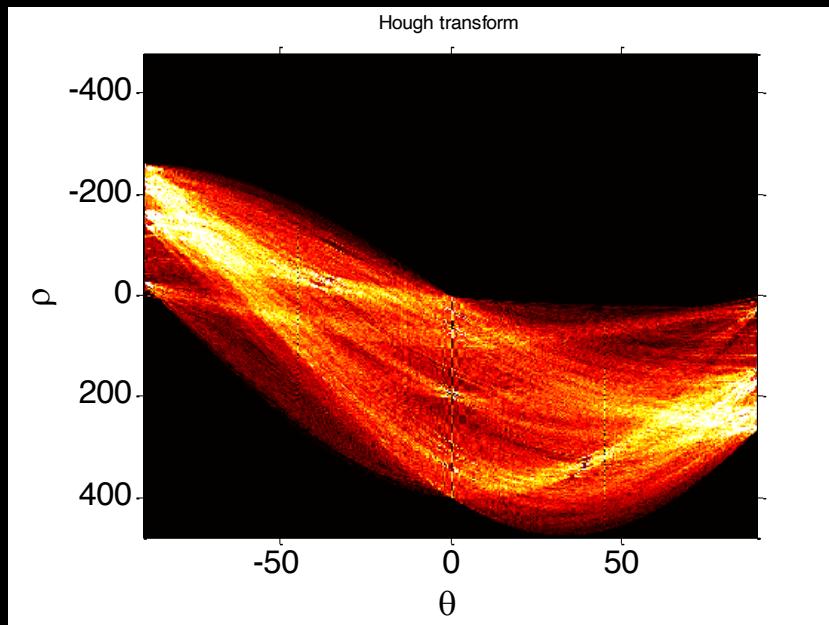


... Or use existing methods !!

# What did you learn today?

- Construct a translation, rotation, scaling, and shearing transformation matrix of a point
  - Use transformation matrices to perform point transformations
  - Describe the difference between forward and backward mapping
  - Transform an image using backward mapping and bilinear interpolation
- 
- Describe the image registration
  - Describe the different types of landmarks
  - Annotate a set of image using anatomical landmarks
  - Describe the objective function used for landmark and joint histogram-based registration
  - Compute the optimal translation between two sets of landmarks
  - Use the rigid body transformation for image registration
  - Describe the general "pipeline" for image registration

# Next week: Boundary Tracing (Hough Transformation) and Dynamic programming





# Image Analysis

Tim B. Dyrby

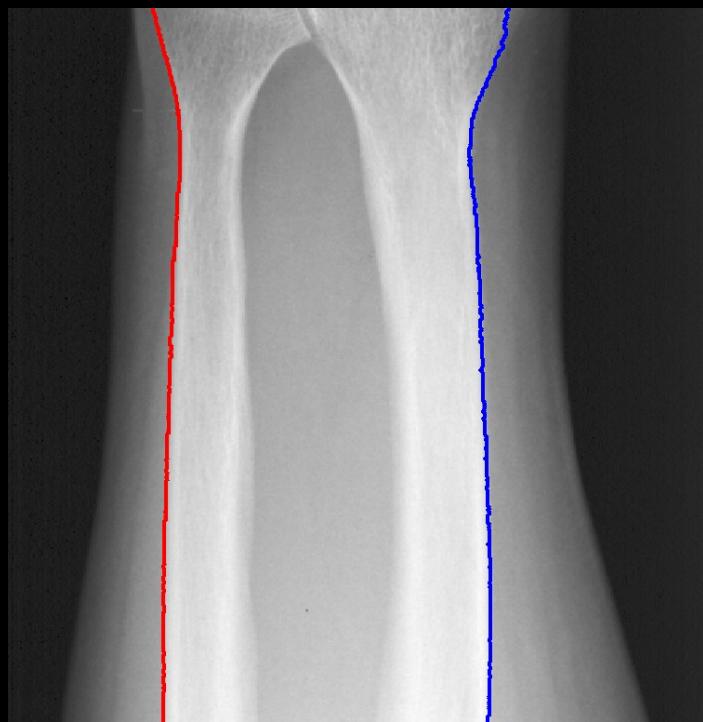
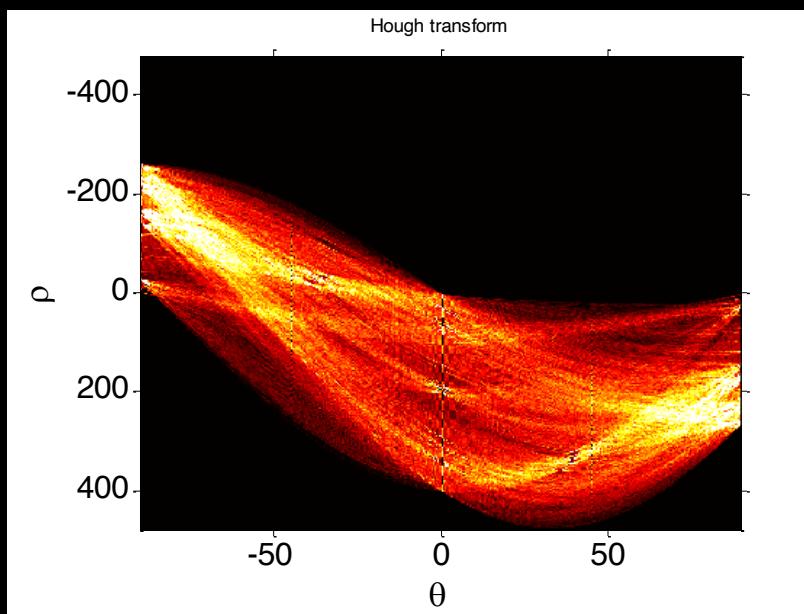
Rasmus R. Paulsen

DTU Compute

[tbdy@dtu.dk](mailto:tbdy@dtu.dk)

<http://www.compute.dtu.dk/courses/02502>

# Lecture 8 – Hough Transformation and Path Tracing





Go to [www.menti.com](http://www.menti.com) and use the code 8743 4620

## Quiz testing your knowledge How long time did it take to develop the Dijkstra algorithm?

### Breaking NEWS!

*One morning I was shopping in Amsterdam with my young fiancée, and tired, we sat down on the café terrace to drink a cup of coffee and I was just thinking about whether I could do this, and I then designed the algorithm for the shortest path. As I said, it was a twenty-minute invention.*

*— Edsger Dijkstra, in an interview with Philip L. Frana, Communications of the ACM, 2001 [3]*

0	0	0	0	0	0
3 years	1 year	2 weeks	1 hour	20 min	37 sec

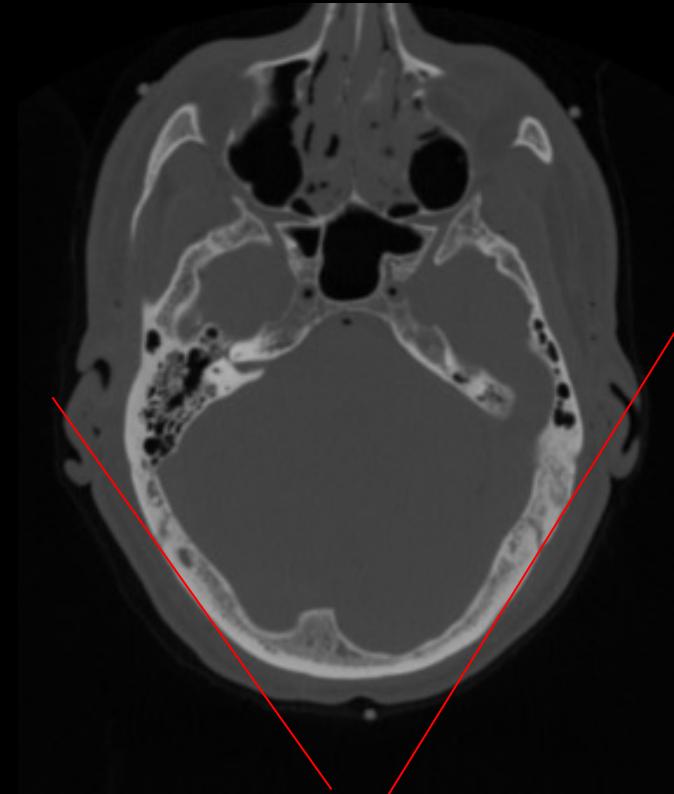
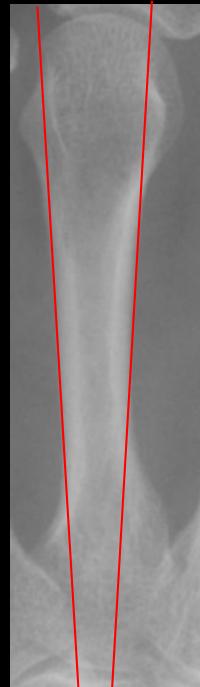


# What can you do after today?

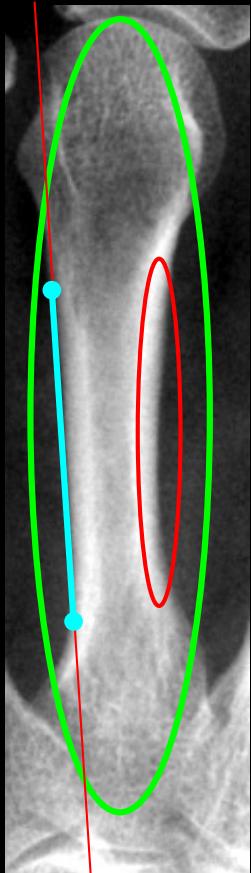
- Use the Hough transform for line detection
- Describe the slope-intercept, the general form and the normalised form of lines
- Describe the connection between lines and the Hough space
- Use edge detection to enhance images for use with the Hough transform
- Use dynamic programming to trace paths in images
- Describe how an image can be used as a graph
- Describe the fundamental properties of a cost image
- Compute the cost of path
- Compute an accumulator image for path tracing
- Compute a back tracing image for path tracing
- Choose appropriate pre-processing steps for path tracing
- Describe how circular structures can be located using path tracing

# Line Detection

- Find the lines in an image



# What is a line?



- It can be the entire object
  - Large scale
- Can also be the border between an object and the background
  - Small scale
- Normally only locally defined

# Enhancing the lines



- We want to locate the borders
  - Enhance them
- Filtering (Prewitt)
- Edge detection

## Prewitt:

Vertical			Horizontal		
-1	0	1	-1	-1	-1
-1	0	1	0	0	0
-1	0	1	1	1	1

# What is a line II?



- Result of the edge filter is a selection of white pixels
- Some of them define a line
  - Not a perfect straight line
  - “Linelike”
- How do we find the collection of points that define a line?

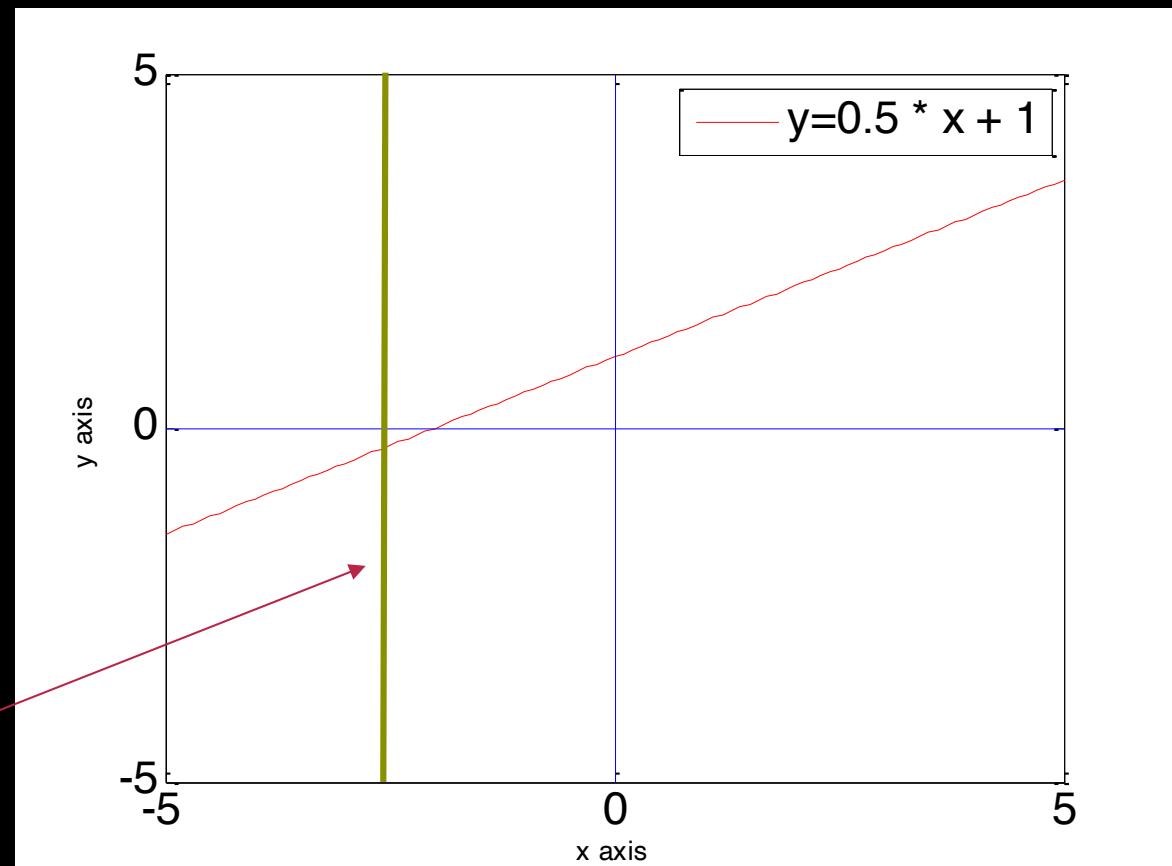
# Mathematical line definition

- The classical definition (slope-intercept form)

$$y = ax + b$$

Slope      Intercept

Can not represent lines that  
are vertical



# Mathematical line definition

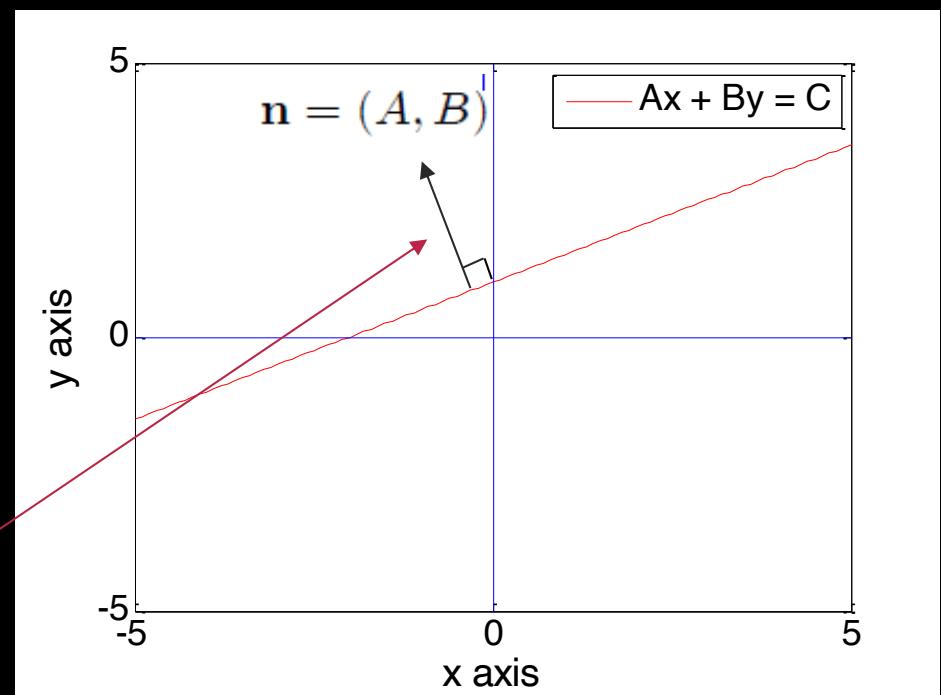
- General definition (the normal form)

$$Ax + By = C$$

- With

$$A^2 + B^2 = 1$$

Line normal



# Mathematical line definition

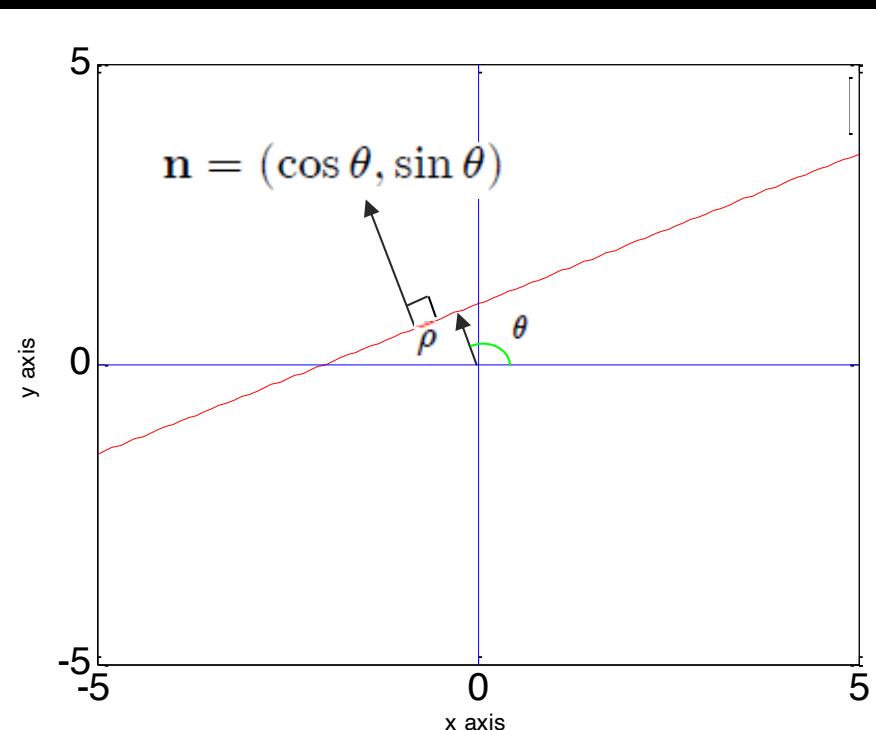
- Normal form parameterisation

$$x \cos \theta + y \sin \theta = \rho$$

- where
  - $\rho$  is the distance from the origin
  - $\theta$  is the angle

$$(\cos \theta)^2 + (\sin \theta)^2 = 1$$

$$A^2 + B^2 = 1$$

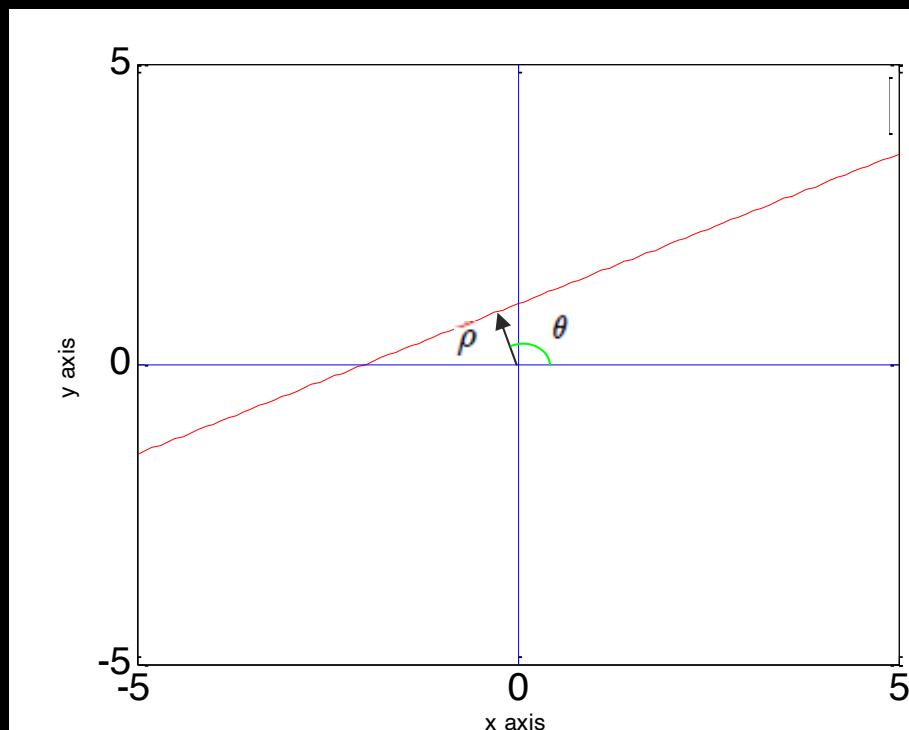


# Mathematical line definition

- Normal form parameterisation

$$x \cos \theta + y \sin \theta = \rho$$

- Therefore a line can be defined by two values
  - $\rho$
  - $\theta$
- A line can therefore also be seen as a *point* in a  $(\theta, \rho)$ -space



# Relation between forms

## ■ From normal form to the slope-intercept form

The normal form:  $p = x \cos \theta + y \sin \theta$

The slope-intercept form:  $y = ax + b$

$$\text{Start: } p = x \cos \theta + y \sin \theta$$

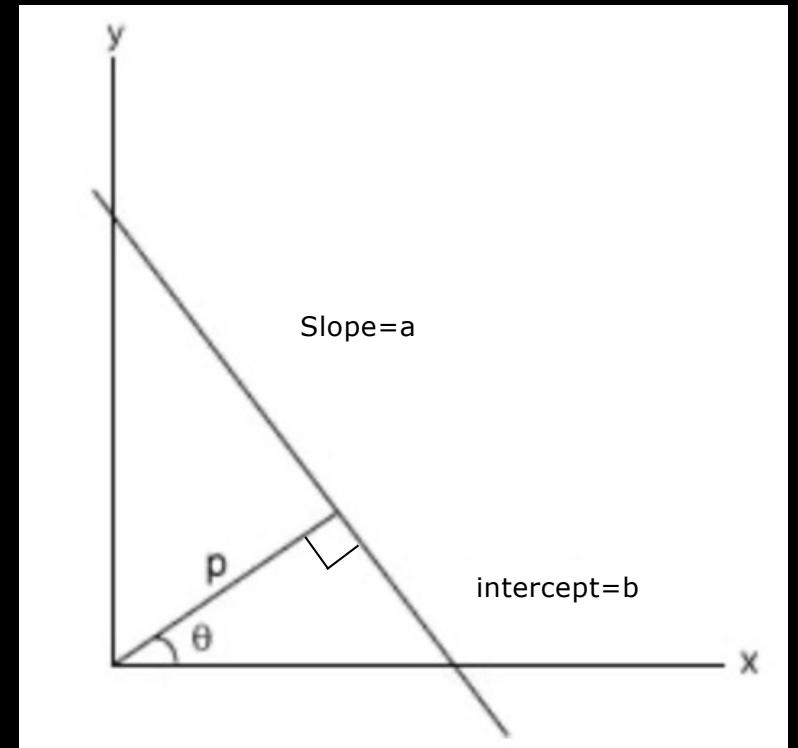
$$-x \cos \theta + p = y \sin \theta$$

$$-x \cot \theta + p \cosec \theta = y$$

$$y = x * (-\cot \theta) + p(\cosec \theta)$$

Slope=a

Intercept=b





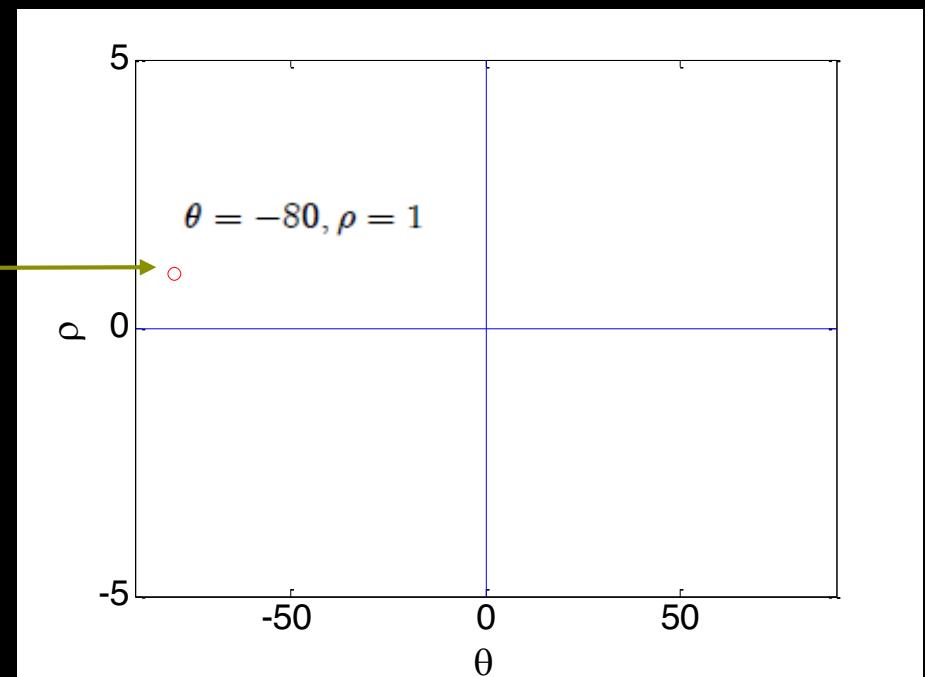
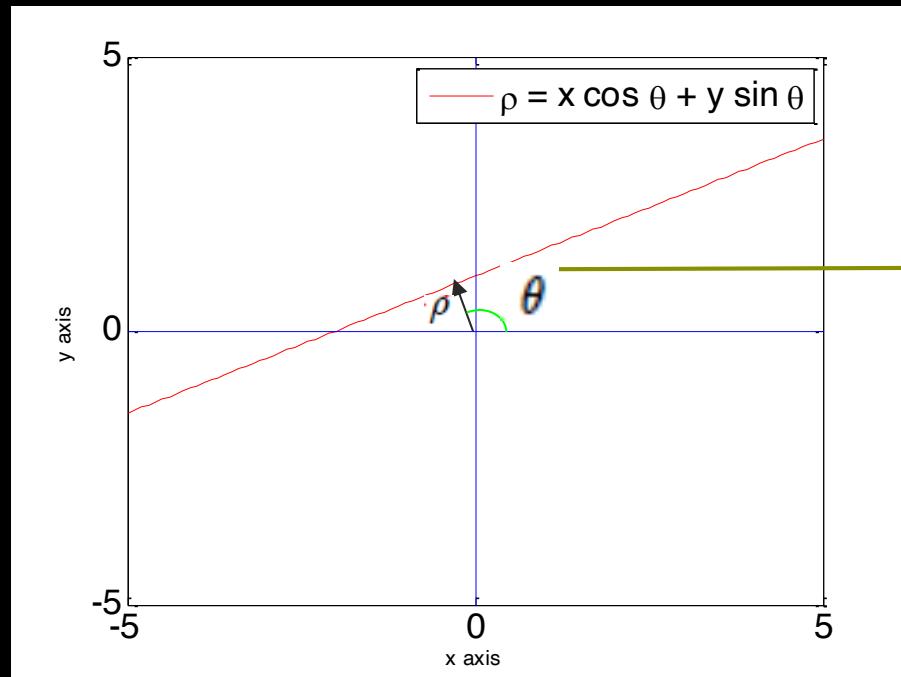
# Something about angles

$\theta \in [0^\circ, 180^\circ]$  In the course notes

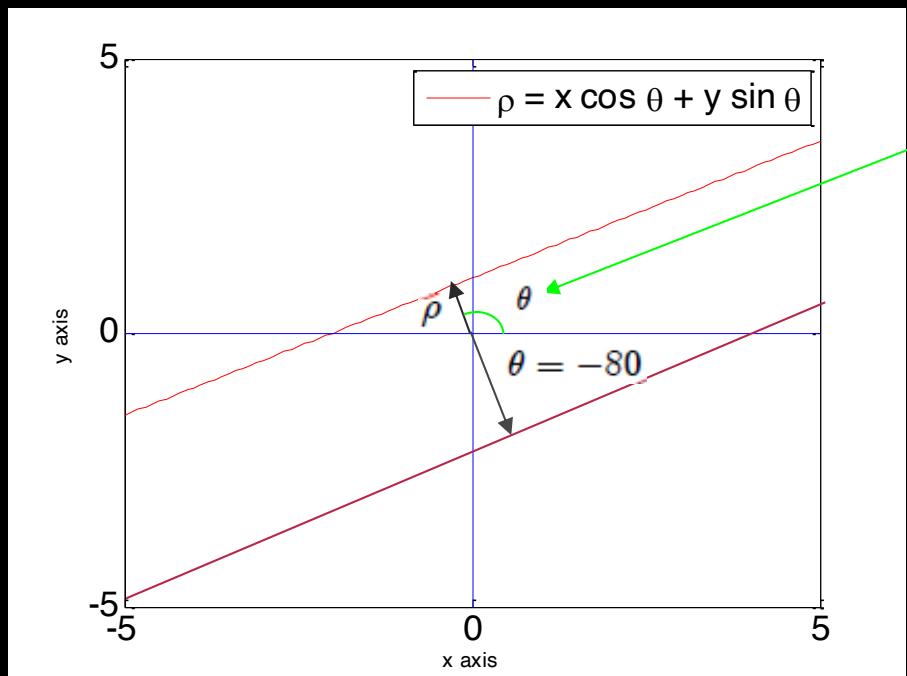
$\theta \in [-90^\circ, 90^\circ]$  In Python and in this presentation

# Hough Space

$$-90^\circ < \theta < 90^\circ$$



# More about angles



$$\theta = -80^\circ$$

Why?

$$\theta = 100^\circ$$

but Python only allows

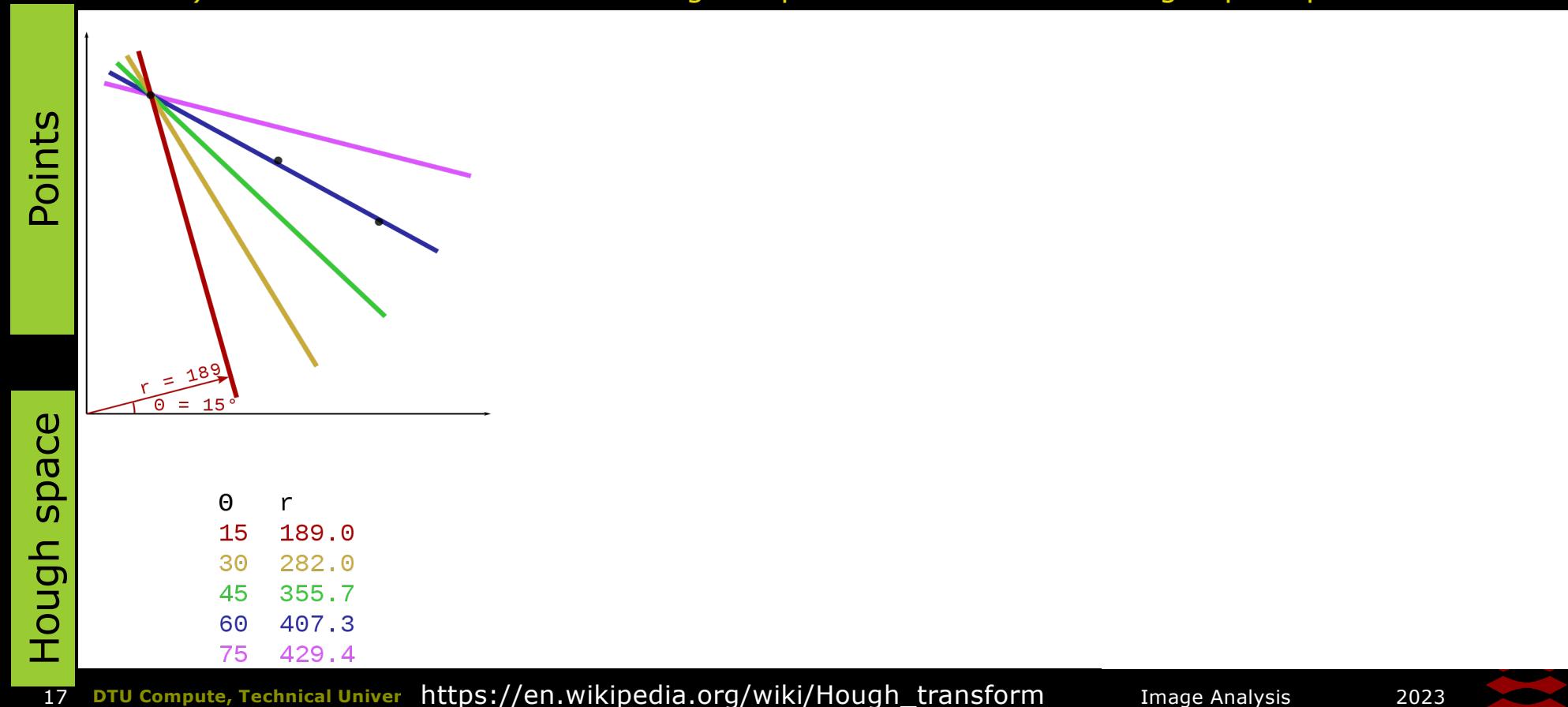
$$-90^\circ < \theta < 90^\circ$$

look at the mirror-projection of the normal

$\rho$  is used to determine if it is the “upper” or “lower line”

# Hough space: Let's vote for a general line

- A tool to find a line through points.
  - 1) Define the origin
  - 2) Select a point coordinate:  $(x, y)$
  - 3) For different  $\theta$ 's, map a line in the normal form through the selected point
  - 4) Map each line as parameters in the Hough space:  $(r, \theta)$  i.e.  $r = x \cos \theta + y \sin \theta$
  - 5) "Vote" which line fit best through all points: Have similar Hough space parameters





# Quiz 1: Hough space

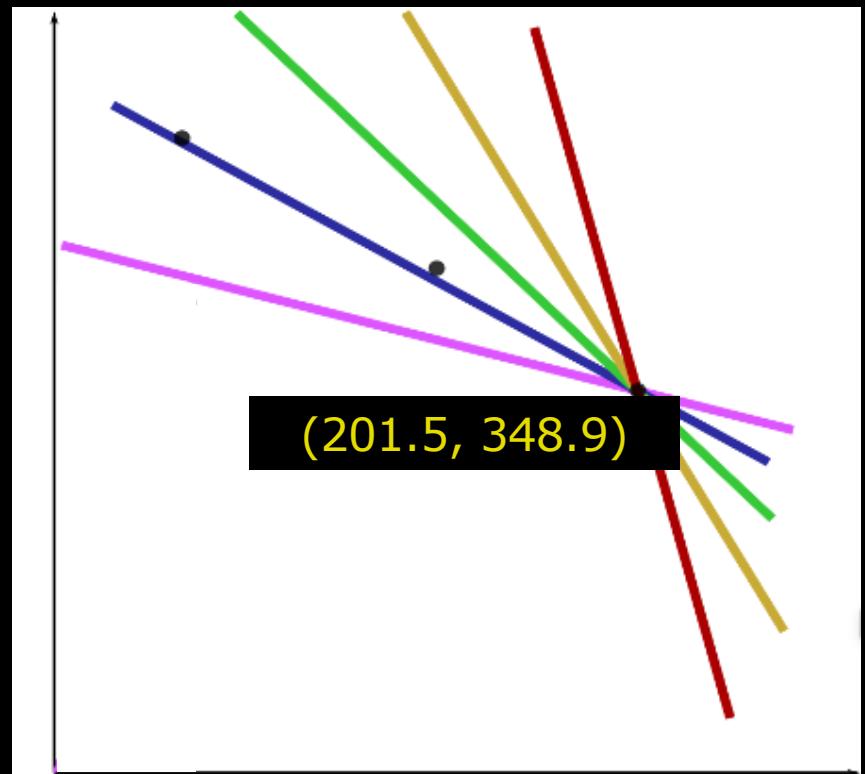
If we select  $\theta$  to 60 degree what is  $r$  when the point is  $(x,y)=(201.5, 348.9)$ ?

- A) 137.1
- B) 402.9
- C) -25.4
- D) 370
- E) -298.3

Solution:

$$x \cos \theta + y \sin \theta = r$$

$$\begin{aligned}201.5 \cos(60 * 0.0175) \\+ 348.9 \sin 60 * 0.0175 = 402.9\end{aligned}$$



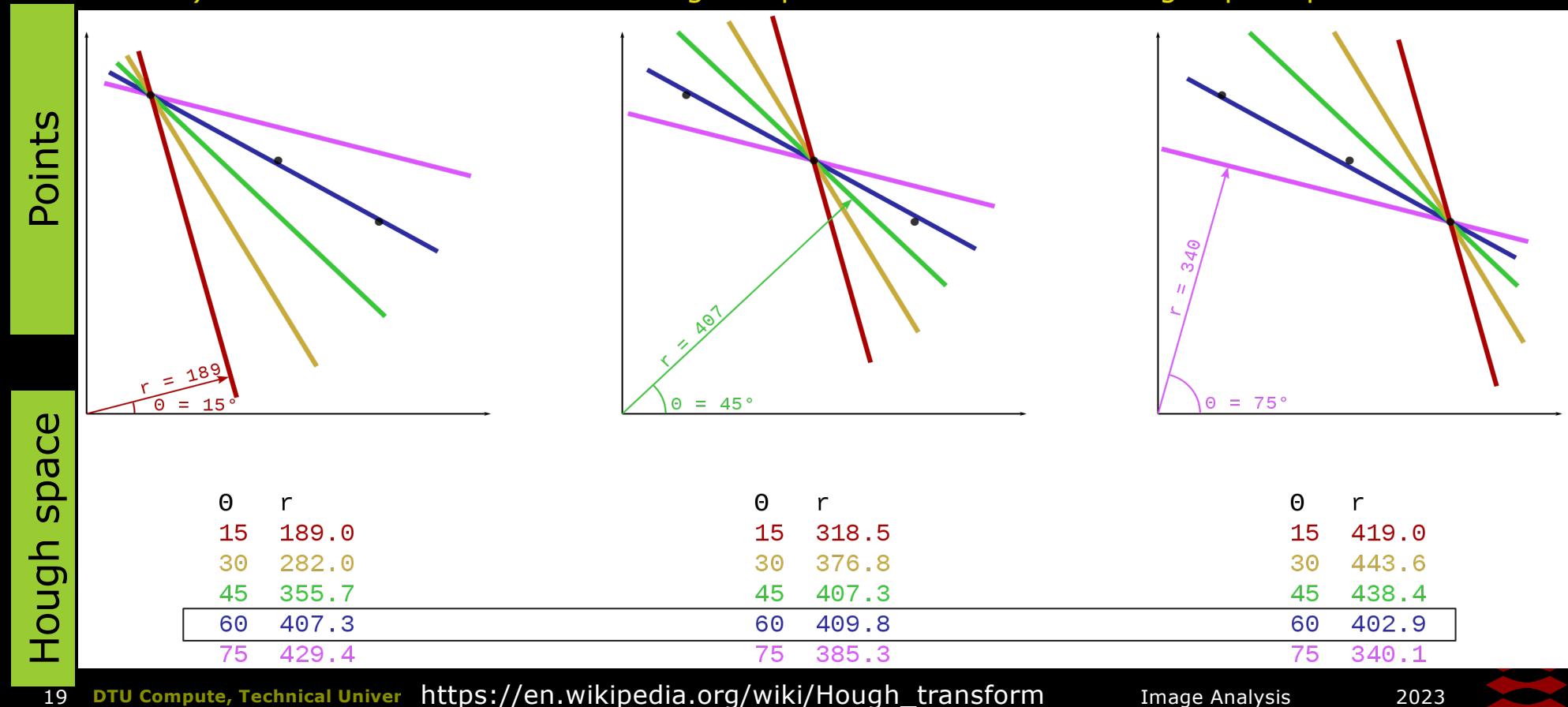
$\theta$	$r$
15	419.0
30	443.6
45	438.4
60	????
75	340.1



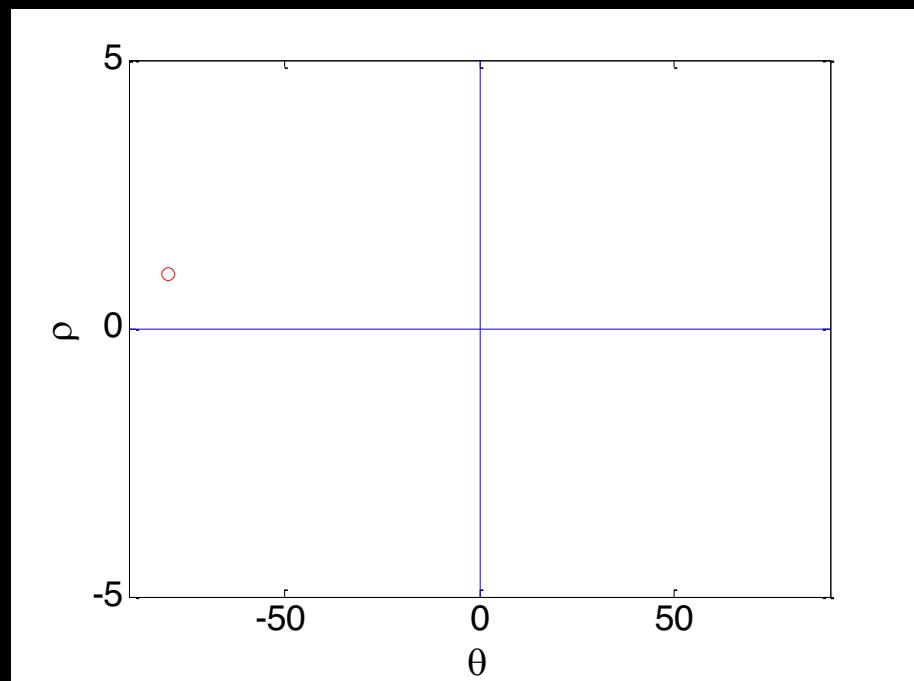
# Hough space

- A tool to find a line through points.

- 1) Define the origin
- 2) Select a point coordinate:  $(x, y)$
- 3) For different  $\theta$ 's, map a line in the normal form through the selected point
- 4) Map each line as parameters in the Hough space:  $(r, \theta)$  i.e.  $r = x \cos \theta + y \sin \theta$
- 5) "Vote" which line fit best through all points: Have similar Hough space parameters

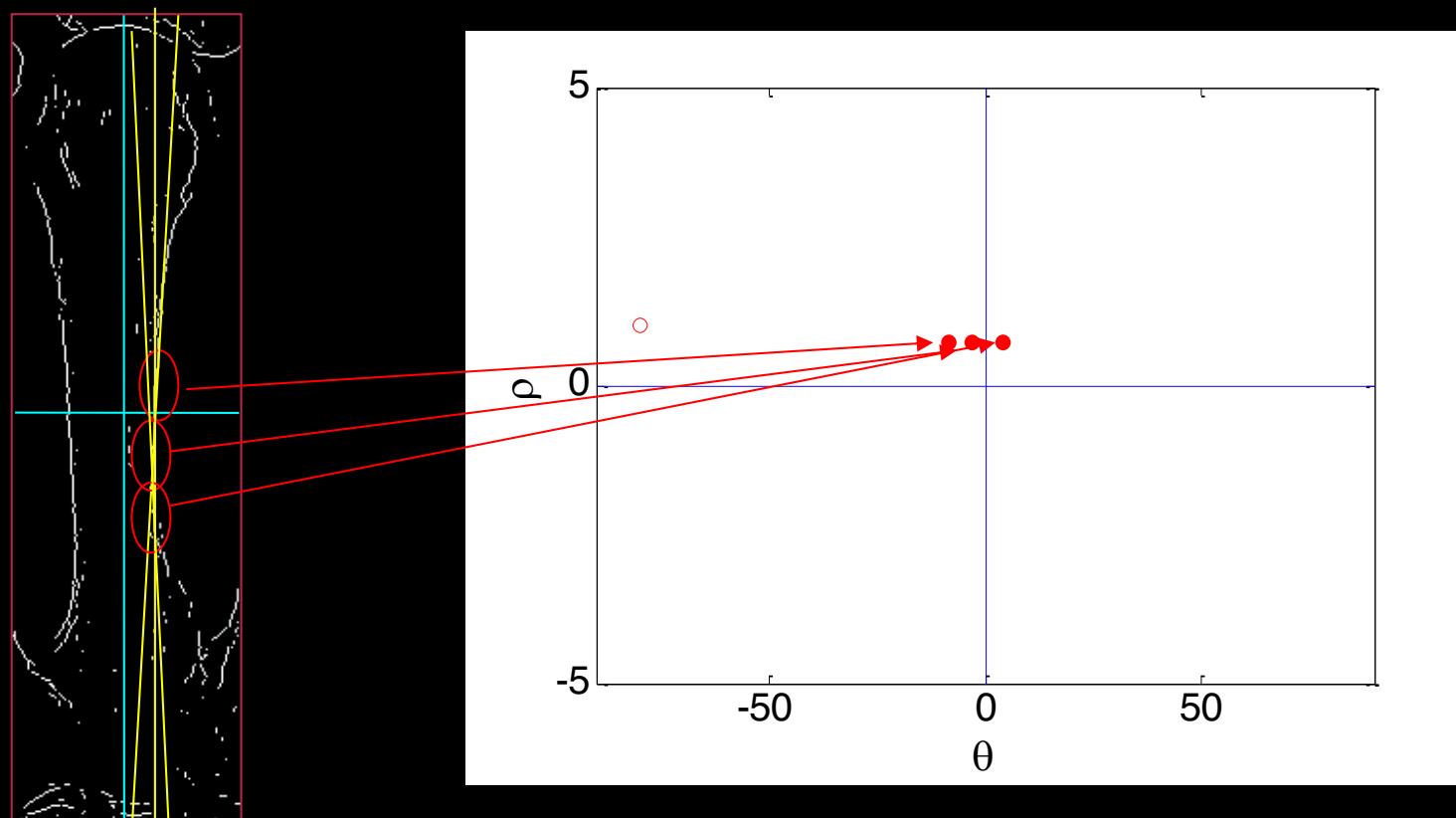


# How do we use the Hough space?



# How do we use the Hough space?

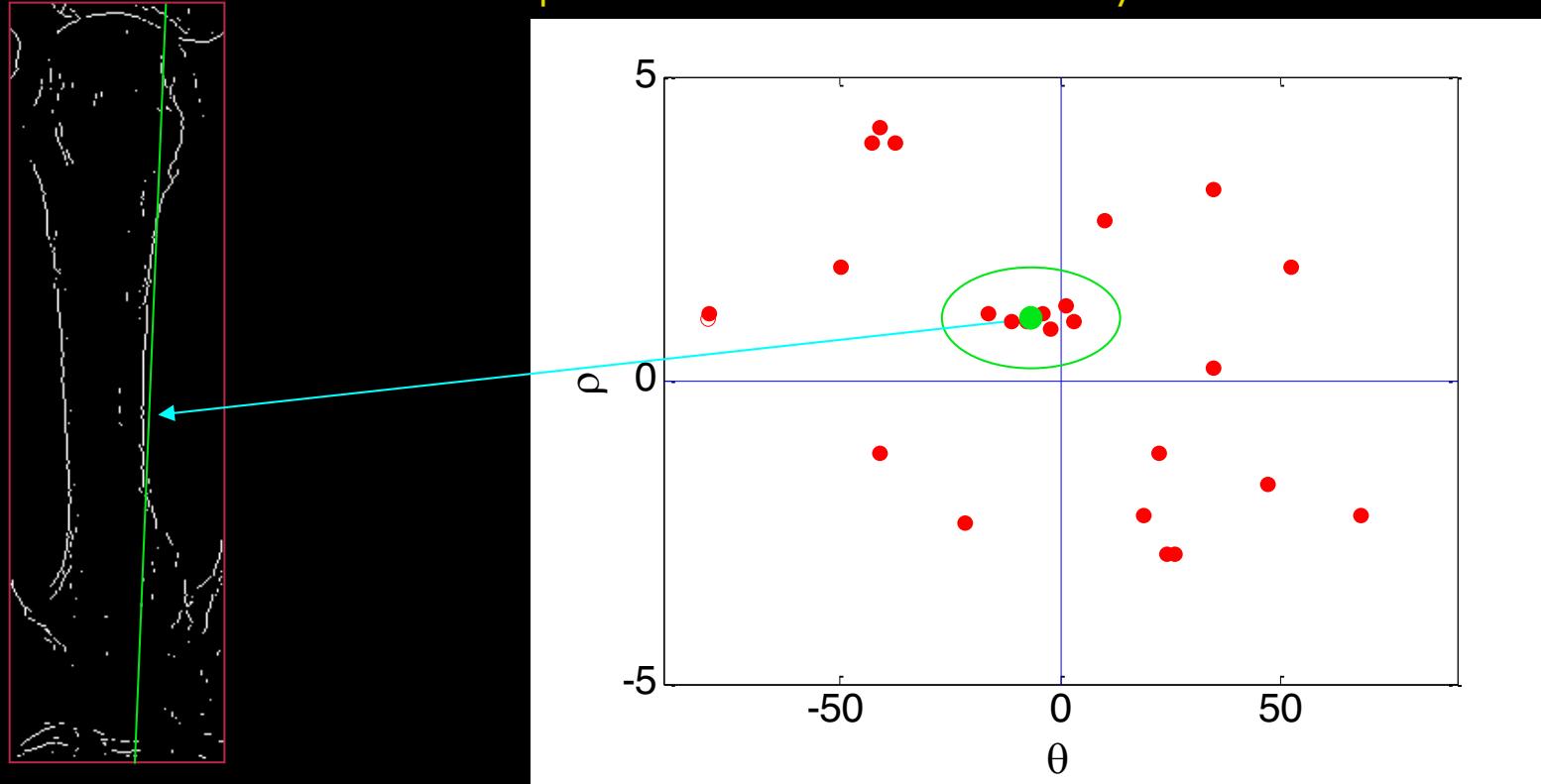
- What if every little “line-segment” was plotted in the Hough-space?



# Filled Hough-Space

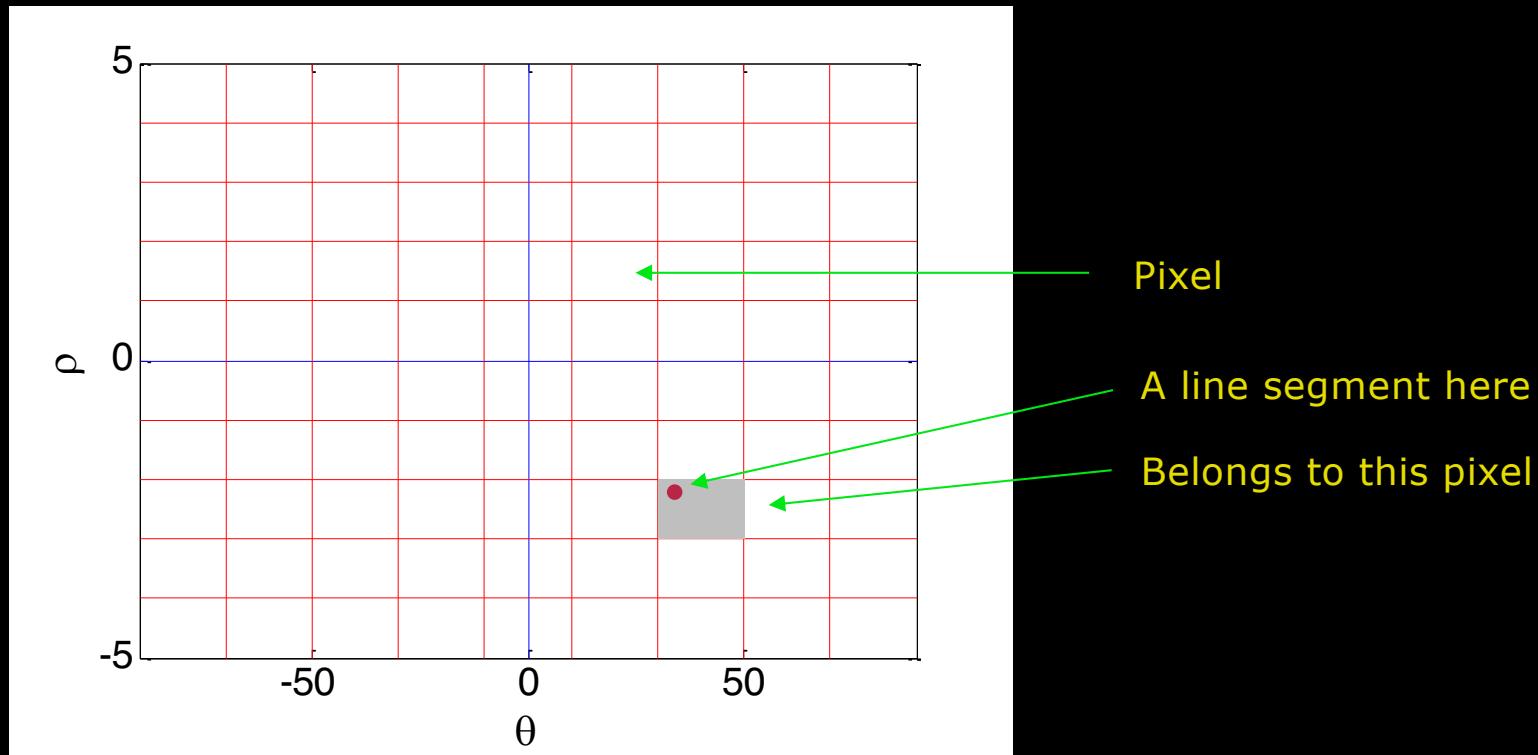
- All “line segments” in the image examined
- A “global line” can now be found as a cluster of points

In practice it is difficult to identify clusters



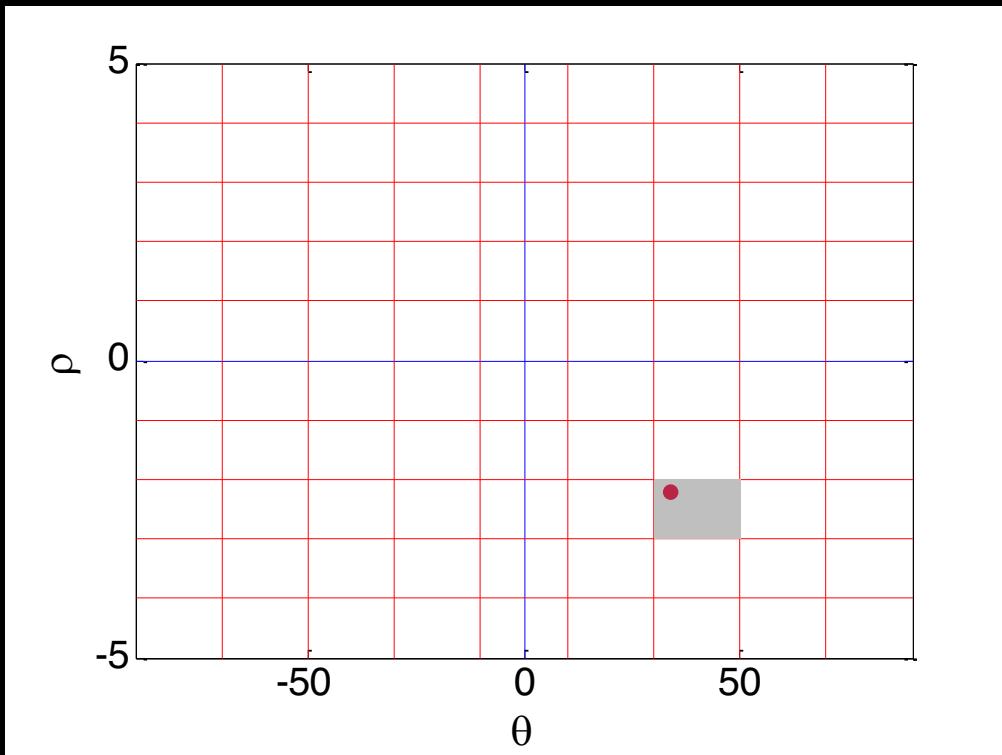
# Hough transform in practise

- Hough Space is represented as an image
- It is *quantized* – made into finite boxes



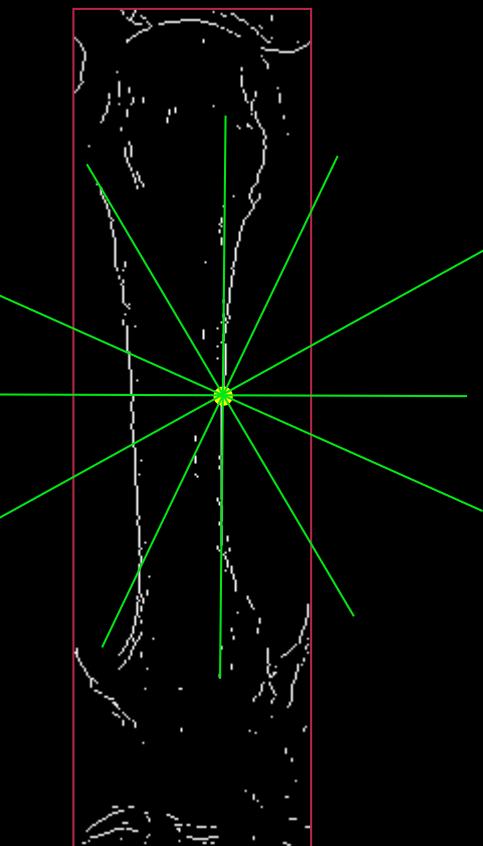
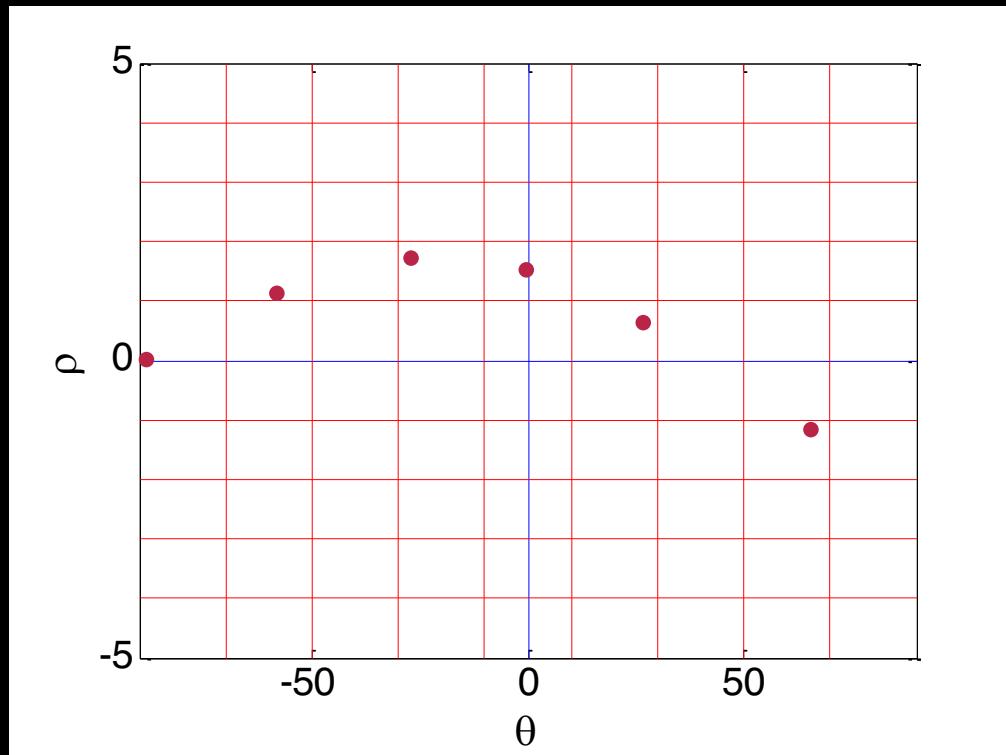
# Hough transform as a voting scheme

- The pixels in the Hough space are used to *vote* for lines.
- Each *line segment* votes by putting *one vote* in a pixel
- The pixels are also called *accumulator cells*



# Hough transform per pixel

- In practise we do not use line segments
- Each pixel in the input image votes for **all** potential lines going through it.



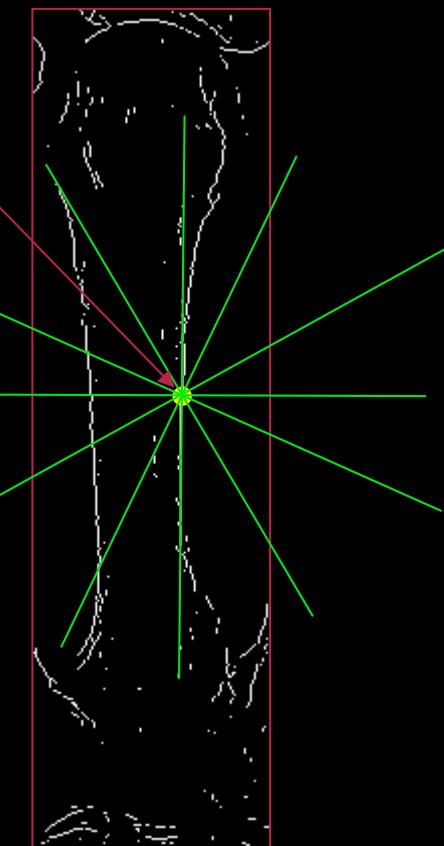
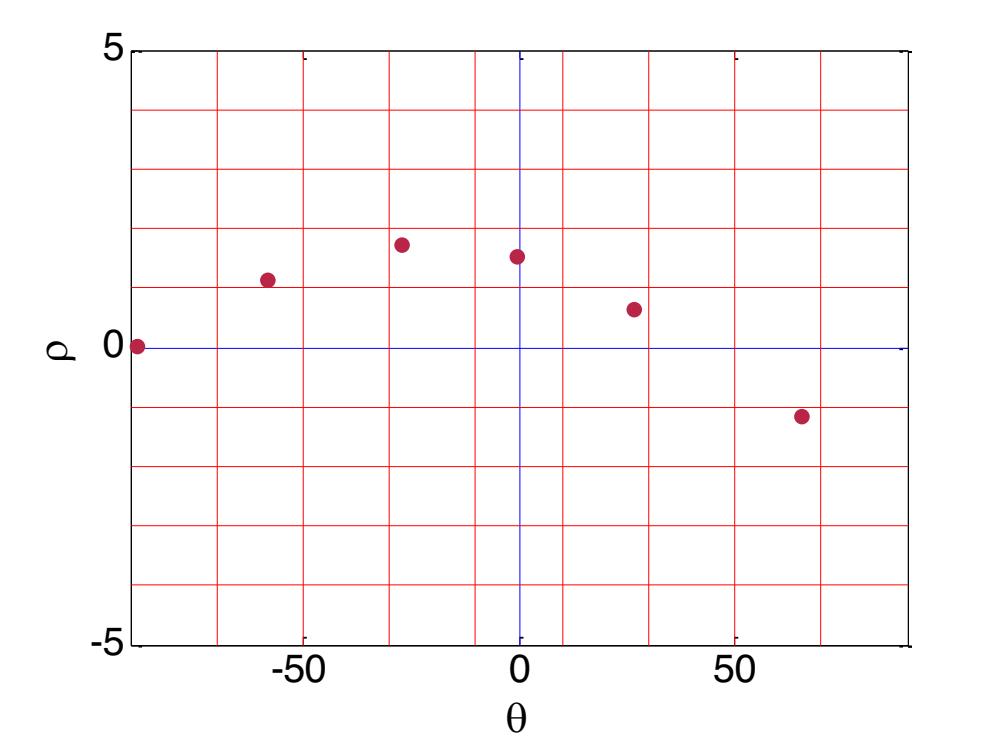
# Hough transform per pixel

$$x \cos \theta + y \sin \theta = \rho$$

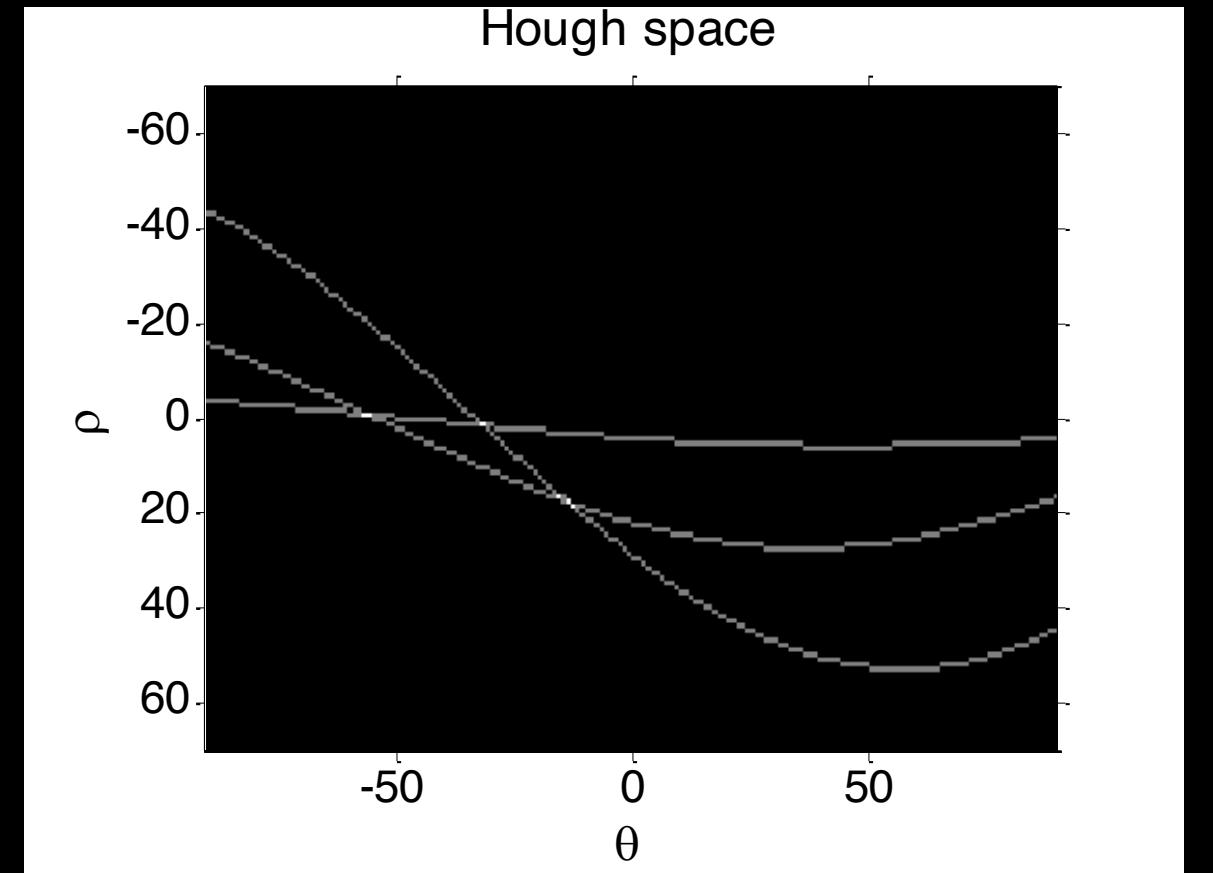
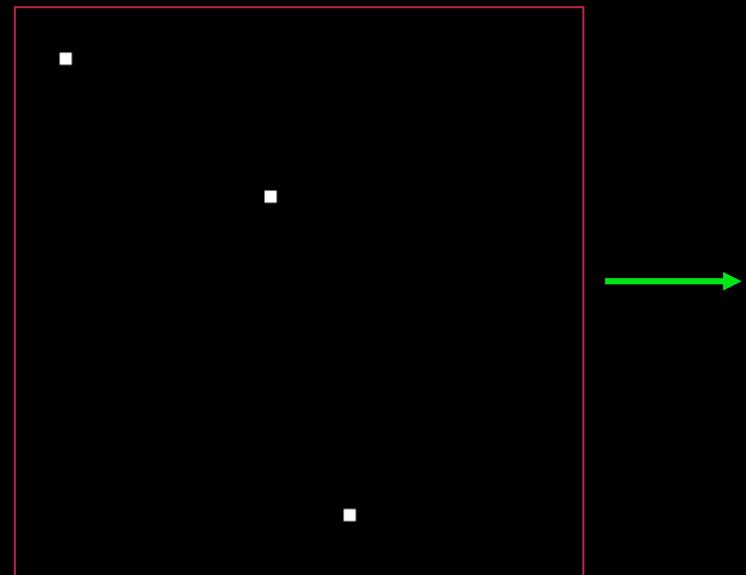
Go through all  $\theta$  and calculate  $\rho$

( $x, y$ ) are fixed

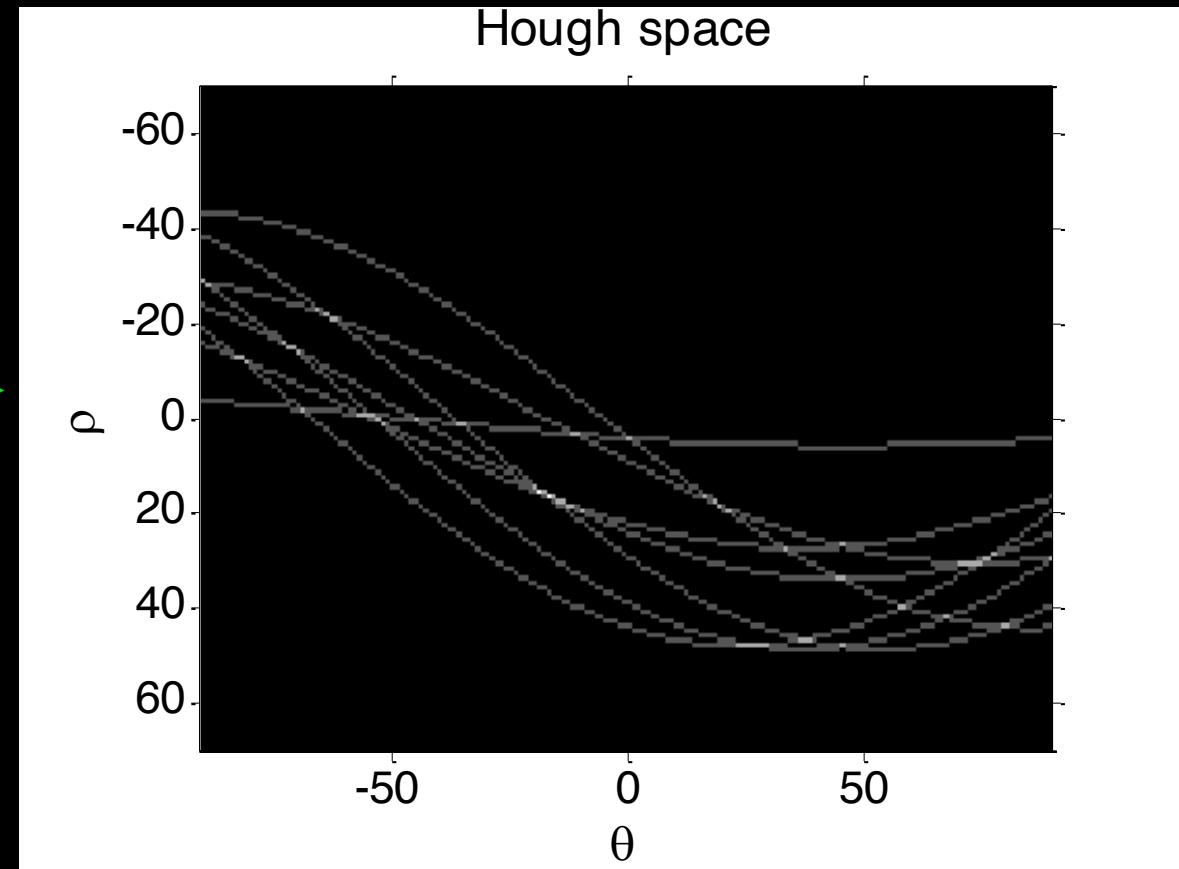
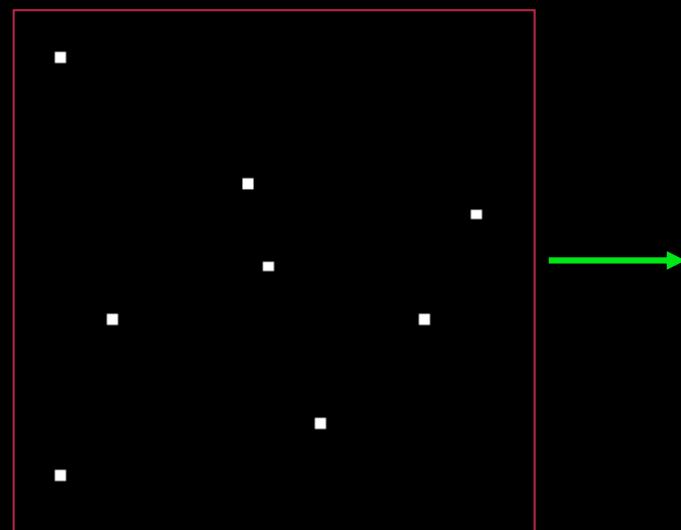
Sinusoid!



# Real Hough Transform



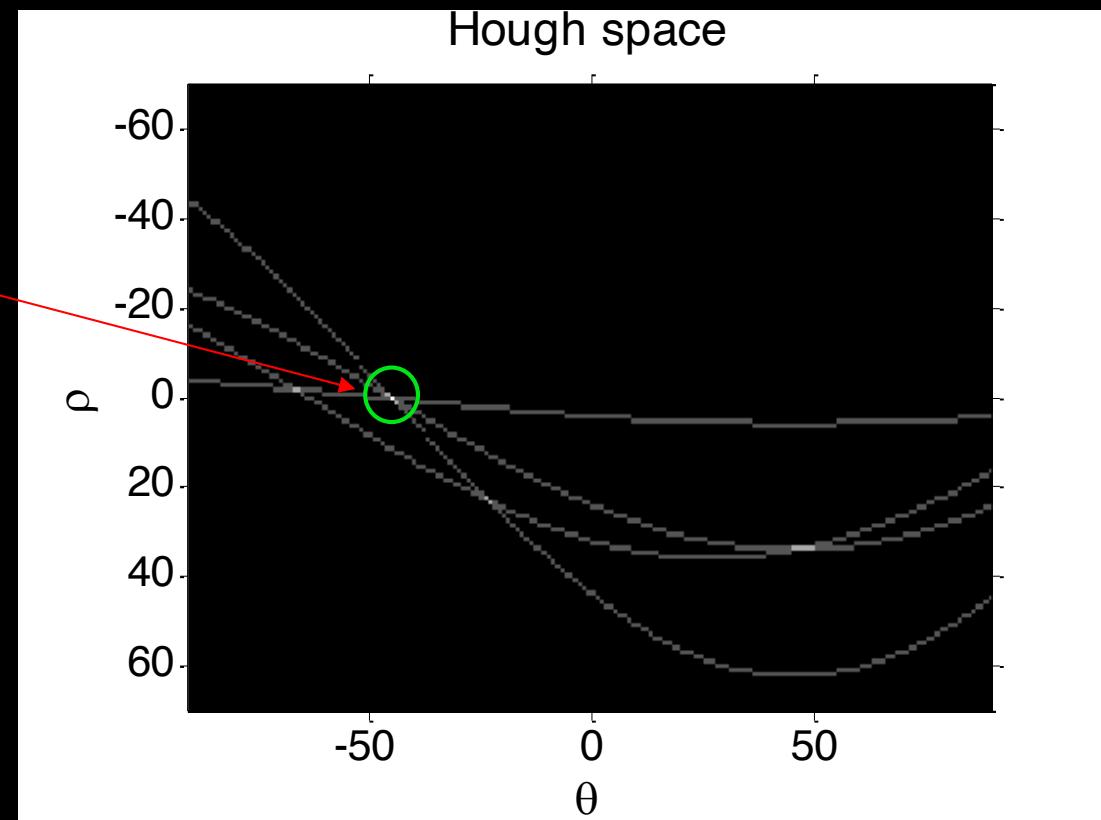
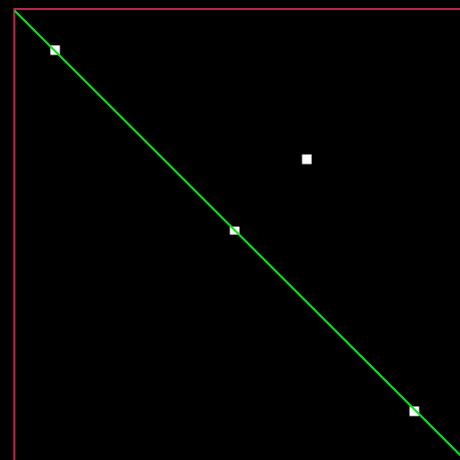
# Real Hough Transform II



# Real Hough Transform and lines

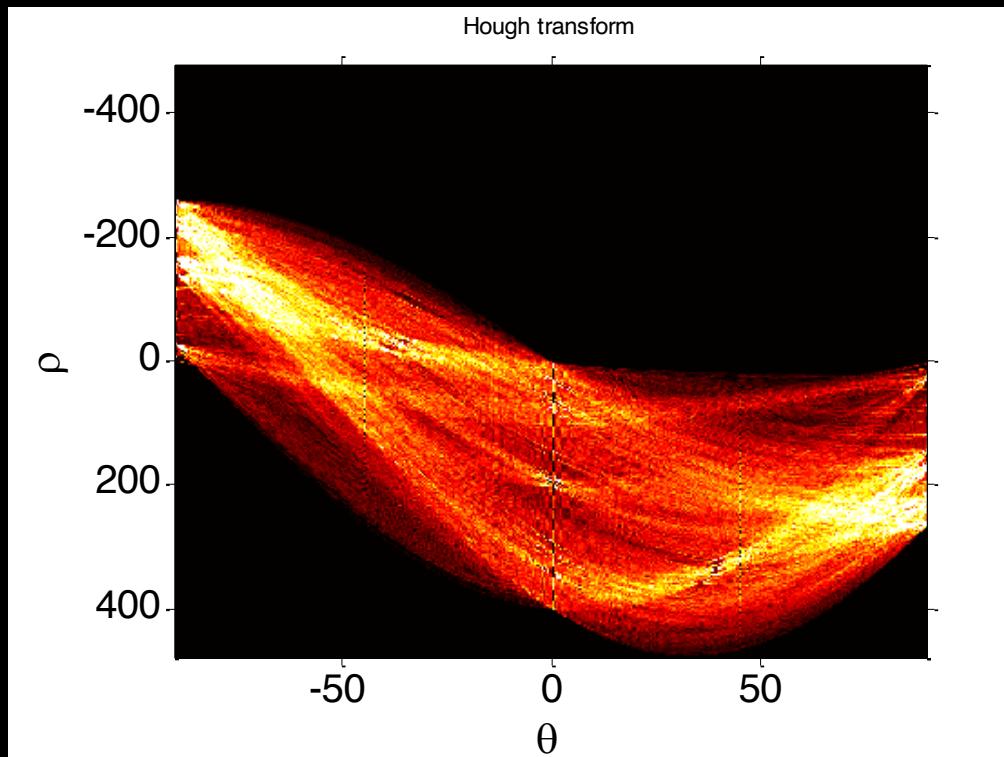
Spot the line!

A maximum where Hough pixel  
has value 3

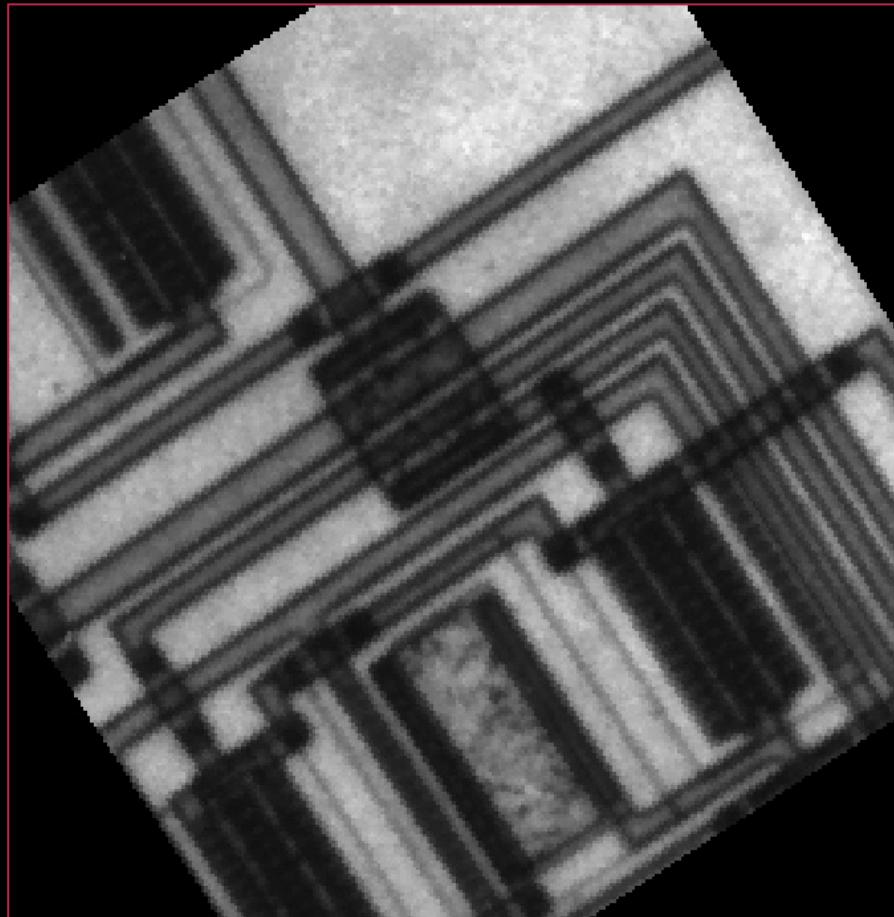


# Finding the lines in Hough space

- The lines are found in Hough space where *most pixels have voted for there being a line*
- Can be found by searching for maxima in Hough Space

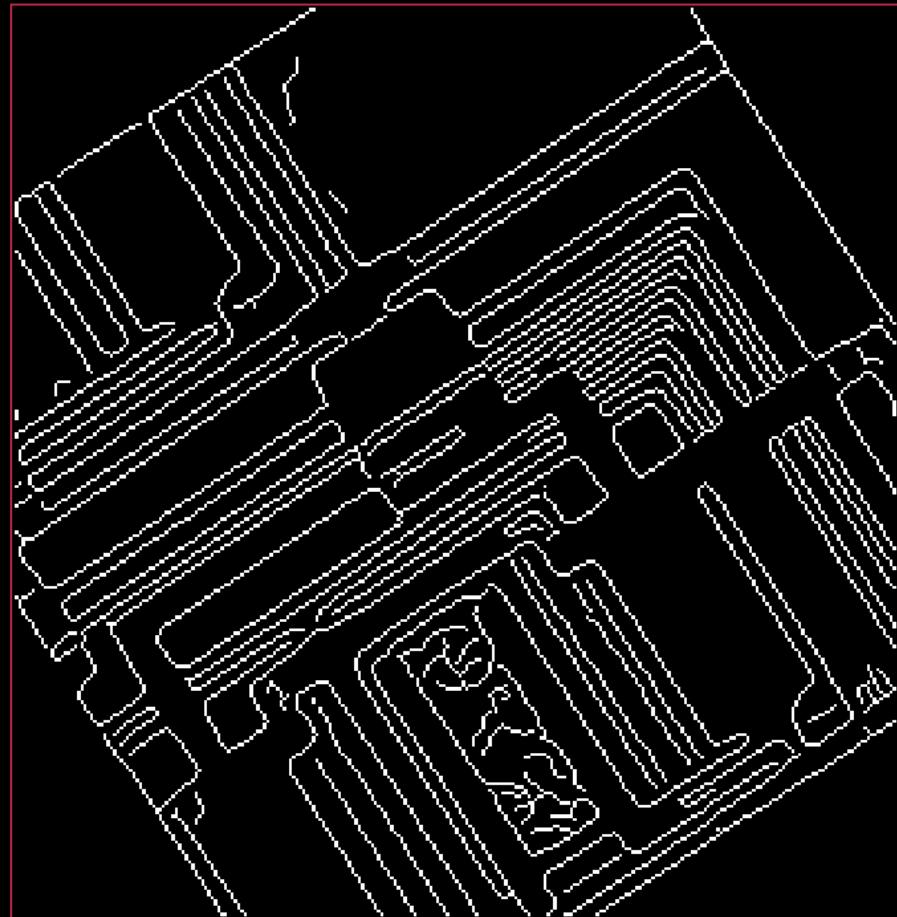


# The practical guide to the Hough Transform



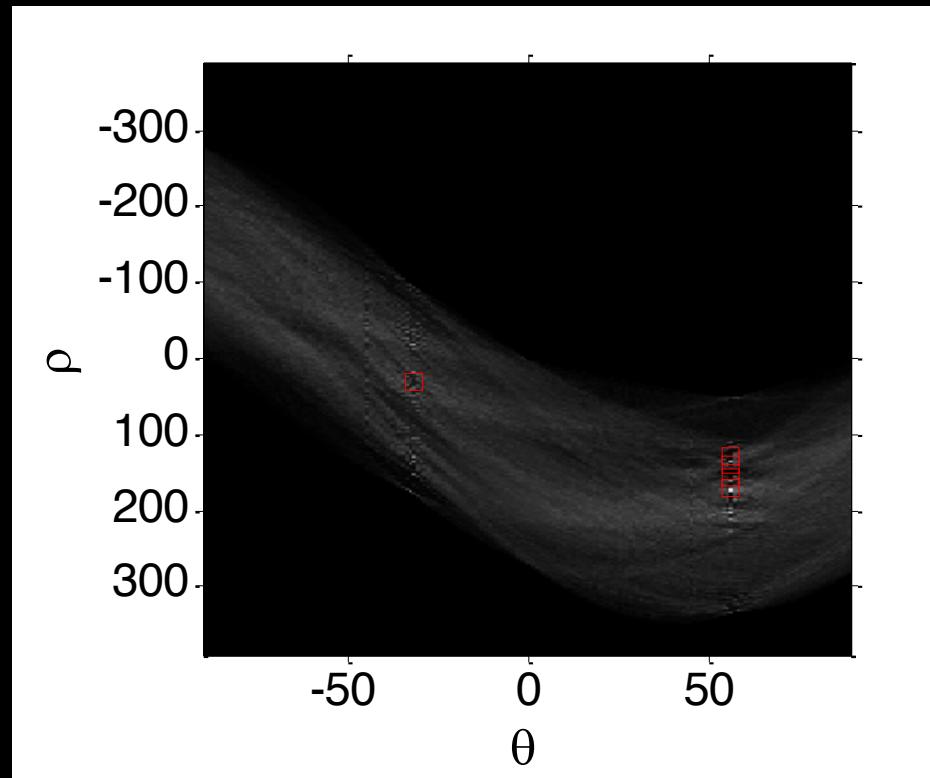
- Start with an input image

# The practical guide to the Hough Transform



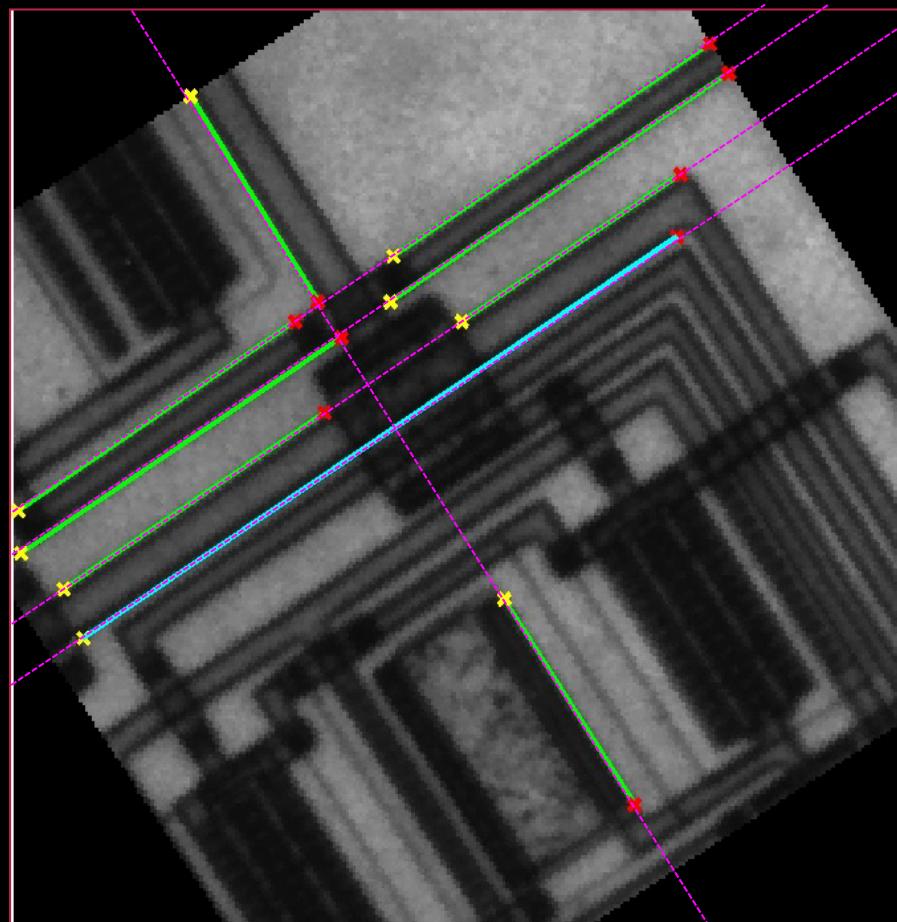
- Detect edges and create a binary image

# The practical guide to the Hough Transform



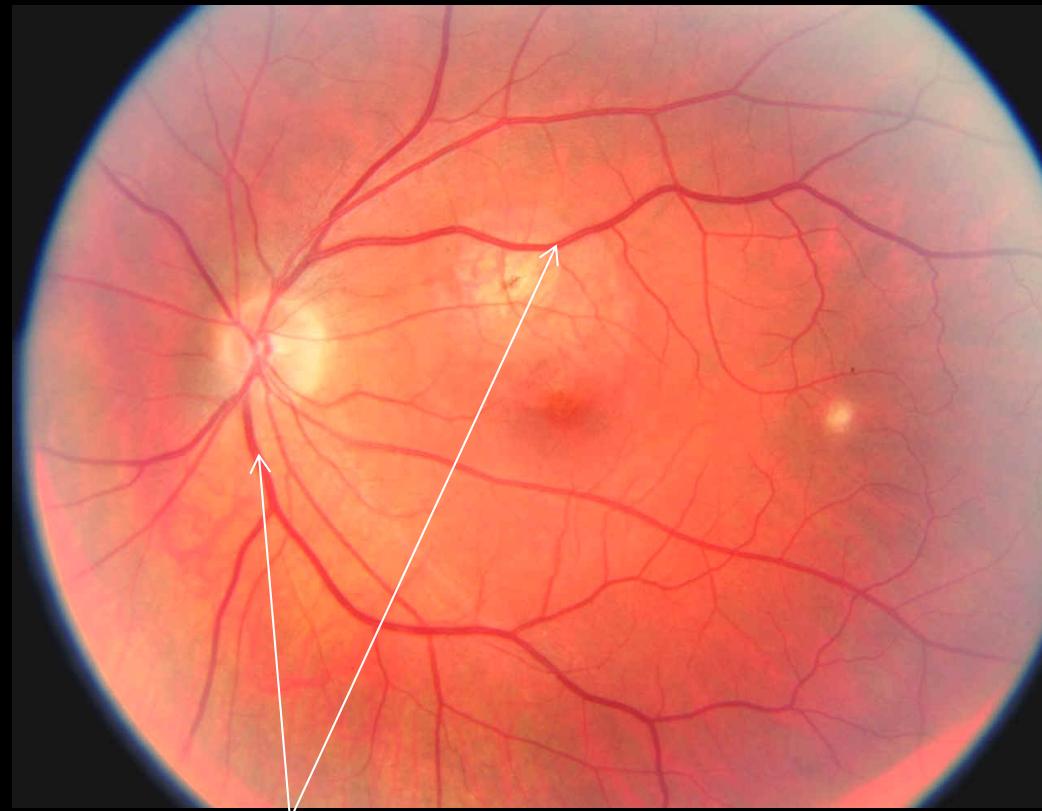
- Compute Hough transform and locate the maxima
- We select the 5 highest points

# The practical guide to the Hough Transform



- Draw the 5 lines corresponding to the found maxima (purple)
- The full lines (green)
- Here the **cyan** line is the longest

# Path Tracing

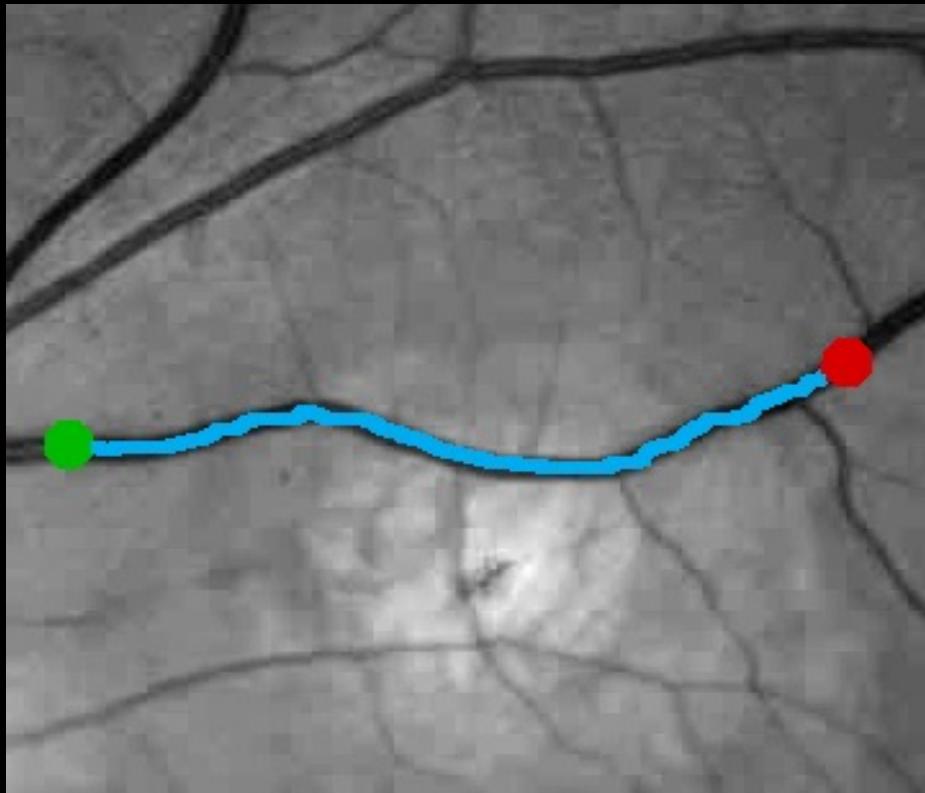


Arteries and veins

Fundus image

- The diameter as function of the distance to the optic cup tells something about the patients health
- We need to find the arteries and veins
- Path tracing is one solution

# Path tracing



- A path is defined as a curve in an image defined as *something that is different from the background*
- In this case it is a dark line
- Pre-processing can for example turn edges into dark lines.

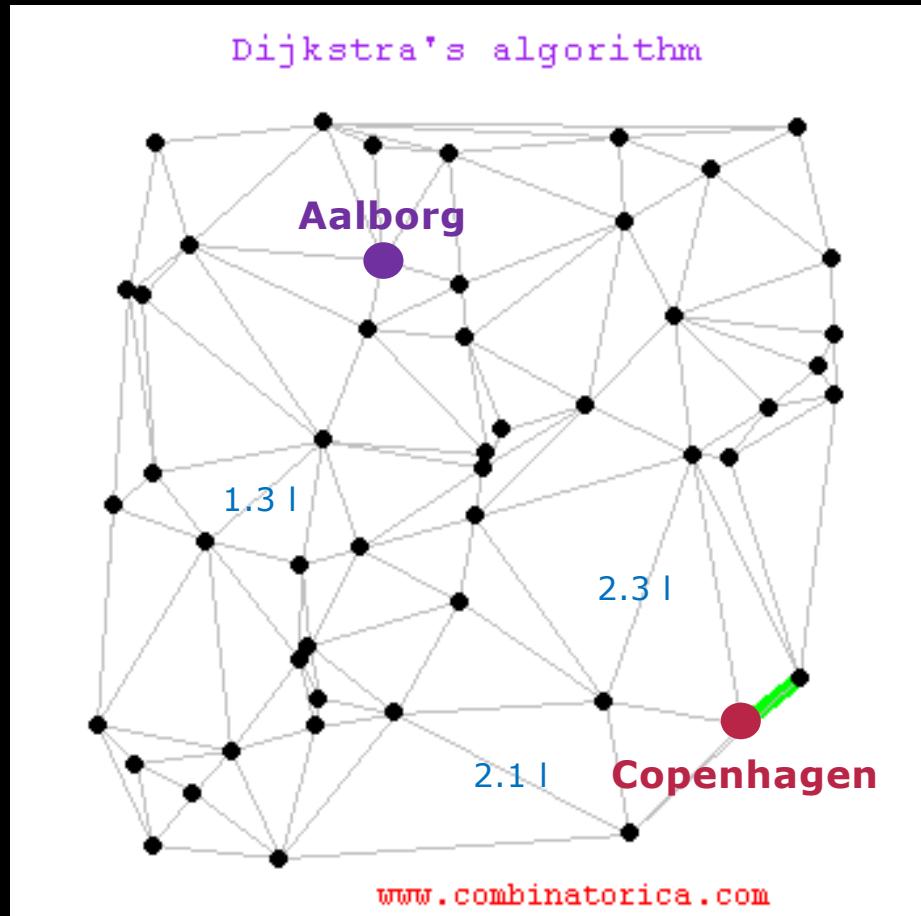
# Dynamic Programming



- Break up large problem into many small sub-problems
- A classic algorithm:
  - Dijkstra's algorithm
  - One source to all nodes shortest path
- We will look at a simplified variant

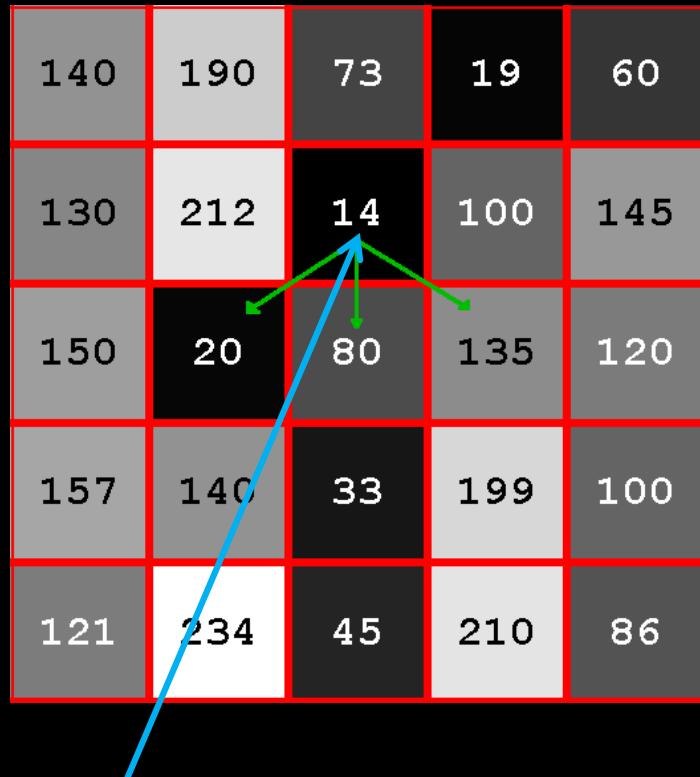
Dijkstra, E. W. (1959). "A note on two problems in connexion with graphs". Numerische Mathematik. 1: 269–271.

# Path tracing



- A GPS device uses path tracing
- Based on *graph algorithms*
  - A city is a node
  - A road is an edge. The weight of the edge is the fuel cost
- How do we come from Copenhagen to Aalborg using the least fuel?
- Dijkstra's algorithm

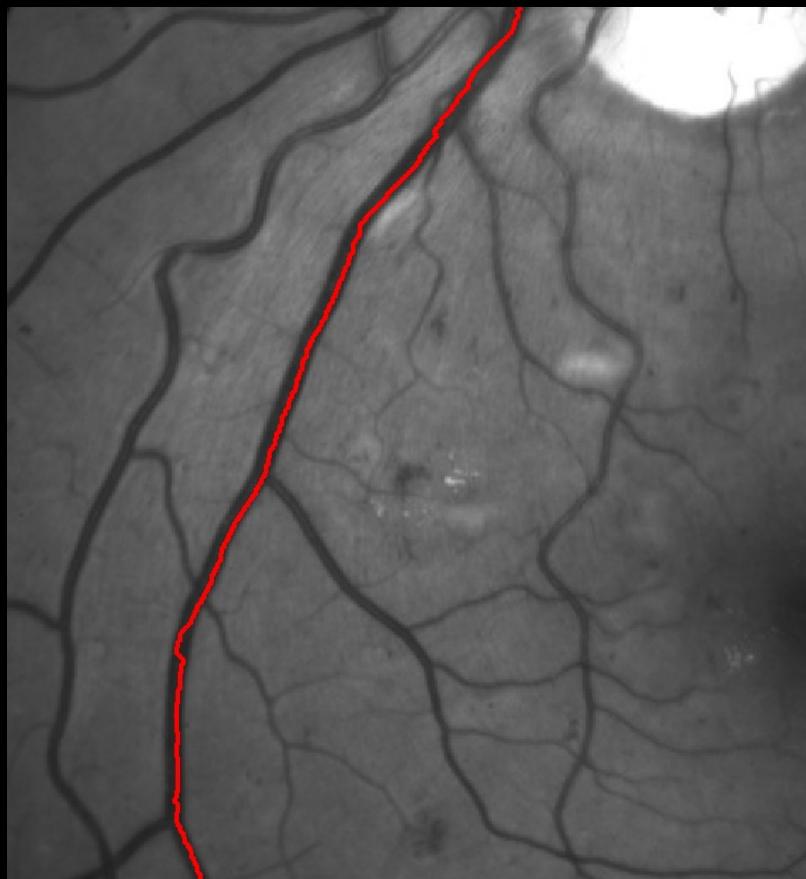
# Images as graphs



$$\mathcal{C}(2, 3) = 14$$

- Each pixel is a node
- Pixel neighbours are connected by edges
- The edge cost ( $c(r, c)$ ) is the pixel value
- Directed graph
- Imagine a car driving on the image
- Called a *cost image*

# Simplified problem



- Track dark lines
- Path going from top to bottom
- No sharp turns – smooth
- Problem:
  - from the top to the bottom
  - Sum of pixel values should be minimal

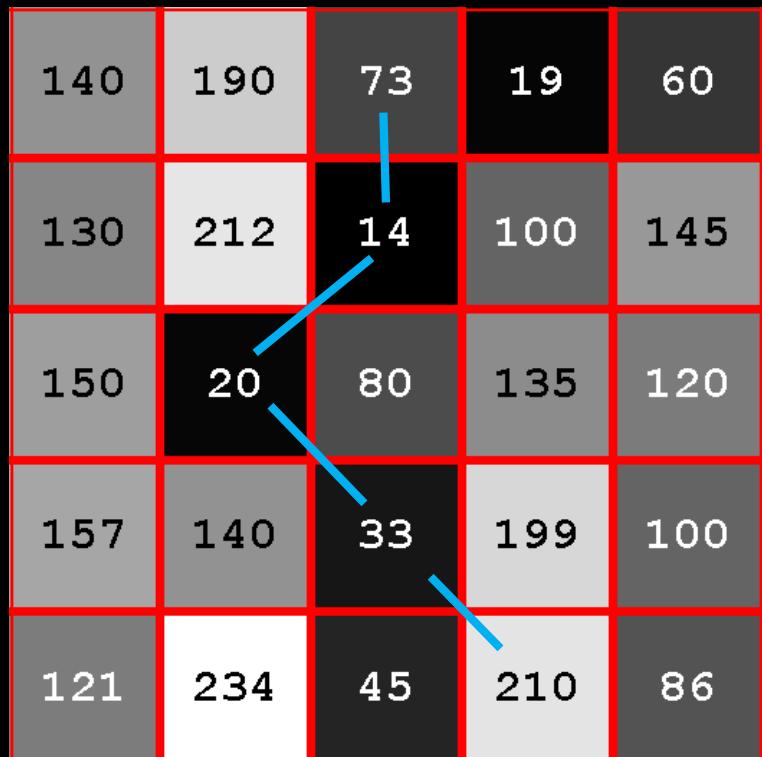
# Simplified problem



- Pixel value at  $(r,c)$  equals the cost  $C(r,c)$
- The path  $P$  consist of pixels
- The sum of pixel values in the path

$$C_{tot} = \sum_{(r,c) \in \mathcal{P}} C(r,c)$$

# Path cost



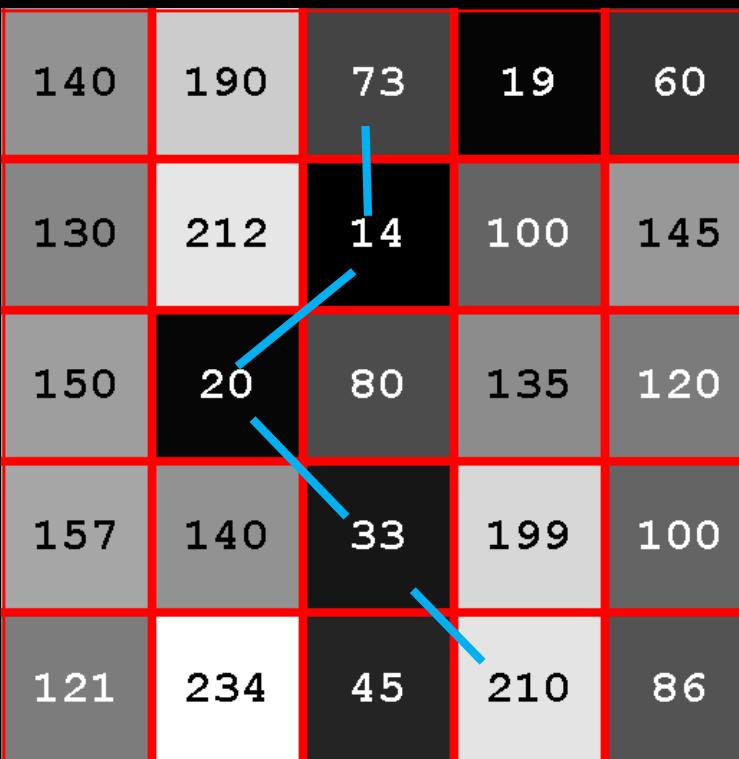
$$P = [(1,3), (2,3), (3, 2), (4,3), (5,4)]$$

- A path is defined as  $(r,c)$  coordinates

$$C_{tot} = \sum_{(r,c) \in \mathcal{P}} C(r, c)$$

## Quiz 2: Total cost – what is $C_{tot}$ ?

- A) 167
- B) 350
- C) 403
- D) 270
- E) 345



140	190	73	19	60
130	212	14	100	145
150	20	80	135	120
157	140	33	199	100
121	234	45	210	86

$$P = [(1,3), (2,3), (3, 2), (4,3), (5,4)]$$

# Path cost

140	190	73	19	60
130	212	14	100	145
150	20	80	135	120
157	140	33	199	100
121	234	45	210	86

$$P = [(1,3), (2,3), (3, 2), (4,3), (5,4)]$$

- This is *NOT* the optimal path
- How do we compute the path  $P$  that has minimum  $C_{tot}$ ?
- Test all possible paths?
  - No! Impossible amount of possibilities

## Quiz 3: Path Cost

- A) 196
- B) 154
- C) 201
- D) 185
- E) 132

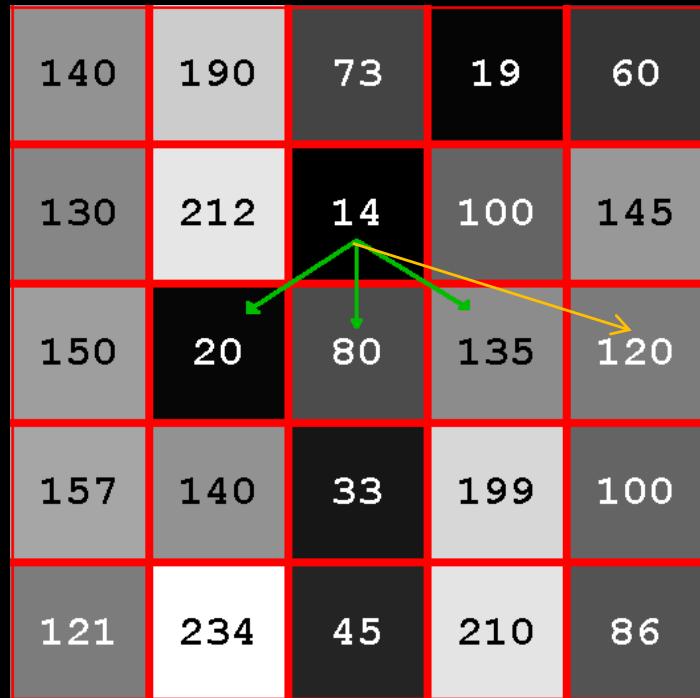
A path has been found in the image  
 $P=[(1,4),(2,4),(3,5),(4,5),(5,5),(6,4)]$ .

What is the total cost of the path?

208	157	234	19	145	79
62	121	73	14	120	135
237	90	193	135	3	42
89	212	192	199	86	154
50	149	97	238	41	67
64	140	145	33	203	167

Figur 1: Grayscale billede

# Path restriction: The rules



- Path is only allowed to
  - Go down
  - Move one pixel left or right
- Longer jumps not allowed

# Accumulator image

140	190	73	19	60
270	285	33	119	164
420	53	113	168	239
210	193	86	312	265
314	320	131	296	354

- Keeps track of the accumulated cost for efficient paths finding
- Path ending here has cost 296
- We use 5 steps to find the shortest path

# Computing the accumulator image

140	190	73	19	60
130	212	14	100	145
150	20	80	135	120
157	140	33	199	100
121	234	45	210	86

140	190	73	19	60
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0

Step 1: Copy first row of input image

# Computing the accumulator image

140	190	73	19	60
130	212	14	100	145
150	20	80	135	120
157	140	33	199	100
121	234	45	210	86

Step 2: Fill second row

$$\mathbf{A}(r, c) = \mathbf{I}(r, c) + \min (\mathbf{A}(r - 1, c - 1), \mathbf{A}(r - 1, c), \mathbf{A}(r - 1, c + 1))$$

140	190	73	19	60
270	285	33	119	164
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0

# Computing the accumulator image

140	190	73	19	60
130	212	14	100	145
150	20	80	135	120
157	140	33	199	100
121	234	45	210	86

140	190	73	19	60
270	285	33	119	164
420	53	113	168	239
210	193	86	312	268
314	320	131	296	354

Step 3: Fill all rows by looking at the previous row

$$\mathbf{A}(r, c) = \mathbf{I}(r, c) + \min(\mathbf{A}(r - 1, c - 1), \mathbf{A}(r - 1, c), \mathbf{A}(r - 1, c + 1))$$



# Quiz 4: Accumulator Image

- A) 57
- B) 167
- C) 301
- D) 241
- E) 145

An optimal path has been found in the image.  
What is the value of the accumulator image in the marked pixel?

117	163	74	210
223	244	171	57
132	61	110	170
241	172	17	215

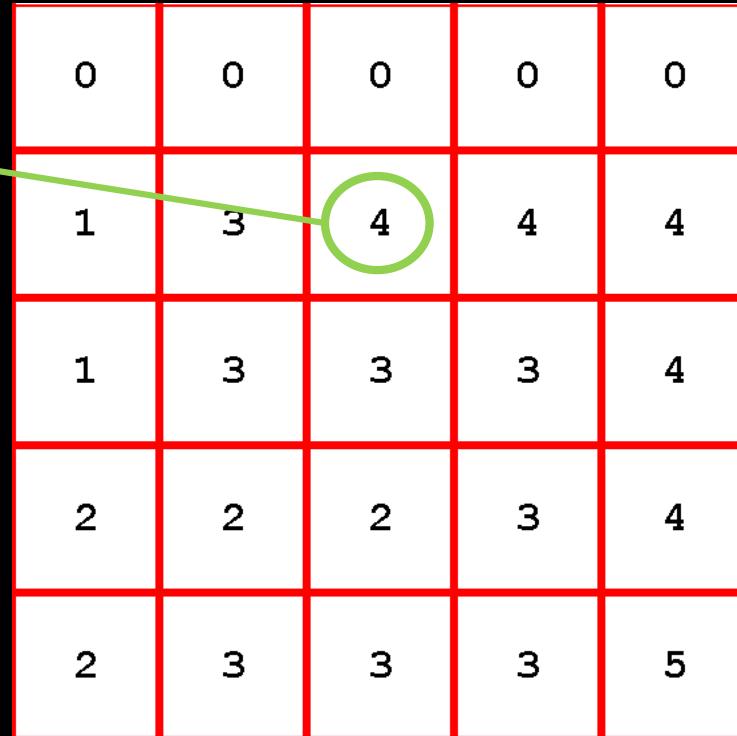
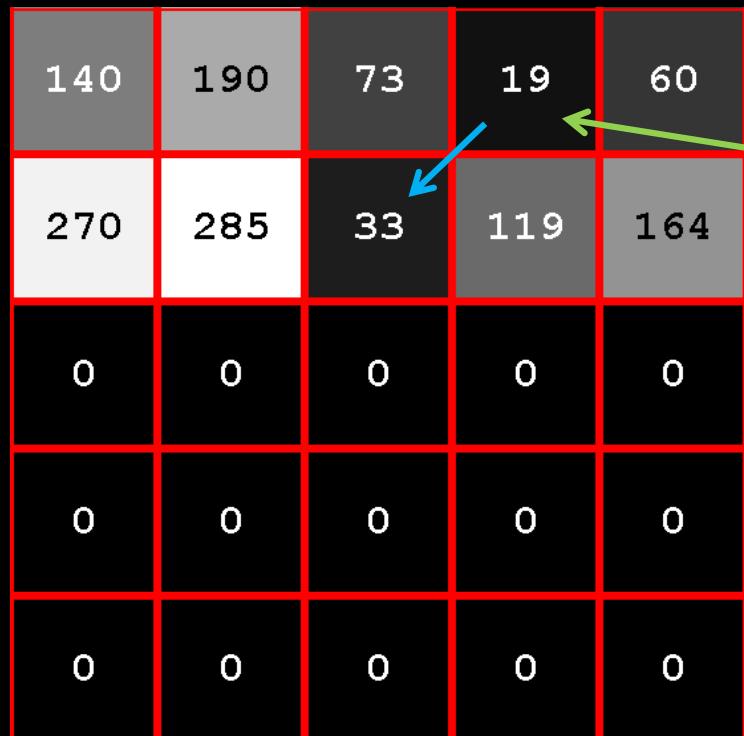
# Using the accumulator image

140	190	73	19	60
130	212	14	100	145
150	20	80	135	120
157	140	33	199	100
121	234	45	210	86

140	190	73	19	60
270	285	33	119	164
420	53	113	168	239
210	193	86	312	268
314	320	131	296	354

Step 4: The end of the optimal path can now be found

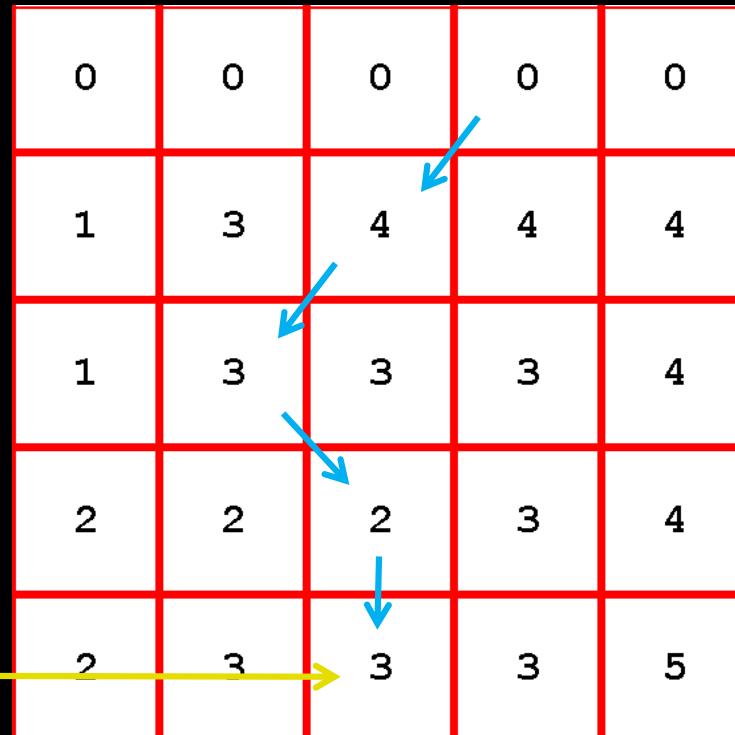
# The backtracing image



- Keeps track of where the path *came* from
- Each pixel stores the column number

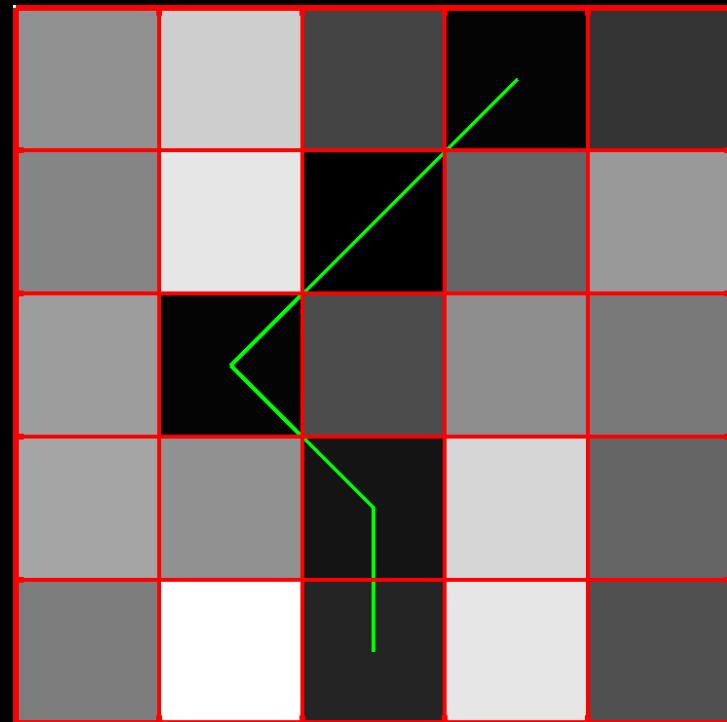
# Using the backtracing image

140	190	73	19	60
270	285	33	119	164
420	53	113	168	239
210	193	86	312	268
314	320	131	296	354



Step 5: Trace the path in the backtracing image

# Using the backtracing image



0	0	0	0	0
1	3	4	4	4
1	3	3	3	4
2	2	2	3	4
2	3	3	3	5

# Quiz 5: Backtracing

- A) 1
- B) 2
- C) 3
- D) 4
- E) 5

An optimal path has been found in an image. The backtracing image is seen below and the optimal path ends in the marked pixel.

What is the optimal path?

0	0	0	0	0
1	3	3	3	5
2	2	2	4	4
1	1	4	5	5
1	1	4	4	4

1.  $\mathcal{P} = [(1, 3), (2, 2), (3, 1), (4, 1), (5, 2)]$
2.  $\mathcal{P} = [(1, 3), (2, 2), (3, 2), (4, 2), (5, 2)]$
3.  $\mathcal{P} = [(1, 2), (2, 2), (3, 2), (4, 1), (5, 2)]$
4.  $\mathcal{P} = [(1, 3), (2, 1), (3, 1), (4, 1), (5, 2)]$
5.  $\mathcal{P} = [(1, 2), (2, 1), (3, 1), (4, 2), (5, 2)]$

# Pre-processing



- We would like to track paths that are not dark curves

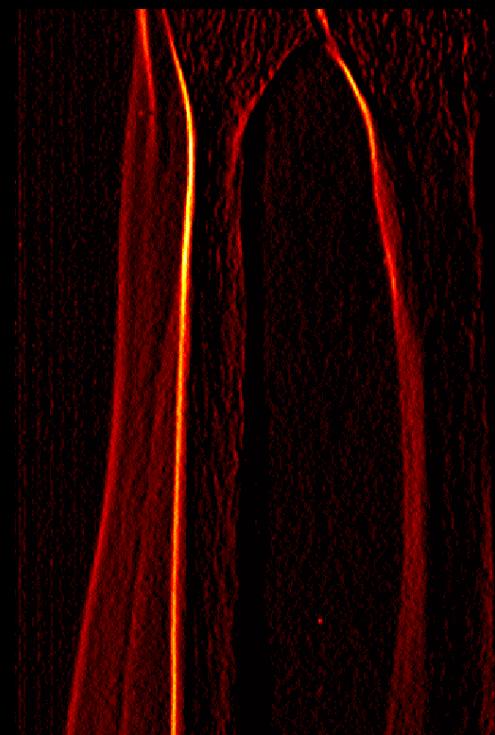


## Quiz 6 : X-ray preprocessing

- A) Gaussian smoothing
- B)  $255 - I$
- C) Gradient filter
- D) Registration
- E) Morphological operation

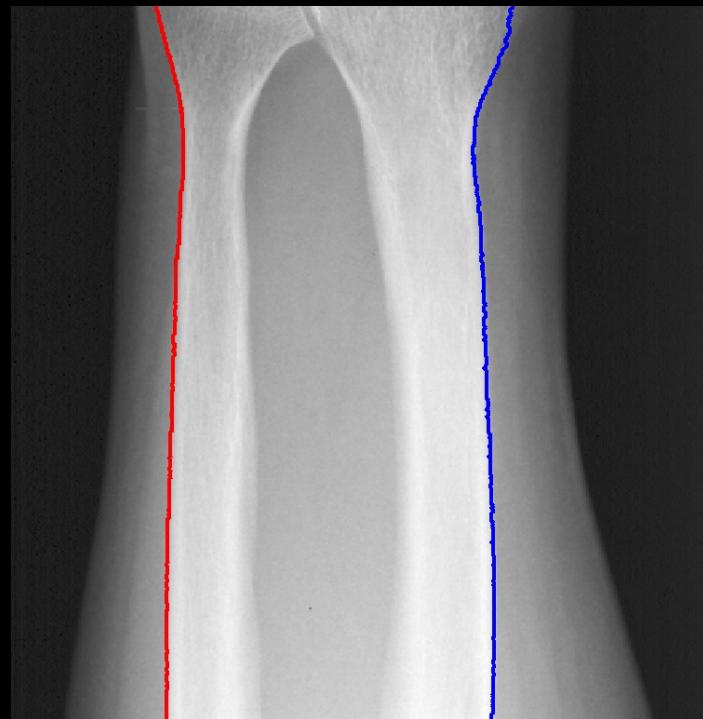
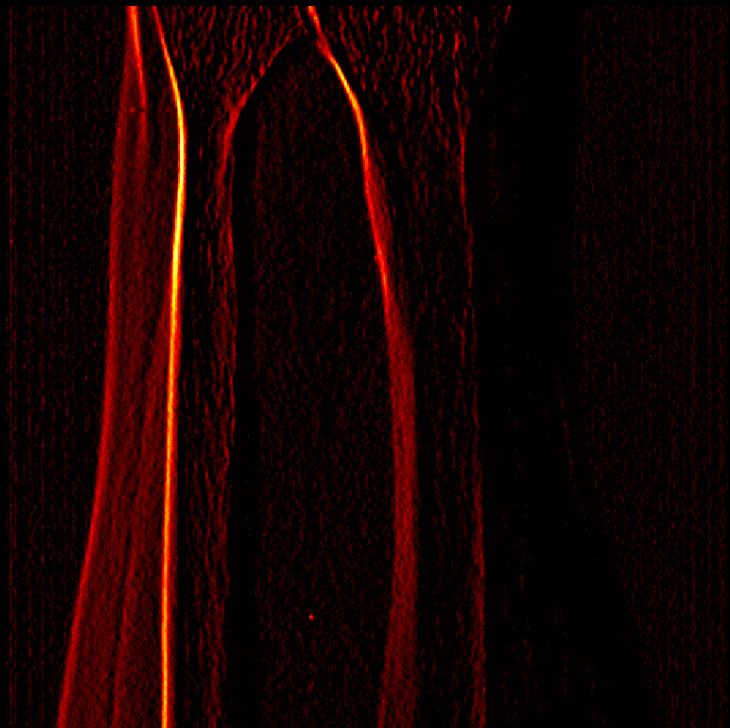


# Pre-processing



Edge filtered image  
(Gaussian smoothing followed by Prewitt)

# Path tracing on pre-processed image



Paths found on pre-processed image and  
intensity inverted



## Quiz 7: Optimal Path 2

- A) 81
- B) 64
- C) 11
- D) 73
- E) 51

A  $5 \times 5$  image is filled with values given the gray level run length encoding: 2, 180, 1, 15, 3, 112, 1, 8, 4, 177, 1, 20, 4, 195, 1, 12, 3, 242, 2, 25, 3, 9. After that the optimal path is found. What is the total cost?

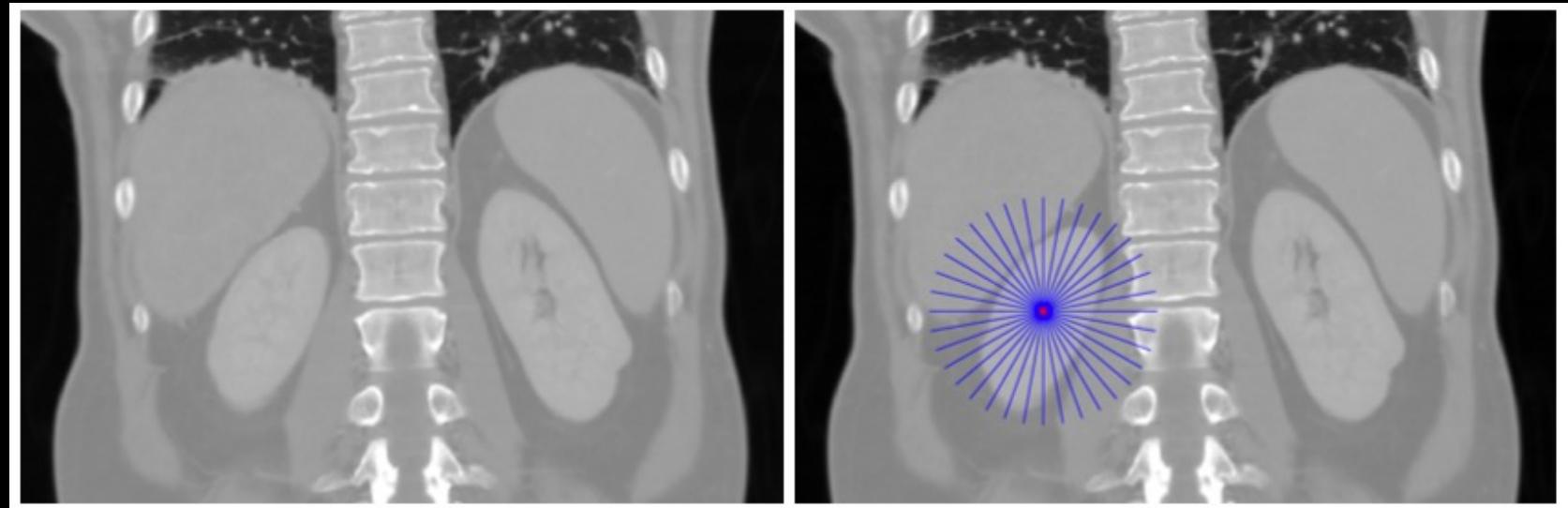
Solution:

$$15+8+20+12+9=64$$

180	180	15	112	112
112	8	177	177	177
177	20	195	195	195
195	12	242	242	242
25	25	9	9	9

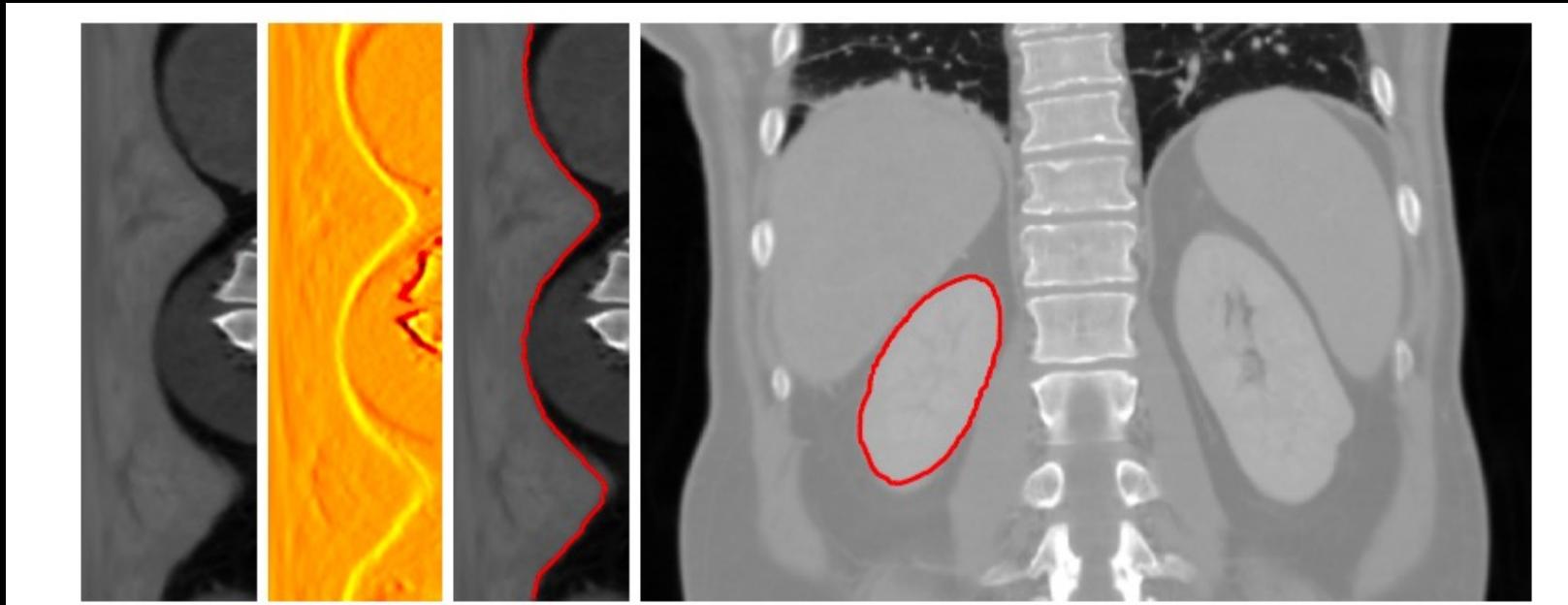


# Locating Circular Structures



- Define origin inside structure
- Send out spokes

# Locating Circular Structures



- Each spoke is a line in a new image (surface- layer detection)
- Prewitt
- Dijkstra's algorithm
- Map back the spokes into image



# What did you learn today?

- Use the Hough transform for line detection
- Describe the slope-intercept, the general form and the normalised form of lines
- Describe the connection between lines and the Hough space
- Use edge detection to enhance images for use with the Hough transform
- Use dynamic programming to trace paths in images
- Describe how an image can be used as a graph
- Describe the fundamental properties of a cost image
- Compute the cost of path
- Compute an accumulator image for path tracing
- Compute a back tracing image for path tracing
- Choose appropriate pre-processing steps for path tracing
- Describe how circular structures can be located using path tracing



# Lecture 9 – Industry presentations

JLIVision  
FOSS Analytics  
Dalux  
Videometer  
IHfood  
TrackMan  
Novo Nordisk  
Radiobotics  
Visiopharm  
Claas E-systems



# Image Analysis

Tim B. Dyrby

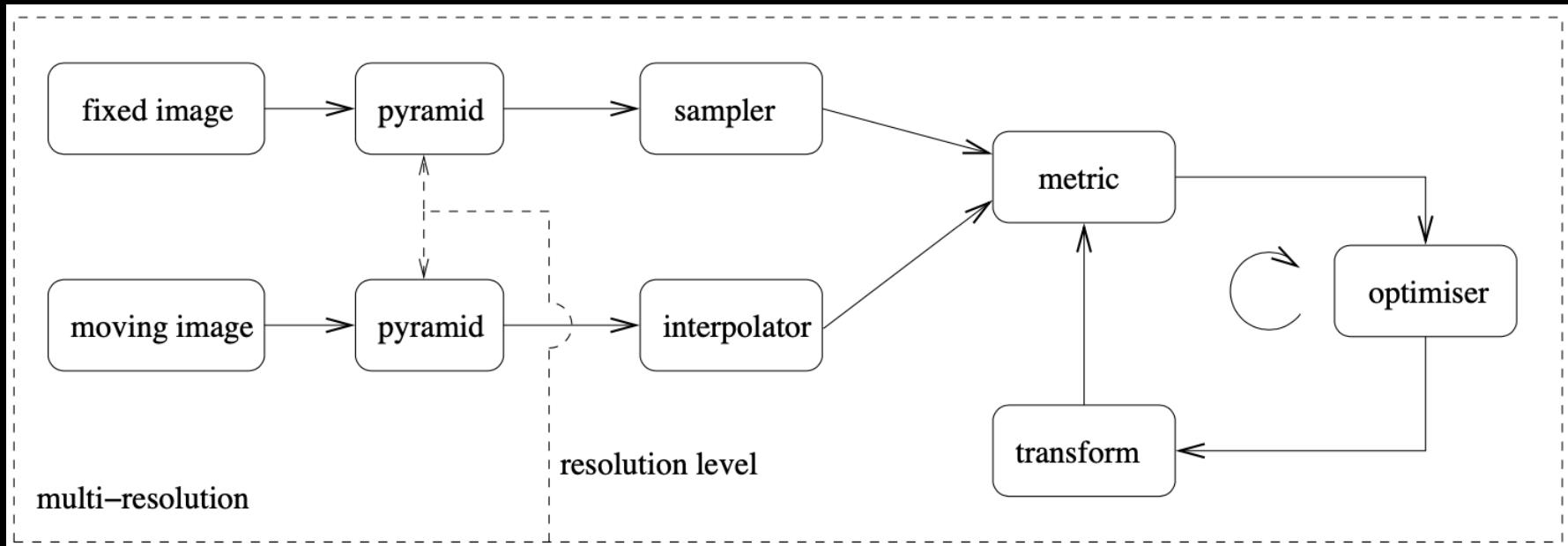
Rasmus R. Paulsen

DTU Compute

[tbdy@dtu.dk](mailto:tbdy@dtu.dk)

<http://www.compute.dtu.dk/courses/02502>

# Lecture 10 – Advanced image registration



Klein et al 2010. (IEEE Trans Med Img)

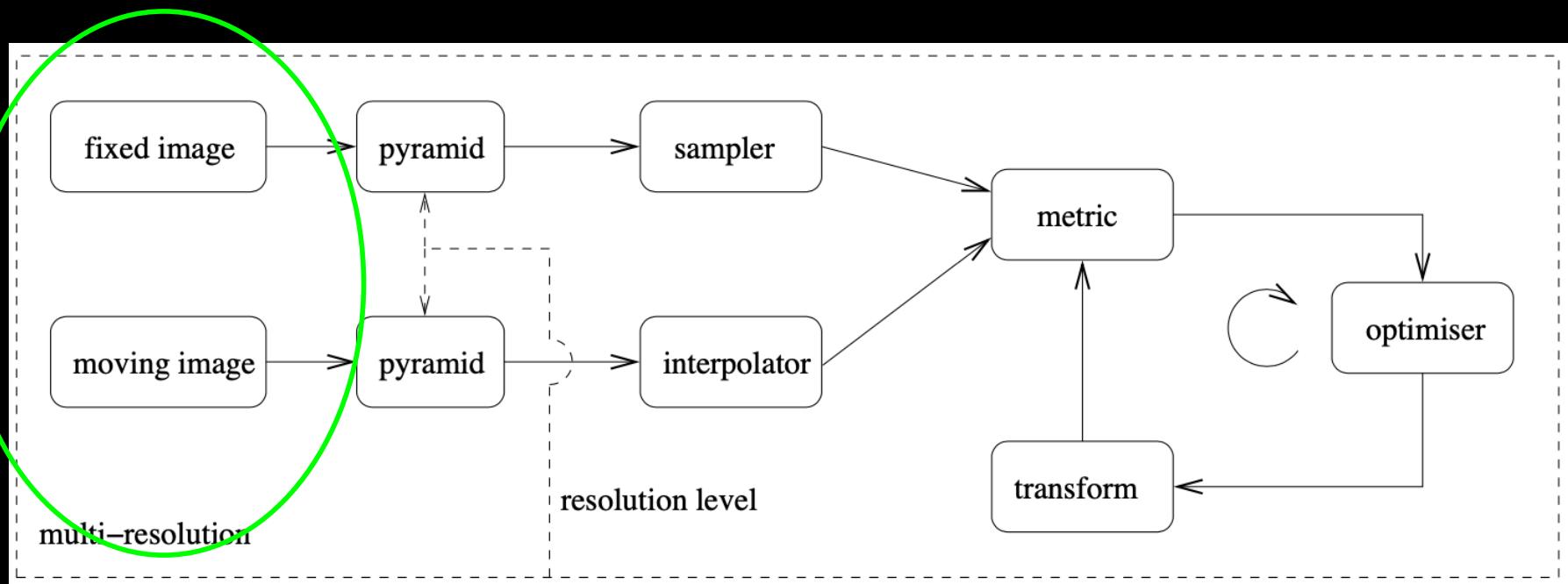
<https://elastix.lumc.nl>

# What can you do after today?

- Describe difference between a pixel and voxel
- Choose a general image-to-image registration pipeline
- Apply 3D geometrical affine transformations
- Use the Homogeneous coordinate system to combine transformations
- Compute a suitable intensity-based similarity metric given the image modalities to register
- Compute the normalized correlation coefficient (NNC) between two images
- Compute Entropy
- Describe the concept of iterative optimizers
- Compute steps in the gradient descent optimization algorithm
- Apply the pyramidal principle for multi-resolution strategies
- Select a relevant registration strategy: 2D to 3D, Within- and between objects and moving images

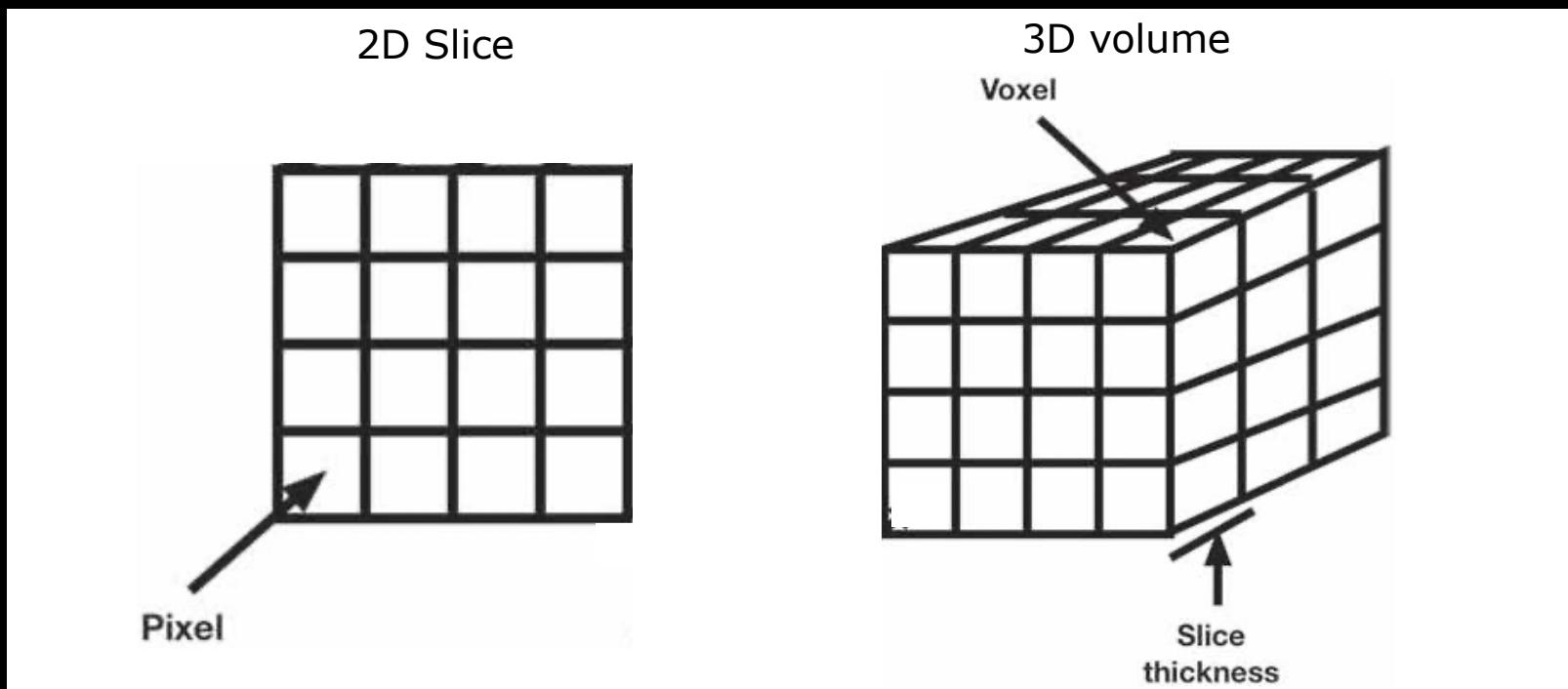
# Image Registration pipeline

- The input images
  - Fixed image: Reference image
  - Moving image: Template image



# Image volumes

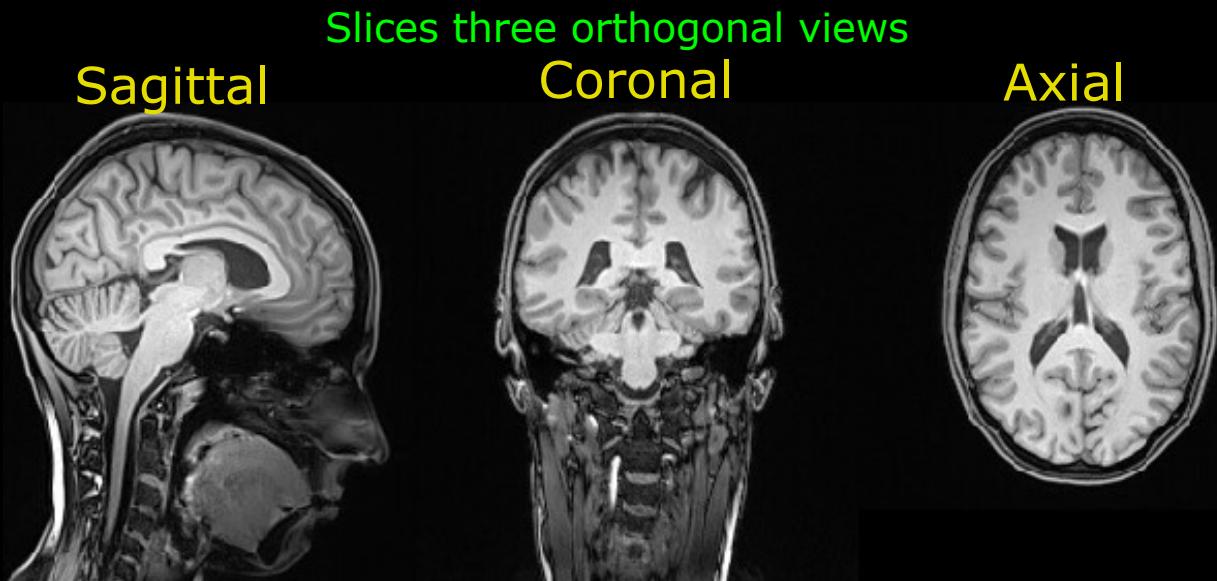
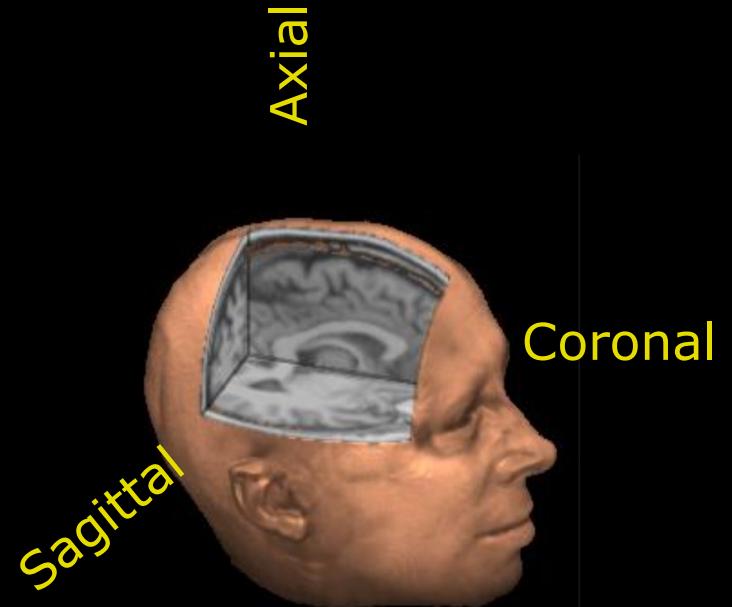
- Image slice: 2D ( $N \times M$ ) matrix of pixels
- Image volumes: 3D ( $N \times M \times P$ ) matrix of voxels
  - An element is a **volume pixel** i.e. voxel
- Pixel vs voxel intensity
  - Integrated information within an area or volume





# 3D image viewing

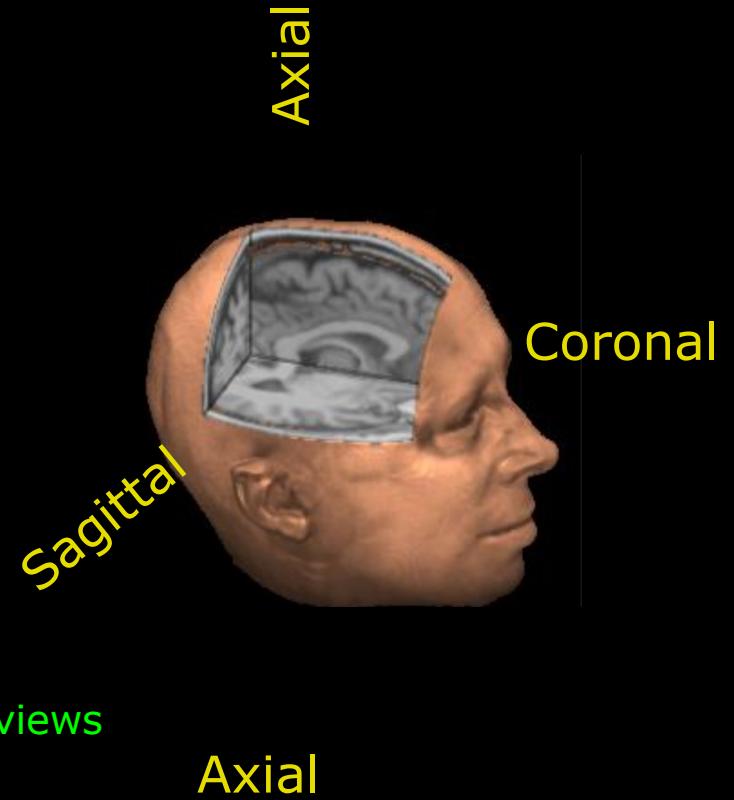
- Three orthogonal views
  - Fine structural details at slice level
  - Hard to get 3D surface insight
- Rendering of surfaces
  - Surface insight
  - Limited types of surfaces to visualise





# 3D image viewing

- Three orthogonal views
  - Fine structural details at slice level
  - Hard to get 3D surface insight
- Rendering of surfaces
  - Surface insight
  - Limited types of surfaces to visualise



Slices three orthogonal views

Sagittal

Coronal

Axial

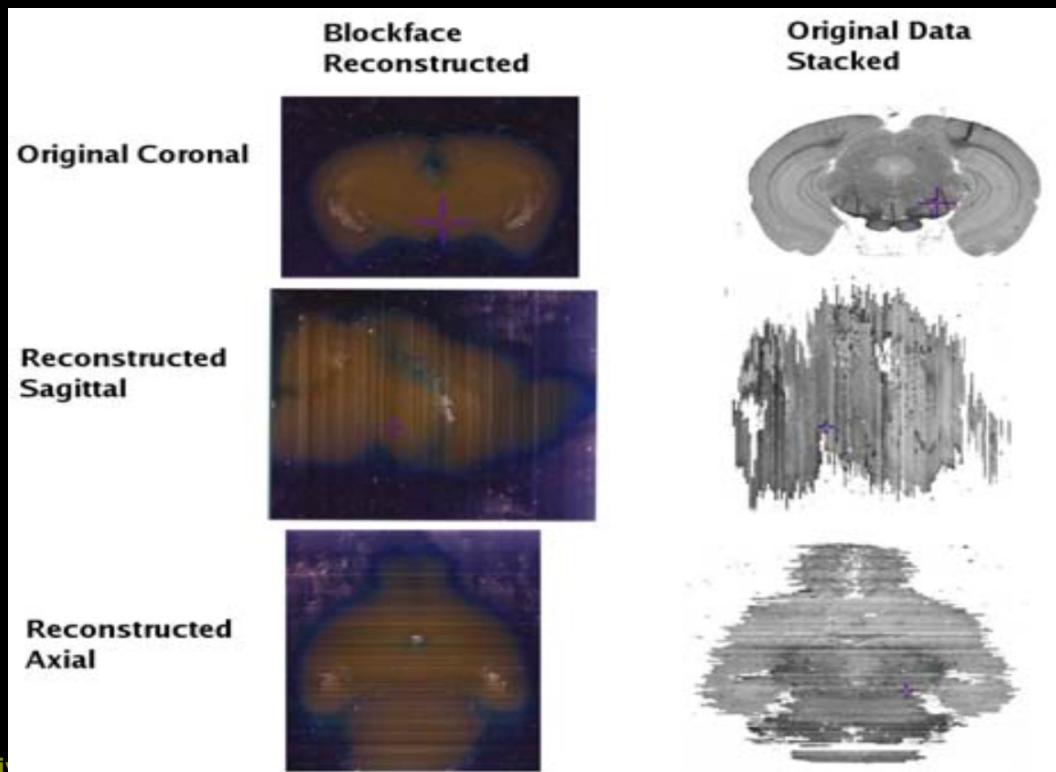


[www.dreamstime.com/illustration/truck-top-view.html](http://www.dreamstime.com/illustration/truck-top-view.html)



# Image volumes

- Stacked slices: 2D to 3D
  - Object cut into slices, imaged and stacked
  - Still pixels – not voxel
- Registration challenges
  - Geometrical distortions between slices

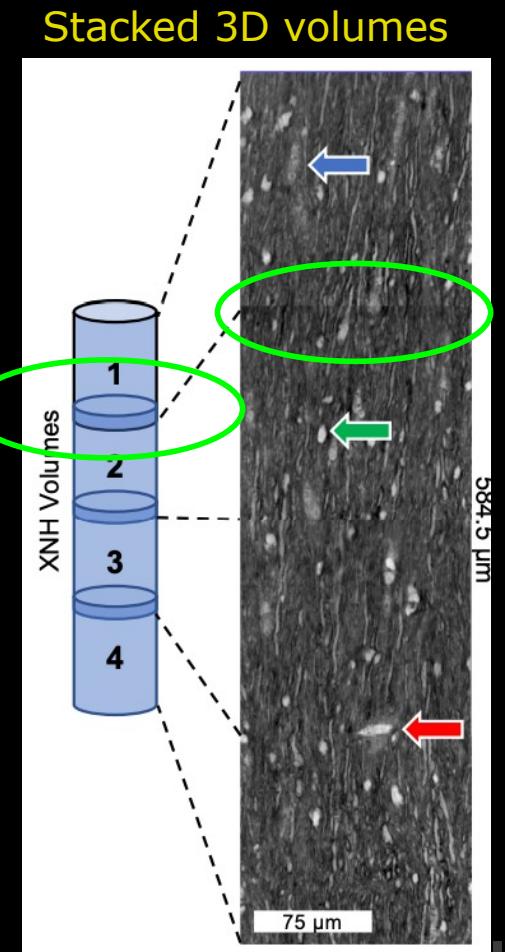
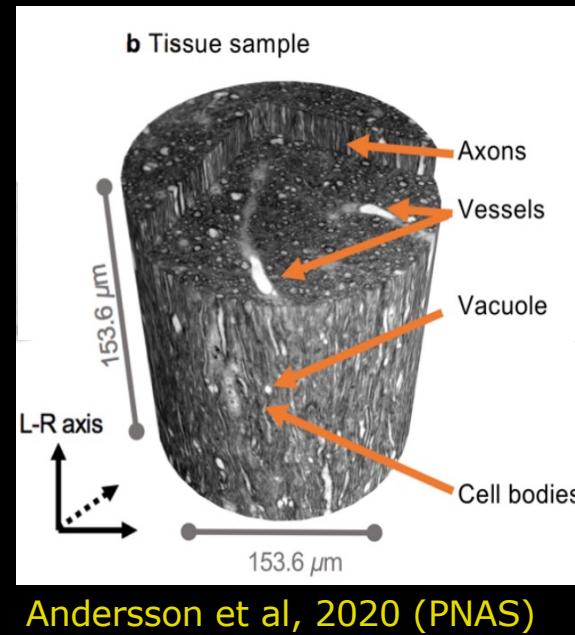
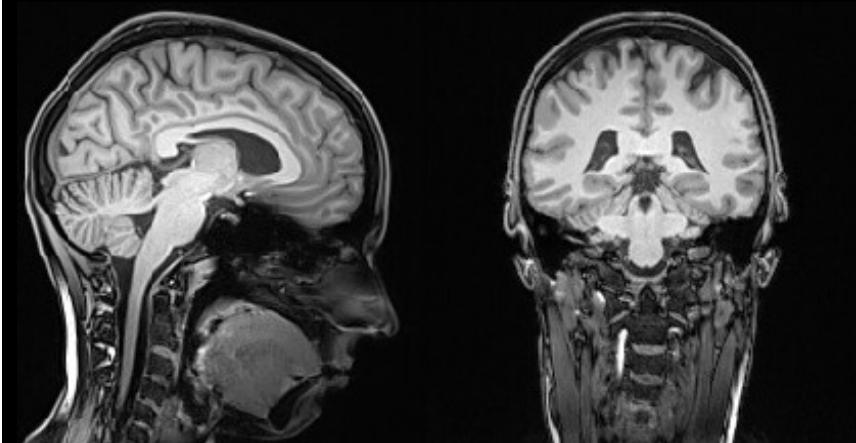


## Synchrotron x-ray imaging Tissue sample 1mm 75 nm isotropic resolution voxels

# Image volumes

- Intact sample
  - No sample cutting
- Registration challenges:
  - Stacking 3D volumes

MRI  
Whole brain  
1 mm isotropic resolution voxels



# Image volumes

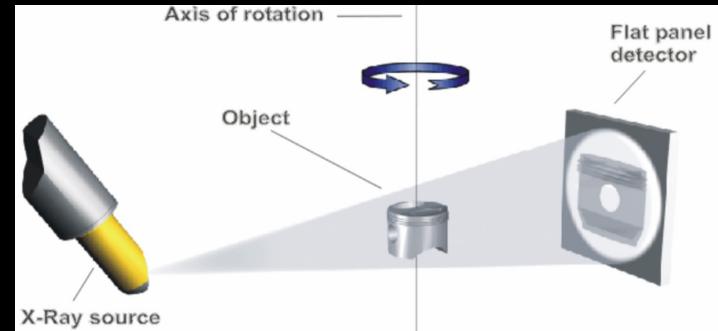
## ■ Intact sample

- No sample cutting

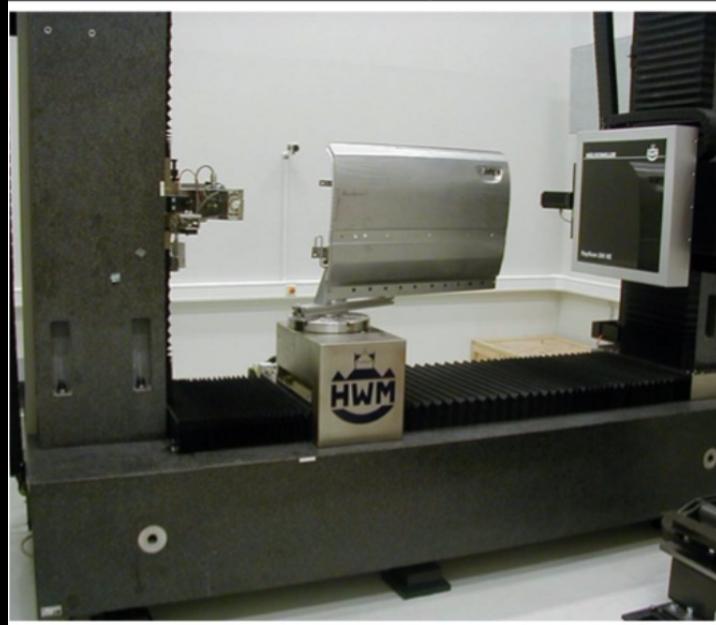
## ■ Registration challenges:

- Multi image resolution: Fit Region-of-interest image to whole object image

Rotating sample in x-ray tomography



CT scanning

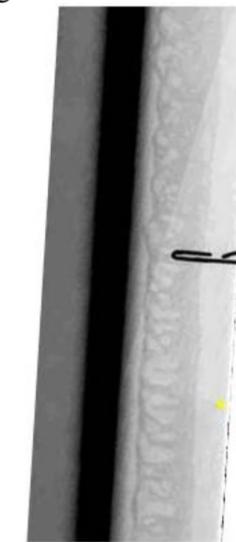


Car door AUDI A8, size: 1150 mm

Region of interest (ROI)



CT of ROI  
(non-destructive)



Microscope  
(destructive)



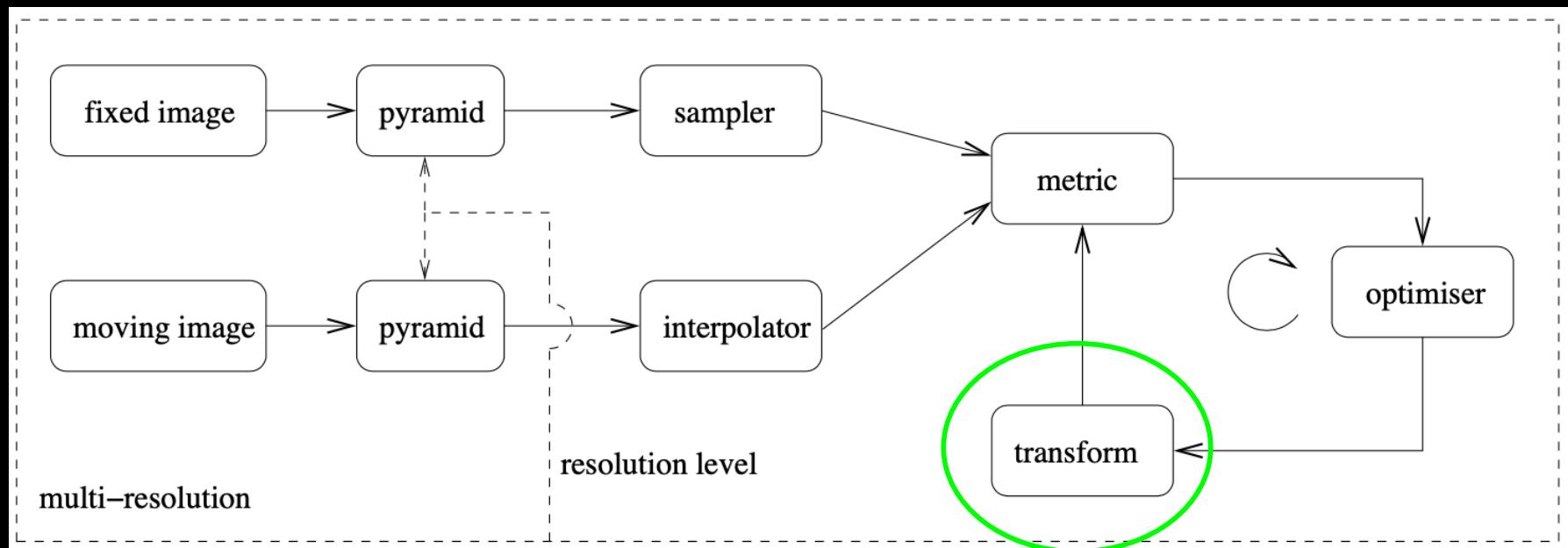
The inspection of a glued joint of a car body

Simon et al, 2006 (ECNDT)

Image Analysis – 02502

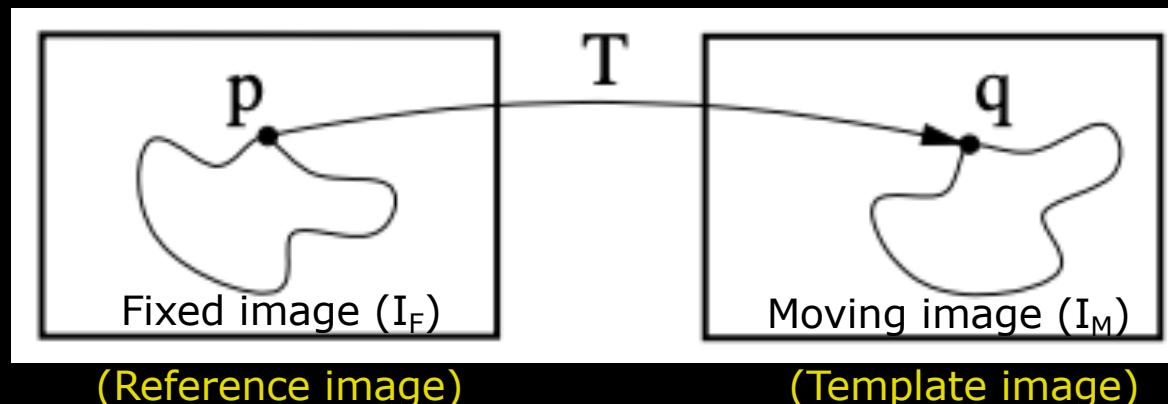
# Image Registration pipeline

## ■ Geometrical transformations



# Geometric transformations

- Translation
- Rotation
- Scaling
- Shearing



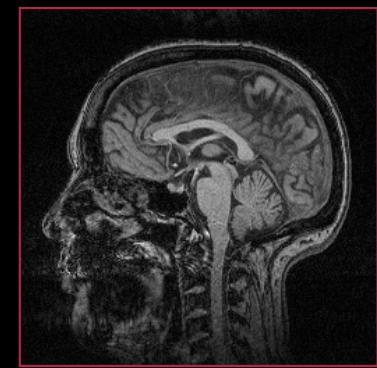
$$\hat{T} = \arg \min_T C(T; I_F, I_M)$$

# Translation 2D vs 3D

- The image is shifted
  - 2D: Inspect one slice plan
  - 3D: Inspect three slice plans

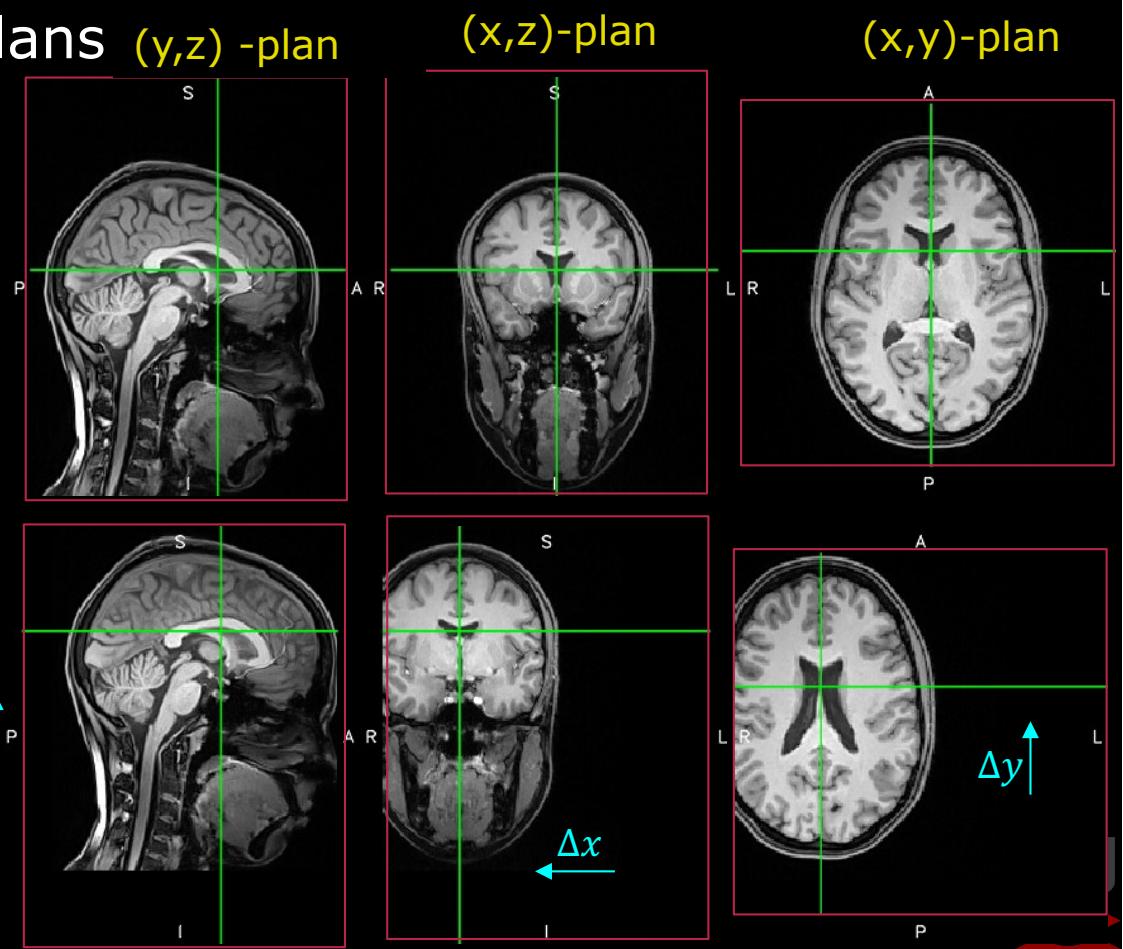
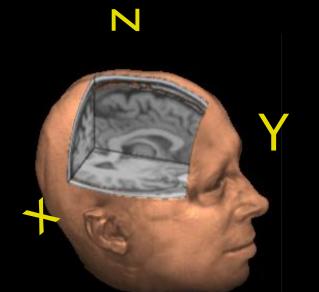
2D: (x,y)-plan

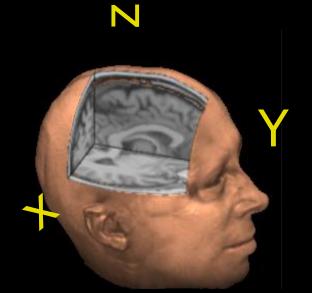
$$\begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} 60 \\ 20 \end{bmatrix}$$



3D: (x,y,z)-plans

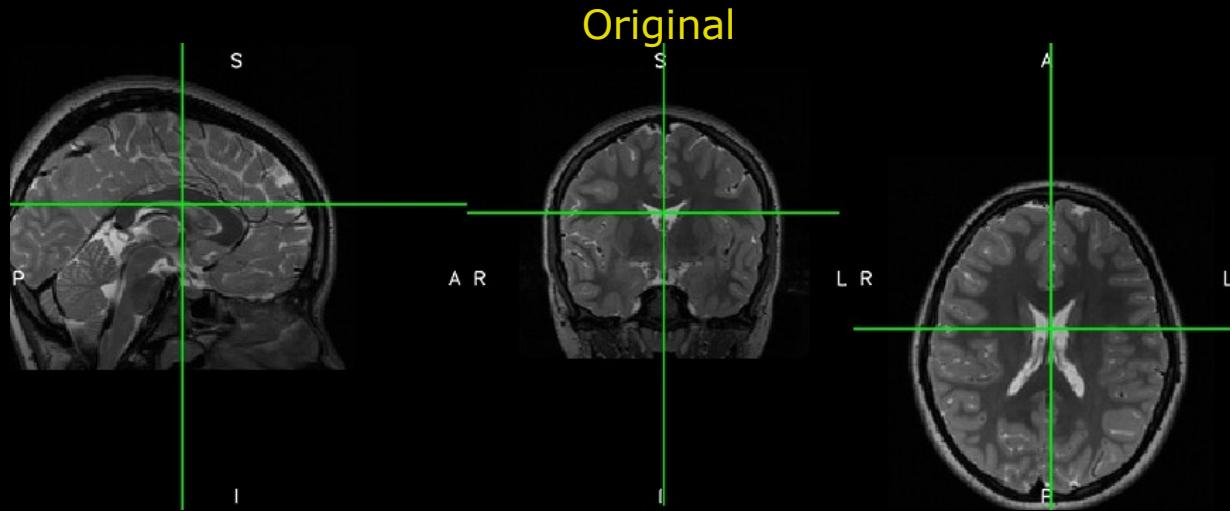
$$\begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix} = - \begin{bmatrix} 60 \\ 20 \\ 15 \end{bmatrix}$$



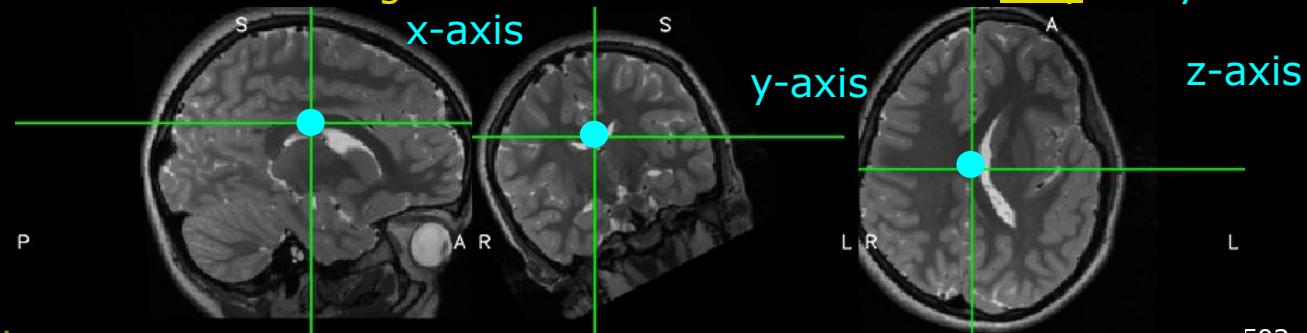


# Rotation 3D

- The image is rotated around an origin (e.g. the centre-of-mass)
- Rotate the object around three axis hence three angles.
  - Inspect all three views to identify a rotation

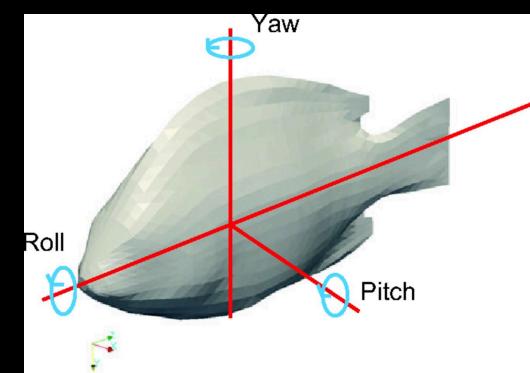
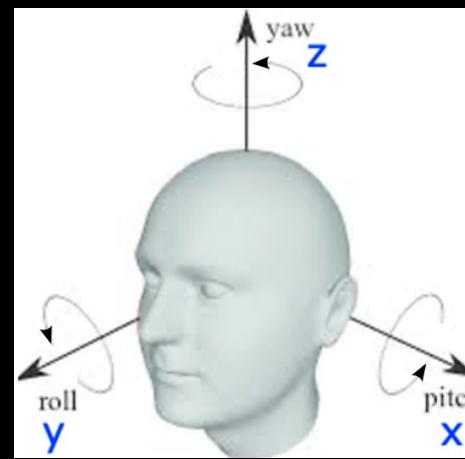
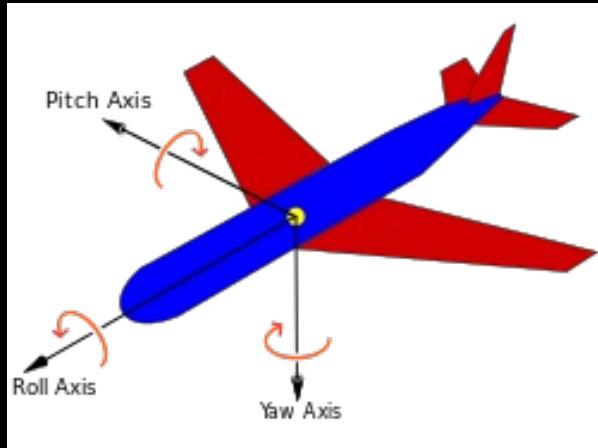


Rotated: 27 degree counter-clockwise around only the y-axis



# 3D Rotation coordinate system

- Three element rotations round the axes of the coordinate system
- Pitch, Yaw and Roll
  - Defined differently for different systems (typ. related to the forward direction)
- Rotation rules
  - Counter clock-wise rotations: Right-hand rule (as in figures) ← We use here
  - Clock-wise rotations: Left-hand rule



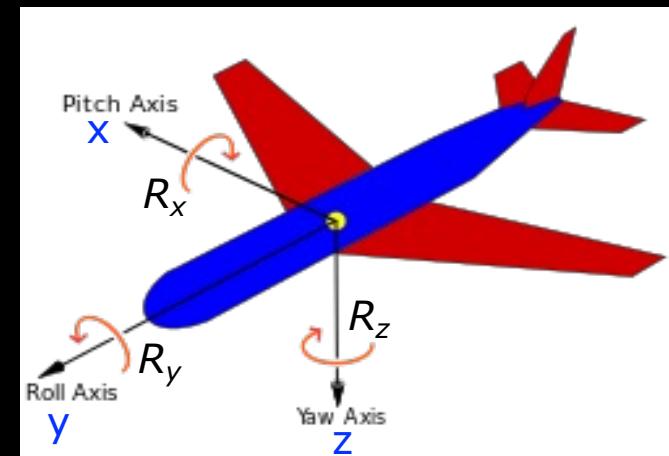
The principal axes of an aircraft according to the air norm DIN 9300



# 3D Rotation coordinate system

- Axis-Angle representation
  - Three composed element rotations
    - Angles:  $\alpha, \beta, \gamma$
    - Counter clock-wise rotations (Right-hand rule)
  - The order matters
    - Several Euler-angle conventions exist
  - Remember: Know your origin!

## Axis-Angle representation

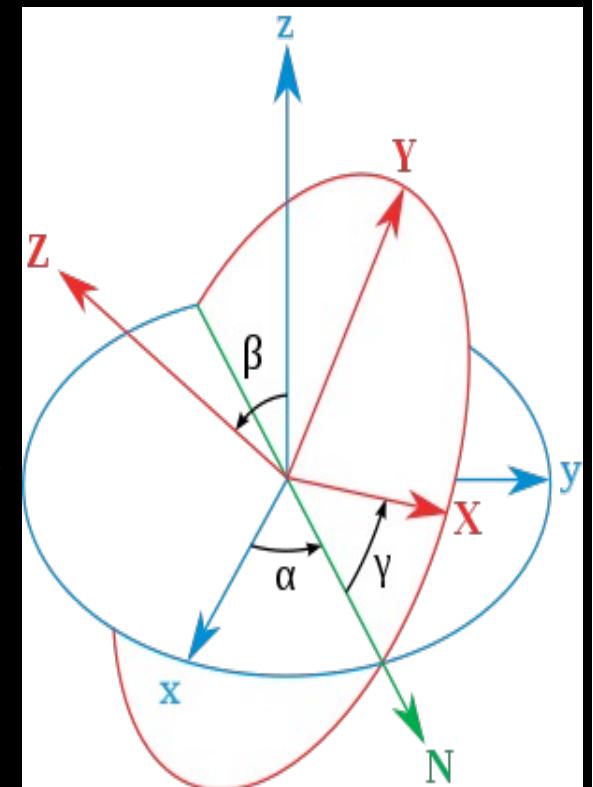
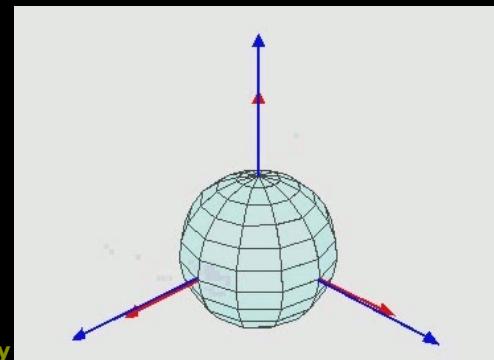


$$R_X = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \alpha & -\sin \alpha \\ 0 & \sin \alpha & \cos \alpha \end{bmatrix} \quad R_Y = \begin{bmatrix} \cos \beta & 0 & \sin \beta \\ 0 & 1 & 0 \\ -\sin \beta & 0 & \cos \beta \end{bmatrix} \quad R_Z = \begin{bmatrix} \cos \gamma & -\sin \gamma & 0 \\ \sin \gamma & \cos \gamma & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

# Euler convention - example

- The intrinsic ZXZ-Euler angle convention (uses the right-hand rule):
  - $\alpha$ : Around the **z-axis**. Defines the **line of nodes (N)**
  - $\beta$ : Around the new **X-axis** defined by **N**
  - $\gamma$ : Around the new **Z-axis** from **N**
- The order of coordinate system rotations:
  - Rotation order around the:
  - **z-axis**: Initial: Original frame ( $x,y,z$ ):  $\alpha$
  - **New X-axis**: First coordinate system rotation ( $X,Y,Z$ ):  $\beta$
  - **New Z-axis**: Second coordinate system rotation ( $X,Y,Z$ ):  $\gamma$

$$A_R = R_Z(\gamma) * R_x(\beta) * R_Z(\alpha)$$

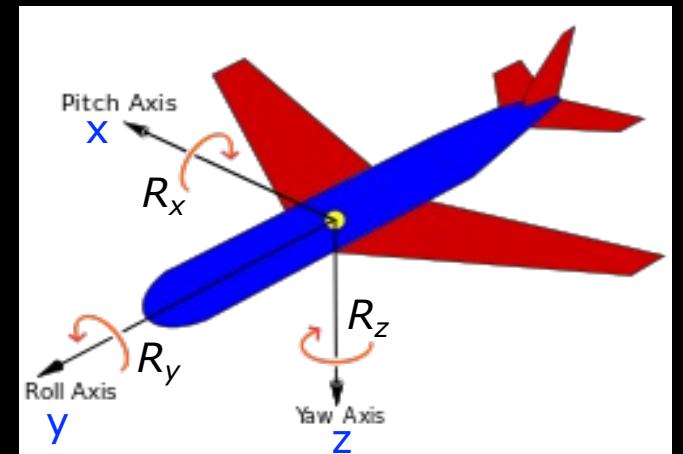


[wikipedia.org/wiki/Euler\\_angles](https://en.wikipedia.org/wiki/Euler_angles)

# Euler convention - example

- The ZYX (Yaw-Pitch-Roll) Euler angle convention (uses the right-hand rule)
- What we use in the course
- Rotation order:
  - *Yaw: rotation around the Z-axis*
  - *Pitch: Rotation around the Y-axis*
  - *Roll: Rotation around the X-axis*

$$A_R = R_X(\gamma) * R_Y(\beta) * R_Z(\alpha)$$





# Quiz 1: Affine 3D transformation

How many parameters?

- A) 6
- B) 5
- C) 16
- D) 12
- E) 3

SOLUTION:

Translation:  $P=3$

Rotation:  $p=3$

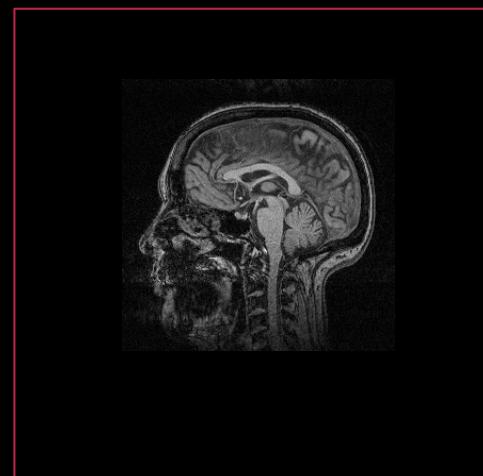
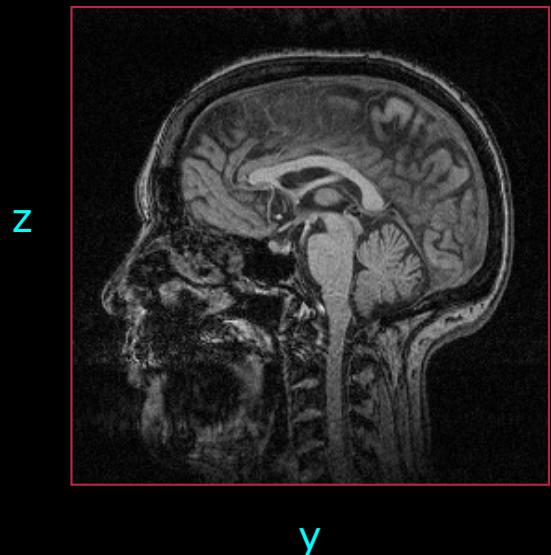
Scaling:  $p=3$

Shearing:  $p=3$

# Scaling in 3D

- The size of the image is changed
- Three parameters:
  - X-scale factor,  $S_x$
  - Y-scale factor,  $S_y$
  - Z-scale factor,  $S_z$
- Isotropic scaling:

$$A = \begin{bmatrix} Sx & 0 & 0 \\ 0 & Sy & 0 \\ 0 & 0 & Sz \end{bmatrix}$$

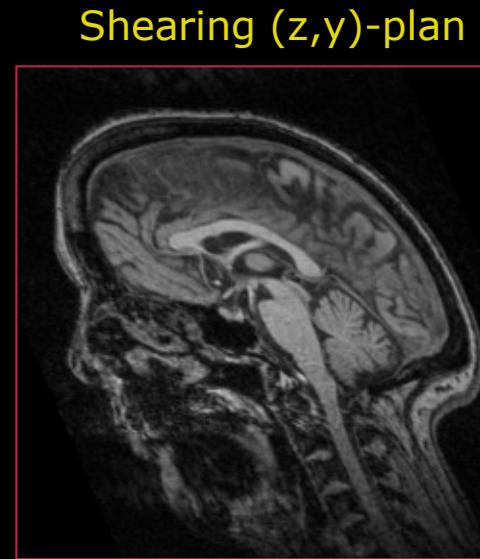
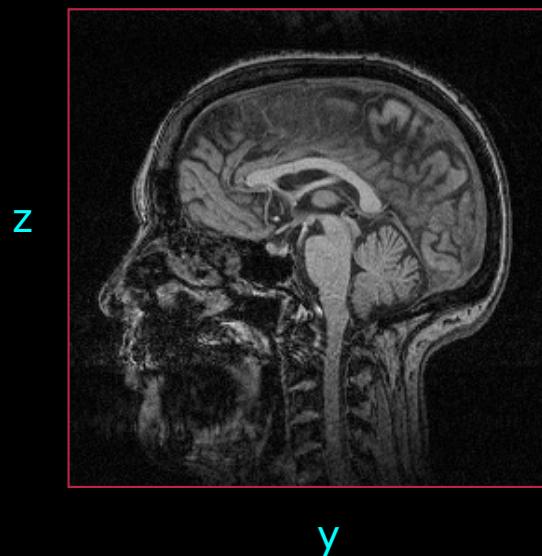


$$A = \begin{bmatrix} 0.5 & 0 & 0 \\ 0 & 0.5 & 0 \\ 0 & 0 & 0.5 \end{bmatrix}$$

# Shearing in 3D

- Pixel shifted horizontally or/and vertically
- Three parameters

$$A = \begin{bmatrix} 1 & S_{yx} & S_{zx} \\ S_{xy} & 1 & S_{yz} \\ S_{xz} & S_{yz} & 1 \end{bmatrix}$$



# Combining transformations

Translation:

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix} + \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

- Translation is a *summation* i.e.  $P' = A + P$
- Rotation, Scale, Shear are *multiplications* i.e.  $P' = A * P$

Rotations,  
Scaling,  
Shear:

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = A \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

- Combine transformations multiplications:

$$A = A_T * AR * A_{shear} * A_s$$

- Not possible with  $A_T$

# Homogeneous coordinates

Cartesian coordinates:

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = A \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

- Projective geometry
  - Used in computer vision
- Adds an extra dimension to vector,  $W$ :

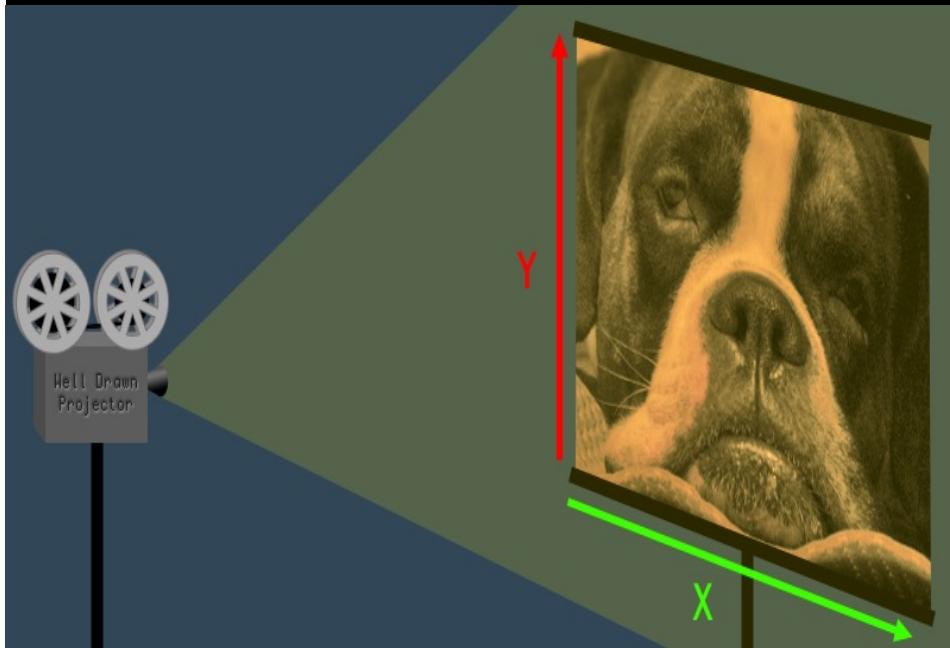
$$[x, y, z, w]$$

Homogeneous coordinates:

$$\begin{bmatrix} x' \\ y' \\ z' \\ w \end{bmatrix} = A \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix}$$

- $W$  scales the  $x$ ,  $y$  and  $z$  dimensions
- $x, y, z$  are “correct” when  $W=1$
- How does it work?

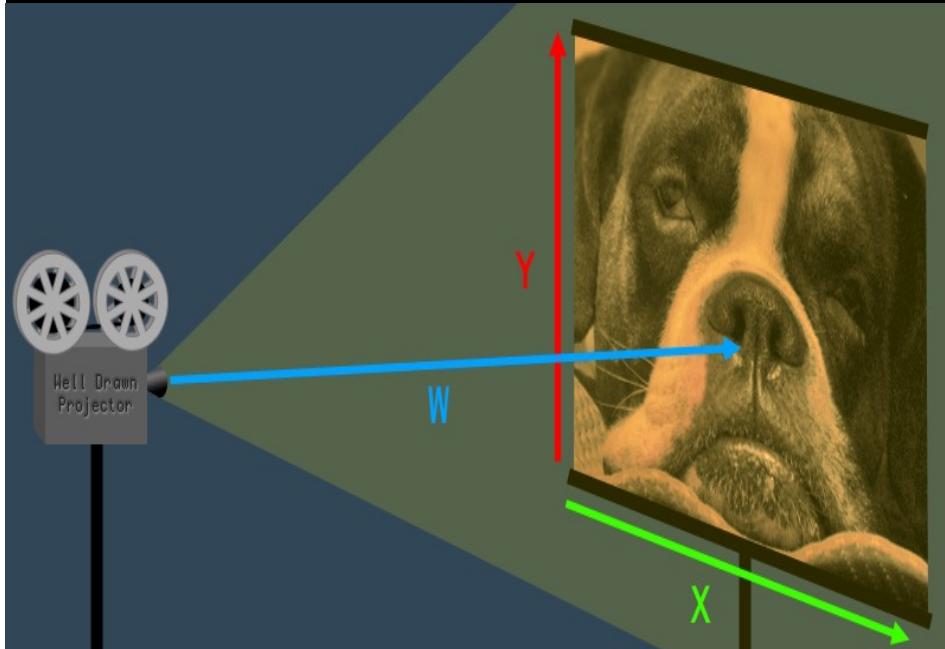
# Homogeneous coordinates



- Euclidean geometry:
  - A point is  $(x,y)$
  - A 2D image
  - Cartesian coordinates

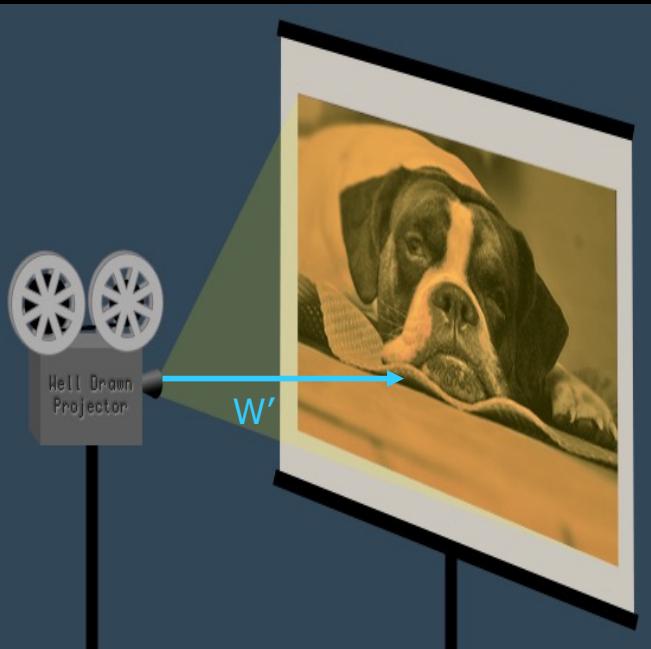
[www.tomdalling.com/blog/modern-opengl/explaining-homogenous-coordinates-and-projective-geometry/](http://www.tomdalling.com/blog/modern-opengl/explaining-homogenous-coordinates-and-projective-geometry/)

# Homogeneous coordinates



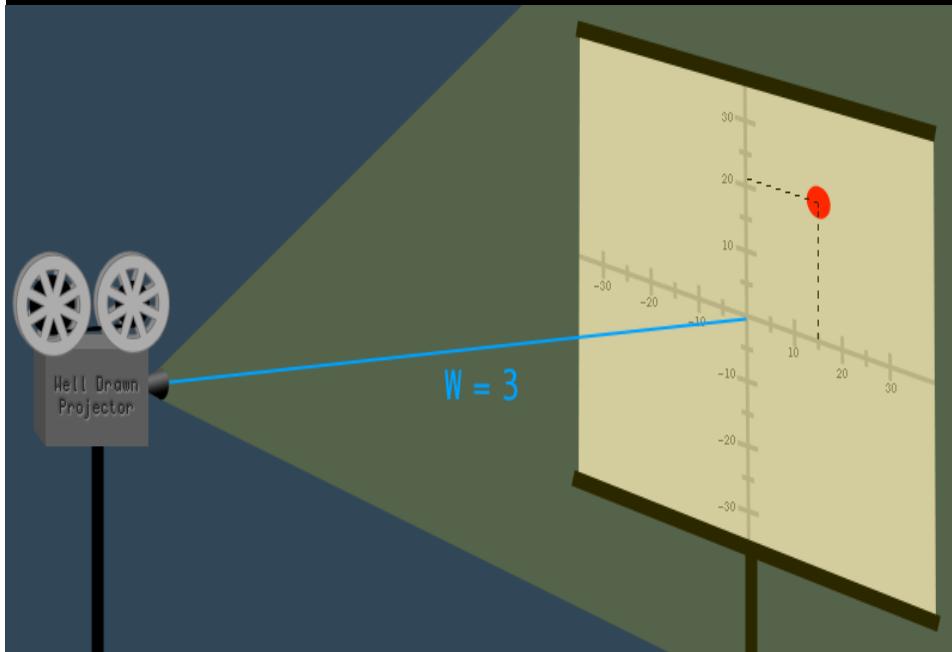
- Euclidean geometry:
  - A point is  $(x, y)$
  - A 2D image
  - Cartesian coordinates
  
- Projective geometry:
  - A point is  $(x, y, W)$
  - "Projective space" adds an extra **projective** dimension,  $W$
  - Changing  $W$  scale factor:
    - No change to the point in projective space
    - Changing perspective/depth

# Homogeneous coordinates



- A point in projective space is  $(x, y, W)$ 
  - Its corresponding Euclidean point is  $(x/W, y/W)$
- Increasing  $W$  (*the same x and y*)
  - The projected point appear closer to the origin
  - The object appear smaller (farther away)
- Scaling to a new depth  $W'$ 
  - Adjusting the point using a scale factor is  $W'/W$  i.e., **new distance/old distance**:  
$$(x*(W'/W), y*(W'/W), W')$$
- When  $W$  or  $W' = 1$ 
  - a projective coordinate  $(x, y, 1)$  corresponds directly to Euclidean point  $(x, y)$

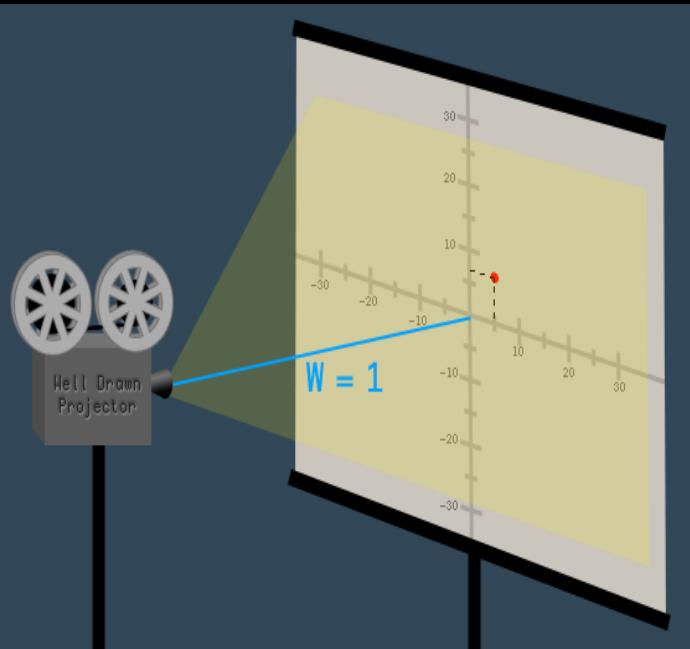
# Homogeneous coordinates



Example:

- Camara:
  - 3 m away from the image,  $W=3$
  - The **dot** on the image is at  $(15, 21)$
- The ***projective coordinate point*** is said to be
  - $(15, 21, 3)$

## Quiz 2: Homogeneous coordinates



SOLUTION:

We move closer to the image i.e.  $W' = 1$  which scales with factor  $(1/3)$  the projective point at  $W=3$  accordingly:

$$(15*(1/3), 21*(1/3), 1) = (5, 7, 1)$$

A camara is placed at distance of 3 meter away from the image and the dot has the projective coordinate of  $(15,21,3)$ .

Now we move the camara closer to the image i.e., 1 m away. What is the new projective coordinate?

- A)  $(5,7,1)$
- B)  $(15,21,3)$
- C)  $(45,63,1)$
- D)  $(5,7,0.33)$
- E)  $(0,0,0)$

# Translation transformation as a matrix

In Euclidian space

Translation:  $\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix}$



In Projective space

$$\begin{bmatrix} x' \\ y' \\ z' \\ W \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \\ W \end{bmatrix} + \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \\ W \end{bmatrix} \quad \text{or} \quad \begin{bmatrix} x' \\ y' \\ z' \\ W \end{bmatrix} = A_T \begin{bmatrix} x \\ y \\ z \\ W \end{bmatrix} \quad \text{where } A_T = \begin{bmatrix} 1 & 0 & 0 & \Delta x \\ 0 & 1 & 0 & \Delta y \\ 0 & 0 & 1 & \Delta z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

## ■ Geometrical transformations

- Use Homogeneous coordinates
- Set  $W=1$  we 'covert' 3D  $\rightarrow$  4D space
- Translation transformation expressed as a **matrix  $A_T$**

# Transformations in Projective space

Translation:  $A_T = \begin{bmatrix} 1 & 0 & 0 & \Delta x \\ 0 & 1 & 0 & \Delta y \\ 0 & 0 & 1 & \Delta z \\ 0 & 0 & 0 & 1 \end{bmatrix}$

Rotations (right-hand rule):  
 - x=pitch  
 - y=roll  
 - z=yaw

$$R_x = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\alpha) & -\sin(\alpha) & 0 \\ 0 & \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad R_y = \begin{bmatrix} \cos(\beta) & 0 & \sin(\beta) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\beta) & 0 & \cos(\beta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad R_z = \begin{bmatrix} \cos(\gamma) & -\sin(\gamma) & 0 & 0 \\ \sin(\gamma) & \cos(\gamma) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

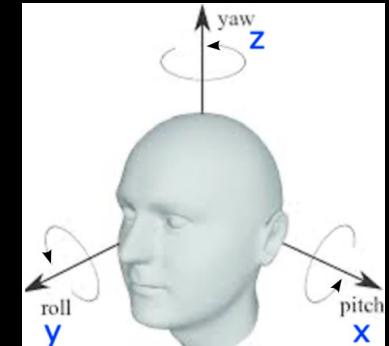
Scaling:

$$A_s = \begin{bmatrix} Sx & 0 & 0 & 0 \\ 0 & Sy & 0 & 0 \\ 0 & 0 & Sz & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Shear:

$$A_z = \begin{bmatrix} 1 & Sxy & Sxz & 0 \\ Sxy & 1 & Syz & 0 \\ Sxz & Syz & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- Axis-Angle representation



Affine transformation:  $A = \underbrace{A_T * (R_x * R_y * R_z)}_{\text{Rigid}} * A_z * A_s$

# Combining transformations – step by step

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix}$$



$$\begin{bmatrix} x' \\ y' \\ z' \\ W \end{bmatrix} = A_T \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \\ W \end{bmatrix}$$

**Remember:**

- Typically calculated in *radians*
- *Same procedure for 2D and 3D images*

- Step 1: Convert 3D to 4D projective space, set  $W=1$ . Make translation into a matrix

$$A = A_T * (R_x * R_y * R_z) * A_z * A_s$$

- Step 2: Multiply all 4D matrices

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = A \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

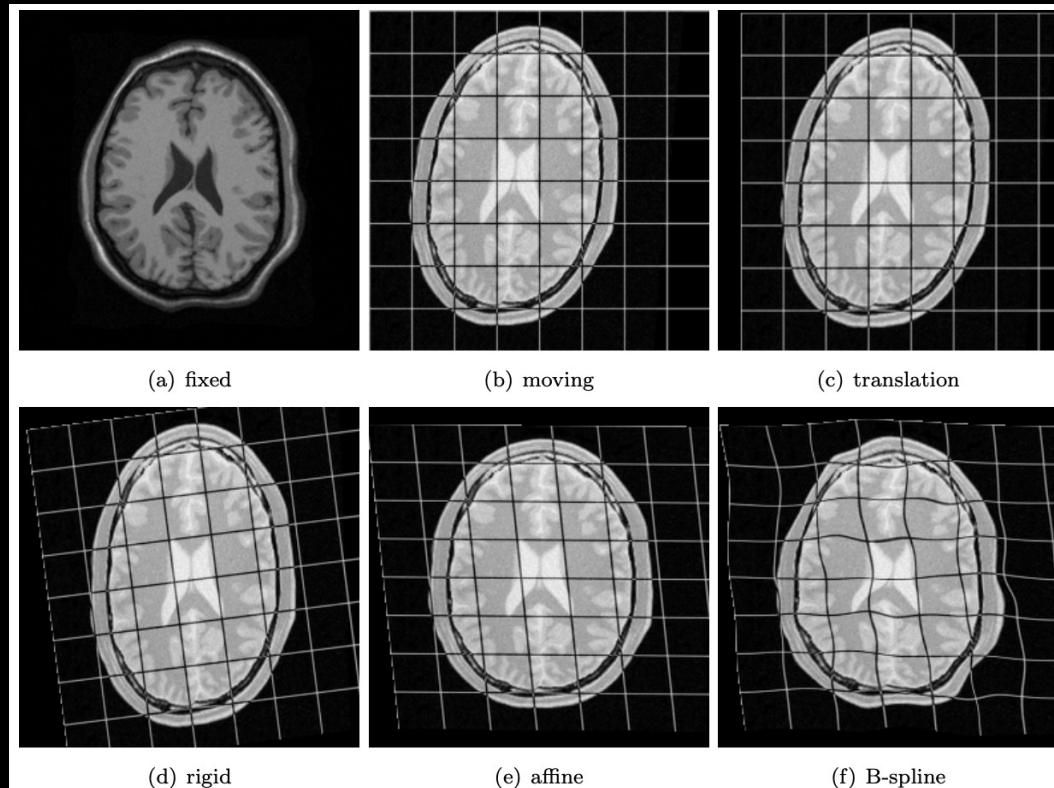
- Step 3: Apply the transformation to a point

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = A \cdot \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

- Step 4: Convert back to 3D Cartesian coordinates by ignoring the  $W$  dimension

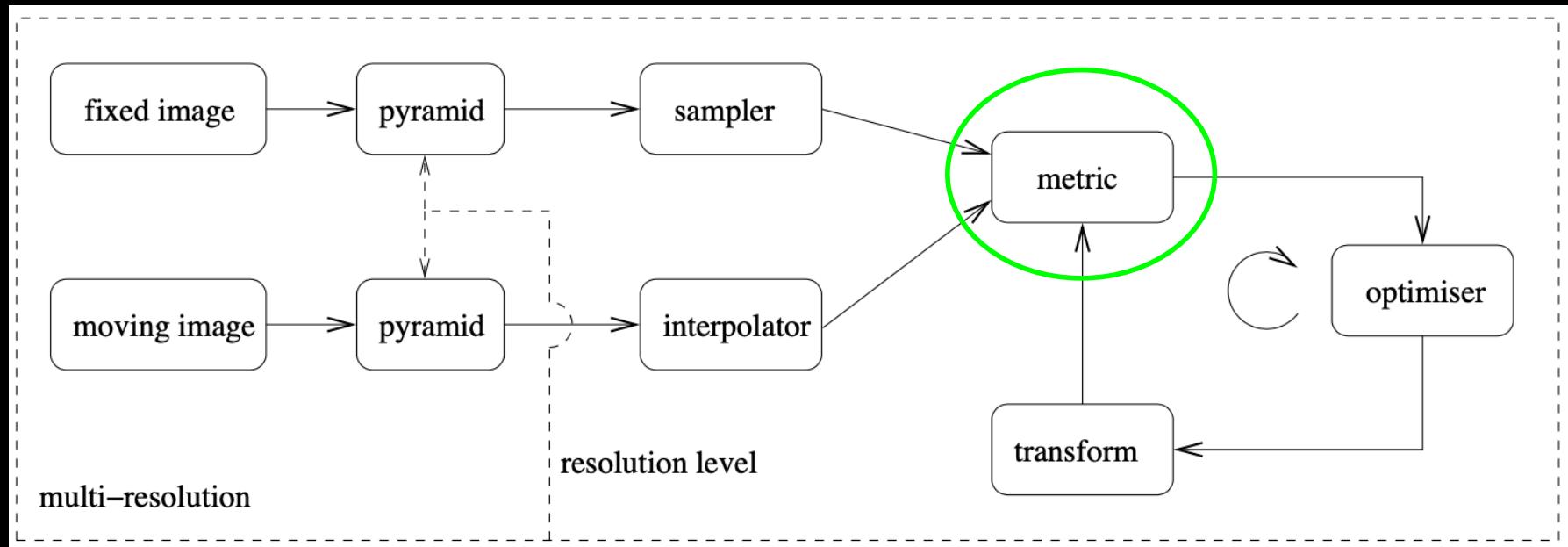
# Different transformations

- Linear: Affine transformation
- Non-linear: Piece-wise affine or B-spline
  - Remember: First to apply the linear transformations!



# Image Registration pipeline

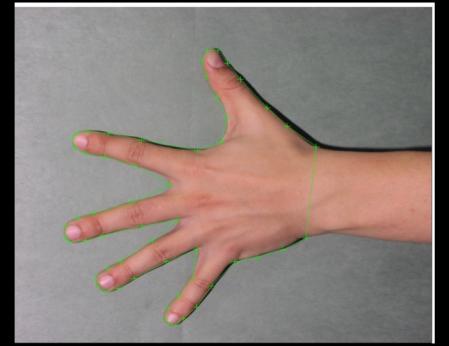
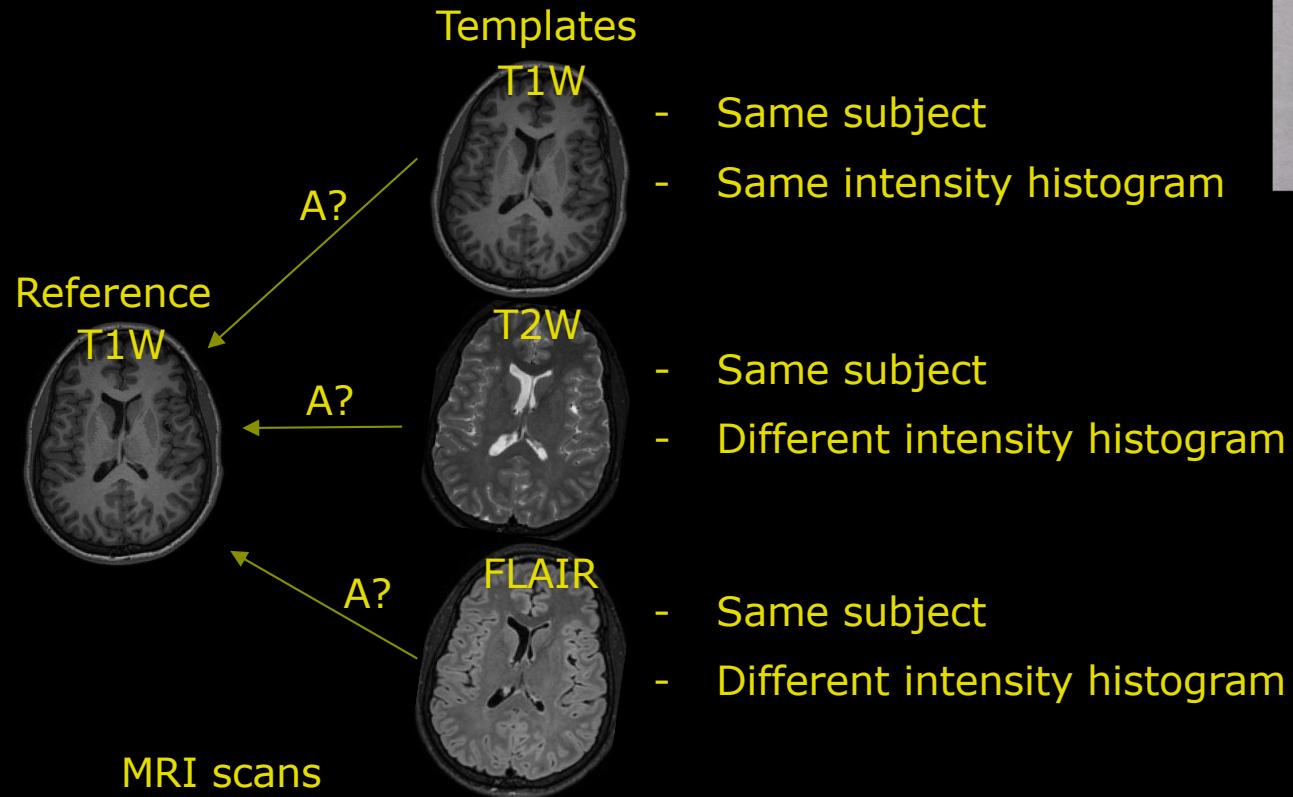
## ■ Similarity measures



# Similarity measures

## ■ Anatomical Landmarks

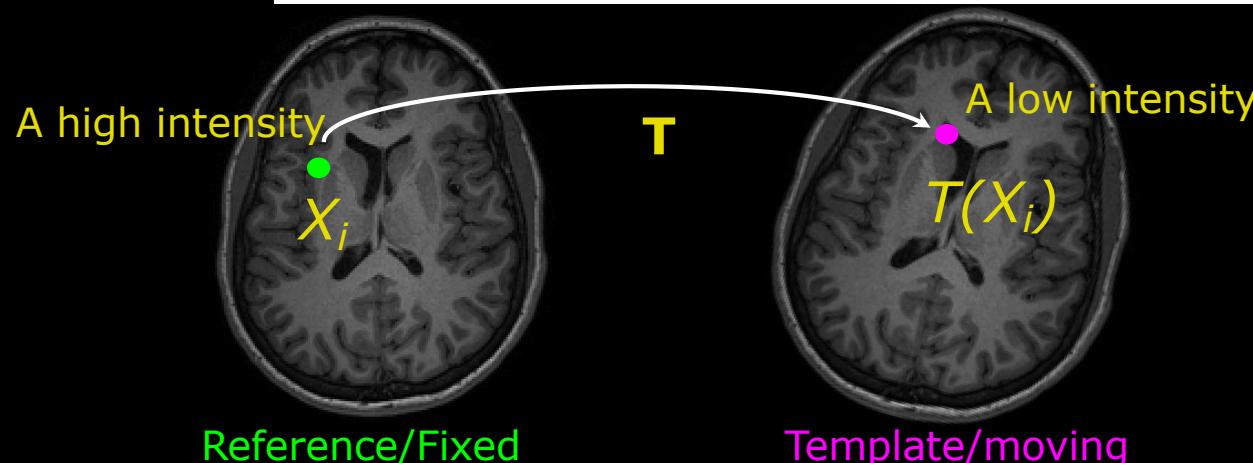
- time consuming to obtain positions manually
- Alternative: **Joint intensity histogram**



# Similarity measure: Mean squared difference (MSD)

- Compare difference in intensities.
  - Same similarity measure we used for anatomical landmarks (positions) in a previous lecture
  - Fast to estimate
- Many local minima's (sub optimal solutions)
  - Intensities are not optimal for this similarity metric

$$\text{MSD}(\boldsymbol{\mu}; I_F, I_M) = \frac{1}{|\Omega_F|} \sum_{\mathbf{x}_i \in \Omega_F} (I_F(\mathbf{x}_i) - I_M(\mathbf{T}_{\boldsymbol{\mu}}(\mathbf{x}_i)))^2,$$



Is  $T$  optimal?

NO!

- Big intensity difference
- Large MSD error

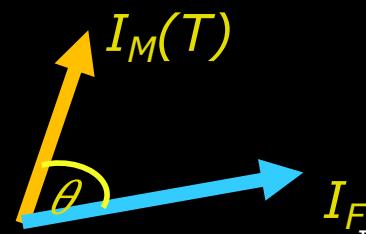
# Similarity measure: Normalised Cross-correlation

- Normalised Cross-correlation of intensities in two images
  - Fast to estimate
- Risk of local minima's (sub optimal solutions)
  - Less robust if image modalities have different intensity histograms
  - Normalise: Reduce the impact of outlier regions

$$\text{NCC}(\boldsymbol{\mu}; I_F, I_M) = \frac{\sum_{\mathbf{x}_i \in \Omega_F} (I_F(\mathbf{x}_i) - \overline{I_F}) (I_M(\mathbf{T}_{\boldsymbol{\mu}}(\mathbf{x}_i)) - \overline{I_M})}{\sqrt{\sum_{\mathbf{x}_i \in \Omega_F} (I_F(\mathbf{x}_i) - \overline{I_F})^2 \sum_{\mathbf{x}_i \in \Omega_F} (I_M(\mathbf{T}_{\boldsymbol{\mu}}(\mathbf{x}_i)) - \overline{I_M})^2}},$$

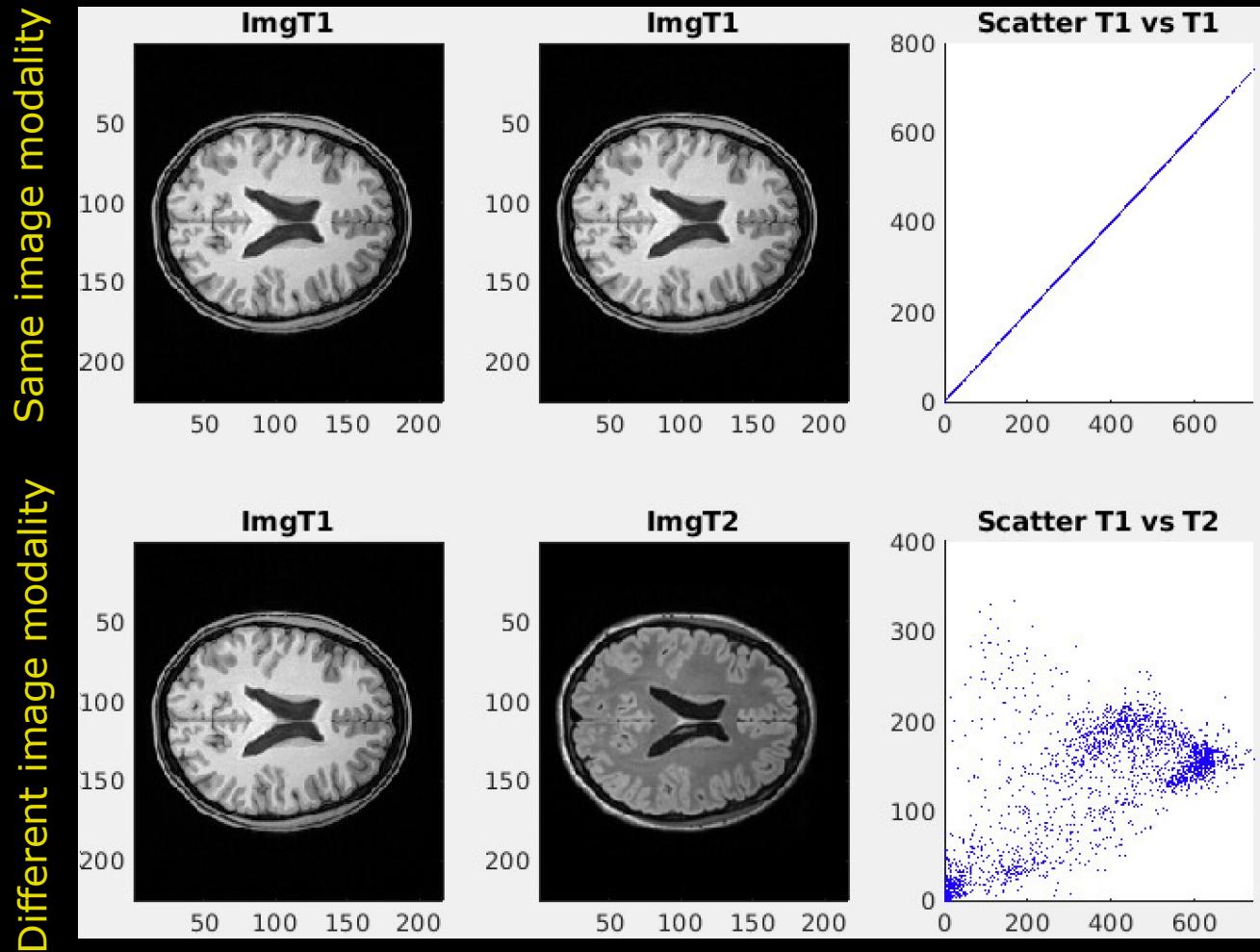
with the average grey-values  $\overline{I_F} = \frac{1}{|\Omega_F|} \sum_{\mathbf{x}_i \in \Omega_F} I_F(\mathbf{x}_i)$  and  $\overline{I_M} = \frac{1}{|\Omega_F|} \sum_{\mathbf{x}_i \in \Omega_F} I_M(\mathbf{T}_{\boldsymbol{\mu}}(\mathbf{x}_i))$ .

- Multiplication is a dot product
  - $I_F \cdot I_M(T) = \|I_F\| \|I_M(T)\| \cos \theta$



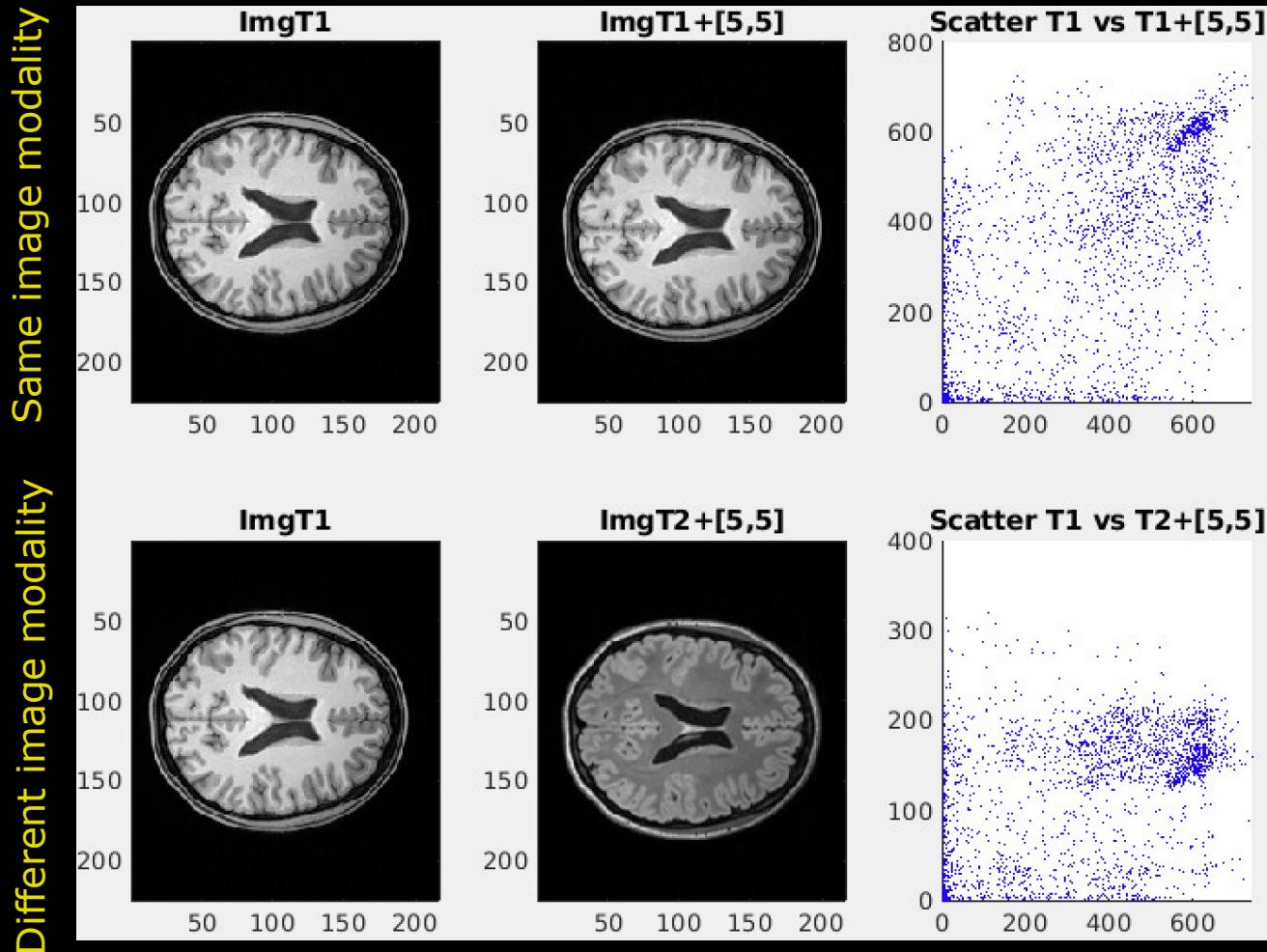
# Joint intensity histograms

- Perfect registered: Optimal joint intensity agreement



# Joint intensity histograms

- Small translation difference: Lower joint intensity agreement



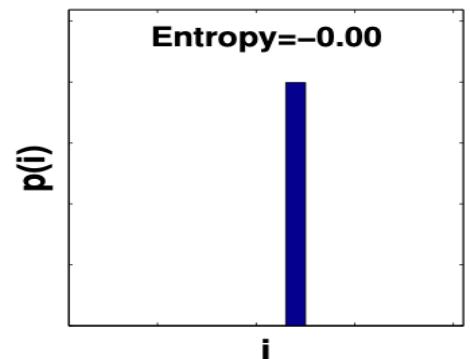
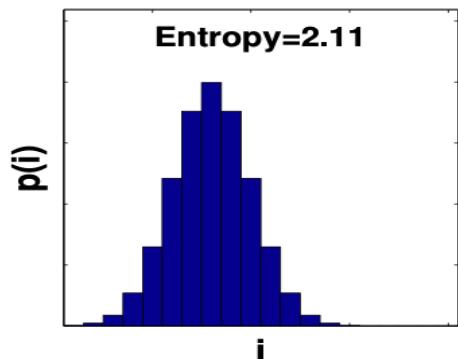
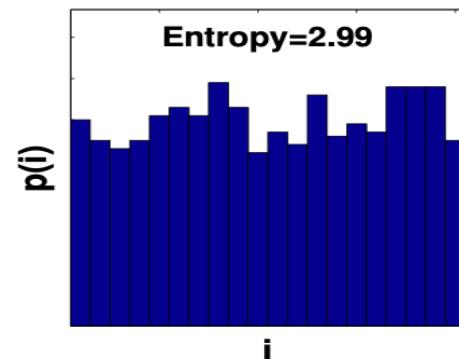
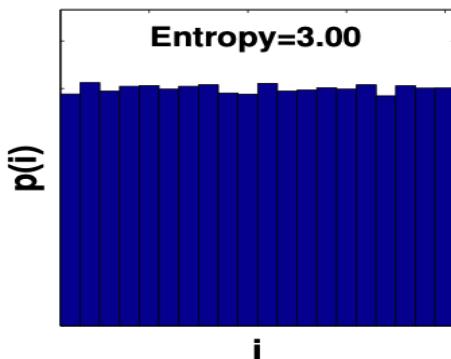
# Similarity measure - Entropy

- Comes from information theory.
  - The higher the entropy the more the information content.
- Entropy (Shannon-Weiner):

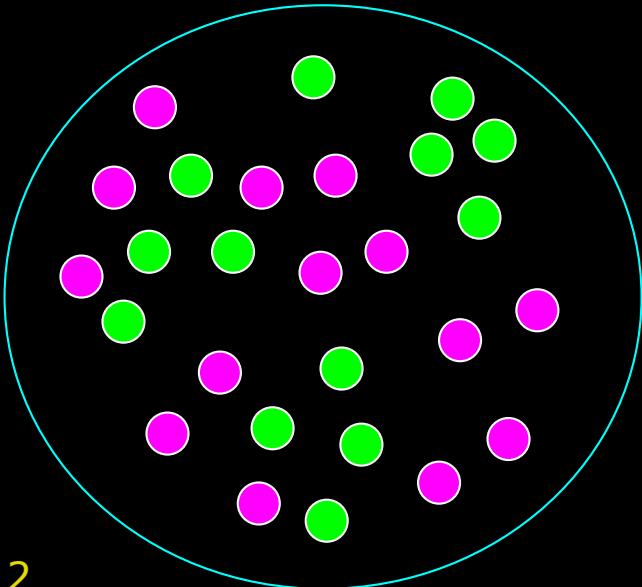
$$H = -\sum_i p_i \log_b p_i$$

Where  $b$ : the base of the logarithm

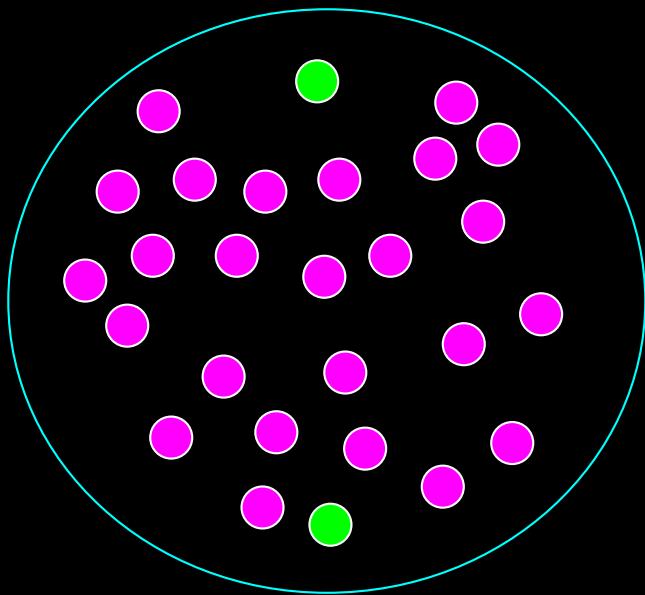
- Bits:  $b=2$  and bans:  $b=10$
- Entropy is typically in bits i.e. typical used in digital information



Candy mix 1



Candy mix 2



- A) Mix 1
- B) Make a new choice
- C) Contain no liquorice
- D) Mix 2
- E) It is not healthy

# Quiz 4: What is the entropy of the candy mix 1?

- A) 0.38
- B) 0.99**
- C) 0.45
- D) 0.23
- E) 0.00

SOLUTION:

Green=13

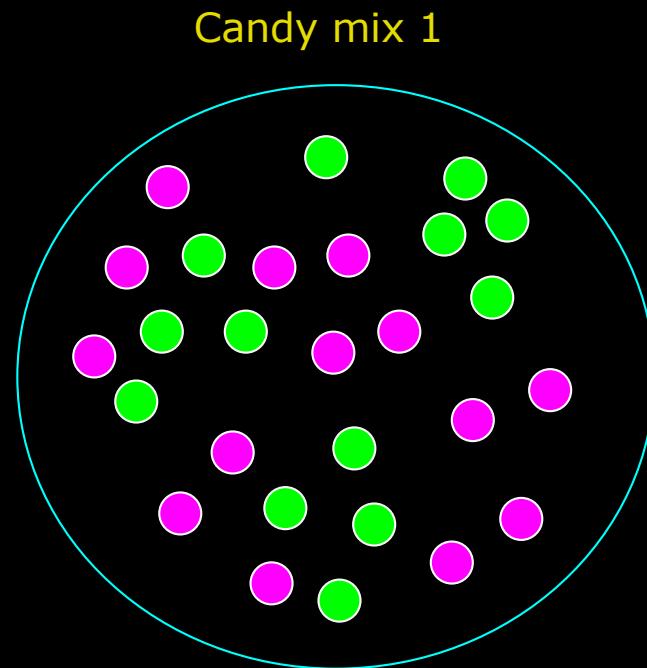
Pink=14

Total=27

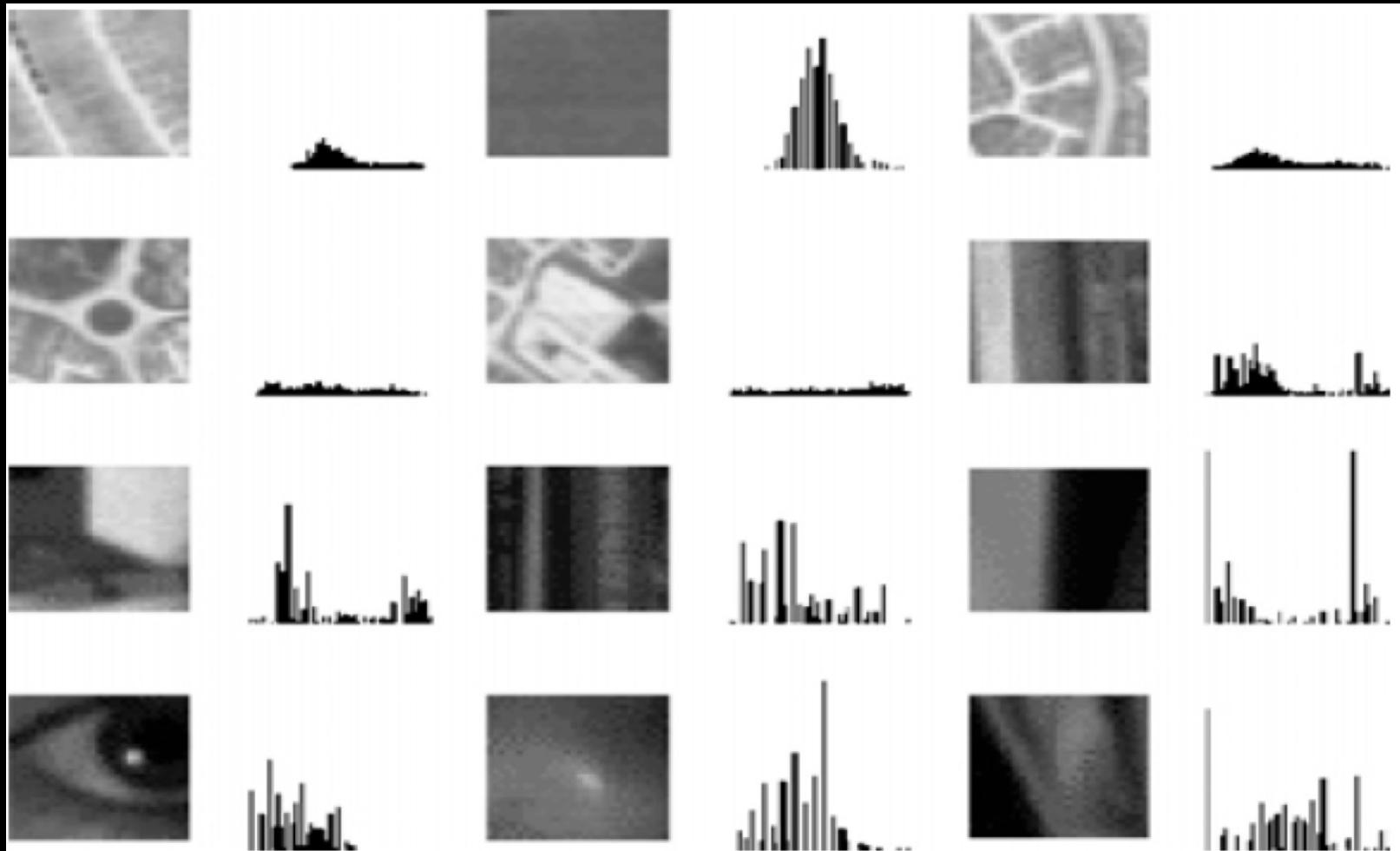
$$pG = 13/27$$

$$pP = 14/27$$

$$\text{Entropy} = -pG \log_2(pG) - pP \log_2(pP) = 0.99$$



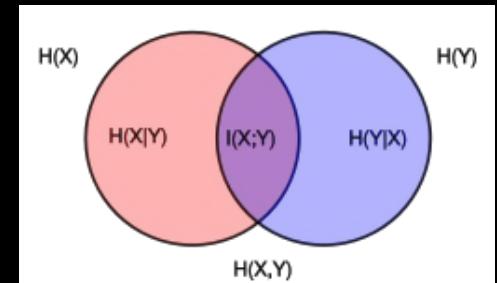
# Histograms of images



# Joint entropy - Mutual information

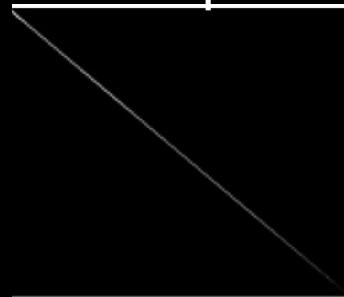
- Joint entropy  $H(X, Y) = - \sum_{X,Y} p_{X,Y} \log p_{X,Y}$
- Similarity measure: The more similar the distributions, the lower the joint entropy compared to the sum of the individual entropies i.e., total area is less spread out

$$H(X, Y) \leq H(X) + H(Y)$$



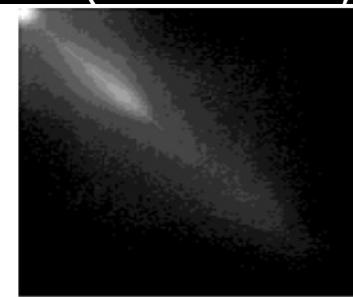
[en.wikipedia.org/wiki/Mutual\\_information](https://en.wikipedia.org/wiki/Mutual_information)

- Example of rotation (Pluim et al., 2003, TMI)



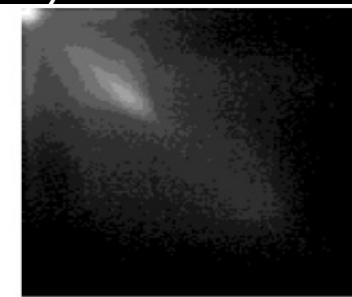
3.82

0 degrees



6.79

2 degrees



6.98

5 degrees



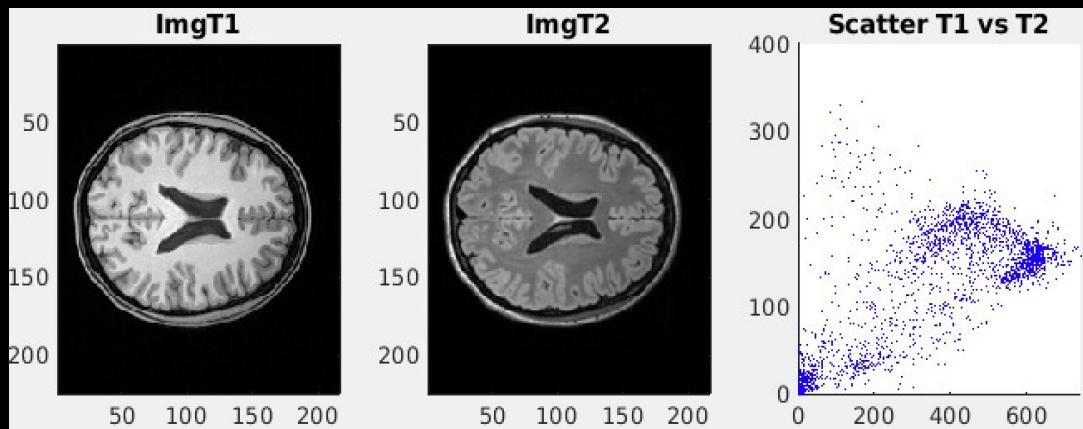
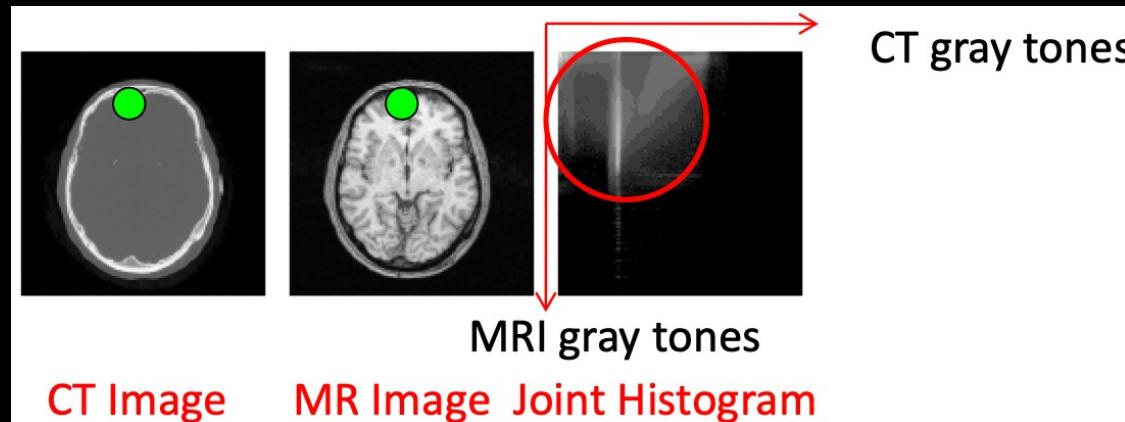
7.15

10 degrees



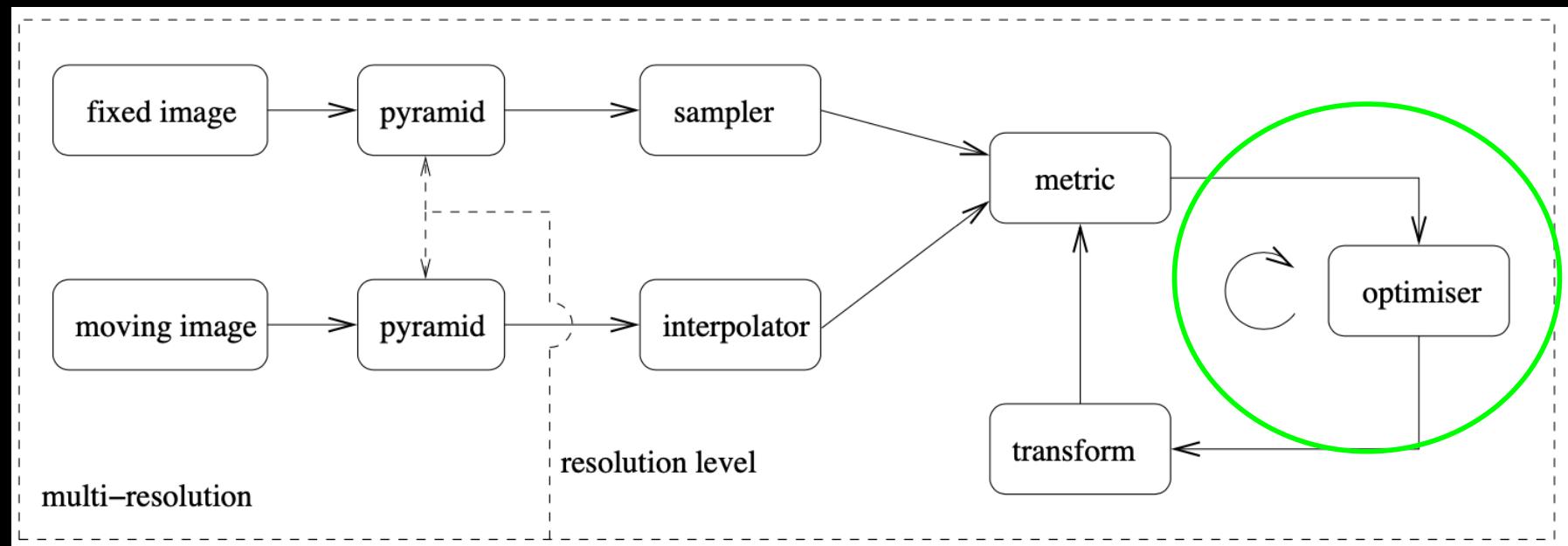
# Contrast in joint histograms

- The histogram of the two images must reflect contrast to similar structures for image registration to be successful



# Image Registration pipeline

- The optimiser
  - How to find the transformation parameters?



# The optimizer

- We have an **objective function** describing:
  - A **cost function ( $C$ )** based on a **similarity metric**
    - Quantifying how well a **geometrical transformation ( $T(w)$ )** maps an image (moving,  $I_M$ ) into another (fixed,  $I_F$ )
- Hence, a good match is a minimum difference:

$$\hat{T}_w = \arg \min_{T_w} C(T_w; I_F, I_M)$$

# The parameters

$$w \in \mathcal{R}^p$$

- The parameters is a vector with p elements
- The type of transformation and the dimension of the dataset set the number of parameters
  - Translation p = 2 or 3 (3D)
  - Rotation p = 1 or 3 (3D)
  - Scaling p = 1

# Optimization by minimization

- Find the parameter set that minimizes the objective function
- How to find the solution?
  - Analytical: Works fine for translation
  - Numerical: Iterative approaches to search for affine transformations

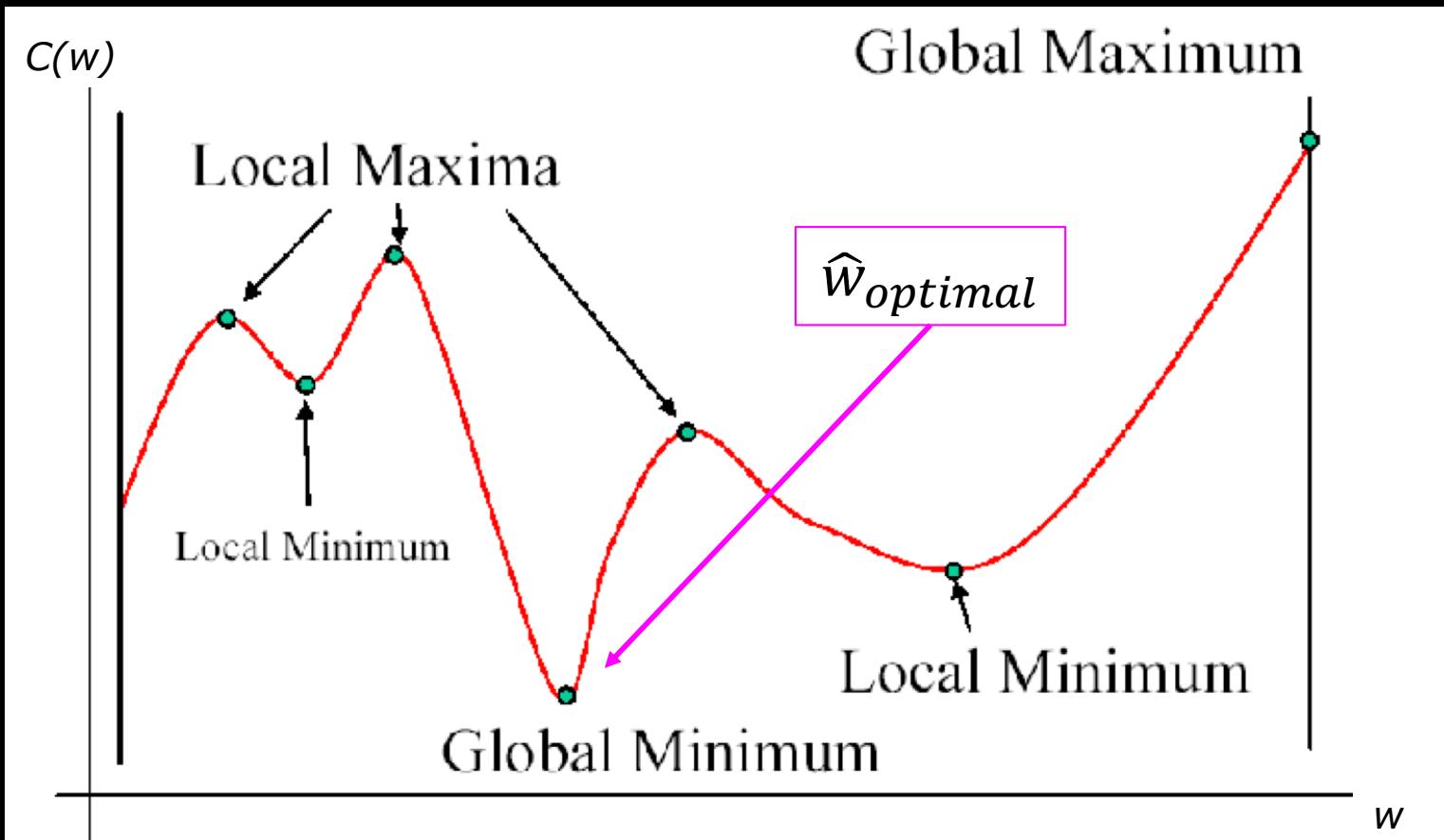
To find:  $\hat{w} = \arg \min_w C$

We simply differentiate w.r.t.  $w$ :

$$\frac{\partial C}{\partial w} = 0$$

# The challenge

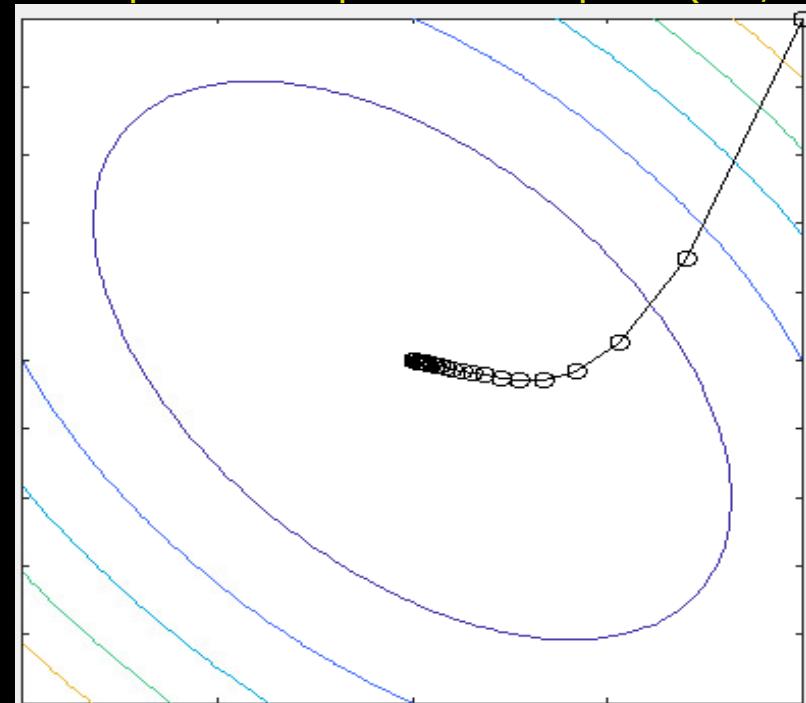
- $w$  span a p-dimensional space  $w = [w_1, w_2, \dots, w_p]^T$
- Complex parameter space with many data points
  - Finding the lowest place in mountains



# Iterative optimisation

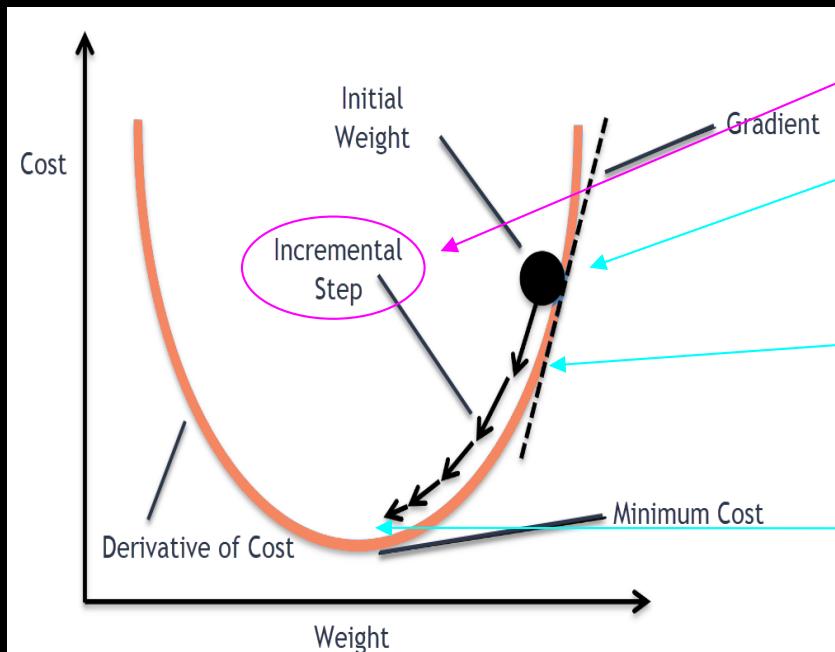
- Aim: Find in parameter space  $w$ :  $\frac{\partial C}{\partial w} = 0$  i.e. a global minima
  - Search all possible combinations of  $w$ ? (not a good idea)
  - Systematically search the parameter space = Good idea
- Iterative optimisation strategies
  - Step-wise searching the parameter space
- Many methods exist
  - Gradient based
  - Genetic evolution
  - ...

Contour plot of 2D parameter space ( $w_1, w_2$ )



# Gradient descent

- Definition:  $C(\mathbf{w})$  is differentiable in neighbourhood of a point  $w_n$
- $C(\mathbf{w})$  decreases in the *negative* gradient direction of  $w_n$ .
- $w_{n+1} = w_n - \gamma \nabla C(w_n)$ 
  - $\nabla C(w_n)$ : Gradient direction at point  $w_n$
  - $\gamma$ : Step length --> If small enough:  $C(w_n) \geq C(w_{n+1})$



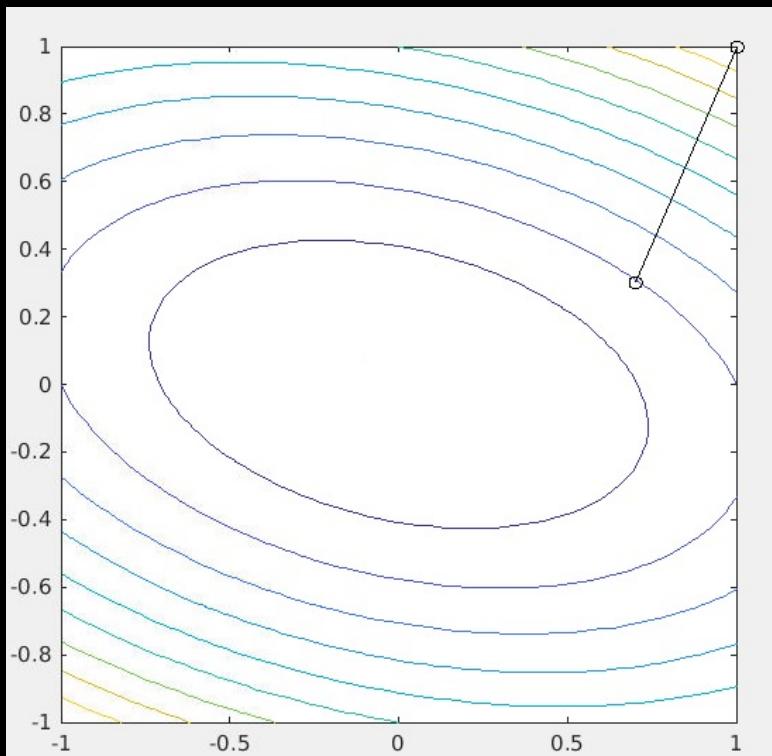
## Procedure:

- 0) Define a step length  $\gamma$
- 1) Start guess of a position  $\nabla C(w_0)$
- 2) Find gradient  $\nabla C(w_1)$
- 3) Take a step
- 4) Repeat 2)+3)
- 5) Solution: Global minima  $\nabla C(w_{n+1}) = \frac{\partial C}{\partial w} \approx 0$

# Gradient descent

- Cost function:  $C(x) = x_1^2 + x_1x_2 + 3x_2^2$
- Gradient at point  $x_n$ :  $-\nabla C(x_n) = -\begin{bmatrix} 2x_1 + x_2 \\ x_1 + 6x_2 \end{bmatrix}$
- Step length:  $\gamma=0.1$ ;
- Max steps: 1000
- Start position:  $x_0=[1,1]^T$

Iteration: 1

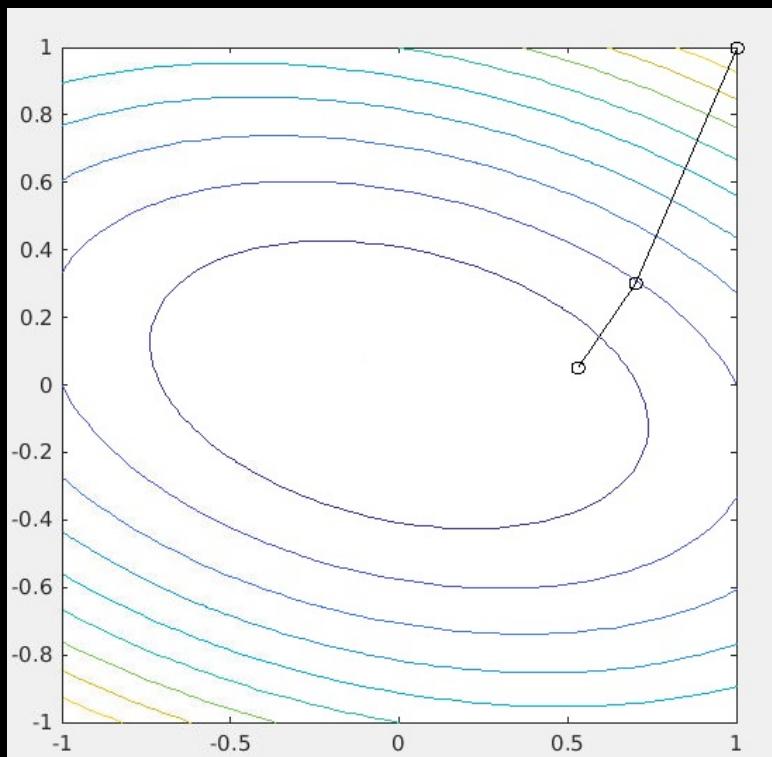


From a Matlab function: *grad\_descent.m*  
By James T. Allison

# Gradient descent

- Cost function:  $C(x) = x_1^2 + x_1x_2 + 3x_2^2$
- Gradient at point  $x_n$ :  $-\nabla C(x_n) = -\begin{bmatrix} 2x_1 + x_2 \\ x_1 + 6x_2 \end{bmatrix}$
- Step length:  $\gamma=0.1$ ;
- Max steps: 1000
- Start position:  $x_0=[1,1]^T$

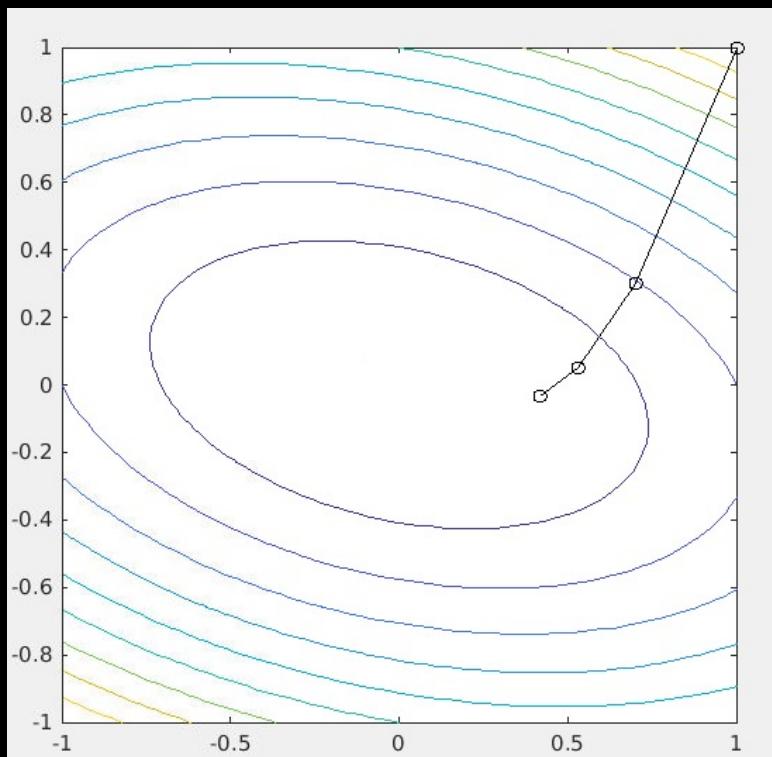
Iteration: 2



# Gradient descent

- Cost function:  $C(x) = x_1^2 + x_1x_2 + 3x_2^2$
- Gradient at point  $x_n$ :  $-\nabla C(x_n) = -\begin{bmatrix} 2x_1 + x_2 \\ x_1 + 6x_2 \end{bmatrix}$
- Step length:  $\gamma=0.1$ ;
- Max steps: 1000
- Start position:  $x_0=[1,1]^T$

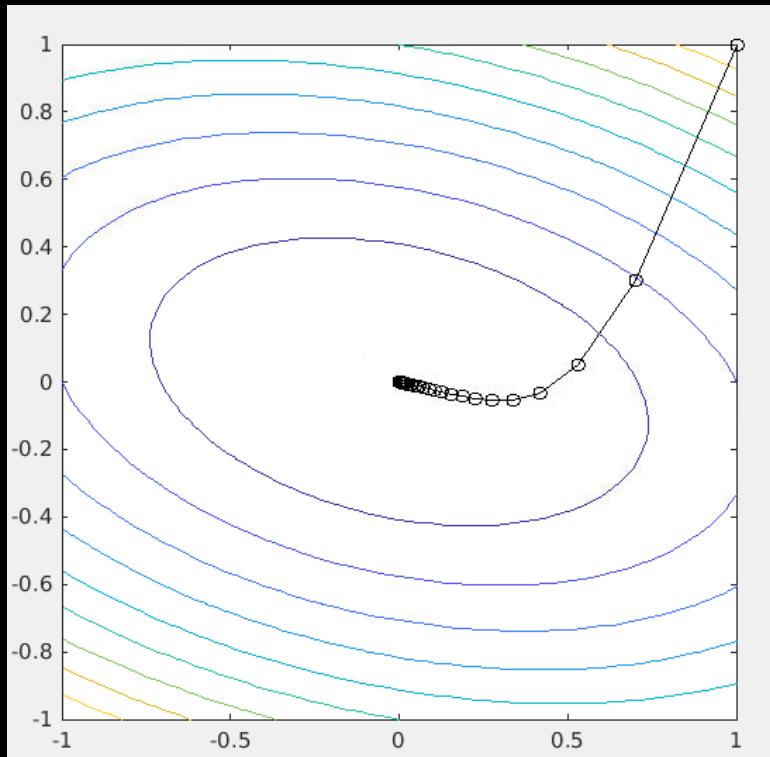
Iteration:3



# Gradient descent

- Cost function:  $C(x) = x_1^2 + x_1x_2 + 3x_2^2$
- Gradient at point  $x_n$ :  $-\nabla C(x_n) = -\begin{bmatrix} 2x_1 + x_2 \\ x_1 + 6x_2 \end{bmatrix}$
- Step length:  $\gamma=0.1$ ;
- Max steps: 1000
- Start position:  $x_0=[1,1]^T$

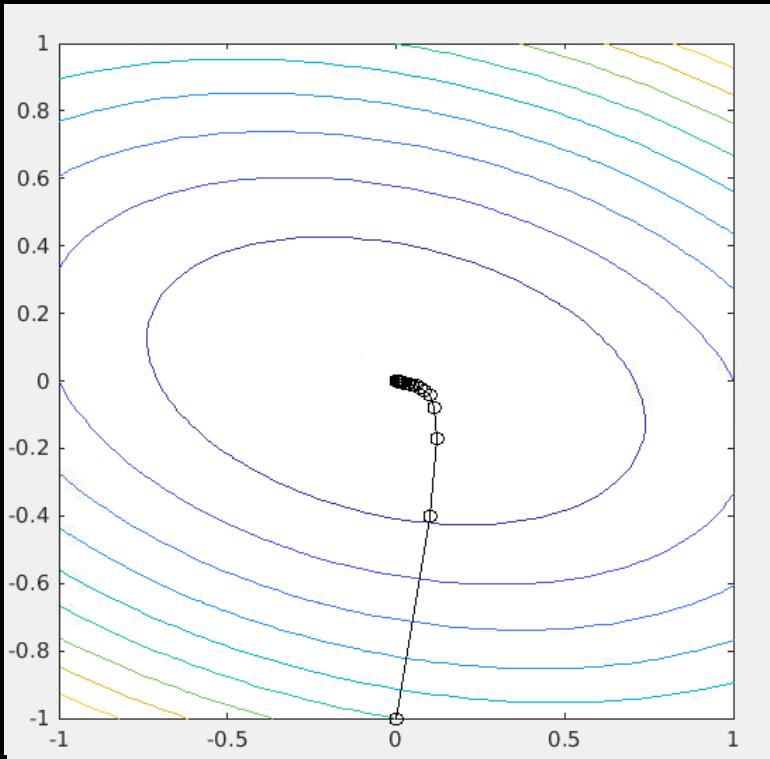
Iteration: 37 (final)



# Gradient descent

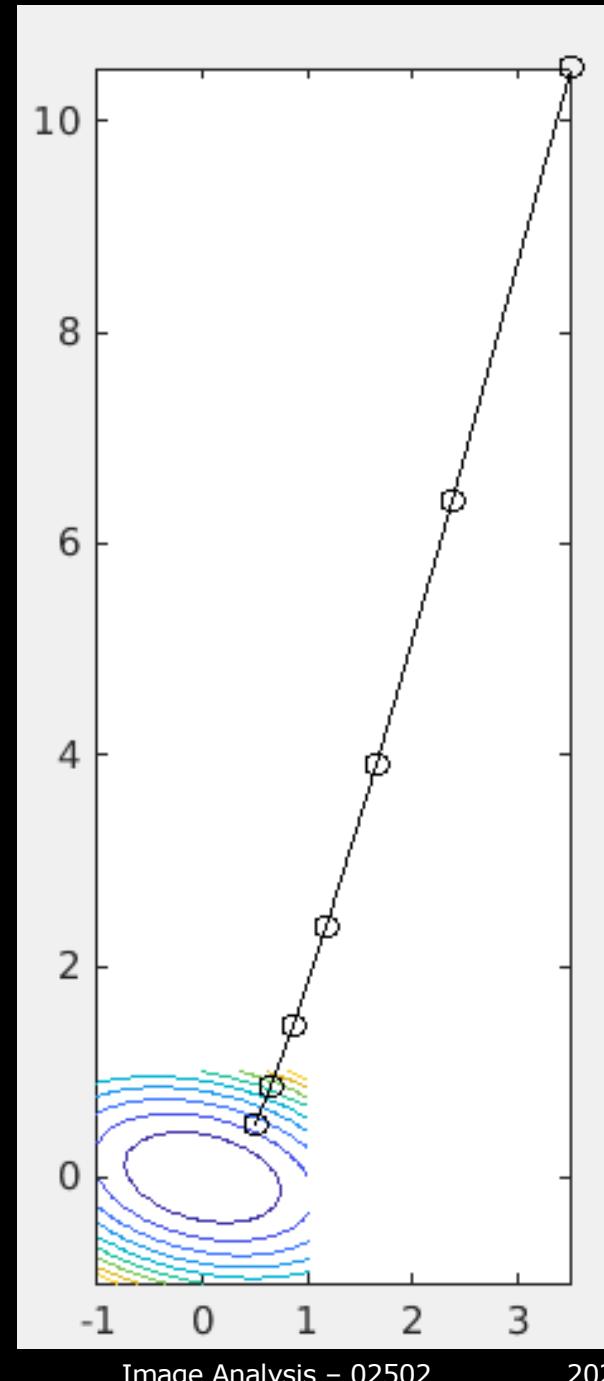
- Cost function:  $C(x) = x_1^2 + x_1x_2 + 3x_2^2$
- Gradient at point  $x_n$ :  $-\nabla C(x_n) = -\begin{bmatrix} 2x_1 + x_2 \\ x_1 + 6x_2 \end{bmatrix}$
- Step length:  $\gamma=0.1$ ;
- Max steps: 1000
- Start position:  $x_0=[0, -1]^T$
- Can find solution from any place
- No local minima's nearby

Iteration: 31 (final)



# Gradient descent

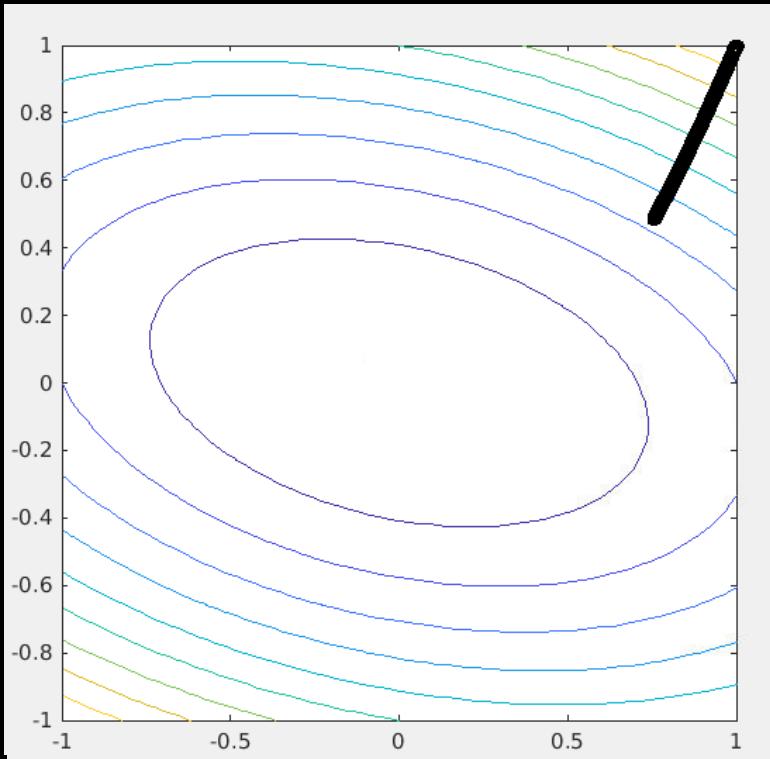
- Cost function:  $C(x) = x_1^2 + x_1x_2 + 3x_2^2$
- Gradient at point  $x_n$ :  $+\nabla C(x_n) = + \begin{bmatrix} 2x_1 + x_2 \\ x_1 + 6x_2 \end{bmatrix}$
- Step length:  $\gamma=0.1$ ;
- Max steps: 1000
- Start position:  $x_0=[0.5,0.5]^T$
- If use positive gradient
  - WRONG DIRECTION!



# Gradient descent

- Cost function:  $C(x) = x_1^2 + x_1x_2 + 3x_2^2$
- Gradient at point  $x_n$ :  $-\nabla C(x_n) = -\begin{bmatrix} 2x_1 + x_2 \\ x_1 + 6x_2 \end{bmatrix}$
- Step length:  $\gamma=0.0001$ ;
- Max steps: 1000
- Start position:  $x_0=[1,1]^T$
- Too small step size –many steps
- Do not find a solution

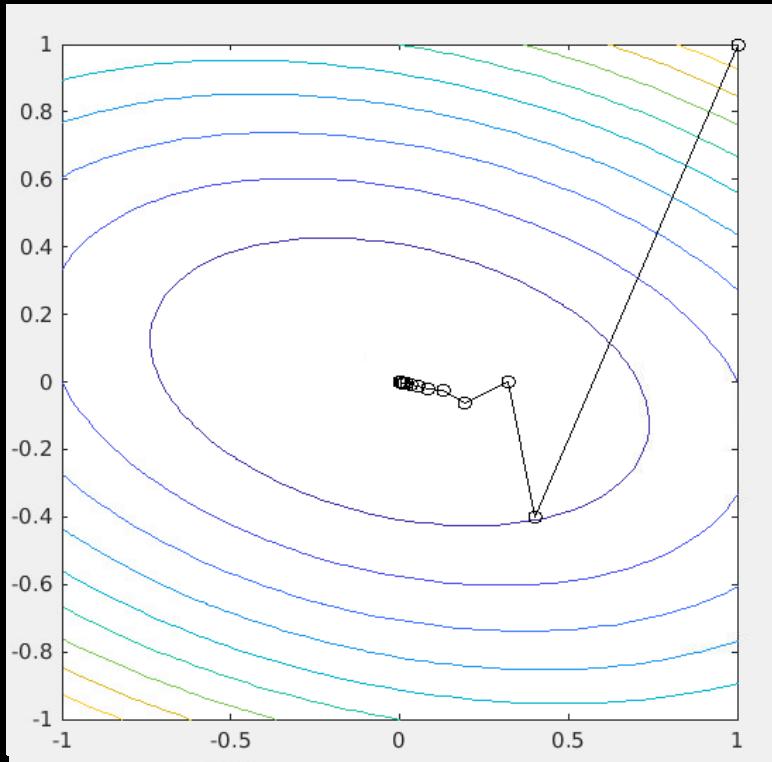
Iteration: 1000 (final)



# Gradient descent

- Cost function:  $C(x) = x_1^2 + x_1x_2 + 3x_2^2$
- Gradient at point  $x_n$ :  $-\nabla C(x_n) = -\begin{bmatrix} 2x_1 + x_2 \\ x_1 + 6x_2 \end{bmatrix}$
- Step length:  $\gamma=0.2$  (optimal)
- Max steps: 1000
- Start position:  $x_0=[1,1]^T$
- Few steps: Optimal step size

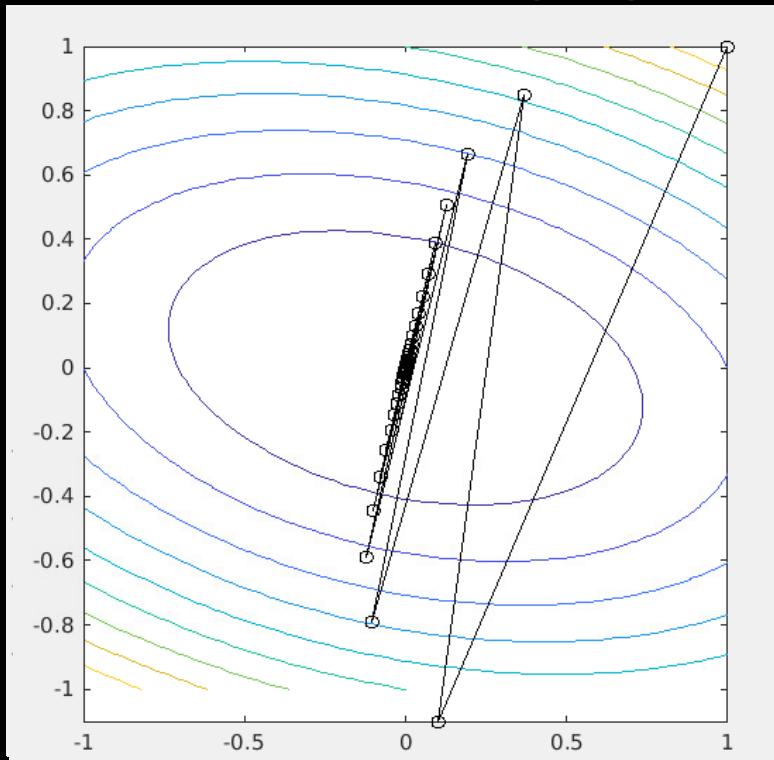
Iteration: 17 (final)



# Gradient descent

- Cost function:  $C(x) = x_1^2 + x_1x_2 + 3x_2^2$
- Gradient at point  $x_n$ :  $-\nabla C(x_n) = -\begin{bmatrix} 2x_1 + x_2 \\ x_1 + 6x_2 \end{bmatrix}$
- Step length:  $\gamma=0.3$
- Max steps: 1000
- Start position:  $x_0=[1,1]^T$
- Too large step size – unstable
- Sensitive to local minima's
- Solution: Dynamic step length

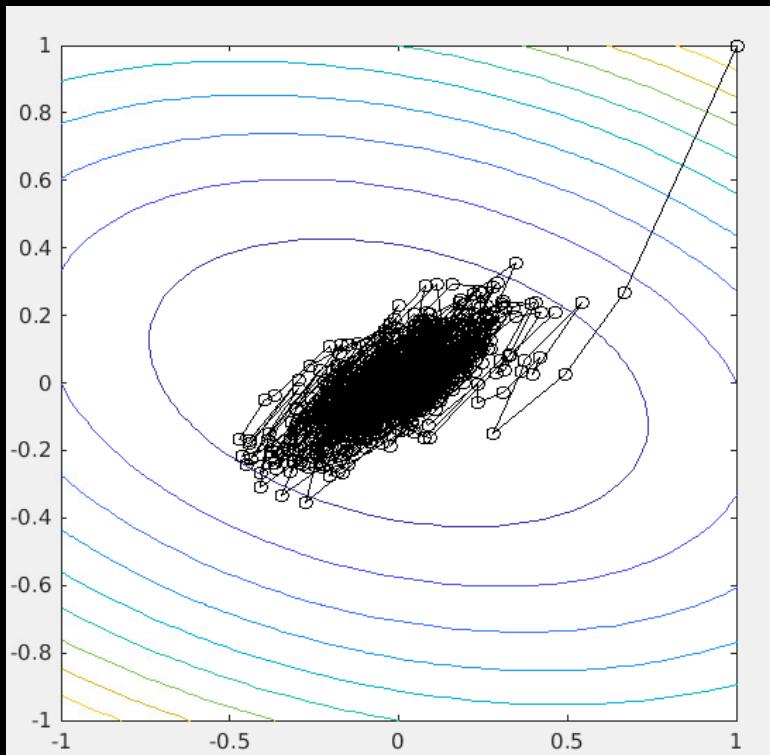
Iteration: 65 (final)



# Gradient descent

- Cost function:  $C(x) = x_1^2 + x_1x_2 + 3x_2^2$
- Gradient at point  $x_n$ :  $-\nabla C(x_n) = -\begin{bmatrix} 2x_1 + x_2 \\ x_1 + 6x_2 \end{bmatrix}$
- Step length:  $\gamma=0.1$
- Max steps: 1000
- Start position:  $x_0=[1,1]^T$
- Noisy data: Cannot find optimum

Iteration: 1000 (final)





# Quiz 5: What is the updated position $x_{\text{new}}$ ?

Model fitting uses a cost function:  $C(x) = x_1^2 + x_1x_2 + 3x_2^2$   
and an iterative optimizer Gradient descent with a step length of 0.2

What is the new position of  $x_{\text{new}} = [?, ?]^T$  after one step from position  $x = [1, 0]^T$ ?

- A)  $[0.3, 2.3]^T$
- B)  $[-1.7, 0.3]^T$
- C)  $[1.4, 0.2]^T$
- D)  $[0.6, -0.2]^T$
- E)  $[5.2, 2.2]^T$

Solution:

1) Calculate the gradient for  $x = [1, 0]^T$

- differentiate  $C(x)$ :  $\nabla C(x) = \begin{bmatrix} 2x_1 + x_2 \\ x_1 + 6x_2 \end{bmatrix}$

$$\nabla C([1, 0]^T) = [2, 1]^T$$

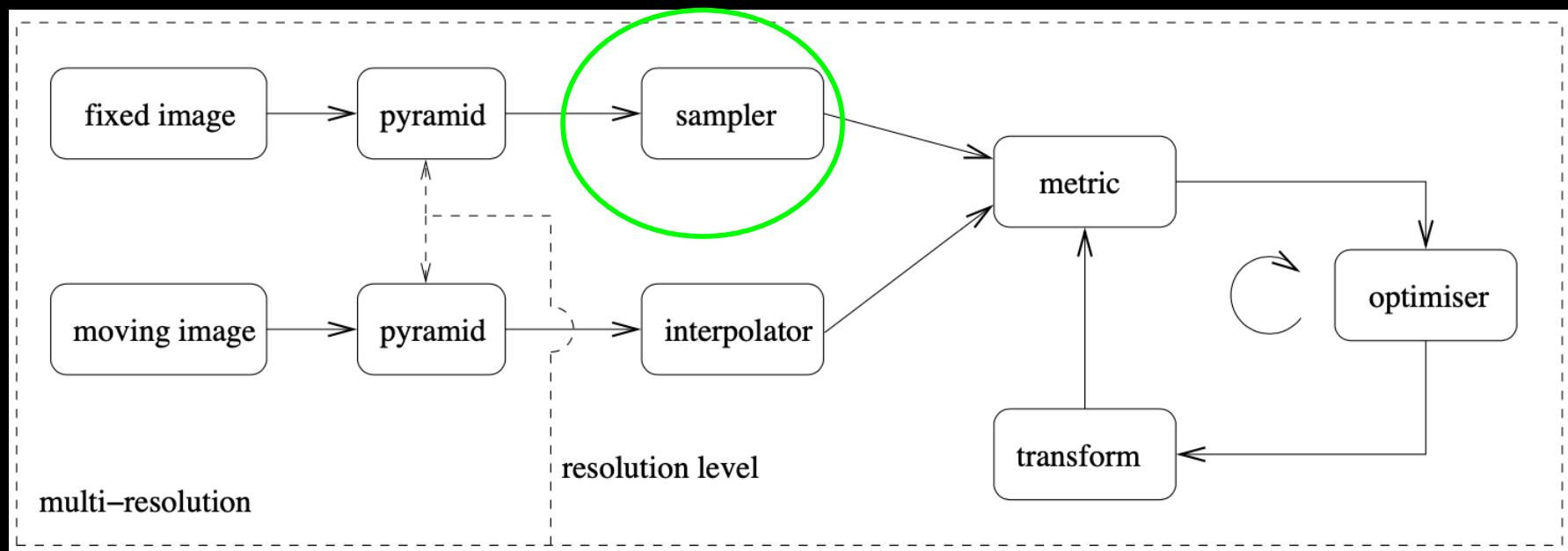
2) Update the step:  $x_{\text{new}} = x - \nabla C * \text{stepLength}$

- $x_{\text{new}} = [1, 0]^T - 0.2 * [2, 1]^T = [0.6, -0.2]^T$

# Image Registration pipeline

## ■ The sampler

- How many data points for a robust similarity measure?





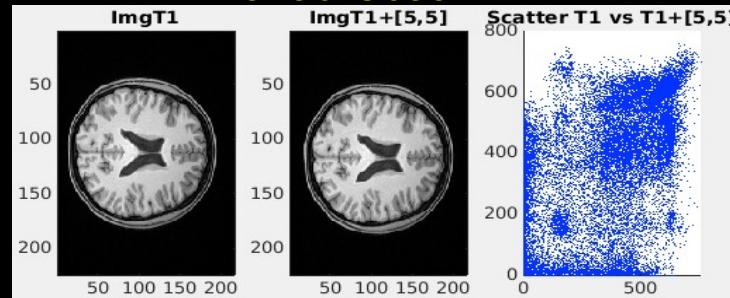
# The sampler

- Calculating the similarity metrics:
  - Summing over all pixels/voxels in an image is VERY time consuming
- Selecting a sparse sampling strategy
  - Reducing CPU load and reduce memory load when
  - Efficient selection of image points



# The sampler

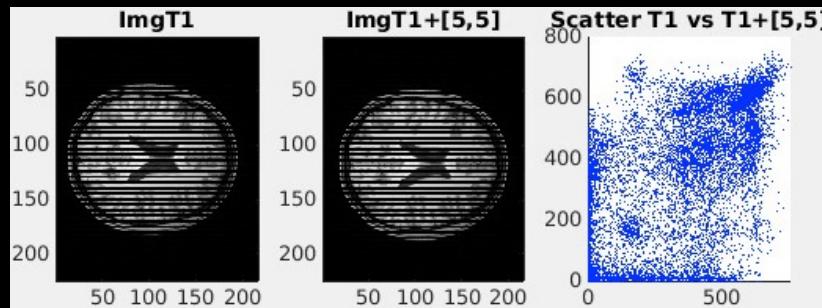
All samples



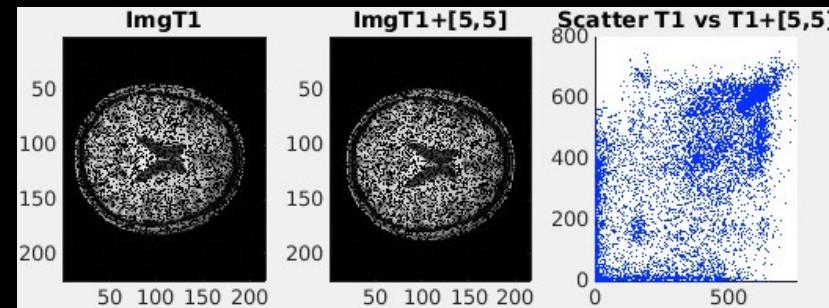
- Sparser sampling: Similar scatter plot
  - Define a good compromise (sample the whole image)
- Ordered vs Random
  - Spatial dependency: Dependent on large homogeneous structures
  - Very sparse sampling: Risk not sampling small structures

Every 2nd

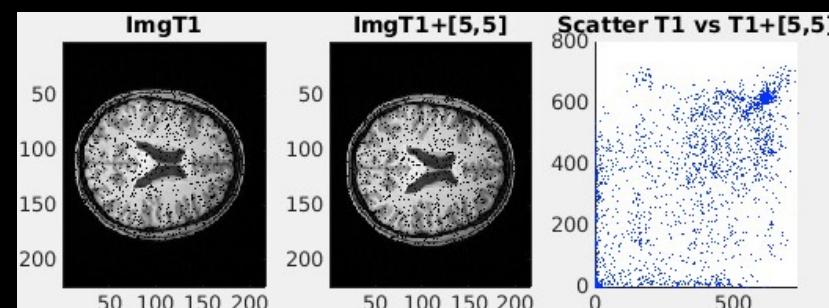
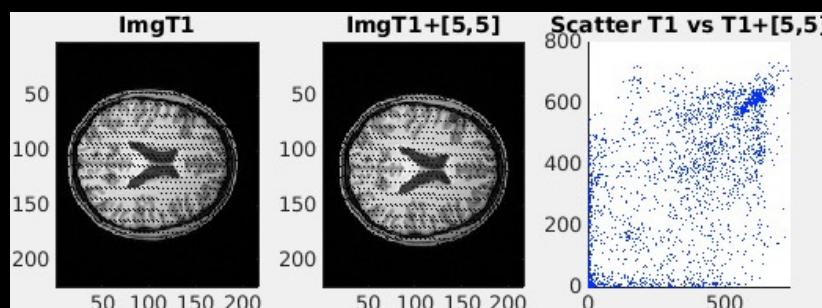
Ordered



Random



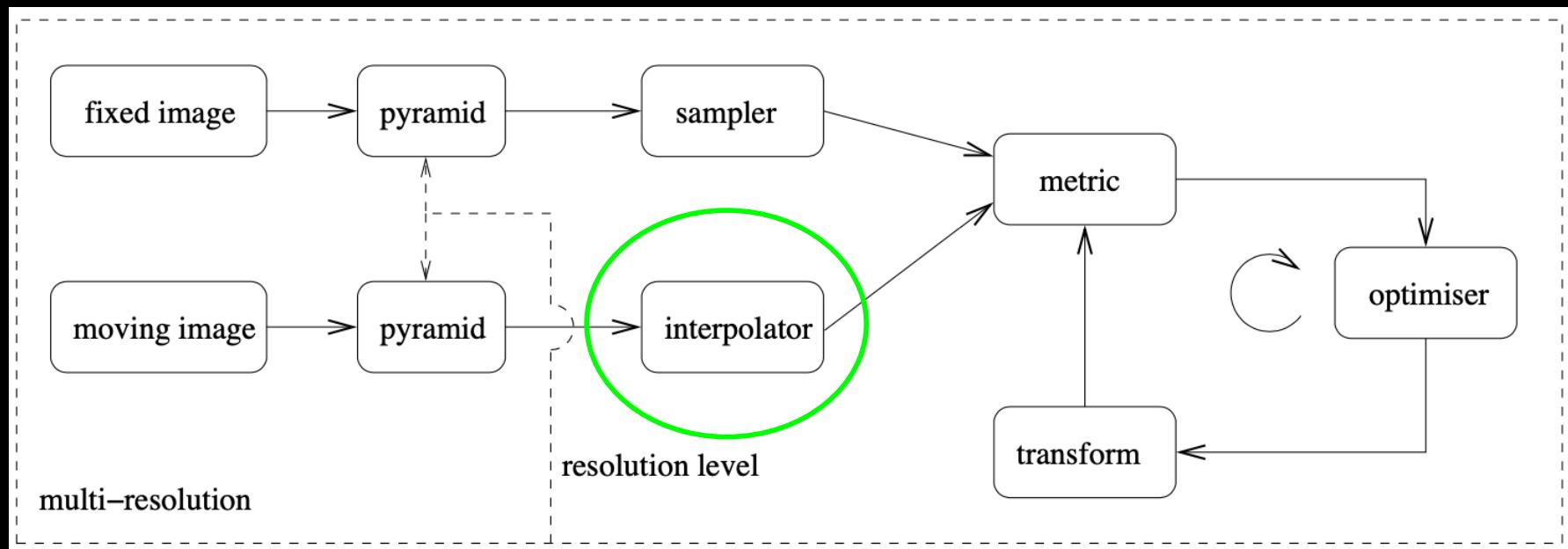
Every 10th



# Image Registration pipeline

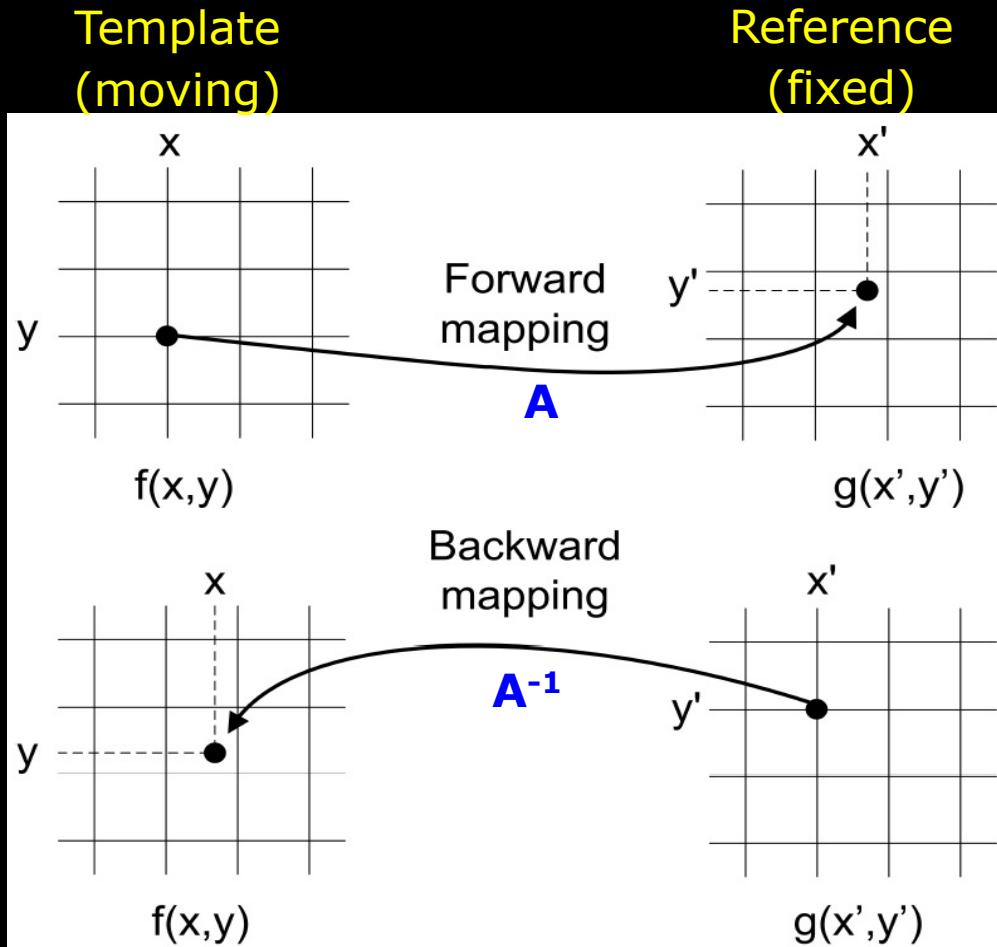
## ■ Interpolation

- To map the intensities from the template image to the grid of the reference image via a transformation matrix



# A FLASH BACK to a previous Lecture: Forward vs Backward mapping

- In a nutshell
  - Going backward we need to invert the transformation



# Interpolation methods

- Enhances structural boundaries
  - Higher-order interpolation methods: Reduce blurring
- May visually appear “sharper”
  - Do not change the image information!
  - Only if combining interpolated images w. different information of the same object – e.g. different angles of moving object e.g. car  
→ Super resolution (another topic)

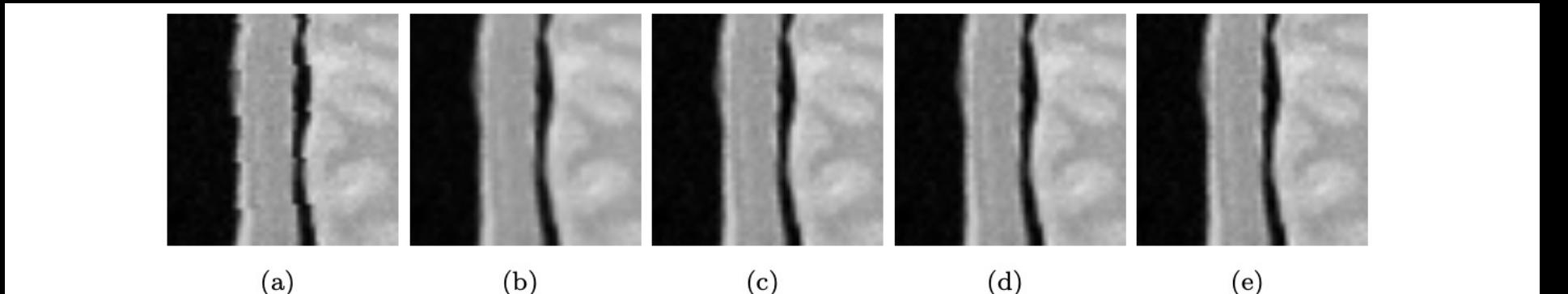
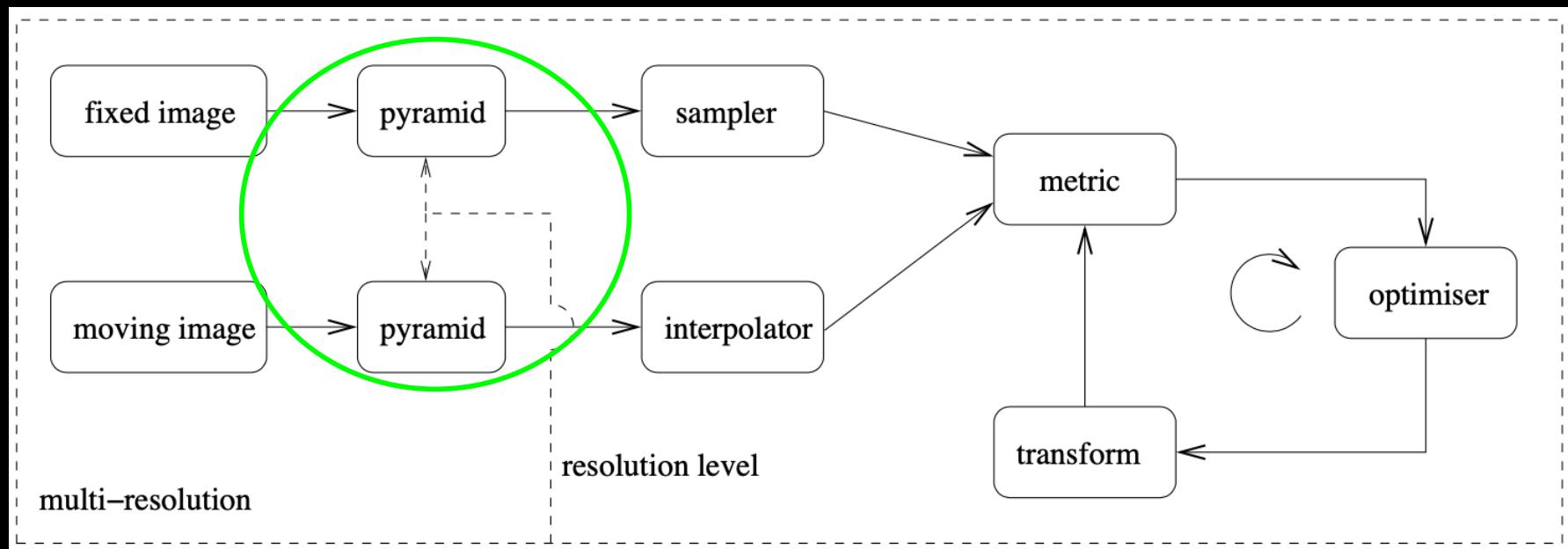


Figure 2.4: Interpolation. (a) nearest neighbour, (b) linear, (c) B-spline  $N = 2$ , (d) B-spline  $N = 3$ , (e) B-spline  $N = 5$ .

# Image Registration pipeline

## ■ Pyramid



# The Pyramid Principle

- To ensure robust image registration

Some stones?



Pretty close



Walking distance



From a bird



From space?



Very detailed

Good overview

Too coarse

# The Pyramid Principle

- To ensure robust image registration

Some stones?



Pretty close



Walking distance



From a bird



From space?



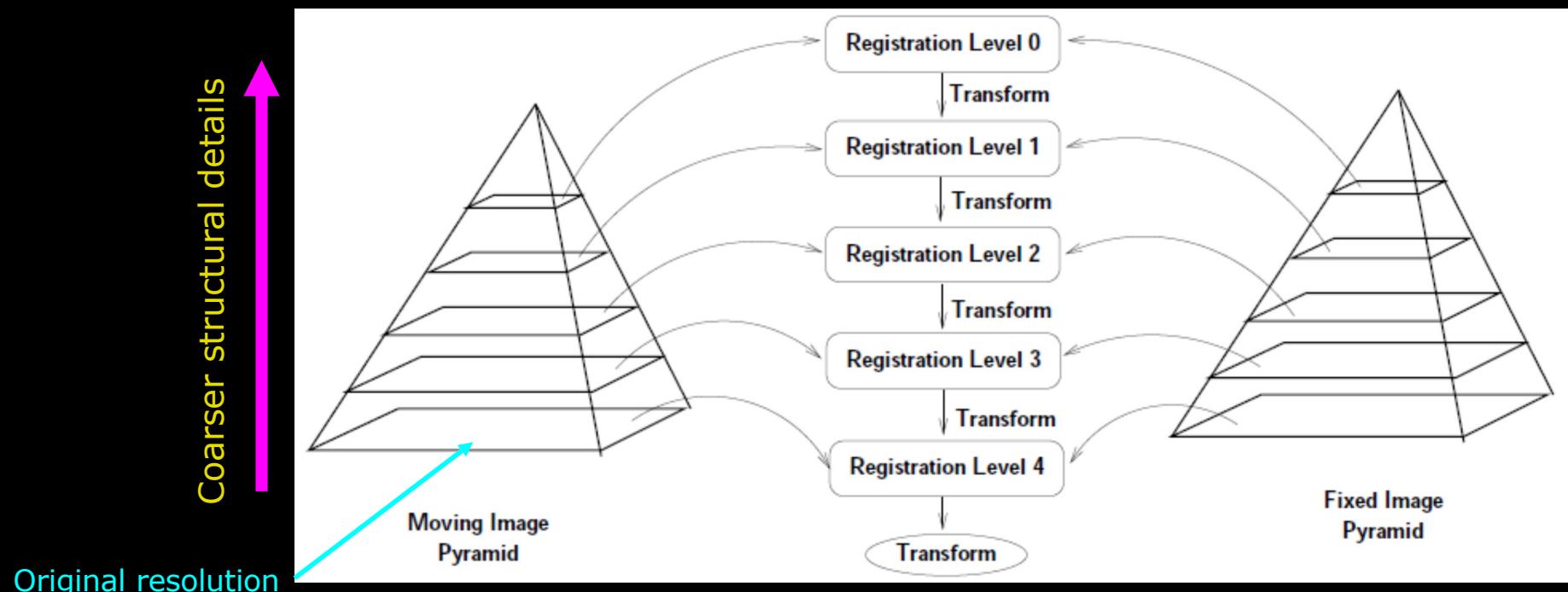
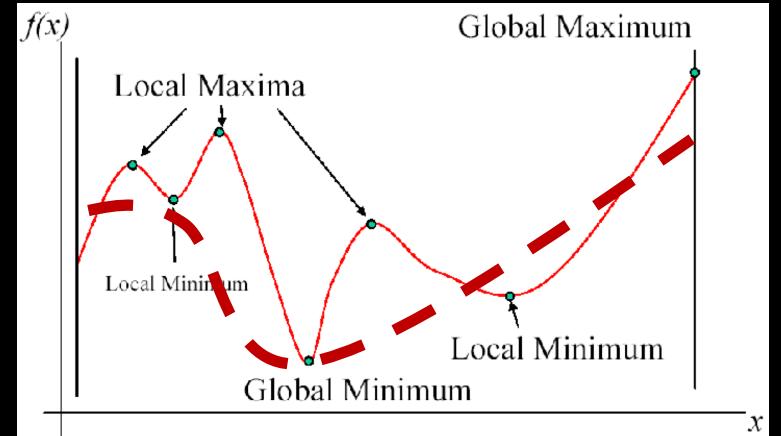
Very detailed

Good overview

Too coarse

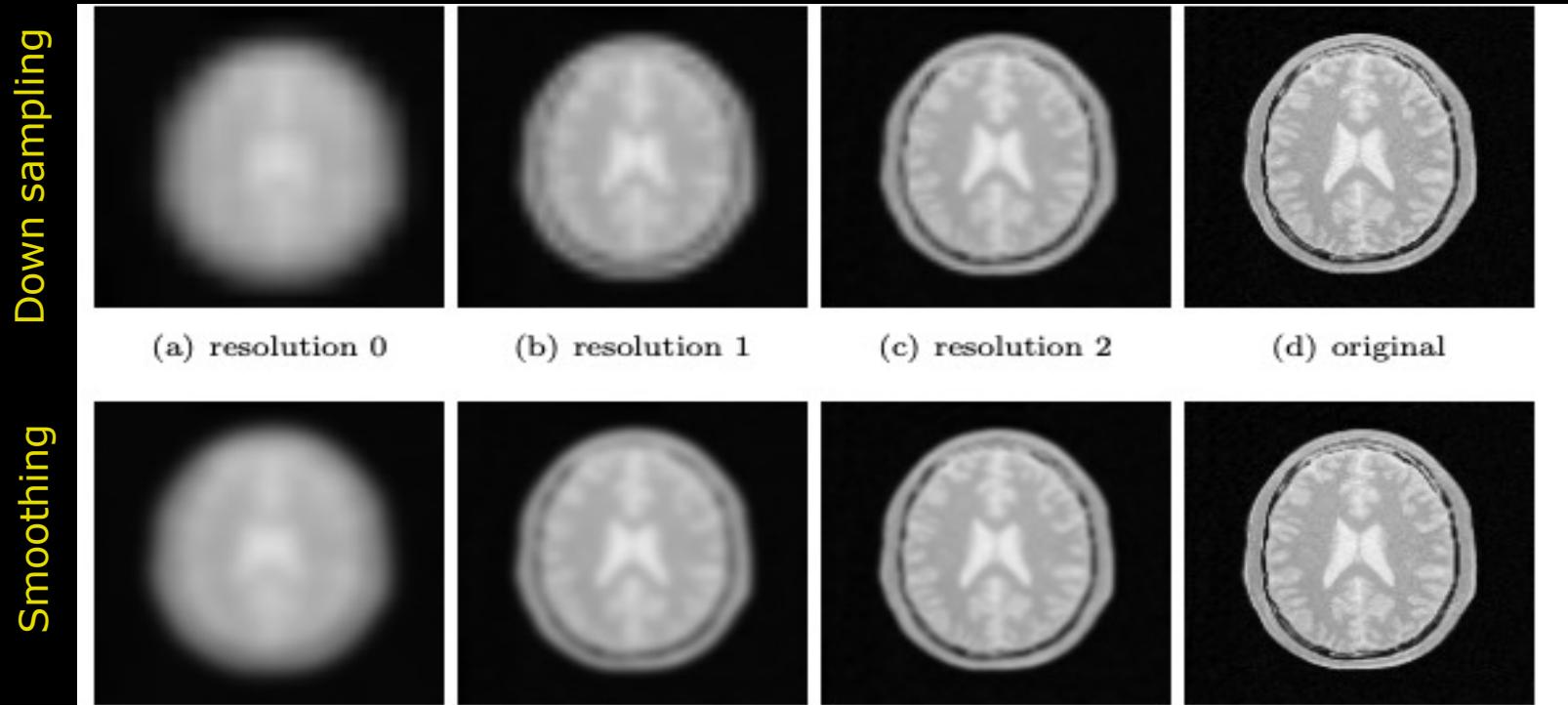
# The Pyramid Principle

- A Multi-resolution strategy
- To ensure robust image registration
  - To reduce local minima's
  - What is a proper image resolution level ?



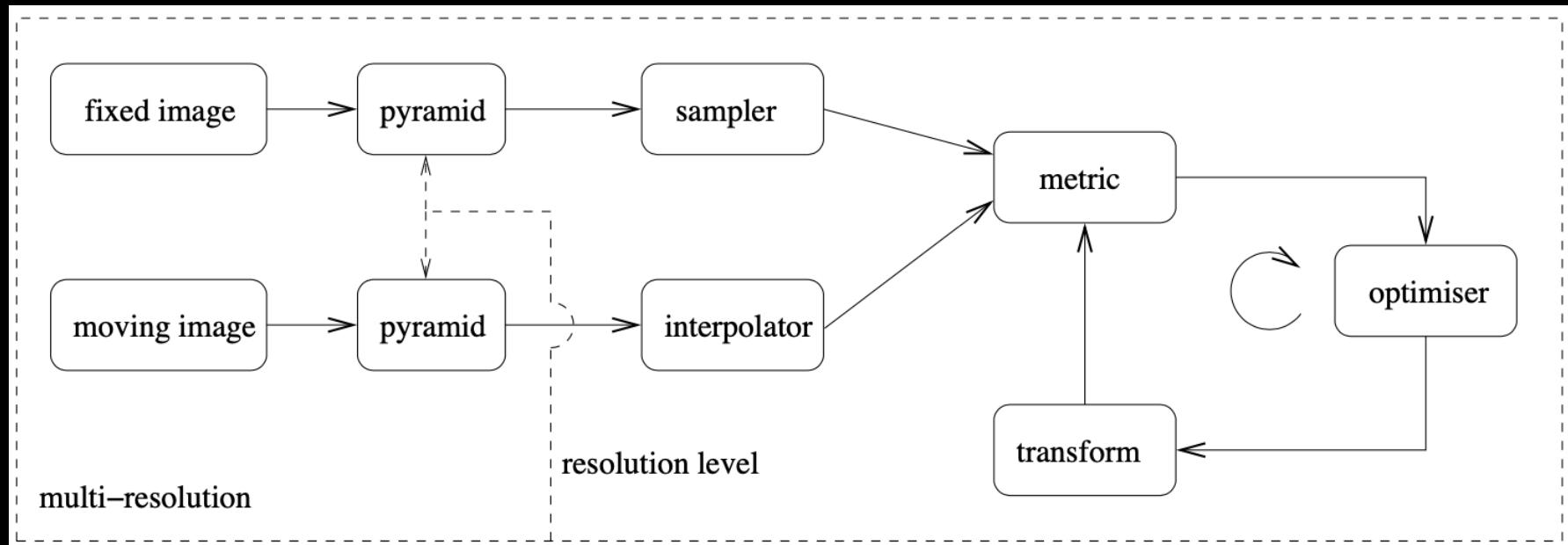
# The Pyramid Principle

- Lower image resolution
  - Down sampling (memory reduction, fewer data)
- Less structural details
  - Smoothing (Complex method settings become more general)



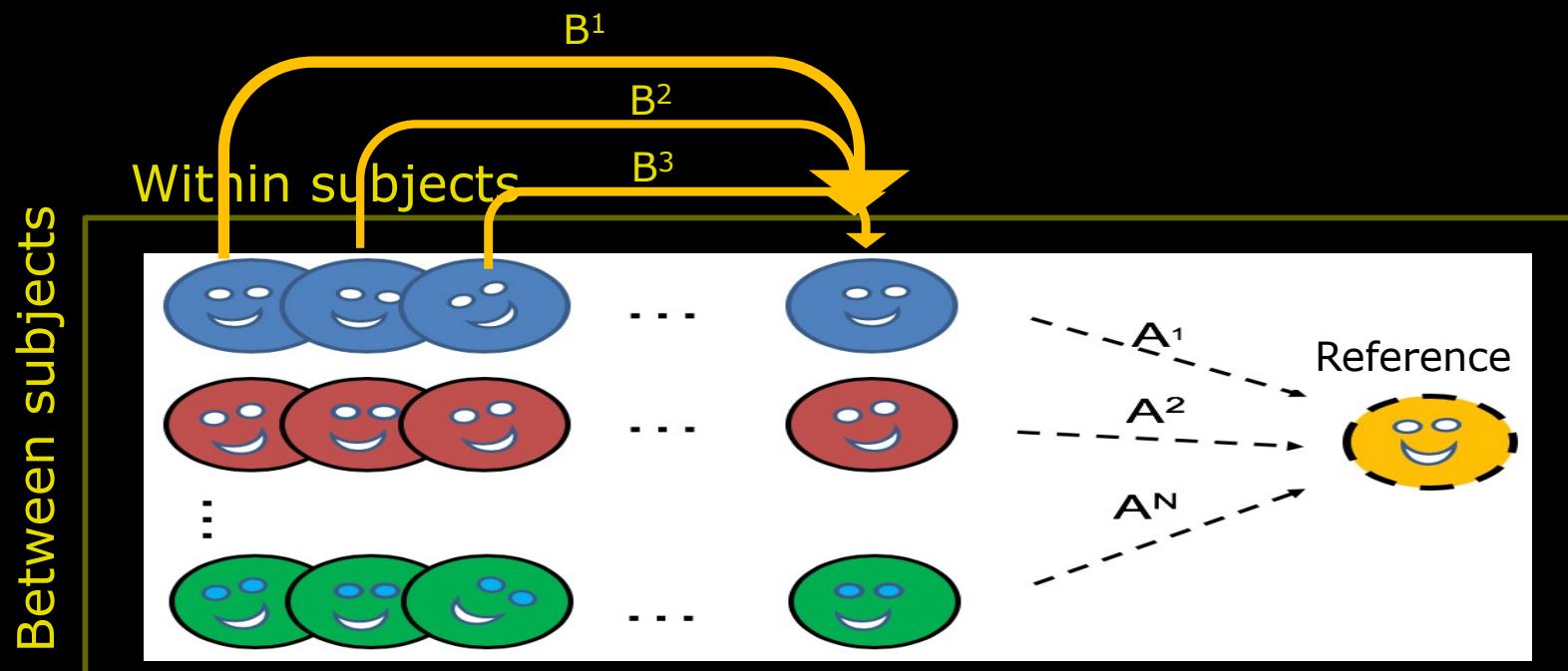
# Image Registration pipeline

- At the end we just select an existing tool
- Still, we need how too select method settings ☺
  - This was the first step in the registration pipeline



# Combining Image Registration pipelines

- First step : Within subjects (Same structure + temporal)
- Second step: Between subjects (different structure+ temporal)
  - Can use an iterative procedure to improve registration
- Combine subject-wise transformation metrics by multiplication
  - Apply only one interpolation at the end to minimise blurring





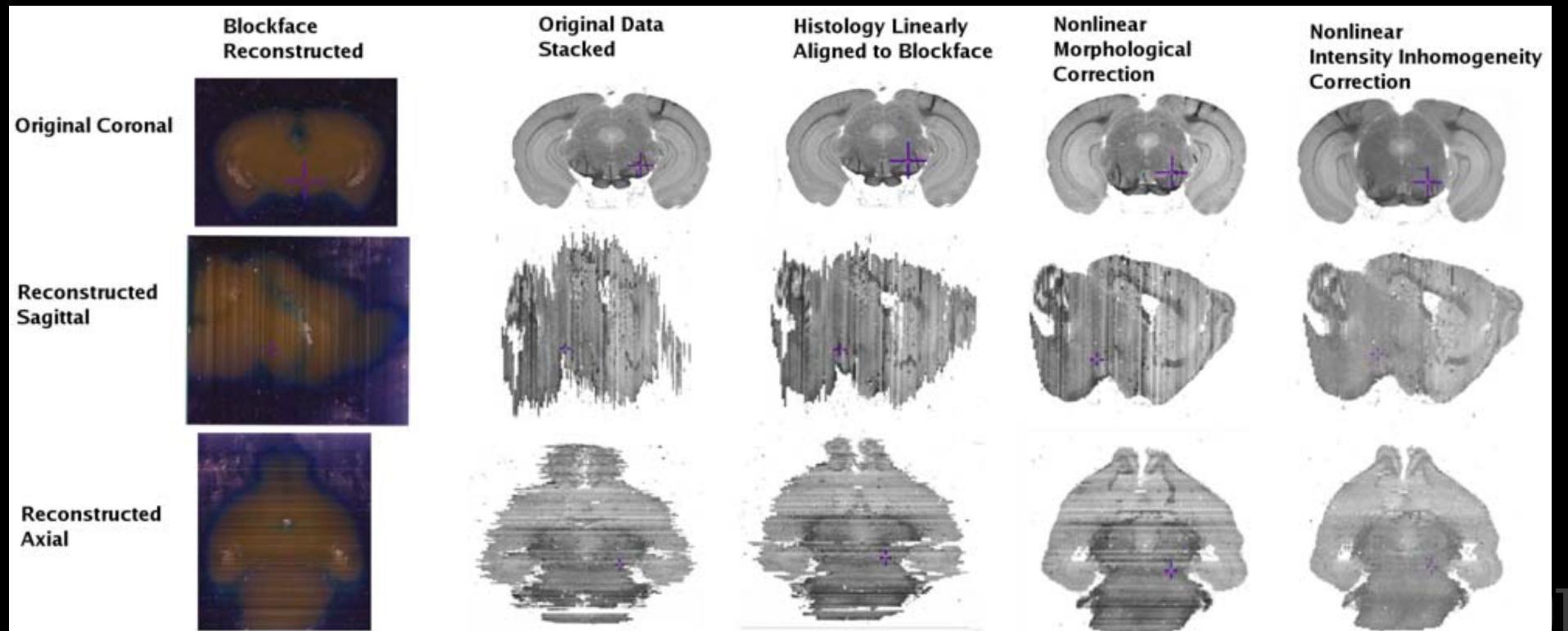
# Quiz 6: Quality inspection - How

How to quality assurance (QA) the image registration results?

- A) Use a similarity measure
- B) Visual inspection
- C) No need it to - just works
- D) Sum of square difference
- E) Search the internet for experience

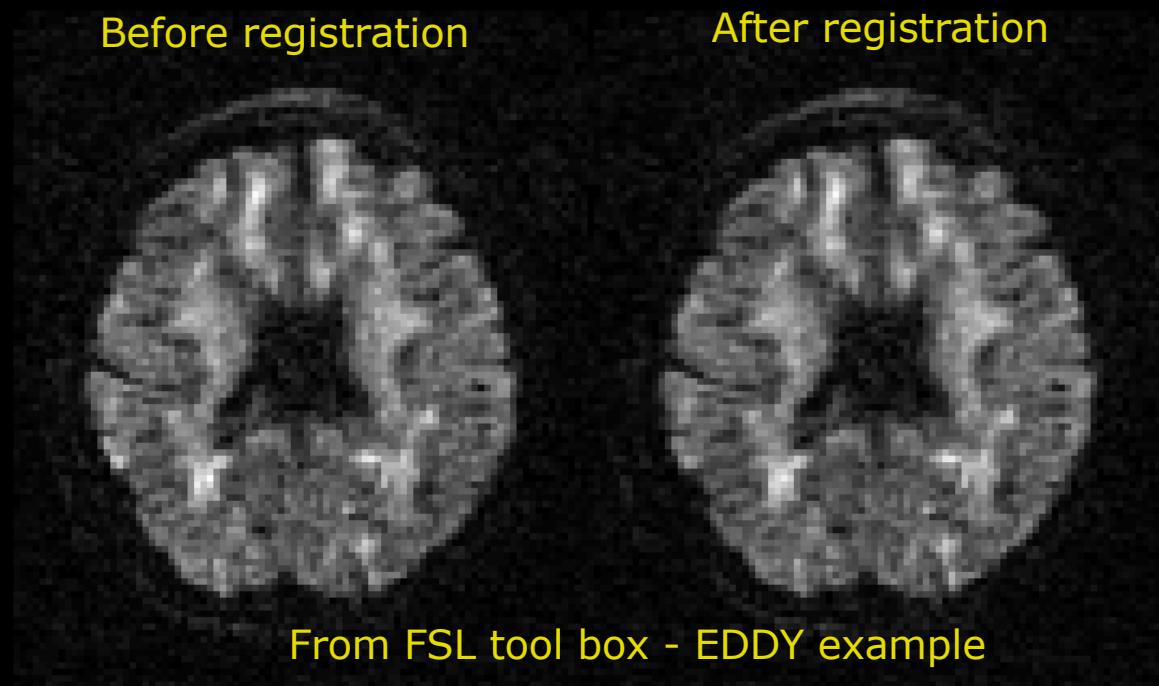
# Image Registration pipeline strategy

- Within subjects and between challenges
  - E.g. Histology 2D → 3D: Structural difference between slices
  - Visually inspect your results!!



# Image Registration pipeline strategy

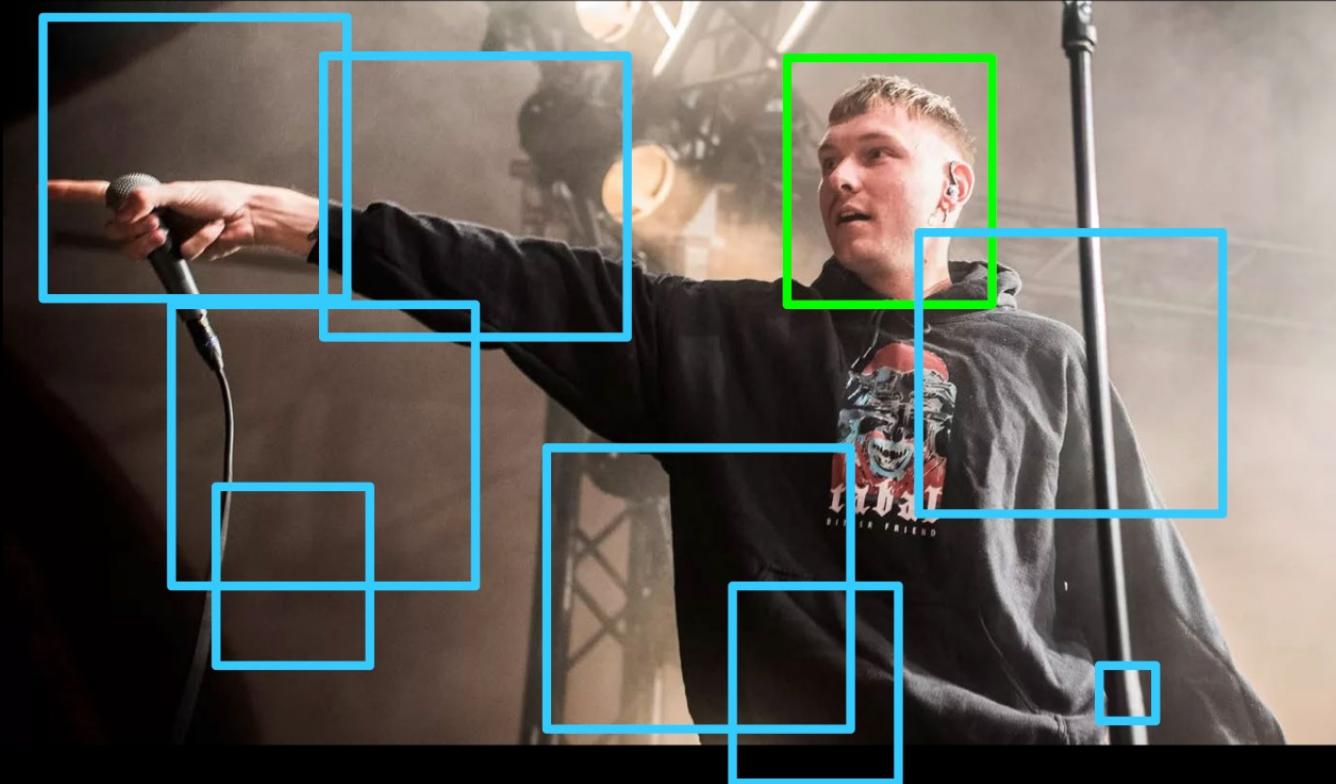
- Within subjects across time points (temporal)
  - Remove image distortions + subjection motion
- Visually inspect your results!!



# What can you do after today?

- Describe difference between a pixel and voxel
- Choose a general image-to-image registration pipeline
- Apply 3D geometrical affine transformations
- Use the Homogeneous coordinate system to combine transformations
- Compute a suitable intensity-based similarity metric given the image modalities to register
- Compute the normalized correlation coefficient (NNC) between two images
- Compute Entropy
- Describe the concept of iterative optimizers
- Compute steps in the gradient descent optimization algorithm
- Apply the pyramidal principle for multi-resolution strategies
- Select a relevant registration strategy: 2D to 3D, Within- and between objects and moving images

# Next week – Real-time face detection using Viola Jones method





# Image Analysis

Rasmus R. Paulsen

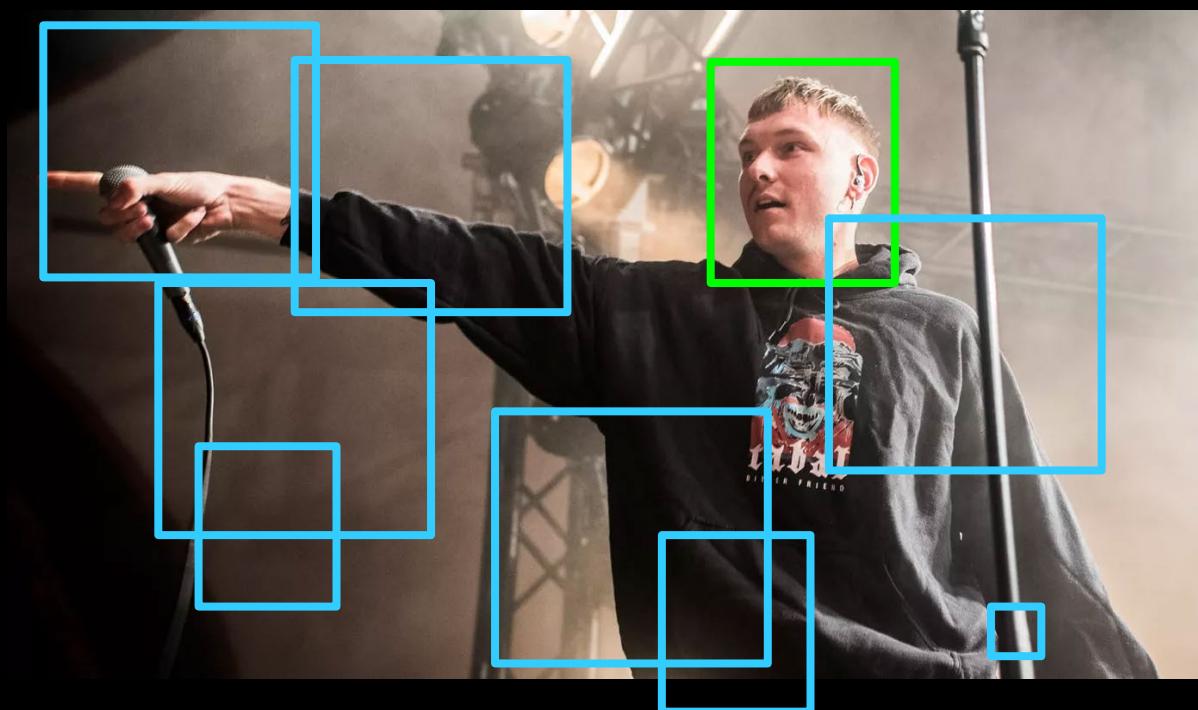
Tim B. Dyrby

DTU Compute

[rappa@dtu.dk](mailto:rappa@dtu.dk)

<http://www.compute.dtu.dk/courses/02502>

# Lecture 11 – Face detection using the Viola Jones method



# What can you do after today?

- Describe the concept of face detection
- Describe the concept of Haar features
- Compute the values of 2, 3 and 4 rectangle Haar features
- Describe the integral image
- Compute the sum of pixels values in a rectangle using an integral image
- Describe the concept of a weak classifier
- Describe how several weak classifiers can be combined into a strong classifier
- Describe the attentional cascade
- Describe how faces can be detected using a moving window

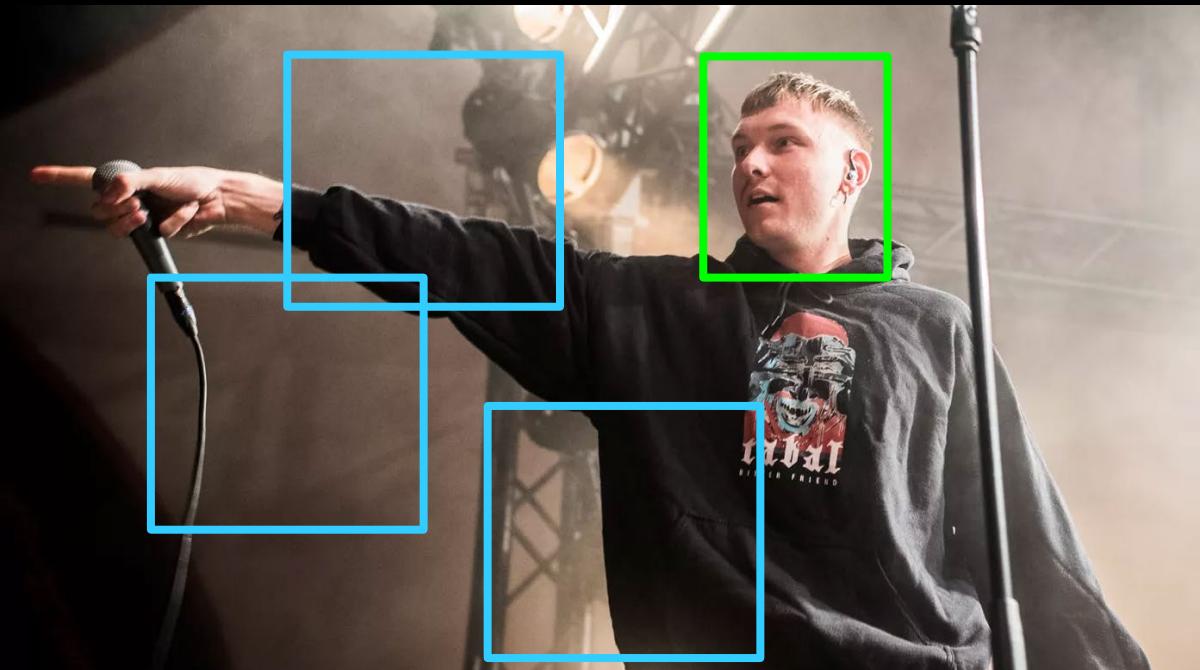
# Face detection



## ■ First problem

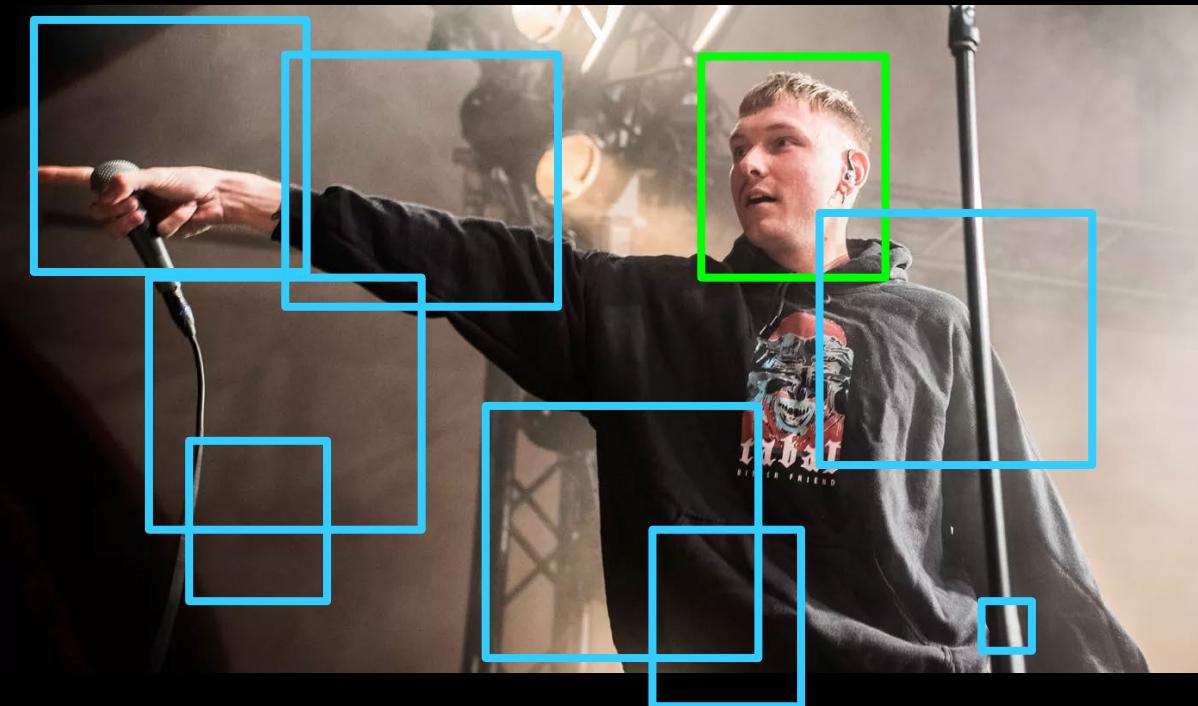
- Analyze a **window** in an image
- Is there a face in that window?

# Face detection



- Slightly more advanced
  - Analyze many **windows** in an image
  - How many (if any) **windows** contain faces?

# Face detection



## ■ Ideal

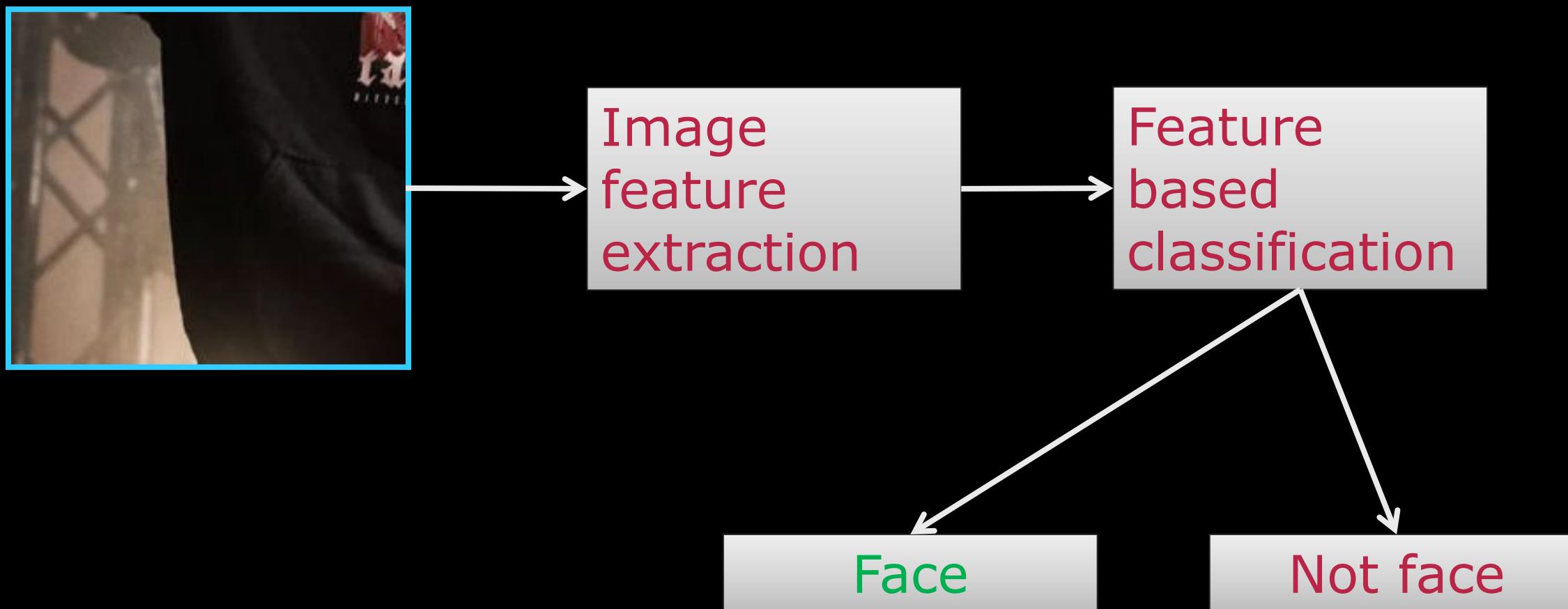
- Analyze (almost) all possible **windows** in an image
- How many (if any) **windows** contain faces?

# What is needed?



- A fast method to determine if a *window* contains a face

# Primary task – image feature based classification



## Image based features - what features can you think of?



landmarks  
angles  
blobs  
blob  
lines  
edges  
entropy  
color  
contrast  
between  
analysis  
shape  
facial pca  
connectivity  
histogram

Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)

# Viola Jones – fast features and smart classification



Many image  
features  
very fast

Boosted  
cascade  
classifier

Face

Not face

# Training data



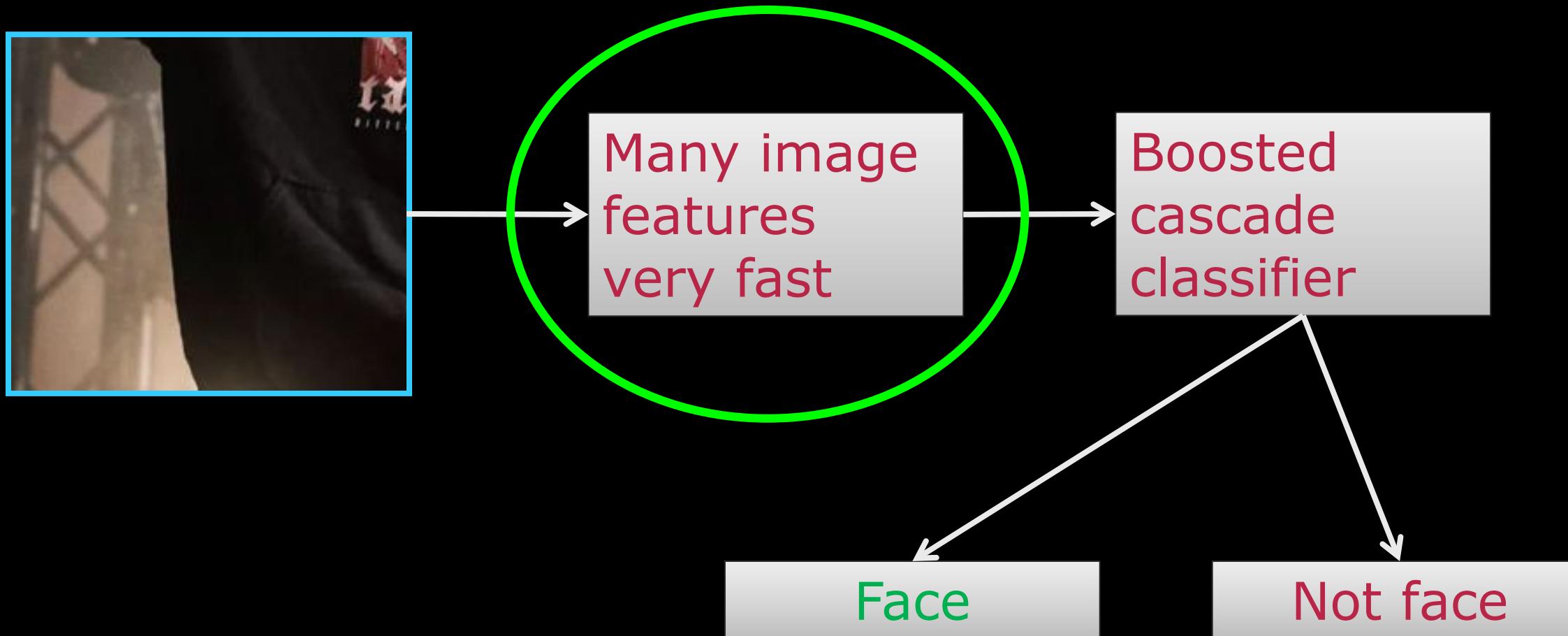
## ■ Face images:

- 4916 hand labelled faces
- Aligned and scaled to 24x24 pixels

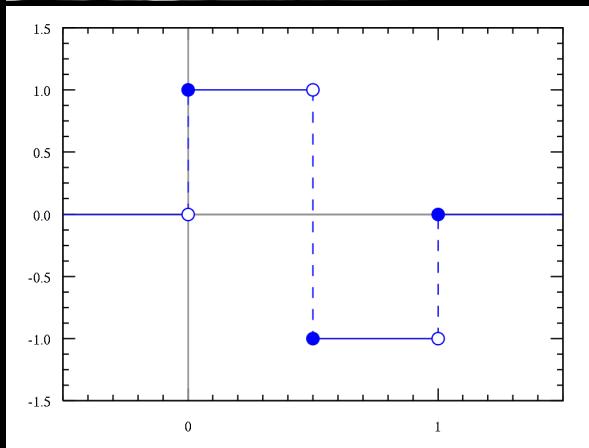
## ■ Non-face images:

- 9544 images with no faces
- 350 million sub-windows sampled from these

# Viola Jones – fast features and smart classification



# Haar features



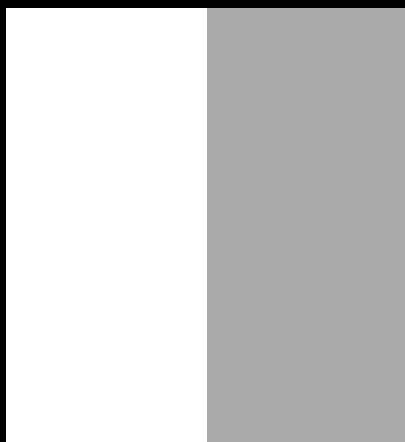
- Alfred Haar (1885-1933)
  - Hungarian Mathematician
- Introduced the Haar wavelet in 1909
- *A wavelet is a wave-like oscillation with an amplitude that begins at zero, increases or decreases, and then returns to zero one or more times.*
- Simplest possible wavelet

<https://en.wikipedia.org/wiki/Wavelet>

[https://en.wikipedia.org/wiki/Haar\\_wavelet](https://en.wikipedia.org/wiki/Haar_wavelet)

# Haar features

Two rectangle features



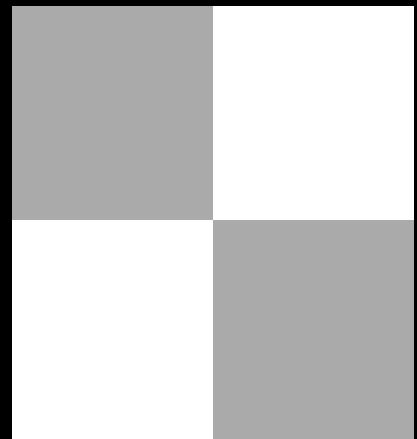
A

Three rectangle feature



C

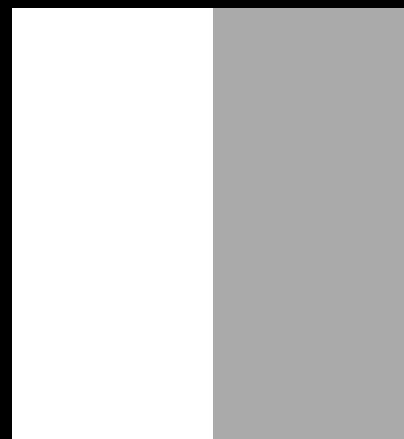
Four rectangle feature



D

$$\text{Feature} = \boxed{\text{Sum of pixel values in image}} - \boxed{\text{Sum of pixel values in image}}$$

# One Haar feature



A

3	42	115	137	1	66
86	154	21	254	198	204
41	67	58	20	208	110
203	167	233	113	222	232
79	176	39	27	22	46
135	191	211	245	102	67

$$\text{Feature} = 254 + 198 + 20 + 208 + 113 + 222 - 154 - 21 - 67 - 58 - 167 - 233 = 1015 - 700 = 315$$

## Four rectangle Haar feature - what is the feature value?

567

179

-611

-113

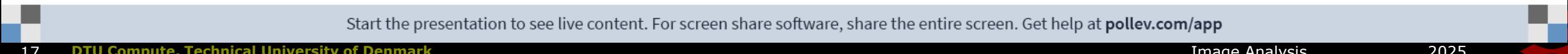
76

I do not know

Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)



Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)



## Four rectangle Haar feature - what is the feature value?

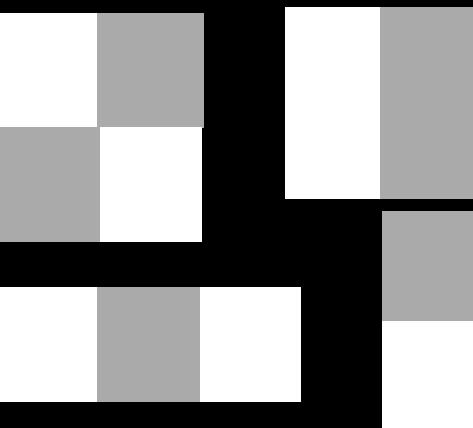


Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)

# Fast computing of Haar features

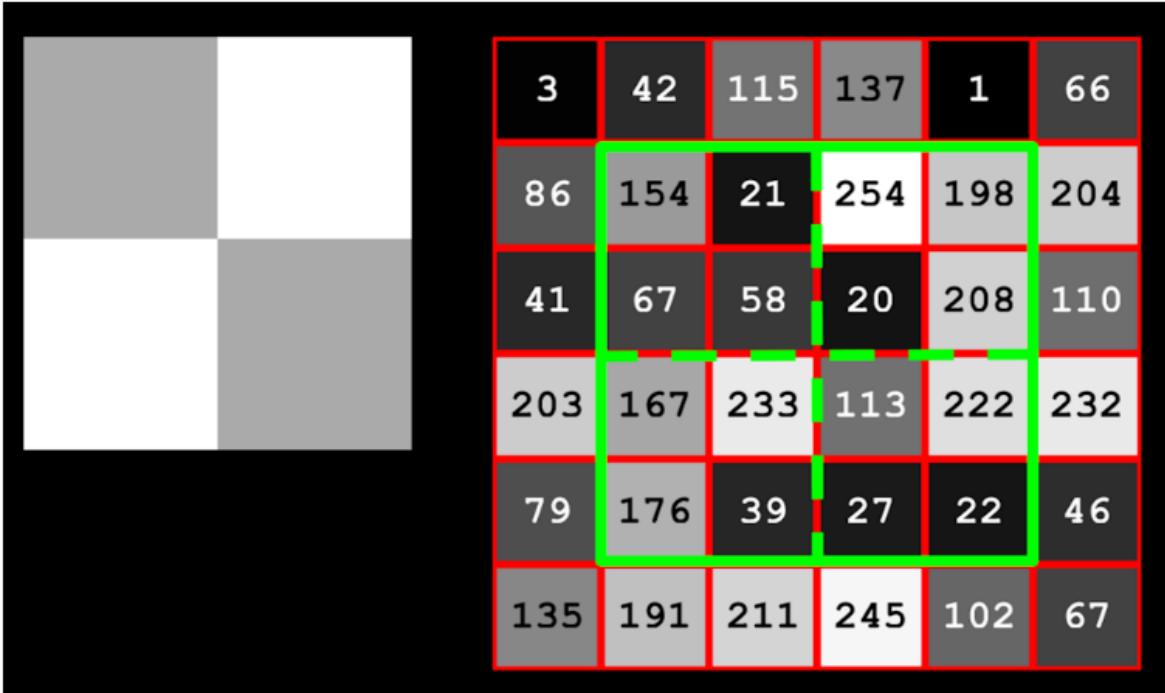


24 x 24 pixels



- Even for small Haar features, there are quite a lot of basic operations
- The larger the Haar feature, the more operations
- We need a fast way to compute Haar features

# How many basic operations (plus and minus) are needed to compute the feature?



15

6

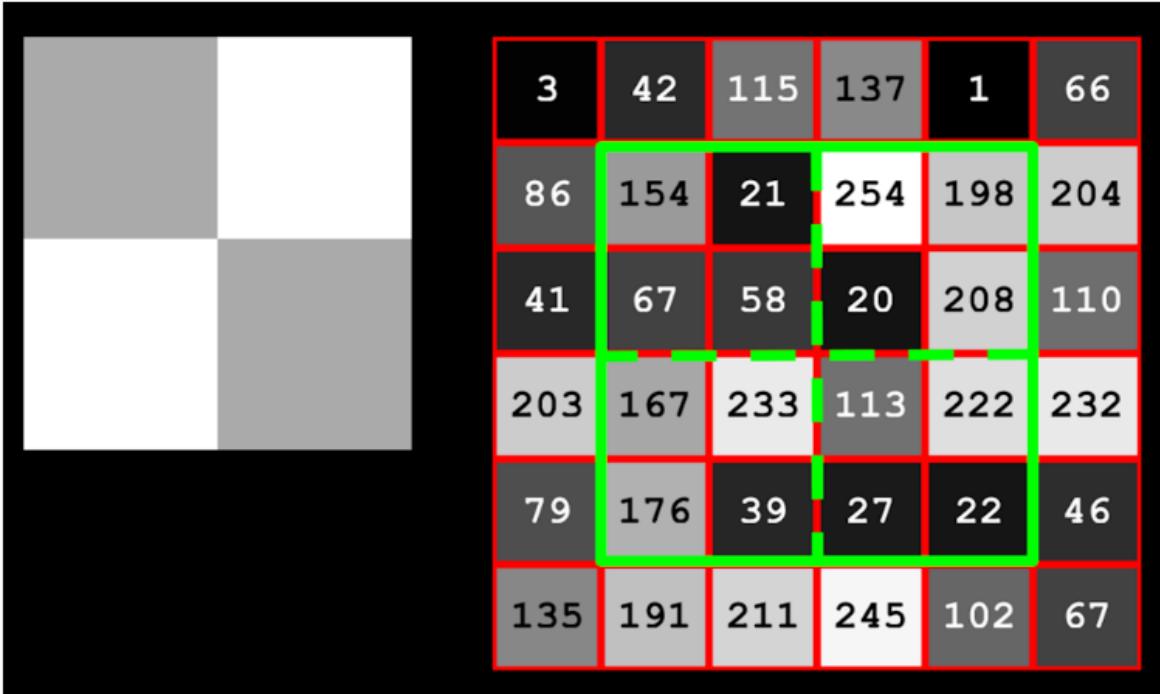
9

21

3

I do not know

## How many basic operations (plus and minus) are needed to compute the feature?



15 ✓



6

9

21

3

93%

0%

0%

7%

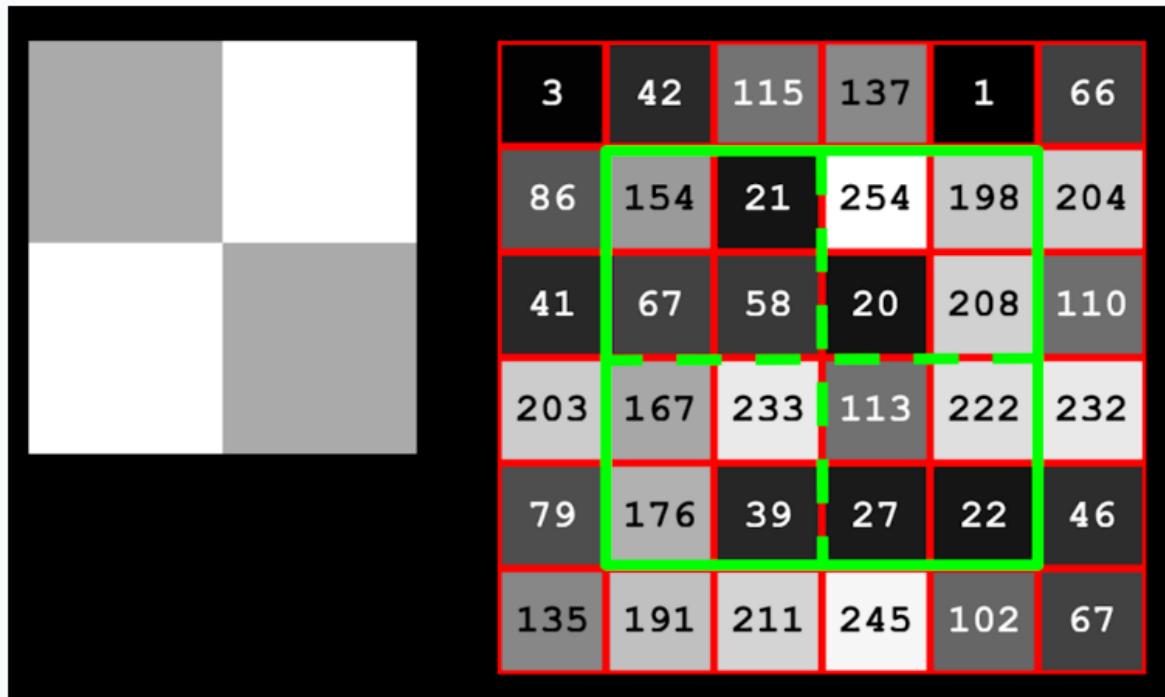
0%

I do not know

0%

Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)

## How many basic operations (plus and minus) are needed to compute the feature?



15 ✓



6

9

21

3

0%

0%

7%

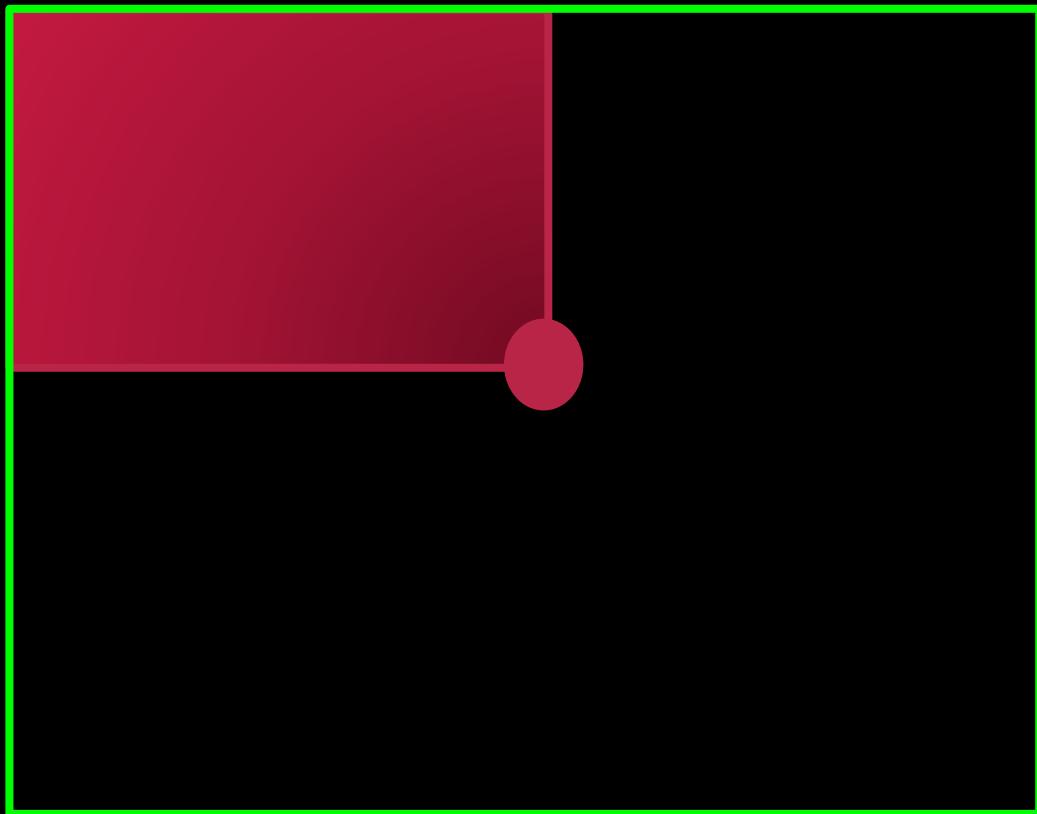
0%

I do not know

0%

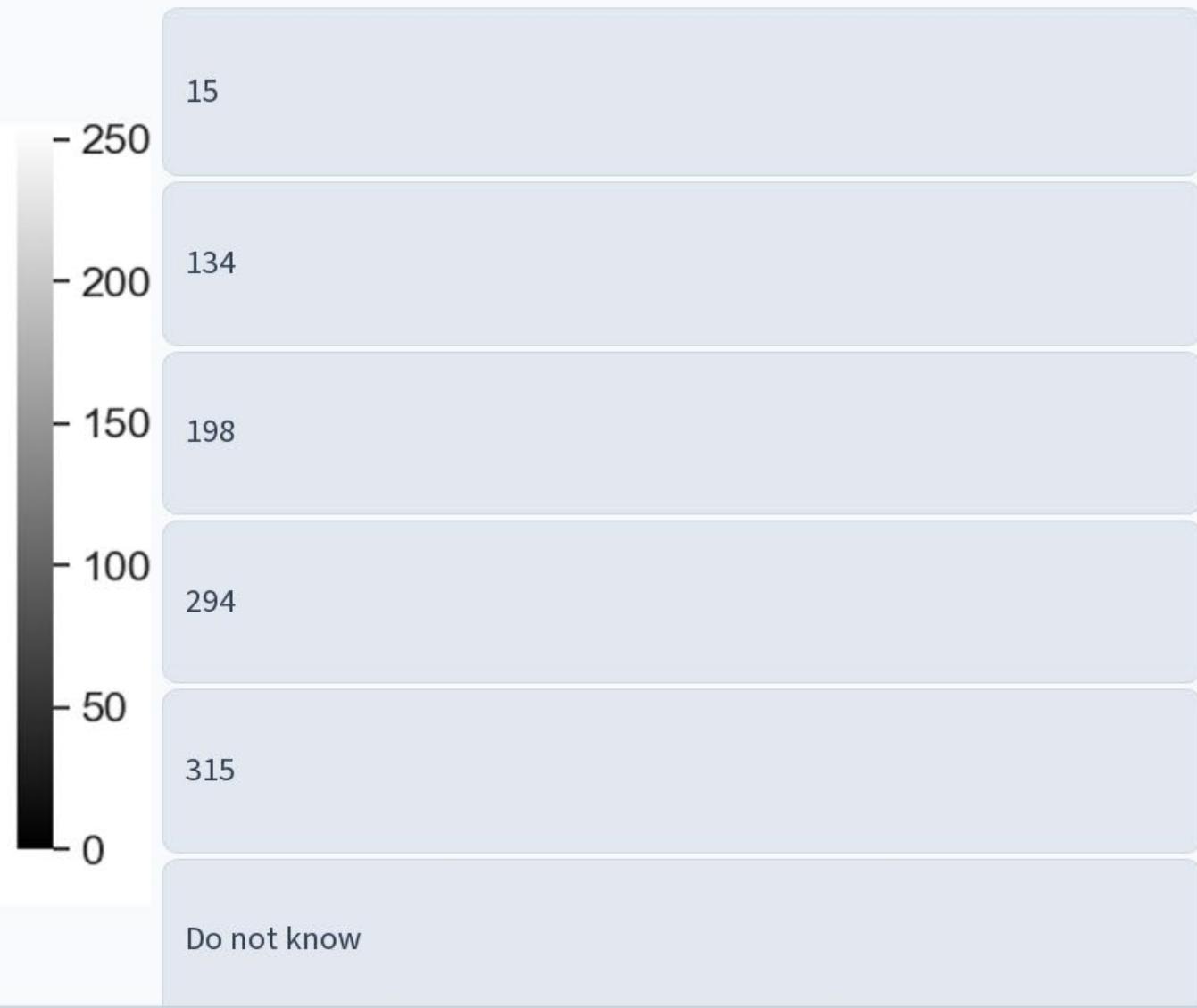
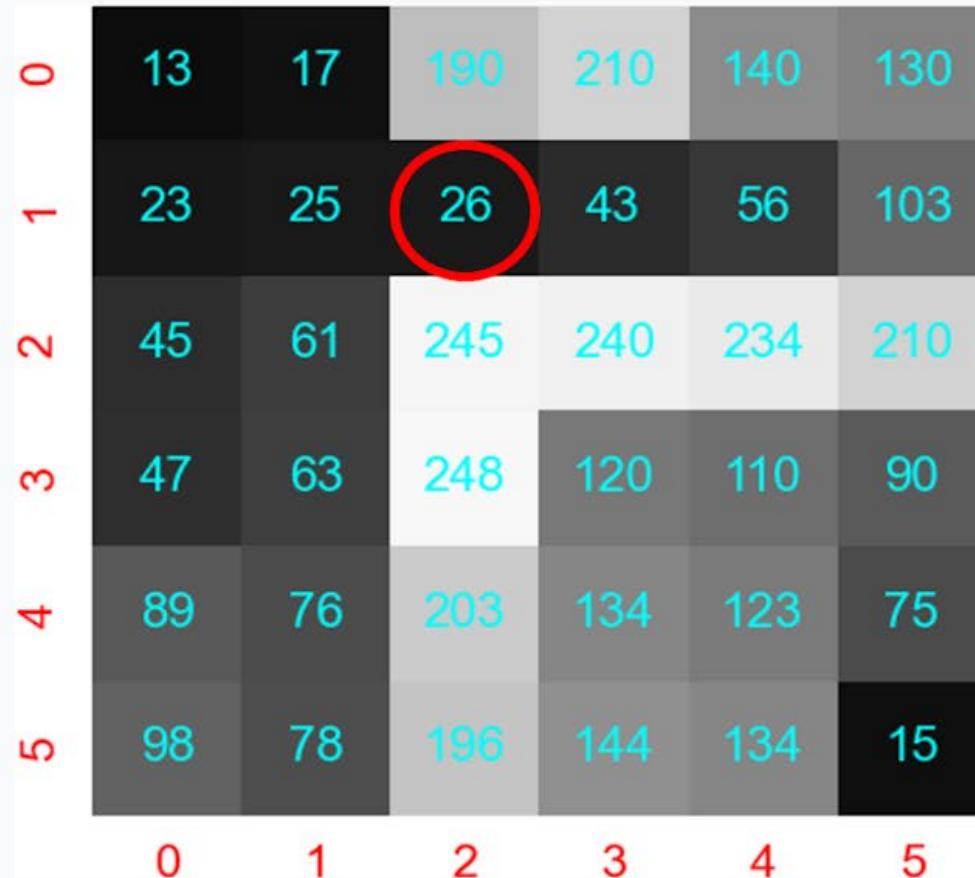
Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)

# Fast computation of Haar features – the integral image



- In an integral image the pixel value is:
  - The sum of pixel above it and to the left of it in the original image
  - Including the pixel itself
  
- Can be computed very fast

## Computing the integral image - what is the value in the marked pixel?



Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)

## Computing the integral image - what is the value in the marked pixel?

15

0%

134

0%

198

0%

294

100%

315

0%

Do not know

0%

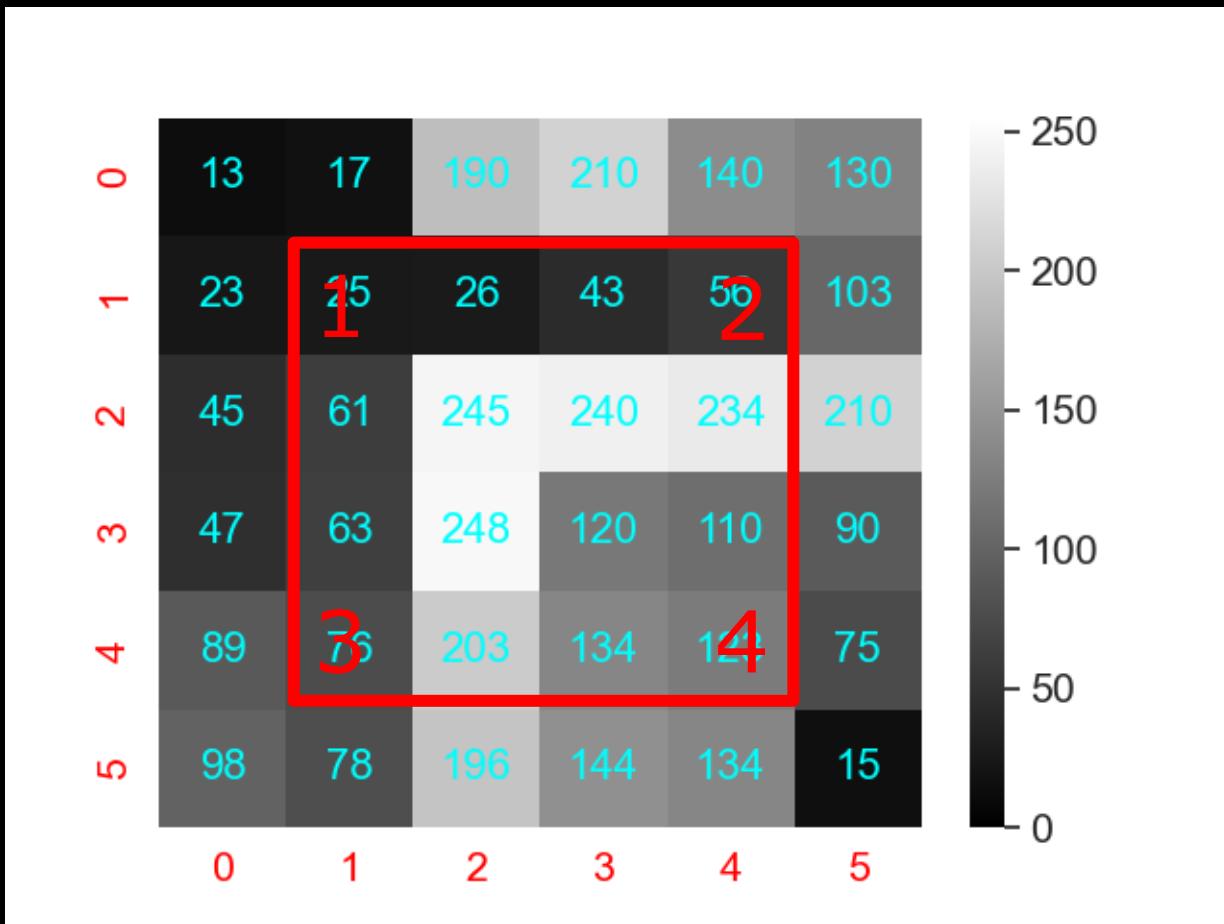
Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)

## Computing the integral image - what is the value in the marked pixel?



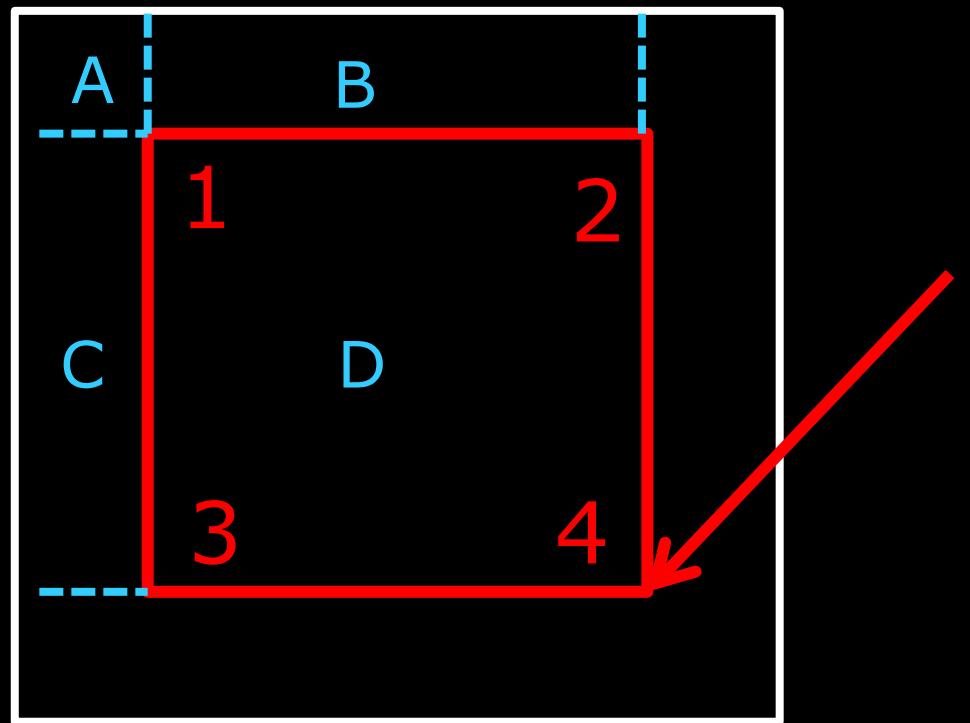
Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)

# Using the integral image



- We want to compute the pixel sum in the rectangle
- Defined by four corners: 1, 2, 3, 4

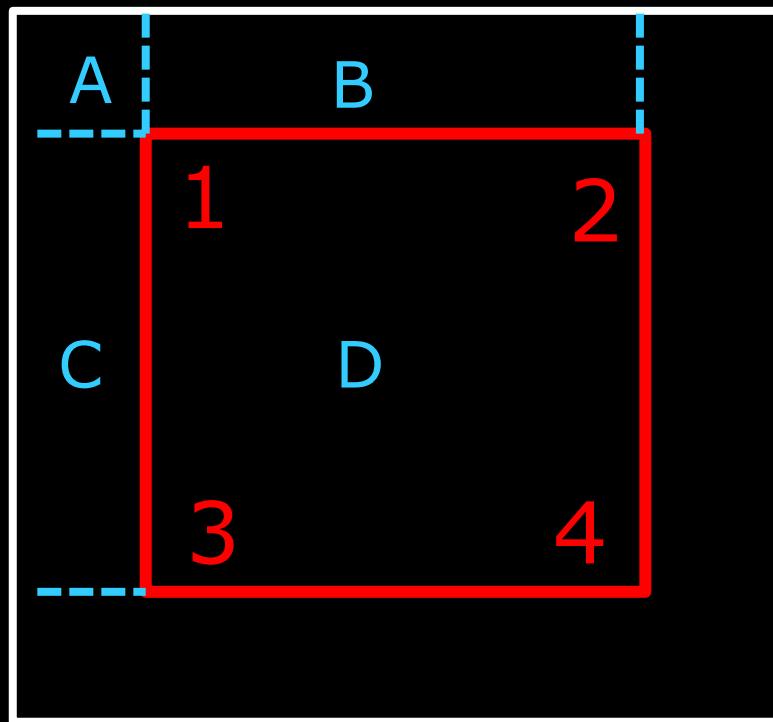
# Using the integral image



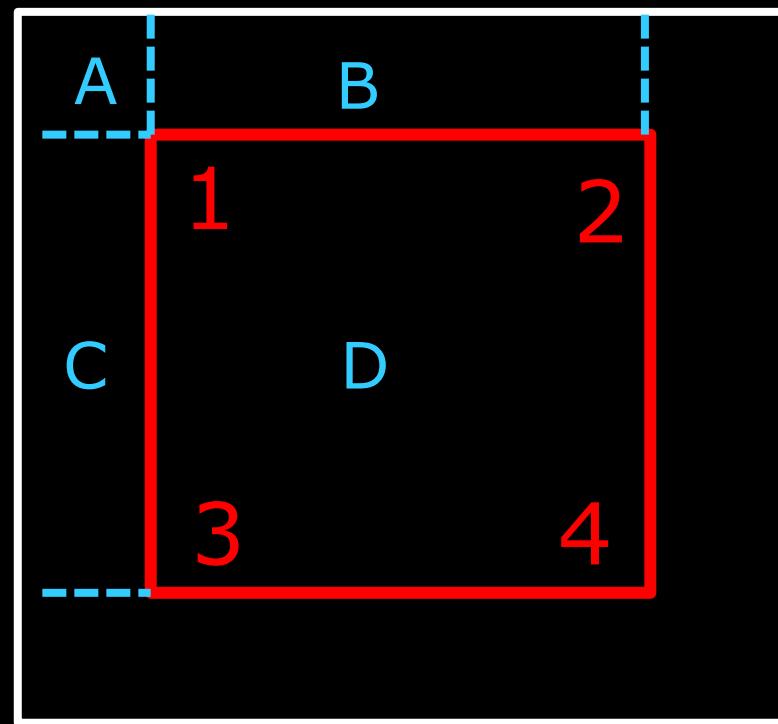
- Define four regions:
  - A, B, C, D
- The sum of pixels in the area
  - $A+B+C+D$  is the value of the integral image at point 4

# Using the integral image

- The sum of pixels in the area
  - $A+B$  is the value of the integral image at point 2
  - $A+C$  is the value of the integral image at point 3



# Using the integral image – short notation



- The sum of pixels in the area
  - $\text{ii}(2) = A+B$
  - $\text{ii}(3) = A+C$
  - $\text{ii}(4) = A+B+C+D$
  - $\text{ii}(1) = A$
  
  - $\text{ii}(4)-\text{ii}(3)-\text{ii}(2) = D - A$
  
- $\text{ii}(4)-\text{ii}(3)-\text{ii}(2)+\text{ii}(1) = D$

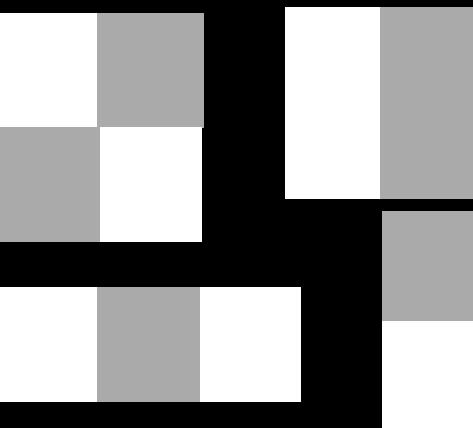
## Course evaluation

- Very important to get your feedback on the course
- Please do it now – log into DTU Inside and fill in the evaluation
- What works well – so we should keep it and strengthen that part
- What can be improved and how?
- The question about "The teacher gave me feedback on my progress"
  - Very hard with large courses
  - We try with quizzes, TAs, exercise solutions

# Haar features in an image window



24 x 24 pixels



- Image window of 24 x 24 pixels
- All possible sizes and shapes of Haar features
- More than 180.000 features according to Viola and Jones
- They are *overcomplete* – meaning there is a very high redundancy
- We need *feature selection*

# Possible features

$$f_1 = \begin{matrix} & \\ & \text{■} \\ & & \text{■} \\ \text{■} & & & \end{matrix}$$

$$f_5 = \begin{matrix} & & \\ & \text{■} & \\ & & \text{■} \\ \text{■} & & & \end{matrix}$$

$$f_2 = \begin{matrix} & \\ & \text{■} \\ \text{■} & & \end{matrix}$$

$$f_6 = \begin{matrix} & \\ & \text{■} \\ \text{■} & & \end{matrix}$$

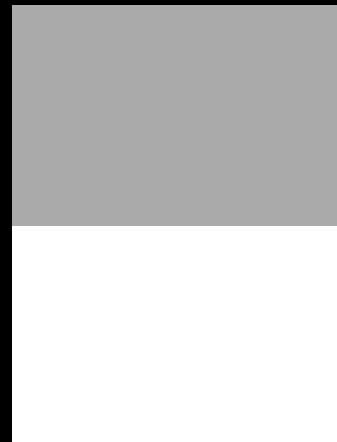
$$f_3 = \begin{matrix} & \\ & \text{■} \\ & & \text{■} \\ \text{■} & & & \end{matrix}$$

$$f_7 = \begin{matrix} & \\ & \text{■} \\ & & \text{■} \\ & & & \text{■} \\ \text{■} & & & & \end{matrix}$$

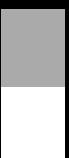
$$f_4 = \begin{matrix} & \\ & \text{■} \\ \text{■} & & \end{matrix}$$

$$f_8 = \begin{matrix} & \\ & \text{■} \\ & & \text{■} \\ & & & \text{■} \\ & & & & \text{■} \\ \text{■} & & & & & \end{matrix}$$

...  $f_{180000} =$



# Feature selection – from the article



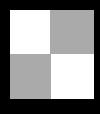
- There are over 180,000 rectangle features associated with each image sub-window, a number far larger than the number of pixels.
- Even though each feature can be computed very efficiently, computing the complete set is prohibitively expensive.
- Our hypothesis, which is borne out by experiment, is that a very small number of these features can be combined to form an effective classifier.
- The main challenge is to find these features

# Learning Classification Functions

## Weak classifier

 $x =$ 

24 x 24 sub-  
window

 $f_j =$  

Feature value computed on the sub-window

 $p_j \in [-1, 1]$ 

Parity – determines if the feature value should be positive or negative

 $\theta_j$ 

Feature threshold

## Weak classifier

$$h_j(x) = \begin{cases} 1 & \text{if } p_j f_j(x) < p_j \theta_j \\ 0 & \text{otherwise} \end{cases}$$

$$x = \boxed{\text{Image of a person's arm}} \quad f_j(\boxed{\text{Image of a person's arm}}) = \boxed{\text{Image of a person's arm with a 3x3 grid overlay}} = 2049$$

Learnt by training:  $p_j = 1$      $\theta_j = 456$

$$\rightarrow 1 * 2049 < 1 * 456 \rightarrow h_j(\boxed{\text{Image of a person's arm}}) = 0$$



# What is this parity?

## Weak classifier

$$h_j(x) = \begin{cases} 1 & \text{if } p_j f_j(x) < p_j \theta_j \\ 0 & \text{otherwise} \end{cases}$$

$$x = \boxed{\text{Image of a person's leg}} \quad f_j(\boxed{\text{Image of a person's leg}}) = \boxed{\text{Image of a person's leg with a 3x3 gray mask}} = 2049$$

Learnt by training:  $p_j = -1 \quad \theta_j = 456$

$$\rightarrow -1 * 2049 < -1 * 456 \rightarrow h_j(\boxed{\text{Image of a person's leg}}) = 1$$

# Creating a strong classifier from weak classifiers

$$h(x) = \begin{cases} 1 & \sum_{t=1}^T \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^T \alpha_t \\ 0 & \text{otherwise} \end{cases}$$

$$h_1(\boxed{\text{ }}) = \boxed{\text{ }} \quad \text{Image: A person's arm with a red flag in the background}$$

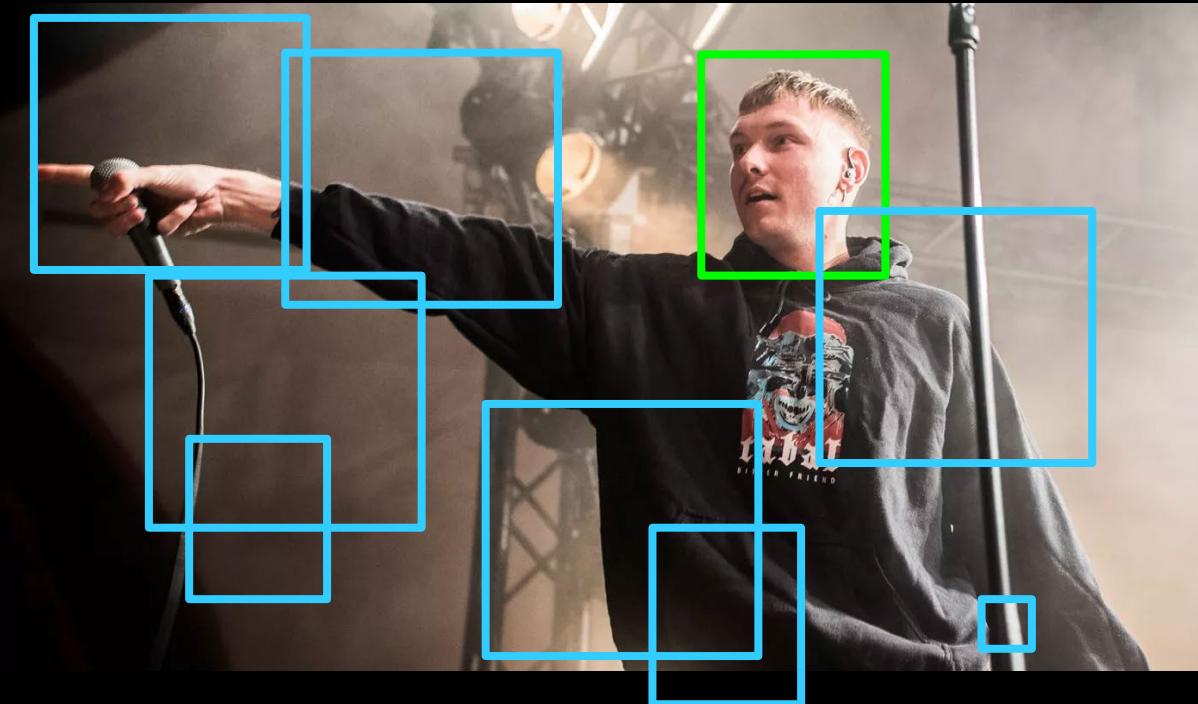
$$h_2(\boxed{\text{ }}) = \boxed{\text{ }} \quad \text{Image: A person's arm with a red flag in the background}$$

...

$$h(\boxed{\text{ }}) = \alpha_1 h_1 + \alpha_2 h_2 + \cdots + \alpha_T h_T$$

Learnt using AdaBoost

# Boosted features – good performance but not enough



- Frontal face classifier with
  - T=200 features
  - Detection rate 95%
  - False positives 1 in 14084
  - 0.7 seconds for a 384 x 288

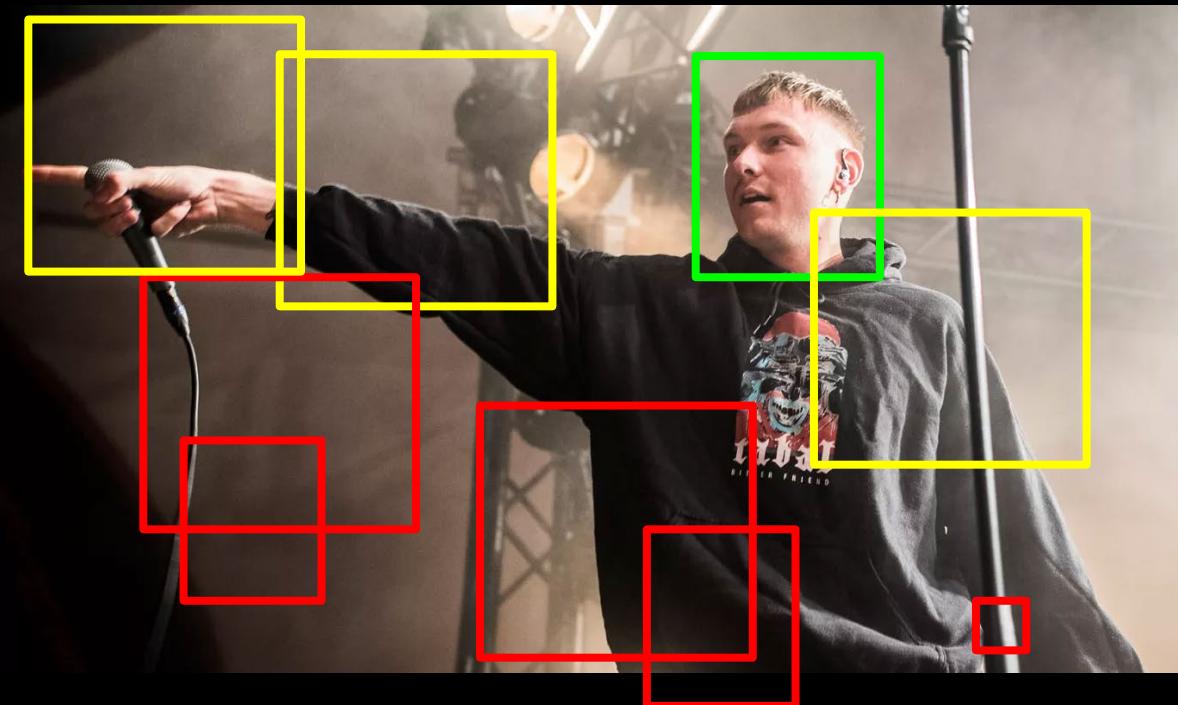
$$h_1(\boxed{\phantom{0}}) = \alpha_1 h_1 + \alpha_2 h_2 + \dots + \alpha_T h_T$$

# The Attentional Cascade



# Image Attention

- The process of focusing on specific parts of an image
  - Followed by fine grained analysis of selected windows



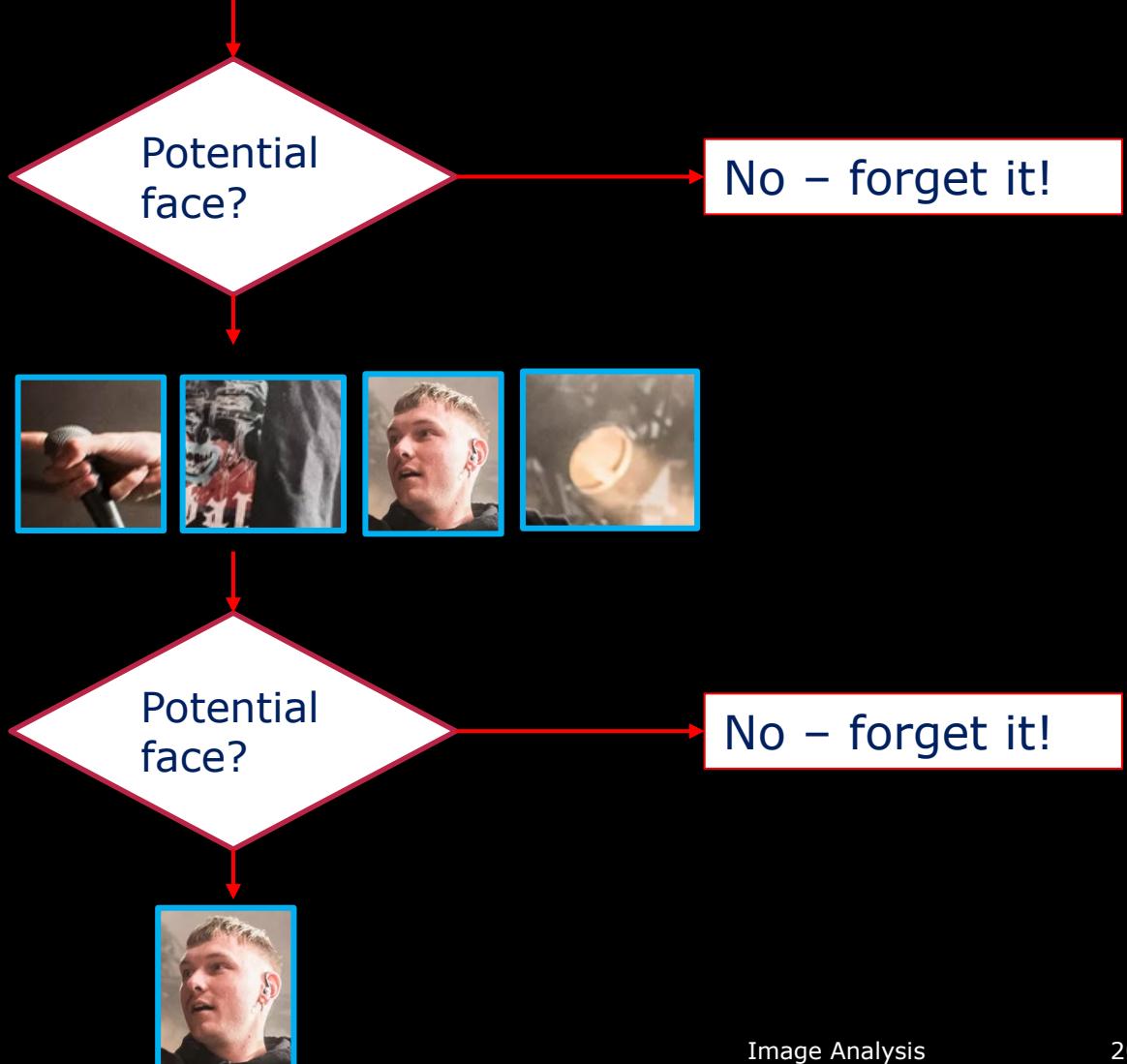
Focusing on potential face regions

# Cascaded classifier



Also called a *degenerate decision tree*

Input image windows



## What is a false negative?

A face window classified as face window

A background window classified as a face window

A face window classified as a background window

A background window classified as a background window

I do not know

Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)

## What is a false negative?

A face window classified as face window

0%

A background window classified as a face window

10%

A face window classified as a background window 

90%

A background window classified as a background window

0%

I do not know

0%

Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)

## What is a false negative?

A face window classified as face window 0%

A background window classified as a face window 10%

A face window classified as a background window ✓ 90%

A background window classified as a background window 0%

I do not know 0%

Start the presentation to see live content. For screen share software, share the entire screen. Get help at [pollev.com/app](https://pollev.com/app)

# The attentional cascade



- Quickly reject negative sub-windows
  - Detect almost all positive sub-windows
  - False-negatives close to zero
    - Keep all potential face windows
  - Using the training set to find weights that fulfils this criterion

- Later more complex classifier
  - Low false positive rate



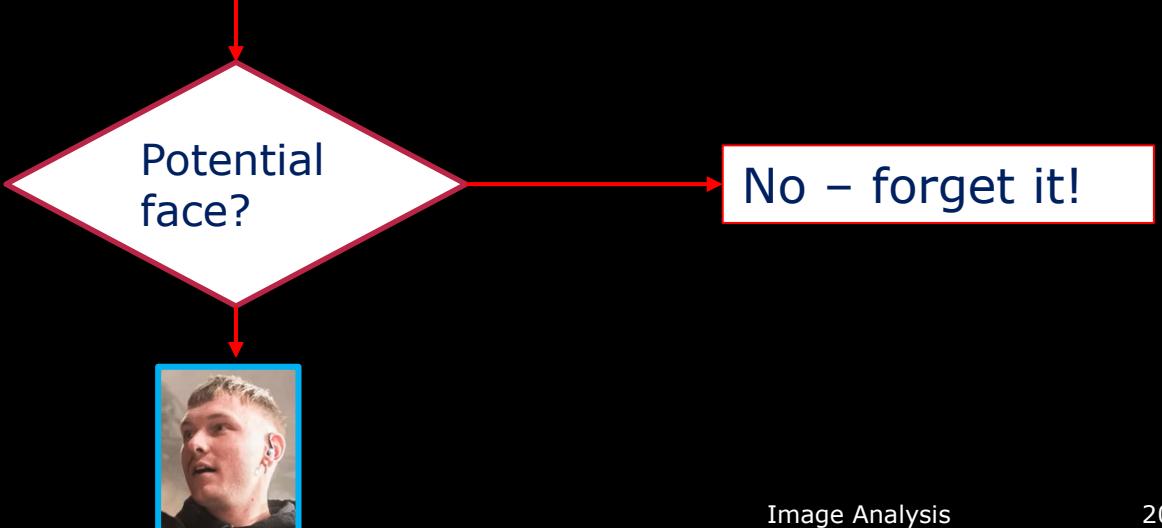
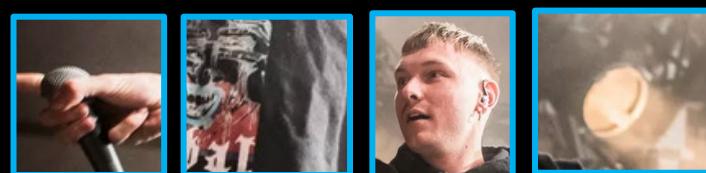
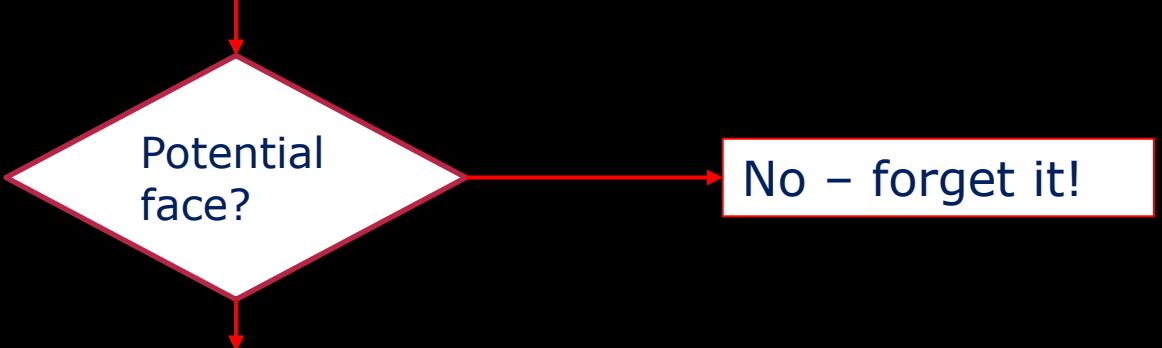
# Training a cascade

$$h(\boxed{\text{ }}) = \alpha_1 h_1 + \alpha_2 h_2 + \dots + \alpha_T h_T$$

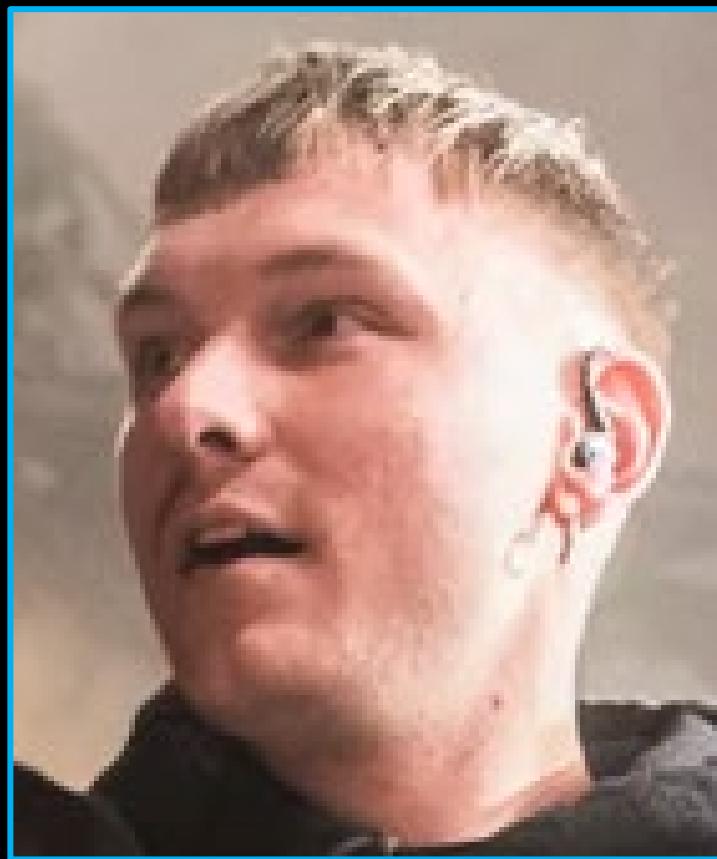
Learnt using AdaBoost

$$h(\boxed{\text{ }}) = \alpha_1 h_1 + \alpha_2 h_2 + \dots + \alpha_T h_T$$

Learnt using AdaBoost



# First stage classifier

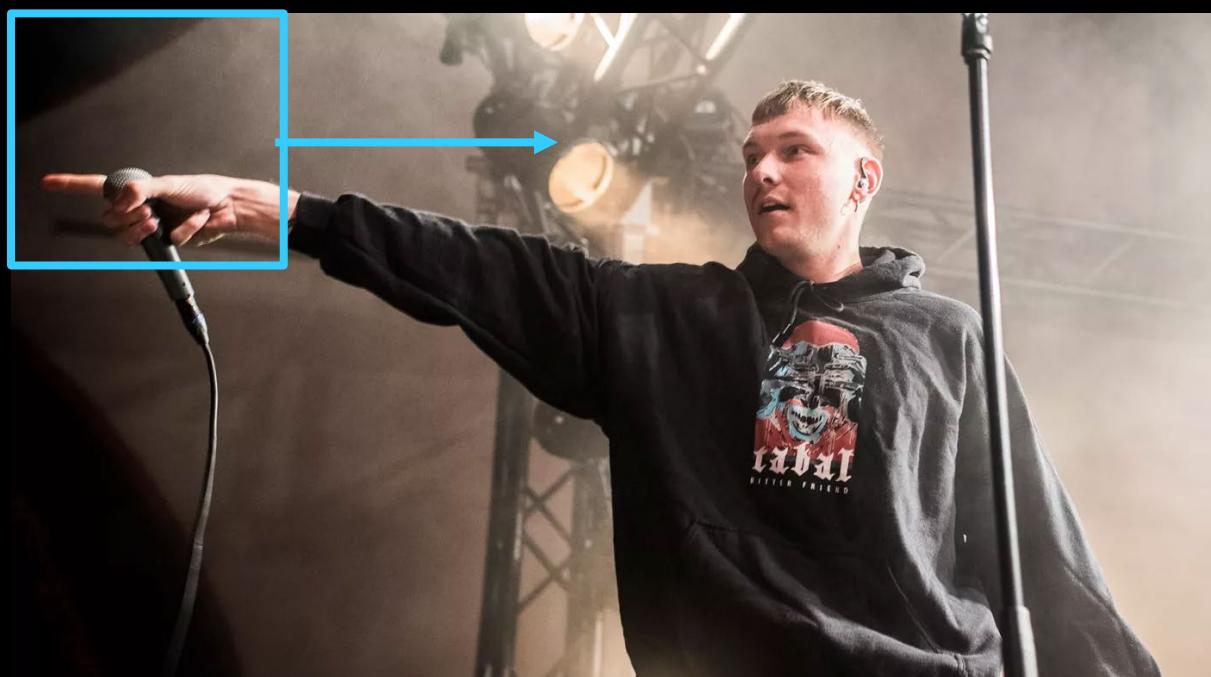


## Final classifier



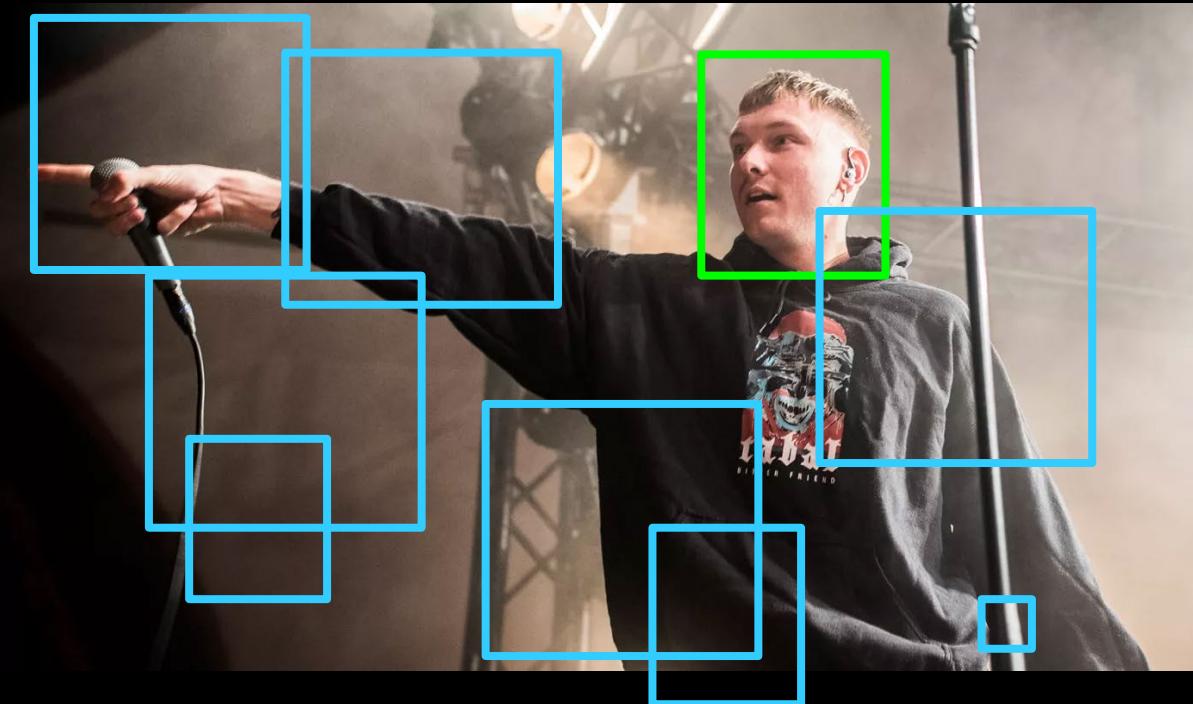
- 38 stages (step in the cascade)
- Total 6000 features (over the entire cascade)
- Faces are detected using on average 10 features per sub-window

# Finding all faces in an image



- Slide a sub-window over the entire image
- Do a face detection for all positions
- Scale the features in a certain interval
  - To find faces of different sizes

# Conclusion



- One of the most important algorithms before deep learning
- Uses many interesting concepts
  - Attention
  - Boosted weak classifiers
  - Very fast feature computation



# Demo

# Next week(s)

- Statistical models of shape and appearance



# Image Analysis

Rasmus R. Paulsen

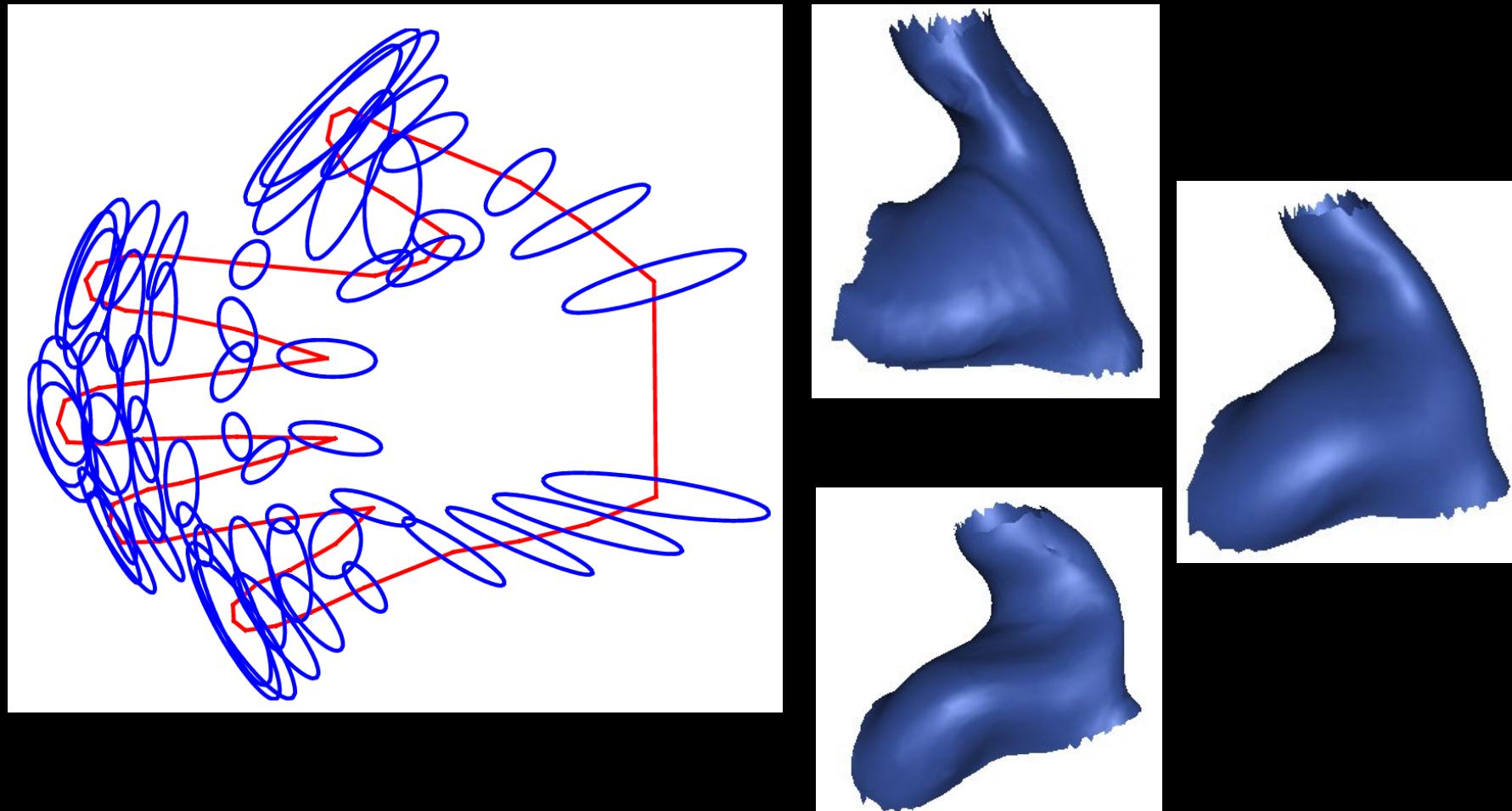
Tim B. Dyrby

DTU Compute

[rapa@dtu.dk](mailto:rapa@dtu.dk)

<http://courses.compute.dtu.dk/02502>

# Lecture 12 – Statistical models of shape and appearance



# Today's Learning Objectives

- Describe the concept of shape models
- Define the shape of an object using landmarks
- Describe point correspondence
- Describe and use the vector representation of a shape
- Describe how a shape can be seen as a point in high-dimensional space
- Explain how shapes can be aligned
- Describe how principal component analysis can be used to model shape variation
- Explain the similarity between Eigenfaces and shape and appearance models

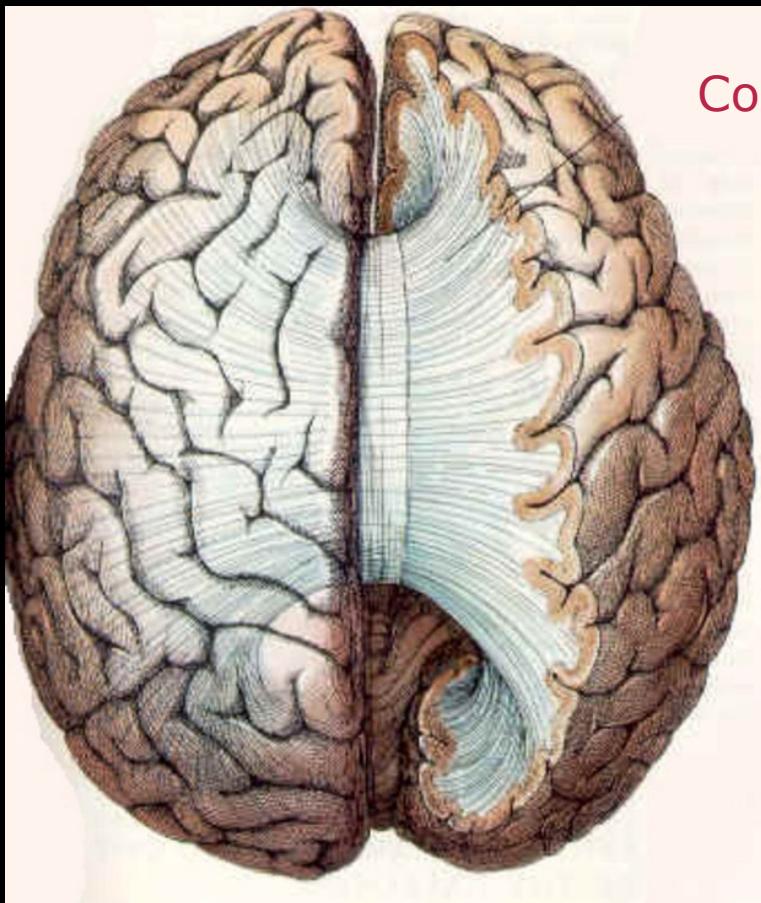
# A typical scenario



- Doctor X believes that he can “see” on a hand X-ray if the patient is in risk of arthritis!
- Specifically Doctor X is sure that the *shape of the joints* is an estimator for arthritis!

Can we verify that?

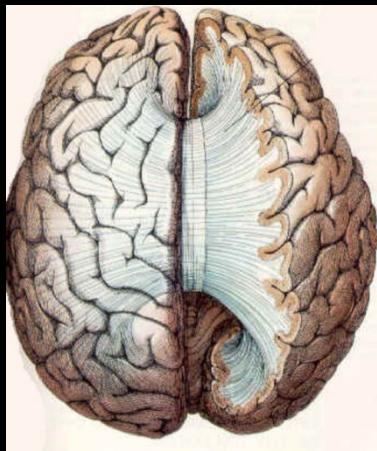
## Scenario II



Corpus Callosum

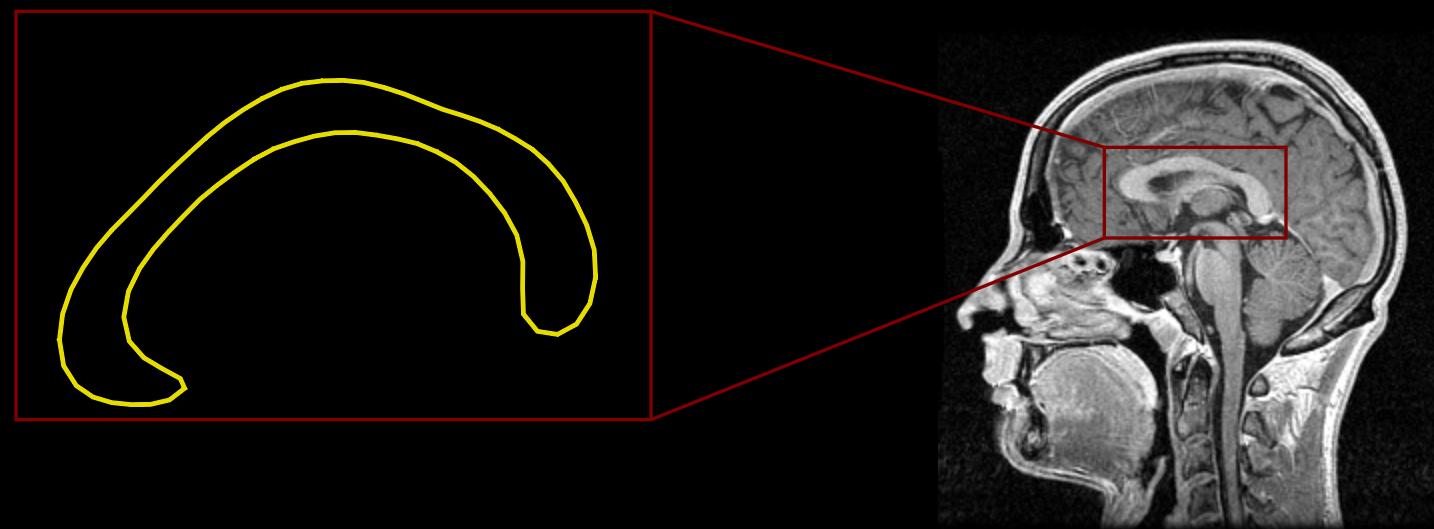
- MR images have been captured of a large group of people
- Cognitive abilities measured as well
- Is there a correlation between *how the brain looks* and how we behave?
- Does the shape of corpus callosum tell us something?

## Scenario II



Corpus Callosum

- We can get the MR slice with the corpus callosum from all the patients



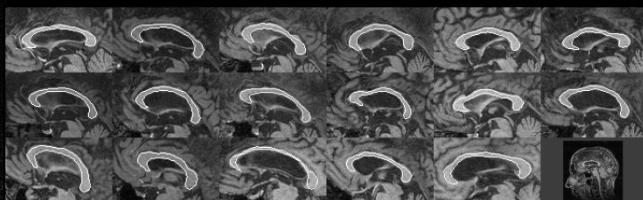
## Scenario III



- An experienced hearing aid fitter has seen a lot of ears!
- Some hearing aid users are very difficult to fit. Why?
- Large variation in the shape of ears
- Ear canals change shape when people chew
- Is it possible to learn about the shape and use it?



# Shape Analysis



600 MR scans and behavioural data

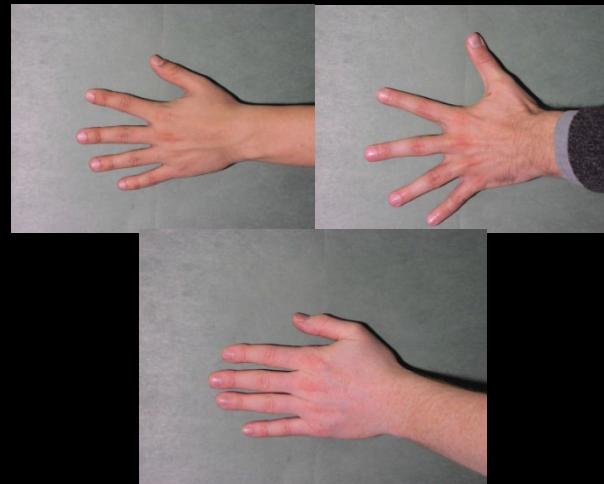
- What can we learn from shape?
- What can we use it for?
- How do we do it?



1000 historical X-rays

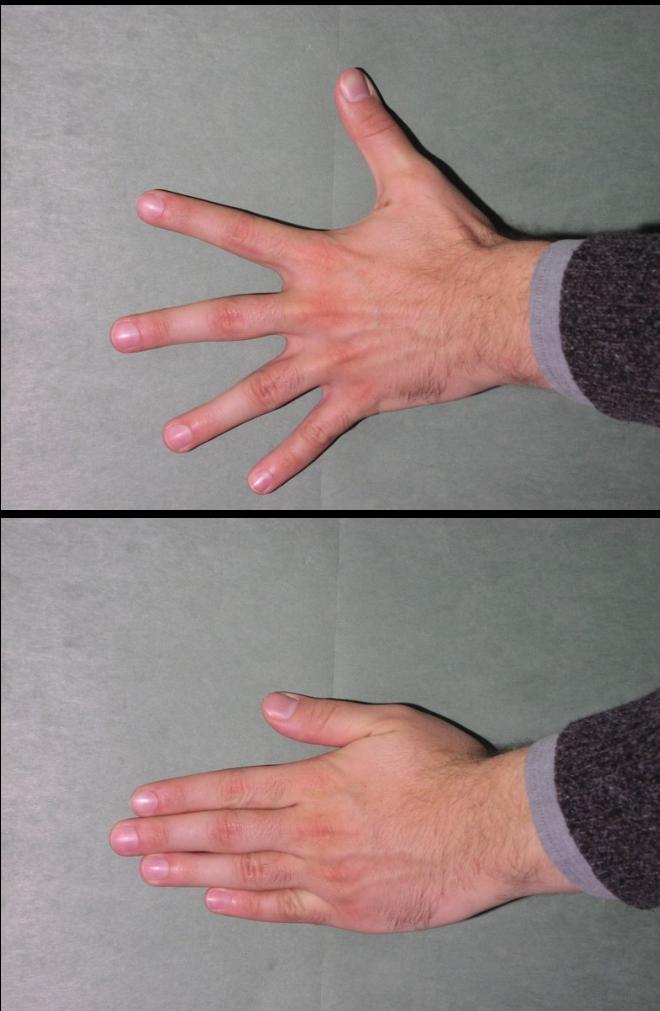


A boxful of something that look like ear canals



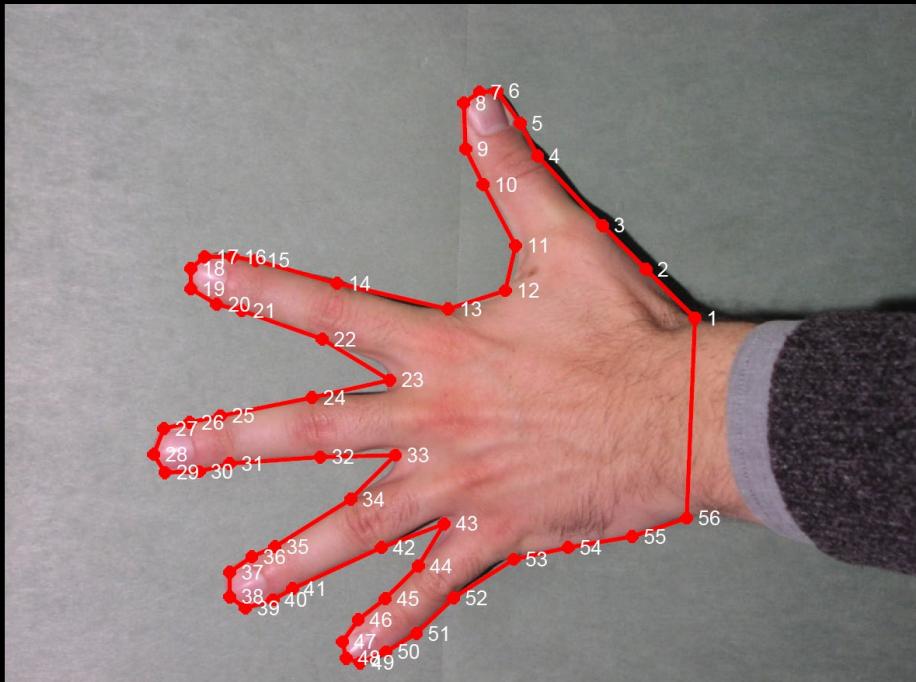
A set of hand photographs

# What is shape?



- How do we define the *shape* of this hand?
- What is the shape difference between the two hands?

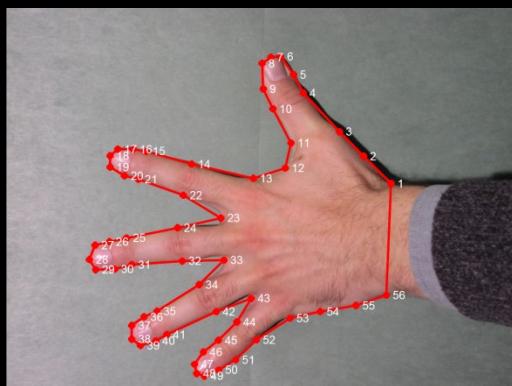
# Shape definition



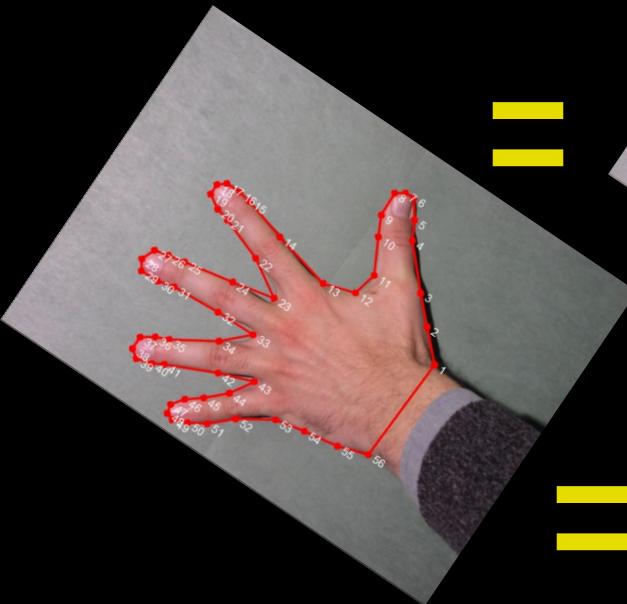
- Shape is defined using landmarks
  - Placed by an expert
- In this case the outer contour of the hand
- Just one of many ways!

# Shape definition

Shape is all geometrical information that remains when location, scale, and rotational effects are removed



=



=



=

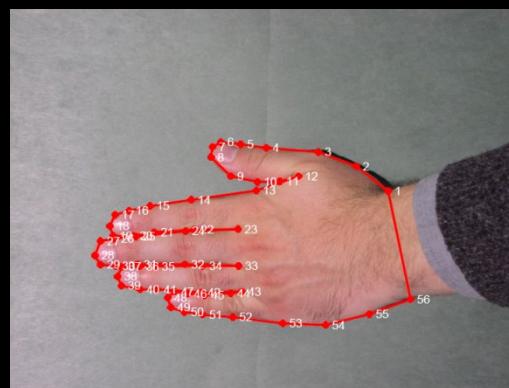


# Shape definition

Shape is all geometrical information that remains when location, scale, and rotational effects are removed

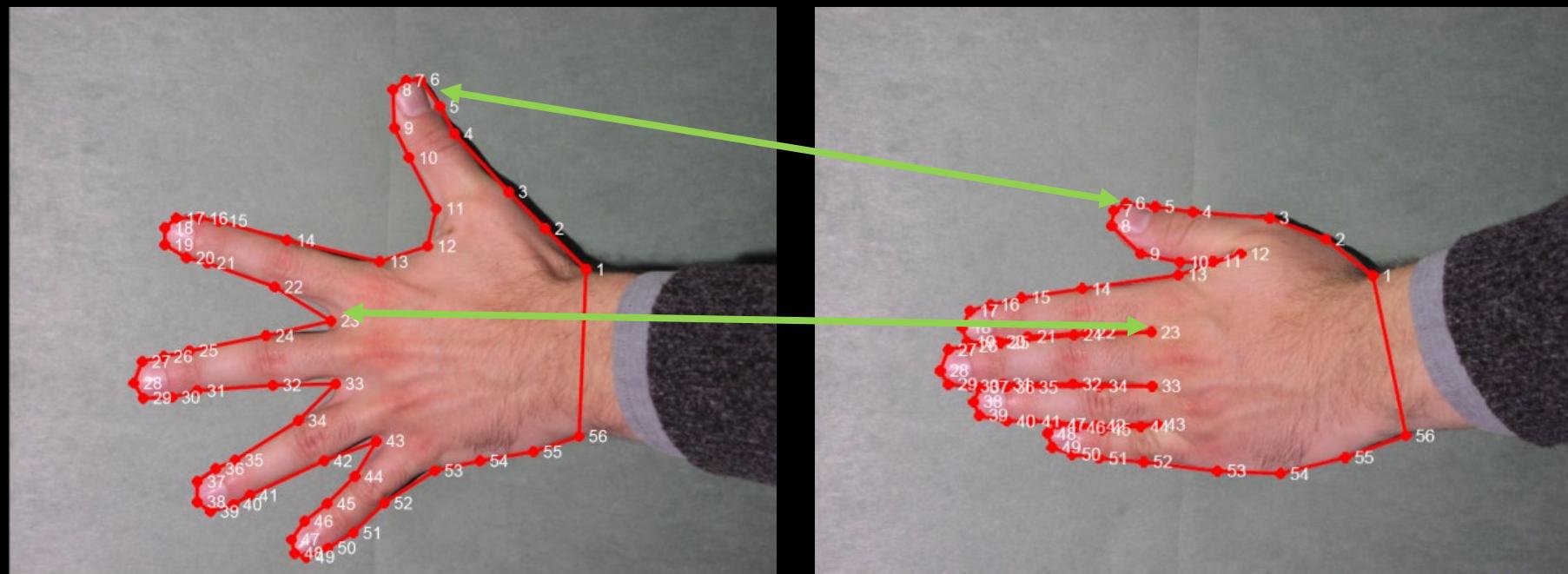


$\neq$



# Landmarks and point correspondence

Landmarks are placed on the same place on all shapes in the training set



# Shape as a vector

1 :  $(x_1, y_1)$

2 :  $(x_2, y_2)$

:  
:

$N$  :  $(x_n, y_n)$



- The shape is represented as an array of  $(x, y)$  coordinates
- Trick number one!  
All coordinates are put into one vector!
- $n=56$  points
  - Vector with 112 elements!

$$\mathbf{x} = [x_1, x_2, \dots, x_n, y_1, y_2, \dots, y_n]^T$$

# Shapes in high-dimensional space

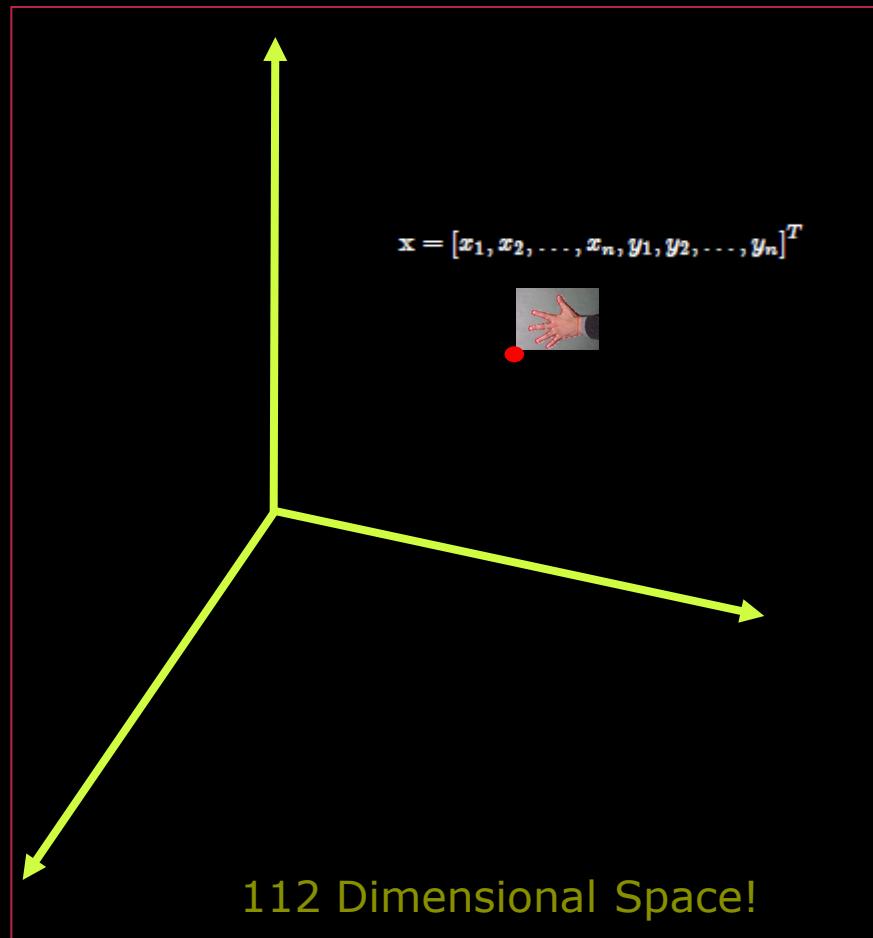


$$\mathbf{x} = [x_1, x_2, \dots, x_n, y_1, y_2, \dots, y_n]^T$$

- One hand is now described using one vector
- A vector can also be seen as a point in space!

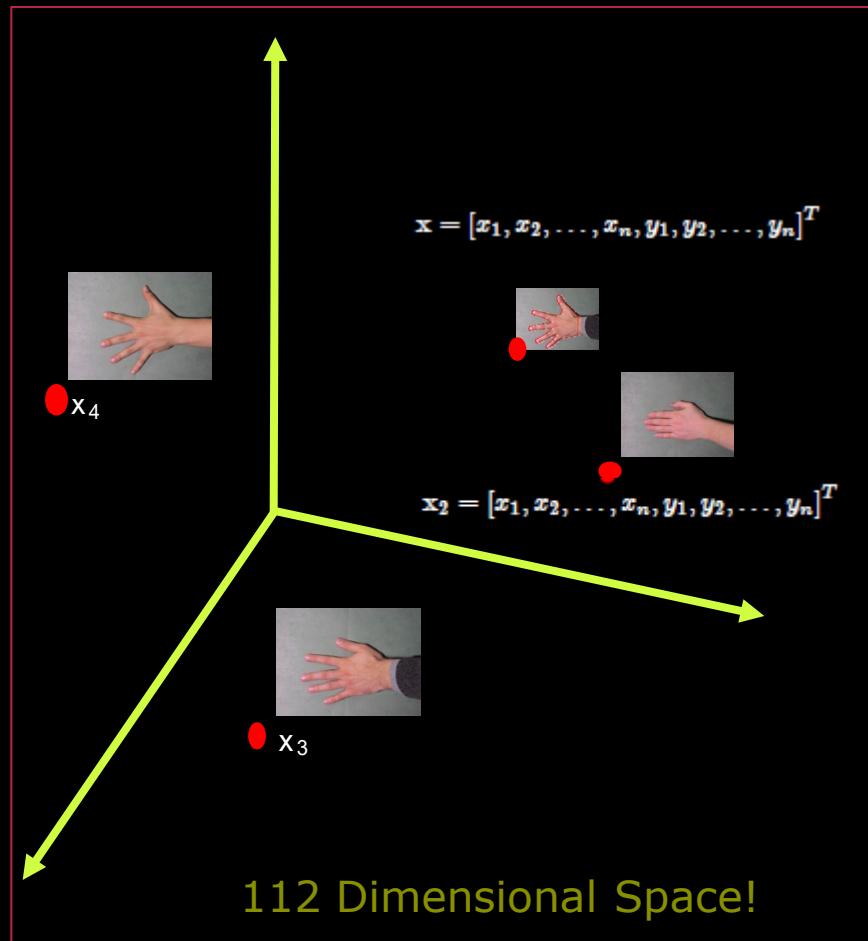
Trick number  
two!

# Coordinates in space



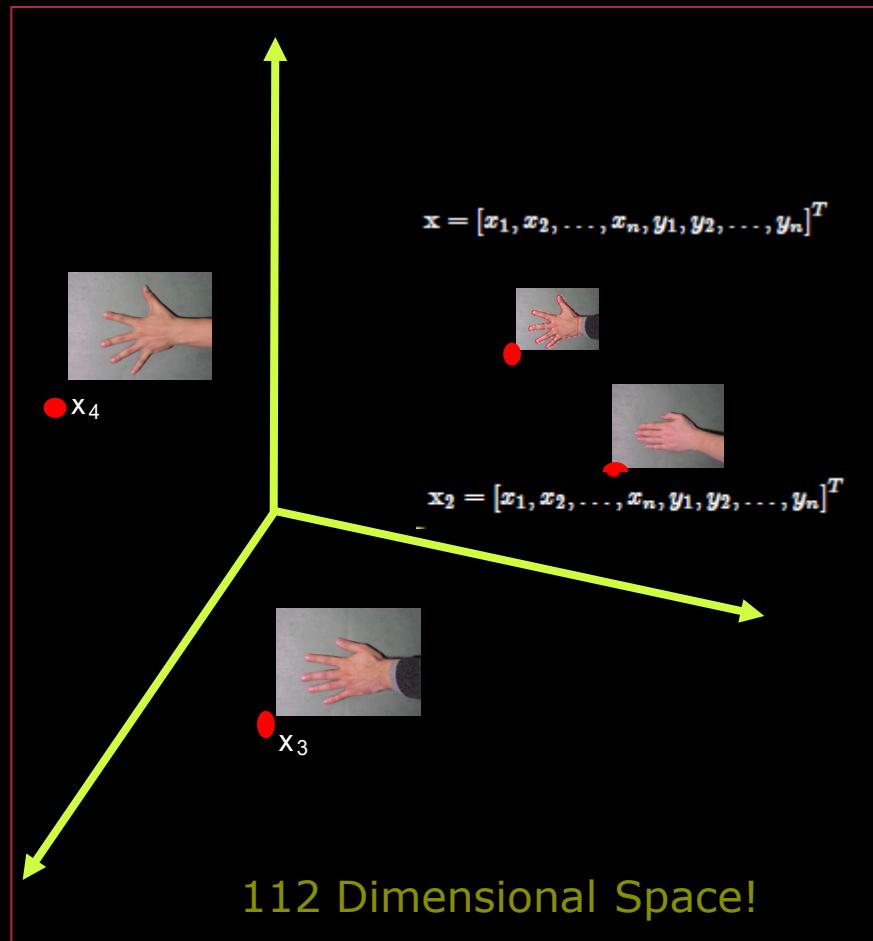
- On hand is now described using one vector
- A vector can also be seen as a coordinate in space!
- Not 2D space, not 3D space, not 4D space...
- 112 Dimensional Space!
- A hand has a position in this space!

# Hands in Space



- A hand has a position in space!
- Another hand appears
  - in the same space
  - different position = different shape
- All hands have a place in this space!

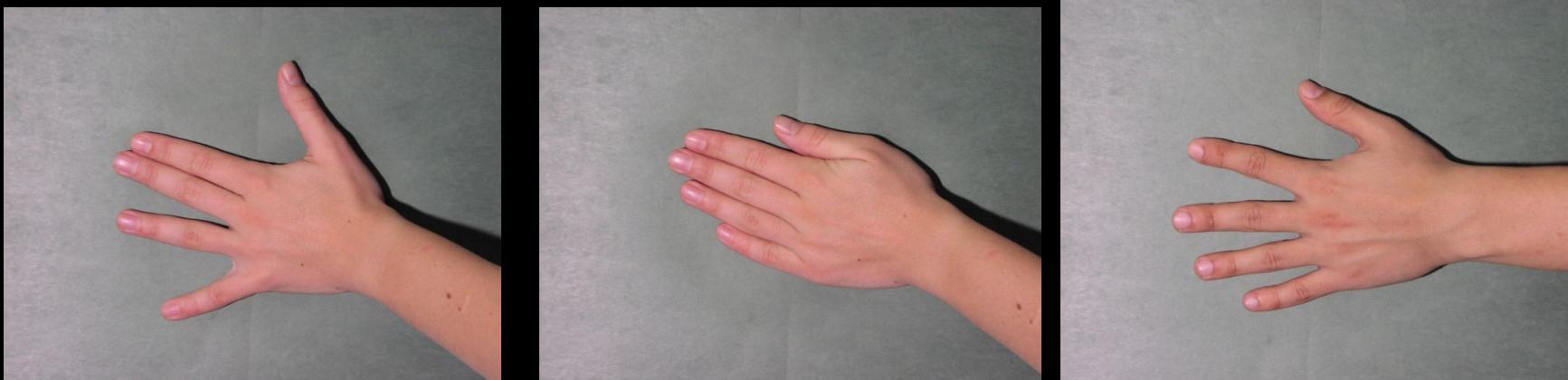
# Shape Analysis



## ■ Shape analysis

- Similar shapes are placed on “planes” in the shape-space
- Also called a manifold

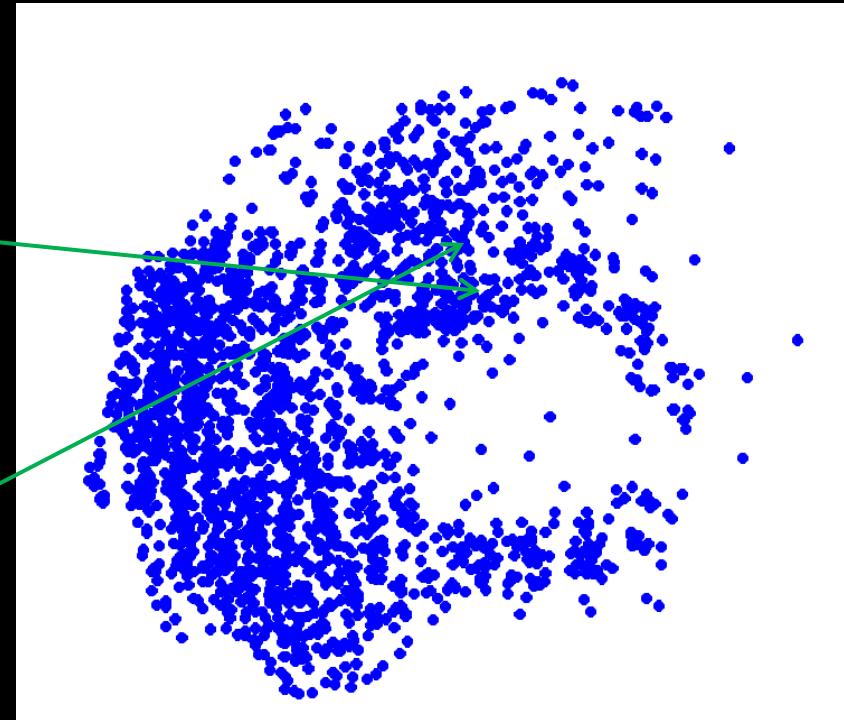
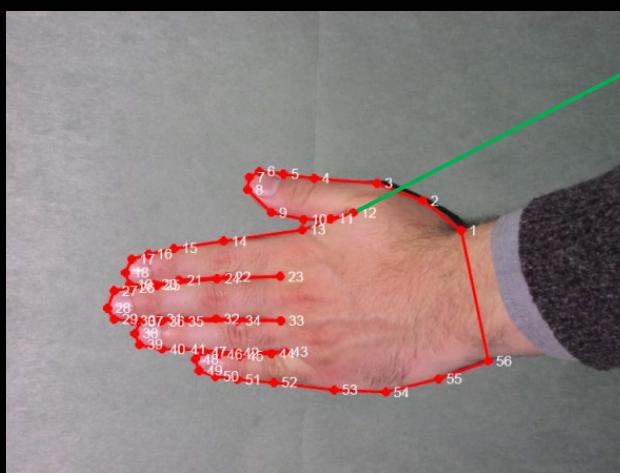
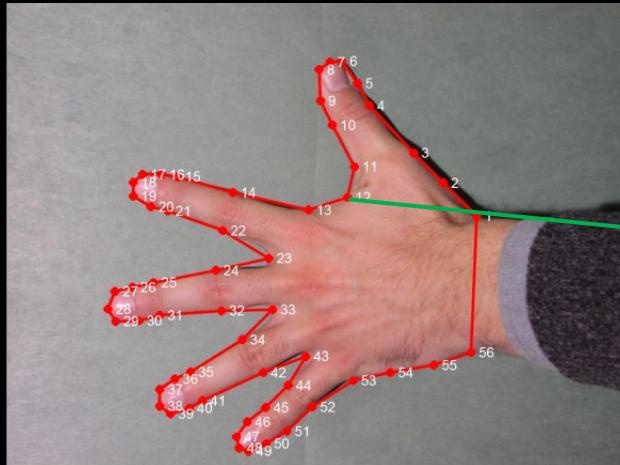
## Shape alignment



- 40 training images of hands
- 56 landmarks on each
- Placed in random location (translation+rotation)

# Shape alignment

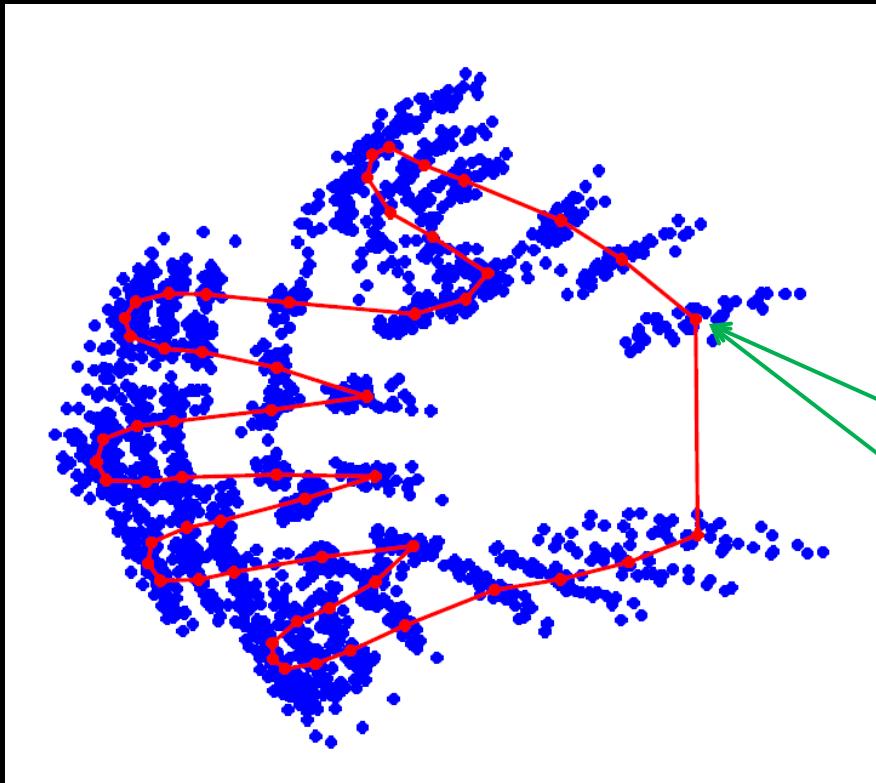
Before alignment



Landmarks from all hands

Needs alignment!

# What is alignment?



Average shape

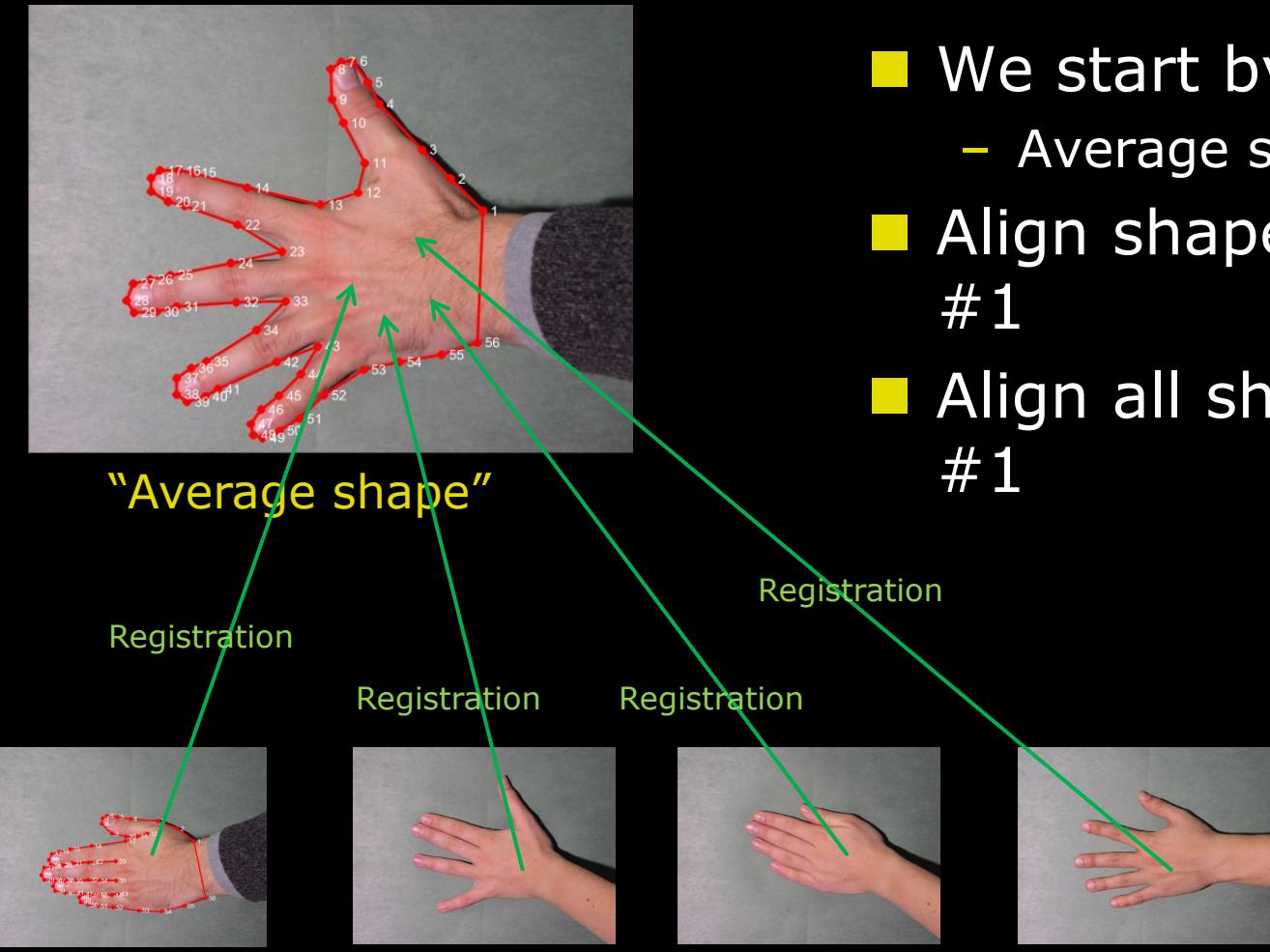
- Group wise registration
  - Not one-to-one
  - All to the average shape

$$\bar{\mathbf{x}} = \frac{1}{N} \sum_{i=1}^N \mathbf{x}_i$$

$$\bar{\mathbf{x}} = [\bar{x}_1, \bar{x}_2, \dots, \bar{x}_n, \bar{y}_1, \bar{y}_2, \dots, \bar{y}_n]^T$$

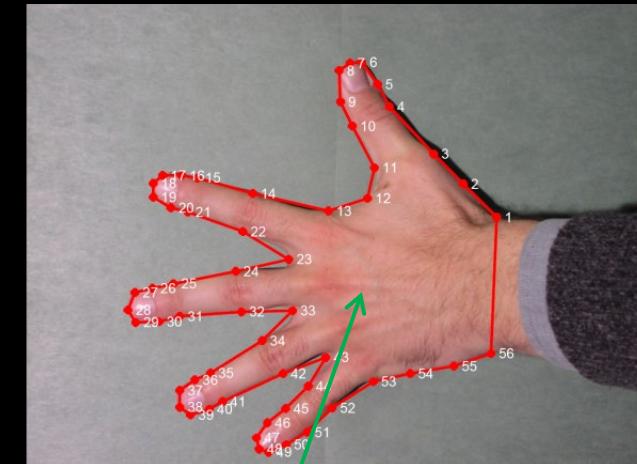
But hey! We do not have an average shape?

# Procrustes Analysis (alignment)



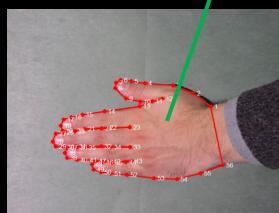
- We start by defining
  - Average shape = Shape #1
- Align shape #2 to shape #1
- Align all shapes to shape #1

# Landmark based registration



“Average shape”

Registration

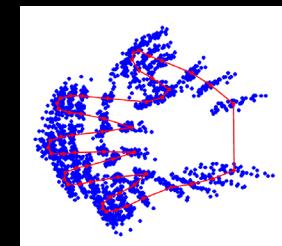
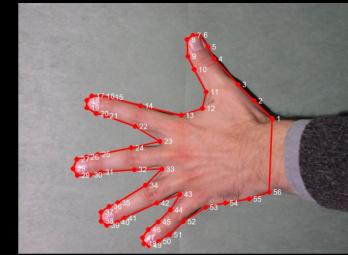


Shape #2

- Shape #2 is transformed to fit the average shape
  - Translation
  - Rotation
  - Scaling
  - = Similarity Transform
- Result
  - Shape #2 is placed *on top of* the average shape

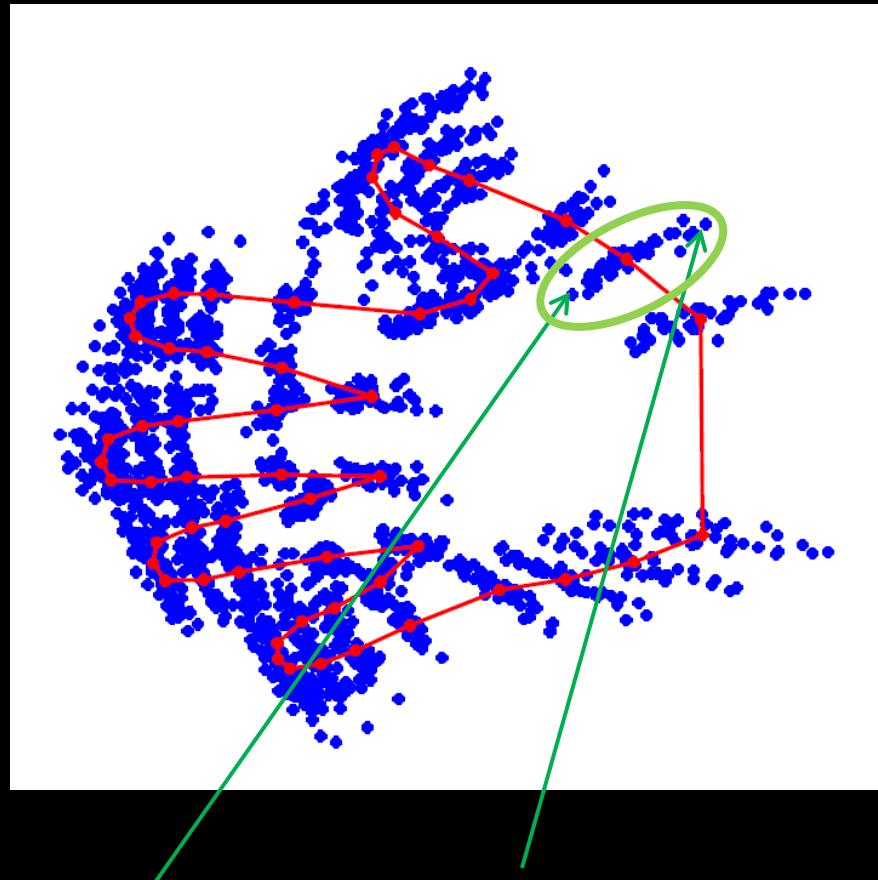
# Procrustes Analysis

1. Average shape is set to shape #1
2. Register all shapes to the average shape
  - Landmark based registration
3. Recompute the average shape
4. If average shape changed return to step 2.



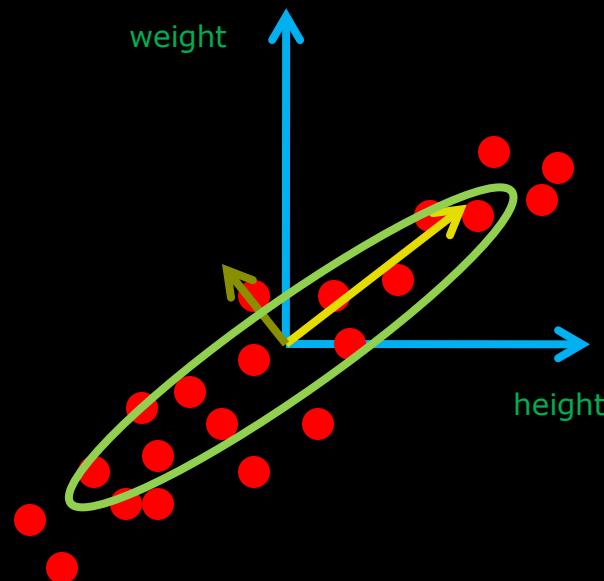
$$\bar{\mathbf{x}} = \frac{1}{N} \sum_{i=1}^N \mathbf{x}_i$$

# Aligned shapes – what now



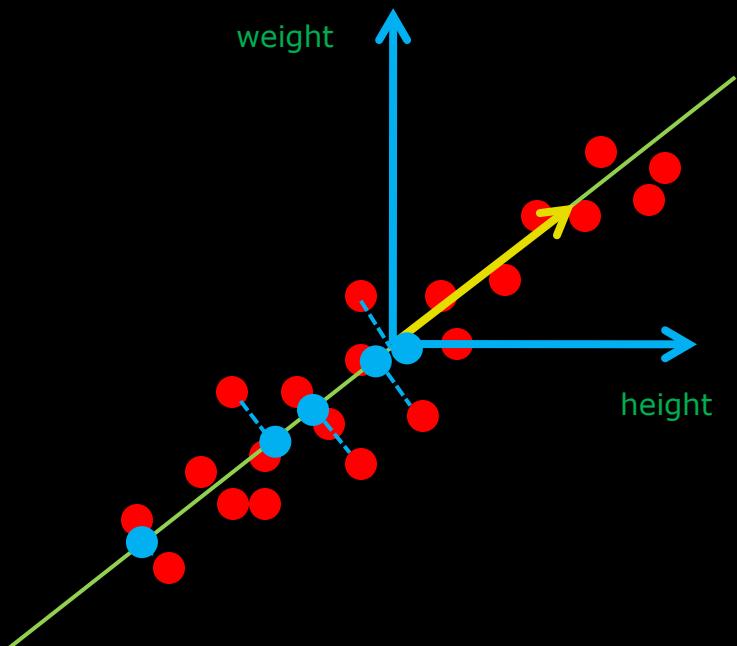
- Individual landmark variation
  - Over the training set
- What shape is the variation?

# Principal Component Analysis (PCA)



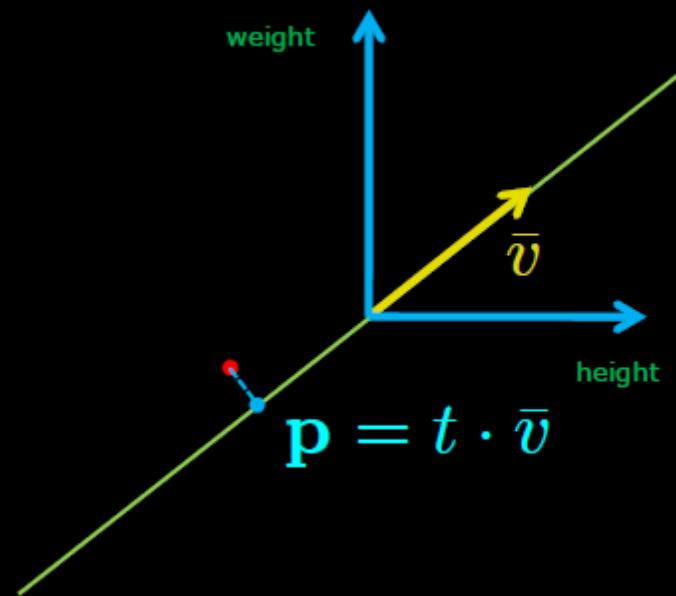
- PCA
  - Main axis in data
  - Eigenvectors
  - Eigenvalues
- Size of Eigenvalues describe explained variance

# Principal Component Analysis (PCA)



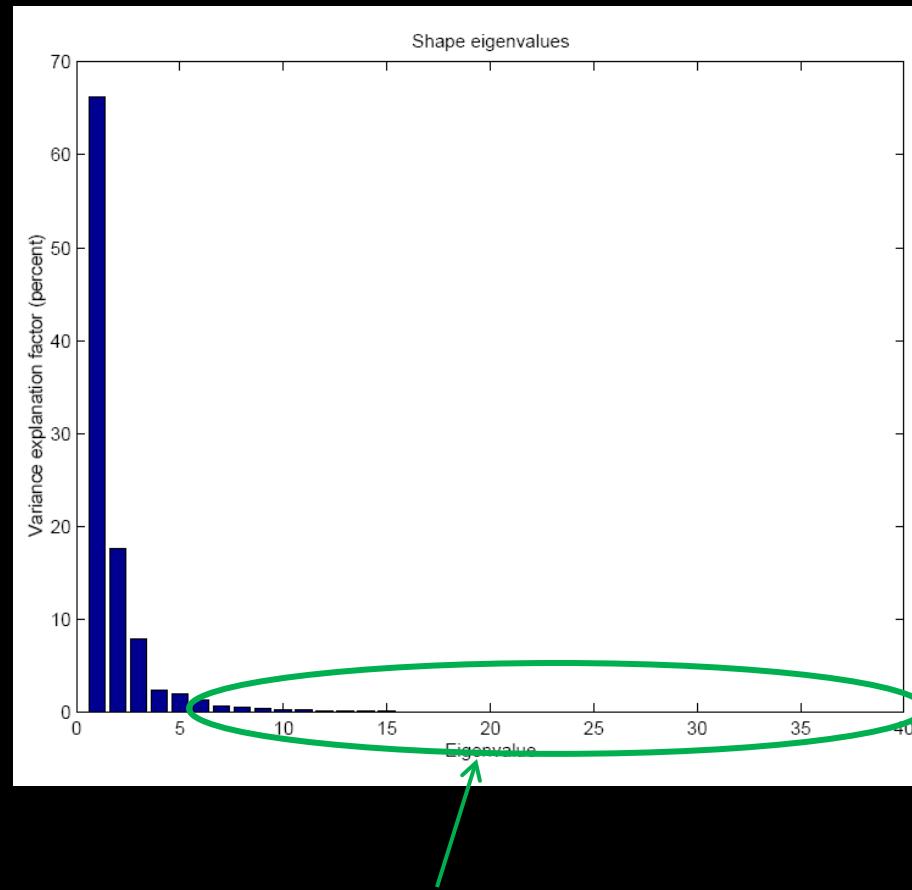
- We throw away the *noise dimensions*
- Points projected to the line

# Principal Component Analysis (PCA)



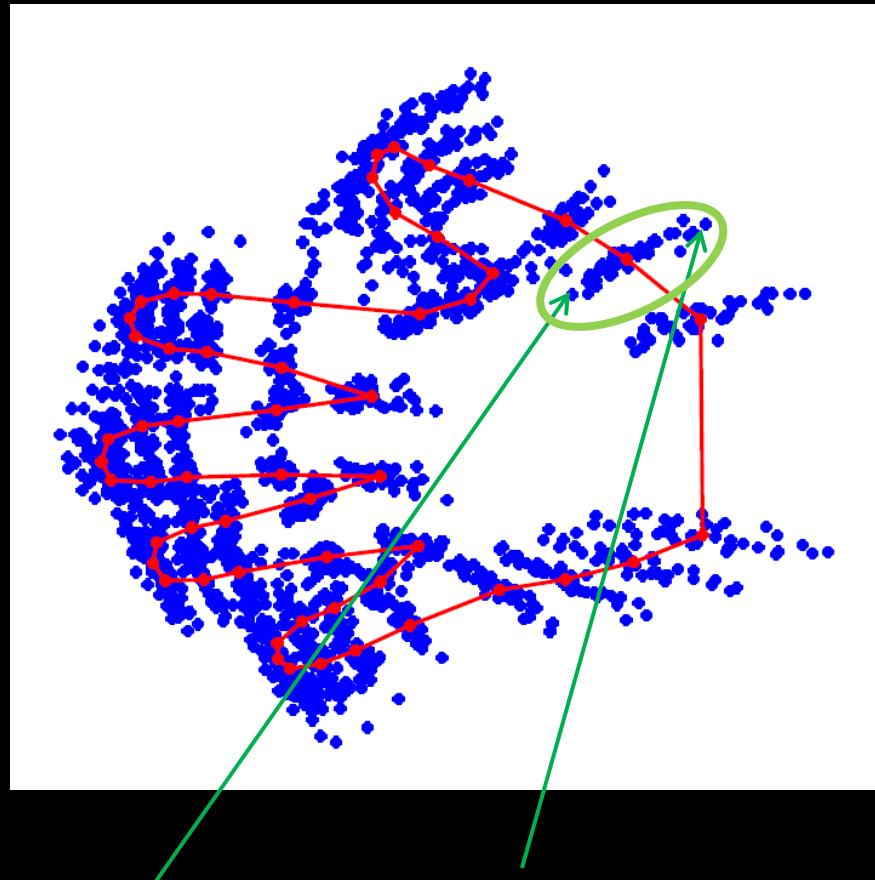
- We throw away the *noise dimensions*
- Points projected to the line
- A point can now be described by one parameter  $t$
- We have reduced the number of dimensions

# How many dimensions should we keep?



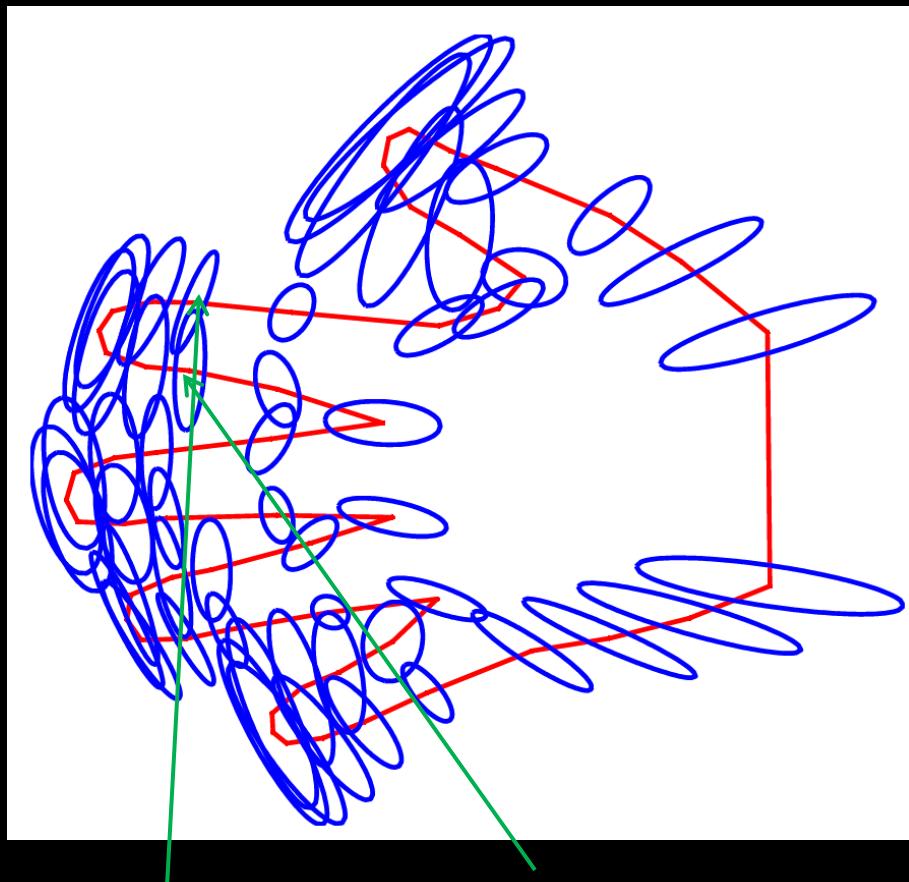
- Plot the Eigenvalues
- Explains how *important* each dimension is
- Cut away noise dimensions

# Aligned shapes – what now



- Individual landmark variation
  - Over the training set
- What shape is the variation?

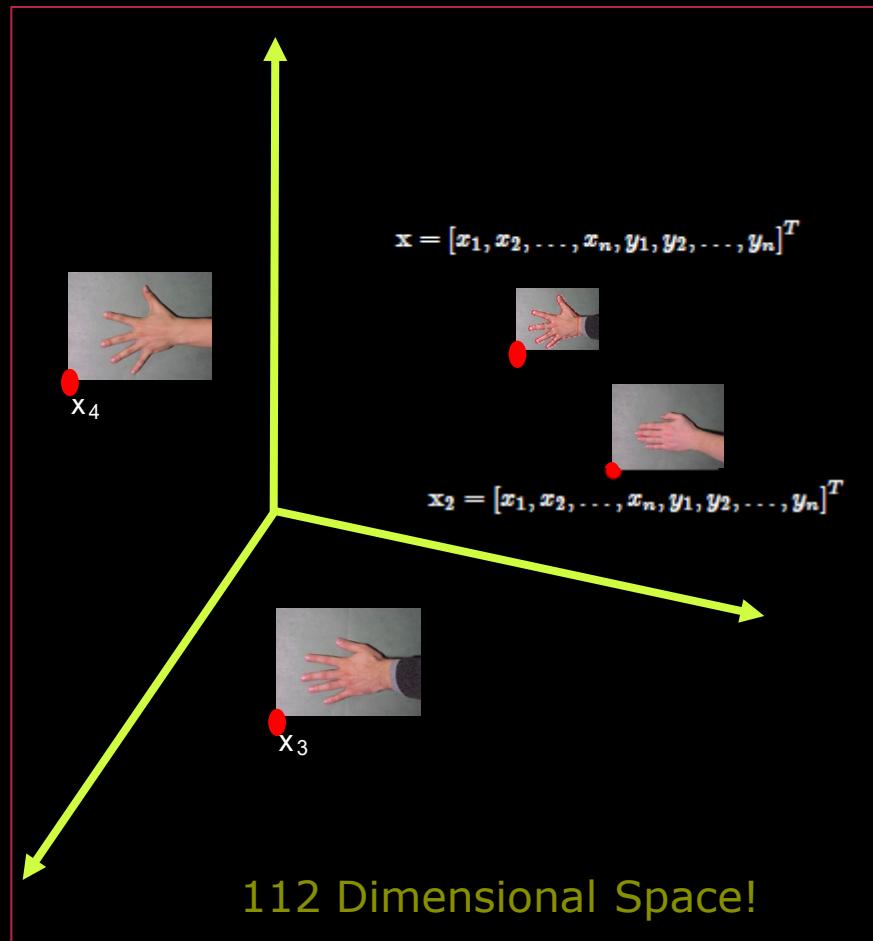
# PCA Analysis



Landmark #14   Landmark #22

- PCA analysis on individual landmarks
- Describes the major direction of variation
- **Landmarks are correlated!**
- The movement over the shape is connected
- Return to shape space

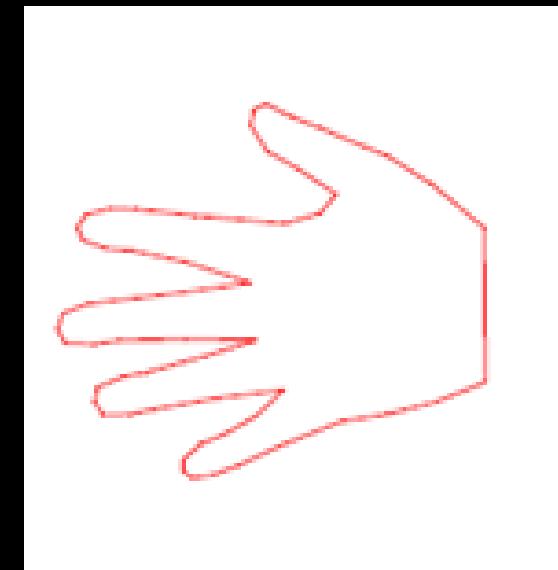
# PCA in shape space



- Instead of doing PCA on 2D points we do it on 112D points
- Examine if our 40 *shapes is lying on a plane* in 112D space
- We find the directions that spans the maximum variation in shape space

# Start by computing the shape average

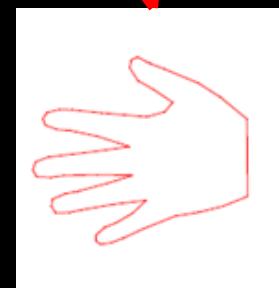
$$\bar{\mathbf{x}} = \frac{1}{s} \sum_{i=1}^s \mathbf{x}_i$$



Since we do this on the aligned shapes –  
this is the Procrustes average

# Do the eigenvector analysis

$$\mathbf{S} = \frac{1}{s-1} \sum_{i=1}^s (\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^T$$

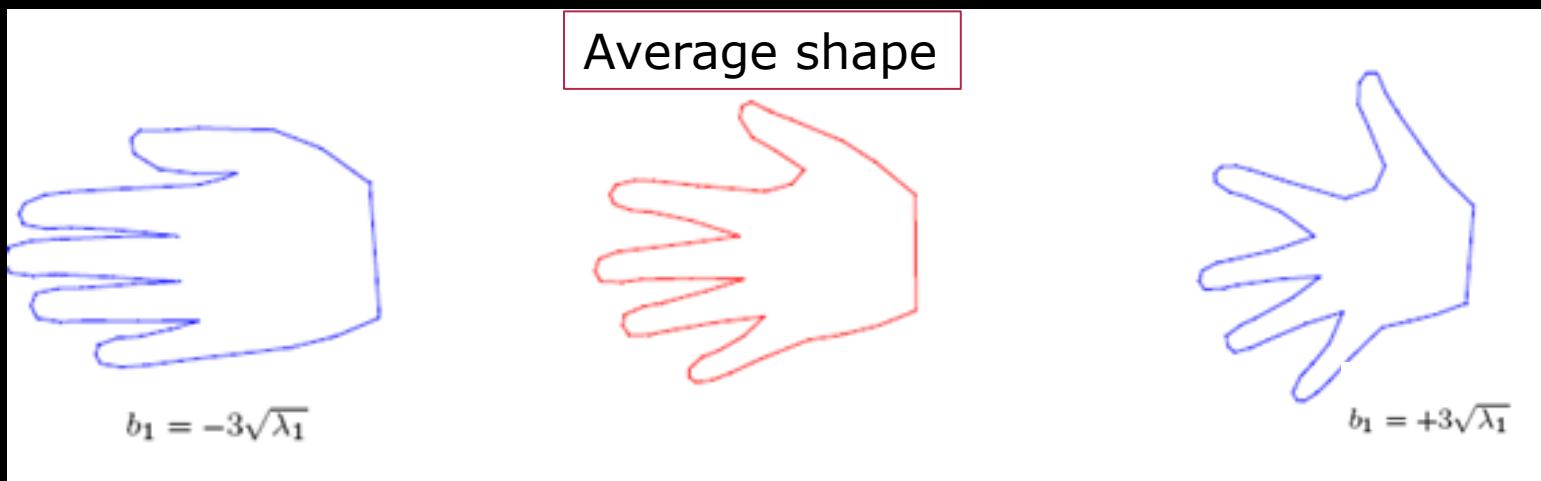


Average shape

Shape number  $i$  in the training set

- Computing the covariance of the shape data

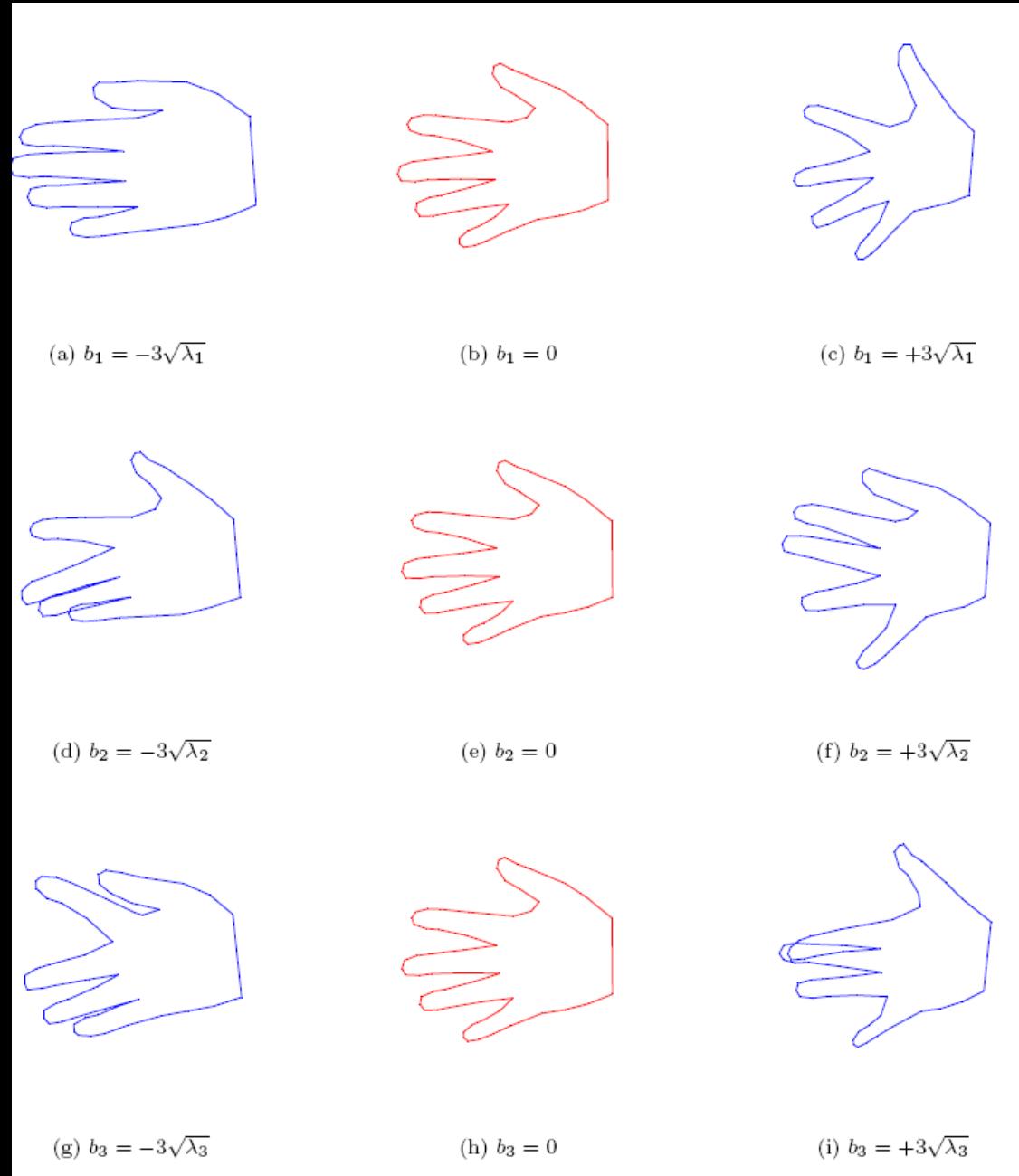
# Visualizing variation



Visualizing the first principal component

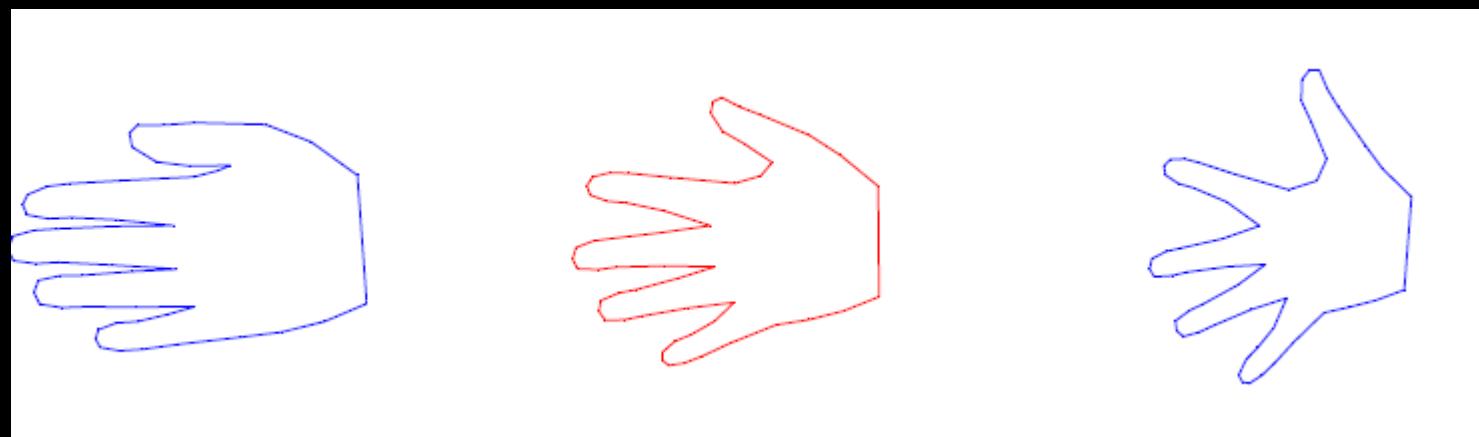
$$\mathbf{x} \approx \bar{\mathbf{x}} + \Phi \mathbf{b}$$

$\Phi$  contains the  $t$  eigenvectors



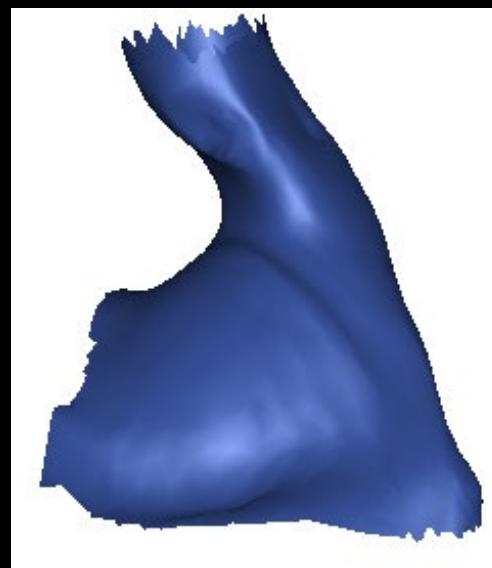
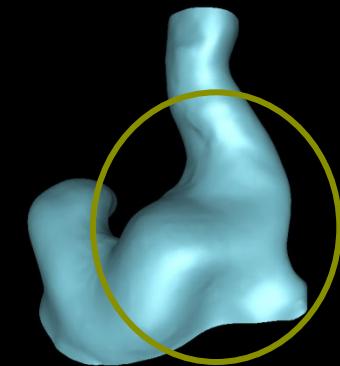
# Results of Shape Analysis

- Visualisation of the major variation of the shape over a population

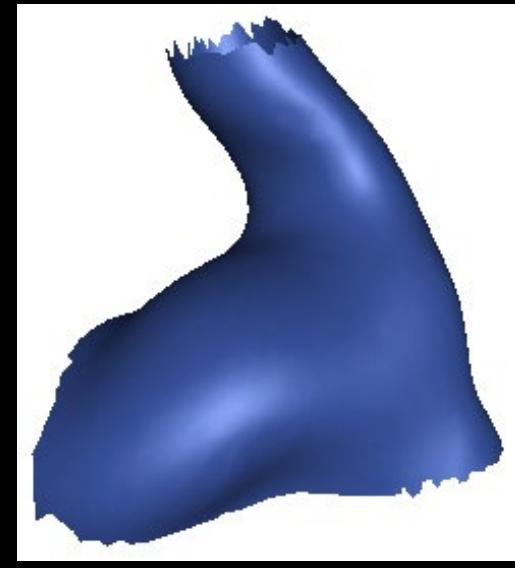


# Hearing Aid Design

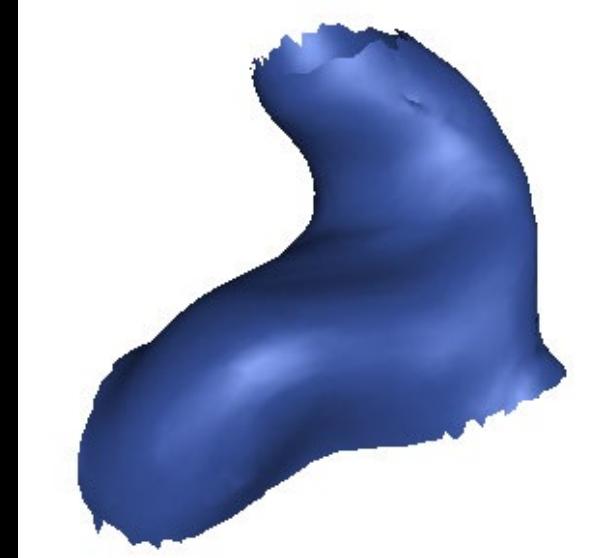
- Main variation of the shape of the ear canal
- Found using principal component analysis
- First mode of variation
- 7 modes explain 95% of the total variation



Average-1. mode



Average



Average+1. mode

## Modelling shape and appearance

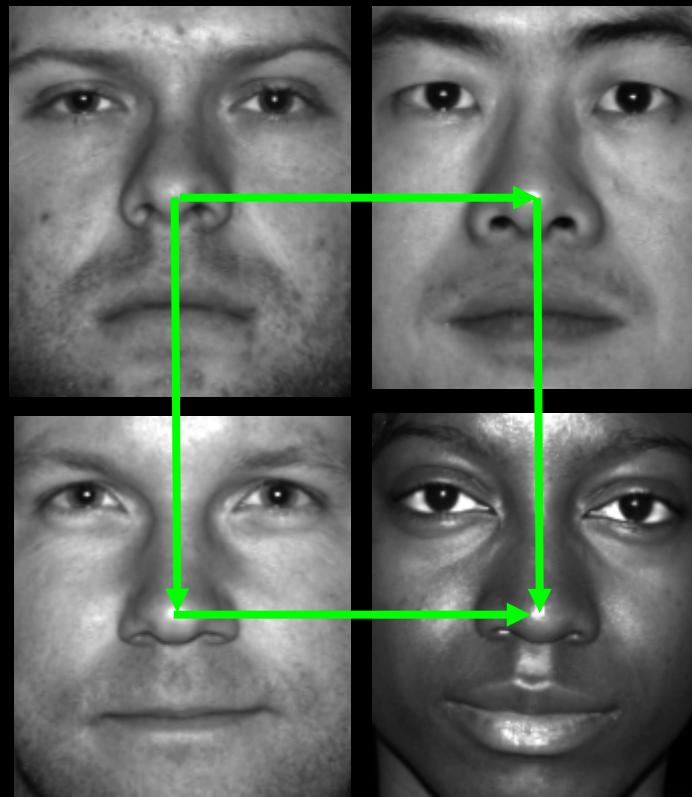
- A model that can both model the shape of an object and the appearance (the texture)
- **Texture:** The pattern of intensities (or colors) across an image patch





# Back to lecture 3: Eigenfaces

# Face data

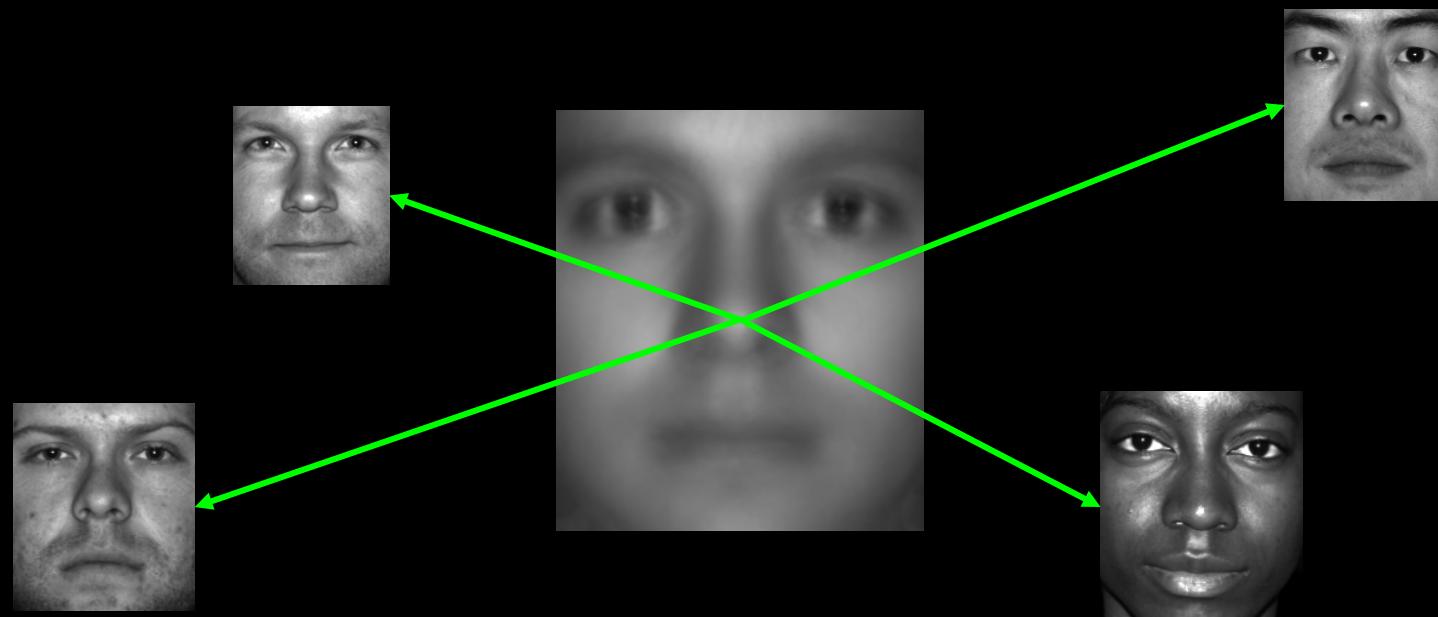


- 38 face images
  - $168 \times 192$  grayscale
- Aligned
  - The anatomy is placed "in the same position in all image"
- Same illumination conditions on the images we use

The Extended Yale Face Database B

# Analyzing the deviation from the mean face

- We want to do the principal component analysis on the *deviations from the average face*



# Visualizing the PCA faces

*Main deviations from the average face*



First PC – 40% of variation



-PC

Average face

+PC

Second PC – 8% of variation

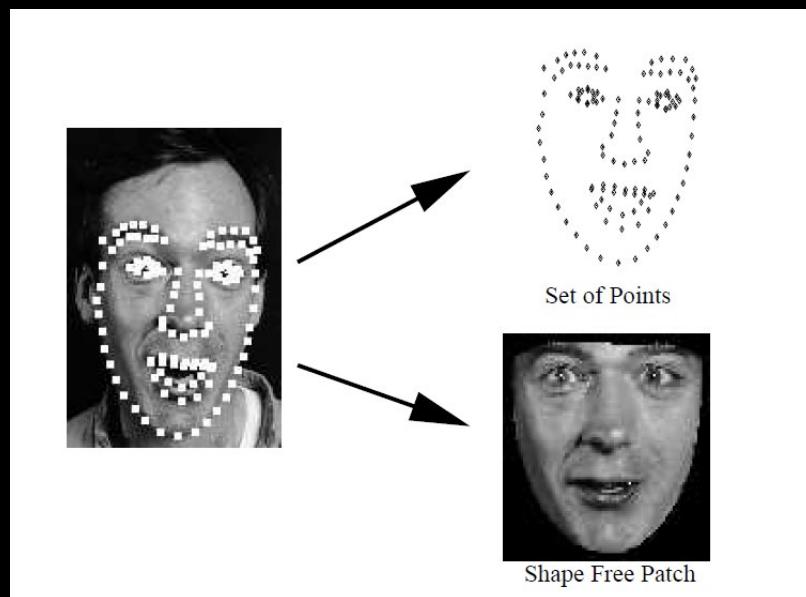
A tool to see major variations –  
brow lifting

# Eigenfaces: Modelling texture

- The modelling of the pure appearance
- Without removing variation in shape
- No *decoupling* of shape and appearance

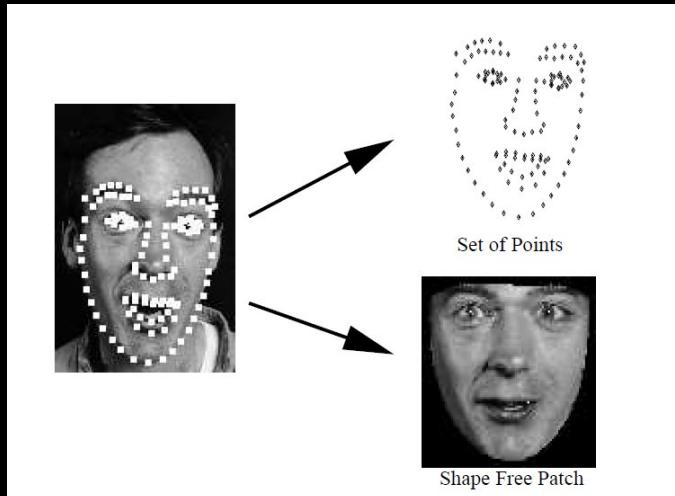


# Decoupling shape and texture



- Warp each face to average shape using the landmarks
- Non-linear geometrical transformation
- Sample the texture from the warped face

# Eigenfaces on warped faces



- Same PCA modelling as in the Eigenfaces approach
- Just slightly different notation



$$\mathbf{g} = \bar{\mathbf{g}} + \mathbf{P}_g \mathbf{b}_g$$

# Combined shape and appearance model

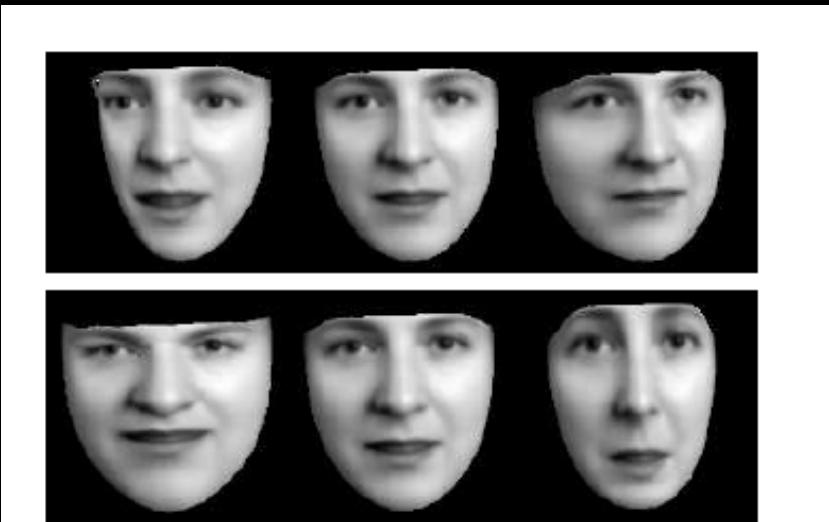


Figure 5.2: First two modes of shape variation ( $\pm 3$  sd)



Figure 5.3: First two modes of grey-level variation ( $\pm 3$  sd)



# Facial Analysis

- Demo of AAM explorer



# Image Analysis

Rasmus R. Paulsen

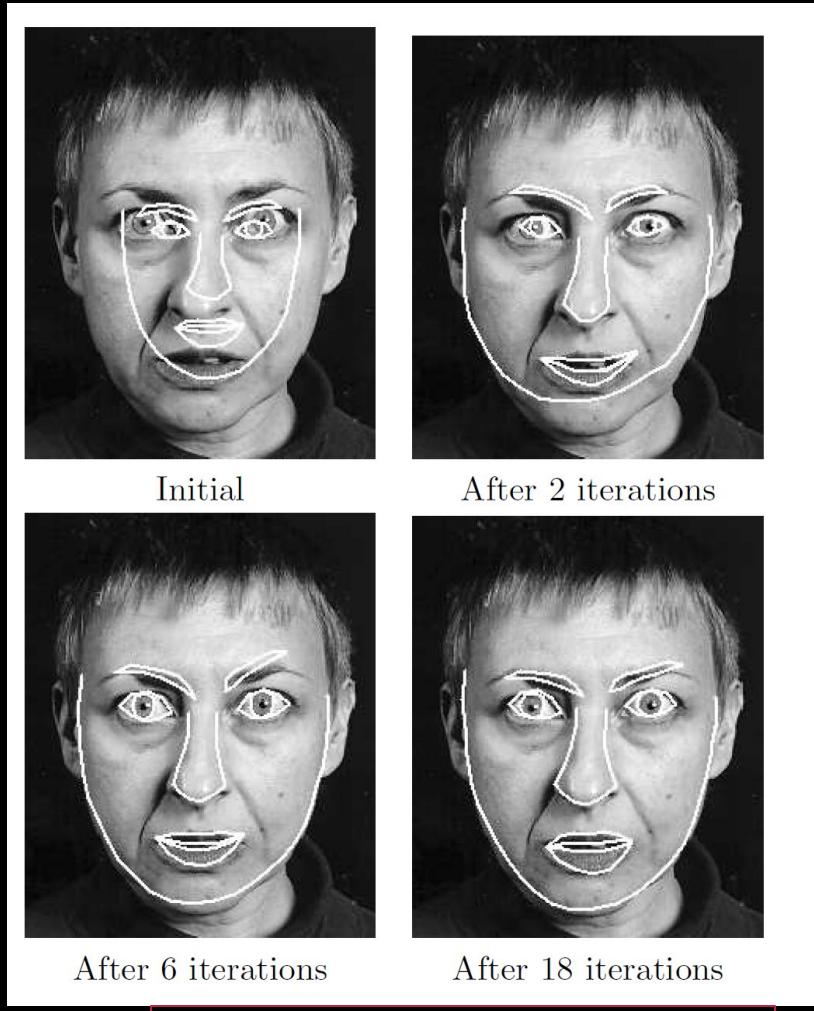
Tim B. Dyrby

DTU Compute

[rapa@dtu.dk](mailto:rapa@dtu.dk)

<http://courses.compute.dtu.dk/02502>

# Lecture 12b – Active shape models

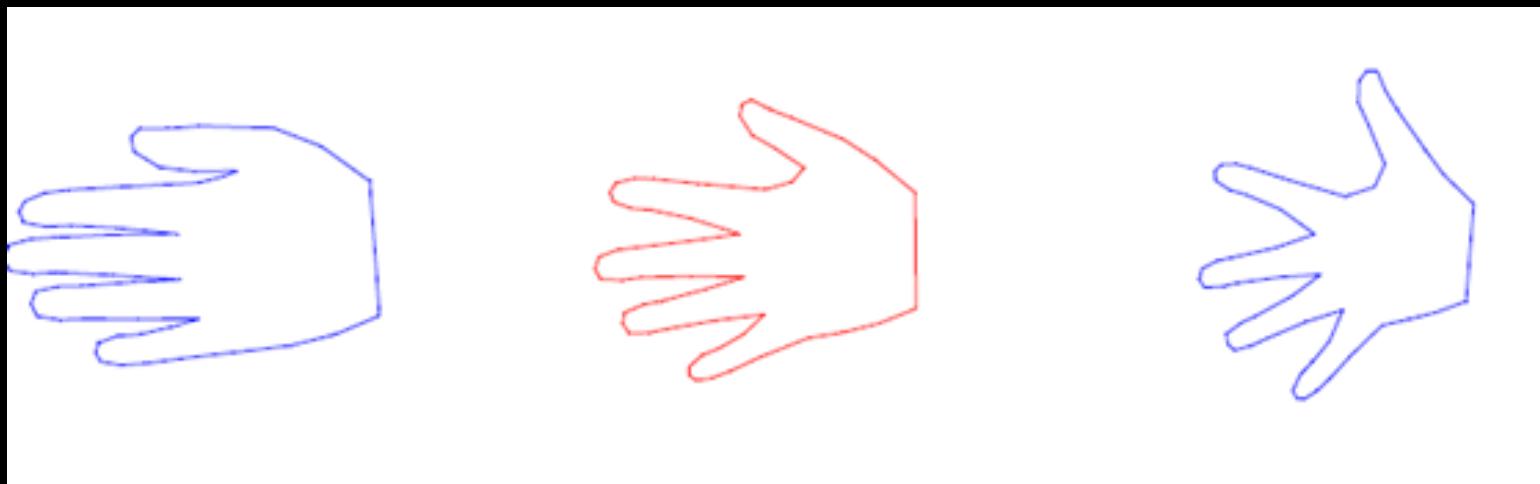


Tim Cootes: Active shape models

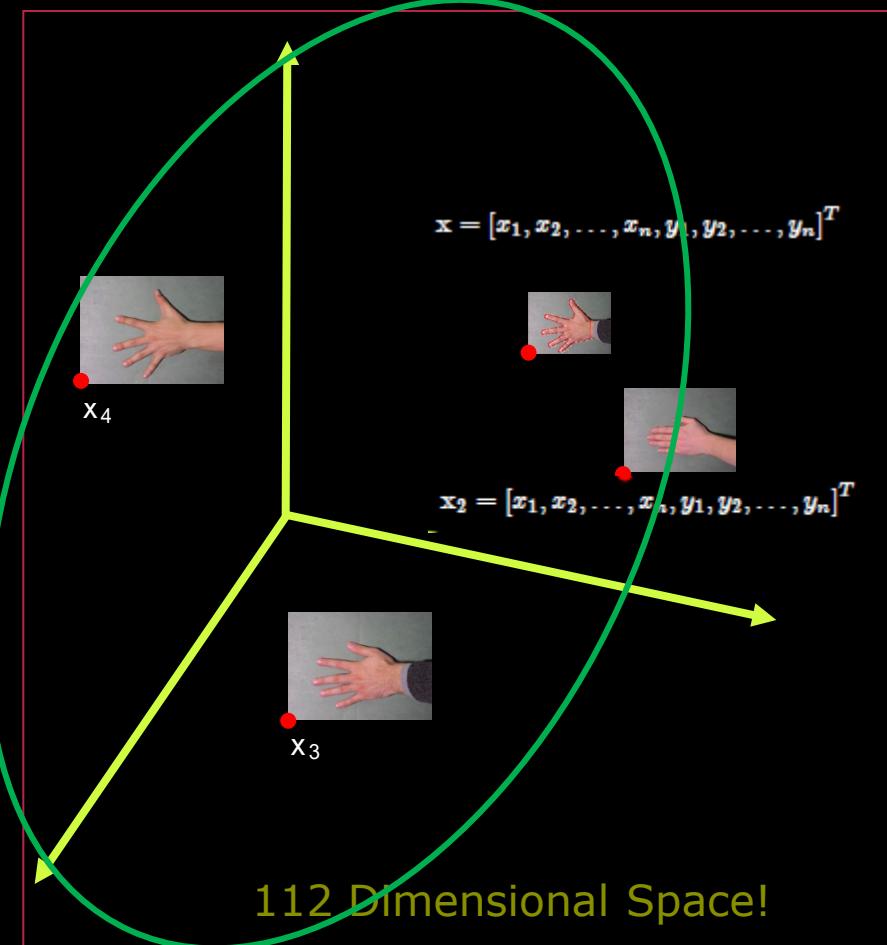
# Today's Learning Objectives

- Describe how shapes can be synthesized using the shape space
- Describe the generative model based on a statistical shape model
- Describe the concept of analysis by synthesis
- Describe how the Eigenvectors and Eigenvalues can be used to constrain a shape model
- Describe how a statistical shape model can be fitted using the gradients in an image
- Describe how a statistical shape model can be fitted by modelling local variation
- Explain the problem of strong priors in statistical models

# We have a statistical model of shape

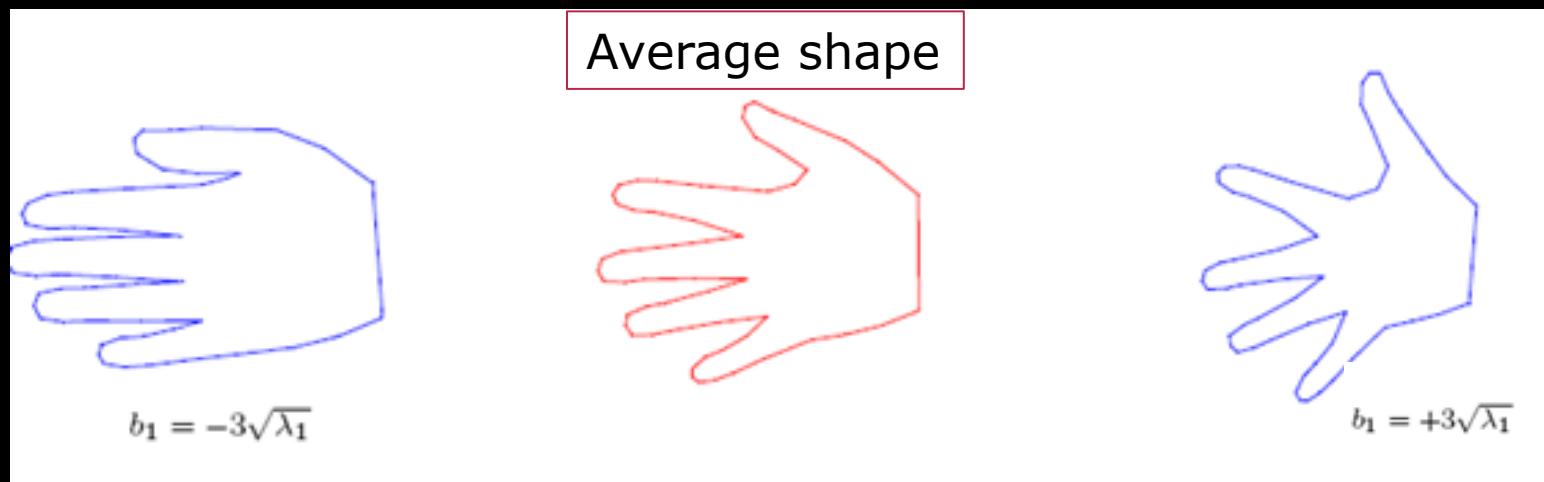


# Shape space



- A mapping of the shape space
- PCA based description of the “hand space”

# Synthesizing new shapes

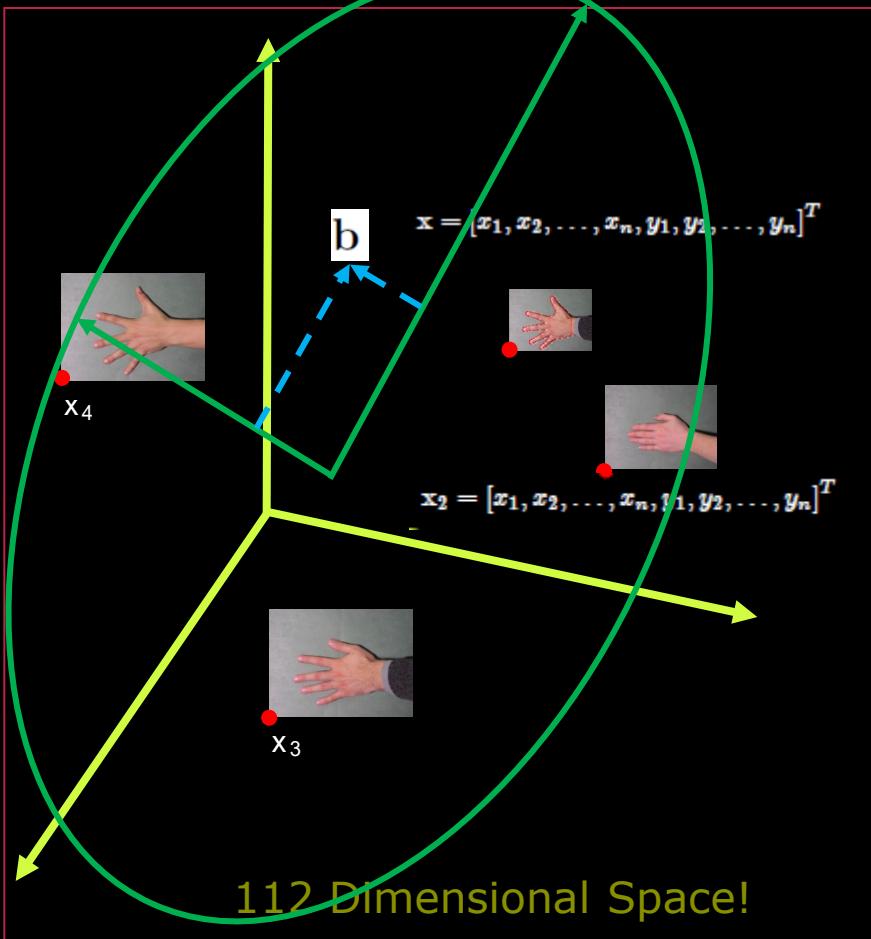


$$\mathbf{x} \approx \bar{\mathbf{x}} + \Phi \mathbf{b}$$

$\Phi$  contains the  $t$  eigenvectors of



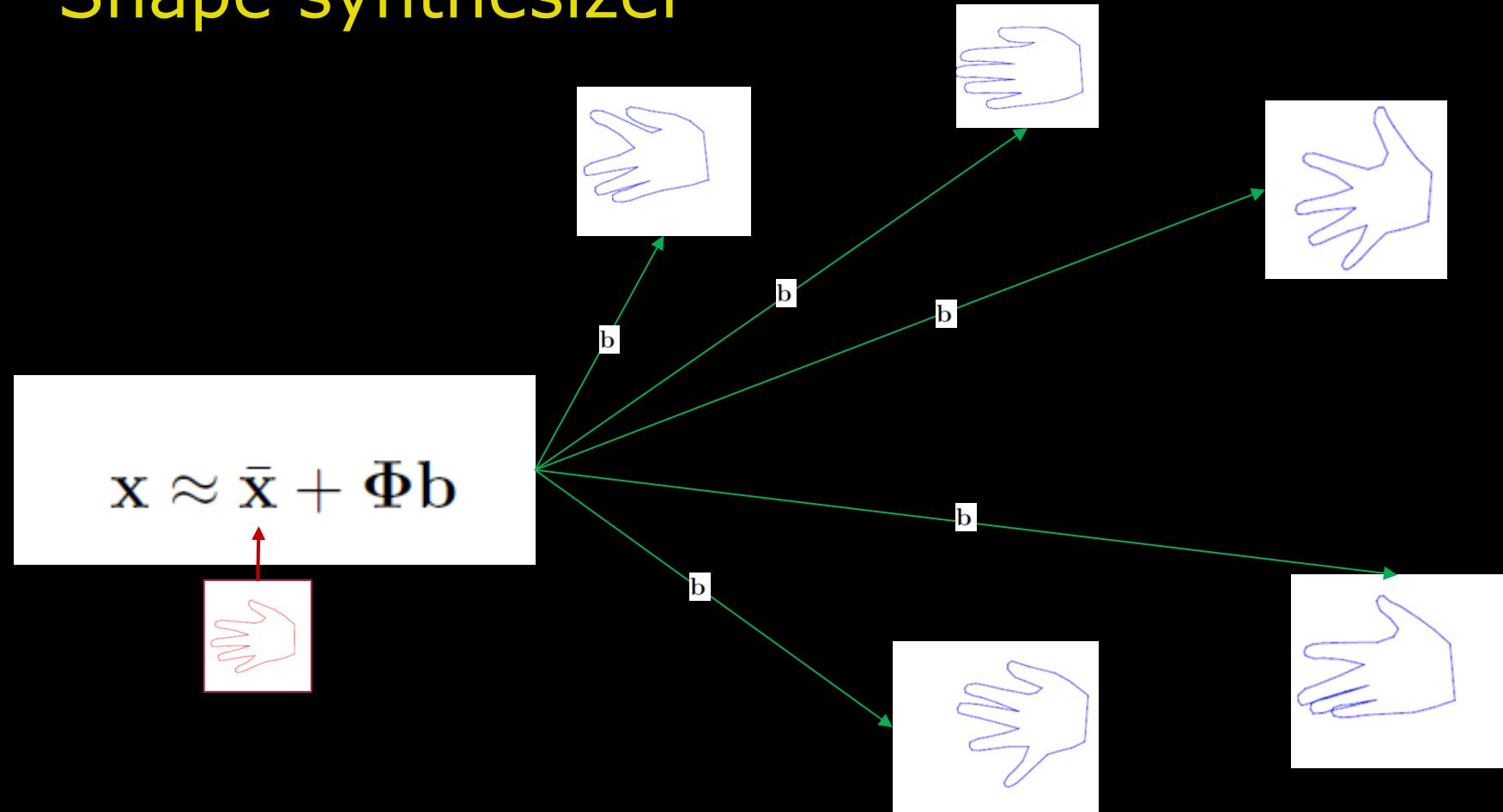
# Shape space



- We can *sample* new shapes by moving around in shape space
- $b$  are the *coordinates* in shape space
- The shape space is defined by the *Eigenvectors*
- $b$  are the coordinates on the Eigenvectors

$$\mathbf{x} \approx \bar{\mathbf{x}} + \Phi \mathbf{b}$$

# Shape synthesizer



A *generative* model

# Shape synthesizer



b

$$\mathbf{x} \approx \bar{\mathbf{x}} + \Phi \mathbf{b}$$



A *generative* model

- b needs to be *constrained*
- Should be bounded by the learned shape space
- Using the size of the Eigenvalues

$$-3\sqrt{\lambda_1} < b_1 < 3\sqrt{\lambda_1}$$

# Shape synthesizer + geometrical transformation

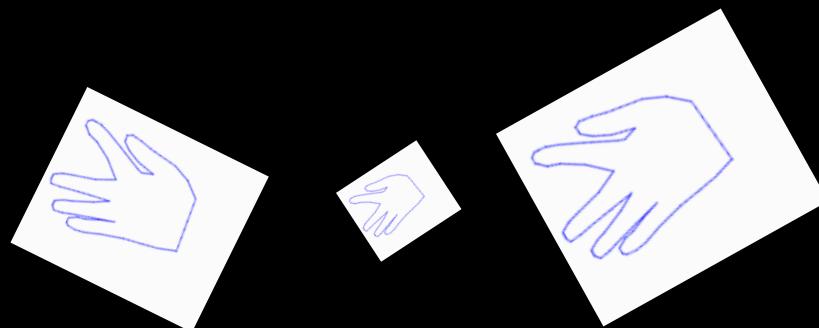


b

$$\mathbf{x} \approx \bar{\mathbf{x}} + \Phi \mathbf{b}$$



- Adding a geometrical transformation
  - Translation  $X_t, Y_t$
  - Scale  $s$
  - Rotation  $\theta$



A *generative* model

# Pattern recognition

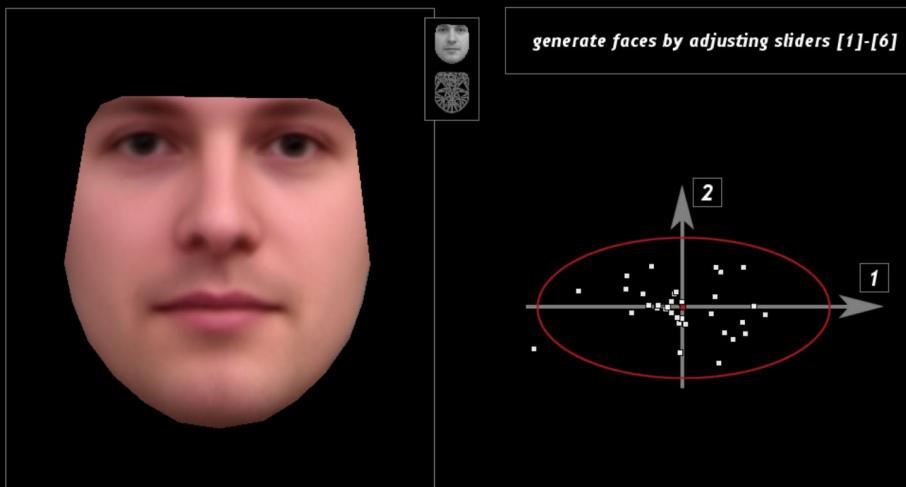


- Classical image analysis
- Hand crafting features
  - Eye detector
  - Nose detector
  - Mouth detector

Hybrid approach – Viola Jones. Learning a limited sets of features

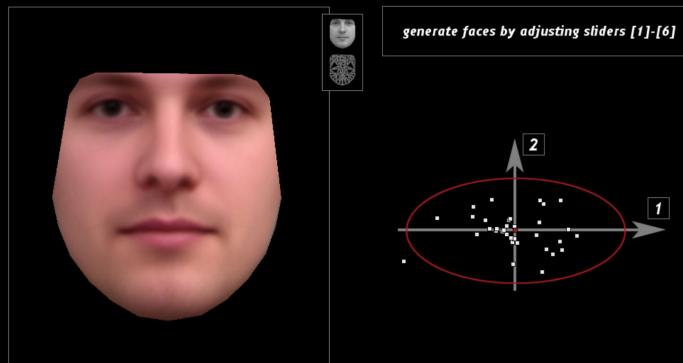
# Analysis by synthesis

- We have a generative model



- A face synthesizer
- A face is represented by a few parameters:  $b$

# Analysis by synthesis



Generative model



Target

- Compare synthetic face with target face
  - Sum of squared differences
- Change parameters of model until difference is minimal
  - Position, rotation, scaling
  - b vector

Similar to image registration with a deformable *moving image*

# Fitting a shape and appearance model

- Finding the optimal set of parameters: position, rotation, size and b vector of model
- An *optimization* problem
- In general very hard
- Custom solutions exist

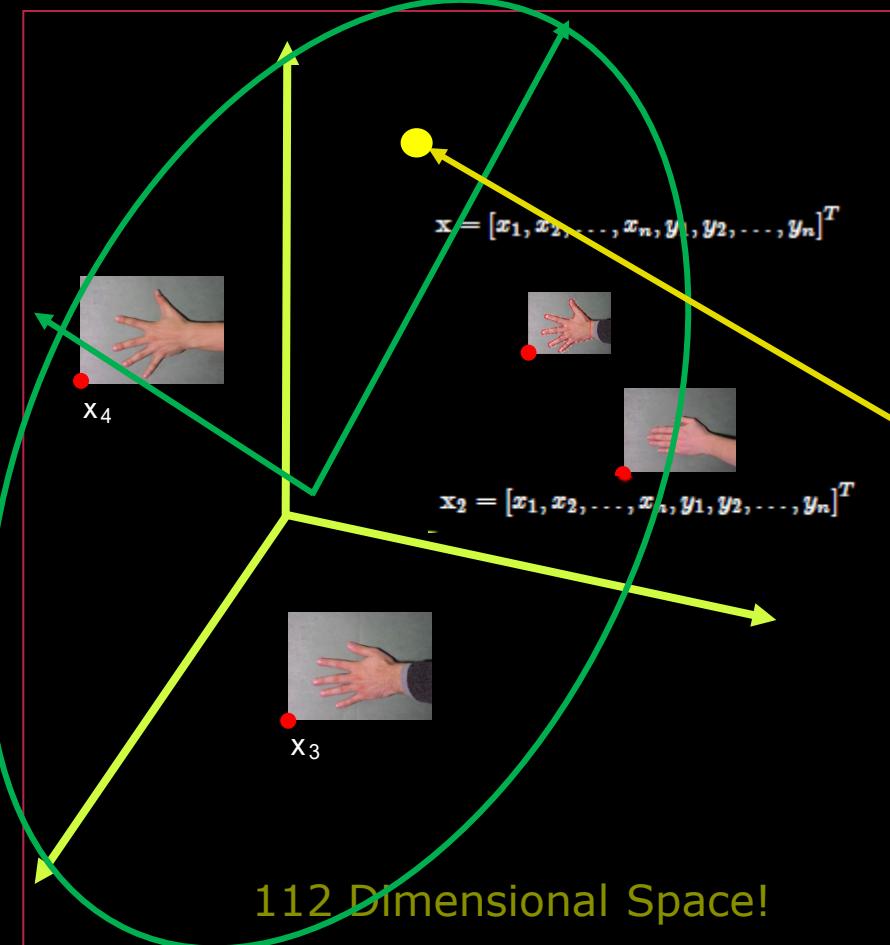


Left: Fitted model  
Right: Real photo



Tim Cootes: Active Appearance models

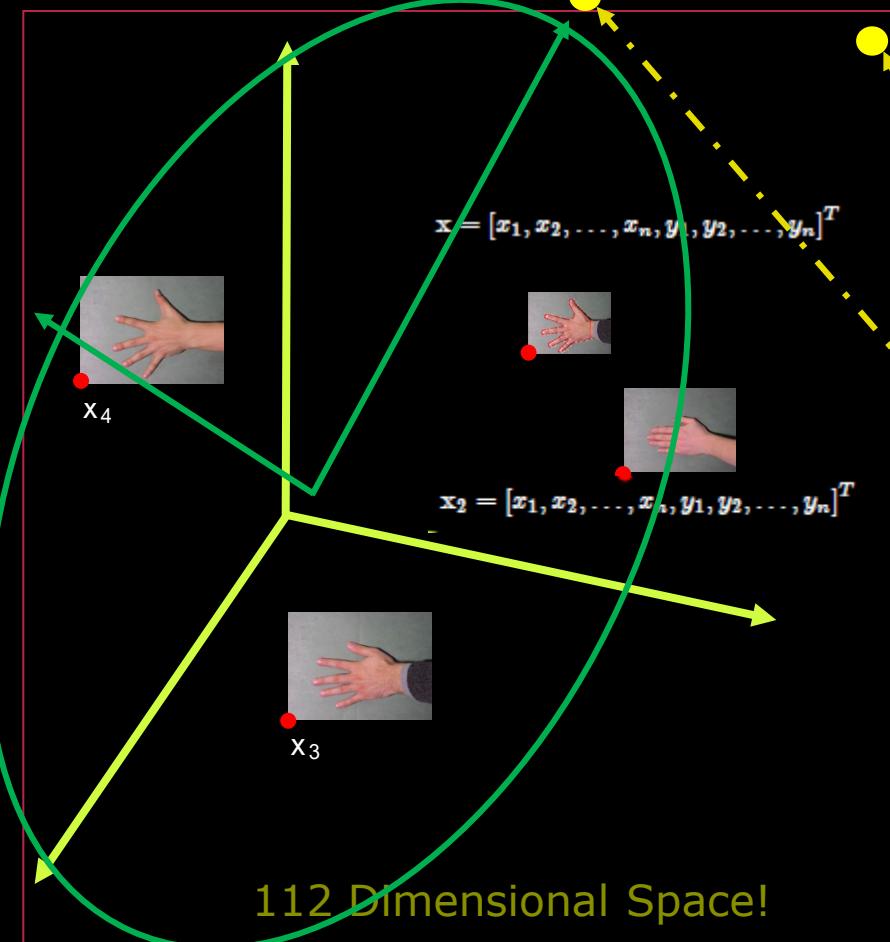
# Using the shape space



- Given a shape
  - It can be placed in shape space

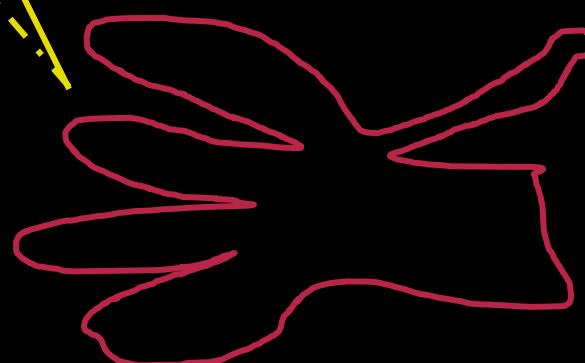


# Using the shape space

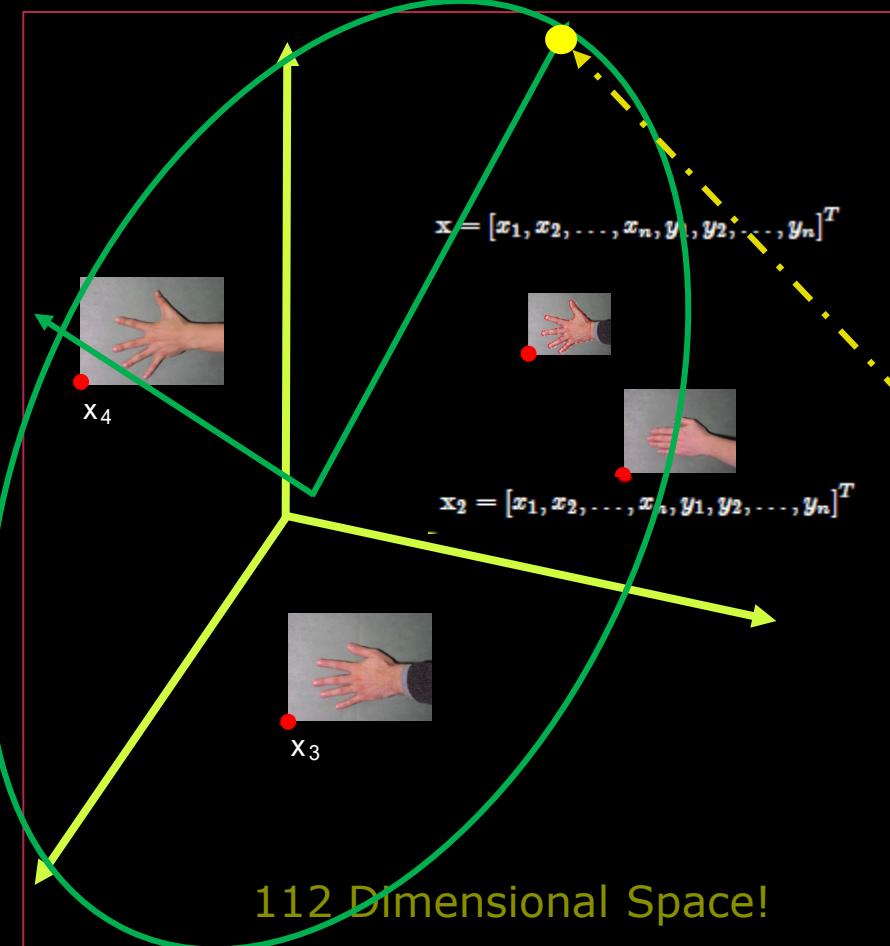


- Given a shape
  - It can be placed in shape space
- It can be projected to the Eigenvectors

Not anatomically plausible



# Using the shape space



- Given a shape
  - It can be placed in shape space
- It can be projected to the Eigenvector
- And bounded by the Eigenvalues

$$-3\sqrt{\lambda_1} < b_1 < 3\sqrt{\lambda_1}$$



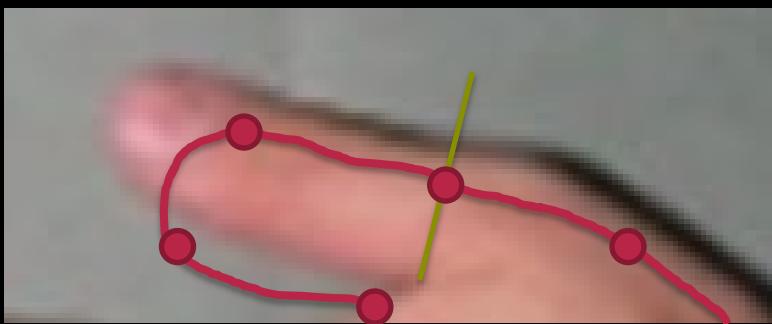
Closest anatomically plausible shape

# Fitting the active shape model to a new image



- Place the average shape on top
- Fit model points to actual image

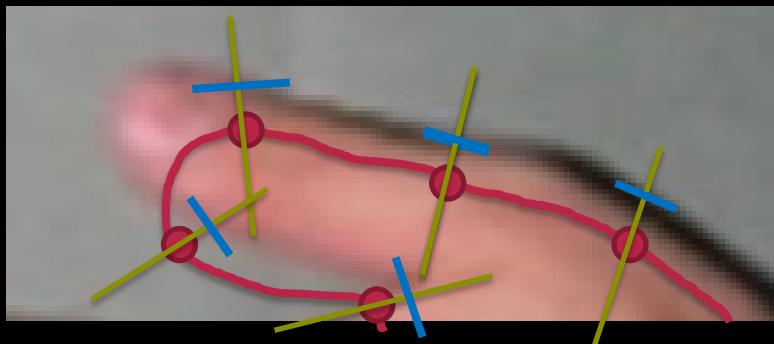
# Fitting the active shape model to a new image



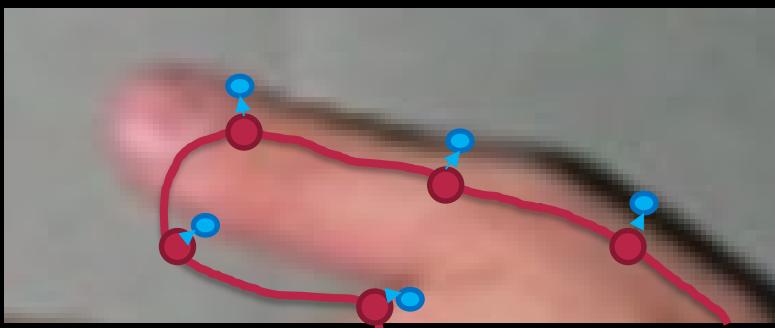
- Fit model points to actual image
- For each point:
  - Search along normal direction
  - Find highest grey level gradient

# Fitting the active shape model to a new image

- Fit model points to actual image
- For each point:
  - Search along normal direction
  - Find highest grey level gradient

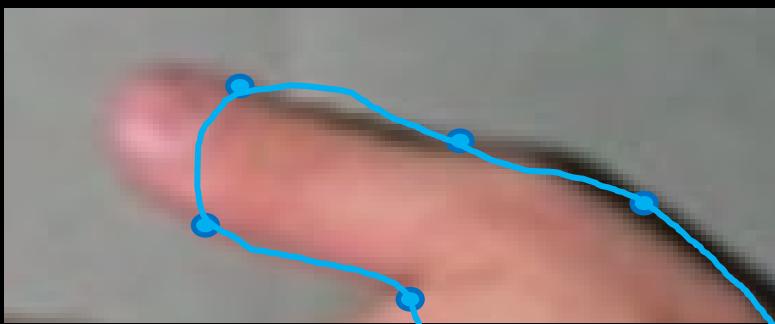


# Fitting the active shape model to a new image



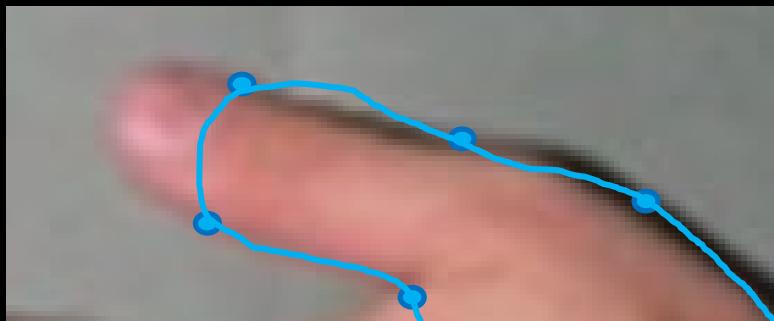
- Compute translation, rotation and scaling
  - Landmark based registration
- Move points to create **new shape**

# Fitting the active shape model to a new image



- Compute translation, rotation and scaling
  - Landmark based registration
- Move points to create **new shape**

# Fitting the active shape model to a new image



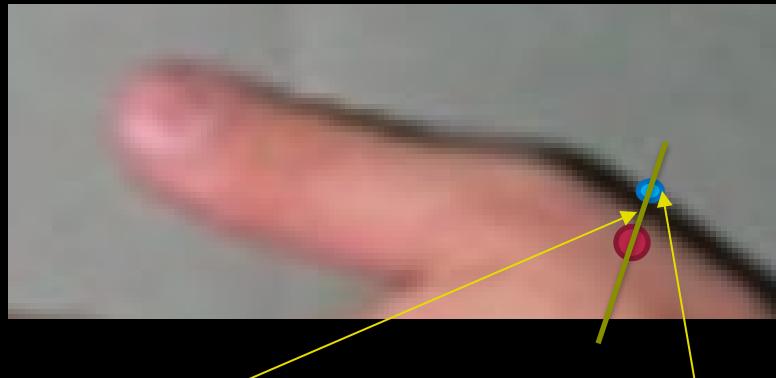
- Put new shape in shape space
- Project on Eigenvectors
- Constrain using Eigenvalues
  - Also called *regularization*

**Result:** Shape that matches image and that is anatomically plausible

$$-3\sqrt{\lambda_1} < b_1 < 3\sqrt{\lambda_1}$$

# Modelling local structure

- The boundary is not always where there is highest gradient

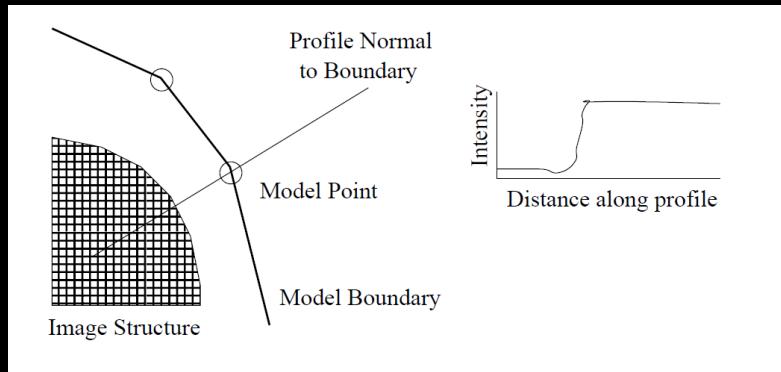
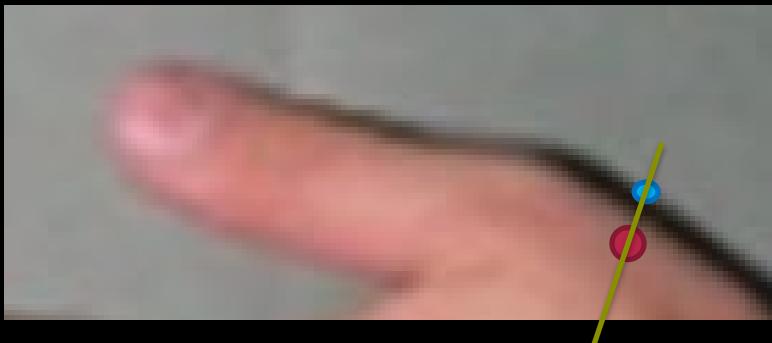


True boundary

Highest gradient

# Modelling local structure

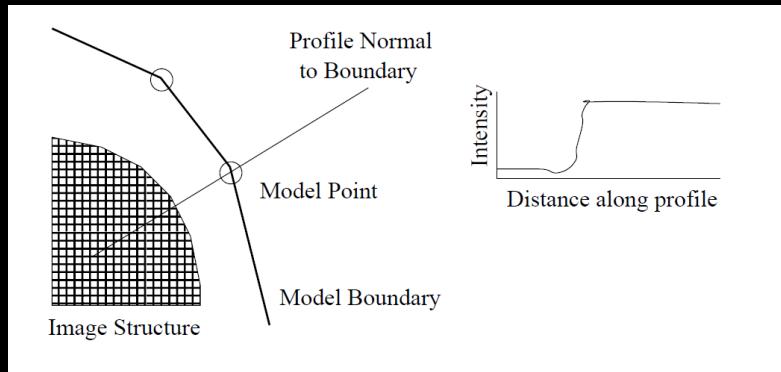
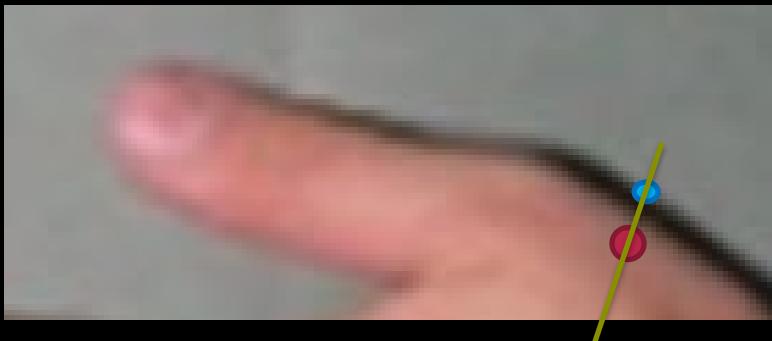
- Sample along profile
- Normalise using sum of values



$$\mathbf{g}_i \rightarrow \frac{1}{\sum_j |g_{ij}|} \mathbf{g}_i$$

# Modelling local structure

- Approximate distribution of samples
  - Multivariate Gaussian

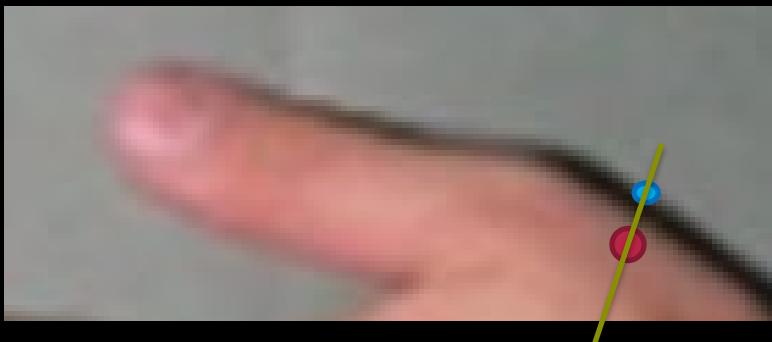


mean  $\bar{\mathbf{g}}$  and covariance  $\mathbf{S}_g$

$$\mathbf{g}_i \rightarrow \frac{1}{\sum_j |g_{ij}|} \mathbf{g}_i$$

# Modelling local structure

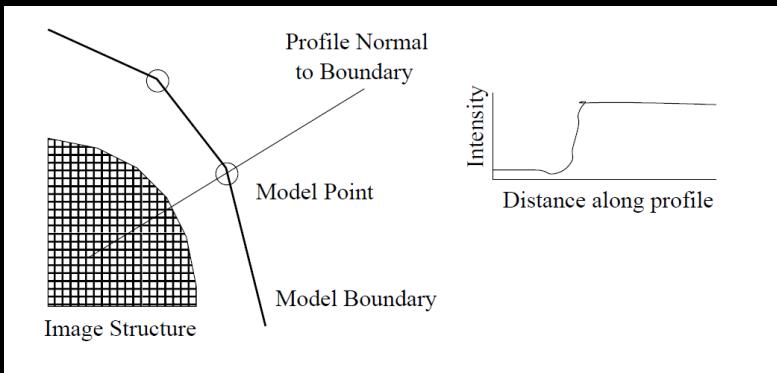
- Instead of using the gradient to search, a quality of fit is used:



The quality of fit of a new sample,  $\mathbf{g}_s$ , to the model is given by

$$f(\mathbf{g}_s) = (\mathbf{g}_s - \bar{\mathbf{g}})^T \mathbf{S}_g^{-1} (\mathbf{g}_s - \bar{\mathbf{g}})$$

This is the Mahalanobis distance of the sample from the model mean



$$\mathbf{g}_i \rightarrow \frac{1}{\sum_j |g_{ij}|} \mathbf{g}_i$$

Tim Cootes: Active Appearance models

# Modelling local structure

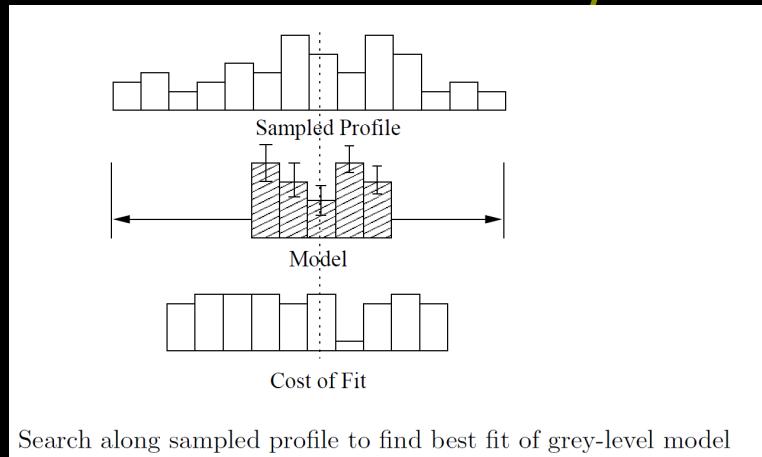
- Instead of using the gradient to search, a quality of fit is used:



The quality of fit of a new sample,  $\mathbf{g}_s$ , to the model is given by

$$f(\mathbf{g}_s) = (\mathbf{g}_s - \bar{\mathbf{g}})^T \mathbf{S}_g^{-1} (\mathbf{g}_s - \bar{\mathbf{g}})$$

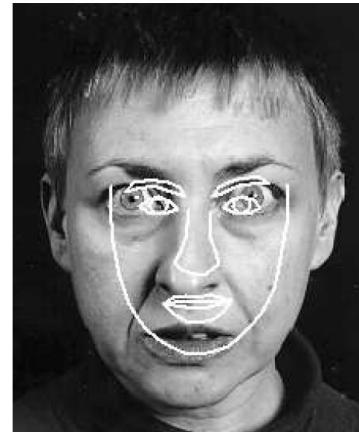
This is the Mahalanobis distance of the sample from the model mean



Search along sampled profile to find best fit of grey-level model

Tim Cootes: Active Appearance models

# Fitting to a new shape



Initial



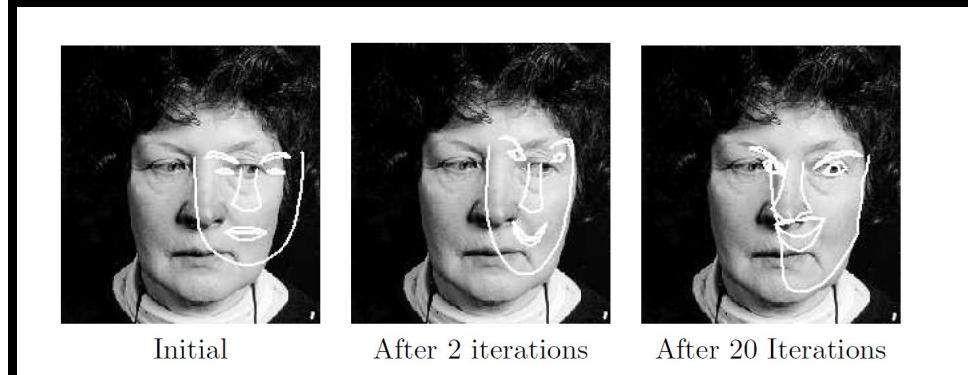
After 2 iterations



After 6 iterations



After 18 iterations



Initial

After 2 iterations

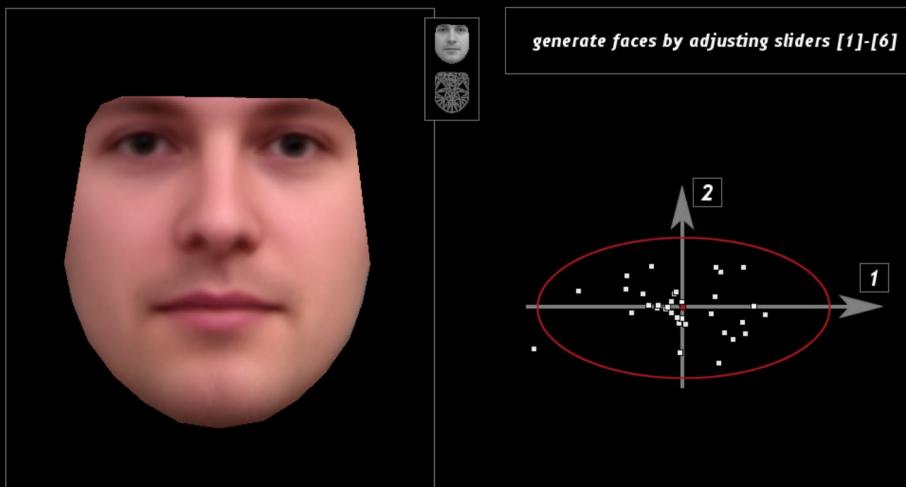
After 20 Iterations

Tim Cootes: Active Appearance models

# The problem with strong priors

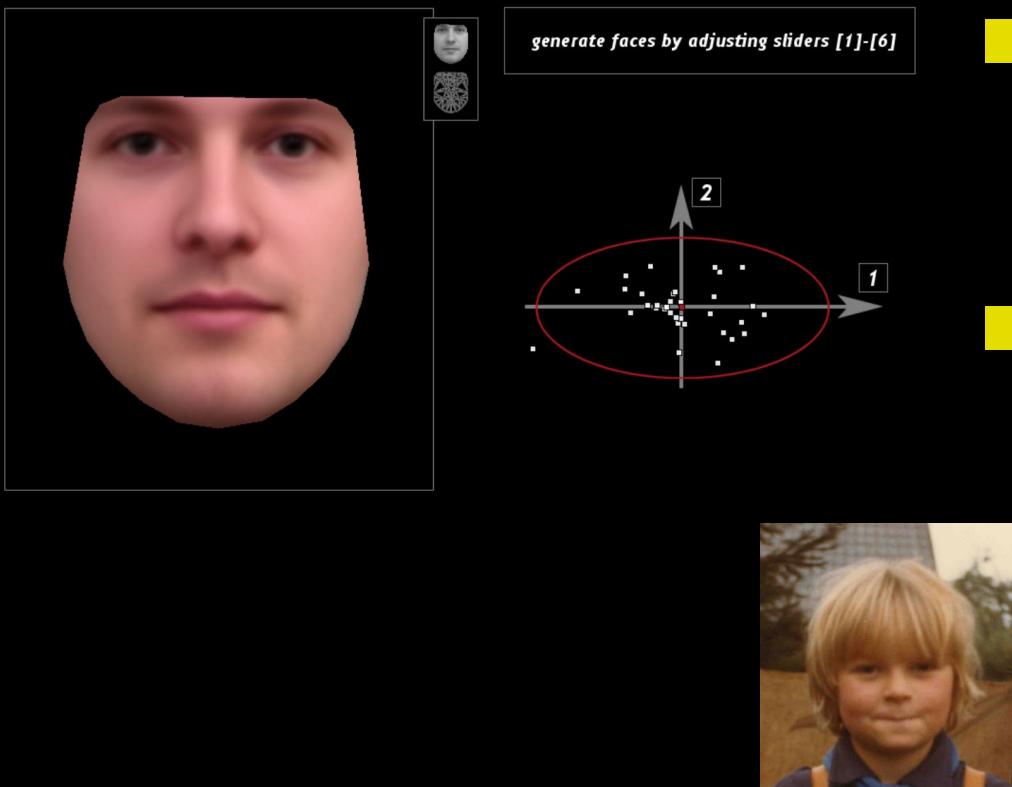
## ■ *A prior*

- What was known before
- A statistical shape model



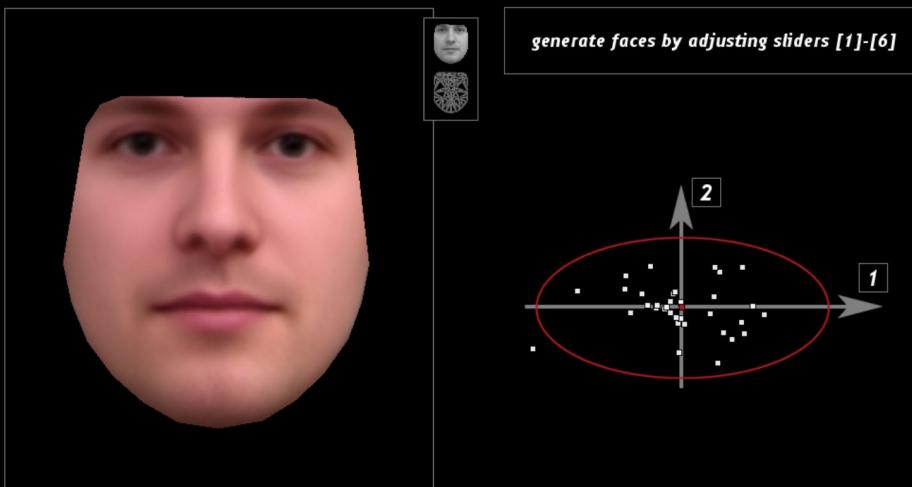
# The problem with strong priors

- Model is trained on images of adults
- Will try to force all fits to *look like adults*
- Will not work well with images outside the *prior*

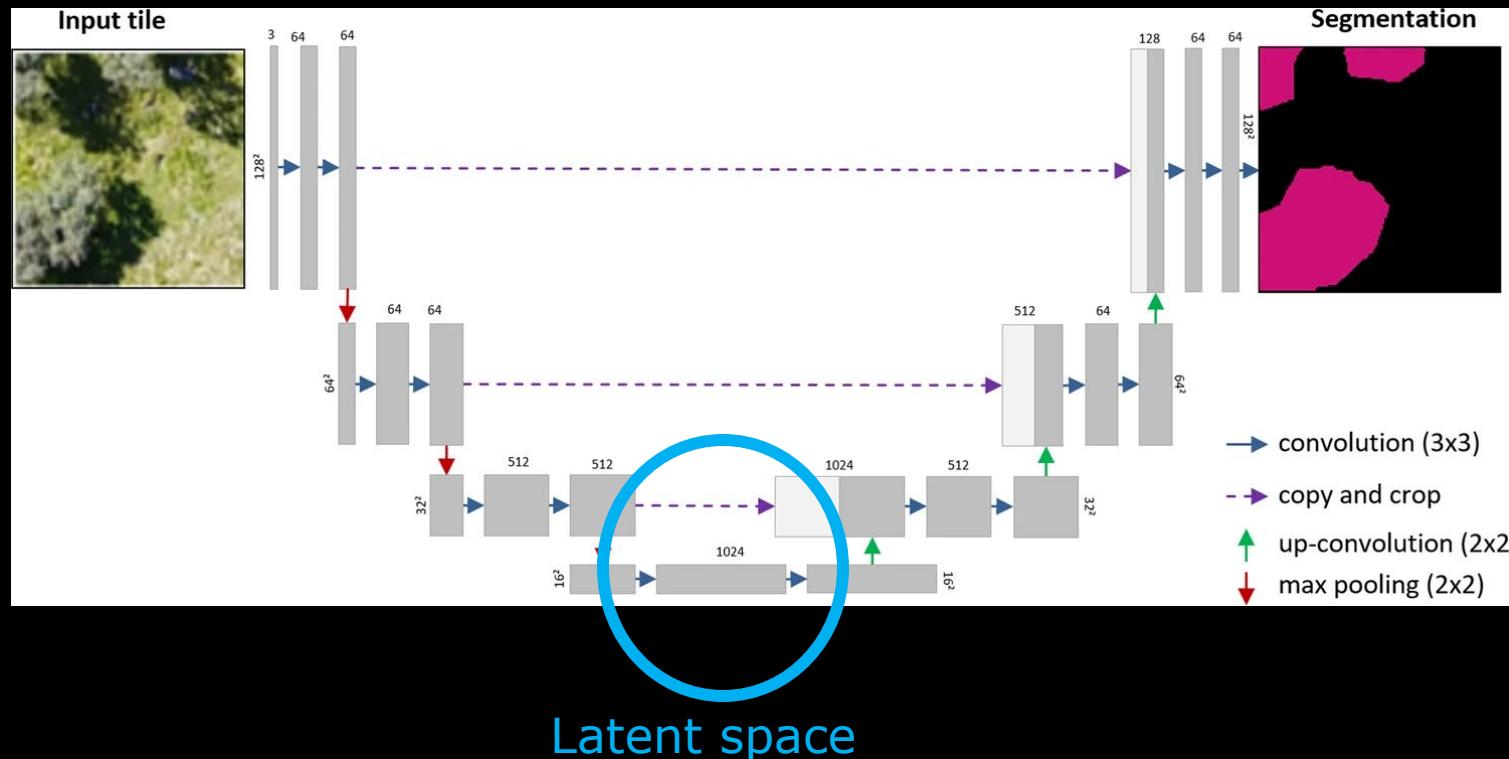


# Testing the model

- Important to the model on independent data
- How it *generalizes*
- Is the prior too strong?



# PCA space vs. Latent space



Kattenborn, T., Eichel, J. & Fassnacht, F.E. Convolutional Neural Networks enable efficient, accurate and fine-grained segmentation of plant species and communities from high-resolution UAV imagery. *Sci Rep* **9**, 17656 (2019).



## About the exam

- 4 hours written exam on DTU Campus
- **Very important:** Be sure you are seated at the right table
- Digital exam – multiple choice. 20-25 questions
- All aids allowed
  - Notes, computer, but not telephone
- **NEW!** Open net: You can access the internet
- **Very important:** You are not allowed to communicate with anyone during exam.



# The appendix / mellemregninger

- You should upload your code/prompts etc
- As a PDF or a text format
- NOT used for grading
- For plagiarism check and validations

# The exam

- What should I do if I find a problem with a question
  - Contact one of the monitors/tilsyn in the room
  - They will contact the exam administration and we will then come to the room
  - A formal procedure is necessary for logging, time extensions, IT support and fairness
  
- **DO NOT** contact the teachers or TAs directly using email – use the formal procedure above.



# The exam – about cheating and fairness

- By default, we believe you do not cheat and follow the code of conduct.
- The exam should be fair and measure the students ability to fulfil the learning objectives
- BUT having an open net of course introduces a risk of answers being distributed during the exam
  
- **DO NOT SHARE OR RECEIVE ANSWERS:**
  - Both the ones sharing an answer and the ones receiving an answer will undergo a juridical procedure and risk being expelled from DTU



# The use of AI tools during exam

You are allowed to use AI tools during the exam. By AI tools, we mean tools like Github Copilot and ChatGPT.

You are obliged to upload your appendix/code/solution/prompts on the "mellemregninger/appendix" part of the exam.

The grade is solely based on the answers of the multiple-choice part of the exam. The appendix is used for plagiarism and validation checks but is not used for grading.

On a personal note, we do believe that AI tools can give some help, but personally we would still validate the results and there is no better validation than being well prepared and have solved all course exercises.

We have informed the study office and all the observers of the exam should be aware of the allowed use of AI tools. In case of any doubts during the exam, let the exam supervisor know and they will contact us.