Image registration

Outline

Introduction

- What is image *registration*?
- *Motivation* and main applications

Geometric transformations

- Modify the *coordinates* of an image
- Basic *types* of transformations

General framework

- *Features* : which information to use in the registration
- *Similarity metrics* : measure how similar two images are
- *Transforms* : deformation model to transform one image into another
- *Optimizers* : algorithm to estimate the transformation
- *Interpolators* : how to compute common coordinates from different images

Image registration

- **Registration** is the *process of finding the transformation* (**T**) that puts different images (**f** and **g**) into spatial correspondence

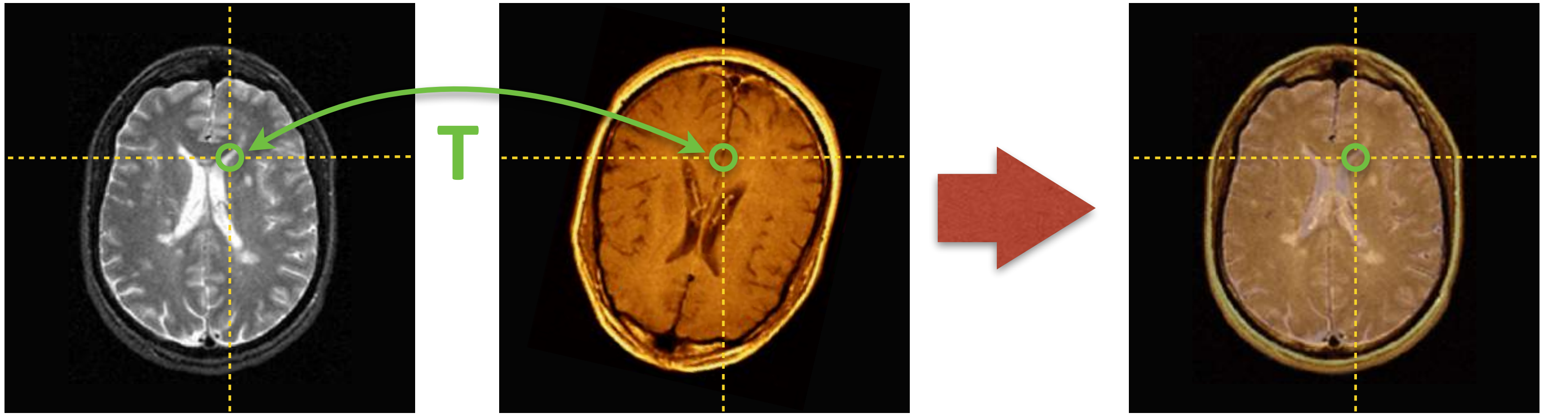


image **f** image **g** **g** “aligned” to **f**

- A **geometric transformation** is a function that maps each image coordinate pair (x,y) to a new location (x',y')

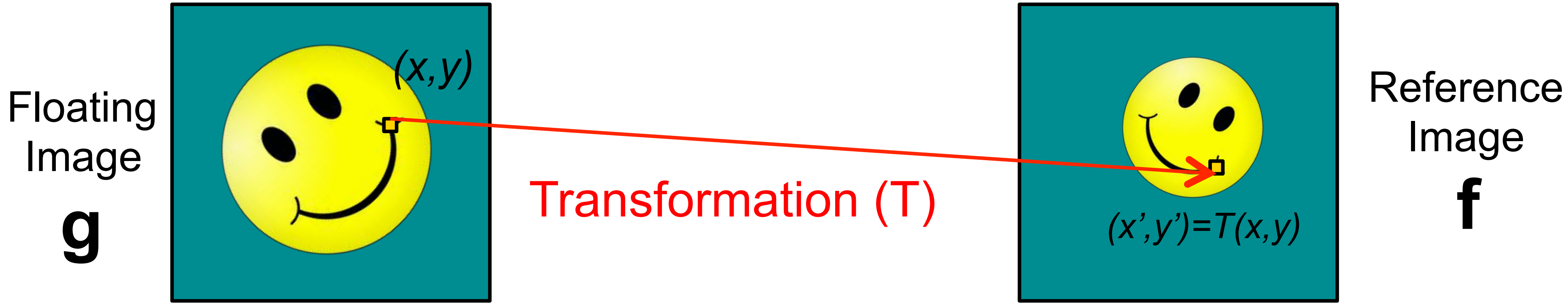
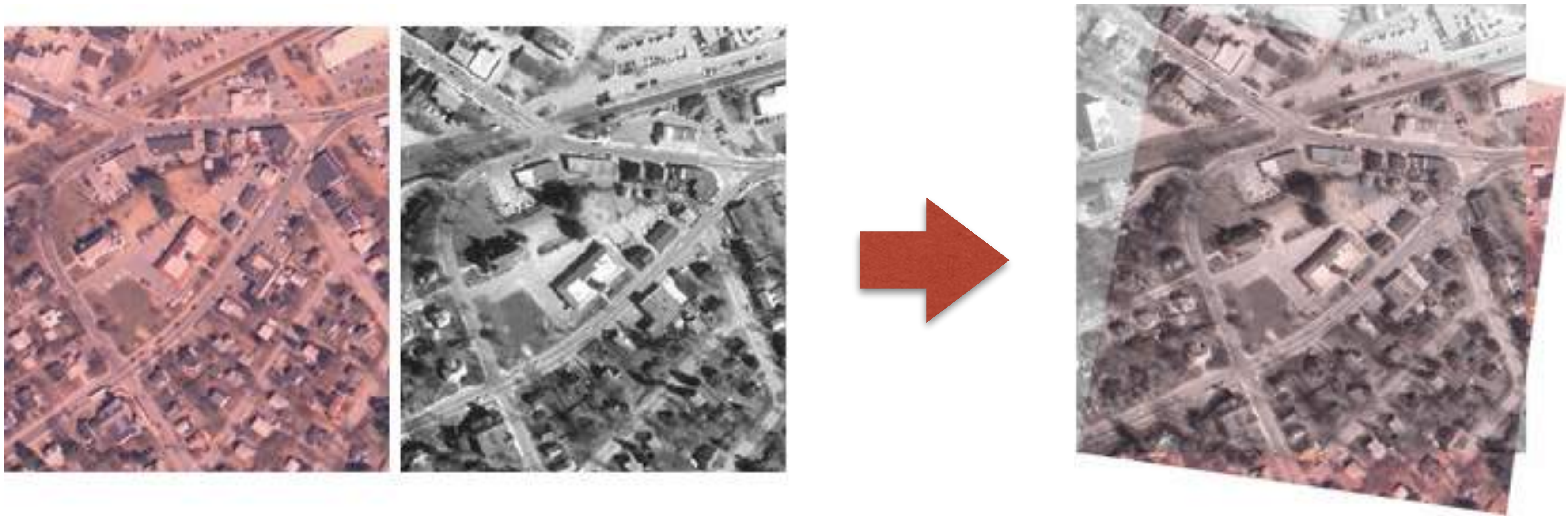


Image registration (2/2)

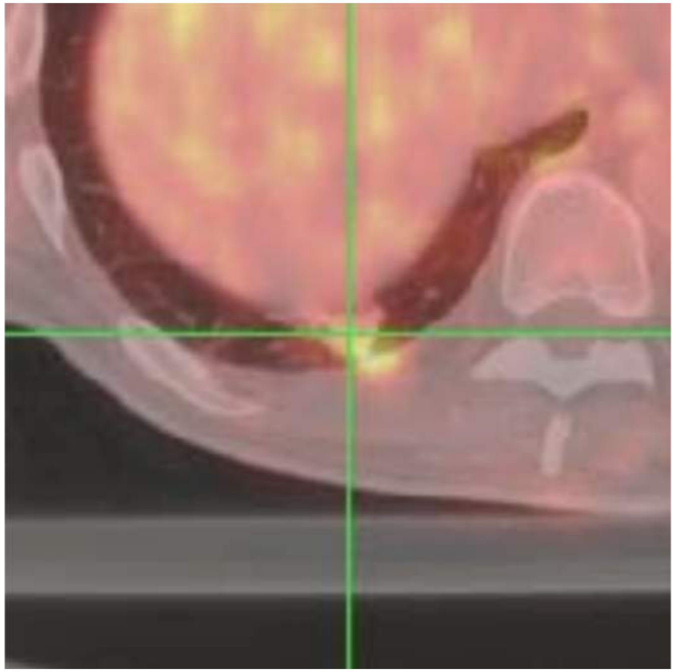
- Basic examples



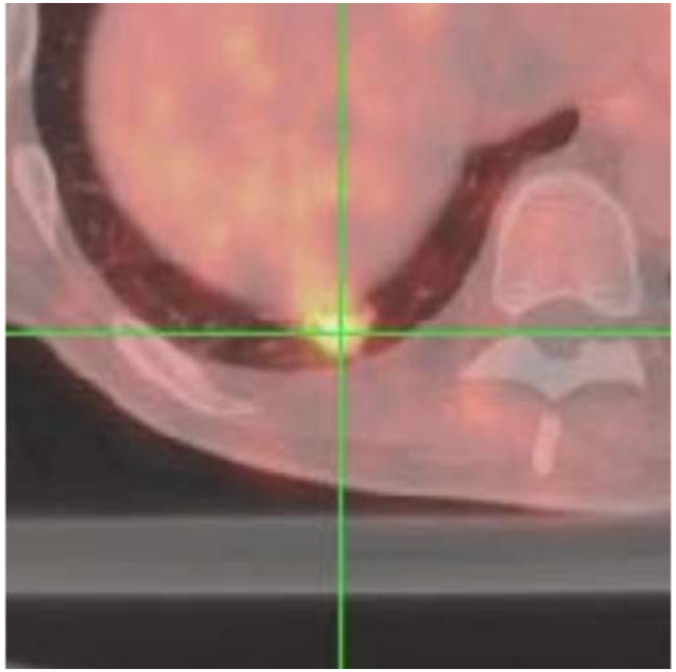
Registration in medical imaging is very important

- Example: PET-MRI registration to study tumor location

Registration algorithm 1



Registration algorithm 2

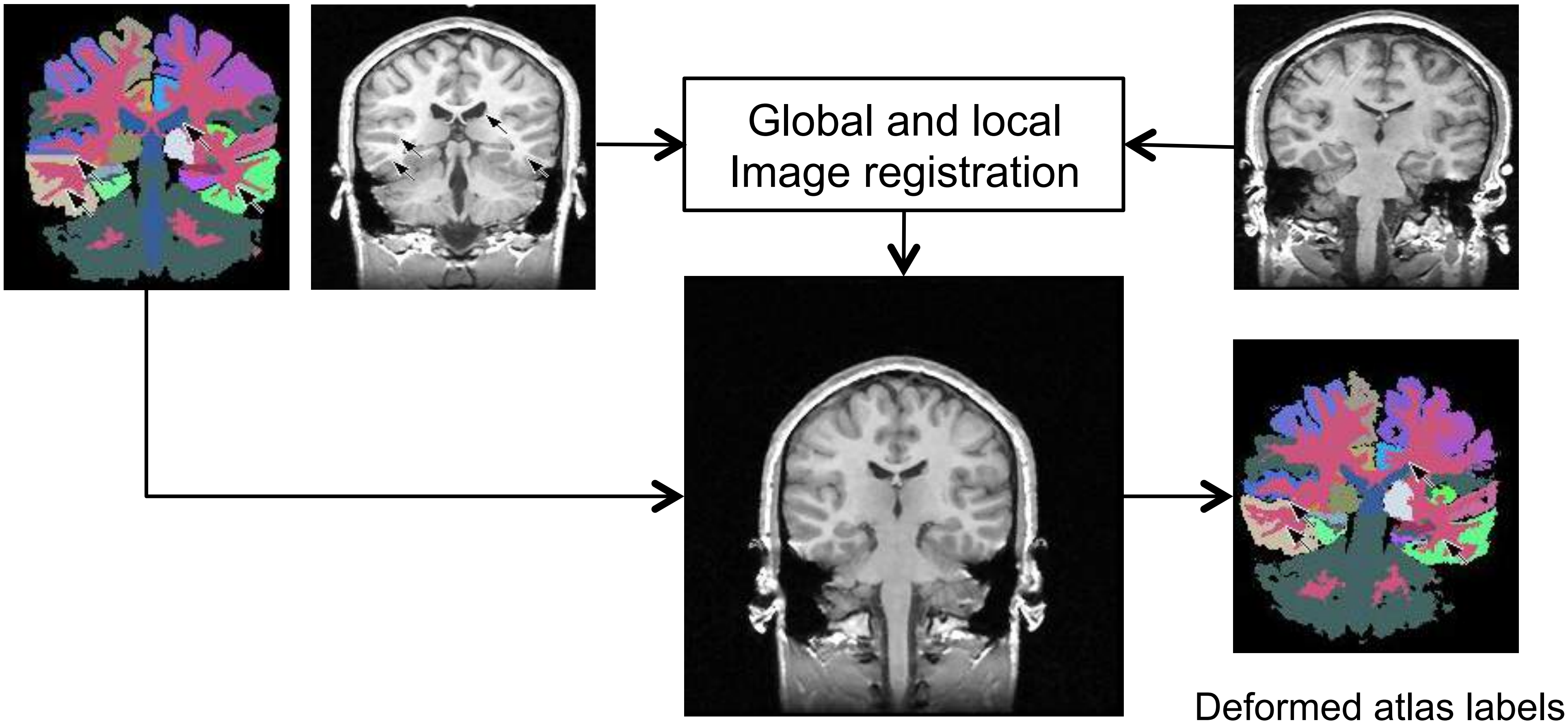


- Is the tumor in the lung only?
- Algorithm #2 looks more plausible:
  are you ready to risk your software against getting sued?

Why do we need to register medical images? (1/7)

Atlas-based **segmentation**

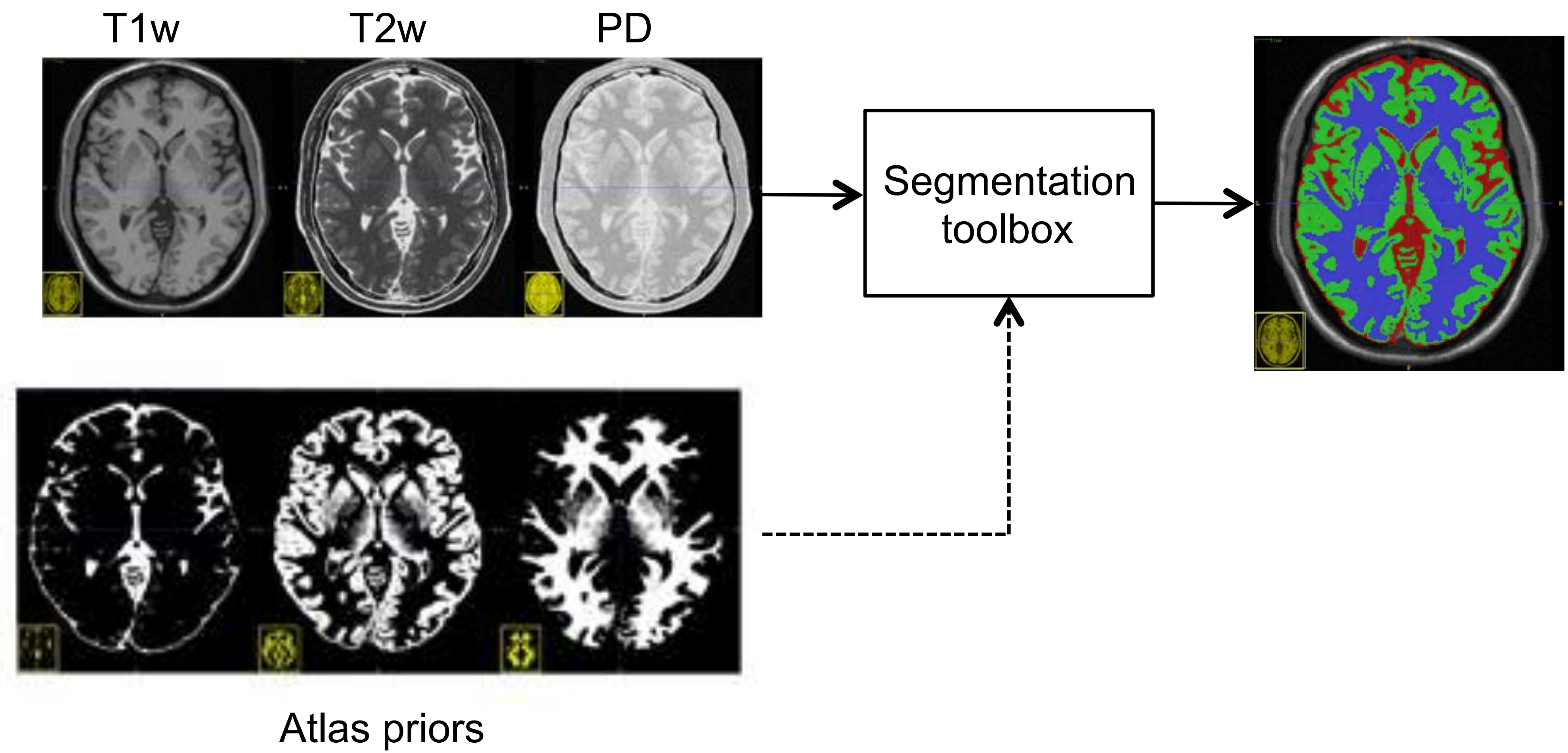
- Use an *accurate atlas* to define one subject's anatomy



(PhD thesis of S. Gorthi @ EPFL)

Multi-spectral **segmentation**

- Use more than one modality to *improve the segmentation* of different structures

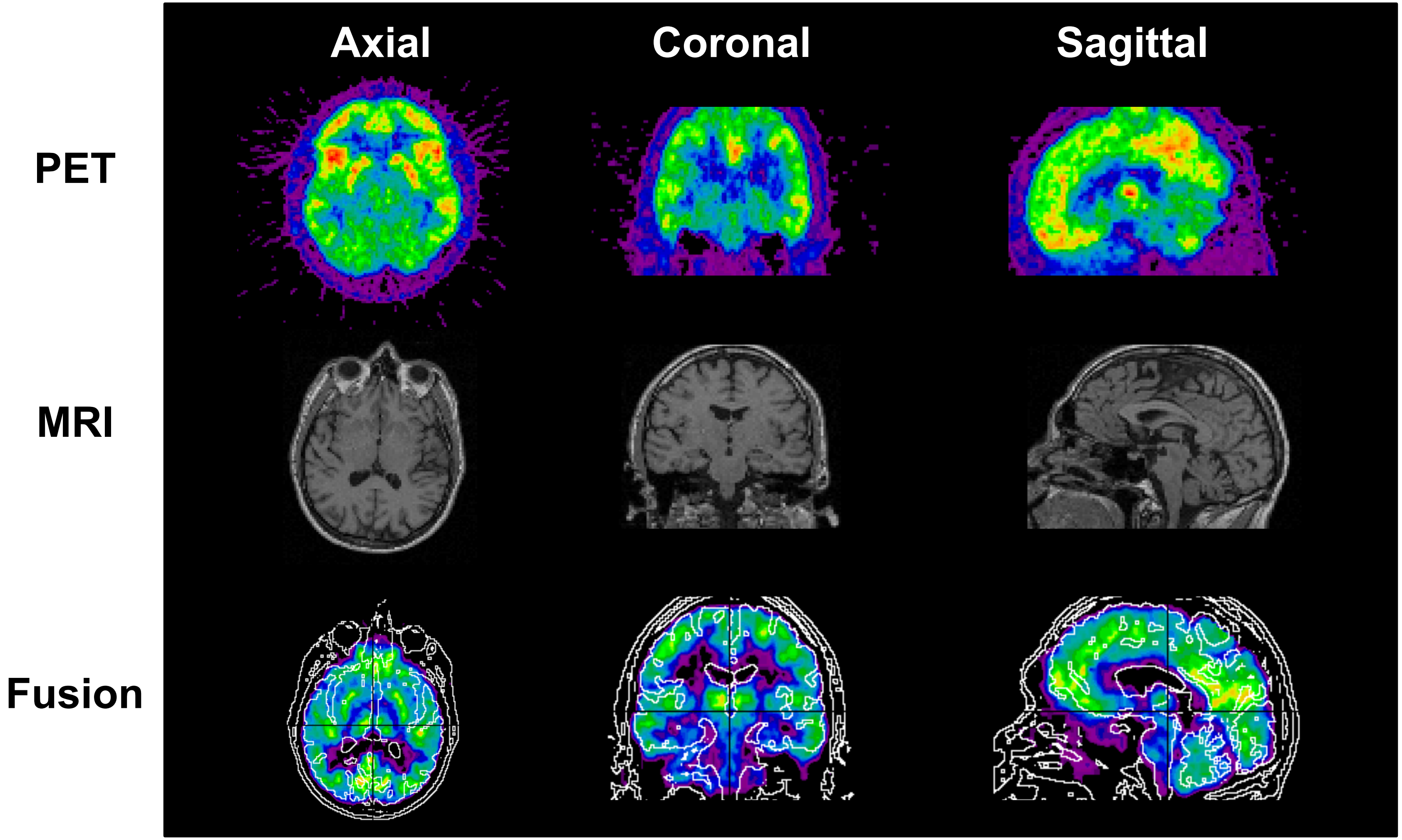


(PhD thesis of O. Esteban @ Madrid+EPFL)

Why do we need to register medical images? (3/7)

Improve diagnosis

- Combining information from *multiple imaging modalities*



Why do we need to register medical images? (4/7)

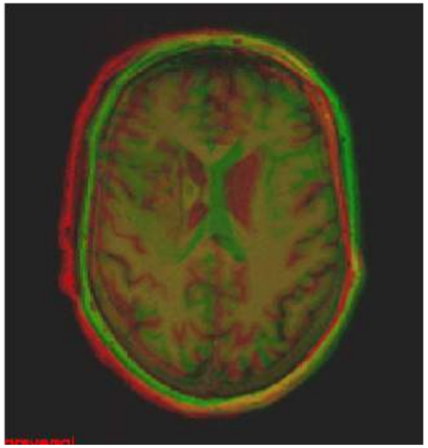
Study **disease progression**

- Monitoring changes in size, shape, position or image intensity over time

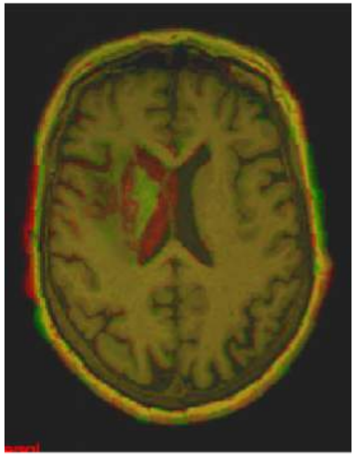
2000 2001 2002

Time 1

Time 2



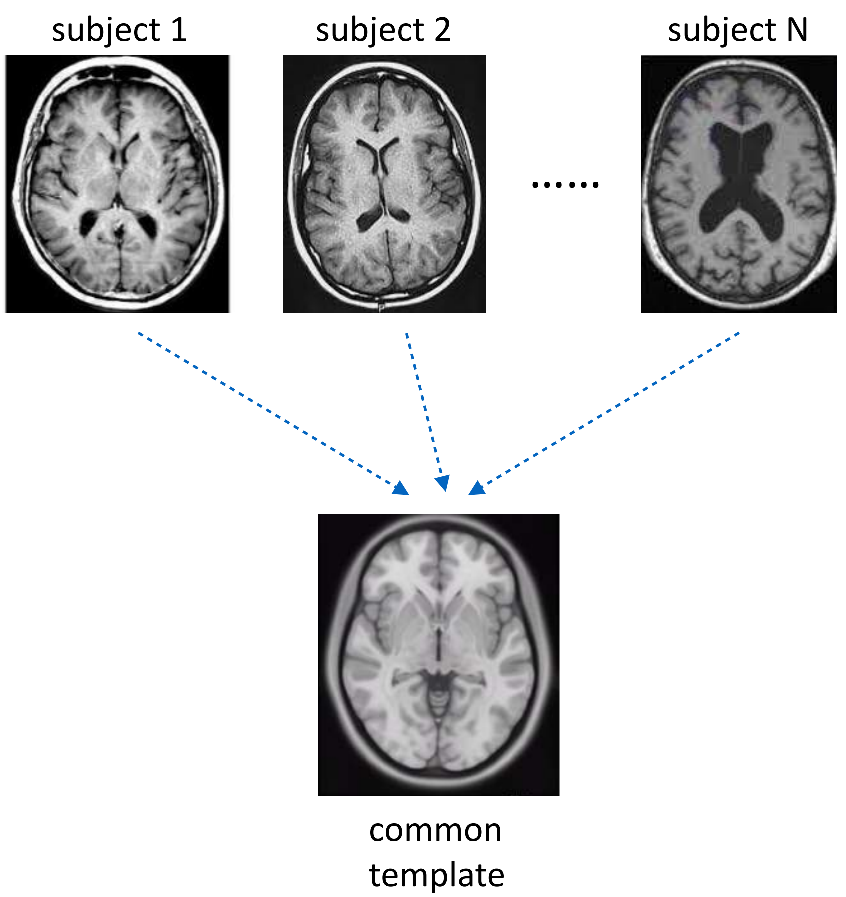
Before registration

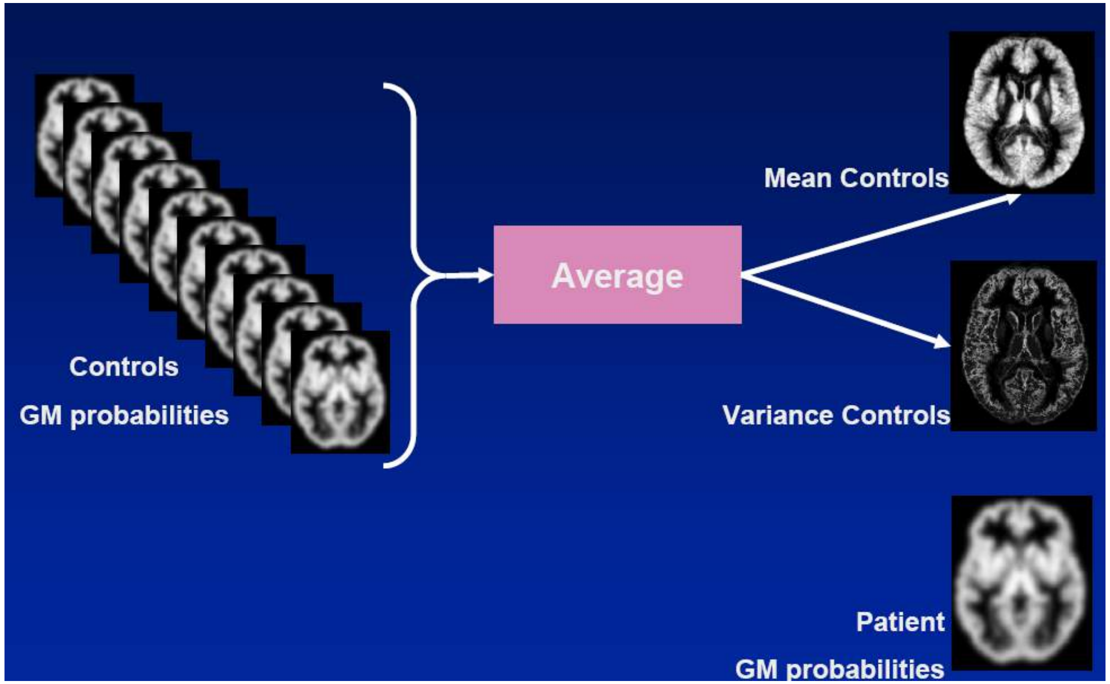


After registration

Population studies, i.e. compare patients vs control subjects

- Relating one individual's anatomy to a standardized atlas or group of subjects

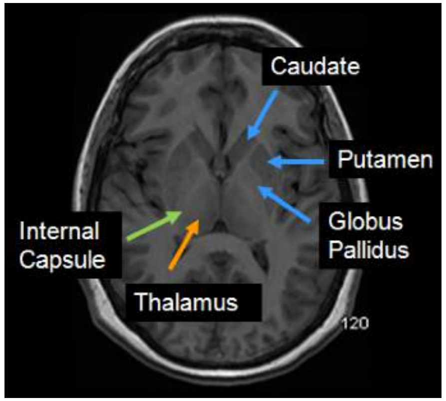


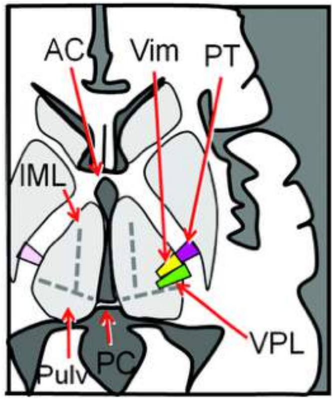


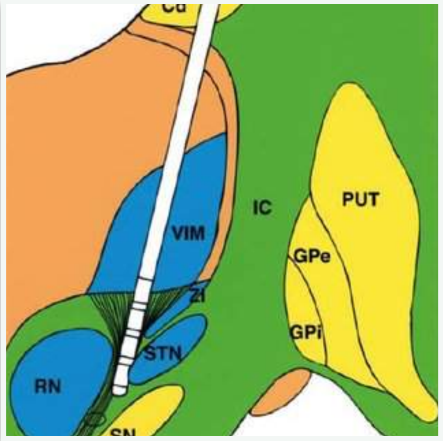
Why do we need to register medical images? (6/7)

Image guided surgery or radiotherapy

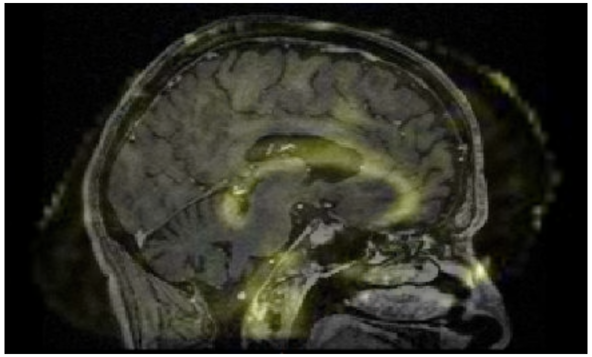
- *VIM targeting* for therapy of movement disorders, e.g. *Parkinson*
  - **T1w** : thalamus segmentation/delineation
  - **DWI** : clustering of thalamus nuclei

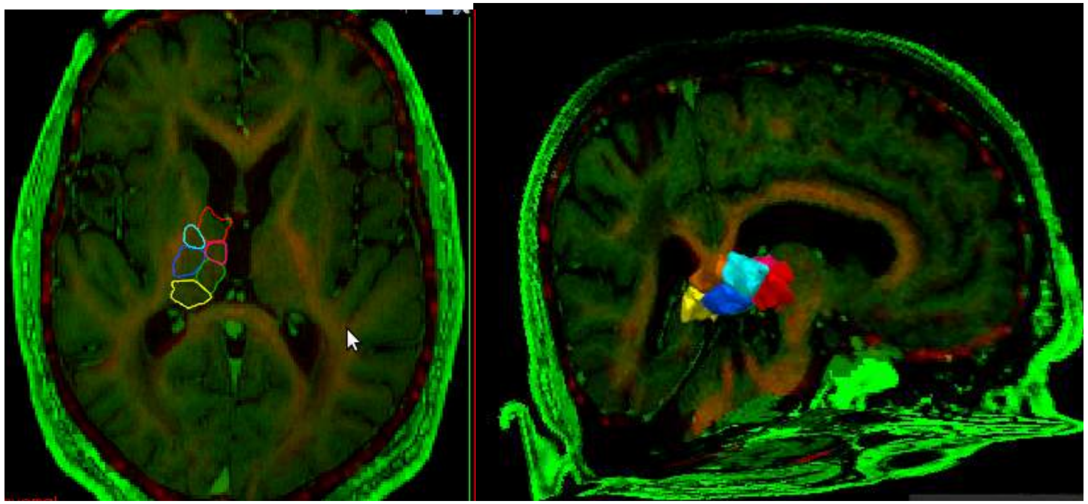






Before registration





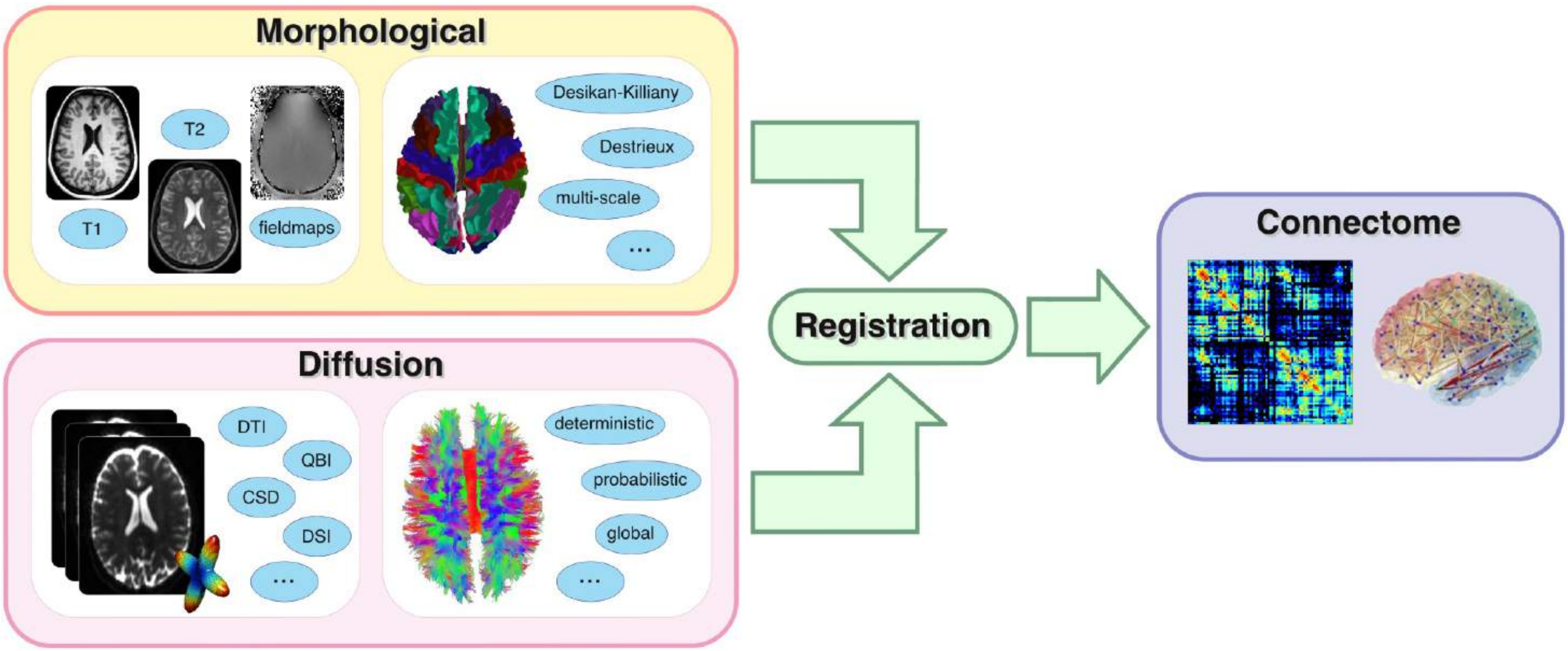
After registration

(PhD thesis of E. Najdenovska @ EPFL)

Why do we need to register medical images? (7/7)

Estimating **brain connectivity** from diffusion MRI

- Estimate *fiber bundles* from diffusion MRI, i.e. DWI
- Define *cortical segmentation* from structural MRI, e.g. T1w



http://hardi.epfl.ch

Many available software packages

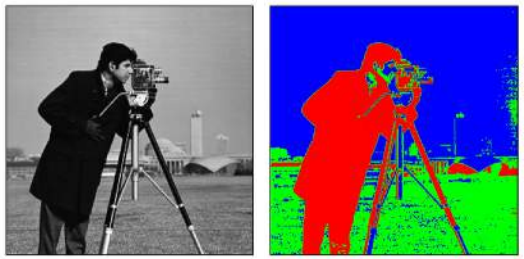
- **ITK.org** (Segmentation & Registration Toolkit)
  - MITK, MedINRIA, Slicer3D, etc
- Elastix
  - **Power of ITK** with **simple interface**
- **F**unctional MRI of the Brain **S**oftware **L**ibrary (FSL)
  - Mostly used for **functional MRI** analyses
- Freesurfer
  - Reconstruction of geometrically accurate models of the gray/white surface
- Statistical **P**arametric **M**apping (SPM)
- **A**dvanced **N**ormalization **T**ools (ANTs)
- ....

Geometric transformations

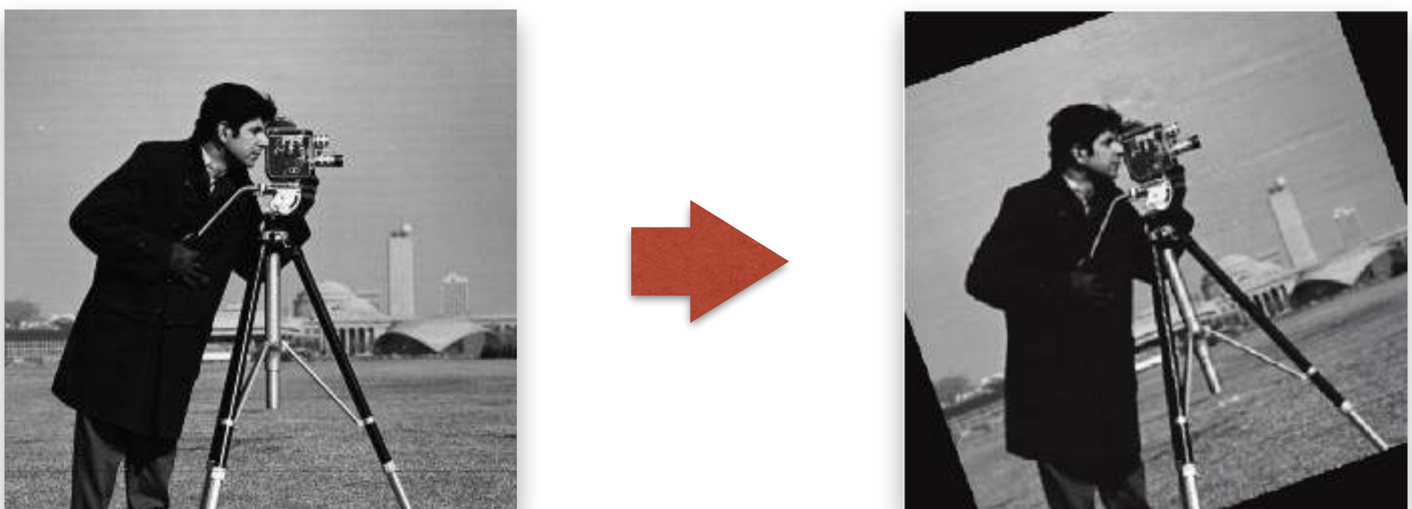
New type of operation on images

- **So far**, the image processing operations we have discussed *modify only the intensities of pixels* in a given image



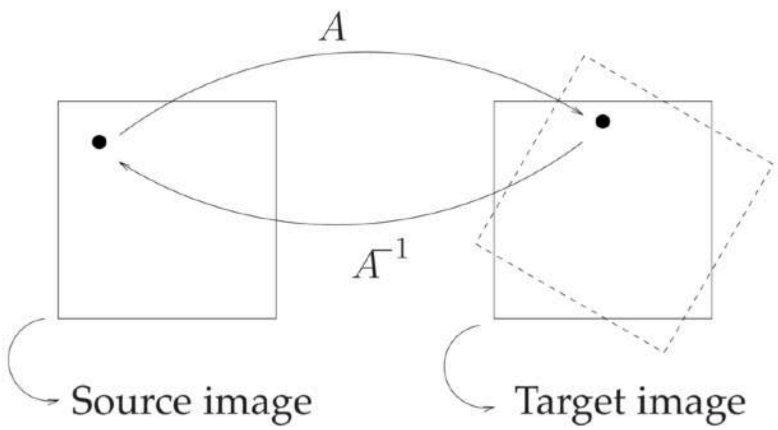


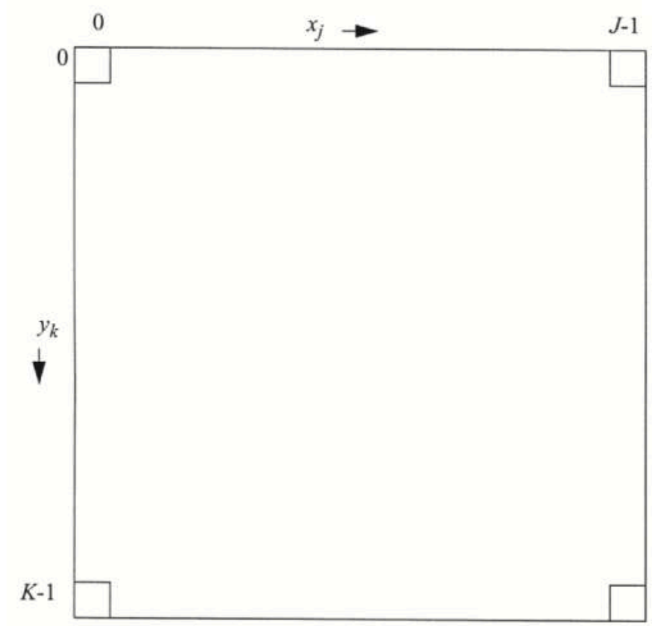
- With geometric transformations, we modify the **positions of the pixels in a image**, but keep their colors "unchanged"



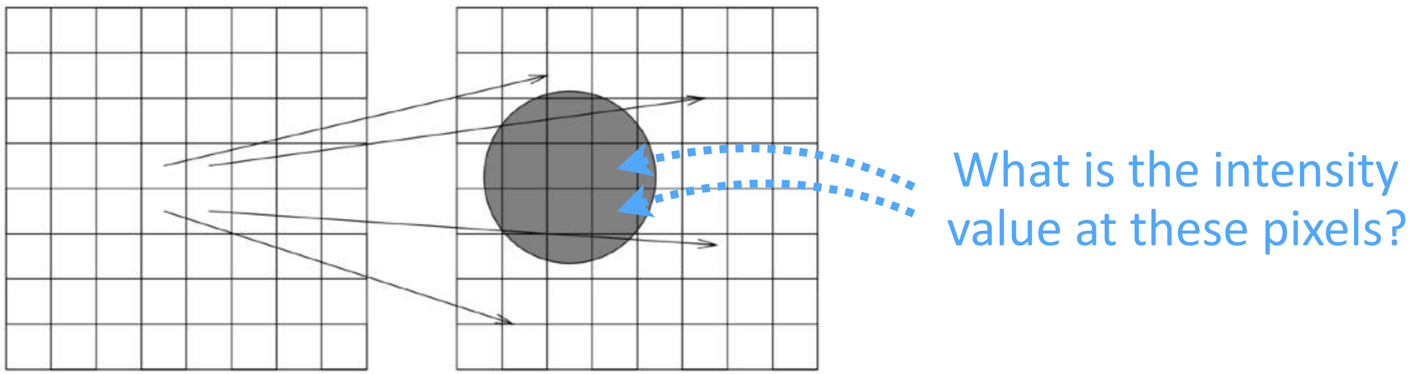
How to define a transformation? (1/2)

- **Forward** vs **inverse** mapping





- **Forward** mapping
  - **For each location** (x,y) **of input image**, compute the spatial location (x',y') of the corresponding pixel in the output image using directly the transformation A
  - **PROBLEM**: some output pixels *may not be assigned* at all

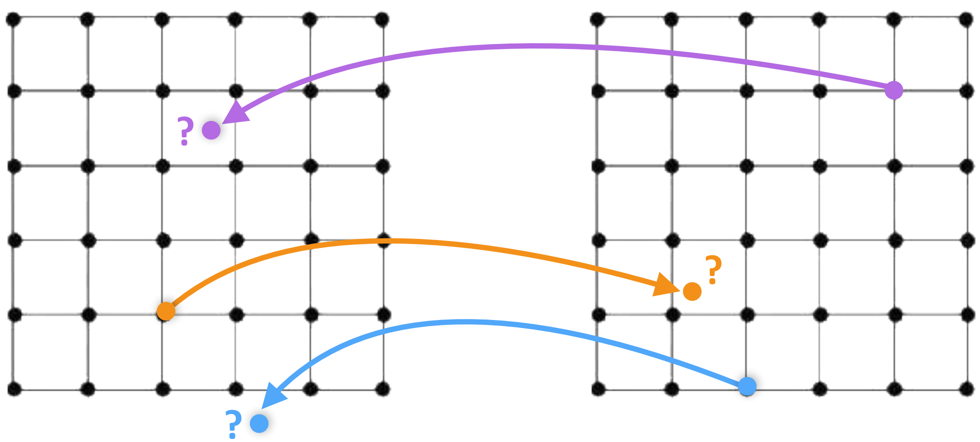


How to define a transformation? (2/2)

Inverse mapping

- **Scans the output pixel locations** and, at each location, (x',y'), computes the corresponding location in the input image using $(x,y) = \mathbf{T}^{-1}(x',y')$
- By using inverse mapping, the previous problem vanishes
  (NB: MATLAB uses this convention)

NB: some coordinates may be mapped (i) **between discrete pixel locations** or (ii) **outside** the corresponding image pixels



- **Interpolation** among the nearest input pixels and **extrapolation** are needed to determine the intensity of the output pixel value (see later in the presentation)

Scaling, translation and rotation (1/2)

- **Scaling** of point $\mathbf{p}=[x\ y]^\top$ by factors s_x *and* s_y

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

- **Translation** of point $\mathbf{p}=[x\ y]^\top$ by vector $\mathbf{T}=[t_x\ t_y]^\top$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x + t_x \\ y + t_y \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \boxed{\begin{bmatrix} t_x \\ t_y \end{bmatrix}}$$

- **Rotation** of point $\mathbf{p}=[x\ y]^\top$ by angle θ is given by

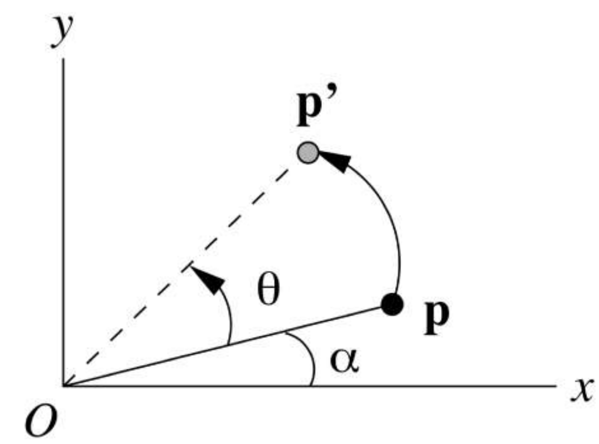
$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \boxed{\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}} \begin{bmatrix} x \\ y \end{bmatrix}$$

Scaling, translation and rotation (2/2)

- **NB**: rotation is normally performed **about the origin**

- Derivation of *rotation matrix*
  - Let ρ denote the magnitude of the vector $\mathbf{p}=[x\ y]^\top$

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \rho\cos\alpha \\ \rho\sin\alpha \end{bmatrix}$$



  - After rotating by θ, point $\mathbf{p}$ becomes $\mathbf{p'}=[x'\ y']^\top$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \rho\cos(\alpha+\theta) \\ \rho\sin(\alpha+\theta) \end{bmatrix} = \begin{bmatrix} \rho\,(\cos\alpha\cos\theta - \sin\alpha\sin\theta) \\ \rho\,(\sin\alpha\cos\theta + \cos\alpha\sin\theta) \end{bmatrix}$$

$$= \begin{bmatrix} x\cos\theta - y\sin\theta \\ x\sin\theta + y\cos\theta \end{bmatrix}$$

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

Rigid transformation

- We can easily **combine translation and rotation** in one formula

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \boxed{\begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix}} \begin{bmatrix} x \\ y \end{bmatrix} + \boxed{\begin{bmatrix} t_x \\ t_y \end{bmatrix}}$$

  - This transformation is called **rigid**
  - For it allows moving around (*translation+rotation*) a "**rigid body**"

- Can be expressed as a **single transformation matrix**

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \underbrace{\begin{matrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{matrix}}_{\text{rotation}} & \underbrace{\begin{matrix} t_x \\ t_y \end{matrix}}_{\text{translation}} \\ \begin{matrix} 0 & \quad 0 \end{matrix} & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- This *compact representation* was only possible because of the **homogeneous coordinates**

Homogeneous coordinates (1/2)

- **Homogeneous coordinates** of the 2D point

$$\mathbf{p} = \begin{bmatrix} x \\ y \end{bmatrix}$$

are

$$\begin{bmatrix} cx \\ cy \\ c \end{bmatrix}$$

for any *non-zero scalar* c.

- The 2D vector $\mathbf{p}$ becomes a 3D vector
  - Homogeneous coordinates apply to 3D points as well (by adding a 4th component)

- Given a point $[x\ y\ z]^\top$ in **homogeneous coordinates**, its 2D **Cartesian coordinates** are $[x/z\ y/z]^\top$
  - If $z = 0$, then this is a point at infinity

Homogeneous coordinates (2/2)

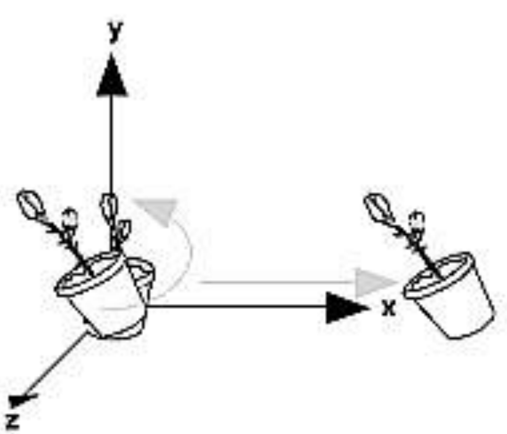
- Real power of homogeneous coordinates is that they provide the framework to **concatenate a sequence of transformations**

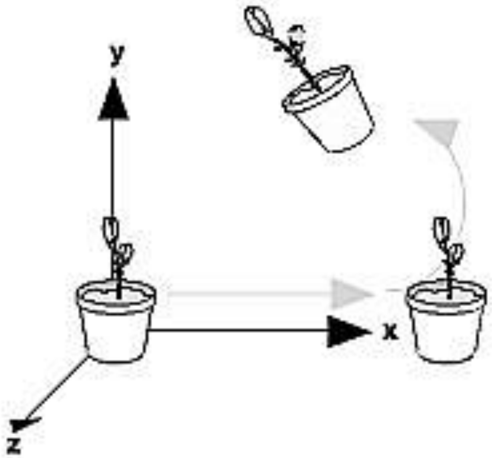
$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} s & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} s\cos\theta & -s\sin\theta & t_x \\ s\sin\theta & s\cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- **NB**: the **order of the transformations** is important!

rotate
then
translate





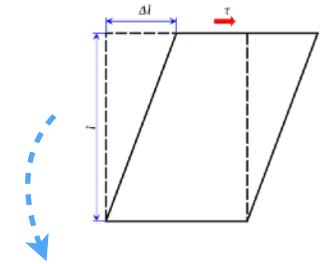
translate
then
rotate

Affine transformation

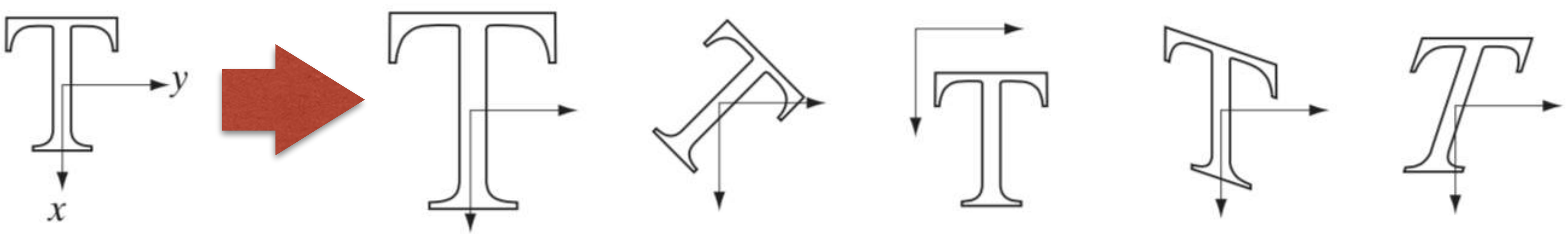
- Affine transform is a **generalization of rigid transformation**

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

  - Now the constants a_{ij} can be any number
  - **Most general** linear transformation

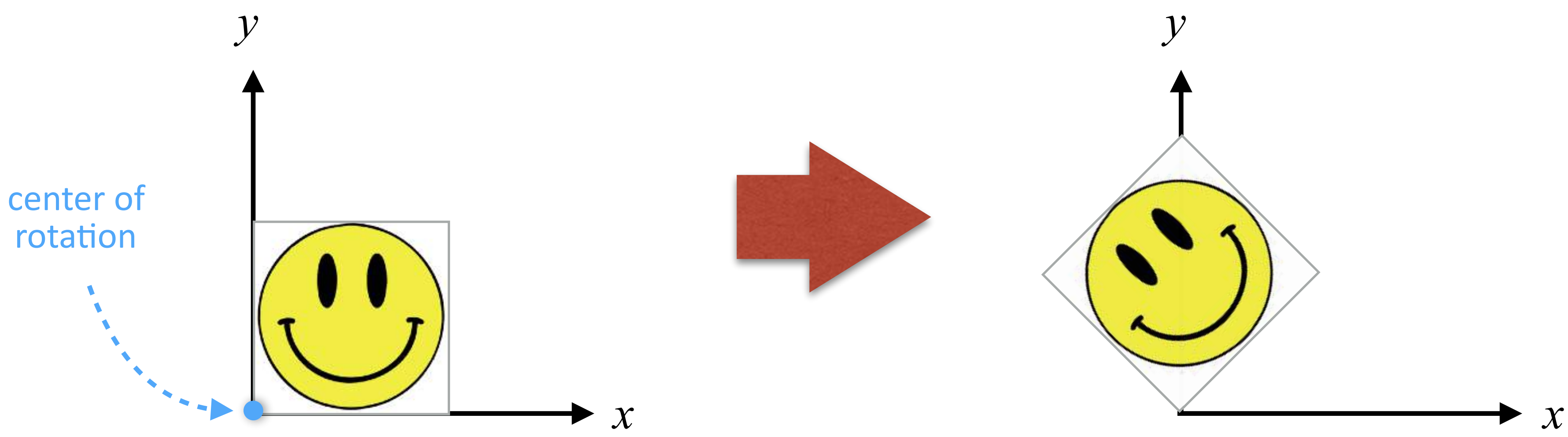


- Affine transformation can **scale**, **rotate**, **translate** and **shear** a set of coordinate points, depending on values in matrix $\mathbf{T}$

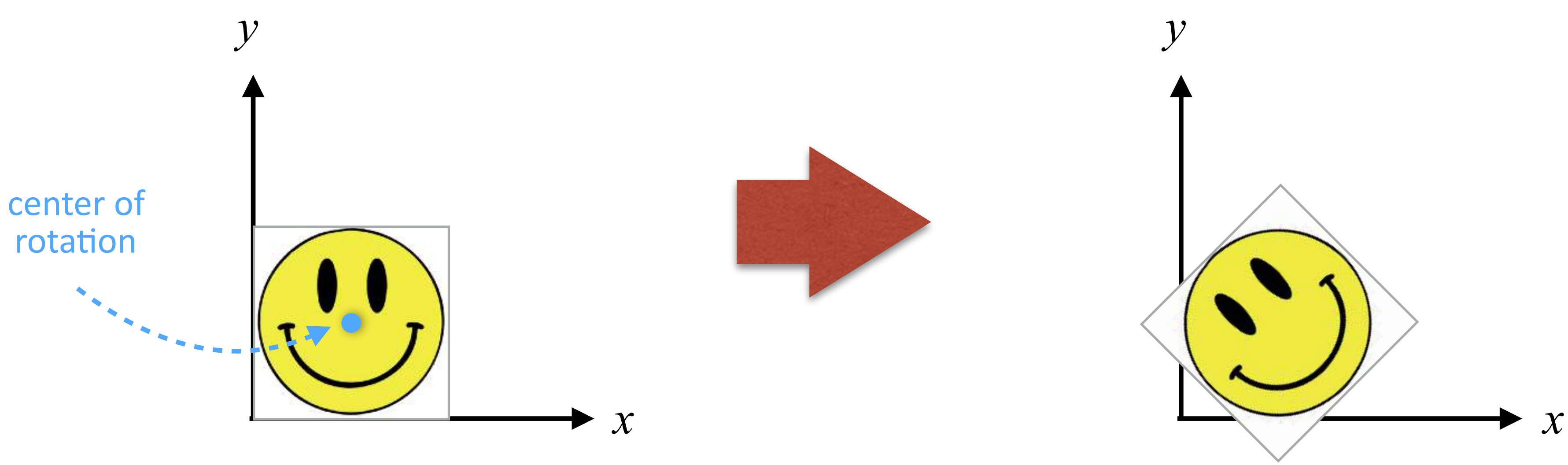


Rotation around its center

- How can we rotate an image by angle θ around its center?



- What we want is this:



Rotation around its center

- **Solution**: concatenate three transformations

1) **Translate** the image so that its center is at the coordinate (0,0)

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} -N_x/2 \\ -N_y/2 \end{bmatrix}$$

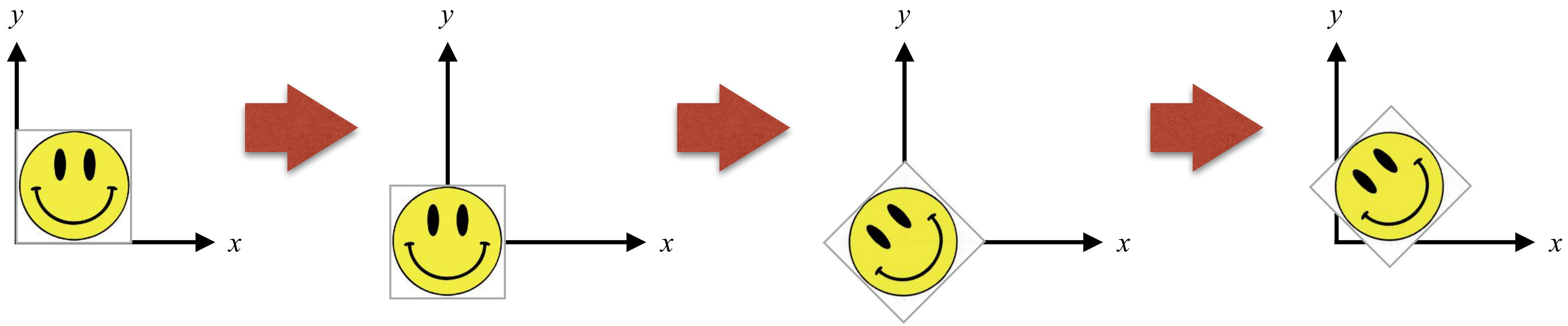
N_x and N_y are the dimensions of the image

2) **Rotate** it by angle θ as usual

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

3) **Translate back** to its original position

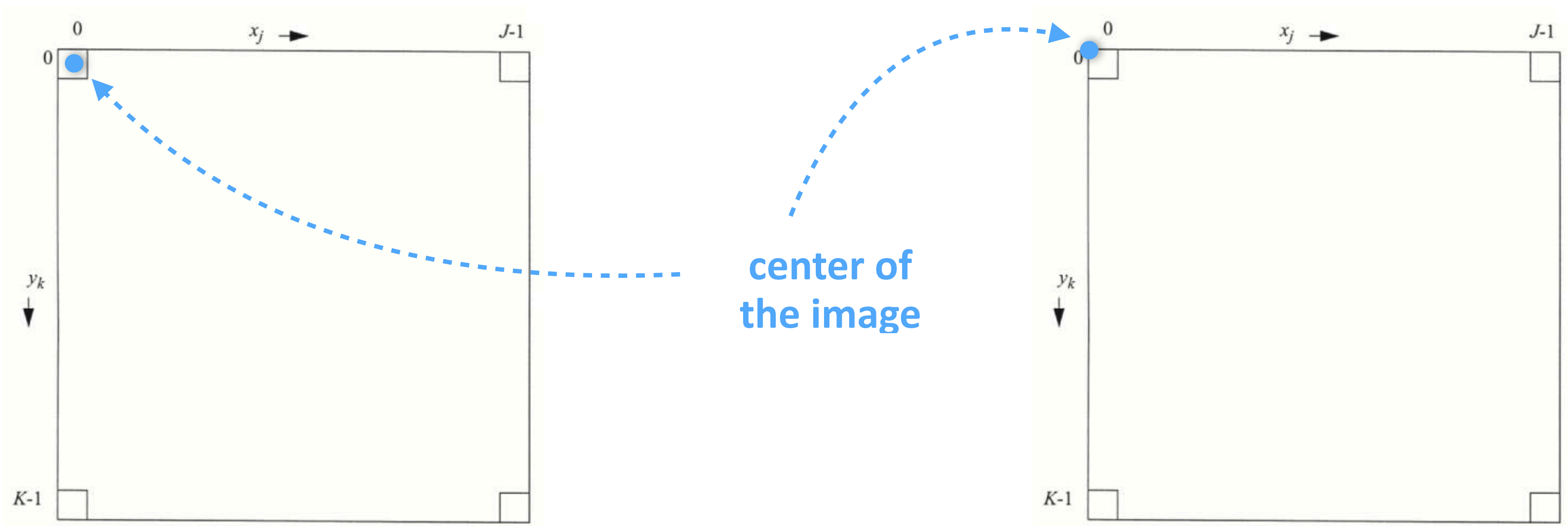
$$\begin{bmatrix} x''' \\ y''' \end{bmatrix} = \begin{bmatrix} x'' \\ y'' \end{bmatrix} + \begin{bmatrix} N_x/2 \\ N_y/2 \end{bmatrix}$$



Rotation around its center (3/3)

- Using **homogeneous coordinates** we can rewrite the previous expression using a single matrix. How?

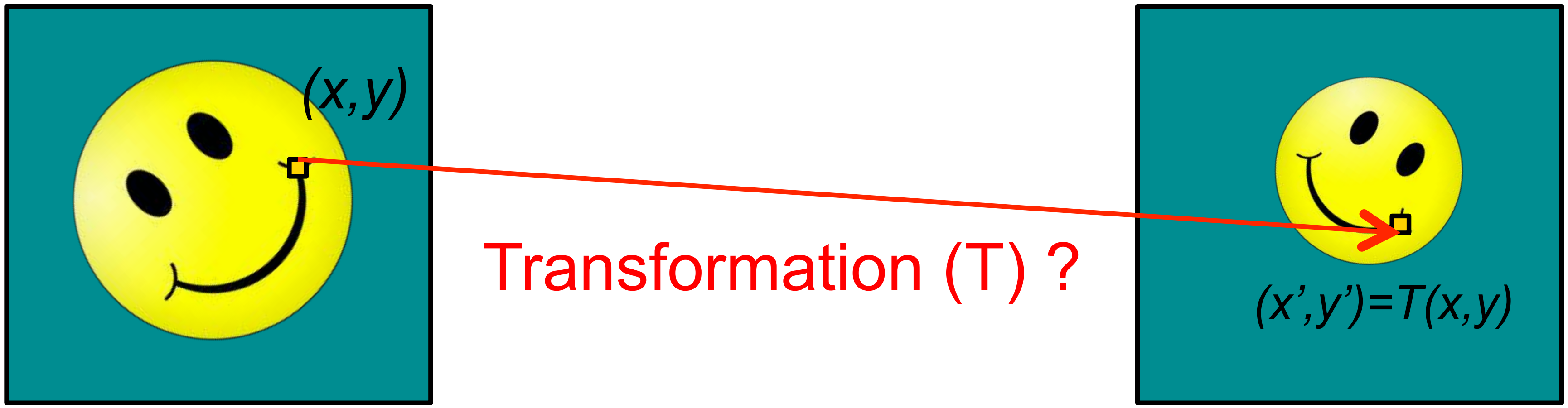
- This can be useful to deal with different conventions for the **coordinate of the center of the image**
  - Center of top-left corner voxel?
  - Top-left of it?



Registration framework

Registration is an inverse problem

- "…*find the spatial transformation* that maps points from one image B to the corresponding points in another image A…"



Registration is an inverse problem

- "*...find the spatial transformation* that maps points from one image B to the corresponding points in another image A..."
- Usually solved as **energy minimization problem** (or maximization)

optimal transformation

$$\mathcal{T}^* = \underset{\mathcal{T} \in \mathbb{T}}{\operatorname{argmin}}\, d(A, B^{\mathcal{T}})$$

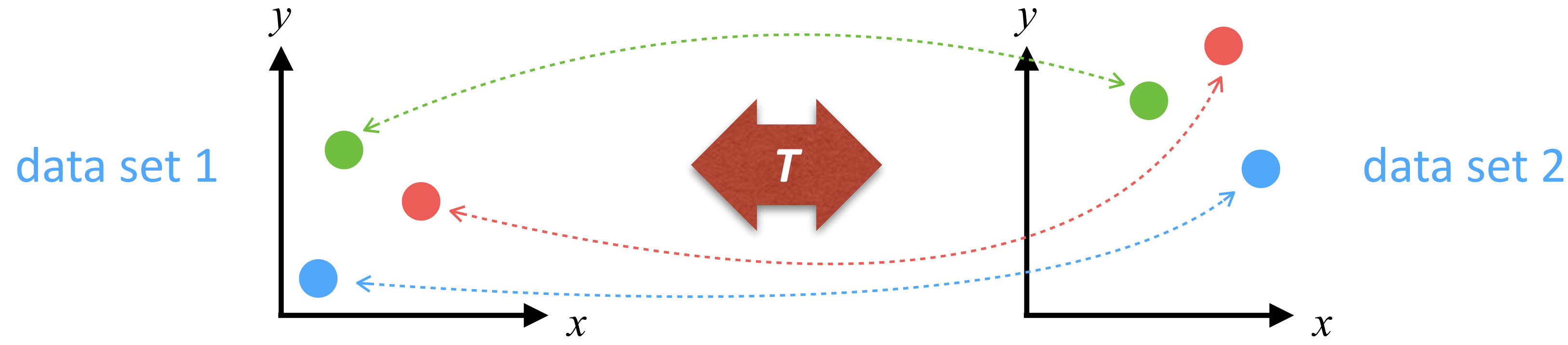
images

space of all possible transformations

similarity/dissimilarity between the images

- Notation
  - $A : \mathbf{x}_A \in \Omega_A \mapsto A(\mathbf{x_A})$ Intensity of image A at location $\mathbf{x}$
  - $\mathbf{T} : \mathbf{x}_B \mapsto \mathbf{x}_A \iff \mathbf{T}(\mathbf{x}_B) = \mathbf{x}_A$ Transforms a position $\mathbf{x}$ from one image to another
  - $\mathcal{T}$ Transforms an image (both coordinates $\mathbf{x}$ and intensities)
  - $B^{\mathcal{T}}$ Image B transformed
  - $\Omega^{\mathcal{T}}_{A,B} = \{\mathbf{x}_A \in \Omega_A | \mathbf{T}^{-1}(\mathbf{x}_A) \in \Omega_B\}$ Overlap domain after transformation $\mathbf{T}$

- Let's imagine that:
  - We have **two clouds of corresponding points**



  - Corresponding points are related via an (unknown) **affine transformation *T***

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- **Find** the transformation ***T***

- NB: affine transformation has **6 parameters** (in 2D)
  - Need **at least 3 pairs** of corresponding points
  - *Use more* to obtain *more robust* estimates of the parameters

Simple example (2/4)

- The n pairs of corresponding points are related by

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad \Rightarrow \quad \begin{aligned} x'_i &= a_{11}\,x_i + a_{12}\,y_i + a_{13} \\ y'_i &= a_{21}\,x_i + a_{22}\,y_i + a_{23} \end{aligned} \qquad \text{for } i=1,\ldots,n$$

- We have **two sets of linear equations** of the form $\mathbf{Ma=b}$:

first set
$$\begin{bmatrix} x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots \\ x_n & y_n & 1 \end{bmatrix} \begin{bmatrix} a_{11} \\ a_{12} \\ a_{13} \end{bmatrix} = \begin{bmatrix} x'_1 \\ \vdots \\ x'_n \end{bmatrix}$$

$$\begin{bmatrix} x_1 & y_1 & 1 \\ \vdots & \vdots & \vdots \\ x_n & y_n & 1 \end{bmatrix} \begin{bmatrix} a_{21} \\ a_{22} \\ a_{23} \end{bmatrix} = \begin{bmatrix} y'_1 \\ \vdots \\ y'_n \end{bmatrix}$$
second set

- Notes
  - Number of equations > number of unknowns ➜ *no exact solution* (i.e. *overdetermined system*)
    - Can compute **best fitting** a_{ij} , i.e. $\min \|\mathbf{Ma-b}\|_2$, for each set independently
    - Use **linear least squares** to compute this approximation
  - Number of equations > number of unknowns ➜ $\mathbf{M}$ *is not square*
    - $\mathbf{M}$ has no *inverse* (i.e. $\mathbf{M}^{-1}$)
    - How to cope with this?

$$\cancel{\mathbf{Ma=b} \Rightarrow \mathbf{a=M^{-1}b}}$$

- **NB**: $\mathbf{M}$ is not square and has no inverse…
  - …but $\mathbf{M}^T\mathbf{M}$ is *square* and (typically) *has inverse*

$$\cancel{\mathbf{Ma}=\mathbf{b} \Rightarrow \mathbf{a}=\mathbf{M}^{-1}\mathbf{b}}$$

- **Pseudo-inverse of $\mathbf{M}$**

$$\mathbf{M}\,\mathbf{a} = \mathbf{b}$$

$$\begin{aligned} \mathbf{M}^\top\mathbf{M}\,\mathbf{a} &= \mathbf{M}^\top\mathbf{b} \\ \mathbf{a} &= \underbrace{\boxed{(\mathbf{M}^\top\mathbf{M})^{-1}\mathbf{M}^\top}}_{\text{pseudo-inverse of } \mathbf{M}}\mathbf{b} \end{aligned}$$

  - *Pseudo-inverse* gives the **least squared error solution**

- Notes
  - Pseudo-inverse can be a **very large matrix**: don't use it directly for large problems
  - MATLAB provides *optimized functions* for computing the least square solution of such linear problems

Simple example (4/4)

- **Alternative method**: put the x' and y' parts in the same matrix

$$\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 \\ & & \vdots & & & \\ x_n & y_n & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_1 & y_1 & 1 \\ & & \vdots & & & \\ 0 & 0 & 0 & x_n & y_n & 1 \end{bmatrix} \begin{bmatrix} a_{21} \\ a_{22} \\ a_{23} \\ a_{21} \\ a_{22} \\ a_{23} \end{bmatrix} = \begin{bmatrix} x'_1 \\ \vdots \\ x'_n \\ y'_1 \\ \vdots \\ y'_n \end{bmatrix}$$

  - Solve as before
  - **NB**: x' and y' parts are *still independent of each other*

- **Typical mistake**: *wrong way* to form the linear system

$$\begin{bmatrix} x_1 & y_1 & 1 & x_1 & y_1 & 1 \\ & & \vdots & & & \\ x_n & y_n & 1 & x_n & y_n & 1 \end{bmatrix} \begin{bmatrix} a_{11} \\ a_{12} \\ a_{13} \\ a_{21} \\ a_{22} \\ a_{23} \end{bmatrix} = \begin{bmatrix} x'_1 + y'_1 \\ x'_2 + y'_2 \\ \vdots \\ x'_n + y'_n \end{bmatrix}$$

  - **6 unknowns**, but only **3 independent columns** in the matrix

General framework

Typical algorithm

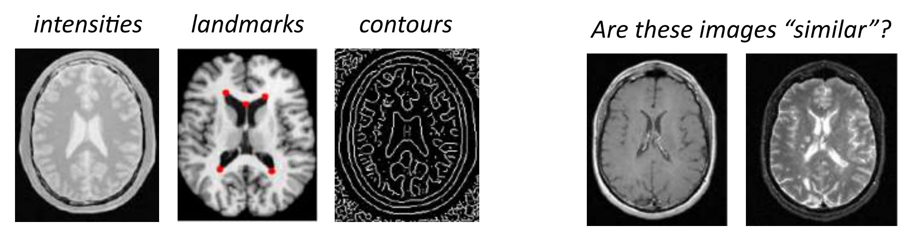
Target Image → Feature → Cost function/Metric → Optimizer

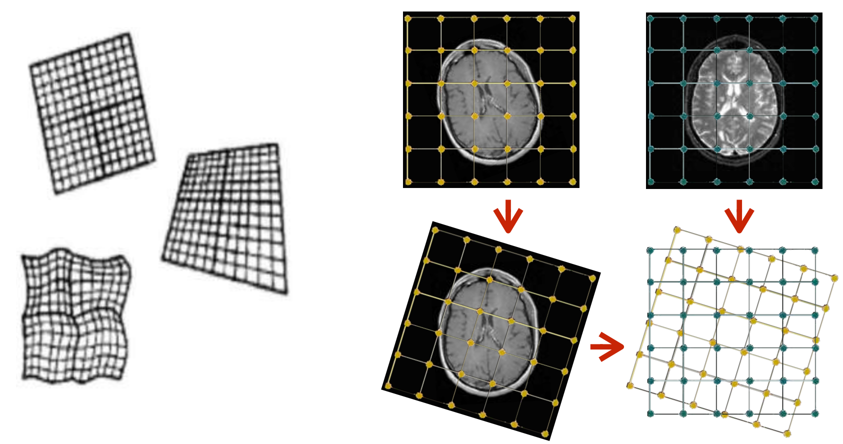
Source Image → Feature → Interpolator ← Transform

Interpolator → Cost function/Metric; Optimizer → Transform

The main actors

- **Feature**
  - Which information to use for driving the registration
- **Similarity metric**
  - Measures of how similar the features are in the two images
- **Interpolator**
  - How to compute similarity metrics from different grids
- **Transform**
  - The deformation model to transform one image into another
- **Optimizer**
  - The optimization algorithm to estimate the transformation

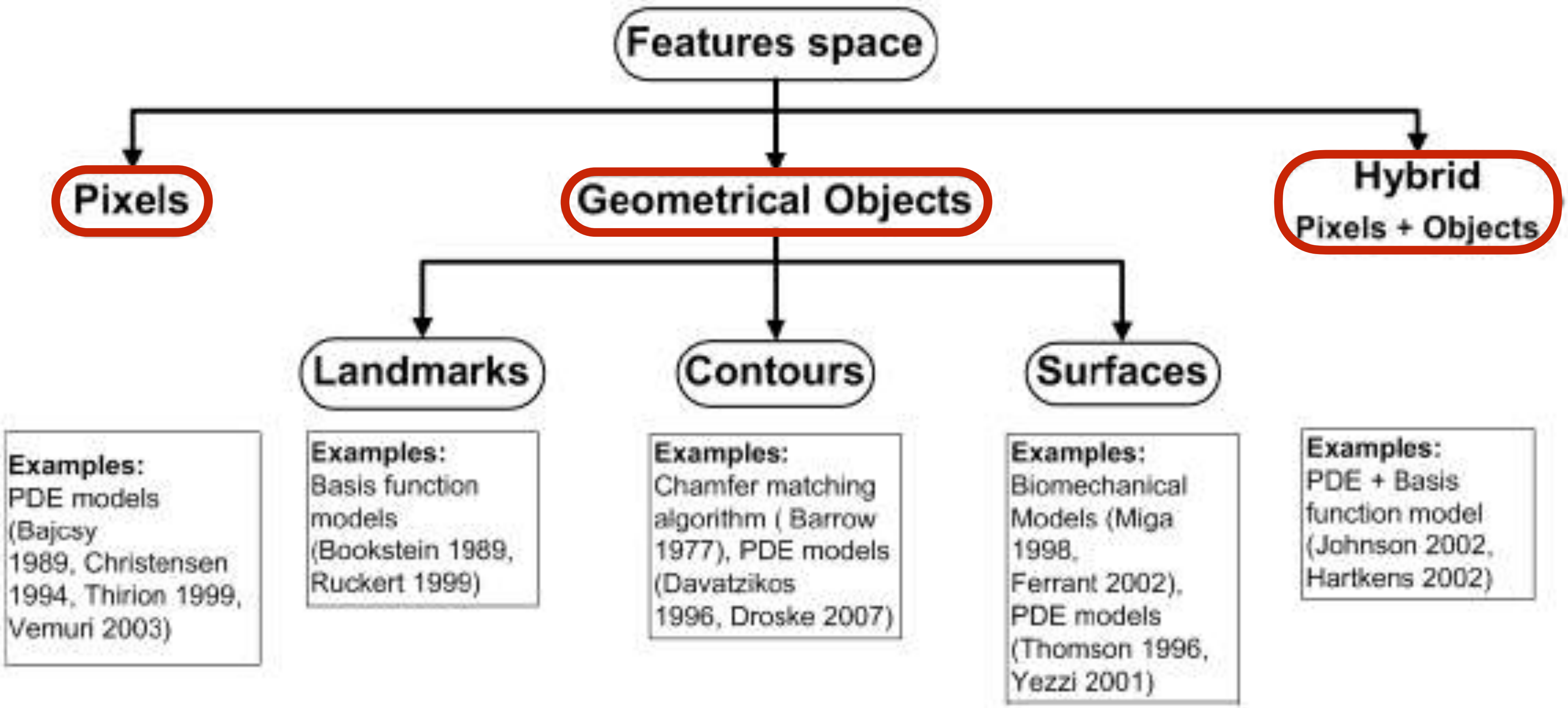




I - Features of interest (1/3)

Two main approaches

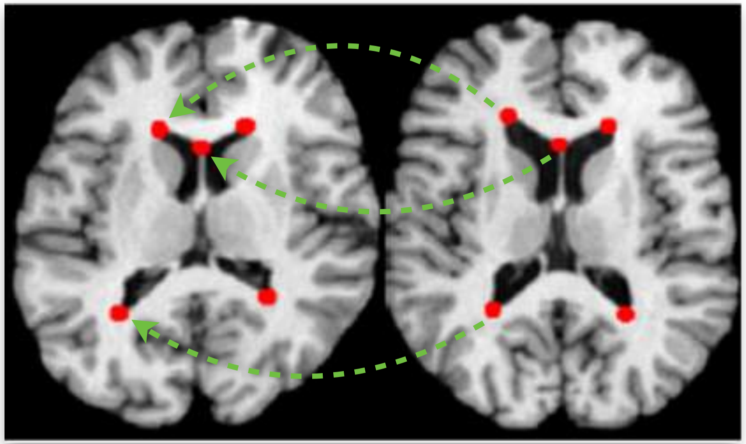
- **Feature based**: use corresponding points or features in the images to align them
- **Intensity based**: operate directly on the image intensities

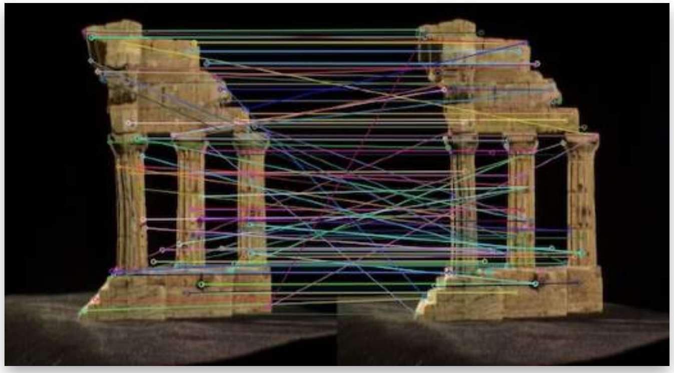


(PhD thesis of V. Duay @ EPFL)

<u>**Feature based** approach</u>

- **Extract specific features** from both images
  - e.g. points representing "fiducial markers"
  - Internal *anatomical structures*, e.g. anterior commissure
  - Pins/markers fixed to the patient, e.g. skin markers
- **Match pairs** of corresponding points
  - Either *manually* or *automatically*
- **Compute parameters of transformation $\mathbf{T}$** between corresponding points
  - Assume all pairs of points are related by *same transformation*
  - Minimize some "*measure of distance*" between them





Important to match accurately the points

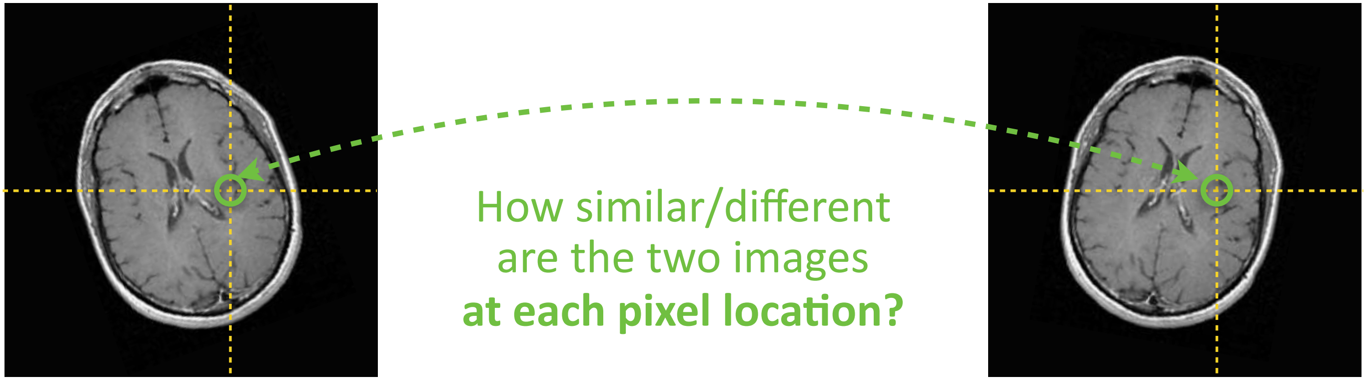
- But algorithms exist to cope with errors, e.g. *RANSAC*

Can be **extended** to other features

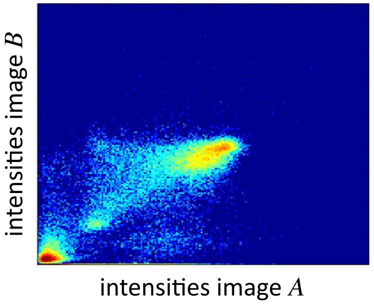
- e.g. edges, *contours, surfaces* etc…
- **Critical**: define suitable *similarity metric* for that feature

Intensity based approach

- Compare *directly the intensities* at each pixel location in the two images



- Transformation $\mathbf{T}$ computed by *comparing intensity patterns* in both images via "**pixel similarity metrics**"
- These are based on the **joint histogram**
- **No need to delineate** corresponding structures
  - Sort of having "features = pixels"

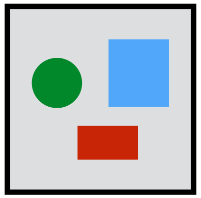
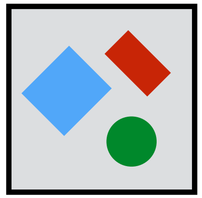


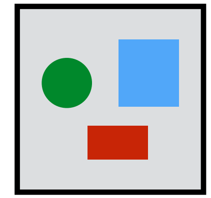
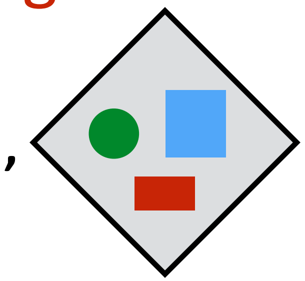
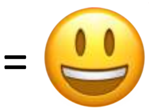
It's the **most used approach** in medical imaging

NB: we will focus on this approach in this course

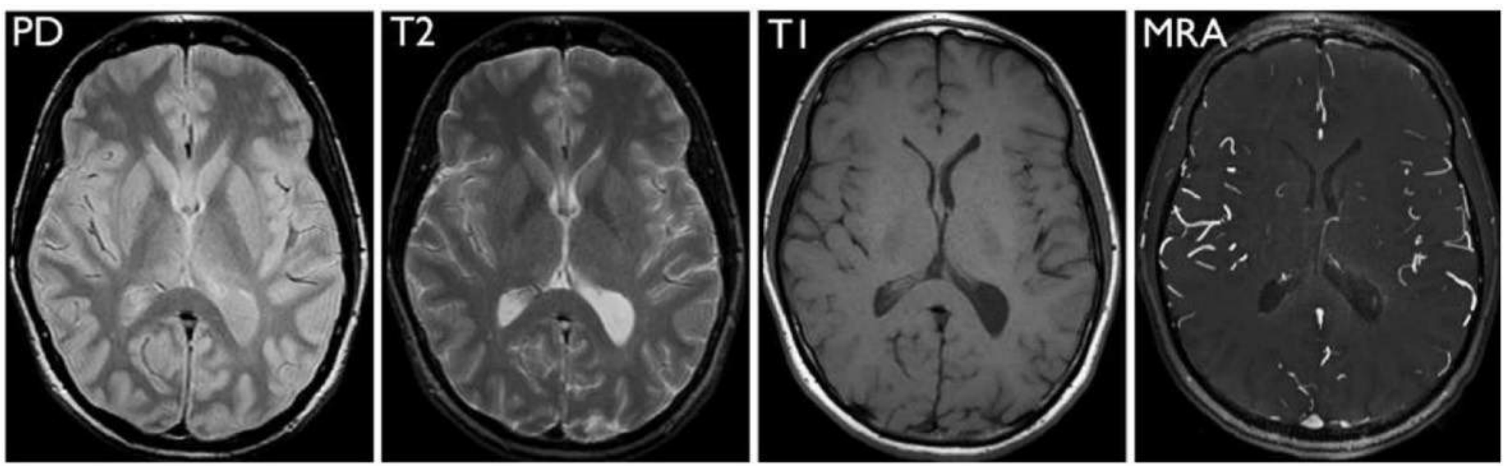
II - Similarity measures (1/5)

- Quantify **degree of similarity** between two images

similarity( , ) = 

similarity( , ) = 

- Example
  - Same subject/session, but *images from different modalities look different*

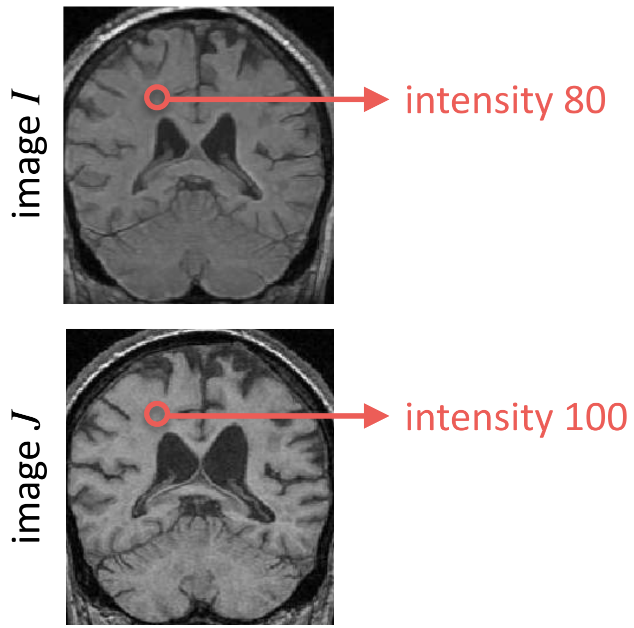


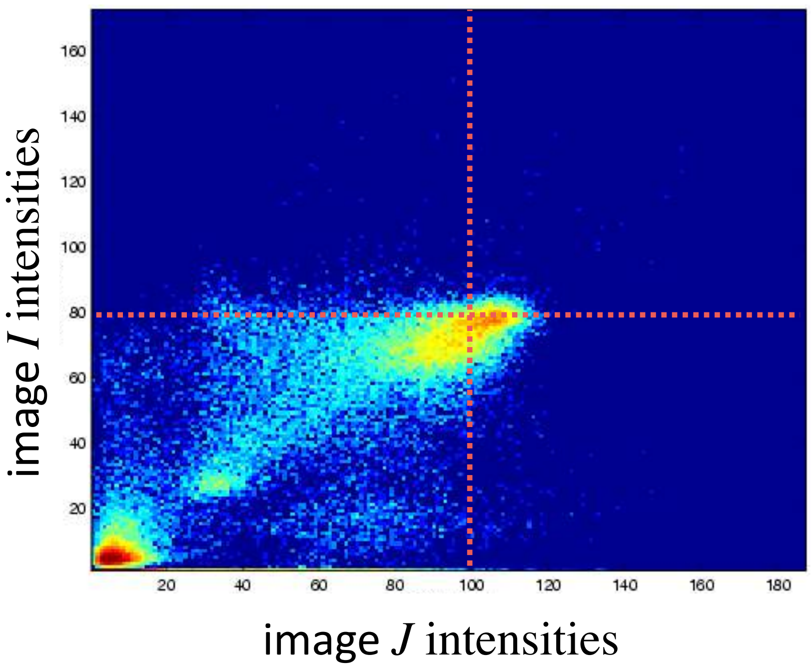
  - How to construct a metric to realize they are all the “same object”?
  - $\sum_{\mathbf{x}\in\Omega_A} |A(\mathbf{x}) - B^T(\mathbf{x})|$ would be very high in any case. Any idea?

II - Similarity measures (1/5)

Joint histogram

$$H_{I,J}(i,j) = \text{Card}\,\{(x,y) | I(x,y) = i \text{ and } J(x,y) = j\}$$





Notes

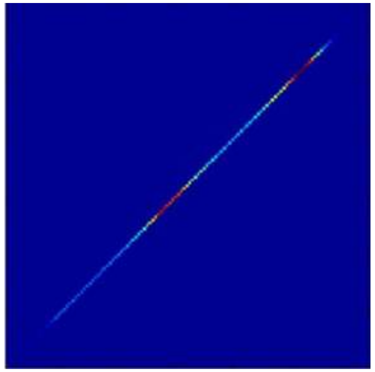
- I and J must have the *same dimensions, e.g* $M \times N$ (**NB**: in this context $J = B^T$)
- If I and J have *intensities* in [0...255]
  - $size(\ H_{I,J}\ )$ = 256×256 and $sum(\ H_{I,J}\ ) = M \cdot N$

II - Similarity measures (1/5)

Examples

image I

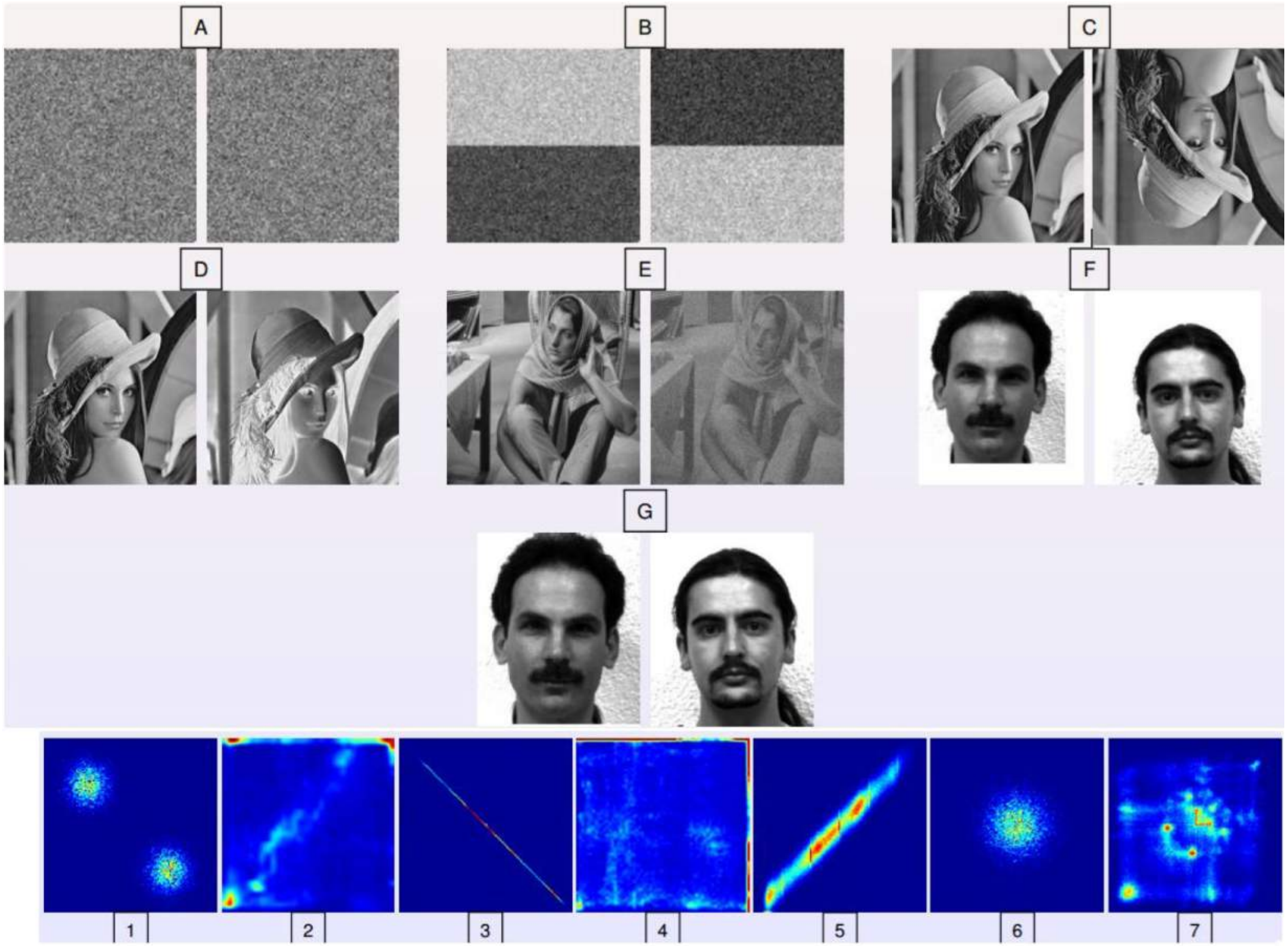
image J=I



$H_{I,J}$

Examples

A→6, B→1, C→7, D→3, E→5, F→4 , G→2



II - Similarity measures (2/5)

Minimizing intensity differences

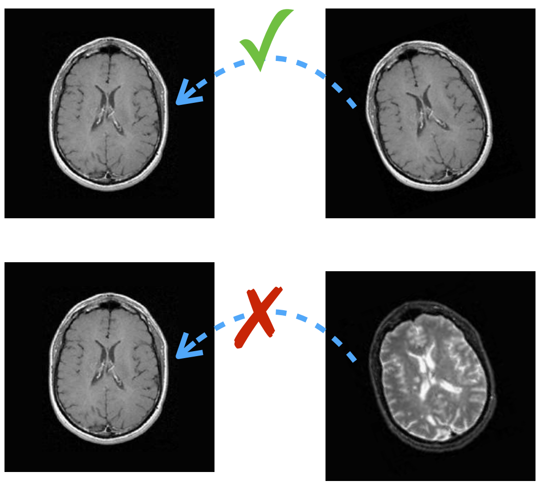
- **Sum of squared differences** (SSD)

$$\mathrm{SSD} = \sum_{\mathbf{x}_A \in \Omega^{\mathcal{T}}_{A,B}} \left| A(\mathbf{x}_A) - B^{\mathcal{T}}(\mathbf{x}_A) \right|^2$$

- SSD *very sensitive* to few voxels with very different intensities between images
  - e.g. contrast agent is injected between two acquisitions
- **Sum of absolute differences** (SAD) reduces the effect of these outliers

$$\mathrm{SAD} = \sum_{\mathbf{x}_A \in \Omega^{\mathcal{T}}_{A,B}} \left| A(\mathbf{x}_A) - B^{\mathcal{T}}(\mathbf{x}_A) \right|$$

Notes



- Computed from $H_{I,J}$: $\mathrm{SSD} = \sum_{i,j} H(i,j) \cdot (i-j)^2$
- SSD/SAD can be used only when “images are the same”
  - Same *modality*, same *contrast*, same *scaling, same visible details*…
  - …but, in practice, this is never the case
  - **NB**: *implicit assumption*: after registration the images differ only by Gaussian noise

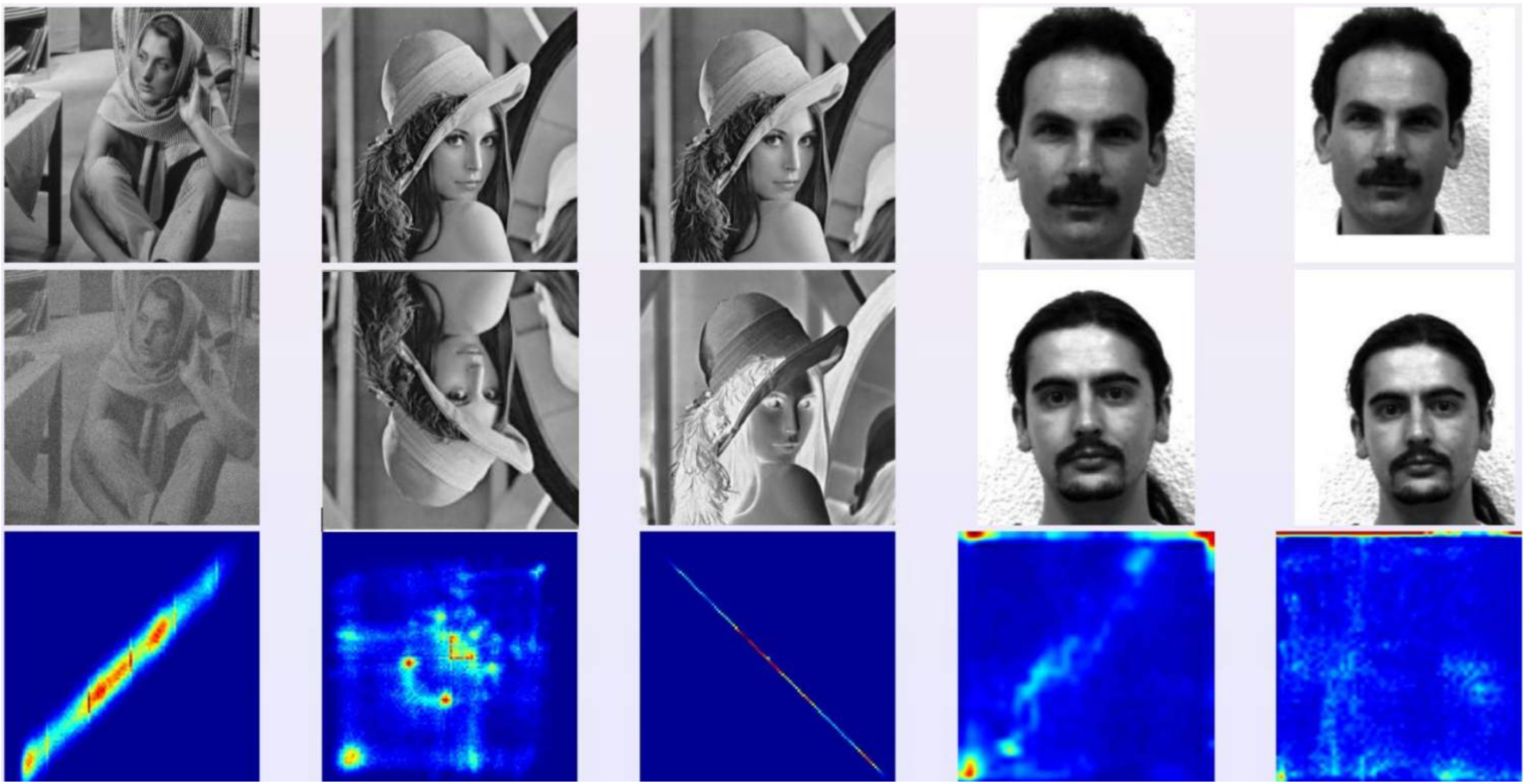
II - Similarity measures (2/5)

SSD examples

I

J

$H_{I,J}$



SSD 652 10^9 10^{10} 5600 13400

Correlation approach

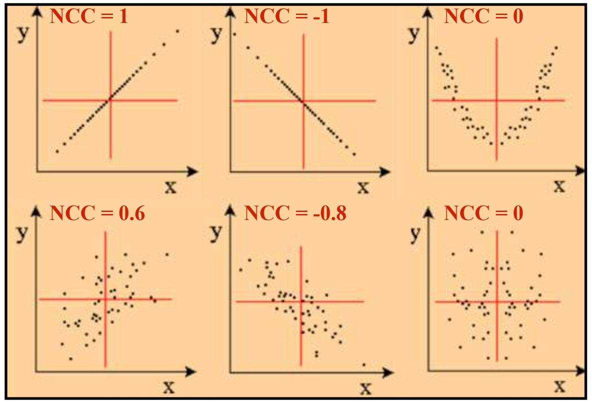
- Use a *slightly less strict assumption*
  - We don't try to have $B^{\mathcal{T}} = A$ at registration
  - We require only a relationship of the form $B^{\mathcal{T}} = \alpha A + \beta$ (linear)

- **Cross-Correlation** (CC)

$$\mathrm{CC} = \sum_{\mathbf{x}_A \in \Omega^{\mathcal{T}}_{A,B}} A(\mathbf{x}_A) \cdot B^{\mathcal{T}}(\mathbf{x}_A)$$

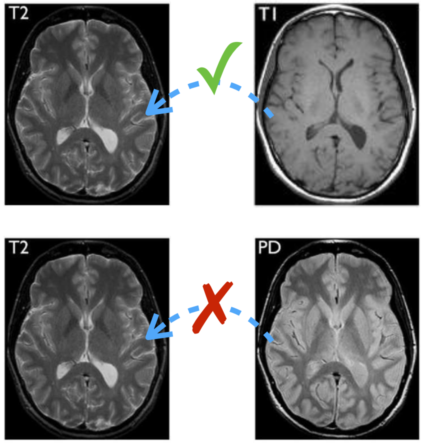
- **Normalized Cross-Correlation** (NCC)

$$\mathrm{NCC} = \frac{\sum_{\mathbf{x}_A \in \Omega^{\mathcal{T}}_{A,B}} (A(\mathbf{x}_A) - \bar{A}) \cdot (B^{\mathcal{T}}(\mathbf{x}_A) - \bar{B})}{\sqrt{\sum_{\mathbf{x}_A \in \Omega^{\mathcal{T}}_{A,B}} (A(\mathbf{x}_A) - \bar{A})^2} \cdot \sqrt{\sum_{\mathbf{x}_A \in \Omega^{\mathcal{T}}_{A,B}} (B^{\mathcal{T}}(\mathbf{x}_A) - \bar{B})^2}}$$



Notes

- $\mathrm{NCC}(I,J) \in [-1,1]\ \forall I,J$. $\mathrm{NCC}(I,J)=0$ ➜ no correlation
- Can be computed from $H_{I,J}$
- Have to be *maximized*
- Model *contrast differences*, only if linearly dependent

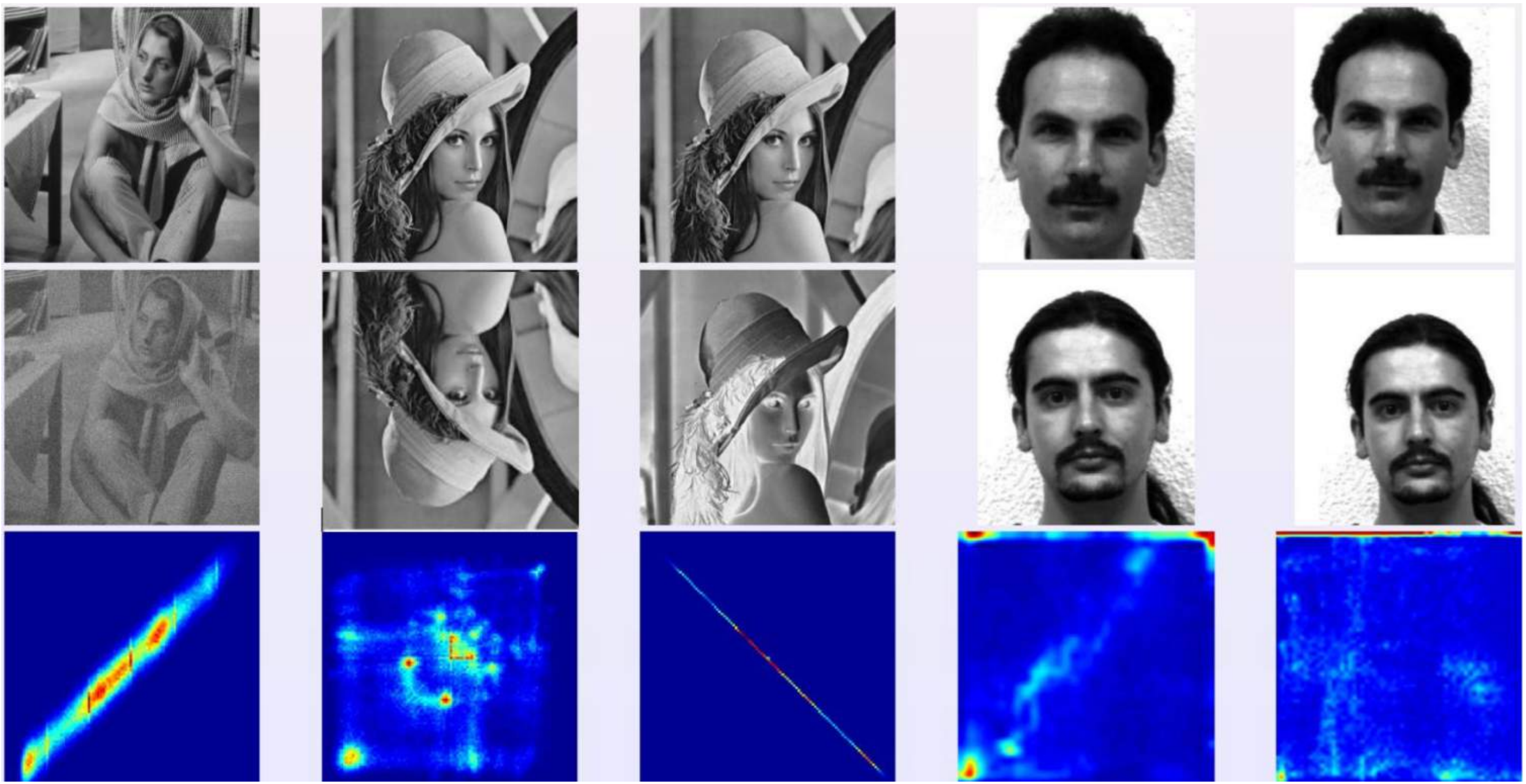


SSD vs NCC

I

J

$H_{I,J}$



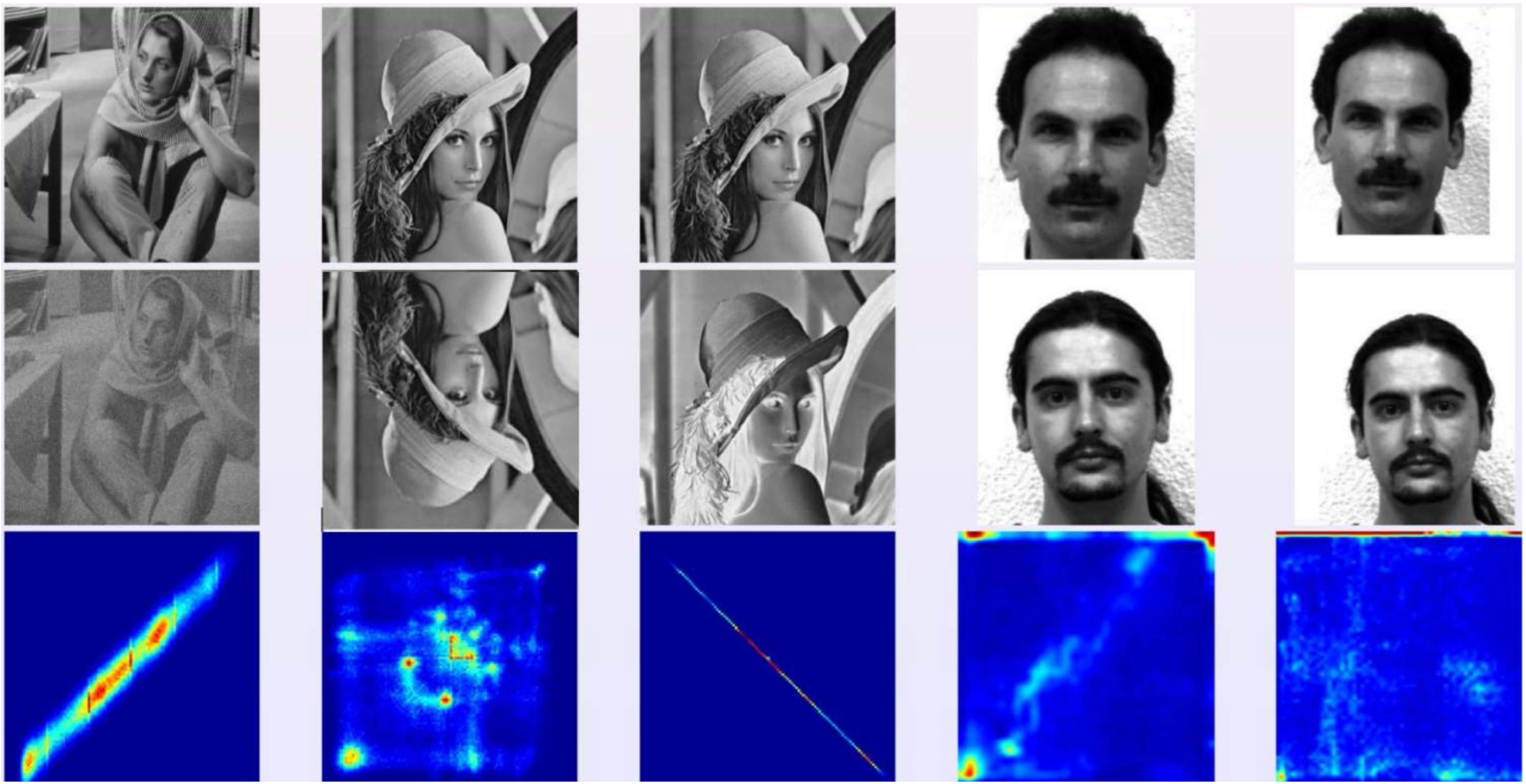
SSD	652	10^9	10^{10}	5600	13400

SSD vs NCC

I

J

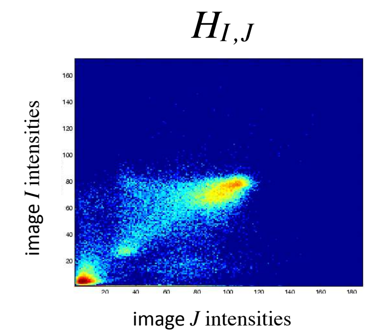
$H_{I,J}$



NCC	0.95	-0.16	-1	0.54	0.14

Statistical approach

- $H_{I,J}(i,j)$ = "*probability* that a randomly chosen pixel has *intensity i in the image I* and *intensity j in the image J*"
- Suggests the use of **statistical/information theory** techniques



Uncertainty and information

- When we say something obvious *(e.g. tomorrow the sun will rise)* it's **not interesting**, there's **no information/uncertainty** in it
- When something unlikely happens *(e.g. tomorrow a meteor will hit the Earth)* it's **very interesting**, it's an **important information**

Entropy is *measure of uncertainty* of a system

$$\mathrm{H} = -\sum_i p_i \log p_i$$

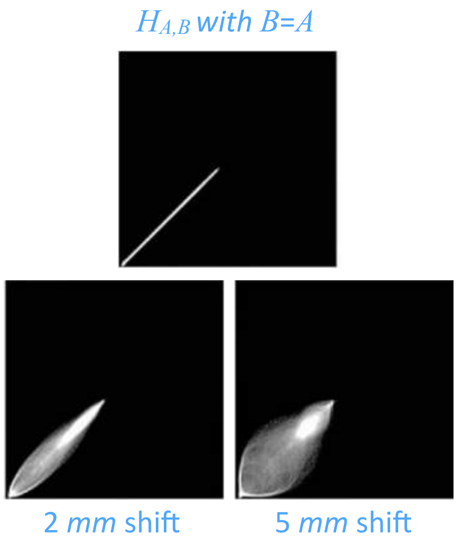
Developed by **Claude Shannon**
(*"father of information theory"*)

- Set of n symbols with probability of occurrence $p_1, p_2, \ldots, p_n$
  - All symbols have equal probability ➜ max uncertainty/information ➜ H **is max**
  - One has probability 1, the rest 0 ➜ no uncertainty/information ➜ H **is min**

Entropy for **image registration**

- Two images to align, so *two symbols* at each pixel
- $H_{I,J}(i,j)$ = **joint probability distribution** of images A and B (let's call it p_{AB})
- **Joint entropy** measures the information in the two images combined:

$$\mathrm{H}(A,B) = -\sum_a \sum_b p_{AB}(a,b)\ \log p_{AB}(a,b)$$



Registration seen as seeking to *reduce the amount of information* in the combined image

- Sharper $H_{I,J}$ ➜ lower $\mathrm{H}(A,B)$ ➜ reduced uncertainty

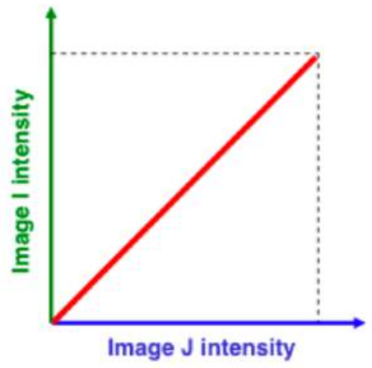
Mutual Information (MI) usually preferred

$$\mathrm{MI}(A,B) = \sum_a \sum_b\ p_{AB}(a,b)\ \log \frac{p_{AB}(a,b)}{p_A(a) \cdot p_B(b)} = \mathrm{H}(A) + \mathrm{H}(B) - \mathrm{H}(A,B)$$

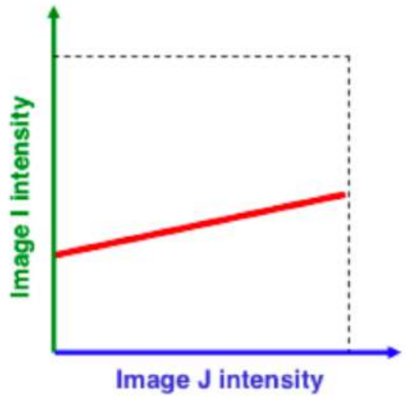
- Measures how well **one image explains the other**
- MI is *maximimum* at optimal alignment

Summary

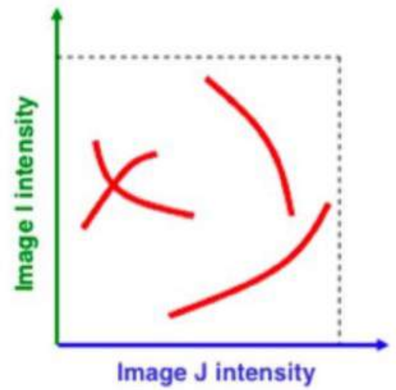
- **Minimizing intensity differences**
  - ***Sum of squared differences*** (SSD) or ***sum of absolute differences*** (SAD)
  - To be *minimized*
  - Suited for *mono-modal*, *intra-subject* registration
  - Strong assumption on intensities



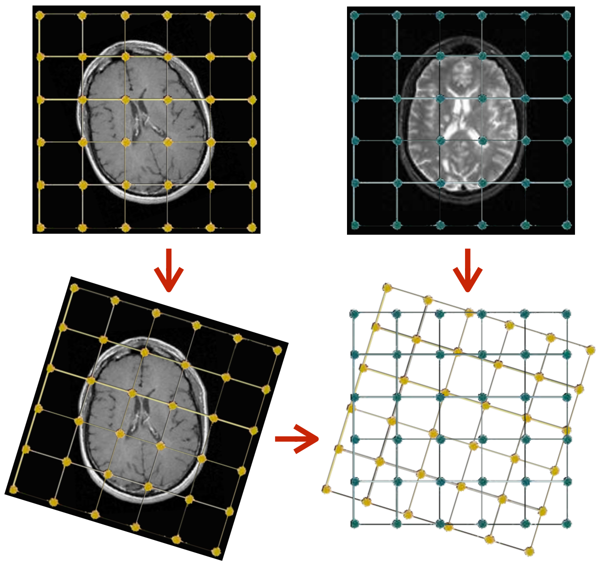
- **Correlation approach**
  - Relaxes the previous assumption, allowing linear dependence
  - ***Normalized Cross-Correlation*** (NCC)
  - To be *maximized*
  - Suited for *mono-modal*, *intra-* or *inter-subject* registration



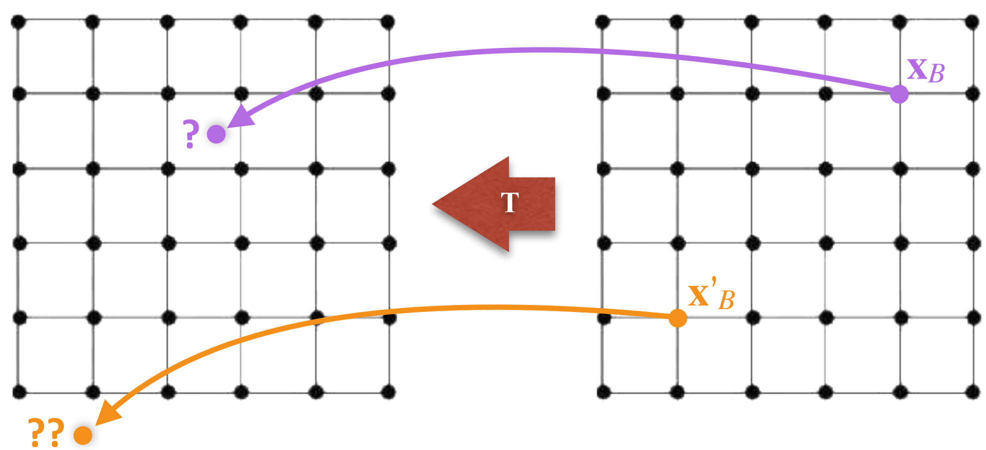
- **Statistical interpretation**
  - Weakest assumption on the relationship between intensities
  - ***Mutual Information*** (MI)
  - To be *maximized*
  - Suited for *multi-modal* registration (*intra-* and *inter-subject*)



- **To compute distance/similarity** $d(A, B^{\mathcal{T}})$ **we need to compare features/intensities at same locations on both images**
  - If $\mathcal{T}$ maps the pixels of B exactly at the same locations of the pixels of A, there are no problems
  - But usually the locations/grids do not match



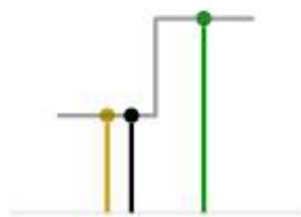
- **Two cases**
  - **Interpolation**
    - For the points $\mathbf{T}(\mathbf{x}_B)$ falling *inside* the grid of A (but not on the grid points themselves)
    - Value for these points needs to be estimated from the *neighboring pixels*
  - **Extrapolation**
    - For the points $\mathbf{T}(\mathbf{x}_B)$ falling *outside* the grid of A
    - Points not considered? Mirror or extend pixels?



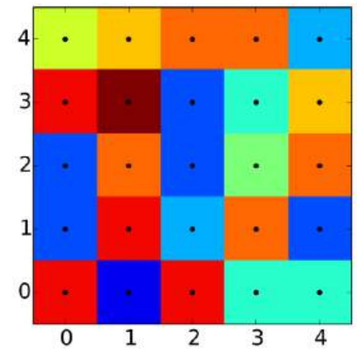
Most common choices

- **Nearest neighbor**

1D



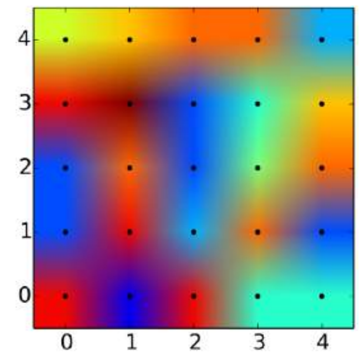
2D

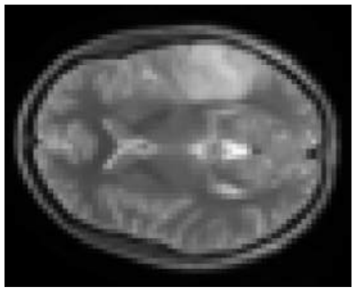


- **Linear**

1D

2D

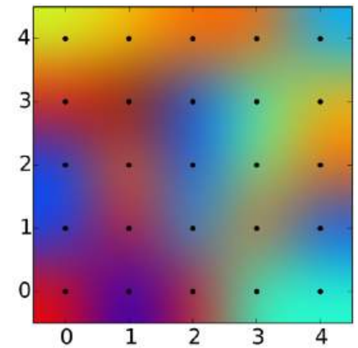


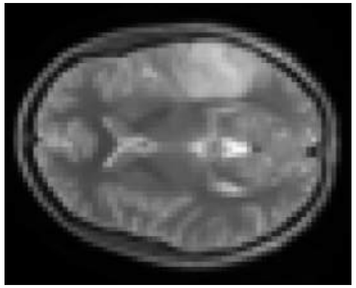


- **Higher order**, e.g. cubic or B-spline

1D

2D

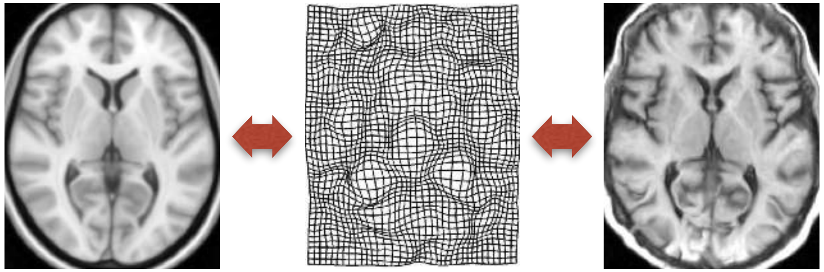


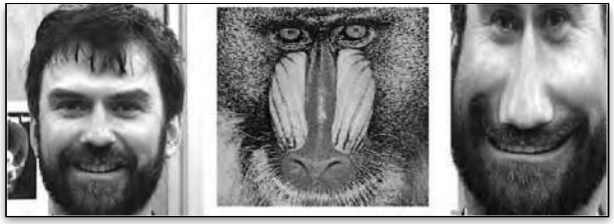


- Two main categories
  - **Linear** (a.k.a. *rigid*)
    - Only a limited number of *degrees of freedom* is allowed

$$\begin{pmatrix} \cos\beta\cos\gamma & \cos\alpha\sin\gamma + \sin\alpha\sin\beta\cos\gamma & \sin\alpha\sin\gamma - \cos\alpha\sin\beta\cos\gamma & t_x \\ -\cos\beta\sin\gamma & \cos\alpha\cos\gamma - \sin\alpha\sin\beta\sin\gamma & \sin\alpha\cos\gamma + \cos\alpha\sin\beta\sin\gamma & t_y \\ \sin\beta & -\sin\alpha\cos\beta & \cos\alpha\cos\beta & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

  - **Non-linear** (a.k.a. *non-rigid*)
    - Virtually any transformation/deformation is possible



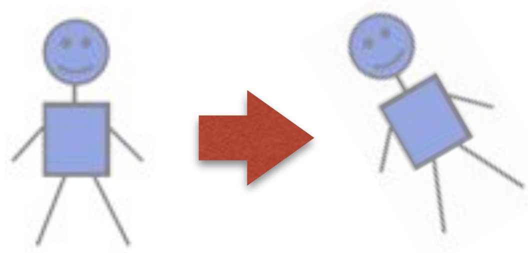


- **NB**: the **choice of the deformation model** to use depends on the application, i.e. which tissue/structure to register
  - *Bones of the skull* restrict the movement of the brain
  - *Soft tissue* tends to deform in more complicated ways

IV - Deformation models (2/4)

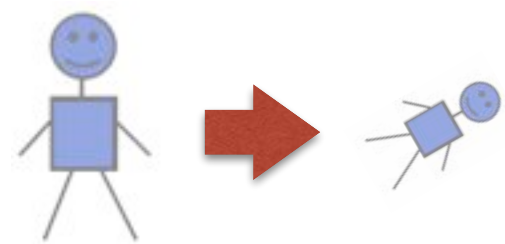
Linear transformations

- **Rigid :** $\mathbf{T}(\mathbf{x}) = \mathbf{R}\mathbf{x} + \mathbf{t}$

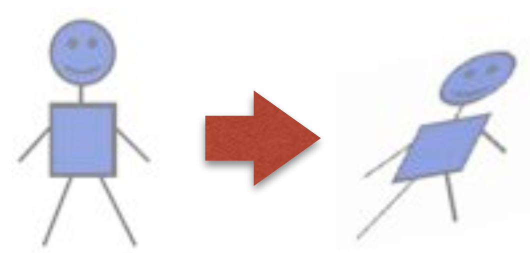
  - **6 parameters** : rotation ($\mathbf{R}$) and translation ($\mathbf{t}$)
  - *Invariants*: distances (isometric), curvature, angles, lines
  - *Use*: same structure in a different position

- **Similitude :** $\mathbf{T}(\mathbf{x}) = \mathbf{s}\mathbf{R}\mathbf{x} + \mathbf{t}$

  - **7 parameters**: adds a scaling factor ($\mathbf{s}$)
  - *Invariants*: distance ratios, angles, lines

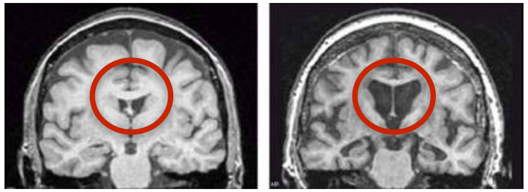
- **Affine :** $\mathbf{T}(\mathbf{x}) = \mathbf{A}\mathbf{x} + \mathbf{t}$

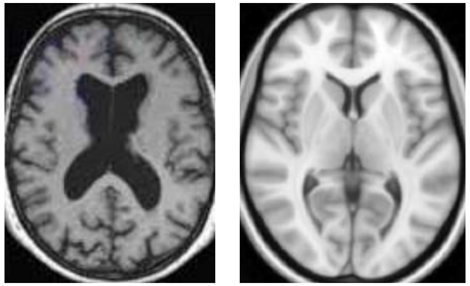
  - **12 parameters**: $\mathbf{A}$ includes stretching and shearing
  - *Invariants*: lines, parallelism
  - *Use*: - correct for scanner deformations/artifacts
    - find approximate alignment before nonlinear registration

<u>**Nonlinear transformations**</u> required when registering:

- An image of one individual and **atlas**
- Image from **different individuals**
- **Tissue that deforms** over time



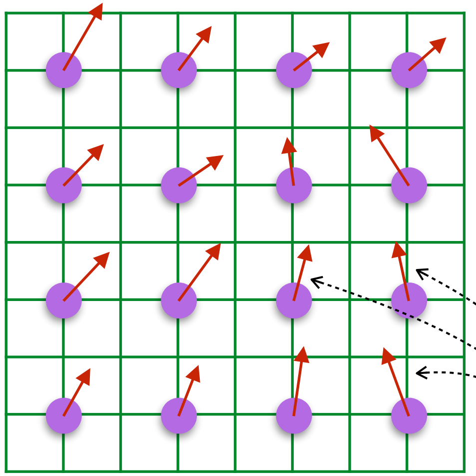
time 1 time 2

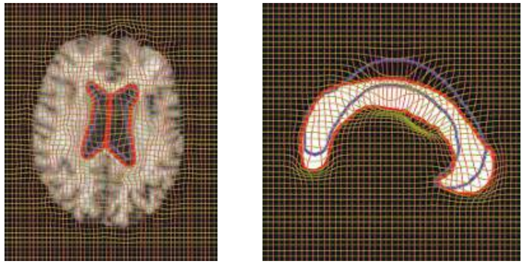


subject template

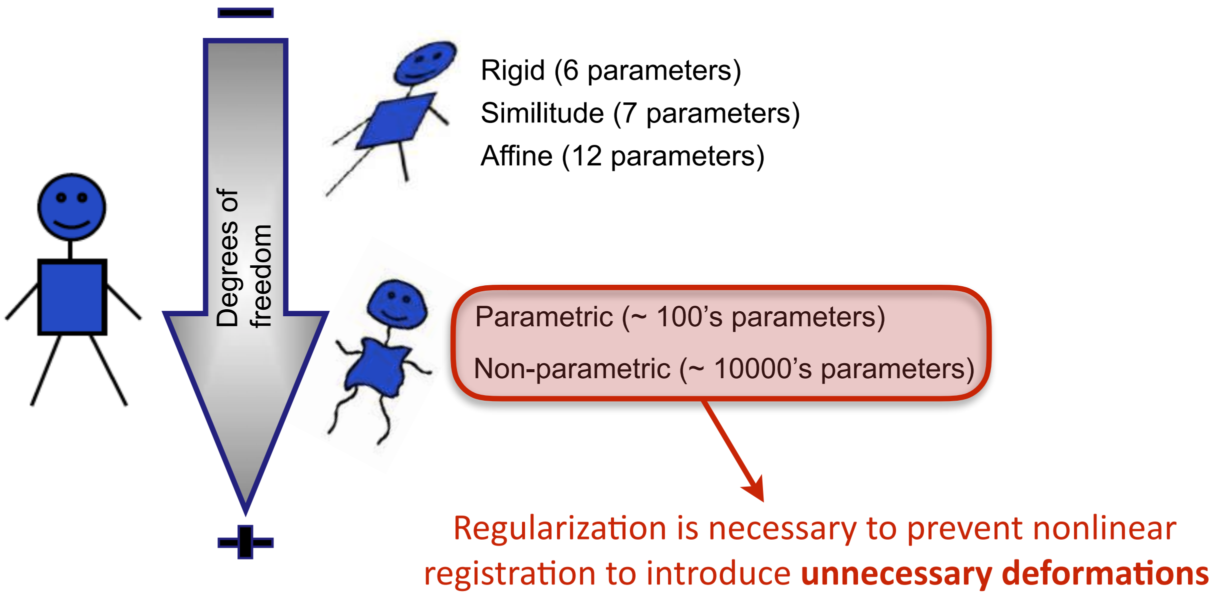
General approach

- **Each pixel** can virtually be moved independently
  - One displacement per pixel
  - Actual *tissue deformations* are usually more smooth/regular
- Usually **grids** of **control points** are defined
  - One displacement $\boldsymbol{u}(\mathbf{x})$ (↗) per control point (●)
  - *Smoothness constraints* are usually added to obtain "anatomically reasonable" deformations
  - Control points are *not independent*
- Several solutions **inspired by physics**
  - *Elastic, viscous fluid, optical flow, diffusion model* (demons) …

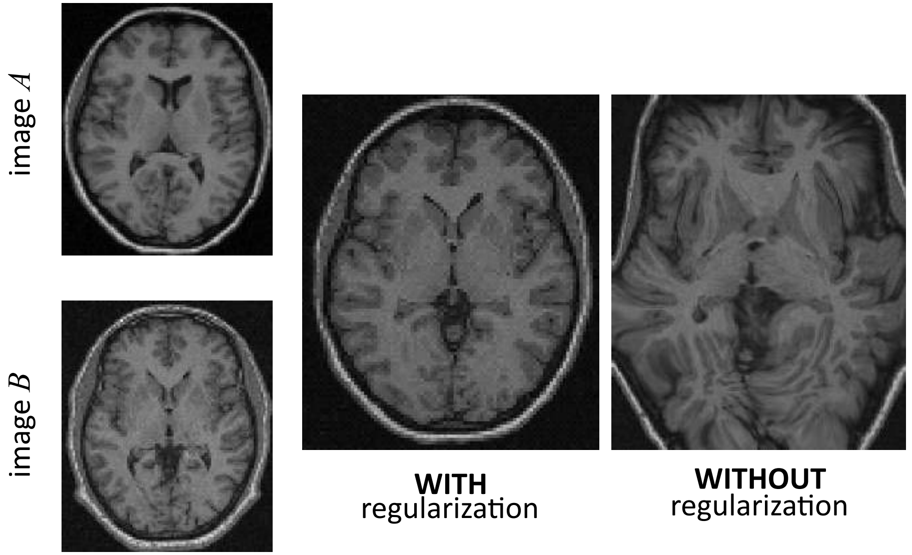




- **Increased complexity**: *overfitting* and *regularization*

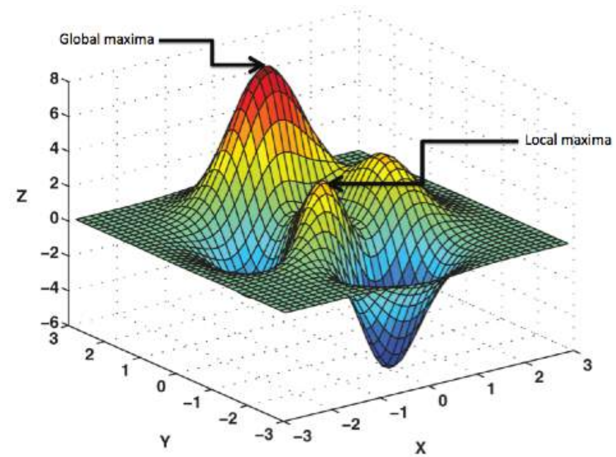


- Example

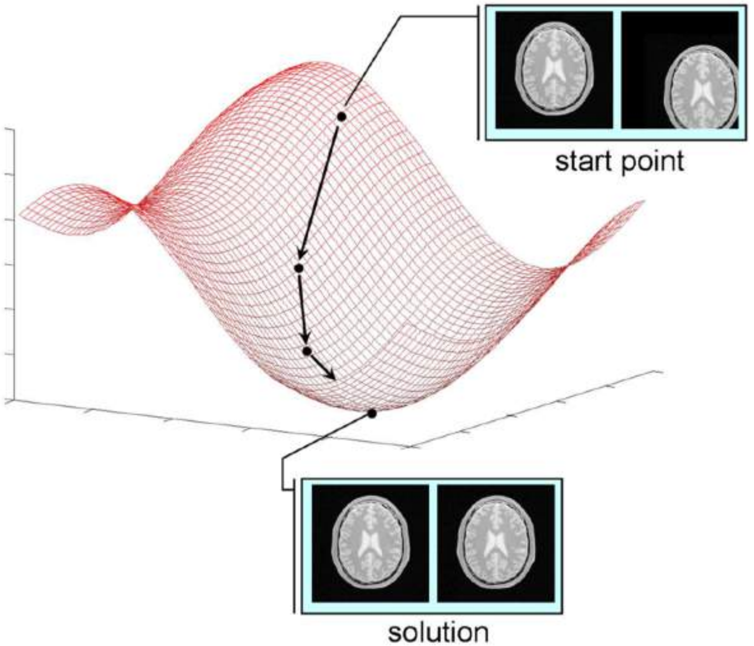


V - Optimizers

- Registration is an **optimization problem**
  - The search space is **high dimensional**
    (i.e. space of all possible transformations)
  - The problem is **nonlinear**
    (possibly with many **local minima**)



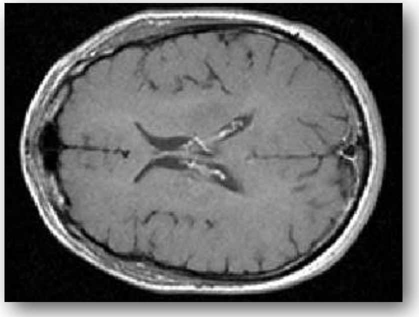
- Usually **iterative approaches** are used
  - Start with *initial estimate* of transformation, $\mathbf{T}^0$
  - At each iteration *t*, current estimate $\mathbf{T}^t$ is used to compute a *similarity measure* $d(A, B^{\mathcal{T}})$
  - Using d, *refine the transformation* $\mathbf{T}^t \rightarrow \mathbf{T}^{t+1}$
  - Continues until convergence



- Classical algorithms
  - *Gauss-Newton, (stochastic) gradient descent* etc...

Trick 1: image sampling strategy

- So far, similarity metrics between two images were computed over **all the voxels**

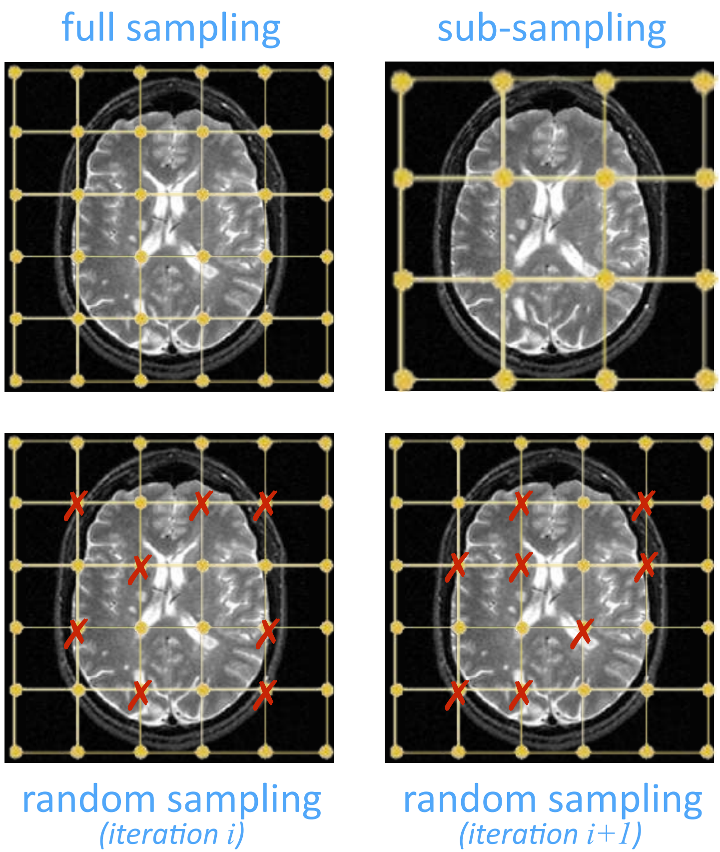


$$\mathrm{SSD} = \sum_{\mathbf{x}_A \in \Omega^{\mathcal{T}}_{A,B}} \left| A(\mathbf{x}_A) - B^{\mathcal{T}}(\mathbf{x}_A) \right|^2 \qquad \mathrm{CC} = \sum_{\mathbf{x}_A \in \Omega^{\mathcal{T}}_{A,B}} A(\mathbf{x}_A) \cdot B^{\mathcal{T}}(\mathbf{x}_A) \qquad \textit{etc...}$$

  - Images contain a lot of *redundancy* ➜ it's **not necessary to evaluate all voxels**
  - In most situations, a **subset may suffice**

Common strategies

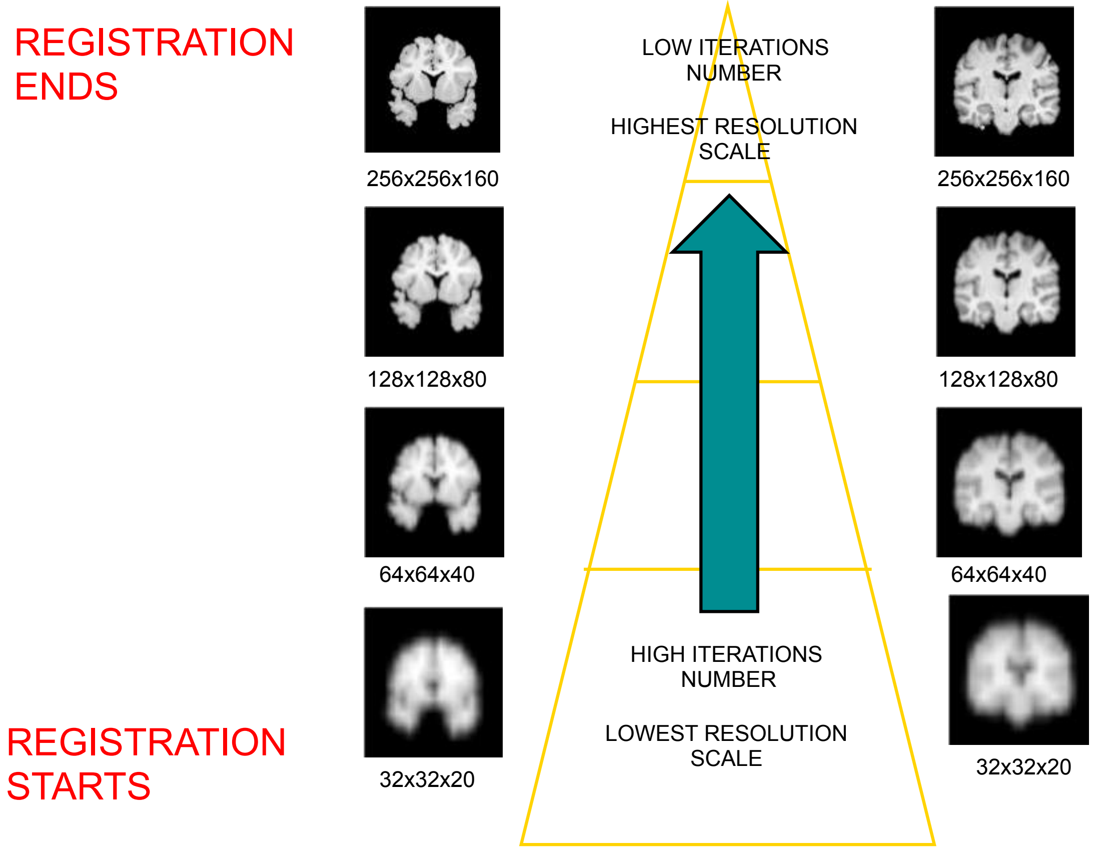
- **Full sampling**
  - Similarity metrics computed on *all voxels of the image*
- **Sub-sampled regular grid**
  - Only *voxels on a coarser regular grid* are evaluated
  - The size of this grid (i.e. downsampling factor) can be adapted
- **Random**
  - Voxels to be compared are *randomly selected*
  - Coordinates can be *discrete* (voxels) or *continuous* (sub-voxel)
  - Important to resample at every iteration



Trick 2: multi-scale registration

Strategy to try **avoiding local minima**

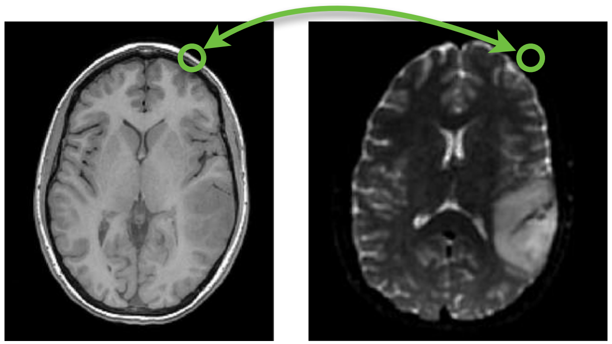
- Start the registration using **images with low complexity** → *smoothing* / *downsampling*
- At convergence, **increase the complexity/details of the images** and repeat
- This reduces the chance of falling in **local minima** (bad registration)



Trick 3: use of masks

Sometimes it is desirable to **align only portions of an image**

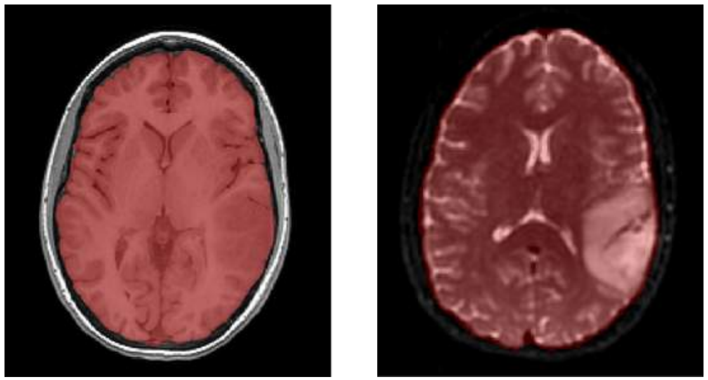
- We are interested **only on a portion or some details** of the image
- We need to **ignore parts of the images** that can confound the registration (e.g. *artificial edges*)



Some anatomical details are not visible in both images

With a *mask*, the registration is **constrained to a region**

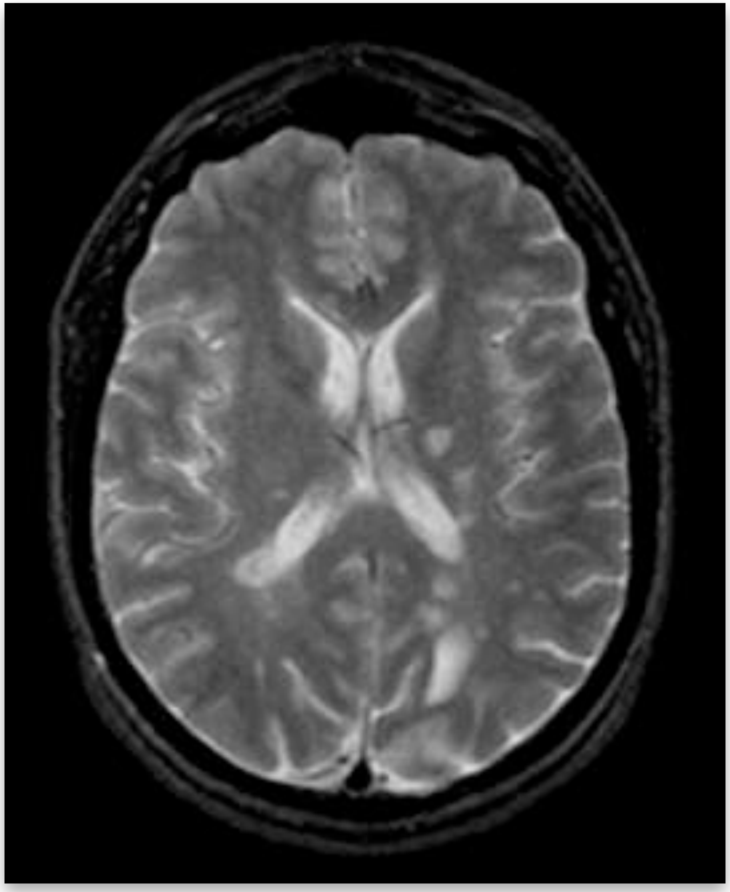
- A mask is a **binary image**
  - **"1"** ➡ the pixel in *considered*
  - **"0"** ➡ the pixel in *ignored*

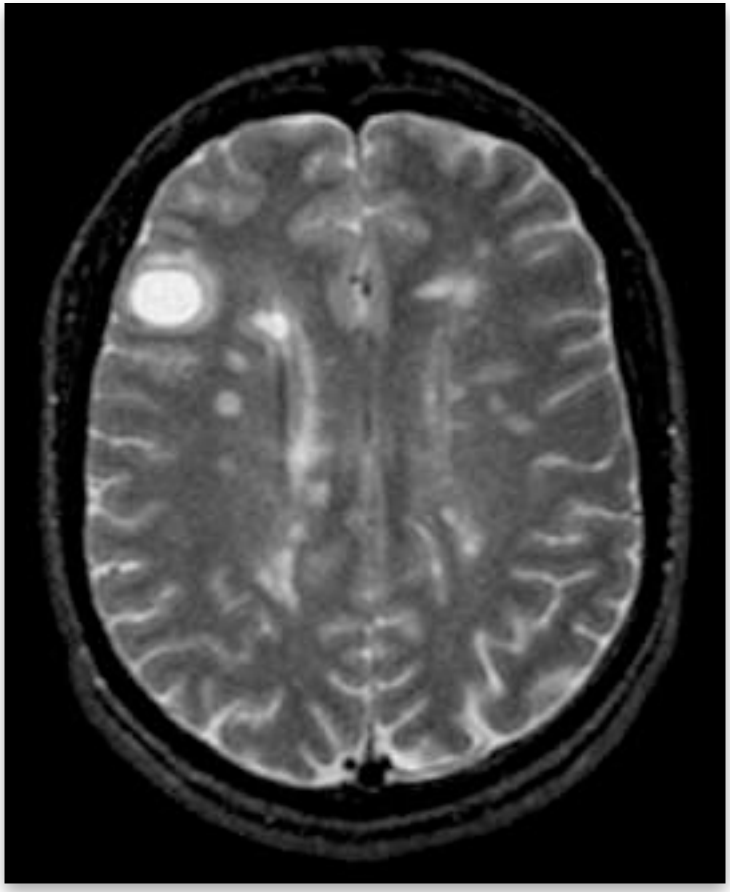


- A **fixed image mask is usually sufficient** to focus the registration on a region, since samples are drawn from the domain of the fixed image

What would you use in these cases? (1/5)

- Intra-subject, same modality, follow-up scans

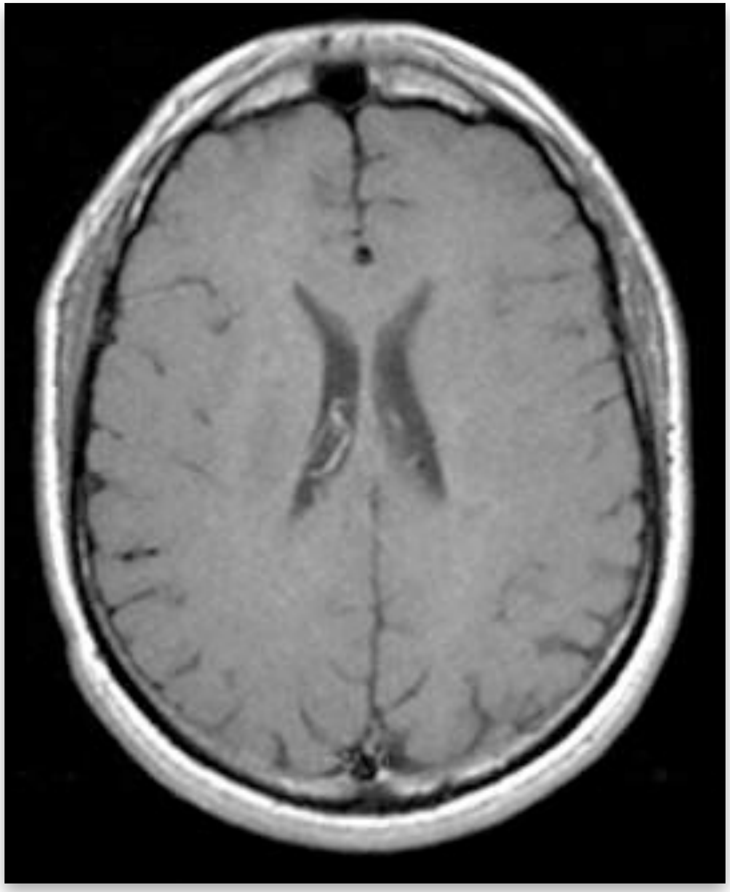


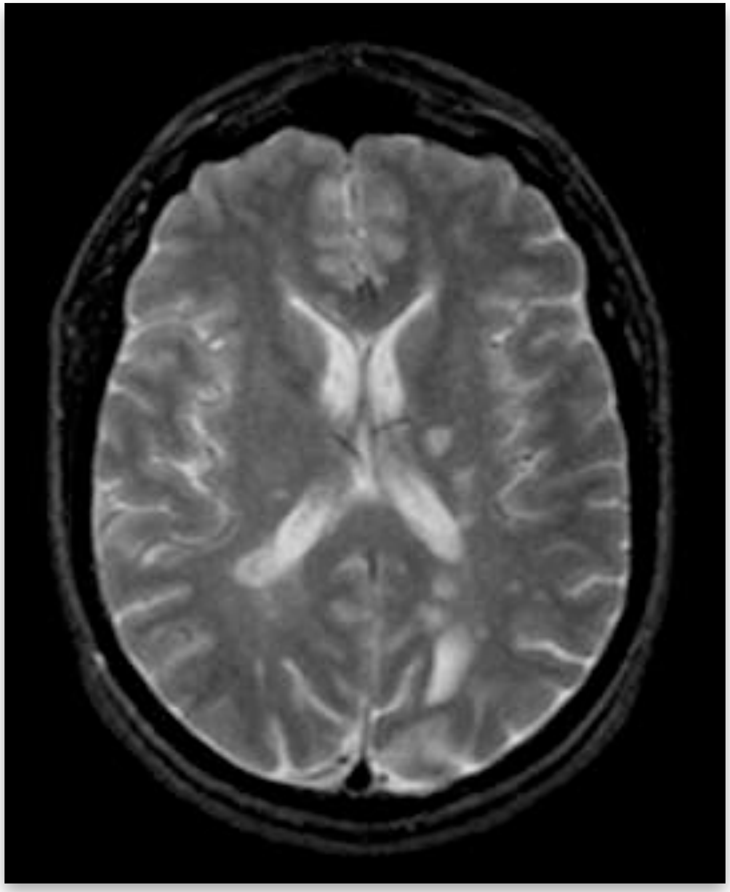


- Rigid transformation (**6 DOF**)
  Sum of squared differences (**SSD**)

What would you use in these cases? (2/5)

- Intra-subject, different modalities

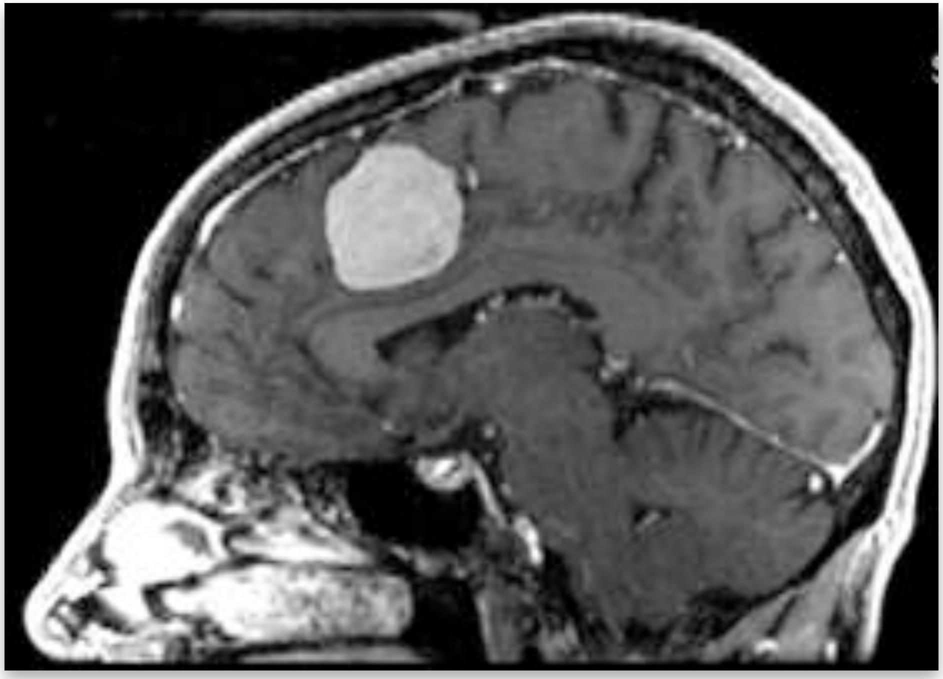


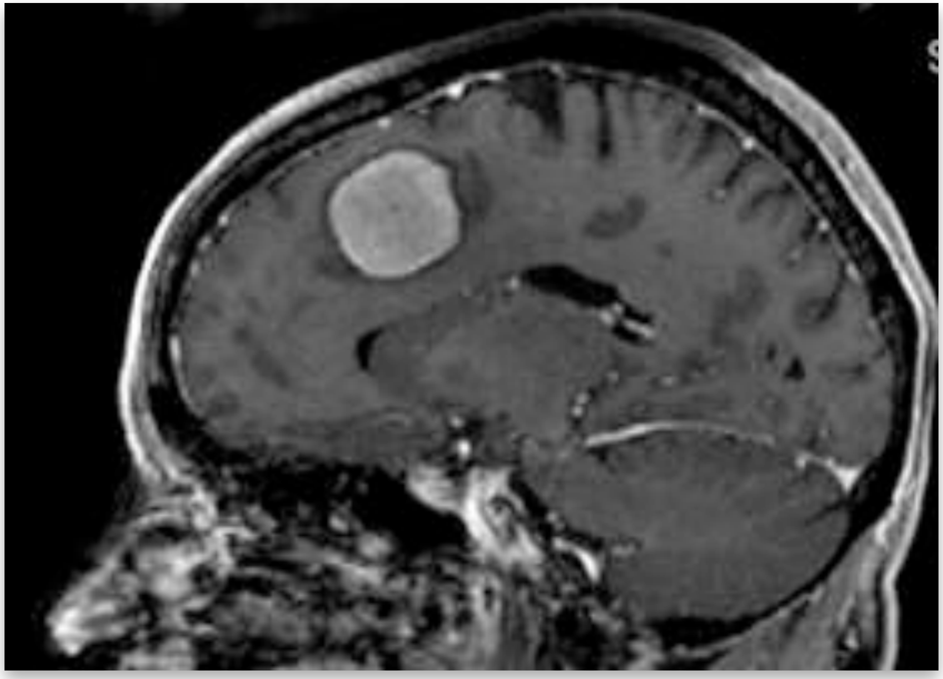


- Rigid transformation (**6 DOF**)
  Normalized Cross-Correlation (**NCC**) or Mutual Information (**MI**)

What would you use in these cases? (3/5)

- Intra-subject, same modality, monitor the growth of a tumor

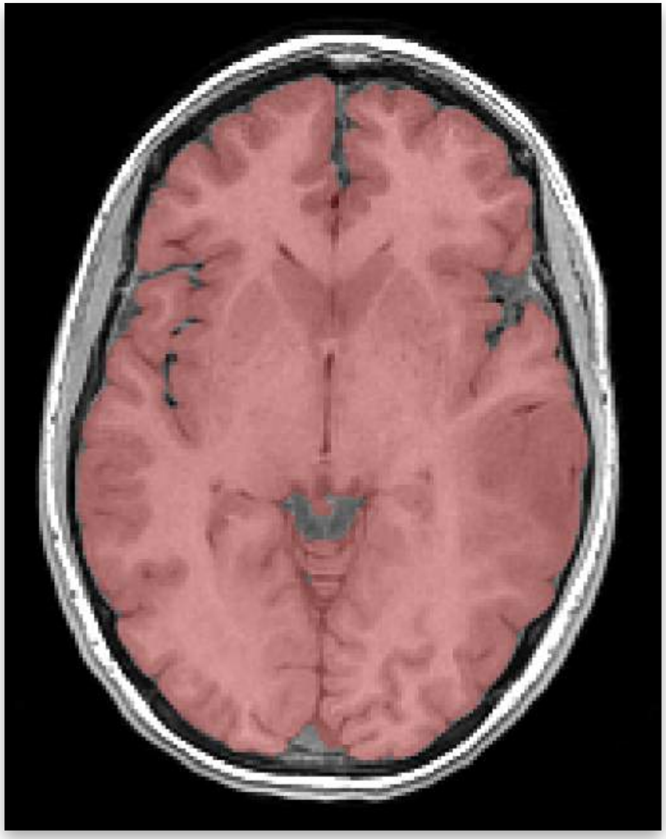


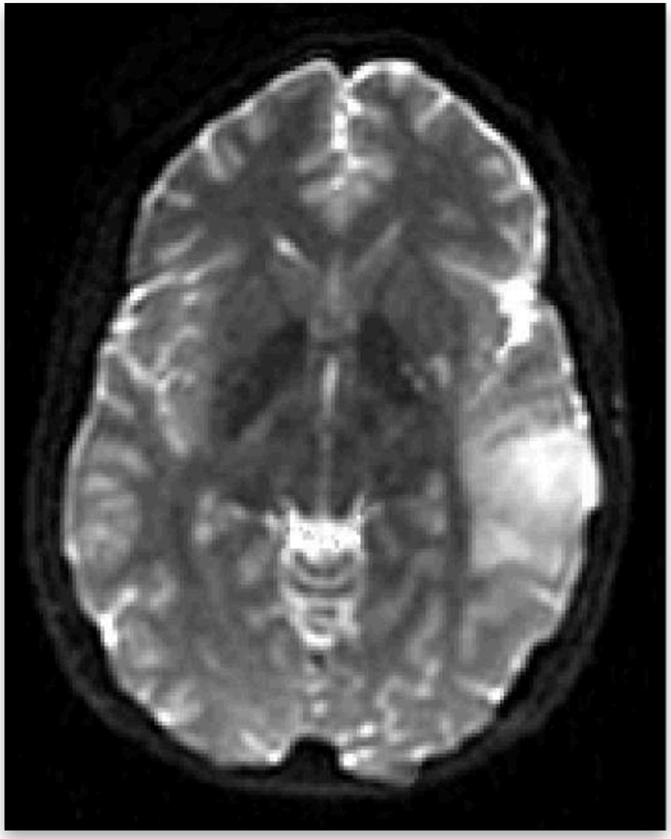


- Affine transformation (**12 DOF**) or non-rigid
  Sum of squared differences (**SSD**) or Mutual Information (**MI**)
  - Tumor can deform the tissues in a non-rigid way
  - MI might be helpful to cope with very large differences of intensities near tumor

What would you use in these cases? (4/5)

- Intra-subject, different modalities, different details/artifacts

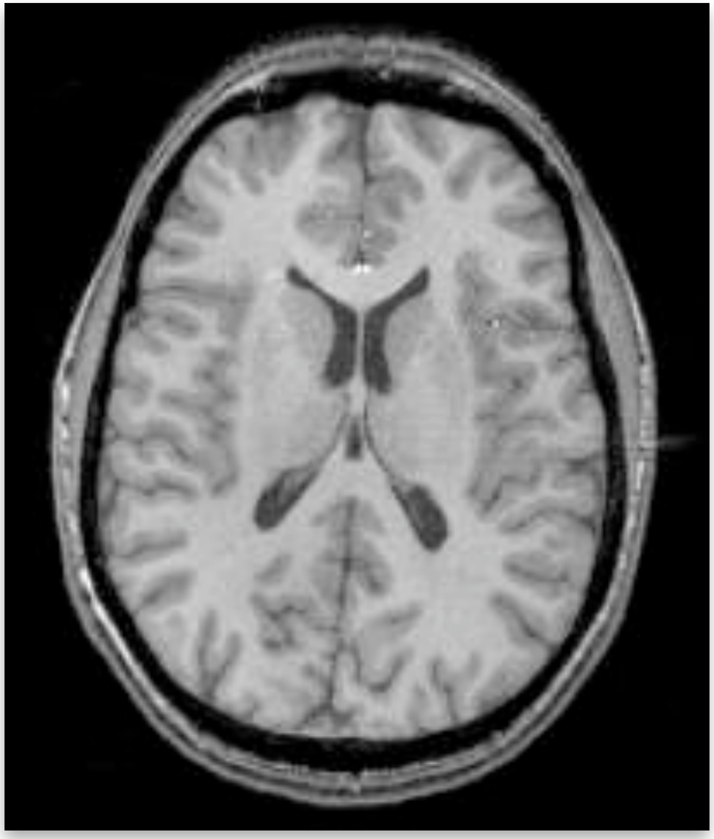




- Affine transformation (**12 DOF**) or non-rigid
  Mutual Information (**MI**)
  Use a mask to delineate the brain

What would you use in these cases? (5/5)

- Inter-subject, different modalities



- **Non-rigid** transformation
  Mutual Information (**MI**)