# Introduction to image registration

Pietro Gori

Enseignant-chercheur
Equipe IMAGES - Télécom Paris
pietro.gori@telecom-paris.fr

# Plan

# Summary

# Image registration

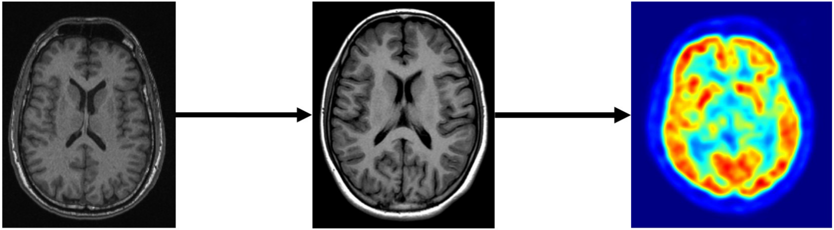
## Definition

- **Geometric**: find the *optimal* parameters of a *geometric transformation* to spatially align two different images of the same object. It establishes *spatial correspondence* between the pixels of the source (or moving) image with the ones of the target image
- Photometric: Modify the intensity of the pixels and not their position

## Medical Image Applications

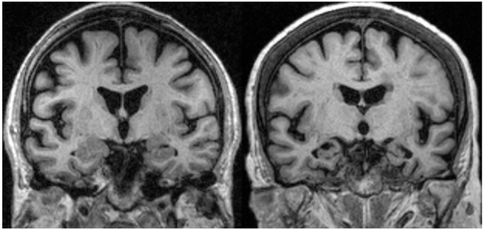
- Compare two (or more) images of the same modality (e.g. T1-w MRI of the brain of two different subjects)
- Combine information from multiple modalities (e.g. PET, DWI, T1-w MRI of the brain of the same subject)
- Longitudinal studies (e.g. monitor anatomical or functional changes of the brain over time)
- Relate preoperative and postoperative images after surgery

# Medical Image registration - Applications

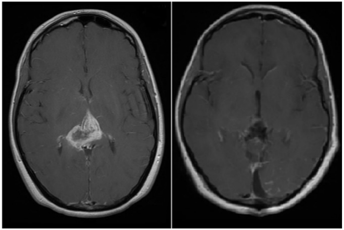


Same modality
Different subjects

Different modalities
Same subject



Longitudinal study



Pre and postoperative

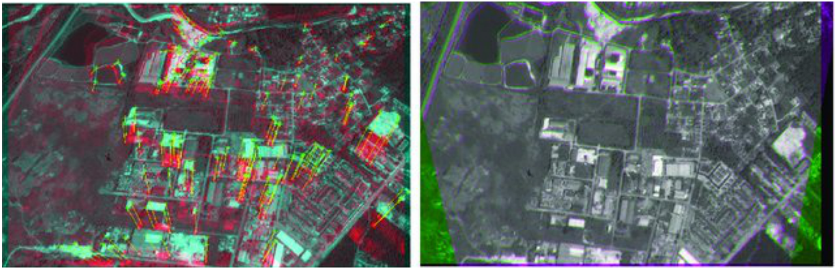
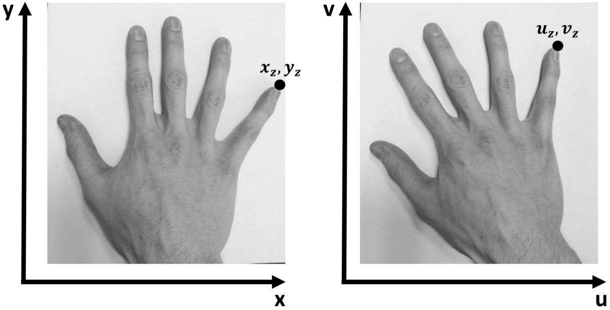


Figure 1: Remote sensing example - From 'Geometric feature descriptor and dissimilarity-based registration of remotely sensed imagery' PLoS One, 2018

# Image registration components

- **Dimensionality**: 2D/2D, 3D/3D, 2D/3D
- **Transformation** (linear / non-linear)
- **Similarity metric** (e.g. intensities, landmarks, edges, surfaces)
- **Optimization procedure**
- **Interaction** (automatic / semi-automatic / interactive)
- **Modalities** (mono-modal / multi-modal)
- **Subjects** (intra-subject / inter-subject / atlas construction)

- Let $I$ and $J$ be the source and target images. They show the same anatomical object, most of the time with a different field of view and resolution (sampling)
- $I(x,y)$ and $J(u,v)$ represent the intensity values of the pixels located onto two regular grids: $\{x,y\} \in \Omega_I$ and $\{u,v\} \in \Omega_J$
- For the same subject, the same anatomical point $z$ can be in position $x_z, y_z$ in $I$ and in $u_z, v_z$ in $J$



# Mathematical definition

- Both $I$ and $J$ are functions:

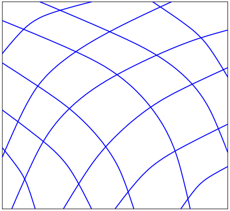
$$
\begin{array}{cccc}
I(x,y): & \Omega_I \subseteq \mathbb{R}^2 & \rightarrow & \mathbb{R} \\
 & (x,y) & \rightarrow & I(x,y)
\end{array}
$$

- We look for a geometric transformation $\mathbf{T}$, which is a 2D warping parametric function that belongs to a certain family $\Gamma$:

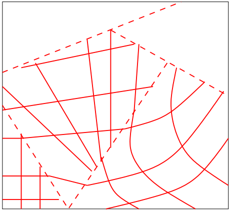
$$
\begin{array}{cccc}
\mathbf{T}_\phi(x,y): & \mathbb{R}^2 & \rightarrow & \mathbb{R}^2 \\
 & (x,y;\phi) & \rightarrow & \mathbf{T}_\phi(x,y)
\end{array}
$$

- $\phi$ is the vector of parameters of $\mathbf{T}$. We look for a transformation that maps $(x_z, y_z)$ to $(u_z, v_z)$

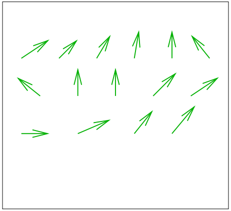
- Most of the time, $I$ and $J$ are simply seen as matrices whose coordinates $(x, y)$ and $(u, v)$ are thus integer-valued (number of line and column)
- The values of the intensities of the pixels can be real numbers $\mathbb{R}$ (better for computations) or integer, usually in the range $[0, 255]$ for a gray-scale image
- There are several kind of transformations:



modèle global



modèle par morceaux (régional)



modèle local

# Summary

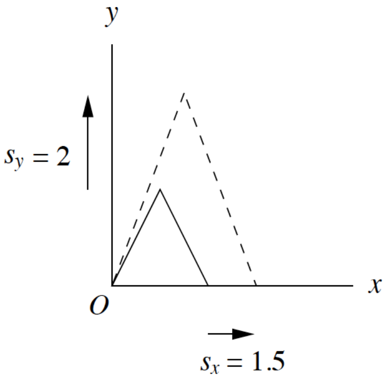
# Global transformations

- Global transformations can be defined with matrices
- Application: $\mathbf{T}_\phi(x,y) = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = A\mathbf{x}$
- Inverse (if invertible): $\mathbf{T}_\phi^{-1}(x,y) = A^{-1}\mathbf{x}$
- Composition: $\mathbf{T}_1(\mathbf{T}_2(x,y)) = (\mathbf{T}_1 \circ \mathbf{T}_2)(x,y) = A_1 A_2 \mathbf{x}$
- Note: order of transformation is important : $A_1 A_2$ is not equal to $A_2 A_1$ in general

# Global transformations

- Global means that the transformation is the same for any points $p$
- **Scaling** - multiply each coordinate by a scalar

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \quad (1)$$



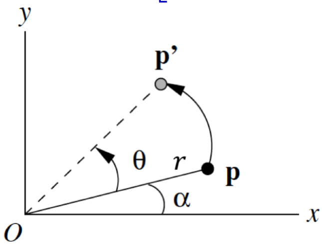
# Global transformations

- **Rotation** - WRT origin. Let $p = [x\ y]^T$ and $p' = [u\ v]^T$

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} r\cos(\alpha) \\ r\sin(\alpha) \end{bmatrix}$$

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} r\cos(\alpha+\theta) \\ r\sin(\alpha+\theta) \end{bmatrix} = \begin{bmatrix} r(\cos(\alpha)\cos(\theta) - \sin(\alpha)\sin(\theta)) \\ r(\sin(\alpha)\cos(\theta) + \cos(\alpha)\sin(\theta)) \end{bmatrix}$$

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




- $R = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}$
- $\det(R) = 1$
- $R^{-1} = R^T$

# Global transformations

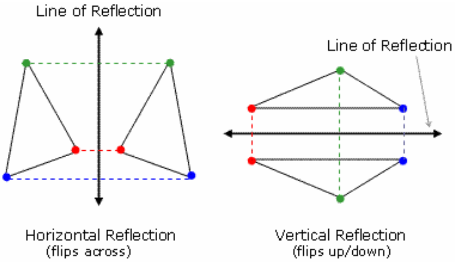
- **Reflection**
- Horizontal (Y-axis)

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} r\cos(\pi-\alpha) \\ r\sin(\pi-\alpha) \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

- Vertical (X-axis)

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} r\cos(\frac{3}{2}\pi+\alpha) \\ r\sin(\frac{3}{2}\pi+\alpha) \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

- $\det(R) = -1$
- $R^{-1} = R^T$



- **Shear** - Transvection in french

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} 1 & \lambda_x \\ \lambda_y & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 1 & \tan(\phi) \\ \tan(\psi) & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

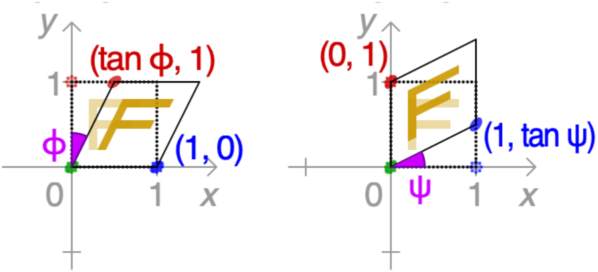



Figure 2: Shear in x and y direction. Image taken from Wikipedia.

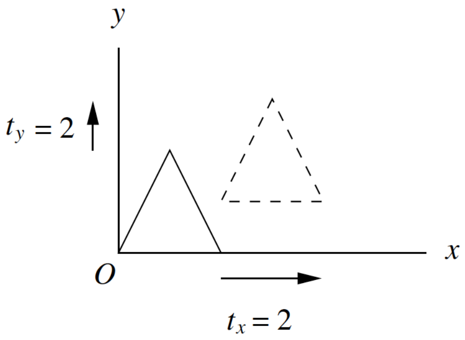
# 2D Linear transformations

$$\begin{bmatrix} u \\ v \end{bmatrix} = \underbrace{\begin{bmatrix} \pm 1 & 0 \\ 0 & \pm 1 \end{bmatrix} \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \begin{bmatrix} 1 & \lambda_x \\ \lambda_y & 1 \end{bmatrix}}_{\begin{bmatrix} a & b \\ c & d \end{bmatrix}} \begin{bmatrix} x \\ y \end{bmatrix}$$

- Linear transformations are combinations of:
  - scaling
  - rotation
  - reflection
  - shear
- Properties of linear transformations:
  - origin is always transformed to origin
  - parallel lines remain parallel
  - ratios are preserved
  - lines remain lines

- It does not have a fixed point $\rightarrow$ no matrix multiplication

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix} = \begin{bmatrix} x + t_x \\ y + t_y \end{bmatrix} \tag{2}$$



# Homogeneous coordinates - 2D affine transformation

- Instead than 2D matrices we use 3D matrices.
- **Affine transformation**: combination of linear transformations and translations
- Let $p = [x \ y]^T$ and $p' = [u \ v]^T$, we obtain $p' = \mathbf{T}p$

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} a & b & t_x \\ c & d & t_y \\ 0 & 0 & 1 \end{bmatrix}}_{\mathbf{T}} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- $\mathbf{T} = \begin{bmatrix} A & \mathbf{t} \\ \mathbf{0}^t & 1 \end{bmatrix}$, where $A$ is the 4 dof linear component and $\mathbf{t}$ the 2 dof translation

# 2D affine transformation

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} a & b & t_x \\ c & d & t_y \\ 0 & 0 & 1 \end{bmatrix}}_{\mathbf{T}} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- Properties of affine transformations:
  - origin is *not* always transformed to origin
  - parallel lines remain parallel
  - ratios are preserved
  - lines remain lines

# 2D Projective transformations (homographies)

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} a & b & g \\ c & d & h \\ e & f & 1 \end{bmatrix}}_{\mathbf{T}} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

- $\mathbf{T} = \begin{bmatrix} A & \mathbf{t} \\ \mathbf{v}^t & 1 \end{bmatrix}$, where $A$ is the 4 dof affine component, $\mathbf{t}$ the 2 dof translation and $\mathbf{v}$ the 2 dof elation component
- If we consider only $\mathbf{v}$, so $\mathbf{t} = \mathbf{0}$ and $A = \mathcal{I}$:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ e & f & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ ex + fy + 1 \end{bmatrix}$$

$$u = \frac{x}{ex + fy + 1} \quad v = \frac{y}{ex + fy + 1}$$

# 2D Projective transformations (homographies)

$$u = \frac{x}{ex + fy + 1} \quad v = \frac{y}{ex + fy + 1}$$

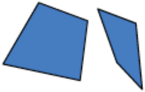
- **Elation**: points are scaled by a scaling factor which is a linear combination of $x$ and $y$. Points can be mapped to (resp. from) infinite to (resp. from ) a finite scalar value. (Ex. if you set f=0 and e=1, then the point $[\infty, \infty]$ is mapped to $[1, 1]$)
- Properties of projective transformations:
  - origin is *not* always transformed to origin
  - parallel lines do *not* necessarily remain parallel
  - ratios, length and angle are *not* preserved
  - lines remain lines

# Examples

A square transforms to:

Projective
8dof

$$\begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix}$$

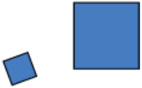


Affine
6dof

$$\begin{bmatrix} a_{11} & a_{12} & t_x \\ a_{21} & a_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix}$$

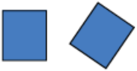
Similarity
4dof

$$\begin{bmatrix} sr_{11} & sr_{12} & t_x \\ sr_{21} & sr_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix}$$



Euclidean
3dof

$$\begin{bmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix}$$

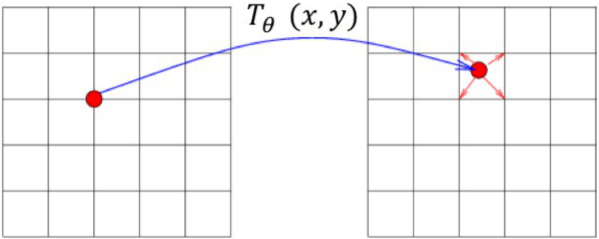


# Summary

# Forward warping

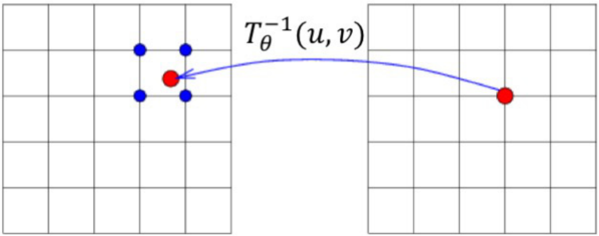
$$\begin{array}{rccc} \mathbf{T}_\phi(x,y): & \mathbb{R}^2 & \rightarrow & \mathbb{R}^2 \\ & (x,y;\phi) & \rightarrow & \mathbf{T}_\phi(x,y) \end{array}$$

- Ideally, if $\Omega_I$ and $\Omega_J$ were continuous domains, we would simply map $(x,y)$ to $(u,v)=\mathbf{T}_\phi(x,y)$ and then compare $I(x,y)$ with $J(u,v)$
- However, $\Omega_I$ and $\Omega_J$ are regular grids ! What if $\mathbf{T}_\phi(x,y)$ is not on the grid $\Omega_J$ ? $\rightarrow$ Splatting: add weighted contribution to neighbor pixels.

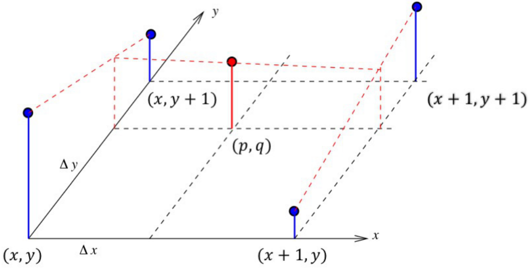


# Inverse warping

- Find the pixel intensities for the deformed image $I_{\mathbf{T}}$ starting from $\Omega_J$: $(x, y) = \mathbf{T}_{\phi}^{-1}(u, v)$
- Assign to $I_{\mathbf{T}}(u, v)$ the pixel intensity in $I(x, y)$
- What if $\mathbf{T}_{\phi}^{-1}(u, v)$ is not on $\Omega_I$ ? $\rightarrow$ Interpolation !

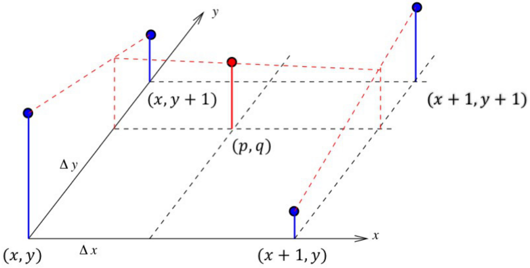


- **Goal**: estimate the intensity value on points not located onto the regular grids
  - nearest neighbor
  - bilinear
  - cubic
  - lanczos
  - ...



# Interpolation

- **Nearest neighbor**: $J(u,v) = I(\text{round}(x), \text{round}(y))$
- **Bilinear**: $\frac{f(x,q)-f(x,y)}{\Delta y} = \frac{f(x,y+1)-f(x,q)}{1-\Delta y} \rightarrow$
  $f(x,q) = (1-\Delta y) f(x,y) + f(x,y+1)\Delta y$. Similarly,
  $f(x+1,q) = (1-\Delta y) f(x+1,y) + f(x+1,y+1)\Delta y$. Then,
  $f(p,q) = f(x,q)(1-\Delta x) + f(x+1,q)\Delta x$



# Interpolation examples

Every time we deform an image, we need to interpolate it. For instance, this is the result on Lena after 10 rotations of 36 degrees:

Figure 3: Original image - Nearest Neighbour - Bilinear

Source : http://bigwww.epfl.ch/demo/jaffine/index.html (Michael Unser)

# Deformation algorithm

Recipe:

- Given a source image $I$ and a global transformation $\mathbf{T}$, compute the forward warping of $I$. The min and max values of the $x$ and $y$ coordinates of the resulting deformed image $I_{\mathbf{T}}$ will give the bounding box
- Given the bounding box of $I_{\mathbf{T}}$, create a new grid within it with, for instance, the same shape of $\Omega_J$ to allow direct comparison between $I_{\mathbf{T}}$ and $J$
- Use the inverse warping and interpolation to compute the intensity values at the grid points of the warped image $I_{\mathbf{T}}$ (we avoid holes)
- Be careful! During the inverse warping, points that are mapped outside $\Omega_I$ are rejected.

# Recap - Matrix inversion

- The inverse of a square (invertible) matrix $\mathbf{T}$ can be computed as the ratio between the adjoint of $\mathbf{T}$ and its determinant: $\mathbf{T}^{-1} = \mathsf{adj}(T)/\det(T)$
- The adjoint $\mathsf{adj}(\mathbf{T})$ of $\mathbf{T}$ is the transpose of its cofactor matrix
- Given $T = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$ its cofactor matrix $\mathbf{C}$ is:

$$\mathbf{C} = \begin{pmatrix} +\begin{vmatrix} a_{22} & a_{23} \\ a_{32} & a_{33} \end{vmatrix} & -\begin{vmatrix} a_{21} & a_{23} \\ a_{31} & a_{33} \end{vmatrix} & +\begin{vmatrix} a_{21} & a_{22} \\ a_{31} & a_{32} \end{vmatrix} \\ \\ -\begin{vmatrix} a_{12} & a_{13} \\ a_{32} & a_{33} \end{vmatrix} & +\begin{vmatrix} a_{11} & a_{13} \\ a_{31} & a_{33} \end{vmatrix} & -\begin{vmatrix} a_{11} & a_{12} \\ a_{31} & a_{32} \end{vmatrix} \\ \\ +\begin{vmatrix} a_{12} & a_{13} \\ a_{22} & a_{23} \end{vmatrix} & -\begin{vmatrix} a_{11} & a_{13} \\ a_{21} & a_{23} \end{vmatrix} & +\begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix} \end{pmatrix},$$

# Transformation parameters

- Given a transformation $\mathbf{T}$ defined by a set of parameter $\theta$ and two images $I$ and $J$, how do we estimate $\theta$ ?

# Transformation parameters

- Given a transformation $\mathbf{T}$ defined by a set of parameter $\theta$ and two images $I$ and $J$, how do we estimate $\theta$ ?
- By minimizing a cost function:

$$\theta^* = \arg\min_{\theta} d(I_{\mathbf{T}}, J) \tag{3}$$

- The similarity measure $d$ might be based on the pixel intensities and/or on corresponding geometric objects such as control points (i.e. landmarks), curves or surfaces

# Summary

**Same modality**

- Sum of squared intensity differences (SSD). Best measure when $I$ and $J$ only differ by Gaussian noise. Very sensitive to "outliers" pixels, namely pixels whose intensity difference is very large compared to others.

$$d(I_{\mathbf{T}}, J) = \sum_u \sum_v (J(u,v) - I(\mathbf{T}_\phi^{-1}(u,v)))^2 = (J(u,v) - I_{\mathbf{T}}(u,v))^2 \tag{4}$$

- Correlation coefficient. Assumption is that there is a linear relationship between the intensity of the images

$$d(I_{\mathbf{T}}, J) = \frac{\sum_u \sum_v (J(u,v) - \bar{J})(I_{\mathbf{T}}(u,v) - \bar{I}_{\mathbf{T}})}{\sqrt{\sum_u \sum_v (J(u,v) - \bar{J})^2 \sum_u \sum_v (I_{\mathbf{T}}(u,v) - \bar{I}_{\mathbf{T}})^2}} \tag{5}$$

**Multi modality**

- Mutual information. We first need to define the joint histogram between $I_{\mathbf{T}}$ and $J$. The value in $(a,b)$ is equal to the number of locations $(u,v)$ that have intensity $a$ in $I_{\mathbf{T}}(u,v)$ and intensity $b$ in $J(u,v)$. For example, a joint histogram which has the value of 2 in the position (4,3) means that we have found two locations $(u,v)$ where the intensity of the first image was 4 ($I_{\mathbf{T}}(u,v)=4$) and the intensity of the second was 3 ($J(u,v)=3$). By dividing by the total number of pixels $N$, we obtain a joint probability density function (pdf) $p_{I_{\mathbf{T}},J}$.
- The sum over the rows or columns gives the marginal pdf of $J$ ($p_J$) and of $I_{\mathbf{T}}$ ($p_{I_{\mathbf{T}}}$) respectively

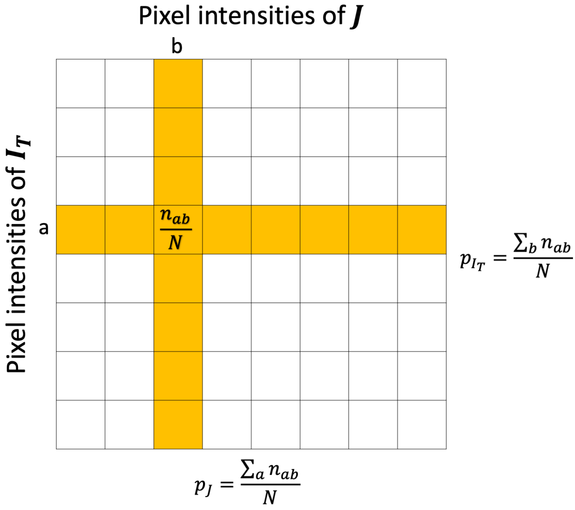



Figure 4: Normalized joint histogram example. $n_{ab}$ indicates the number of locations where the intensity of $I_T$ is equal to $a$ and the intensity of $J$ is equal to $b$

# Intensity based registration - similarity measures

- We suppose that the pixels of $I_T$ and $J$ take only 7 different intensity values, or that we can group them into 7 bins
- What's the difference between the two normalized histograms ? Which one represents a perfect alignment ?

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0.5 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0.1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0.15 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0.1 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0.1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0.1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0.2 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0.2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0.01 | 0.03 | 0.05 |
| 0 | 0 | 0.04 | 0.01 | 0.1 | 0.05 | 0.01 | 0.05 |
| 0 | 0 | 0.02 | 0.01 | 0.05 | 0.01 | 0.01 | 0.05 |
| 0 | 0 | 0.02 | 0 | 0.01 | 0.04 | 0.01 | 0.01 |
| 0 | 0.01 | 0.2 | 0.01 | 0.03 | 0.01 | 0.01 | 0.01 |
| 0.02 | 0.01 | 0.01 | 0.01 | 0.01 | 0.01 | 0.01 | 0 |
| 0.01 | 0.02 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0.02 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

# Intensity based registration - similarity measures

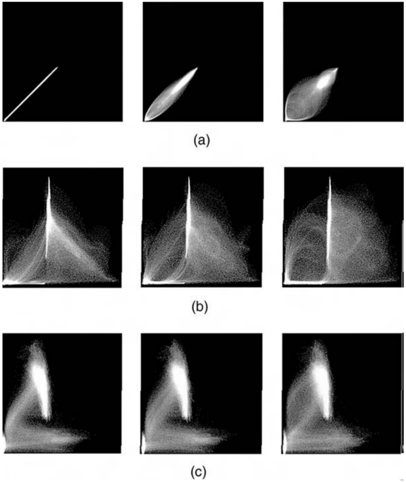



Figure 5: Joint histogram of a) same modality (IRM) b) different modality (MR-CT) c) different modality (MR-PET). First column, images are aligned. 2nd and 3rd columns images are translated. Taken from [2].

- The definition of joint entropy is:

$$H(I_{\mathbf{T}}, J) = -\sum_a \sum_b p_{I_{\mathbf{T}},J}(a,b) \log(p_{I_{\mathbf{T}},J}(a,b)) \tag{6}$$

- where $a$ and $b$ are defined within the range of intensities in $I_{\mathbf{T}}$ and $J$ respectively
- The individual entropies of $I_{\mathbf{T}}$ and $J$ are: $H(I_{\mathbf{T}}) = -\sum_a p_{I_{\mathbf{T}}}(a) \log(p_{I_{\mathbf{T}}}(a))$ and $H(J) = -\sum_b p_J(b) \log(p_J(b))$ respectively.
- We know that $H(I_{\mathbf{T}}, J) \leq H(I_{\mathbf{T}}) + H(J)$ and the more similar the distributions $p_{I_{\mathbf{T}}}$ and $p_J$, the lower the joint entropy compared to the sum of individual entropies

# Intensity based registration - similarity measures

- The definition of Mutual information is:

$$M(I_{\mathbf{T}},J)=H(I_{\mathbf{T}})+H(J)-H(I_{\mathbf{T}},J) \tag{7}$$

- It results:

$$M(I_{\mathbf{T}},J)=\sum_a\sum_b p_{I_{\mathbf{T}},J}(a,b)\log\frac{p_{I_{\mathbf{T}},J}(a,b)}{p_{I_{\mathbf{T}}}(a)p_J(b)} \tag{8}$$

- It can be seen as the Kullback–Leibler divergence between $p_{I_{\mathbf{T}},J}$ and $p_{I_{\mathbf{T}}}\otimes p_J$. It measures the cost for considering $I_{\mathbf{T}}$ and $J$ as independent random variables, when in reality they are not.

$$M(I_{\mathbf{T}},J)=H(I_{\mathbf{T}})+H(J)-H(I_{\mathbf{T}},J)=H(J)-H(J|I_{\mathbf{T}}) \tag{9}$$

- where the conditional entropy $H(J|I_{\mathbf{T}})=-\sum_a \sum_b p_{I_{\mathbf{T}},J}(a,b)\log p_{J|I_{\mathbf{T}}}(b|a)$.
- Mutual information measures the amount of uncertainty about $J$ minus the uncertainty about $J$ when $I_{\mathbf{T}}$ is known, that is to say, how much we reduce the uncertainty about $J$ after observing $I_{\mathbf{T}}$. It is maximized when the two images are aligned. [8]
- Maximizing $M$ means finding a transformations $\mathbf{T}$ that makes $I_{\mathbf{T}}$ the best predictor for $J$. Or, equivalently, knowing the intensity $I_{\mathbf{T}}(u,v)$ allows us to perfectly predict $J(u,v)$.

# Intensity based registration - optimization procedure

- Pixel-based similarity measures need an **iterative** approach where an initial estimate of the transformation is gradually refined using the gradient and, depending on the method, also the Hessian of the similarity measure with respect to the parameters $\theta$
- Possible algorithms: gradient descent, Gauss-Newton, Newton-Raphson, Levenberg-Marquardt, etc.
- Problem of the "local minima" $\rightarrow$ stochastic optimization, line search, trust region, multi-resolution (first low resolution and then higher resolution)
- **Validation** $\rightarrow$ visual inspection, alignment of manually segmented objects, value of similarity measure

# Intensity based registration

- The goal is to minimize a similarity measure $d$ between a transformed source image $I_{\mathbf{T}}$ and a target image $J$ with respect to the parameters $\theta$ of the transformation $\mathbf{T}$
- If we choose the SSD as similarity measure, it results:

$$\begin{aligned}\theta^* = \arg\min_{\theta} \sum_u \sum_v (I(\mathbf{T}_{\phi}^{-1}(u,v)) - J(u,v))^2 = \\ \sum_u \sum_v (I_{\mathbf{T}}(u,v;\theta) - J(u,v))^2\end{aligned} \tag{10}$$

- This a non-linear optimization procedure even if the transformation $\mathbf{T}$ is linear in $\theta$ because the pixel intensities are (in general) not (linearly) related to the pixel coordinates $(u,v)$

# Intensity based registration - Lucas-Kanade Algorithm

- Probably the first image registration algorithm was the Lucas-Kanade one (1981)
- We start with an initial guess for the parameters $\theta$ and we look for the best increment to the parameters $\Delta\theta$, by minimizing:

$$\Delta\theta^* = \arg\min_{\Delta\theta} \sum_u \sum_v (I_{\mathbf{T}}(u,v;\theta+\Delta\theta) - J(u,v))^2 \tag{11}$$

- After that, the parameters are updated as: $\theta^* = \theta + \Delta\theta^*$
- These two steps are iterated until convergence (typically $||\Delta\theta||_{<\epsilon}$)

- We first linearize the non-linear expression in Eq.11 by performing a first order Taylor expansion on $I_{\mathbf{T}}(u,v;\theta+\Delta\theta))$, obtaining:

$$\sum_u \sum_v (I_{\mathbf{T}}(u,v;\theta) + \nabla I_{\mathbf{T}}(u,v;\theta)^T \frac{\partial \mathbf{T}}{\partial \theta} \Delta\theta - J(u,v))^2 \tag{12}$$

- Reminder: the first order Taylor expansion of a composite scalar function is:

$$f(g(x+h)) \approx f(g(x)) + f'(g(x))g'(x)h \tag{13}$$

$$\sum_u \sum_v (I_{\mathbf{T}}(u,v;\theta) + \nabla I_{\mathbf{T}}(u,v;\theta)^T \frac{\partial \mathbf{T}}{\partial \theta} \Delta\theta - J(u,v))^2 \tag{14}$$

- $\nabla I_{\mathbf{T}}(u,v;\theta) = \left(\frac{\partial I_{\mathbf{T}}(u,v;\theta)}{\partial u}, \frac{\partial I_{\mathbf{T}}(u,v;\theta)}{\partial v}\right)^T$ is a $[2 \times 1]$ column vector and is the *gradient* of the image $I$ evaluated at $\mathbf{T}_\phi^{-1}(u,v)$. This means computing the gradient of $\nabla I$ in the coordinate frame of $I$ and then warp it back onto the coordinate frame of $J$ using the current estimate of $\mathbf{T}$

# Reminder - Image gradient

- The image gradient can be computed as the convolution of the original image $I$ with one filter for the x direction and one for the y direction
- $\nabla I = (\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y})^T$ where $\frac{\partial I}{\partial x} = G_x * I$ and $\frac{\partial I}{\partial y} = G_y * I$
- Common choices for $G_x$ and $G_y$ are:
  - **Sobel**: $G_x = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$ and $G_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$
  - **Scharr**: $G_x = \begin{bmatrix} 3 & 0 & -3 \\ 10 & 0 & -10 \\ 3 & 0 & -3 \end{bmatrix}$ and $G_y = \begin{bmatrix} 3 & 10 & 3 \\ 0 & 0 & 0 \\ -3 & -10 & -3 \end{bmatrix}$

$$\sum_u \sum_v (I_{\mathbf{T}}(u,v;\theta) + \nabla I_{\mathbf{T}}(u,v;\theta)^T \frac{\partial \mathbf{T}}{\partial \theta} \Delta\theta - J(u,v))^2 \tag{15}$$

- $\frac{\partial \mathbf{T}}{\partial \theta}$ is a $[2$ x $d]$ matrix where $d$ is the number of parameters $\theta$ and is the *Jacobian* of the transformation. Let $\mathbf{T}(u,v;\theta) = (T_u(u,v;\theta), T_v(u,v;\theta))^T$ be a 2D column vector then:

$$\frac{\partial \mathbf{T}}{\partial \theta} = \begin{bmatrix} \frac{\partial T_u}{\partial \theta_1} & \frac{\partial T_u}{\partial \theta_2} & \dots & \frac{\partial T_u}{\partial \theta_d} \\ \frac{\partial T_v}{\partial \theta_1} & \frac{\partial T_v}{\partial \theta_2} & \dots & \frac{\partial T_v}{\partial \theta_d} \end{bmatrix} \tag{16}$$

- We follow the notational convention that the partial derivatives with respect to a column vector are laid out as a row vector. This convention has the advantage that the chain rule results in a matrix multiplication as in Eq.15

- By substituting the linearization in the original cost function Eq.11, we obtain:

$$\Delta\theta^* = \arg\min_{\Delta\theta} \sum_u \sum_v (I_{\mathbf{T}}(u,v;\theta) + \nabla I_{\mathbf{T}}(u,v;\theta)^T \frac{\partial \mathbf{T}}{\partial \theta} \Delta\theta - J(u,v))^2 \tag{17}$$

- The partial derivative with respect to $\Delta\theta$ is:

$$\begin{aligned} 2 \sum_u \sum_v \left( \nabla I_{\mathbf{T}}(u,v;\theta)^T \frac{\partial \mathbf{T}}{\partial \theta} \right)^T & \\ (I_{\mathbf{T}}(u,v;\theta) + \nabla I_{\mathbf{T}}(u,v;\theta)^T \frac{\partial \mathbf{T}}{\partial \theta} \Delta\theta - J(u,v)) & \end{aligned} \tag{18}$$

# Intensity based registration - Lucas-Kanade Algorithm

- Setting equal to zero the previous Eq., we obtain:

$$\Delta\theta = H^{-1} \sum_u \sum_v \left( \nabla I_{\mathbf{T}}(u,v;\theta)^T \frac{\partial \mathbf{T}}{\partial \theta} \right)^T (J(u,v) - I_{\mathbf{T}}(u,v;\theta)) \tag{19}$$

- where $H$ is a $[d \times d]$ matrix and is an approximation of the Hessian matrix. This is a *Gauss-Newton gradient descent algorithm*.

$$H = \sum_u \sum_v \left( \nabla I_{\mathbf{T}}(u,v;\theta)^T \frac{\partial \mathbf{T}}{\partial \theta} \right)^T \left( \nabla I_{\mathbf{T}}(u,v;\theta)^T \frac{\partial \mathbf{T}}{\partial \theta} \right) \tag{20}$$

- Please note that if we approximate $H$ with an identity function, we obtain the *steepest descent parameter updates*

- Let $d$ the number of parameters for the transformation $T$ and $n$ the number of pixels in $J$, the total computational cost of each iteration is $O(nd^2 + d^3)$
- The two most expensive steps are: computing the Hessian matrix ($O(nd^2)$) and inverting it ($O(d^3)$).
- The Lucas-Kanade Algorithm is one possible solution but other approaches exist. See [9] for more details.

# Intensity based registration - Lucas-Kanade Algorithm

**Algorithm 1** Lucas-Kanade Algorithm

Get $(r,c)$ the number of rows and columns of $J$, initialize $\theta_0$, maximum number of iterations $K$ and iteration index $k=0$

**while** $||\Delta\theta||_2 >= \epsilon$ and $||J(u,v) - I_{\mathbf{T}}(u,v;\theta)||_2 >= \tau$ and $k < K$ **do**

  Initialize $\Delta\theta = (0,...,0)^T$, $H = ((0,0),(0,0))$

  **for** $u = 1$ to $r$ **do**

    **for** $v = 1$ to $c$ **do**

      Compute $I_{\mathbf{T}}(u,v;\theta_k)$ and $\nabla I_{\mathbf{T}}(u,v;\theta_k)$

      Evaluate the Jacobian $\frac{\partial \mathbf{T}}{\partial \theta}(u,v;\theta)$

      $\Delta\theta \mathrel{+}= \left(\nabla I_{\mathbf{T}}(u,v;\theta_k)^T \frac{\partial \mathbf{T}}{\partial \theta}(u,v;\theta_k)\right)^T (J(u,v) - I_{\mathbf{T}}(u,v;\theta_k))$

      $H \mathrel{+}= \left(\nabla I_{\mathbf{T}}(u,v;\theta_k)^T \frac{\partial \mathbf{T}}{\partial \theta}\right)^T \left(\nabla I_{\mathbf{T}}(u,v;\theta_k)^T \frac{\partial \mathbf{T}}{\partial \theta}\right)$

    **end for**

  **end for**

  $\Delta\theta = H^{-1}\Delta\theta$

  $k = k+1$

  $\theta_k = \theta_{k-1} + \Delta\theta$

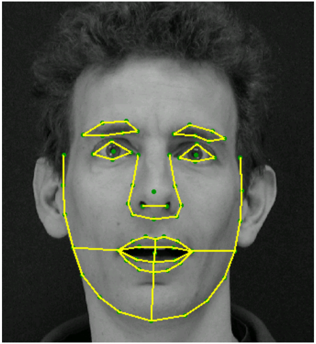
**end while**

# Summary

# Anatomical landmarks

## Definition: anatomical landmark

An anatomical landmark is a point precisely defined onto an anatomical structure which establishes a correspondence among the population of homologous anatomical objects



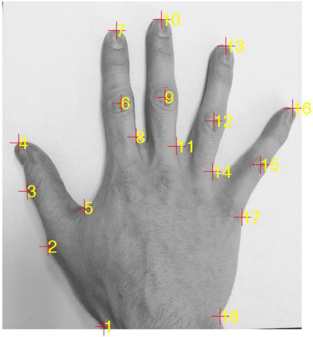



Figure 6: Example of manually labeled landmarks

# Anatomical landmarks

- Given a set of $N$ landmarks $\mathbf{i}$ defined on $I$ and $N$ corresponding landmarks $\mathbf{j}$ defined on $J$, we seek to minimize:

$$\theta^* = \arg\min_{\theta} \sum_{p=1}^{N} ||\mathbf{T}(i_p) - j_p||^2 \tag{21}$$

- where $\mathbf{T}(i_p)$ means that we apply the deformation $\mathbf{T}$ to the $p$-th landmark $i_p$
- We suppose that all landmarks belong to $\mathbb{R}^2$ (it would be similar for $\mathbb{R}^3$)
- The metric is the Euclidean norm (or Frobenius when using matrices)

# Landmark based affine registration

- We define:

$$\mathbf{T}(i_p) = Ai_p + t \qquad \forall p \in [1, N] \tag{22}$$

- Thus, $\theta = A, t$

$$A^*, t^* = \arg\min_{A,t} f(A,t) = \sum_{p=1}^{N} ||Ai_p + t - j_p||^2 \tag{23}$$

- From which it results:

$$\frac{\partial f}{\partial t} = 2\sum_{p=1}^{N}(Ai_p + t - j_p) = 0 \rightarrow t^* = \bar{j} - A\bar{i} \tag{24}$$

- We notice that if we center the data (i.e. $\tilde{i}_p = i_p - \bar{i}$ and $\tilde{j}_p = j_p - \bar{j}$) then $t^* = 0$. The criterion thus becomes $f = \sum_{p=1}^{N} ||A\tilde{i}_p - \tilde{j}_p||^2$

- Now we differentiate wrt $A$. :

$$\begin{aligned}\frac{\partial f}{\partial A} =& \sum_{p=1}^{N} \frac{\partial ||A\tilde{i}_p||^2}{\partial A} - 2\frac{\partial \langle A\tilde{i}_p, \tilde{j}_p\rangle}{\partial A} \\ =& 2\sum_{p=1}^{N} A\tilde{i}_p\tilde{i}_p^T - \tilde{j}_p\tilde{i}_p^T = 2\sum_{p=1}^{N}(A\tilde{i}_p - \tilde{j}_p)\tilde{i}_p^T = 0\end{aligned} \tag{25}$$

- It results:

$$A^* = \left(\sum_{p=1}^{N} \tilde{j}_p\tilde{i}_p^T\right)\left(\sum_{p=1}^{N} \tilde{i}_p\tilde{i}_p^T\right)^{-1} \tag{26}$$

The matrix $\left(\sum_{p=1}^{N} \tilde{i}_p\tilde{i}_p^T\right)$ is invertible if the landmarks are not all aligned on a straight line.

# Landmark based affine registration

- From a computational point of view, it is easier to use homogeneous coordinates:

$$\mathbf{T}^* = \arg\min_{\mathbf{T}} ||\mathbf{x}\mathbf{T} - \mathbf{y}||_F^2 \tag{27}$$

- where we define

$$\underbrace{\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 \\ & & \vdots & & & \\ x_N & y_N & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_1 & y_1 & 1 \\ & & \vdots & & & \\ 0 & 0 & 0 & x_N & y_N & 1 \end{bmatrix}}_{\mathbf{x}} \underbrace{\begin{bmatrix} A_{11} \\ A_{12} \\ t_1 \\ A_{21} \\ A_{22} \\ t_2 \end{bmatrix}}_{\mathbf{T}} - \underbrace{\begin{bmatrix} u_1 \\ \vdots \\ u_N \\ v_1 \\ \vdots \\ v_N \end{bmatrix}}_{\mathbf{y}} \tag{28}$$

$$\mathbf{T}^* = (\mathbf{x}^T\mathbf{x})^{-1}\mathbf{x}^T\mathbf{y} \tag{29}$$

# Procrustes superimposition (similarity transformation)

- Using the same notation as Schönemann ($R \rightarrow R^T$), we seek to minimize:

$$(s,R,t)^* = \arg\min_{s,R,t} \sum_{p=1}^{N} ||sR^T i_p + t - j_p||_2^2 \tag{30}$$

- where $s$ is a uniform scaling factor (scalar) and $R$ is a rotation matrix. The translation vector $t$ is, as before, equal to $t^* = \bar{j} - sR^T\bar{i}$. Thus, by centering the data ($X_c$,$Y_c$), we obtain:

$$(s,R)^* = \arg\min_{s,R} f(s,R) = \sum_{p=1}^{N} ||sR^T\tilde{i}_p - \tilde{j}_p||_2^2 = ||sX_cR - Y_c||_F^2 \tag{31}$$

- Remember that: $\sum_{p=1}^{N} ||\tilde{i}_p - \tilde{j}_p||_2^2 = ||X_c - Y_c||_F^2$ where $X = [\tilde{i}_1^T; \tilde{i}_2^T; ...; \tilde{i}_N^T]$ and $||X||_F^2 = \mathrm{Tr}(X^TX)$

# Procrustes superimposition (similarity transformation)

- We minimize wrt $s$:

$$\begin{aligned} f(s,R) &\propto s^2||X_cR||_F^2 - 2s\langle X_cR, Y_c\rangle_F \\ \frac{\partial f(s,R)}{\partial s} &= 2s||X_c||_F^2 - 2\langle X_cR, Y_c\rangle_F \\ s^* &= \frac{\langle X_cR, Y_c\rangle_F}{||X_c||_F^2} \end{aligned} \tag{32}$$

- where we use the fact that $R^TR = RR^T = \mathbb{I}$. Substituting into $f$:

$$\begin{aligned} R^* &= \arg\min_R f(R) = -\frac{(\langle X_cR, Y_c\rangle_F)^2}{||X_c||_F^2} \\ &= \arg\max_R |\langle X_cR, Y_c\rangle_F| = |\langle R, X_c^TY_c\rangle_F| \\ &= \arg\max_R |\langle R, U\Sigma V^T\rangle_F| = |\langle U^TRV, \Sigma\rangle_F| = |\langle Z, \Sigma\rangle_F| \end{aligned} \tag{33}$$

- where we use the SVD decomposition $X_c^TY_c = U\Sigma V^T$ and the definition of the trace $\langle X_cR, Y_c\rangle_F = \text{Tr}(R^TX_c^TY_c) = \langle R, X_c^TY_c\rangle_F$

# Procrustes superimposition (similarity transformation)

- Notice that $Z = U^T R V$ is an orthogonal matrix since it is the product of orthogonal matrices. Thus $Z^T Z = \mathbb{I}$ and $z_j^T z_j = 1$. It follows that $z_{ij} \leq 1$.

$$R^* = \arg\max_R |\langle Z, \Sigma \rangle_F| = |\mathsf{Tr}(\Sigma^T Z)| = \\ = |\mathsf{Tr}\left( \begin{bmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{bmatrix} \begin{bmatrix} z_{11} & z_{12} \\ z_{21} & z_{22} \end{bmatrix} \right)| = \sum_{d=1}^{2} \sigma_d z_{dd} \leq \sum_{d=1}^{2} \sigma_d \tag{34}$$

- The maximum is obtained when $z_{dd} = 1 \quad \forall d$, which means when:

$$Z = U^T R V = I \rightarrow R^* = U V^T \tag{35}$$

- In order to be sure that $R$ is a rotation matrix ($\det(R) = 1$), we compute [5]:

$$R^* = U \begin{bmatrix} 1 & 0 \\ 0 & \det(UV^T) \end{bmatrix} V^T = USV^T \tag{36}$$

- To recap [5]:

$$
\begin{aligned}
R^* &= USV^T \\
s^* &= \frac{\langle R, Y_c X_c^T \rangle_F}{||X_c||_F^2} = \frac{\mathsf{Tr}(S\Sigma)}{||X_c||_F^2} \\
t^* &= \bar{j} - \frac{1}{N} \sum_{p=1}^{N} sRi_p
\end{aligned} \tag{37}
$$

# Non-linear registration (small displacement)

- We define the deformation of a pixel at location $z = (x, y)$ as:

$$\mathbf{T}(z) = z + v(z) \quad \text{with} \quad v(z) = \sum_{p=1}^{N} K(z, i_p)\alpha_p \tag{38}$$

- where $K(z, i_p)$ is a kernel, for instance $K(z, i_p) = \exp\left(-\frac{||z-i_p||_2^2}{\lambda^2}\right)$ and $\alpha_p$ is a 2D vector which need to be estimated.
- The displacement at any point $z$ depends on the displacement of the neighbor landmarks

# Non-linear registration (small displacement)

- We minimize

$$\begin{aligned}\boldsymbol{\alpha}^* = \arg\min_{\boldsymbol{\alpha}} f(\alpha;\lambda) &= \sum_{p=1}^{N} ||i_p + v(i_p) - j_p||_2^2 \\ &= \sum_{p=1}^{N} ||i_p + (\sum_{d=1}^{N} K(i_p, i_d)\alpha_d) - j_p||_2^2 \\ &= ||\mathbf{i} + \mathbf{K}\boldsymbol{\alpha} - \mathbf{j}||_F^2\end{aligned} \tag{39}$$

- where $\mathbf{K} = \begin{bmatrix} 1 & K(i_1,i_2) & ... & K(i_1,i_N) \\ K(i_2,i_1) & 1 & ... & K(i_2,i_N) \\ ... & ... & ... & ... \\ K(i_N,i_1) & K(i_N,i_2) & ... & 1 \end{bmatrix}$ and $\boldsymbol{\alpha} = [\alpha_1^T; ...; \alpha_N^T]$

- By differentiating wrt $\boldsymbol{\alpha}$:

$$\frac{\partial ||\mathbf{i} + \mathbf{K}\boldsymbol{\alpha} - \mathbf{j}||_F^2}{\partial \boldsymbol{\alpha}} = 2\mathbf{K}^T(\mathbf{i} + \mathbf{K}\boldsymbol{\alpha} - \mathbf{j}) = 0$$
$$\boldsymbol{\alpha}^* = \mathbf{K}^{-1}(\mathbf{j} - \mathbf{i}) \quad (40)$$

- The matrix $\mathbf{K}$ might not always be invertible (if $\lambda$ is too big for instance). We need to regularize it. A possible solution is to use a Tikhonov matrix such as $\boldsymbol{\alpha}^T\mathbf{K}\boldsymbol{\alpha}$, thus obtaining:

$$\boldsymbol{\alpha}^* = \arg\min_{\boldsymbol{\alpha}} f(\alpha; \lambda, \gamma) = ||\mathbf{i} + \mathbf{K}\boldsymbol{\alpha} - \mathbf{j}||_F^2 + \gamma\boldsymbol{\alpha}^T\mathbf{K}\boldsymbol{\alpha} \quad (41)$$

$$\frac{\partial f(\alpha; \lambda, \gamma)}{\partial \boldsymbol{\alpha}} = 2\mathbf{K}^T(\mathbf{i} + \mathbf{K}\boldsymbol{\alpha} - \mathbf{j}) + 2\gamma\mathbf{K}\boldsymbol{\alpha} = 0$$
$$\boldsymbol{\alpha}^* = (\mathbf{K} + \gamma\mathbb{I})^{-1}(\mathbf{j} - \mathbf{i}) \quad (42)$$

# Non-linear registration (small displacement)

- Why only small displacement ? $\rightarrow$ We could approximate the inverse of $\mathbf{T}(z)$ as $\mathbf{T}^{-1}(z') = z' - v(z')$ obtaining:

$$\mathbf{T}(\mathbf{T}^{-1}(z')) = z' - v(z') + v(z' - v(z')) \neq z' \tag{43}$$

- The error is small only if $v(z' - v(z')) - v(z')$ is small, which is the case only when the displacement is small !
- We might have intersections, holes or tearing in area where the displacement is large

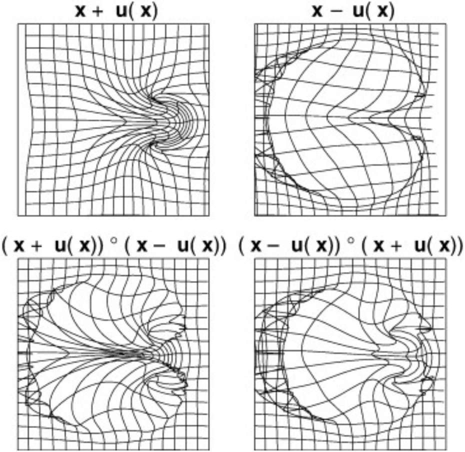



Figure 7: First row: forward and inverse transformation. The first one is a one-to-one mapping whereas the second one presents intersections. Second row: composition of transformations. They should be the identity transforms. Image taken from [7].

# Large-displacement registration: diffeomorphism

- Instead than using small-displacement transforms we should use **diffeomorphisms**
- A diffeomorphism is a differentiable (smooth and continuous) bijective transformation (one-to-one) with differentiable inverse (i.e. nonzero Jacobian determinant)
- Using diffeomorphic transformations we can preserve the topology and spatial organization, namely no intersection, folding or shearing may occur

# Diffeomorphism

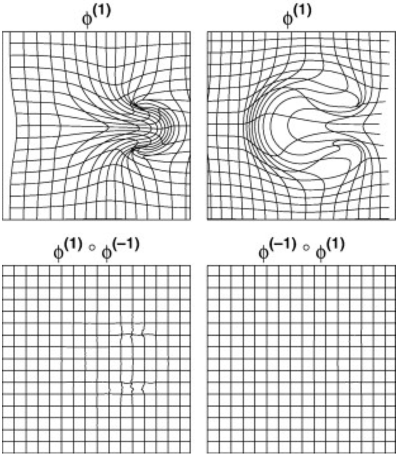
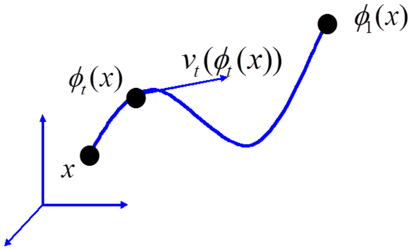


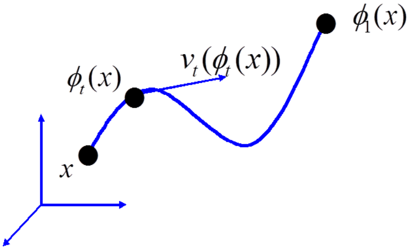

Figure 8: First row: forward and inverse diffeomorphic transformation (both are one-to-one). Second row: composition of forward and inverse transformations. The result is the identity transform (i.e. no deformation). Image taken from [7].

# Diffeomorphism

- One of the most used algorithm in medical imaging to create diffeomorphic deformations is called LDDMM: Large Deformation Diffeomorphic Metric Mapping
- Deformations are built by integrating a time-varying vector field $v_t(x)$ over $t \in [0, 1]$ where $v_t(x)$ represents the instantaneous velocity of any point $x$ at time $t$ (and no more a displacement vector !)

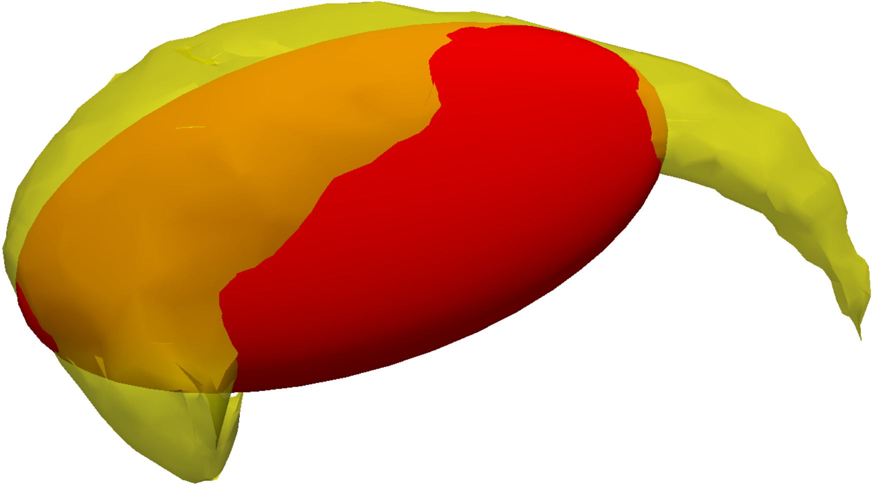


# Diffeomorphism




- Calling $\phi_t(x)$ the position of a point at time $t$ which was located in $x$ at time $t = 0$, its evolution is given by: $\frac{\partial \phi_t(x)}{\partial t} = v_t(\phi_t(x))$ with $\phi_0(x) = x$
- Integrating $\frac{\partial \phi_t(x)}{\partial t} = v_t(\phi_t(x))$ between $t \in [0, 1]$ produces a flow of diffeomorphisms (if $v$ is square integrable). The last diffeomorphism is the one we are interested into.

# Diffeomorphism



# Diffeomorphism


Figure 9: Image taken from T. Mansi - MICCAI - 2009

# Diffeomorphism

http://www.deformetrica.org/

# Diffeomorphism

- The flow of diffeomorphisms produces a dense deformation of the entire 3D space. We know how to deform every point in the space.
- The last diffeomorphism is parametrised by the initial velocity $v_0$
- From a mathematical point of view, to register a source image or mesh $I$ to a target image or mesh $J$ we minimize:

$$\arg\min_{v_0} \mathsf{D}(\phi_1(I), J) + \gamma \mathsf{Reg}(v_0) \tag{44}$$

- where D is a data term, Reg is a regularization term and $\gamma$ their trade-off. We use an optimization scheme (e.g. gradient descent) to estimate the optimal deformation parameters $v_0$.

# References


1. J. Modersitzki (2004). *Numerical methods for image registration*. Oxford university press
2. J. V. Hajnal, D. L.G. Hill, D. J. Hawkes (2001). *Medical image registration*. CRC press
3. G. Wolberg (1990). Digital Image Warping. IEEE
4. The Matrix Cookbook
5. S. Umeyama (1991). Least-squares estimation of transformation parameters.... IEEE TPAMI
6. S. Durrleman et al. (2014). Morphometry of anatomical shape complexes ... . NeuroImage.
7. J. Ashburner (2007). A fast diffeomorphic image registration algorithm. NeuroImage
8. J. P. W. Pluim (2003). Mutual information based registration of medical images: a survey. IEEE TMI
9. S. Baker, I. Matthews (2004). Lucas-Kanade 20 Years On: A Unifying Framework. IJCV