Image Registration

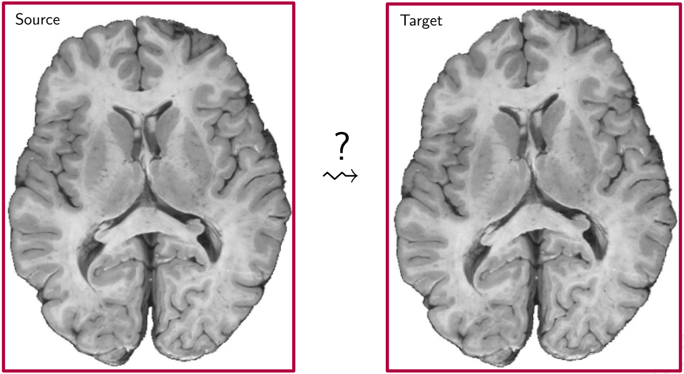


Image courtesy of Marc Niethammer

Image Registration

Purpose

Purpose of registration

Establishing a geometric transformation $h(\mathbf{x})$ of coordinates $\mathbf{x}$

$$\mathbf{x}' = h(\mathbf{x}) = \mathbf{x} + \Delta\mathbf{x}$$

relating points in one image to points in another.

Image Registration

Review – Transformations (Rigid)

A rigid transform is a geometrical transformation that preserves distances (i.e., an isometry)

$$\mathbf{x}' = \mathbf{R}\mathbf{x} + \mathbf{t}$$

where

- $\mathbf{R}$ is a rotation matrix
- $\mathbf{x}$ is a point in $\mathbb{R}^2$ or $\mathbb{R}^3$
- $\mathbf{t}$ is a translation vector

In 3D, $\mathbf{R}$ is a 3×3 orthogonal matrix, i.e.,

$$\mathbf{R}^\top \mathbf{R} = \mathbf{R}\mathbf{R}^\top = \mathbf{I}.$$

Image Registration

Review – Transformations (Rigid)

This class of matrices includes proper (i.e., determinant $+1$) and improper rotations. Improper rotations both *reflect* and *rotate*.

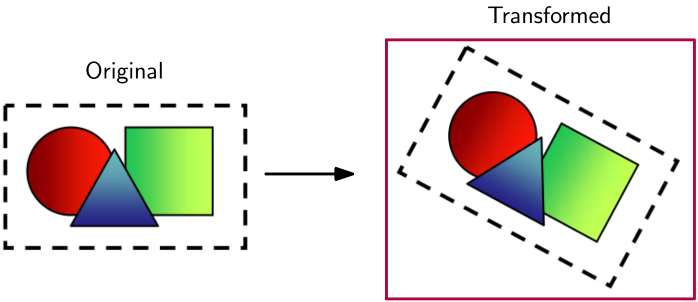


Image Registration

Review – Transformations (Rigid)

Group structure (of the "special orthogonal group" $\mathcal{SO}(3)$):

$$\mathcal{SO}(3) = \{\mathbf{R} \in \mathbb{R}^{3\times 3} : \mathbf{R}^\top \mathbf{R} = \mathbf{I}, \det(\mathbf{R}) = +1\}$$

Proper rotations can be parameterized as 3 rotations. Here is one possible way:

$$\mathbf{R} = \begin{pmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\gamma & -\sin\gamma \\ 0 & \sin\gamma & \cos\gamma \end{pmatrix}$$

We have 3 degrees of freedom (α, β, γ) and thus a rigid transform in 3D has 6 free parameters (i.e., including 3 for translation).

Image Registration

Review – Transformations (Rigid)

Remark

Rigid transformations which *include translation* are *not* linear transforms!

In homogeneous coordinates, though, we can write a rigid transformation as one matrix multiplication

$$\mathbf{T} = \begin{pmatrix} 1 & 0 & 0 & t_x \\ 1 & 0 & 0 & t_y \\ 1 & 0 & 0 & t_z \\ 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} c_\alpha & -s_\alpha & 0 & 0 \\ s_\alpha & c_\alpha & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} c_\beta & 0 & s_\beta & 0 \\ 0 & 1 & 0 & 0 \\ -s_\beta & 0 & c_\beta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & c_\gamma & -s_\gamma & 0 \\ 0 & s_\gamma & c_\gamma & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

with $c_\alpha := \cos\alpha$, $c_\beta := \cos\beta$, $c_\gamma := \cos\gamma$ and $s_\alpha, s_\beta, s_\gamma$ defined appropriately (compare to previous slide).

Image Registration

Review – Transformations (Rigid + Scaling)

The simplest non-rigid transformation is rigid + scaling, i.e.,

$$\mathbf{x}' = \mathbf{R}\mathbf{S}\mathbf{x} + \mathbf{t}$$

with

$$\mathbf{S} = \begin{pmatrix} r_x & 0 & 0 \\ 0 & r_y & 0 \\ 0 & 0 & r_z \end{pmatrix}$$

Note that $\mathbf{x}' = \mathbf{S}\mathbf{R}\mathbf{x} + \mathbf{t}$ produces a different result. Obviously, distances are no longer preserved!

In case scaling is isotropic (i.e., equal in each dimension), we speak of a similarity transform, i.e., $\mathbf{x}' = s\mathbf{R}\mathbf{x} + \mathbf{t}$.

Image Registration

Review – Transformations (Rigid + Scaling)

Illustration of a similarity transform:

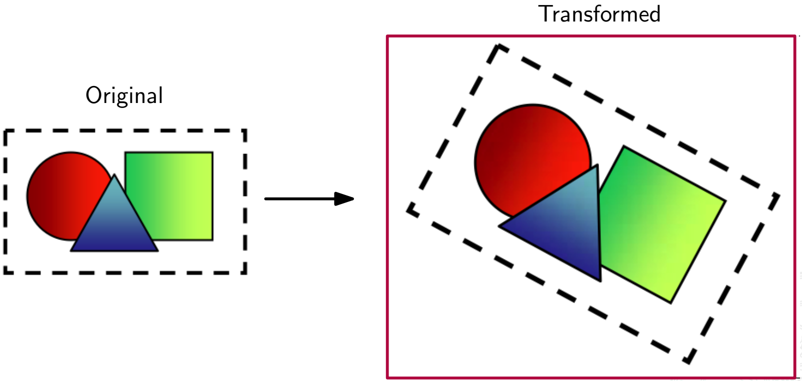


Image Registration

Review – Transformations (Affine)

Scaling transforms are special cases of the more general class of affine transforms

$$\mathbf{x}' = \mathbf{A}\mathbf{x} + \mathbf{t}$$

where there is no restriction on the elements of $\mathbf{A}$. Affine transforms preserve

- straight lines (and hence, the planarity of surfaces)
- parallelism

In homogeneous coordinates, we have

$$\mathbf{T} = \begin{pmatrix} a_{11} & a_{12} & a_{13} & t_x \\ a_{21} & a_{22} & a_{23} & t_y \\ a_{31} & a_{32} & a_{33} & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Image Registration

Review – Transformations (Affine)

Illustration of an affine transform:

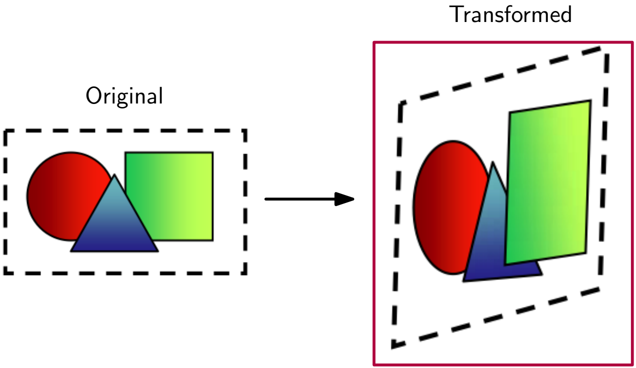


Image Registration

Review – Transformations (Projective)

A projective transform can be written as

$$\mathbf{x}' = \frac{\mathbf{A}\mathbf{x} + \mathbf{t}}{\langle \mathbf{p}, \mathbf{x} \rangle + \alpha}$$

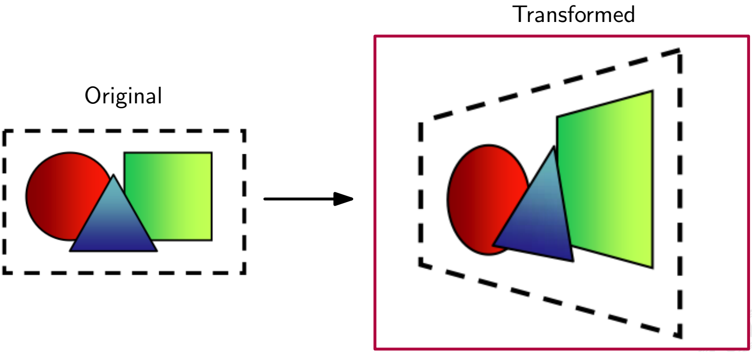
or, in homogeneous coordinates, as

$$\mathbf{T} = \begin{pmatrix} a_{11} & a_{12} & a_{13} & t_x \\ a_{21} & a_{22} & a_{23} & t_y \\ a_{31} & a_{32} & a_{33} & t_z \\ p_1 & p_2 & p_3 & \alpha \end{pmatrix}$$

Image Registration

Review – Transformations (Projective)

Illustration of a projective transform:



Only straight lines (and planarity of surfaces) are preserved!

Image Registration

Review – Transformations (Perspective)

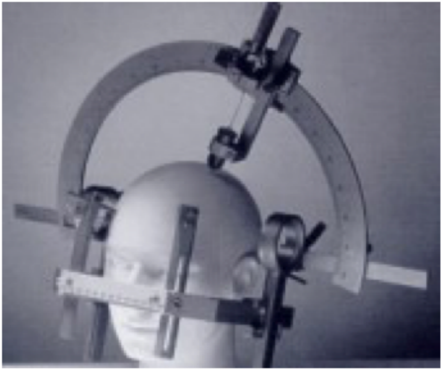
We *do not* cover perspective transforms here (special case of projective transforms of the previous slide).

However, images obtained from X-Ray projection, or microscopy are all two-dimensional views of three-dimensional objects.

Each of these modalities produces a perspective transformation.

Image Registration

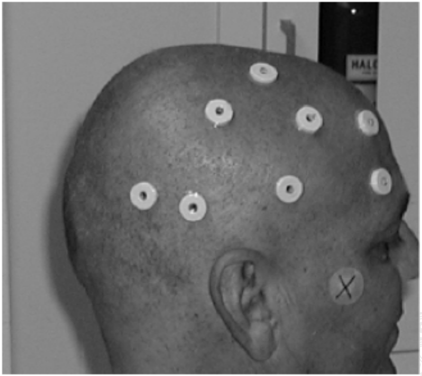
Basics – Extrinsic registration



Stereotactic frame (e.g., used in neurosurgery)

Image Registration

Basics – Extrinsic registration



Markers (e.g., invasive bone fiducials, or non-invasive skin fiducials)

Image Registration

Basics – Extrinsic registration

In either case, we obtain 3D point sets for registration!

Fiducials

Points that can be reliably identified (i.e., clearly discernible features) a-priori are called *fiducial markers*, or simply *fiducials*.

Our focus is on 3D-3D intensity-based registration!

Image Registration

Intensity-based 3D-3D image registration

Initial **T**

Volume A
(fixed image)

Volume B
(moving image)

Image Registration

Intensity-based 3D-3D image registration

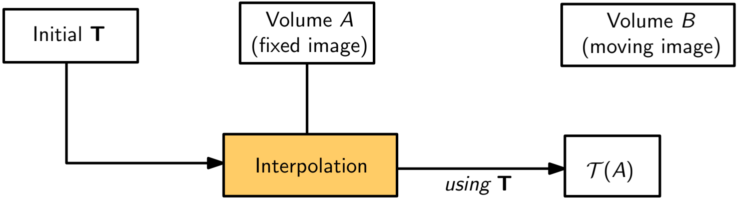


Image Registration

Intensity-based 3D-3D image registration

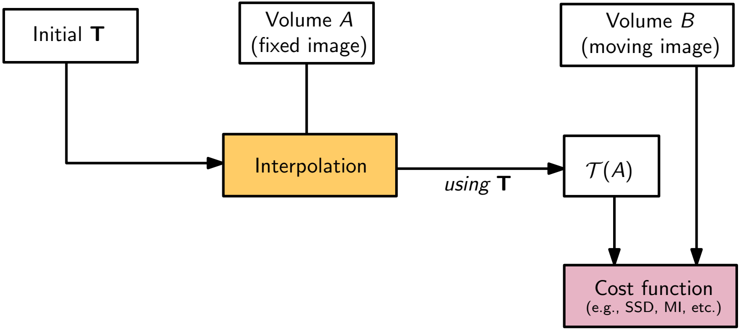


Image Registration

Intensity-based 3D-3D image registration

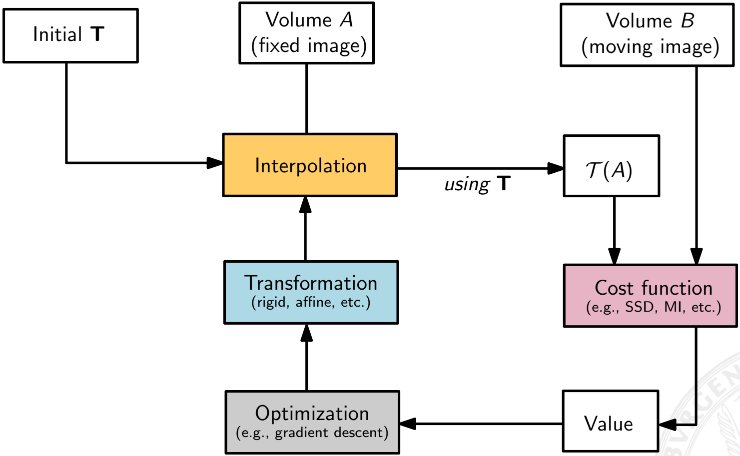


Image Registration

Intensity-based 3D-3D image registration

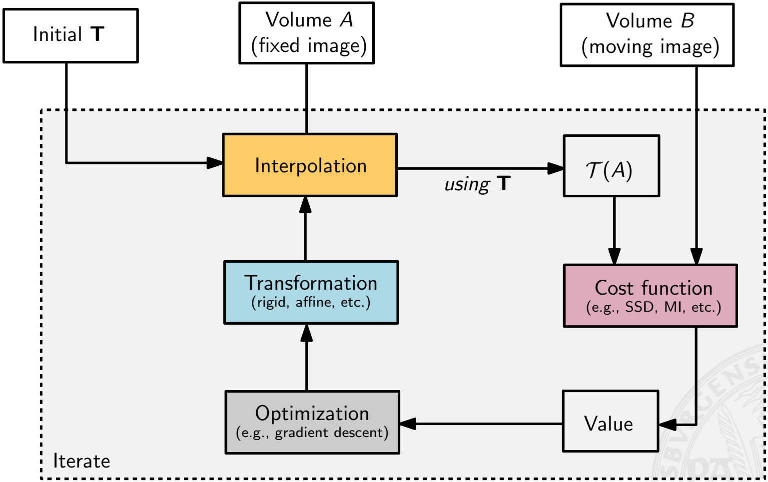


Image Registration

Intensity-based 3D-3D image registration

Intensity-based registration

Intensity-based registration involves calculating a transformation between two images using the pixel or voxel values alone.

The transform is computed by iteratively optimizing a *cost* function between two images, computed from all voxels.

Image Registration

Overview – Components (Optimization)

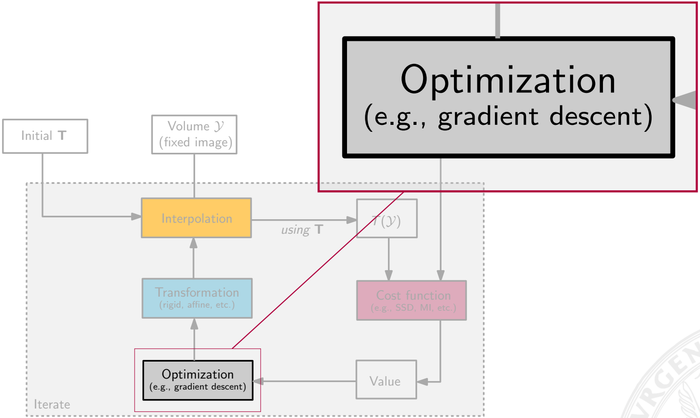


Image Registration

Optimization – Illustration of the principle

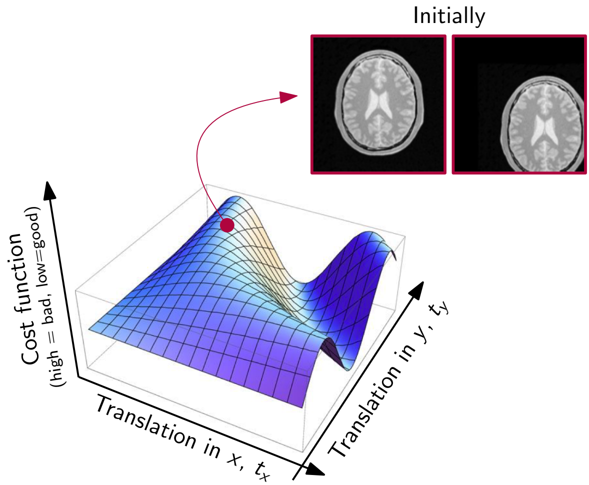


Image Registration

Optimization – Illustration of the principle

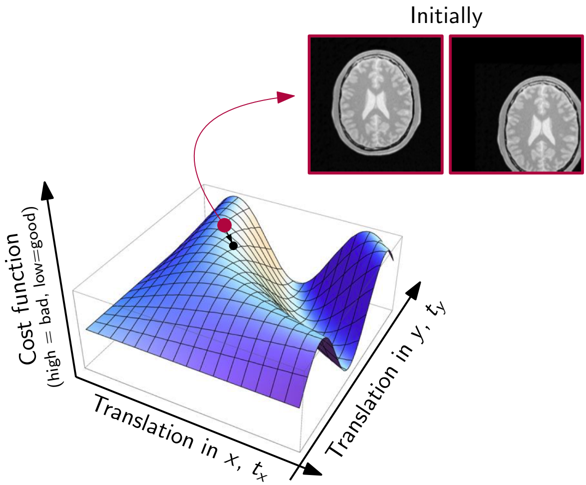


Image Registration

Optimization – Illustration of the principle

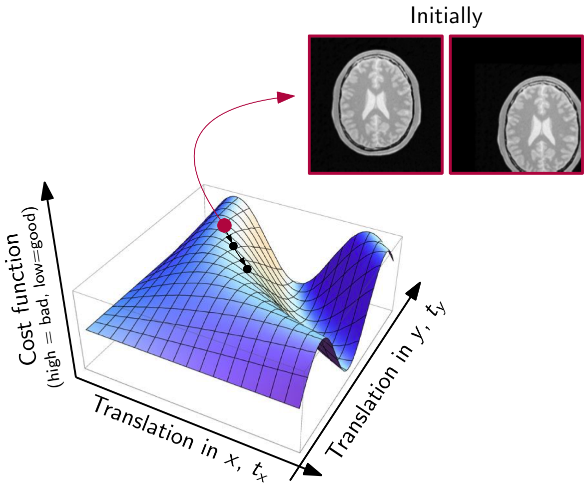


Image Registration

Optimization – Illustration of the principle

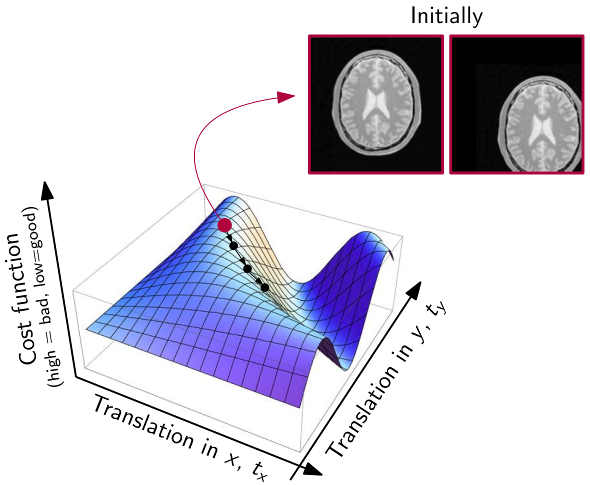


Image Registration

Optimization – Illustration of the principle

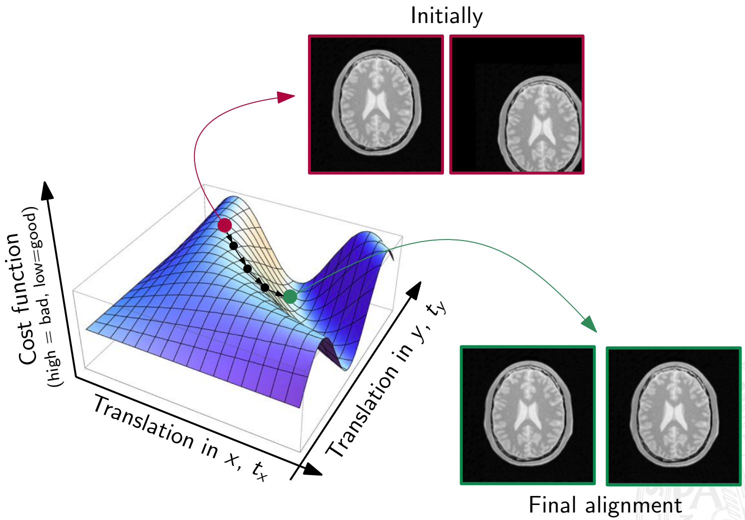


Image Registration

Notation

$\mathbf{x}$	Voxel coordinate
$F(\mathbf{x})$	Fixed image (voxel value at $\mathbf{x}$)
$M(\mathbf{x})$	Moving image (voxel value at $\mathbf{x}$)
$\mathcal{T}(\mathbf{x}, \mathbf{p})$	Transformation function (with parameter $\mathbf{p}$)
$\mathcal{C}(\mathbf{p})$	Cost function between $F(\mathbf{x})$ and $M(\mathcal{T}(\mathbf{x}, \mathbf{p}))$

Caution: Many different types of notation in the literature!

Our objective

Find $\mathbf{p}$ that minimizes $\mathcal{C}(\mathbf{p})$, i.e.,

$$\widehat{\mathbf{p}} = \arg\min_{\mathbf{p}} \mathcal{C}(\mathbf{p})$$

By minimizing cost, we *maximize similarity*.

Image Registration

A more general view

We aim for an iterative parameter update of the form

$$\mathbf{p}^{(n+1)} = \mathbf{p}^{(n)} + a_n \mathbf{d}^{(n)}$$

where a_n is the step size and $\mathbf{d}^{(n)}$ is the search direction. In our previous example, $\mathbf{p} = [p_1, p_2] = [t_x, t_y]$.

When using gradient descent, the search direction is

$$\mathbf{d}^{(n)} = \frac{\partial \mathcal{C}}{\partial \mathbf{p}}(\mathbf{p}^{(n)}) = \mathbf{g}^{(n)} \Rightarrow \mathbf{p}^{(n+1)} = \underbrace{\mathbf{p}^{(n)} - a_n \mathbf{g}^{(n)}}_{\text{in the direction of the negative gradient}}$$

This will obviously depends on the choice of cost function, but illustrates the principle (here, $\mathbf{g}$ denotes the gradient).

Image Registration

Other choices for the search direction

Method	Search direction	Type
Gradient descent	$\mathbf{d}^{(n)} = -\mathbf{g}^{(n)}$	—
Newton	$\mathbf{d}^{(n)} = -[\mathbf{H}^{(n)}]^{-1}\mathbf{g}^{(n)}$	Smart
"quasi" Newton	$\mathbf{d}^{(n)} = -\mathbf{B}^{(n)}\mathbf{g}^{(n)}$	Smart
Stochastic gradient descent	$\mathbf{d}^{(n)} \approx -\mathbf{g}^{(n)}$	Cheap

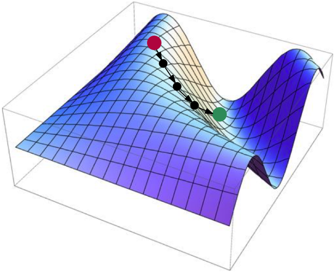
$\mathbf{H}^{(n)}$ ... Hessian matrix, evaluated at $\mathbf{p}^{(n)}$

$\mathbf{B}^{(n)}$ ... Approximation to the Hessian

Image Registration

Smart vs. cheap steps

Smart steps



Cheap steps

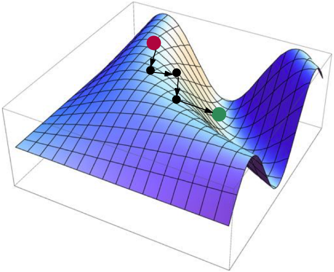


Image Registration

Rigid body registration – Example using Gauss-Newton

Let $b_j(\mathbf{p})$ be the sum-of-squared-difference (SSD), capturing the difference between the moving and fixed image at the j-th voxel.

Our objective is

$$\min_{\mathbf{p}} \mathcal{C}(\mathbf{p}) = \min_{\mathbf{p}} \sum_j b_j(\mathbf{p})^2$$

This sum-of-squared residuals can be iteratively minimized using the Gauss-Newton algorithm as

$$\mathbf{p}^{(n+1)} = \mathbf{p}^{(n)} - \underbrace{(\mathbf{A}^\top \mathbf{A})^{-1} \mathbf{A}^\top \mathbf{b}}_{\mathbf{t}}$$

where (for the update at iteration $n+1$), we have

$$[\mathbf{A}]_{ij} = \frac{\partial b_i}{\partial p_j}(\mathbf{p}^{(n)}) \quad \text{and} \quad \mathbf{b}_j = b_j(\mathbf{p}^{(n)})$$

Image Registration

A more general view

The iterative Gauss-Newton algorithm, with initial estimate $\mathbf{p}^{(0)}$ (close to optimum), stops when the SSD is no longer decreased.

Since, N (number of voxel) is typically $\gg P$ (number of parameters), $(\mathbf{A}^\top \mathbf{A})^{-1}$ is uniquely invertible.

It is important to note that there is *no guarantee for convergence* (not even locally) and we might get stuck in *local minima*.

Image Registration

Step size

Remember:

$$\mathbf{p}^{(n+1)} = \mathbf{p}^{(n)} + a_n \mathbf{d}^{(n)}$$

Some choices of how to select the step size a_n:

1. Constant:

$$a_n = a$$

2. Slowly decaying:

$$a_n = f(n) = \frac{a}{(A+n)^a}$$

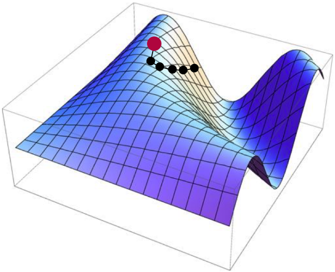
3. Exact "line search":

$$a_n = \arg\min_a \mathcal{C}(\mathbf{p}^{(n)} + a\mathbf{d}^{(n)})$$

Image Registration

Impact of the step size

Too small



Too large

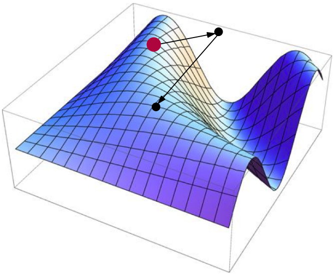


Image Registration

Overview – Cost functions

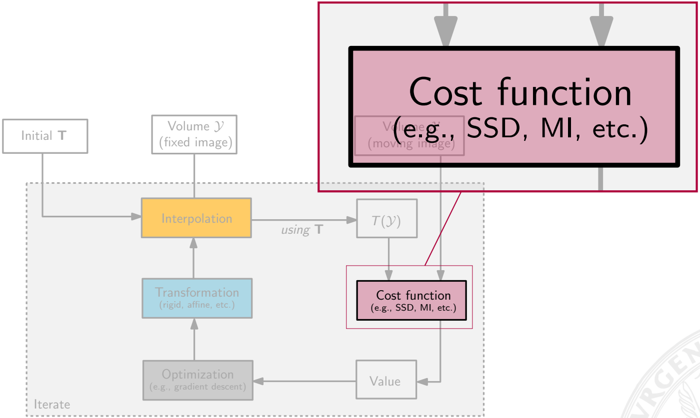


Image Registration

Cost functions

Let $A(i)$ and $B(i)$ be the i-th voxel in images A and B and $B'(i)$ be the same voxel in the transformed image B'.

Sum-of-Squared/Absolute-Differences (SSD/SAD)

$$\mathrm{SSD}(A, B') = \frac{1}{N}\sum_i (A(i) - B'(i))^2, \quad \forall i \in A \cap B'$$

$$\mathrm{SAD}(A, B') = \frac{1}{N}\sum_i |A(i) - B'(i)|, \quad \forall i \in A \cap B'$$

SSD and SAD are two standard choices in image registration and perform well when both images are from the *same modality*.

Image Registration

Cost functions – SSD/SAD

Good example for using SSD: same image, only misalignment

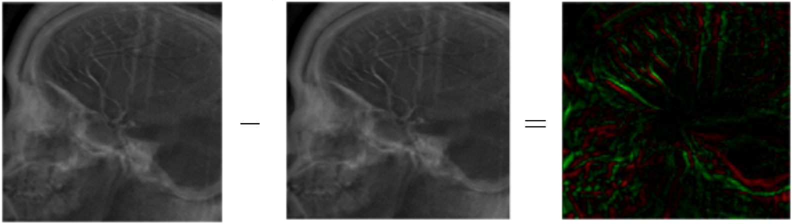
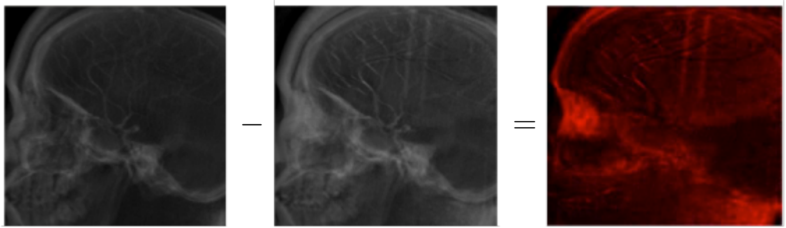


Image Registration

Cost functions – SSD/SAD

Bad example for using SSD: same image, but contrast change!



We get high values in the difference image which can cause problems during optimization and lead to bad registration results.

Image Registration

Cost functions – Correlation coefficient (CC)

Correlation coefficient (CC)

$$\mathrm{CC}(A, B') = \frac{\sum_i [A(i) - \overline{A}][B'(i) - \overline{B'}]}{\sum_i [A(i) - \overline{A}]^2 \sum_i [B'(i) - \overline{B'}]^2}, \quad \forall i \in A \cap B'$$

The CC expresses the linear relationship between voxel values in each image volume.

Image Registration

Cost functions – Information-theoretic approaches (JE, MI)

Given two images A and B (as before), with intensity ranges normalized to $[0, 1]$, the joint histogram J is computed as

$$\underbrace{\mathrm{J}(A, B)[i, j]}_{\text{between } A,\, B \text{ at } (i,j)} = \sum_{\mathbf{x} \in A} \delta(\mathbf{x})$$

with

$$\delta(\mathbf{x}) = \begin{cases} 1, & \text{if } A(\mathbf{x}) \in \left[\frac{i}{B_A}, \frac{i+1}{B_A}\right] \text{ and } B(\mathbf{x}) \in \left[\frac{j}{B_B}, \frac{j+1}{B_B}\right] \\ 0, & \text{otherwise} \end{cases}$$

where B_A and B_B are the bins of the (marginal) histograms of A and B and $A(\mathbf{x})$ denotes the intensity value of A at $\mathbf{x}$.

Image Registration

Cost functions – Information-theoretic approaches

A lot of research has been devoted to devise similarity measures based on the joint histogram of two images.

Why? The observed change in the histogram (with respect to the transform) is qualitatively similar for many modalities.

Motivating observation

The estimated probability density is “clustered” in case of good alignment, but diffuses in case of misregistration.

Image Registration

Cost functions – Information-theoretic approaches

We can use $\mathsf{J}(A, B')$ as an estimator for the joint probability density (PDF) of A and B' and estimate the *joint entropy* from it.

The joint entropy (JE), given our joint histogram, is defined as

$$\mathsf{JE}(A, B') = -\sum_{i,j} \mathsf{J}(A, B')[i, j] \log \mathsf{J}(A, B')[i, j]$$

Image Registration

Cost functions – Information-theoretic approaches

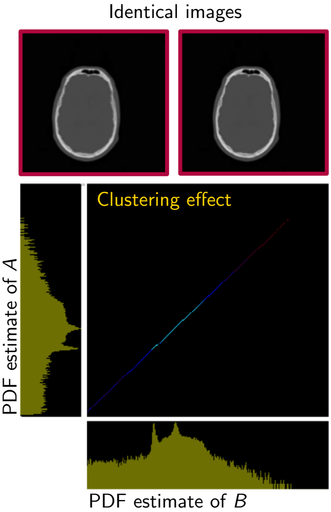


Image Registration

Cost functions – Information-theoretic approaches

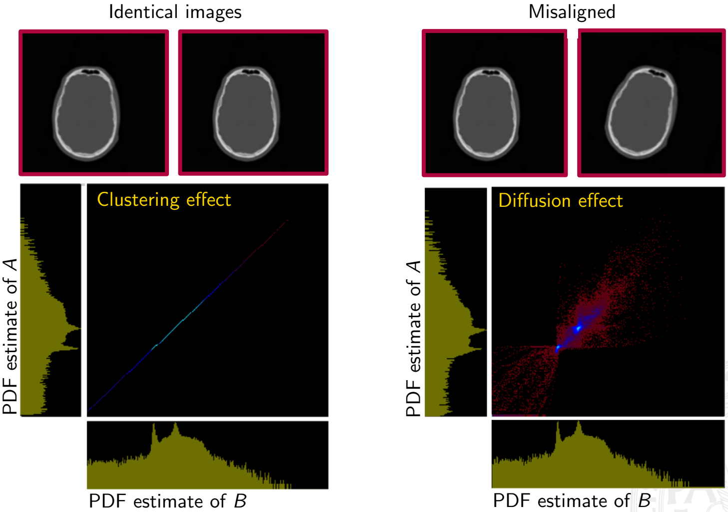


Image Registration

Cost functions – Information-theoretic approaches

We could try to minimize the joint entropy, BUT the PDF is only defined on the *region of overlap* between both images.

The change in overlap, as $\mathcal{T}$ changes, can "mask" the clustering effect and cause problems.

Image Registration

Cost functions – Information-theoretic approaches

Better strategy: Maximization of mutal information (MI)

$$\mathrm{MI}(A, B') = \mathrm{H}(A) + \mathrm{H}(B') - \mathrm{JE}(A, B')$$

where $\mathrm{H}(A)$, $\mathrm{H}(B')$ are the marginal histograms of A and B'.

Any permutation on A, B' voxel values *does not* affect the measure (this is a great property for multimodal registration)!

Maximimization of mutal information is the de-facto standard in image registration today.

Image Registration

Overview – Interpolation

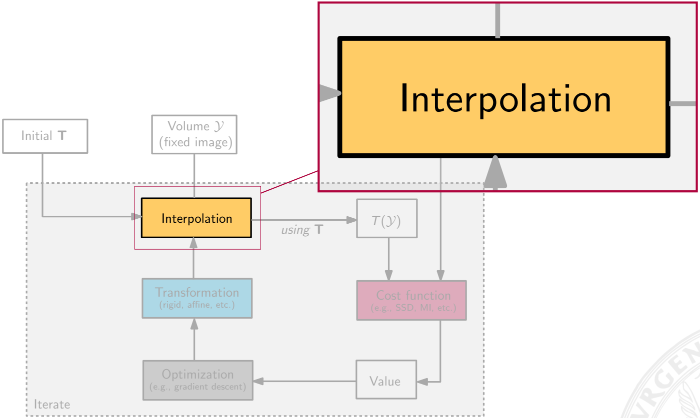


Image Registration

Interpolation – Pushing vs. pulling

An image transformation is typically implemented as a *pulling operation* in practice.

Pulling

Pixel values are pulled from the original image into new location.

vs.

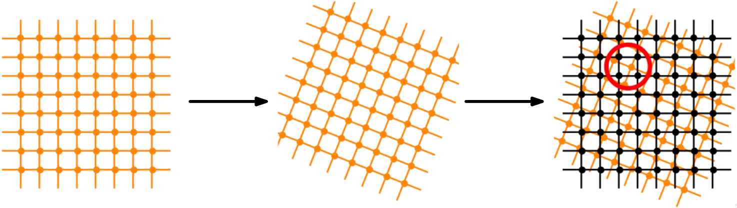
Pushing

Pixels in the original image are pushed into new location.

Image Registration

Interpolation – Pushing vs. pulling

For each voxel in the transformed image, get the value in the original image (will most likely not be on grid $\rightarrow$ interpolate).

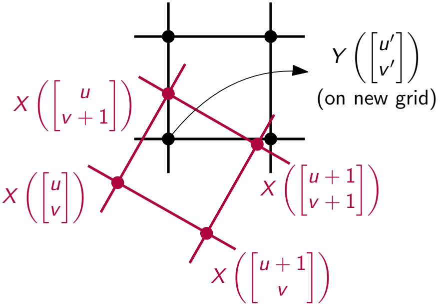


From left to right: Image X, $\mathcal{T}(X, \mathbf{p})$, $Y = \mathcal{T}(X, \mathbf{p})$

Image Registration

Interpolation – Pushing vs. pulling

Detail: Interpolation, in the “backwards” direction, using pulling:



$$\begin{pmatrix}x\\y\end{pmatrix} = \mathcal{T}^{-1}\left(\begin{bmatrix}u'\\v'\end{bmatrix}, \mathbf{p}\right)$$

On the new grid, for $[u', v']^\top$, we get $[x, y]^\top$ in the original image and then interpolate.

Image Registration

Interpolation – Pushing vs. pulling

A simple interpolation strategy is nearest neighbor interpolation:

$$Y\left(\begin{bmatrix}u'\\v'\end{bmatrix}\right) = X\left(\begin{bmatrix}\mathrm{round}(x)\\\mathrm{round}(y)\end{bmatrix}\right)$$

Other choices are bilinear interpolation, or polynomial interpolation for instance.

Image Registration

Conventions (in ITK, used in many state-of-the-art approaches)

Moving vs. fixed image / Physical space

The moving image is expected to be resampled on the grid of the fixed image. Registration is done in *physical space*.

Strategy:

1. Go through every pixel in fixed image
2. Compute intensity that should be assigned to that pixel from the moving image (via pulling)

The optimized transform maps points in the physical space of the fixed image to points in the physical space of the moving image.

Image Registration

Conventions (in ITK, used in many state-of-the-art approaches)

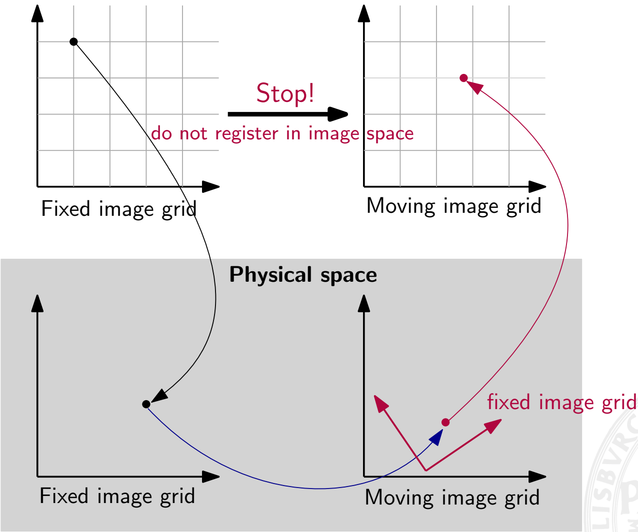


Image Registration

Resources

Here are three popular (open-source) registration packages (with good documentation and examples):

- NiftyReg
- ANTs
- 3DSlicer
- FSL Brain Extraction Tool (BET)

Brain MR data (for fun) can be downloaded from the *OASIS Brain database* here.

One of the standard brain MR atlases (i.e., an *average* brain with segmented regions), the *ICBM 152 atlas*, can be found online at here.