Computing Methods for Experimental Physics and Data Analysis

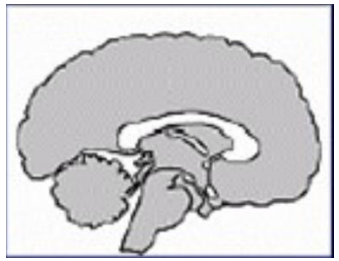
Data Analysis in Medical Physics

Lecture 4: Image transformation; image co-registration; code vectorization in MATLAB

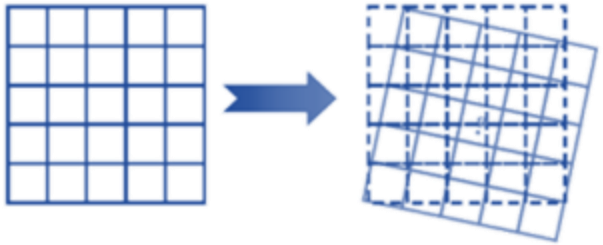
Alessandra Retico
alessandra.retico@pi.infn.it
INFN - Pisa

Image transformations

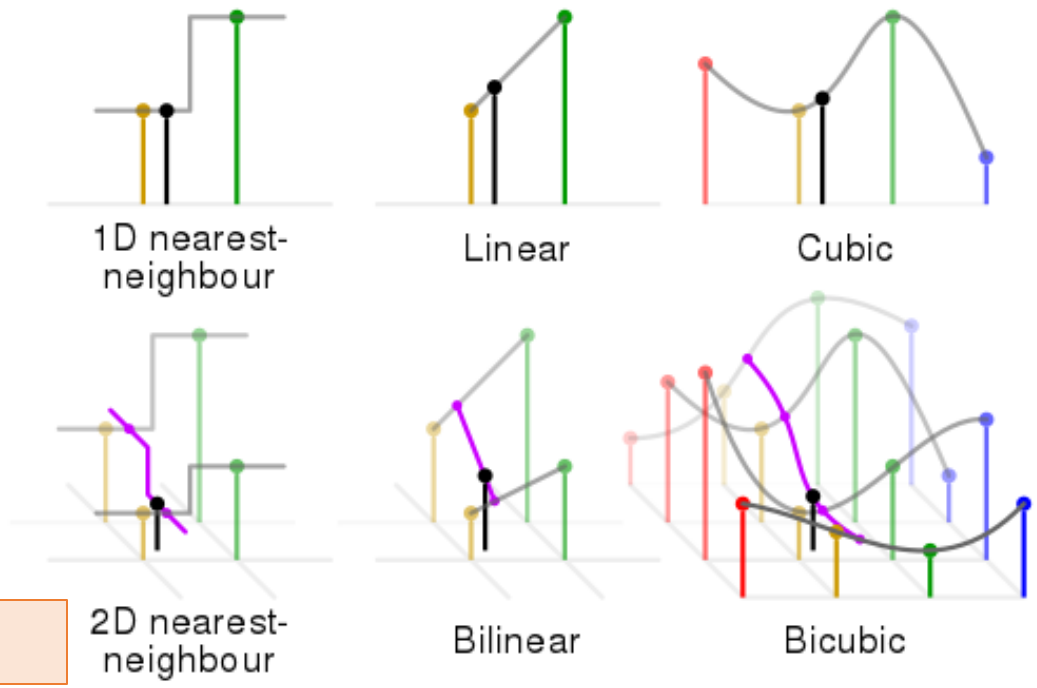
- Geometric transformations: translation, rotation, scaling, shear



Translation Rotation Scaling Shear



- Transformed images often need resampling; an interpolation method should be specified, e.g.:
  - Nearest neighbor
  - Bilinear interpolation
  - Bicubic interpolation



See demo L4_code/demo_resize_transform.mlx

Affine transformation

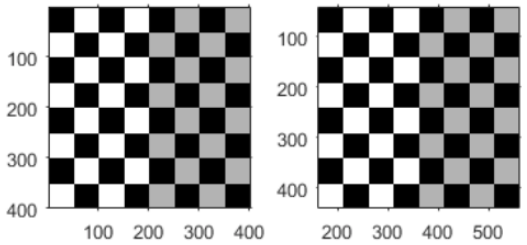
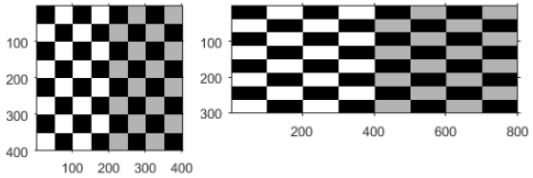
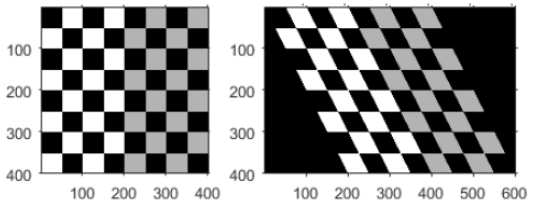
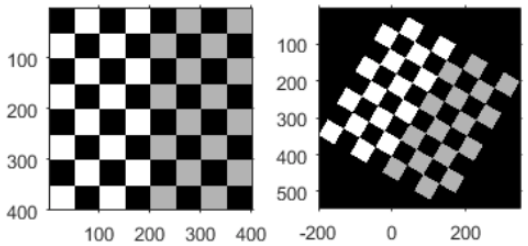
- In Euclidean geometry, an **affine transformation** is a geometric transformation that preserves lines and parallelism (but not necessarily distances and angles).
- An affine map is the composition of two functions: a linear map (multiplication by a matrix **A**) and a translation (addition of a vector ***b***).

$$\vec{y} = f(\vec{x}) = A\vec{x} + \vec{b}.$$

- Using an augmented matrix and an augmented vector, it is possible to represent both the translation and the linear map using a single matrix multiplication.

$$\begin{bmatrix} \vec{y} \\ 1 \end{bmatrix} = \left[\begin{array}{ccc|c} & A & & \vec{b} \\ 0 & \dots & 0 & 1 \end{array}\right] \begin{bmatrix} \vec{x} \\ 1 \end{bmatrix}$$

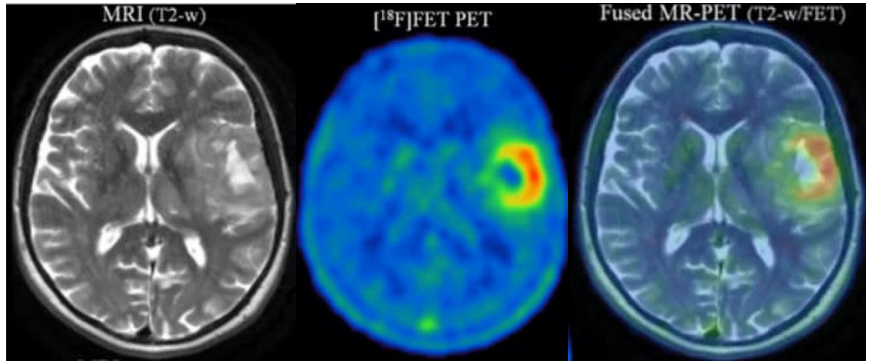
2D affine transformations

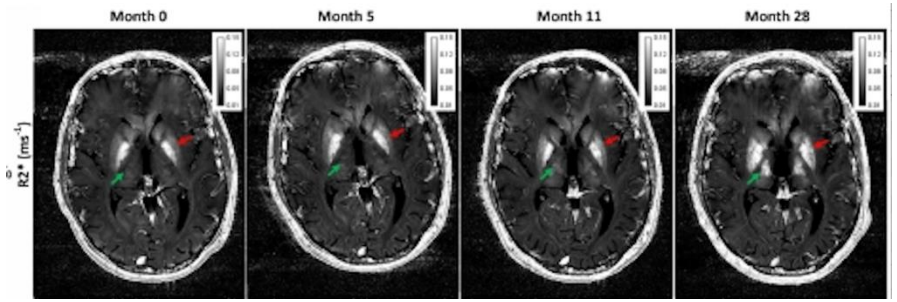
2-D Affine Transformation	Example (Original and Transformed Image)	Transformation Matrix	
Translation		$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ t_x & t_y & 1 \end{bmatrix}$	t_x specifies the displacement along the *x* axis t_y specifies the displacement along the *y* axis. For more information about pixel coordinates, see Image Coordinate Systems.
Scale		$\begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$	s_x specifies the scale factor along the *x* axis s_y specifies the scale factor along the *y* axis.
Shear		$\begin{bmatrix} 1 & sh_y & 0 \\ sh_x & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	sh_x specifies the shear factor along the *x* axis sh_y specifies the shear factor along the *y* axis.
Rotation		$\begin{bmatrix} \cos(q) & \sin(q) & 0 \\ -\sin(q) & \cos(q) & 0 \\ 0 & 0 & 1 \end{bmatrix}$	q specifies the angle of rotation about the origin.

See demo L4_code/ demo_resize_transform.mlx

Image registration

- Image registration is a process that transforms images acquired with different conditions (e.g. geometry of acquisition, resolution, modality) so that they can be put in spatial relation with a reference image
- Registration is necessary to:
  - compare joint information by different imaging modalities of the same subject (multimodal studies)
  - compare image information of the same subjects at different time points (e.g. fMRI data, longitudinal studies)
  - compare image information of different subjects (group analyses)





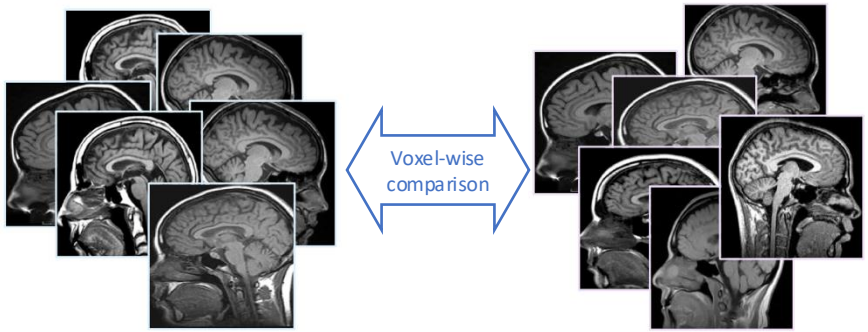
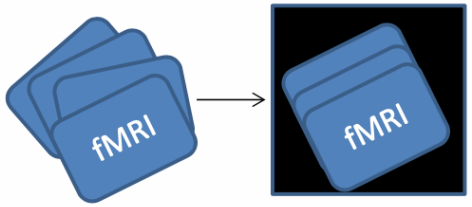
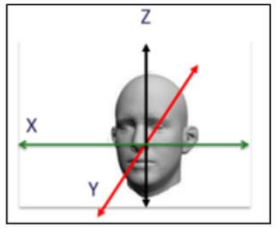
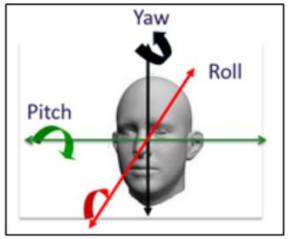


Image registration

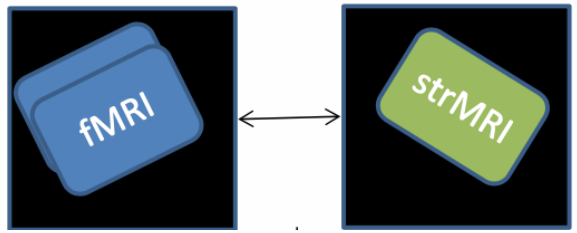
- Realignment:
  - Used in motion correction in fMRI; rigid alignment to a template
  - Rigid transformation: 6 DoF (3 DoF for translations across axes, 3 DoF for rotations) + interpolation



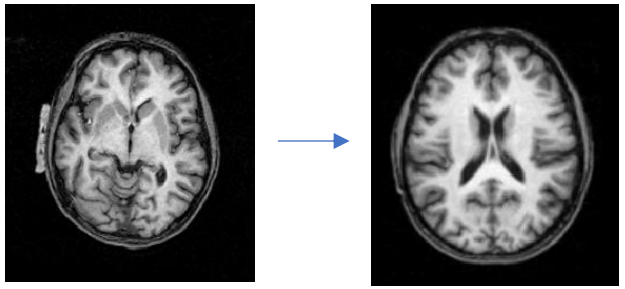




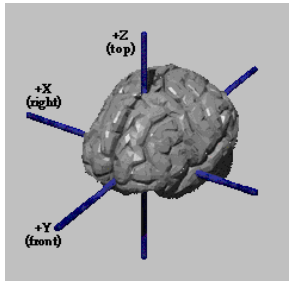
- Coregistration:
  - Used to match modalities
  - Rigid transformation (6 DoF) + interpolation



- Normalization
  - Used to match images to a template
  - Affine transformation (to start): 12 DoF (3 translations, 3 rotations, 3 scaling, 3 skewing)
  - Non-linear deformation/warping (to refine the alignment): 1000s DoF

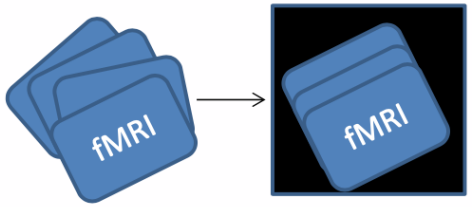


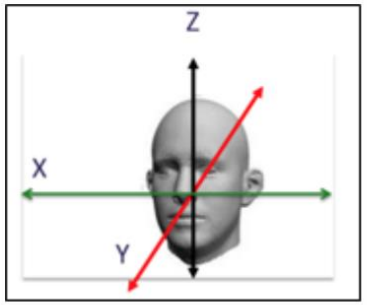
reference orientation

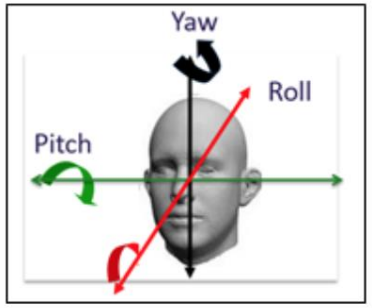


3D Rigid-body Transformations

- A 3D rigid body transform is defined by:
  - 3 translations - in X, Y & Z directions
  - 3 rotations - about X, Y & Z axes
- The order of the operations matters





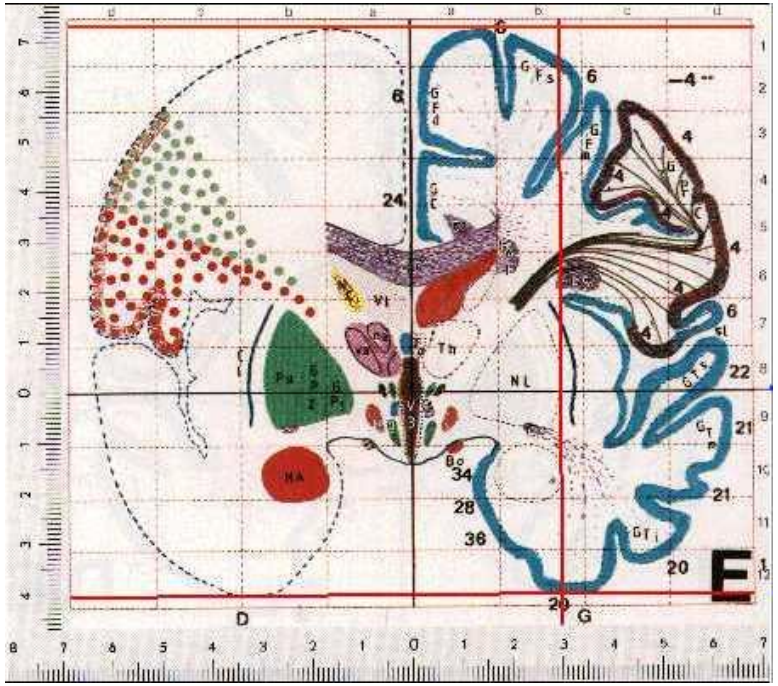


$$\begin{pmatrix} 1 & 0 & 0 & X_{trans} \\ 0 & 1 & 0 & Y_{trans} \\ 0 & 0 & 1 & Z_{trans} \\ 0 & 0 & 0 & 1 \end{pmatrix} \times \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\Phi & \sin\Phi & 0 \\ 0 & -\sin\Phi & \cos\Phi & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \times \begin{pmatrix} \cos\Theta & 0 & \sin\Theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\Theta & 0 & \cos\Theta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \times \begin{pmatrix} \cos\Omega & \sin\Omega & 0 & 0 \\ -\sin\Omega & \cos\Omega & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

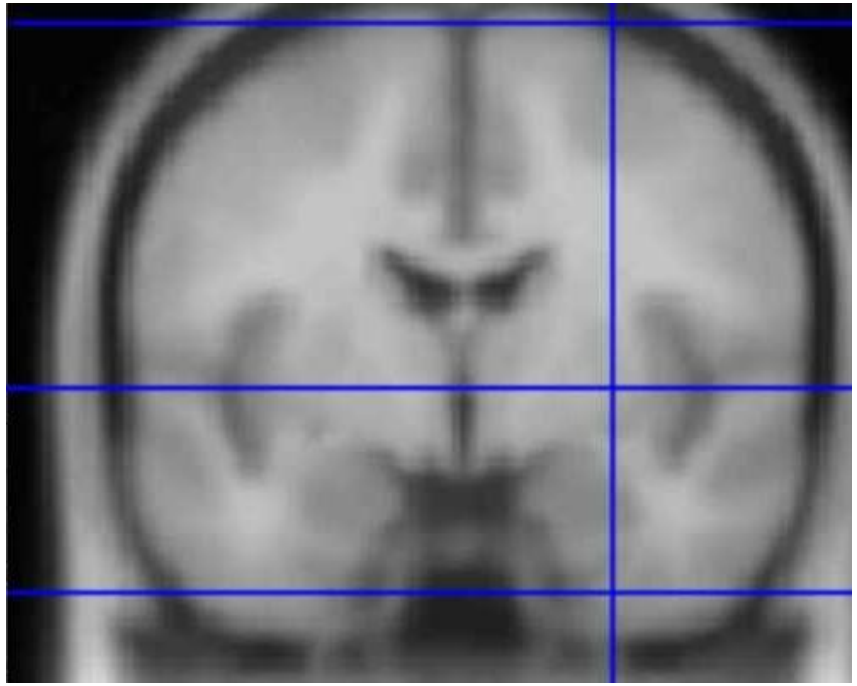
Translations | Pitch about x axis | Roll about y axis | Yaw about z axis

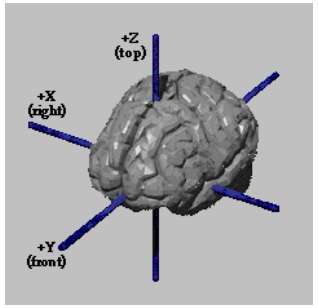
Standard reference space: Montreal Neurological Initiative (MNI)

The Talairach Atlas



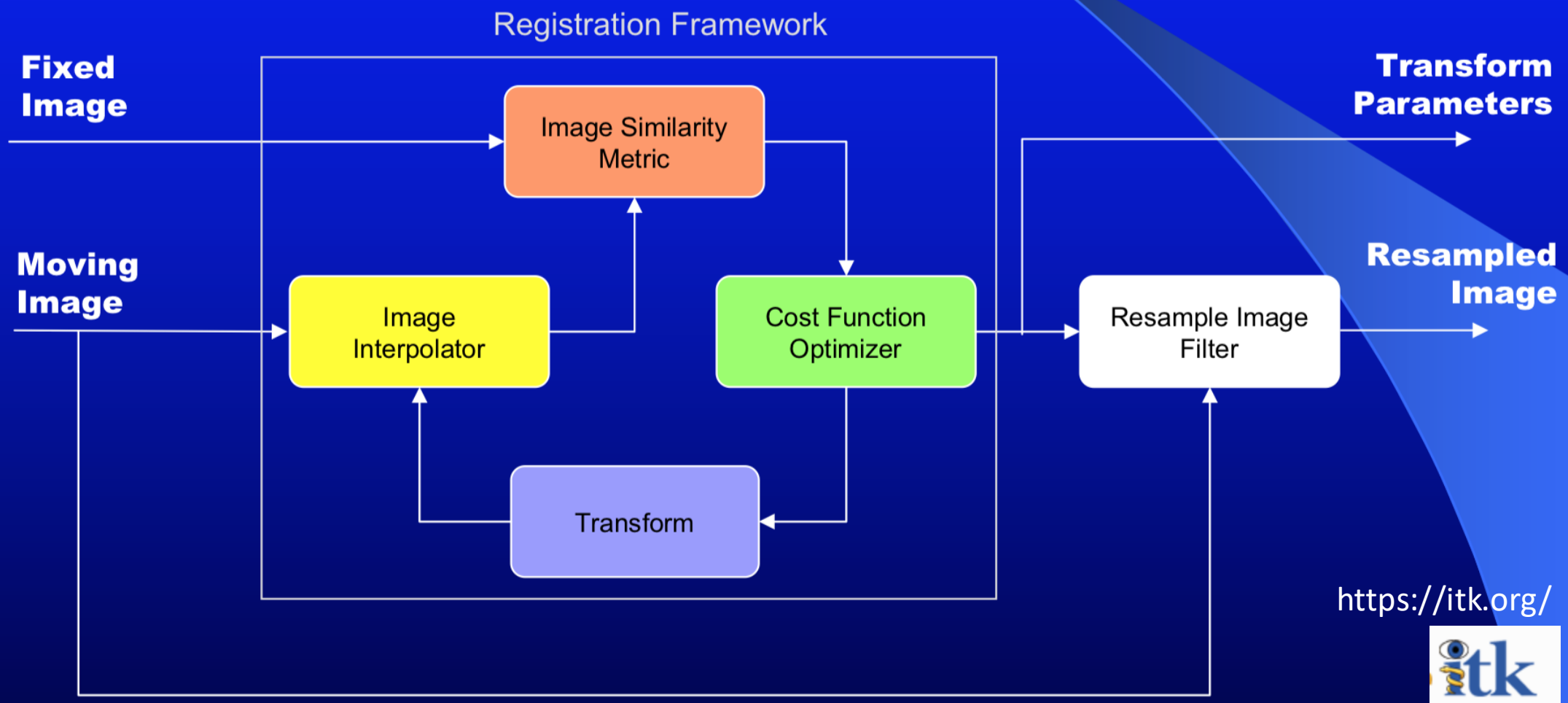
The MNI/ICBM AVG152 Template





The MNI template follows the *convention* of the Talairach Atlas (Talairach and Tournoux, 1988), but does not match the *particular brain*
http://imaging.mrc-cbu.cam.ac.uk/imaging/MniTalairach

Registration Framework Components



https://itk.org/

Image registration key elements

- To formulate an image registration problem, the following elements need to be defined:
  - the feature space – the domain containing the information for matching
    - Focus on intensity-based measures
  - the search space – the class of transformations to be used to establish the correspondence
    - E.g. rigid, affine, diffeomorphisms
  - the search strategy – the optimization process to be conducted to assess the correspondence
  - the similarity metric: the figure of merit chosen in the optimization process

- Then, the registration process consists in the following two operations:
  - Optimise the parameters that describe a spatial transformation between the source (moving) and reference (fixed) images
  - Transforming and resampling the moving images according to the determined transformation parameters

Voxel-to-world Transforms

- Affine transform associated with each image
  - Maps from voxels (x_v=1..n_x, y_v=1..n_y, z_v=1..n_z) to some world coordinate system, (x_w= x_1..x_n *mm*, y_w= y_1..y_n *mm*, z_w= z_1..z_n *mm*) e.g.,:
    - Scanner coordinates
    - Talairach and Tournoux (T&T) or Montreal Neurological Initiative (MNI) coordinates, i.e. spatially normalised images

$$\Rightarrow \quad \mathbf{x}_w^A = M_A \mathbf{x}_v \qquad \mathbf{x}_w^B = M_B \mathbf{x}_v$$

- Registering image B to image A means identifying the transformation matrix T that aligns the moving image B to the fixed image A.
- This operation will update B's voxel-to-world mapping
  - The voxel coordinate in the moving image B space is transformed via M_B to world space, aligned using **T**, and mapped back to the voxel space of the fixed image using M_A^{-1}
  - Mapping from voxels in B to voxels in A is by
    - B-to-world using M_B, then world-to-A using M_A^{-1}, i.e. $M_A^{-1}\, T\, M_B$

$$\Rightarrow \quad M_B \rightarrow M_A^{-1}\, T\, M_B$$

Optimization protocols and cost functions

Image registration is done by optimisation.

Optimisation involves finding some best parameters according to a cost function, which is either minimised or maximised

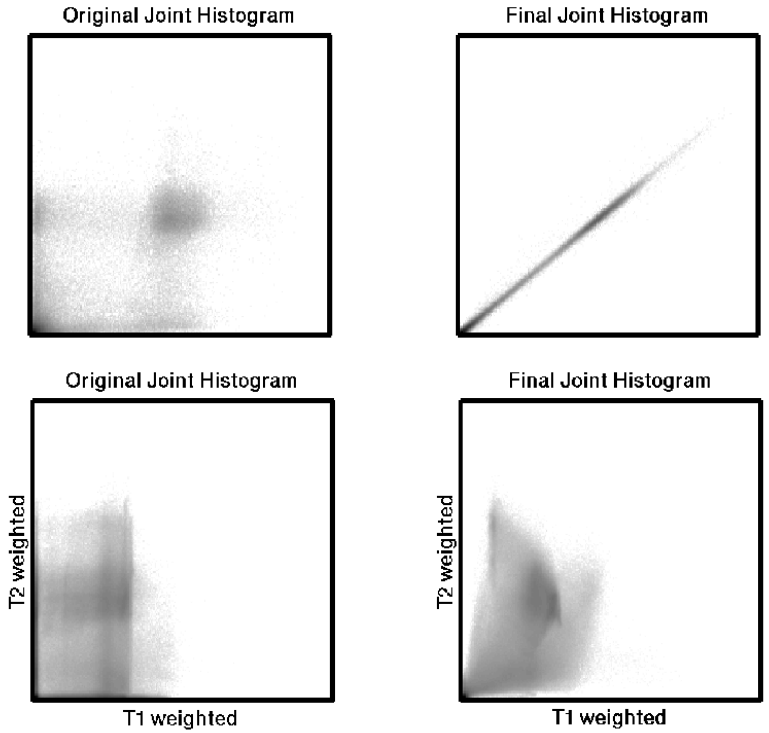
Optimization methods

- Gradient descent
- Genetic algorithms

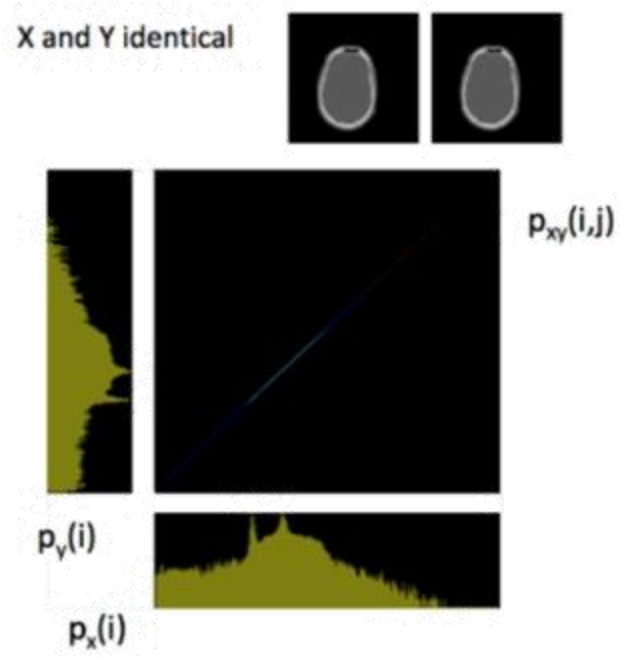
Cost functions

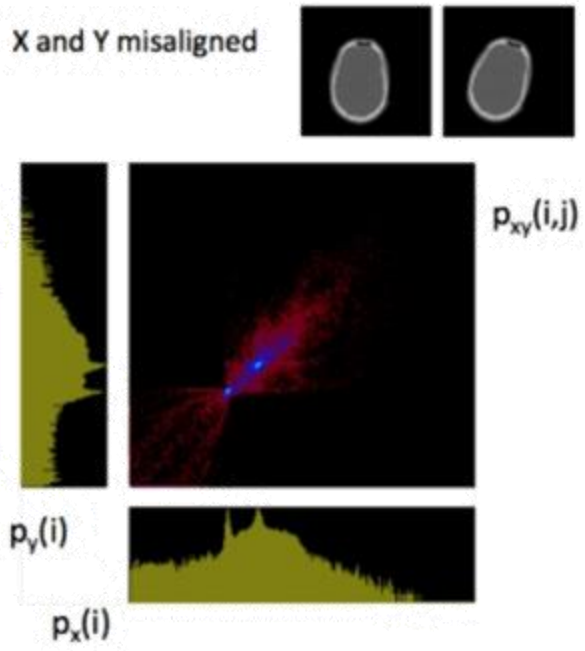
- Intra-modal (e.g. realignment in fMRI)
  - Mean squared difference (minimise)
  - Normalised cross correlation (maximise)
- Inter-modal (or intra-modal)
  - Mutual information (maximise)
  - Normalised mutual information (maximise)
  - Entropy correlation coefficient (maximise)

The joint histogram is a useful tool for visualizing the relationship between the intensities of corresponding voxels in two or more images.

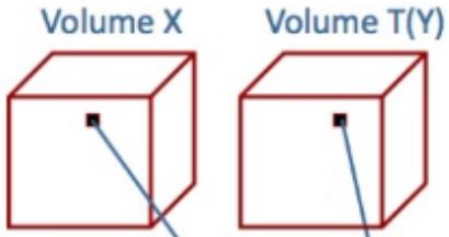


Intra-modal registration





Normalized Cross Correlation (NCC)



$\bar{x}$: Mean

σ_x : Standard deviation

N : Number of pixels

$$NCC = \frac{1}{N}\sum_i \frac{(x_i - \bar{x})(y_i - \bar{y})}{\sigma_x \sigma_y}$$

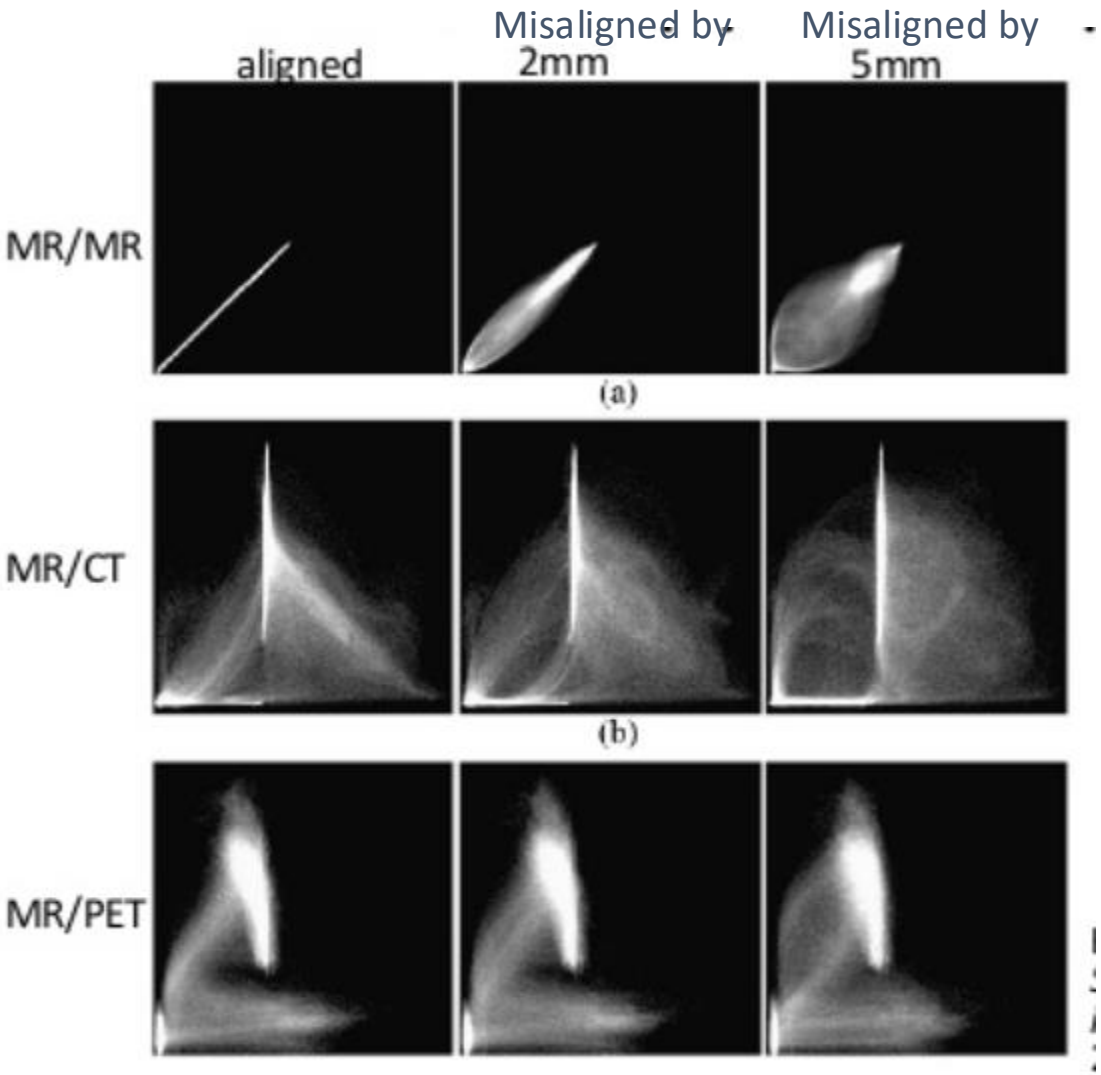
Normalized Cross Correlation:

Expresses the linear relationship between voxel intensities in the two volumes

Joint histograms

Intra-modal

Inter-modal



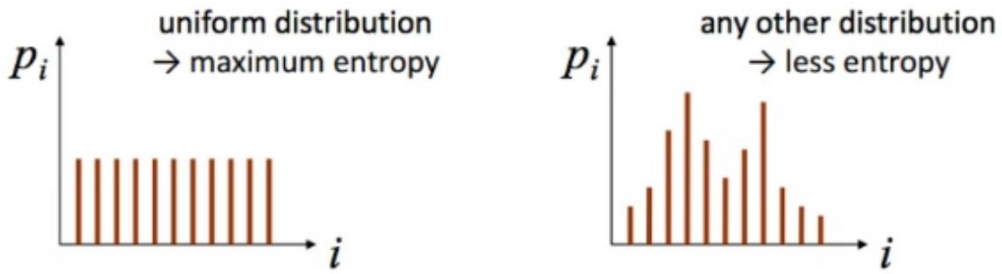
Inter-modal registration

- Image registration is considered as to maximize the amount of shared information in two images
  - reducing the amount of information in the combined image
- Algorithms used
  - Joint entropy
    - Joint entropy measures the amount of information in the two images combined
  - **Mutual information**
    - **A measure of how well one image explains the other, and is maximized at the optimal alignment**
  - Normalized Mutual Information

Entropy for image registration

Shannon Entropy, developed in the 1940s
(communication theory)

$$H = -\sum_i p_i \log p_i$$



- Shannon's **Entropy**

$$H = \sum_i p_i \cdot \log \frac{1}{p_i}$$

  - weighs the information based on the probability that an outcome will occur
  - second term shows the amount of information an event provides is inversely proportional to its probability of occurring

Entropy for Image Registration

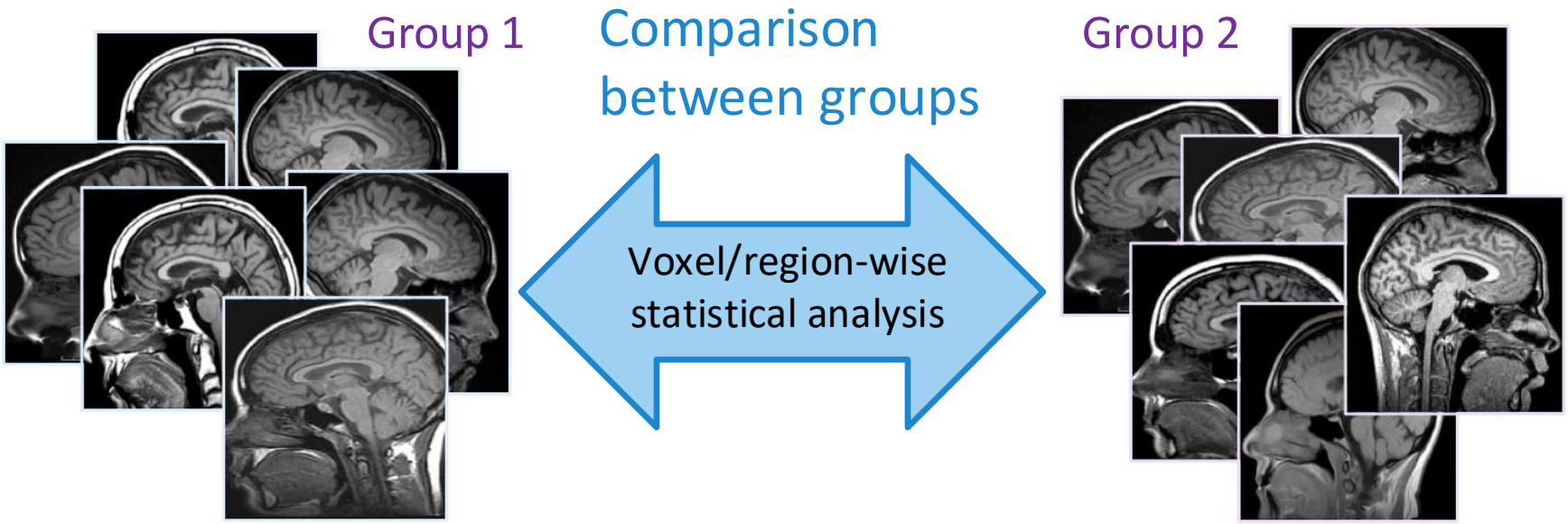
- Using joint entropy for registration
  - Define joint entropy to be:

$$H(A,B) = -\sum_{i,j} p(i,j) \cdot \log[p(i,j)]$$

  - <u>Images are registered when one is transformed relative to the other to minimize the joint entropy</u>
  - The dispersion in the joint histogram is thus minimized

A typical task in neuroimaging studies: voxel-wise comparison between groups of subjects

The inter-subject coregistration is necessary if we want to compare local brain characteristics of different subjects



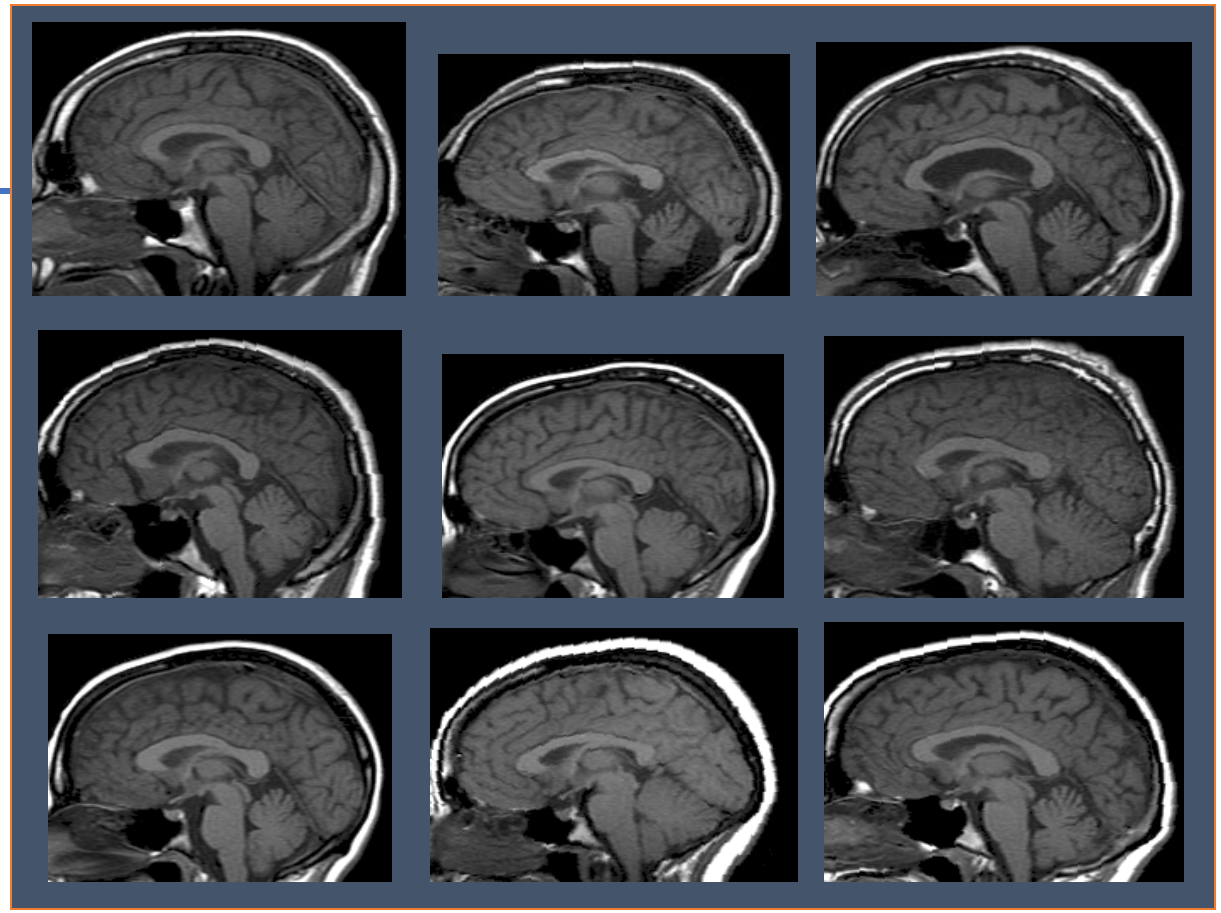
- Comparison between Patients (group 1) and Controls (group 2):
  - to detect pathology specific brain alterations

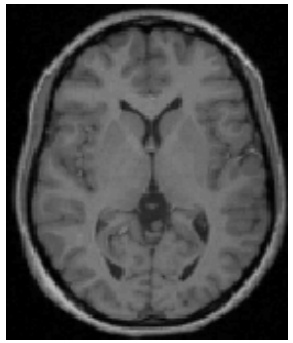
- Need to bring them all into a common anatomical space.
  - Examine homologous regions across subjects
  - Report findings in a common anatomical space (e.g. MNI space)

Coregister to a template

- How can we carry out a voxel-wise comparison of brains of different subjects?

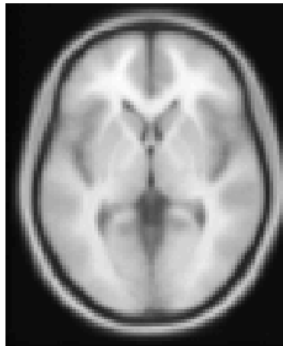
- Images need to be **spatially aligned**





Each image is registered to a T_1 template

12 DoF affine transformation, to match the size and position, possibly + non-linear warping

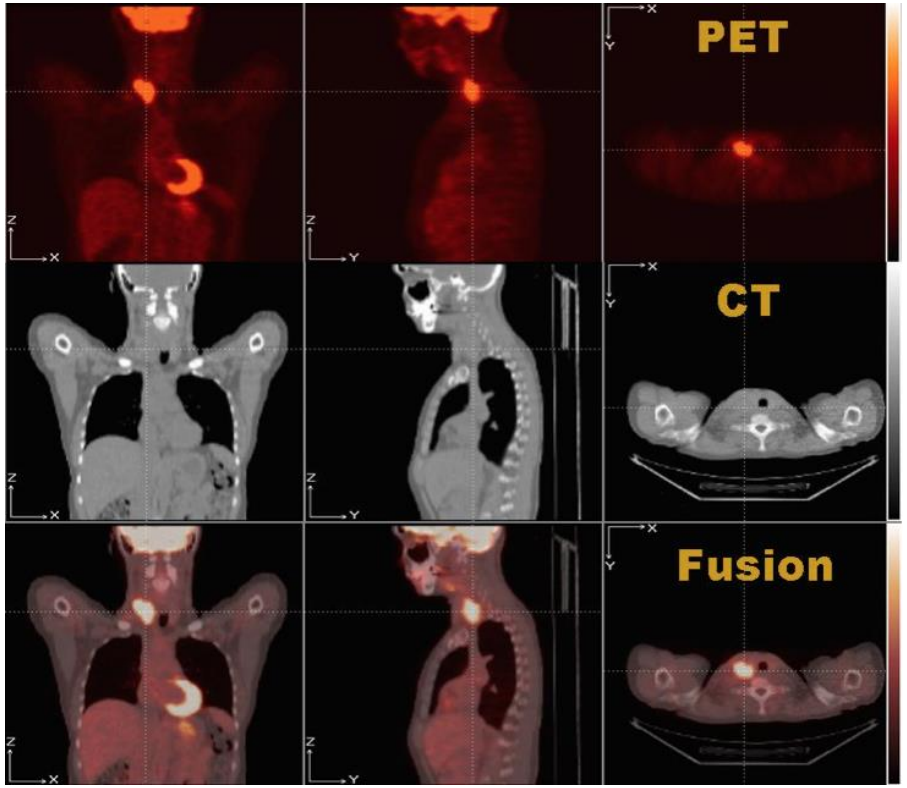


This is a necessary preliminary step in all template-based segmentation software

Coregister different images of the same subject

The intra-subject *coregistration* is useful:

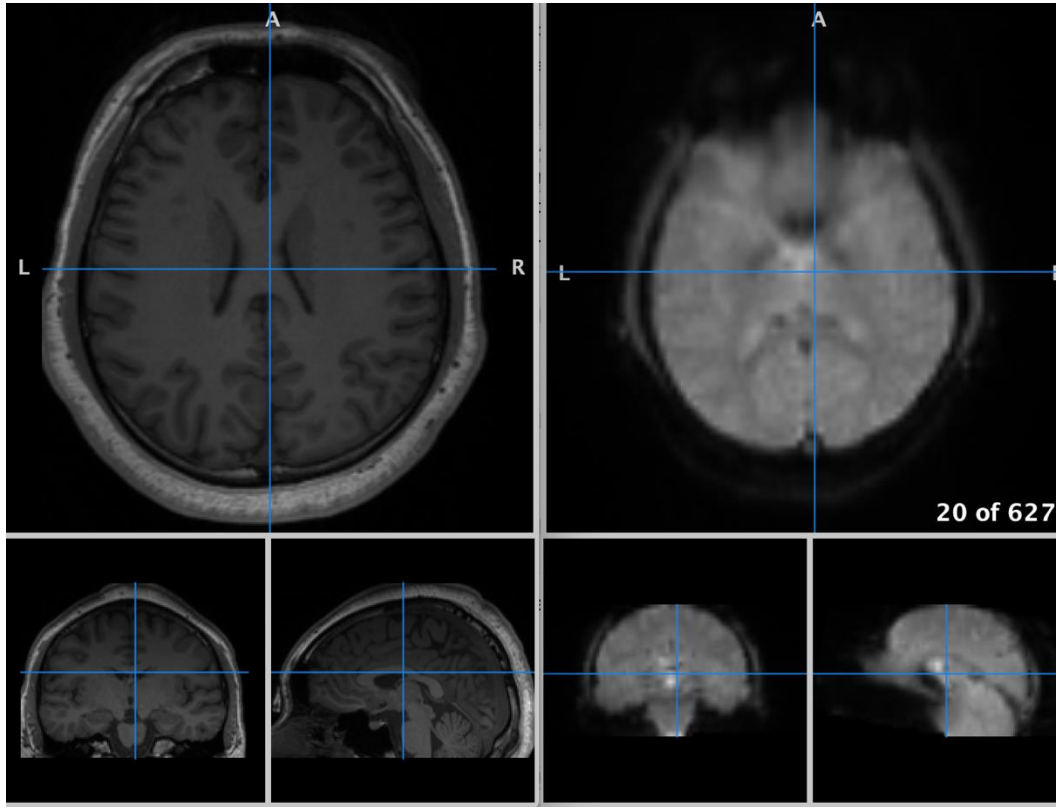
- To reduce motion artifact during sequential acquisitions (e.g. fMRI)
- To detect changes in longitudinal studies (e.g. to study the effect of age or of a disease)
- To *overlay* information acquired with different image modalities (image fusion)

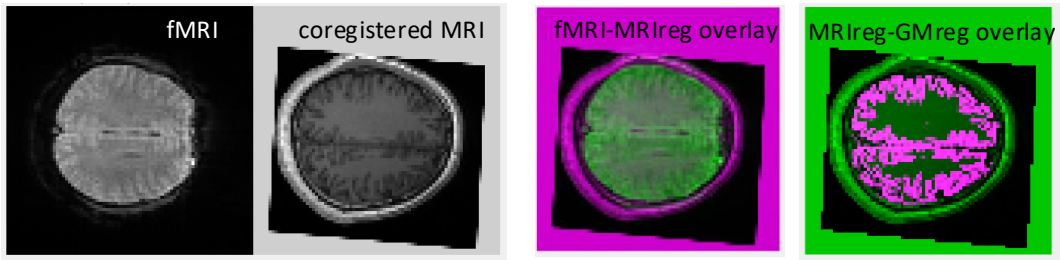


See demo L4_code/demo_coregistration.mlx

Demo on image coregistration

- We will coregister the structural MRI and fMRI data of one subjects
- The volumes of the two different modalities are very roughly similarly oriented
- The voxel size are different
- We move the higher resolution image to match the lower resolution one to avoid a strong resampling of the functional signal
- Once the two modalities are aligned, the same transformation can be applied for example to GM segmented volume, which is originally aligned with the structural MRI volume





The imregdemons function is based on

Medical Image Analysis (1998) volume 2, number 3, pp 243–260

Image matching as a diffusion process: an analogy with Maxwell's demons

J.-P. Thirion*

INRIA, Equipe Epidaure, 2004 Route des Lucioles BP93, 06902 Sophia-Antipolis, France

Abstract
In this paper, we present the concept of diffusing models to perform image-to-image matching. Having two images to match, the main idea is to consider the objects boundaries in one image as semi-permeable membranes and to let the other image, considered as a deformable grid model, diffuse through these interfaces, by the action of effectors situated within the membranes. We illustrate this concept by an analogy with Maxwell's demons. We show that this concept relates to more traditional ones, based on attraction, with an intermediate step being optical flow techniques. We use the concept of diffusing models to derive three different non-rigid matching algorithms, one using all the intensity levels in the static image, one using only contour

NeuroImage 45 (2009) S61–S72

Contents lists available at ScienceDirect

NeuroImage

journal homepage: www.elsevier.com/locate/ynimg

Diffeomorphic demons: Efficient non-parametric image registration

Tom Vercauteren [a,*], Xavier Pennec [b], Aymeric Perchant [a], Nicholas Ayache [b]

[a] Mauna Kea Technologies, 9 rue d'Enghien, 75010 Paris, France
[b] INRIA Sophia Antipolis, Asclepios Research Project, 2004 route des Lucioles - BP 93, 06902 Sophia Antipolis Cedex, France

A B S T R A C T

We propose an efficient non-parametric diffeomorphic image registration algorithm based on Thirion's demons algorithm. In the first part of this paper, we show that Thirion's demons algorithm can be seen as an optimization procedure on the entire space of displacement fields. We provide strong theoretical roots to the different variants of Thirion's demons algorithm. This analysis predicts a theoretical advantage for the symmetric forces variant of the demons algorithm. We show on controlled experiments that this advantage is confirmed in practice and yields a faster convergence. In the second part of this paper, we adapt the optimization procedure underlying the demons algorithm to a space of diffeomorphic transformations. In contrast to many diffeomorphic registration algorithms, our solution is computationally efficient since in practice it only replaces an addition of displacement fields by a few compositions. Our experiments show that in addition to being diffeomorphic, our algorithm provides results that are similar to the ones from the demons algorithm but with transformations that are much smoother and closer to the gold standard, available in controlled experiments, in terms of Jacobians.

NeuroImage

www.elsevier.com/locate/ynimg
NeuroImage 38 (2007) 95–113

A fast diffeomorphic image registration algorithm

John Ashburner

Wellcome Trust Centre for Neuroimaging, 12 Queen Square, London, WC1N 3BG, UK

Received 26 October 2006; revised 14 May 2007; accepted 3 July 2007
Available online 18 July 2007

This paper describes DARTEL, which is an algorithm for diffeo-

Many registration approaches still use a small deformation

See demo L4_code/demo_nonRigidTransform.m

Image processing and analysis resources

https://itk.org/itkindex.html

Kitware

Search

PROJECT RESOURCES HELP OPEN SOURCE

Welcome to the **Insight Toolkit (ITK)**. ITK is an open-source, cross-platform system that provides developers with an extensive suite of software tools for image analysis. Developed through extreme programming methodologies, ITK employs leading-edge algorithms for registering and segmenting multidimensional data. The goals for ITK include:

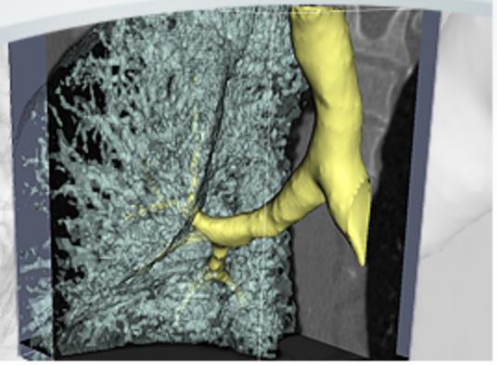
- Supporting the Visible Human Project.
- Establishing a foundation for future research.
- Creating a repository of fundamental algorithms.
- Developing a platform for advanced product development.
- Support commercial application of the technology.
- Create conventions for future work.
- Grow a self-sustaining community of software users and developers.

Start Using ITK ▸

NSF NLM NATIONAL CANCER INSTITUTE NIMH NIDCD NIDCR

ITK

ITK provides leading-edge segmentation and registration algorithms in two, three, and more dimensions; it is distributed as an open-source software package.



ITK: Insight Segmentation and Registration Toolkit, a C++ open source image analysis toolkit

SimpleITK: simplified, open source, interface to ITK

Search docs

TABLE OF CONTENTS

DemonsRegistration1

Overview

This example illustrates how to use the classic Demons registration algorithm. The user supplied parameters for the algorithm are the number of iterations and the standard deviations for the Gaussian smoothing of the total displacement field. Additional methods which control regularization, total field smoothing for elastic model or update field smoothing for viscous model are available.

The underlying assumption of the demons framework is that the intensities of homologous points are equal. The example uses histogram matching to make the two images similar prior to registration. This is relevant for registration of MR images where the assumption is not valid. For other imaging modalities where the assumption is valid, such as CT, this step is not necessary. Additionally, the command design pattern is used to monitor registration progress. The resulting deformation field is written to file.

See also: DemonsRegistration2.

Code

C++ | **Python** | R

DemonsRegistration1

Read the Docs v: master

C++ | Python | R

```python
#!/usr/bin/env python

from __future__ import print_function

import SimpleITK as sitk
import sys
import os


def command_iteration(filter) :
    print("{0:3} = {1:10.5f}".format(filter.GetElapsedIterations(),
                                     filter.GetMetric()))

if len ( sys.argv ) < 4:
    print( "Usage: {0} <fixedImageFilter> <movingImageFile> <outputTransformFile>".format(sys.argv[0]))
    sys.exit ( 1 )


fixed = sitk.ReadImage(sys.argv[1], sitk.sitkFloat32)

moving = sitk.ReadImage(sys.argv[2], sitk.sitkFloat32)


matcher = sitk.HistogramMatchingImageFilter()
matcher.SetNumberOfHistogramLevels(1024)
matcher.SetNumberOfMatchPoints(7)
matcher.ThresholdAtMeanIntensityOn()
moving = matcher.Execute(moving,fixed)

# The basic Demons Registration Filter
# Note there is a whole family of Demons Registration algorithms included in SimpleITK
demons = sitk.DemonsRegistrationFilter()
demons.SetNumberOfIterations( 50 )
# Standard deviation for Gaussian smoothing of displacement field
demons.SetStandardDeviations( 1.0 )
```

Widely used software suites for coregistration in neuroimaging

Help Login

FSL

https://fsl.fmrib.ox.ac.uk/fsl/

FSL is a comprehensive library of analysis tools for FMRI, MRI and DTI brain imaging data.

FMRIB Software Library v6.0

Created by the Analysis Group, FMRIB, Oxford, UK.

ANTs

Advanced Normalization Tools

University of Pennsylvania
Richards Medical Research Laboratories

http://stnava.github.io/ANTs/

Advanced Normalization Tools (ANTs) extracts information from complex datasets that include imaging. ANTs development is led by Brian Avants and supported by other researchers and developers at PICSL and other institutions.

https://www.fil.ion.ucl.ac.uk/spm/software/

UCL University College London SPM

SPM is made freely available to the [neuro]imaging community, to promote collaboration and a common analysis scheme across laboratories. The software represents the implementation of the theoretical concepts of Statistical Parametric Mapping in a complete analysis package.

The SPM software is a suite of MATLAB (The MathWorks, Inc) functions and subroutines with some externally compiled C routines. SPM was written to organise and interpret our functional neuroimaging data. The distributed version is the same as that we use ourselves.

https://afni.nimh.nih.gov

AFNI (Analysis of Functional NeuroImages) is a leading software suite of C, Python, R programs and shell scripts primarily developed for the analysis and display of anatomical and functional MRI (FMRI) data. It is freely available (both in source code and in precompiled binaries) for research purposes. The software is made to run on virtually an Unix system with X11 and Motif displays. Binary Packages are provided for MacOS and Linux systems including Fedora, Ubuntu (including Ubuntu under the Windows Subsytem for Linux)

References and sources

- Image processing and analysis resources
  - ITK - https://itk.org/

- Neuroimaging coregistration software
  - FSL - https://fsl.fmrib.ox.ac.uk/fsl/fslwiki
  - AFNI - https://afni.nimh.nih.gov
  - SPM - https://www.fil.ion.ucl.ac.uk
  - ANTs - http://picsl.upenn.edu/software/ants/

- Sources
  - https://simpleitk.readthedocs.io/en/master/index.html
  - https://simpleitk.readthedocs.io/en/master/tutorialsAndCourses.html
  - https://simpleitk.readthedocs.io/en/master/registrationOverview.html

Techniques to improve performances in MATLAB

Measure the performance of your code using tic toc, timeit or profile

Programming practices to improve performance:

- Preallocate — Instead of continuously resizing arrays, consider preallocating the maximum amount of space required for an array. For more information, see Preallocation.
- Vectorize — Instead of writing loop-based code, consider using MATLAB matrix and vector operations. For more information, see Vectorization.
- Place independent operations outside loops — If code does not evaluate differently with each for or while loop iteration, move it outside of the loop to avoid redundant computations.
- …
- Run code on a GPU or in parallel — If you have a Parallel Computing Toolbox™ license, run code on a GPU by passing gpuArray data to a supported function or run code in parallel using, for example, a parfor-loop. For more information, see Choose a Parallel Computing Solution (Parallel Computing Toolbox).

Performance issues in Matlab

- MATLAB is:
  - very fast on vector and matrix operations
  - correspondingly slow with loops
- MATLAB is a matrix-based language. Avoiding for loops, and using matrices is useful:
  - sometimes for speed
  - sometimes to improve code readability and easy maintenance
- Thus:
  - Try to avoid loops
  - Try to vectorize your code

See demo code (L4_code/Hands-on/):
- show_diamond.m
  - diamond.m
  - (diamond_bad.m)

Code vectorization in MATLAB

- Vectorization is one of the core concepts of MATLAB.
- With one command **it lets you process all elements of an array, avoiding loops and making your code more readable and efficient**.
- The process of revising loop-based, scalar-oriented code to use MATLAB matrix and **vector operations** is called vectorization.
- Vectorizing your code is worthwhile for several reasons:
  - *Appearance*: Vectorized mathematical code appears more like the mathematical expressions found in textbooks, making the code easier to understand.
  - *Less Error Prone*: Without loops, vectorized code is often shorter. Fewer lines of code mean fewer opportunities to introduce programming errors.
  - *Performance*: Vectorized code often runs much faster than the corresponding code containing loops.

a for loop to compute the values of a function at several points

```
i = 0;
for t = 0:.01:10
    i = i + 1;
    y(i) = sin(t);
end
```

vectorized version of the same code

```
t = 0:.01:10;
y = sin(t);
```

- For data stored in numerical arrays, most MATLAB functions are inherently vectorized.

References and sources

- Code profiling
  - https://it.mathworks.com/help/matlab/matlab_prog/profiling-for-improving-performance.html

- Techniques for improving performances in MATLAB
  - https://it.mathworks.com/help/matlab/matlab_prog/techniques-for-improving-performance.html
  - https://it.mathworks.com/help/matlab/matlab_prog/vectorization.html
  - https://it.mathworks.com/help/parallel-computing/parallel-for-loops-parfor.html
  - https://it.mathworks.com/help/parallel-computing/choosing-a-parallel-computing-solution.html
  - https://it.mathworks.com/help/parallel-computing/run-matlab-functions-on-a-gpu.html