Image Registration with pFIRE

CompBioMed Winter School 2019

Phil Tooley
University of Sheffield

14 February 2019

Session Layout

Introductory Talk 09:00 – 09:15

- What is image registration?
- Applications for elastic registration
- How pFIRE works

Hands on tutorial: 09:15 – 10:45

- Access pFIRE on MareNostrum
- How to run pFIRE
- Visualising the output

Coffee Break: 10:45 – 11:00

What is image registration?

Image Registration

Image registration is the process by which one image is transformed by displacing pixels to match another image as closely as possible

The image which is transformed is known as the *moved* image, and the target image to which it is matched is known as the *fixed image*

What is image registration?

Types of registration

Rigid Registration

- Global transformations which affect the whole image in the same way
- E.g translation, scaling, linear shearing and rotation
- Cannot describe non-linear or local deformations
- Relatively simple to compute
- Commonly used for alignment

Types of registration

Elastic registration

- Displacement field which describes how individual image pixels are moved
- Map can be arbitrarily high resolution
- In principle, describe any transformation
- Computationally intensive
- Useful in a wide range of applications

pFIRE performs **elastic** registration

Applications for elastic registration

- Segmentation by registration
  - Manually segment first image
  - Register other images to first image
  - Use results to map segmentation to other images
- Patient specific modelling (PSM)
  - Create a generic model of structure
  - Register patient image to generic model
  - Use result to transform generic $\rightarrow$ patient specific model
  - Apply analysis to new PSM
- Digital Volume Correlation (Stress/strain analysis)
  - Image unstressed sample (μ-CT)
  - Apply stress and image again
  - Register to produce displacement field
  - Differentiate displacement to get strain field in sample

How pFIRE works (briefly!)

- Represent fixed and moved images with functions $f(\mathbf{x})$, $m(\mathbf{x})$
- These give the value of the pixel at the point $\mathbf{x}$
- A perfect mapping is a vector field $\mathbf{R}(\mathbf{x})$, such that

$$m(\mathbf{x} + \mathbf{R}(\mathbf{x})) = f(\mathbf{x}) \tag{1}$$

- To register the images, find a function $\mathbf{R}(\mathbf{x})$ that satisfies this equation

Finding the displacement field

- Taylor expand to get an expression that uses $f(\mathbf{x})$ and $m(\mathbf{x})$

$$f(\mathbf{x}) - m(\mathbf{x}) \simeq \frac{1}{2}\mathbf{R}(\mathbf{x}) \cdot (\nabla \cdot f(\mathbf{x}) + m(\mathbf{x})) \qquad (2)$$

- This now uses the two input images and the gradient of the images

- Problem: equation is underdetermined!

Finding the displacement field

Finding the displacement field

- Need to add additional constraints (Tikhonov Regularisation)
- Constrain allows us to choose solution $\rightarrow$ want smoothest solution
- Displacement is smooth if second derivative is zero everywhere
- Laplacian operator:

$$\nabla^2 \mathbf{R}(\mathbf{x}) = 0 \tag{3}$$

Finding the displacement field

- Problem is actually discrete – have pixels in image and nodes in displacement field
- Typically $N_{pixels} \gg N_{nodes}$, problem has N_{nodes} DOF
- Each map node uses information from many pixels
- Represent as a matrix equation

$$\begin{bmatrix} \mathbf{f} - \mathbf{m} \\ 0 \end{bmatrix} = \begin{bmatrix} \mathbf{T} \\ \mathbf{L} \end{bmatrix} \cdot \mathbf{R} \tag{4}$$

- **T** is a non-square matrix mapping pixel information to map node terms
- **L** is the Laplacian matrix

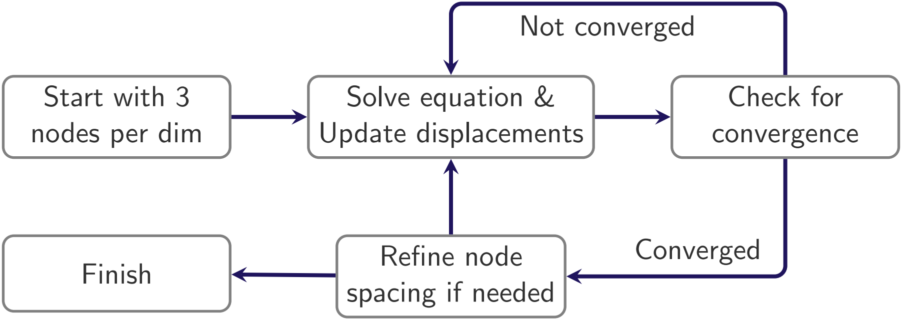
The pFIRE Algorithm

- pFIRE solves the matrix equation:

$$\begin{bmatrix} \mathbf{T}^t & \mathbf{L}^t \end{bmatrix} \cdot \begin{bmatrix} \mathbf{f} - \mathbf{m} \\ 0 \end{bmatrix} = \begin{bmatrix} \mathbf{T}^t\mathbf{T} + \mathbf{L}^t\mathbf{L} \end{bmatrix} \cdot \mathbf{R} \tag{5}$$

- Iterative solution: solve the matrix equation many times
- Start with coarse node spacing
- Each iteration converges towards the solution
- Step converged when displacement field stops changing
- Increasingly refine node spacing when previous step converges

The pFIRE Algorithm



Detailed algorithm description in Barber and Hose (2005) or Barber et al. (2007)

Hands-on tutorial session

- pFIRE Tutorial: https://insigneo.github.io/pFIRE/docs.html

- MareNostrum readme: /gpfs/projects/nct00/nct00016/README
- Follow the instructions to set up pFIRE

- MareNostrum files: /gpfs/projects/nct00/nct00016/tutorial_files
- Copy this to your own account

- Please complete the application survey