An introduction to ITK version 4 registration

Anonymous

July 4, 2011

Abstract

The ITKv4 registration framework is a unified system for performing multi-threaded affine and deformable image registration. The revised framework supports composite transformations, unbiased registration, the simultaneous use of multiple similarity metrics, multi-channel/tensor image registration and geometrically correct transformation of covariant vectors and tensors via the composite transform framework and transformations based on finite element models. ITKv4 also contains new metrics that can be used for registering point sets, curves and surfaces as well as a set of efficiently implemented neighborhood correlation metrics. Despite these significant additions, the user interface to the framework is, at the basic level, unchanged from prior versions of ITK. Furthermore, we provide new optimization strategies that simplify the user experience by reducing the number of parameters that need to be set by the user.

1 Introduction

What is the current state of image registration? How is it applied?

What role has ITK filled in the registration world? What papers use ITK as a standard for comparison? What other software builds on ITK registration?

How does the v4 registration framework build on the past? What does it contribute that's new?

What is it? Why were these the tools we chose to contribute?

What isn't it? Why did we not focus, for instance, on optimal speed implementations?

2 Nomenclature

We will use the nomenclature below to designate an image registration algorithm pictorially. This nomenclature is intended to be a descriptive, but also technically correct, system for visually representing algorithms and applications of registration. Ideally, any standard algorithm can be written in the nomenclature below.

A position: $\mathbf{x} \in \Omega$ where Ω is the domain.

An image: $I\colon \Omega^d \rightarrow \mathbb{R}^n$ where n is the number of components per pixel and d is dimensionality.

Domain map: $\phi\colon \Omega \rightarrow \Omega$ where $\rightarrow$ may be replaced with any mapping symbol below.

Affine mapping: $\rightarrow$

Deformation field: $\rightsquigarrow$ deformation field mapping J to I.

Diffeomorphic mapping: $\leftrightsquigarrow$

Composite mapping: $\phi = \phi_1(\phi_2(\mathbf{x}))$ is defined by $\leftrightsquigarrow\rightarrow$ where ϕ_2 is of type $\leftrightsquigarrow$ which precedes the application of $\rightarrow$.

Not invertible: $\nleftrightarrow$ indicates a mapping that is not guaranteed invertible.

A standard Demons registration application that maps a labeling from one image, I, into another, J, would then be written:

$$J \rightsquigarrow\rightarrow I.$$

The notation means that the algorithm first computes an affine mapping from I to J and then computes a deformation from $I(\rightarrow)$ to J. Note, also, that the tail of the mapping indicates the transform's domain. The head of the arrow indicates its range. This is an important distinction for deformable maps.

3 Overview of the unified framework

3.1 Simplified registration example

3.2 Transform changes

3.2.1 Deformation field transforms

3.2.2 Composite transforms

Composite transform I/O

3.3 Optimizer changes

3.3.1 Composite transform optimization

3.4 Metric changes

The model used for multi-threading.

3.4.1 Correlation metrics

3.4.2 Pointset metrics

I/O and data representation

3.4.3 Revised mutual information metric

3.4.4 Multivariate metric

3.4.5 Multi-channel metric

4 ITKv4 registration by example

4.1 Composite registration

4.2 Demons registration

4.3 BSpline registration

4.4 Multi-channel registration

4.5 Tensor registration

4.6 Diffeomorphic registration

4.7 Diffeomorphic registration with two metrics

4.8 GPU Demons

5 ITKv4 registration benchmarks

Comparison with ANTs and Flirt.
Speed comparison for multi-threaded implementation.
etc.

6 Discussion and future work