Voxel Value Ratio Variance Image Registration Metric for the Insight Toolkit

Geoffrey J Topping – University of British Columbia

Abstract

This document describes a cost function, or metric, for use in image registration using the Insight Toolkit. The RatioVarianceImageToImageMetric class implements a metric which is calculated from the variance of the ratios of voxel values of the moving and fixed images.

Rationale

A registration cost function based on the variance of image values was proposed by Woods et al. in 1992. It is reasonable to assume in many modalities that images of the identical or similar subjects will have similar ratios of image intensities for many voxels when the images are well-registered. Indeed, this has been observed in my previous work coregistering repeated PET scans of the same animal, where Woods et al.'s AIR program performed very well. Having a similar registration metric in ITK could thus be very useful.

Derivations

After transforming a moving image during registration, a subset of voxels in the fixed image will have computable interpolated moving image values associated with them. After discarding fixed image voxels with value 0, the voxel value ratio x(i) at each voxel I can be computed as

$$x(i) = m(i) / f(i)$$

For a sample of n ratio values, the variance M, or measure value, may be estimated

$$M = (1 / (n - 1)) * (sum(x(i)*x(i)) - (1/n)*sum(x(i))*sum(x(i)))$$

where sum(x(i)) indicates the sum of x(i) for all i.

Differentiating M with respect to a registration transform parameter T gives

$$dM/dT = (2 / (n - 1)) * (sum(x(i)*dx(i)/dT) - (1/n)*sum(x(i))*sum(dx(i)/dT))$$

where, because the moving image changes with the transform,

$$dx(i)/dT = (d/dT)(m(i) / f(i)) = (1/f(i)) * dm(i)/dT$$

If

$$f(i) = f(p(i))$$

where $p(i)$ is the physical point corresponding to the point in the fixed image f with index i, and

$$q(i) = S(p(i), T)$$

where $q(i)$ is the physical point in the moving image that the transform S maps physical point p when the transform parameter T is specified, and

$$m(i) = m(q(i)) = m(S(p(i), T))$$

and further

$$\begin{aligned} dm(i)/dT &= (d/dT)(m(i)) = (d/dT)(m(S(p(i), T))) \\ &= (dm(S(p(i), T))/dS) * (dS(p(i), T)/dT) \\ &= (dm(q(i))/dq) * (dS(p(i), T)/dT) \end{aligned}$$

where

$$dm(q(i))/dq$$

is the derivative of the moving image voxel value as the physical point in that image that is being sampled changes, or the gradient of m, and where

$$dS(p(i), T)/dT$$

is the vector of how the physical point q in the moving image that corresponds to the fixed image physical point p is changing as the transform parameter T is changed, or the Jacobian of the transform S.

From this,

$$dx(i)/dT = (1/f(i)) * [\ \mathrm{Grad}(m)\ (\mathrm{dot})\ \mathrm{Jacobian}(S)\ ]$$

where the Gradient of m and Jacobian component for each transform parameter are evaluated for each spatial dimension of f (and m) and summed.

Implementation

The metric value and derivative calculations discussed above are implemented in the RatioVarianceImageToImageMetric class in RatioVarianceImageToImageMetric.h and RatioVarianceImageToImageMetric.txx.

RatioVarianceImageToImageMetric is derived from itk::ImageToImageMetric and both are templated over the fixed and moving image classes. The metric requires no other parameters, and functions similarly to the MeanSquaresImageToImageMetric (for example).

Testing

The new metric is tested with the program in RatioVarainceTest . A CMakeLists.txt file to build the executable RatioVarainceTest is included. The program uses a 5 level multi-resolution 32 iteration registration scheme to coregister 2D images whose file names are the first two input parameters to the program, on the command line. The third input parameter is the output file name.

Input with which to test this program is included. fixed.png is a 2D grayscale image derived from a positron emission tomography (PET) image of a rat. moving.png is a translated and rotated version of fixed.png.

Running, on Linux-like systems,

```
RatioVarianceTest fixed.png moving.png moving_registered_to_fixed.png
```

should output moving_registered_to_fixed.png, which is closely registered to fixed.png. An example output of the program with that name is included in this submission.

A more rigorous test would be to introduce known transformation to an input image and then to attempt to undo this transformation by registration and compare the resulting transform with the original. If necessary, I can do this, however visual inspection of the resulting registered image suggests the metric is functioning as intended.

I have also tested on 3D PET data and observed successful registration, but data for that test is not included in this submission.

Reference

Woods RP, Cherry SR, Mazziotta JC. 1992. Rapid automated algorithm for aligning and reslicing PET images. J Comput Assist Tomogr 16(4):620-33.