N-Dimensional Computation of Strain Tensor Images in the Insight Toolkit

Release 1.0.0

Matthew McCormick[1]

May 3, 2017

[1]Kitware, Inc.

Abstract

Strain quantifies local deformation of a solid body. In medical imaging, strain reflects how tissue deforms under load. Or, it can quantify growth or atrophy of tissue, such as the growth of a tumor. Additionally, strain from the transformation that results from image-to-image registration can be applied as an input to a biomechanical constitutive model.

This document describes N-dimensional computation of strain tensor images in the Insight Toolkit (ITK), www.itk.org. Two filters are described. The first filter computes a strain tensor image from a displacement field image. The second filter computes a strain tensor image from a general spatial transform. In both cases, infinitesimal, Green-Lagrangian, or Eulerian-Almansi strain can be generated.

This paper is accompanied with the source code, input data, parameters and output data that the authors used for validating the algorithm described in this paper. This adheres to the fundamental principle that scientific publications must facilitate reproducibility of the reported results.

Latest version available at the Insight Journal [http://hdl.handle.net/10380/3573]

Contents

1 Background

The result of image registration is a spatial transformation, which is sometimes non-rigid. A spatial transformation's local deformation is quantified with the *strain tensor*. When registration is performed on images

from multiple time points, strain is a measure of local tissue growth or atrophy. For example, strain can quantify the growth of a tumor or atrophy of cerebral gray matter. When registration is performed on tissues images experiencing mechanical stimulation, strain quantifies the amount of local deformation imparted on the tissue. For instance, compression and tension of arterial plaque can be quantified from images of the arteries at multiple points in the cardiac cycle. Finally, when combined with stress in a constitutive model, the inverse problem can be solved to determined tissue viscoelasticity. For example, stiffness of a cirrhotic liver can be imaged.

Strain is a symmetric second rank tensor. *Infinitesimal strain*, also called Cauchy's strain, linear strain, engineering strain, or small strain, is a first order approximation to the Green-Lagrangian or Eulerian-Almansi strain tensors. It is computed from the *deformation gradient*, $\mathbf{F}$, as[1]

$$\varepsilon = \frac{1}{2}\left(\mathbf{F}^T + \mathbf{F}\right) - \mathbf{I} \tag{1}$$

In tensor notation and for three dimensions,

$$\begin{aligned}
\varepsilon_{ij} &= \frac{1}{2}\left(u_{i,j} + u_{j,i}\right) \\
&= \begin{bmatrix} \varepsilon_{11} & \varepsilon_{12} & \varepsilon_{13} \\ \varepsilon_{21} & \varepsilon_{22} & \varepsilon_{23} \\ \varepsilon_{31} & \varepsilon_{32} & \varepsilon_{33} \end{bmatrix} \\
&= \begin{bmatrix} \frac{\partial u_1}{\partial x_1} & \frac{1}{2}\left(\frac{\partial u_1}{\partial x_2} + \frac{\partial u_2}{\partial x_1}\right) & \frac{1}{2}\left(\frac{\partial u_1}{\partial x_3} + \frac{\partial u_3}{\partial x_1}\right) \\ \frac{1}{2}\left(\frac{\partial u_2}{\partial x_1} + \frac{\partial u_1}{\partial x_2}\right) & \frac{\partial u_2}{\partial x_2} & \frac{1}{2}\left(\frac{\partial u_2}{\partial x_3} + \frac{\partial u_3}{\partial x_2}\right) \\ \frac{1}{2}\left(\frac{\partial u_3}{\partial x_1} + \frac{\partial u_1}{\partial x_3}\right) & \frac{1}{2}\left(\frac{\partial u_3}{\partial x_2} + \frac{\partial u_2}{\partial x_3}\right) & \frac{\partial u_3}{\partial x_3} \end{bmatrix}
\end{aligned} \tag{2}$$

The *Green-Lagrangian* strain tensor, where displacement derivatives are taken with respect to material coordinates, is,

$$\mathbf{E} = \frac{1}{2}\left(\mathbf{F}^T \cdot \mathbf{F} - \mathbf{I}\right) \tag{3}$$

or

$$E_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial X_j} + \frac{\partial u_j}{\partial X_i} + \frac{\partial u_k}{\partial X_i}\frac{\partial u_k}{\partial X_j}\right) \tag{4}$$

The *Eulerian-Almansi* strain tensor, where displacement derivatives are taken with respect to spatial coordinates, is,

$$\mathbf{e} = \frac{1}{2}\left(\mathbf{F} \cdot \mathbf{F}^T - \mathbf{I}\right) \tag{5}$$

or

$$e_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} - \frac{\partial u_k}{\partial x_i}\frac{\partial u_k}{\partial x_j}\right) \tag{6}$$

Green-Lagrangian and Eulerian-Almansi are true mathematical tensors, i.e. they are invariant to translations or rotations of the coordinate system.

This article describes implementations of N-dimensional strain image filters for the Insight Toolkit (ITK).

2 Implementation

Two classes are available to compute a strain image; both output an `itk::Image` of `itk::SymmetricSecondRankTensor`'s.

The `itk::StrainImageFilter` takes a displacement field image as an input. Internally, it uses a gradient filter to compute gradients of the displacement field. The default gradient filter is the `itk::GradientImageFilter`, but a different filter can be specified by calling `SetGradientFilter()`. Other possible gradient filters include `itk::GradientRecursiveGaussianImageFilter` and `itk::DifferenceOfGaussiansGradientImageFilter`.

The `itk::TransformToStrainFilter` takes a `itk::Transform` as input an produces a strain tensor image as its output. The transform must implement its `ComputeJacobianWithRespectToPosition()` method. Similar results can be obtained by passing an arbitrary transform through the `itk::TransformToDisplacementFieldFilter` then the `itk::StrainImageFilter`.

Whether infinitesimal, Green-Lagrangian, or Eulerian-Almansi, strain is computed can be specified with `SetStrainForm()` on either filter.

3 Examples

The following example illustrates how to instantiate and use an `itk::StrainImageFilter`.

```
#include "itkStrainImageFilter.h"

[...]
  const unsigned int Dimension = 2;
  typedef float                                          PixelType;
  typedef itk::Vector< PixelType, Dimension >            DisplacementVectorType;
  typedef itk::Image< DisplacementVectorType, Dimension > InputImageType;

  typedef itk::StrainImageFilter< InputImageType, PixelType, PixelType > FilterType;

  FilterType::Pointer filter = FilterType::New();
  filter->SetInput( displacementImage );
  filter->SetStrainForm( FilterType::GREENLAGRANGIAN );

  filter->Update();

  typedef FilterType::OutputImageType TensorImageType;
  TensorImageType strainTensorImage = filter->GetOutput();
[...]
```

The following example illustrates how to instantiate and use an `itk::StrainImageFilter`.

```
#include "itkTransformToStrainFilter.h"

[...]
  const unsigned int Dimension = 2;
  typedef float                                                    PixelType;
  typedef double                                                   CoordRepresentationType;
  typedef itk::Transform< CoordRepresentationType, Dimension, Dimension > TransformType;

  typedef itk::TransformToStrainFilter< TransformType, PixelType, PixelType > TransformToStrainFilterType;

  FilterType::Pointer filter = FilterType::New();
  filter->SetStrainForm( FilterType::GREENLAGRANGIAN );

  // Create output image information.
  typedef TransformToStrainFilterType::SizeType SizeType;
  SizeType size;
  size.Fill( 20 );
  filter->SetSize( size );

  typedef TransformToStrainFilterType::SpacingType SpacingType;
  SpacingType spacing;
  spacing.Fill( 0.7 );
  filter->SetOutputSpacing( spacing );

  typedef TransformToStrainFilterType::PointType OriginType;
  OriginType origin;
  origin.Fill( -10.0 );
  filter->SetOutputOrigin( origin );

  filter->SetInput( inputTransform );

  filter->Update();

  typedef FilterType::OutputImageType TensorImageType;
  TensorImageType strainTensorImage = filter->GetOutput();
[...]
```

For additional information on the interface and how to apply it, see the class' Doxygen documentation and module tests.

References

[1] Wikipedia. Infinitesimal strain theory, 2017. [Online; accessed 03-May-2017]. 1