Helper class for initializing the grid parameters of a BSpline deformable transform by using an image as reference

Release 1.00

Luis Ibanez[1], Matt Turek[1], Stephen Aylward[1]

April 22, 2008

[1]Kitware Inc., Clifton Park, NY

Abstract

This document describes a simple helper class intended for making easy to initialize the grid parameters of a `itk::BSplineDeformableTranform`.

This paper is accompanied with the source code, input data, parameters and output data that we 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.

Contents

1 Introduction

When using a BSplineDeformableTranform for the purpose of performing deformable registration or for the purpose of resampling an image, it is customary to initialize the parameters of the BSpline grid based on the parameters of the fixed image or based on the parameters of image that is the output space of the resampling process. This initialization, however, requires the software developer to remember to set several critical details. Forgetting any of these details can easily lead to a failed registration or a failed resampling. The process of finding the source of the problem can be a very frustrating experience.

The class described in this report is intended to simplify the initialization of a BSplineDeformableTranform and to ensure that no small detail is missed in the process.

2 How to use this class

The two files in the ”Testing” directory of this report provide typical examples of how to use this initializer class. Please refer to that code for a fully functional example.

Just for convenience, a minimal code snipped is shown below

2.1 Usage Case 1

In this example we illustrate the use of the method `SetGridSizeInsideTheImage`.

```
typedef itk::BSplineDeformableTransformInitializer<
               TransformType,
               FixedImageType >      InitializerType;

InitializerType::Pointer transformInitilizer = InitializerType::New();


transformInitilizer->SetTransform( bsplineTransform );
transformInitilizer->SetImage( fixedImage );

typedef TransformType::RegionType RegionType;
RegionType::SizeType   size;

const unsigned int numberOfGridNodesInsideTheImageSupport = 5;

size.Fill( numberOfGridNodesInsideTheImageSupport );

transformInitilizer->SetGridSizeInsideTheImage( size );

transformInitilizer->InitializeTransform();
```

2.2 Usage Case 2

In this case we don’t use the method `SetGridSizeInsideTheImage`. Instead, we invoke the equivalent method `SetNumberOfGridNodesInsideTheImage`.

```
typedef itk::BSplineDeformableTransformInitializer<
               TransformType,
               FixedImageType >      InitializerType;

InitializerType::Pointer transformInitilizer = InitializerType::New();


transformInitilizer->SetTransform( bsplineTransform );
```

```
transformInitilizer->SetImage( fixedImage );
transformInitilizer->SetNumberOfGridNodesInsideTheImage( 5 );

transformInitilizer->InitializeTransform();
```

3 Software Requirements

In order to reproduce the results described in this report you need to have the following software installed:

- Insight Toolkit 3.6.
- CMake 2.4.8

If you want to generate this PDF document itself then you will also need a functional LaTeX installation. This should be trivial in Linux.

A Results

The results described in this section can be reproduced by running the Tests in the `Testing` subdirectory of this report. To build this paper and its associated tests, please do the following:

- cmake .
- make
- ctest

The first command will configure the build tree by using CMake. The second command will build the executables and will build this PDF document. The last command will run the tests that resample an image with a BSplineDeformableTranform.

Figure 1 illustrates the input and output of the first test described in 2.1.

Figure 2 illustrates the input and output of the second test described in 2.2.

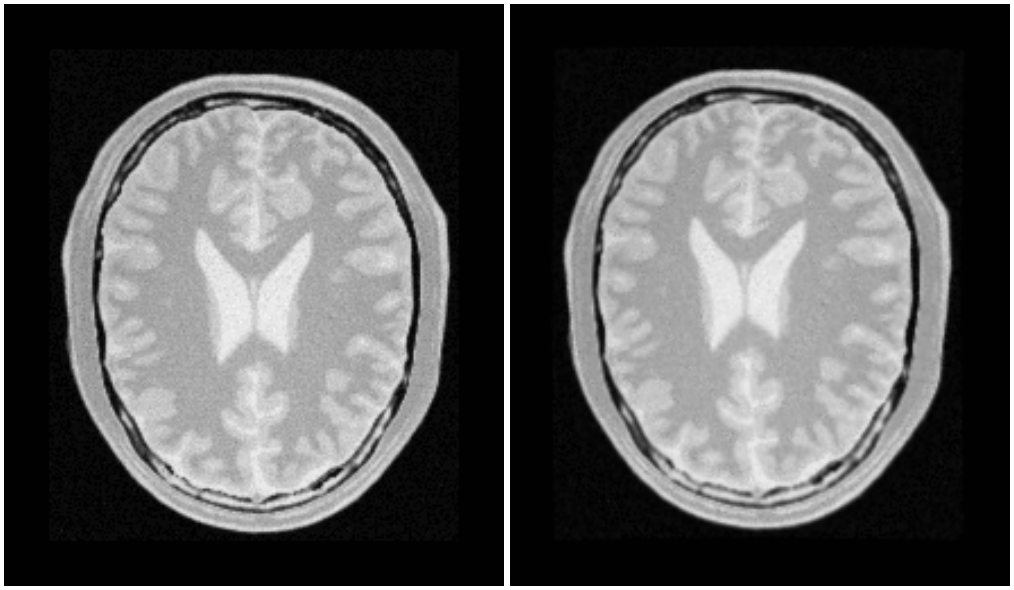

Figure 1: Left: Input to the resampling filter. Right: Output of the resampling filter. The image has been deformed by introducing deformations in five grid nodes located towards the center of the image. This image was generated with the source code illustrated in 2.1.

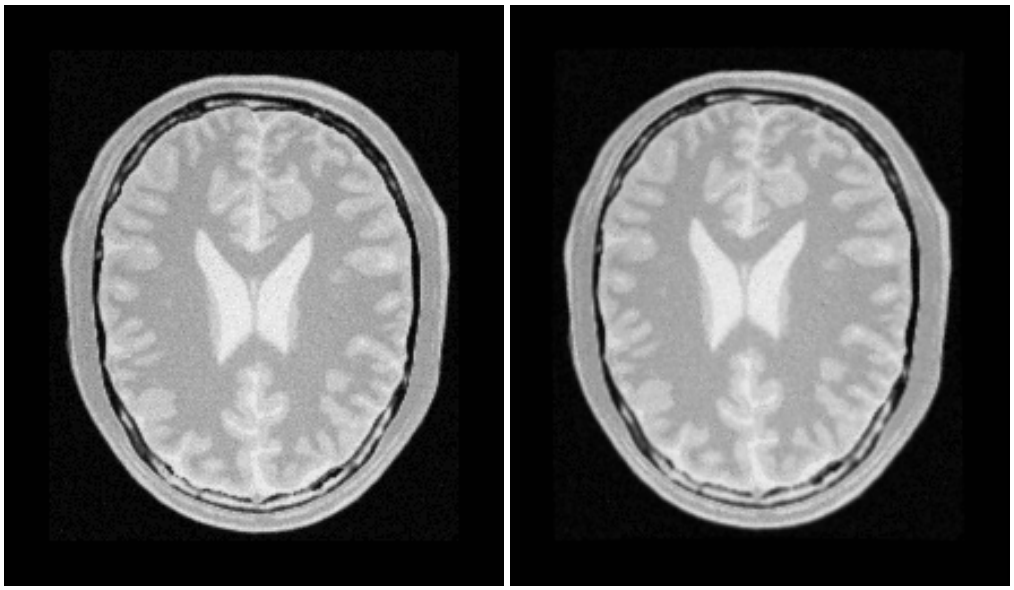

Figure 2: Left: Input to the resampling filter. Right: Output of the resampling filter. The image has been deformed by introducing deformations in five grid nodes located towards the center of the image. This image was generated with the source code illustrated in 2.2.