A Skull-Stripping Filter for ITK

Release 1.0

Stefan Bauer¹, Thomas Fejes¹,² and Mauricio Reyes¹

April 30, 2012

¹Institute for Surgical Technology and Biomechanics, University of Bern, Switzerland
²Department of Neuroradiology, Inselspital, Bern University Hospital, Switzerland

stefan.bauer@istb.unibe.ch

Abstract

Skull-stripping (or brain extraction) is an important pre-processing step in neuroimage analysis. This document describes a skull-stripping filter implemented using the Insight Toolkit ITK, which we named itk::StripTsImageFilter. It is a composite filter based on existing ITK classes. The filter has been implemented with usability, robustness, speed and versatility in mind, rather than accuracy. This makes it useful for many pre-processing tasks in neuroimage analysis.

This paper is accompanied by the source code, input data and a testing environment.

Latest version available at the Insight Journal [http://hdl.handle.net/10380/3353]
Distributed under Creative Commons Attribution License

Contents

1 Introduction 2
2 Methods 2
3 Implementation 3
4 Evaluation 3
5 Conclusion 4
6 User Guide 4
1 Introduction

Skull-stripping is an important pre-processing step for analyzing medical images of the head region. Skull-stripping separates the brain from the skull and other surrounding structures. This is required by many algorithms for automatic segmentation or registration of brain images, as well as for visualization of the results.

In the last years, many different approaches have been proposed to solve this problem. They range from simple morphological operations to advanced model-based methods or sophisticated classification-based approaches [2, 4, 6, 9, 13, 11, 12, 15, 16, 19]. Arguably the most-used tool is the brain extraction tool (BET) developed by Smith et al. [16]. Although there have been a number of comparison papers [5, 7, 10] and even one online resource for directly comparing different algorithms [14], the results are inconclusive. Recently, there have also been methods proposed, which are targeted at specific applications, as an example skull-stripping for tumor-bearing brain images [17, 1].

In general, the majority of the current methods are tuned to work on T$_1$-weighted magnetic resonance images (MRI). The problem with some of the most accurate methods is, that they require a significant amount of computation time, which makes them not very useful for preprocessing purposes. Additionally, most implementations are not publicly available and to the best of our knowledge, there is no filter with publicly available source code for the Insight Toolkit (ITK) [8] www.itk.org. An ITK filter, which is easy to use, versatile, robust and has a fast run-time, would be important for being able to directly integrate this preprocessing step into individual applications. We want to address this need by contributing an ITK class for skull-stripping of medical images.

2 Methods

We implemented a method previously described in [1]. It is based on a two step procedure. Initially, an atlas image is registered to the patient image using an affine transformation model and the brain mask of the atlas is propagated to the patient image with the calculated transformation matrix [20]. This brain mask is eroded and then serves as initialization for a refined brain extraction based on level-sets. The level-set is evolved towards the edge of the brain-skull border with dedicated expansion, curvature and advection terms [18].

As an atlas, we use a customized isotropic T$_1$-weighted MR image from the IXI dataset $^1$ with a manually generated brain mask.

$^1$http://www.brain-development.org/
3 Implementation

The method has been implemented as an ITK filter, named

```
itk::StripTsImageFilter
```

which can be used within a standard ITK pipeline. It is designed as a composite filter, making use of
standard ITK classes, namely the registration framework and the level-set framework. It is derived from the
```
itk::ImageToImageFilter```
class and thus implements the standard functions. The filter requires 3 inputs:

- `SetInput()`
- `SetAtlasImage()`
- `SetAtlasMask()`

and outputs a binary brain mask image, which can be used to extract the brain from the patient image.

In order to make the method more robust, after geometrical initialization with the `itk::CenteredTransformInitializer`, we first perform a multi-resolution `itk::VersorRigid3DTransform`, followed by an `itk::AffineTransform` using the `itk::MattesMutualInformationImageToImageMetric`. This allows us to handle cases, where the atlas has a different modality than the patient image. For speed reasons, the initial registration is performed on a subsampled version of the patient image. The final refinement of the brain-skull border is carried out by the `itk::GeodesicActiveContourLevelSetImageFilter` in two resolution levels in order to increase robustness and speed. Additionally, in order to be able to handle various input image resolutions easily, all computations are performed in a standard space of 1mm isotropic resolution.

The current implementation assumes that the images are 3-dimensional. It can be applied to a wide range of different image modalities and does not require any specific parameter settings, but when necessary, the user can fine-tune the parameters of the registration or level-set segmentation according to his requirements.

4 Evaluation

We applied the proposed method to a T1-weighted MR image from the synthetic Brainweb database [3] and show the result on one axial slice in figure 1.

![Figure 1: Skull-stripping result shown on an axial slice of one T1-weighted Brainweb image. Left: original image with mask overlay. Right: Brain tissues after applying the skull-stripping mask.](http://hdl.handle.net/10380/3353)
In order to demonstrate the versatility of the method, we also applied it to different medical image modalities using the same atlas and parameters settings. Figure 2 illustrates the results on $T_1$, $T_{1\text{contrast}}$, $T_2$ and $T_{2\text{flair}}$ MR images of varying resolutions, as well as one CT image. We also applied the method to a database of around 50 tumor-bearing brain MR images with visually convincing results. Only on fMRI data the results were not satisfactory.

![Figure 2: Skull-stripping results shown on a sagittal slice of different patient images. From left to right: Anisotropic $T_1$ MRI (axial acquisition), isotropic $T_{1\text{contrast}}$ MRI (sagittal acquisition), anisotropic $T_2$ MRI (axial acquisition), anisotropic $T_{2\text{flair}}$ MRI (coronal acquisition), isotropic CT.](image)

In order to provide quantitative evaluation, we applied the filter to a database of 40 healthy $T_1$-weighted MR images from the segmentation validation engine [14]. With standard settings, we achieve an average Dice coefficient of 0.91 with a standard deviation of ±0.02 on these images.

Computation time is between 1 and 4 minutes on a standard PC.

5 Conclusion

We implemented a method for automatic skull-stripping of volumetric medical images as an ITK filter and provide the source code along with this article. The method is based on initial atlas registration with subsequent level-set refinement of the brain mask. The advantages of this filter include its ease-of-use, the robustness also in case of lesions, a reasonable computation time and the ability to be used on a number of different image modalities. The accuracy is sufficient for many tasks in neuroimage analysis.

6 User Guide

We provide a short C++ program `skullStripping.cxx` demonstrating the use of the filter. It works on 3D medical images and can also serve as an example how to use the `itk::StripTsImageFilter` in an individual project.

For the command line program, the user has to provide the patient image filename, the atlas image filename and the atlas mask filename as a command line argument. If the program is called without any arguments, a brief help is displayed.

For example in a Windows environment, you would use:

```
skullStripping.exe patientImageFilename atlasImageFilename atlasMaskFilename
```

Latest version available at the Insight Journal [http://hdl.handle.net/10380/3353]
Distributed under Creative Commons Attribution License
The following excerpt from the skullStripping.cxx main file shows how to use the `itk::StripTsImageFilter` in an individual project and how to use the output for masking the patient image.

```cpp
// set up skull-stripping filter
typedef itk::StripTsImageFilter<ImageType, AtlasImageType, AtlasLabelType> StripTsFilterType;
StripTsFilterType::Pointer stripTsFilter = StripTsFilterType::New();

// set the required inputs for the stripTsImageFilter
stripTsFilter->SetInput( reader->GetOutput() );
stripTsFilter->SetAtlasImage( atlasReader->GetOutput() );
stripTsFilter->SetAtlasBrainMask( labelReader->GetOutput() );

try
{
    stripTsFilter->Update();
} catch ( itk::ExceptionObject &exception )
{
    std::cerr << "Exception caught ! " << std::endl;
    std::cerr << &exception << std::endl;
}

// mask the patient image using the output generated from the stripTsImageFilter as mask
typedef itk::MaskImageFilter<ImageType, AtlasLabelType, ImageType> MaskFilterType;
MaskFilterType::Pointer maskFilter = MaskFilterType::New();

maskFilter->SetInput1( reader->GetOutput() );
maskFilter->SetInput2( stripTsFilter->GetOutput() );

try
{
    maskFilter->Update();
} catch ( itk::ExceptionObject &exception )
{
    std::cerr << "Exception caught ! " << std::endl;
    std::cerr << &exception << std::endl;
}
```

Latest version available at the Insight Journal [http://hdl.handle.net/10380/3353]
Distributed under Creative Commons Attribution License
References


Latest version available at the *Insight Journal* [http://hdl.handle.net/10380/3353] Distributed under *Creative Commons Attribution License*


