Label overlay

Gaëtan Lehmann

April 27, 2006

Unité de Biologie du Développement et de la Reproduction, Institut National de la Recherche Agronomique, 78350 Jouy-en-Josas, France

Abstract

LabelOverlayImageFilter colorize a label image and put it on top of the input image.

1 Introduction

ITK have a class to colorize a label image, `itk::ScalarToRGBPixelFunctor`, but this class doesn't allow to visualize the result on top of the input image. `LabelOverlayImageFilter` colorize a label image and put it on top of the input image. The user can choose the opacity of the colored label image with the method `SetOpacity()`. Another class, `LabelToRGBImageFilter`, is available to colorize the label only. Those two classes are particulary useful with all algorithms which produce a labeled image[1].

2 Implementation

`LabelOverlayImageFilter` is a subclass of `BinaryFunctorImageFilter`. The functor have a predefined list of distinct colors. The following expression is applied for all pixels:

```
output = ( opacity * ( colorList[ label % size ] ) ) + ( ( 1 - opacity ) * input )
```

The Opacity parameter must be between 0 and 1 and defaults to 0.5.

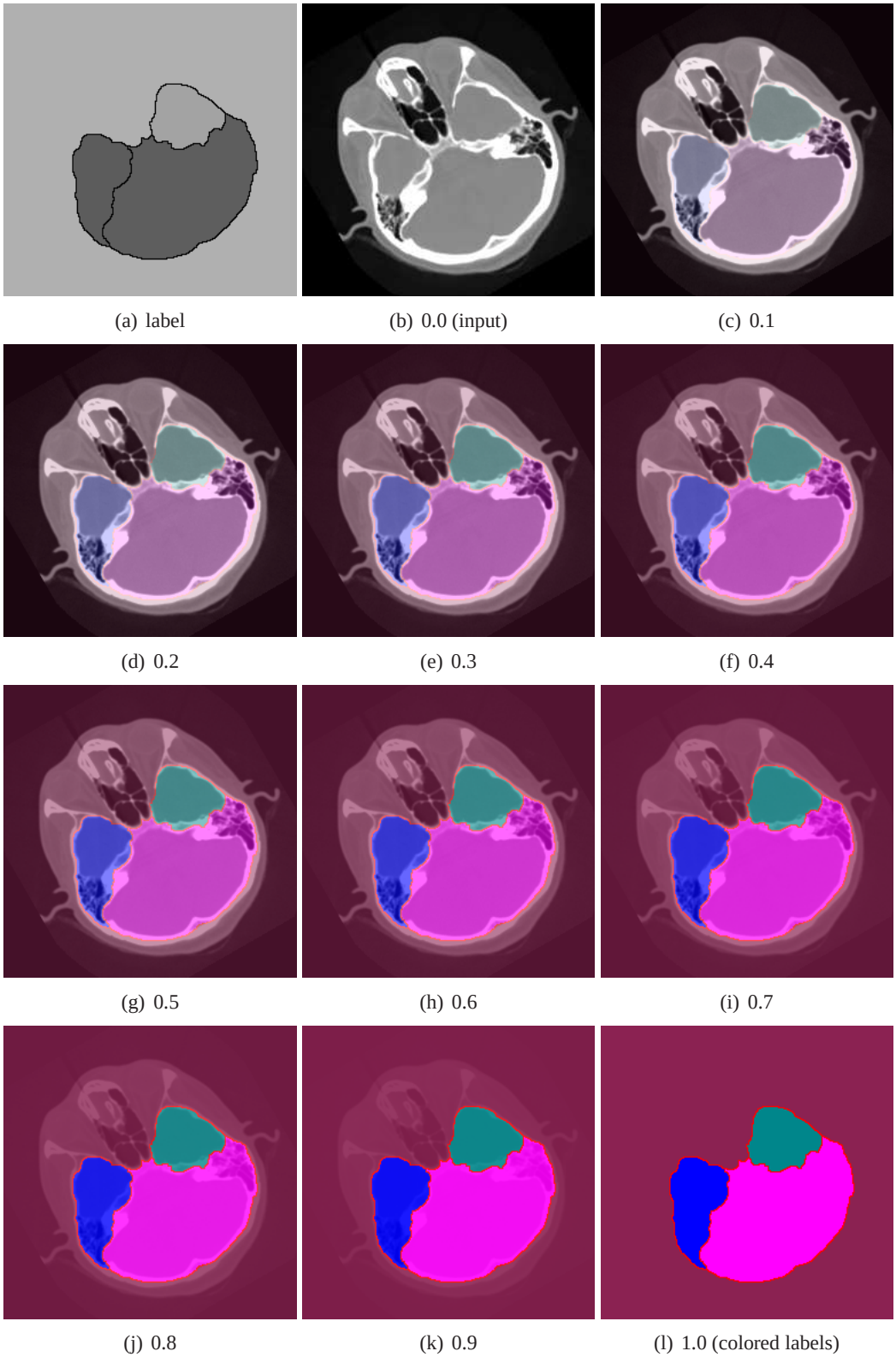
3 Examples

The examples shown here are the result of a segmentation by watershed with markers.

4 Sample code

```
#include "itkImageFileReader.h"
```

[1] Those classes have been developed to be used with the morphological watershed filter (article still in preparation).



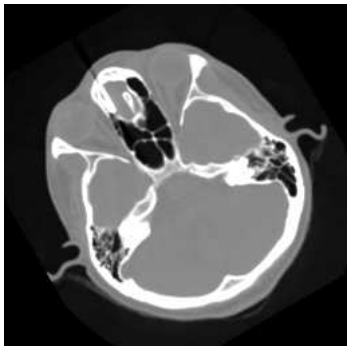
(a) label (b) 0.0 (input) (c) 0.1

(d) 0.2 (e) 0.3 (f) 0.4

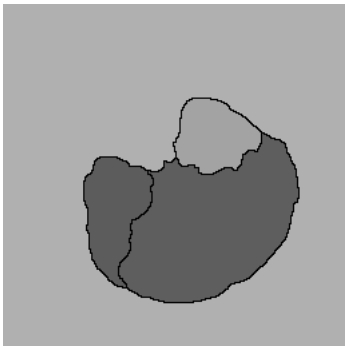
(g) 0.5 (h) 0.6 (i) 0.7

(j) 0.8 (k) 0.9 (l) 1.0 (colored labels)

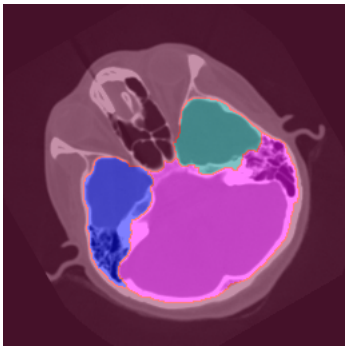
Figure 1: Label image and results of LabelOverlayImageFilter for different opacity values



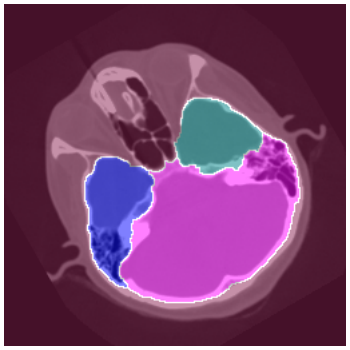
(a) input



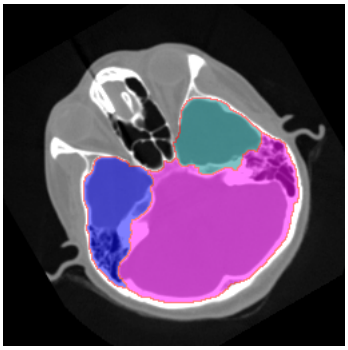
(b) labels



(c) Opacity=0.5, UseBackground=false (default), BackgroundValue=0 (default)

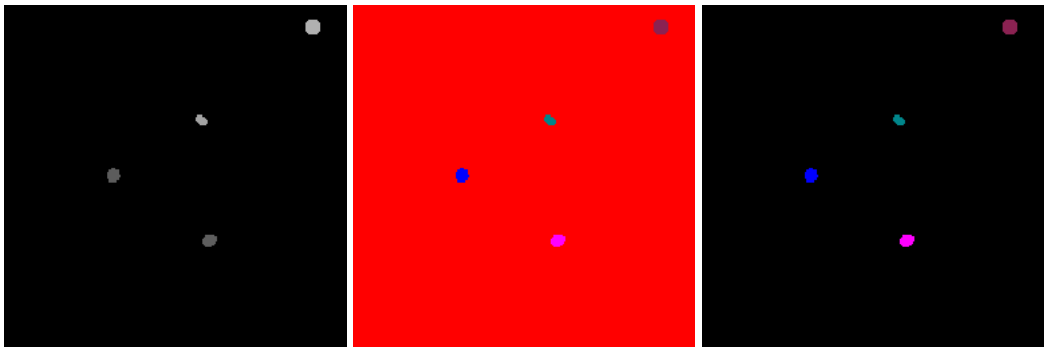


(d) Opacity=0.5, UseBackground=true, BackgroundValue=0 (default)



(e) Opacity=0.5, UseBackground=true, BackgroundValue=176

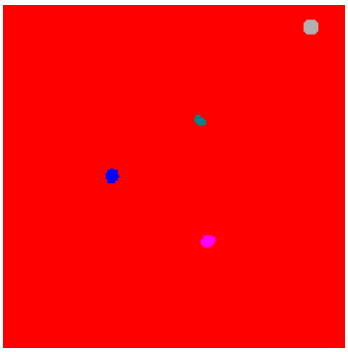
Figure 2: Result of LabelOverlayImageFilter with different parameters.



(a) input

(b) UseBackground=false (default), BackgroundValue=0 (default)

(c) UseBackground=true, BackgroundValue=0 (default)



(d) UseBackground=true, BackgroundValue=176

Figure 3: Label image and results of LabelToRGBImageFilter with different parameters.

```
#include "itkImageFileWriter.h"
#include "itkCommand.h"
#include "itkSimpleFilterWatcher.h"

#include "itkLabelOverlayImageFilter.h"


int main(int argn, char * argv[])
{

  if( argn == 1 )
    {
    std::cerr << "Usage: check2 input label output [opacity [useBg [bg]]]" << std::endl;
    return 1;
    }

  const int dim = 2;

  typedef unsigned char PType;
  typedef itk::Image< PType, dim > IType;

  typedef itk::RGBPixel<unsigned char> CPType;
  typedef itk::Image< CPType, dim > CIType;

  typedef itk::ImageFileReader< IType > ReaderType;
  ReaderType::Pointer reader = ReaderType::New();
  reader->SetFileName( argv[1] );
  ReaderType::Pointer reader2 = ReaderType::New();
  reader2->SetFileName( argv[2] );

  typedef itk::LabelOverlayImageFilter< IType, IType, CIType> FilterType;
  FilterType::Pointer filter = FilterType::New();
  filter->SetInput( reader->GetOutput() );
  filter->SetLabelImage( reader2->GetOutput() );
  if( argn >= 5 )
    { filter->SetOpacity( atof(argv[4]) ); }
  if( argn >= 6 )
    { filter->SetUseBackground( atoi(argv[5]) ); }
  if( argn >= 7 )
    { filter->SetBackgroundValue( atoi(argv[6]) ); }

  itk::SimpleFilterWatcher watcher(filter, "filter");

  typedef itk::ImageFileWriter< CIType > WriterType;
  WriterType::Pointer writer = WriterType::New();
  writer->SetInput( filter->GetOutput() );
  writer->SetFileName( argv[3] );
  writer->Update();

  return 0;
}
```

5 Acknowledgments

I thank Richard Beare for the good set of colors.

References

[1] L. Ibanez and W. Schroeder. *The ITK Software Guide*. Kitware, Inc. ISBN 1-930934-10-6, http://www.itk.org/ItkSoftwareGuide.pdf, 2003.