Tophap by area

Gaëtan Lehmann[1]

November 2, 2010

[1]INRA, UMR 1198; ENVA; CNRS, FRE 2857, Biologie du Développement et Reproduction, Jouy en Josas, F-78350, France.

Abstract

The tophat transform is a well know and widely used transform based of morphological opening or closing. It is often used to remove the background in an image, where the background is defined as the zones which do not fit in the user provided kernel.

Similarly, the tophat by area is used to remove the background in an image, but the criteria of selection is the size of the connected component, either in physical size or in number of pixels, and is independant of the shape of the connected component. This property makes the tophat non dependant of the shape of the object and makes easier to describe the criteria.

Latest version available at the Insight Journal [http://hdl.handle.net/10380/3228]

Contents

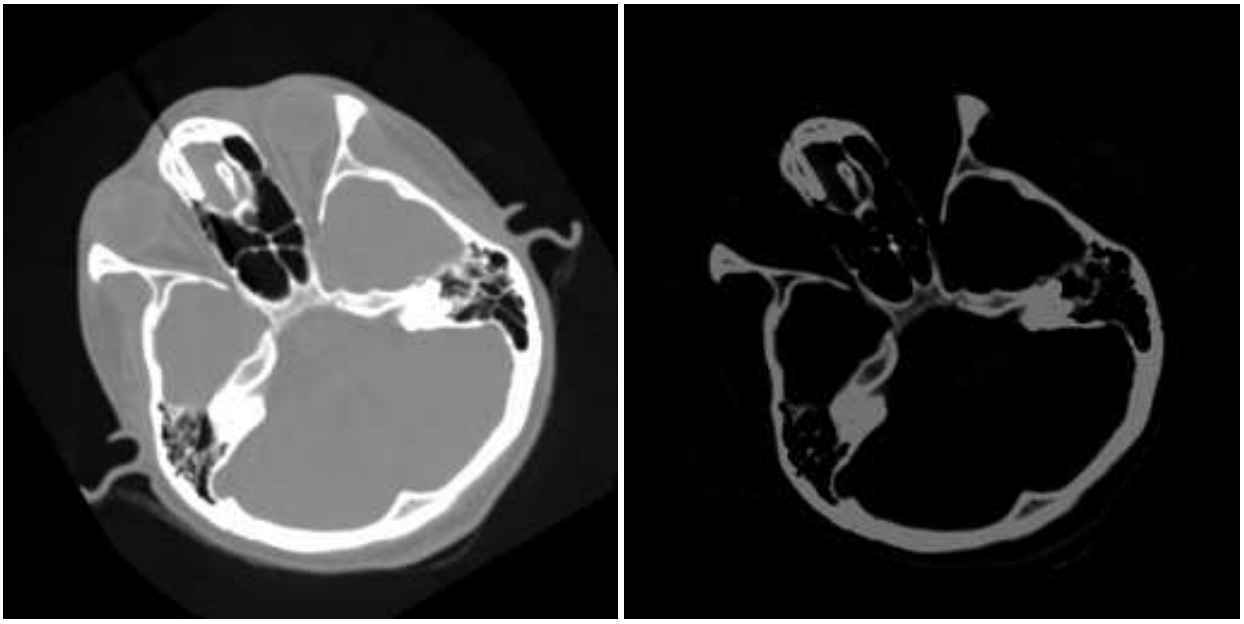
1 Tophat by area

The white tophat by area is used when the background is black to keep only the object smaller than the criteria provided by the user. It is computed as

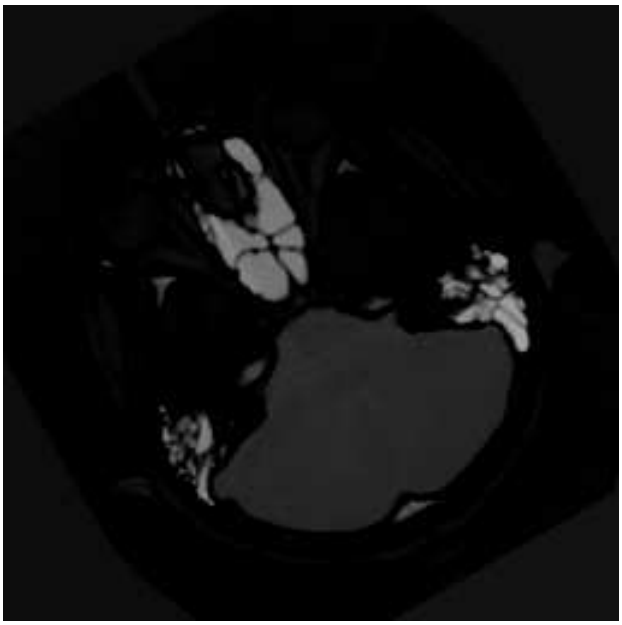
$$whiteTophat = input - areaOpening(input) \tag{1}$$

The black tophat by area is used when the background is white. It is computed as

$$blackTophat = input - areaClosing(input) \tag{2}$$



(a) Input image. (b) White area tophat.



(c) Black area tophat.

Figure 1: White and black tophat with a criteria of 1000, using image spacing and a 4-connectivity.

2 Implementation

The white tophat by area is implemented in `itk::WhiteTopHatByAreaImageFilter` and the black tophat by area in `BlackTopHatByAreaImageFilter`. They both provide the same options to the user:

- `Set/GetLambda()` to set or retrietve the criteria;
- `Set/GetUseImageSpacing()` to set/get whether the image spacing should be used or not. If not used, the size of the connected components is computed in pixels.

Here is an example of usage for the white tophat by area. The black tophap by area filter is used in the exact same way.

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

#include "itkWhiteTopHatByAreaImageFilter.h"


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

  if( argc != 6 )
    {
    std::cerr << "usage: " << argv[0] << " intput output lambda connectivity useSpacing" << std::endl;
    std::cerr << " input: the input image" << std::endl;
    std::cerr << " output: the output image" << std::endl;
    // std::cerr << "  : " << std::endl;
    exit(1);
    }

  const int dim = 2;

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

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

  typedef itk::WhiteTopHatByAreaImageFilter< IType, IType > FilterType;
  FilterType::Pointer filter = FilterType::New();
  filter->SetInput( reader->GetOutput() );
  filter->SetLambda( atof(argv[3]) );
  filter->SetFullyConnected( atoi(argv[4]) );
  filter->SetUseImageSpacing( atoi(argv[5]) );

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

  typedef itk::ImageFileWriter< IType > WriterType;
  WriterType::Pointer writer = WriterType::New();
```

```
  writer->SetInput( filter->GetOutput() );
  writer->SetFileName( argv[2] );
  writer->Update();

  return 0;
}
```

3 Wrapping

The two new filters are wrapped with WrapITK.

References

[1] Richard Beare. Grayscale morphological attribute operations. *The Insight Journal*, 2007.

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