Variance Image Filter

Release 0.01

Robert Tamburo[1]

December 9, 2010

[1]robert.tamburo@gmail.com

Abstract

This paper describes an intensity image filter http://www.itk.org/Doxygen318/html/group__IntensityImageFilters.html for computing the variance of pixel values contained within a neighborhood centered at each input pixel. The output image contains the calculated variance at each input pixel location. This paper is accompanied with source code for the filter and test, test images and parameters, and expected output images.

Latest version available at the Insight Journal [http://hdl.handle.net/10380/3236]

Contents

1 Implementation of Algorithm

The variance image filter is implemented in exactly the same way as itk::NoiseImageFilter.

Figure 1: The input image `sf4.png` for this example.

2 Example

This filter requires setting an input image `SetInput()` and the radius of the neighborhood with `SetRadius()`. The neighborhood of the radius defaults to 1. Example usage is shown below.

```
typedef itk::Image<unsigned char, 2> ImageType;
ImageType::Pointer image = ImageType::New();
typedef itk::Image<unsigned char, 2> ImageType;
typedef itk::VarianceImageFilter<ImageType, ImageType> VarianceFilterType;
VarianceFilterType::Pointer varianceFilter = VarianceFilterType::New();
varianceFilter->SetInput(inputImage); // set the input image
ImageType::SizeType radius;
radius.Fill(10); // set the desired neighborhood radius
radius->SetRadius(radius);
varianceFilter->Update();
```

For this example, `sf4.png` (Fig. 1) from the `Data` directory is used as the input image. Shown in Figure 2 is the output with default radius and shown in Figure 3 is the output with a radius of 5.

3 Testing

The test code included with this article will generate images in Figures 2 and 3 with the folloiwing arguments.

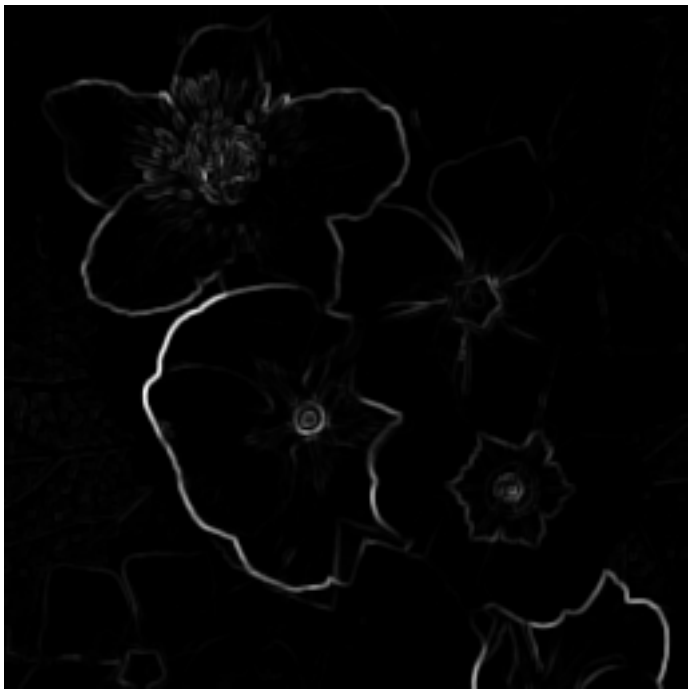

Figure 2: The output image with default radius.

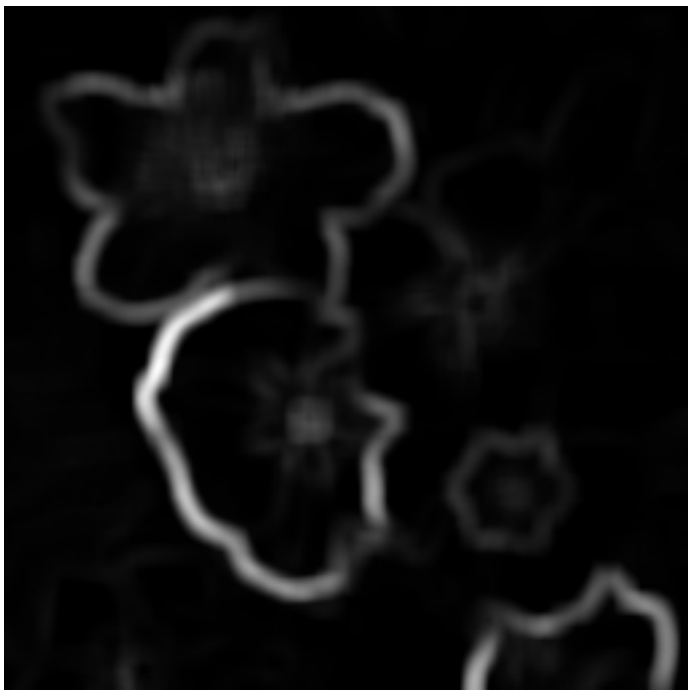

Figure 3: The output image with a radius of 5.

```
VarianceImageFilter.exe sf4.png sf4_out1.png 1
VarianceImageFilter.exe sf4.png sf4_out5.png 5
```

4 Software Used

This filter was developed on a Windows 7 64-bit computer. It has been successfully tested with ITK version 3.18.0, MinGW version 5.1.6, and CMake version 2.8.2 (Windows binary), and gcc version 4.3.4 20090804 release 1 under cygwin.