Stochastic Fractal Dimension Image

Nicholas J. Tustison and James C. Gee

February 7, 2009

Penn Image Computing and Science Laboratory
University of Pennsylvania

Abstract

Fractal analysis for medical image classification and analysis was introduced in [2]. According to the authors, when viewed as an intensity surface, Mandelbrot's fractal theory provides an informative framework for characterizing such a surface. Using the fractional Brownian motion model, the authors provide an algorithm for converting a scalar image to a fractal dimension image for classification purposes or edge enhancement. This submission constitutes a report on the ITK implementation of this algorithm.

1 Introduction

The work in [2] describes a method for converting a scalar image to a stochastic fractal dimension image. The methodology is based on Madelbrot's fractal theory and the concept of fractional Brownian motion and yields images which have been used for classification and edge enhancement.

2 Implementation

This submission includes a single class `itk::ScalarToFractalImageFilter` derived from the `itk::ImageToImageFilter` class. The input is a scalar image, an optional neighborhood radius (default = 2), and an optional mask. The mask can be specified to decrease computation time since, as the authors point out, calculation is time-consuming. The authors write (on page 141)

> It took about 1 week to get a 512×512 size fractal dimension image using a standard personal computer, while a high-speed image computer' cut down the calculation time to only 5 h.

Given that this was written in 1989, a note of comparison is in order. The Lenna example contained in the next section is of the same size and same radius ($= 3$) and took, on a MacBook Pro (first generation), $\sim$10-15 minutes.

Usage is straightforward and is demonstrated with the included testing routine `itkScalarToFractalImageFilterTest.cxx`. The user instantiates the filter and sets the input.

```
  typename FractalFilterType::RadiusType radius;
  radius.Fill( atoi( argv[4] ) );
  fractal->SetNeighborhoodRadius( radius );
```

Optionally, one can set the neighborhood radius

```
typedef itk::ScalarToFractalImageFilter<ImageType, ImageType>
  FractalFilterType;
typename FractalFilterType::Pointer fractal = FractalFilterType::New();
fractal->SetInput( imageReader->GetOutput() );
```

as well as a mask image to decrease computational time.

```
  fractal->SetMaskImage( thresholder->GetOutput() );
```

3 Sample Results

Images taken from the USC-SIPI database [1] are provided in Figure 1. Their corresponding fractal images (radii of 1,2 as specified in the file `CMakeLists.txt`) are located in Figure 2. Additionally, we apply the filter to the Lenna image in Figure 3.

References

[1] Usc-sipi databse. Texture image database, available at http://sipi.usc.edu/database/, 2008. 3, 1

[2] C.-C. Chen, J.S. DaPonte, and M.D. Fox. Fractal feature analysis and classification in medical imaging. *IEEE Transactions on Medical Imaging*, 8(2):133–142, 1989. (document), 1

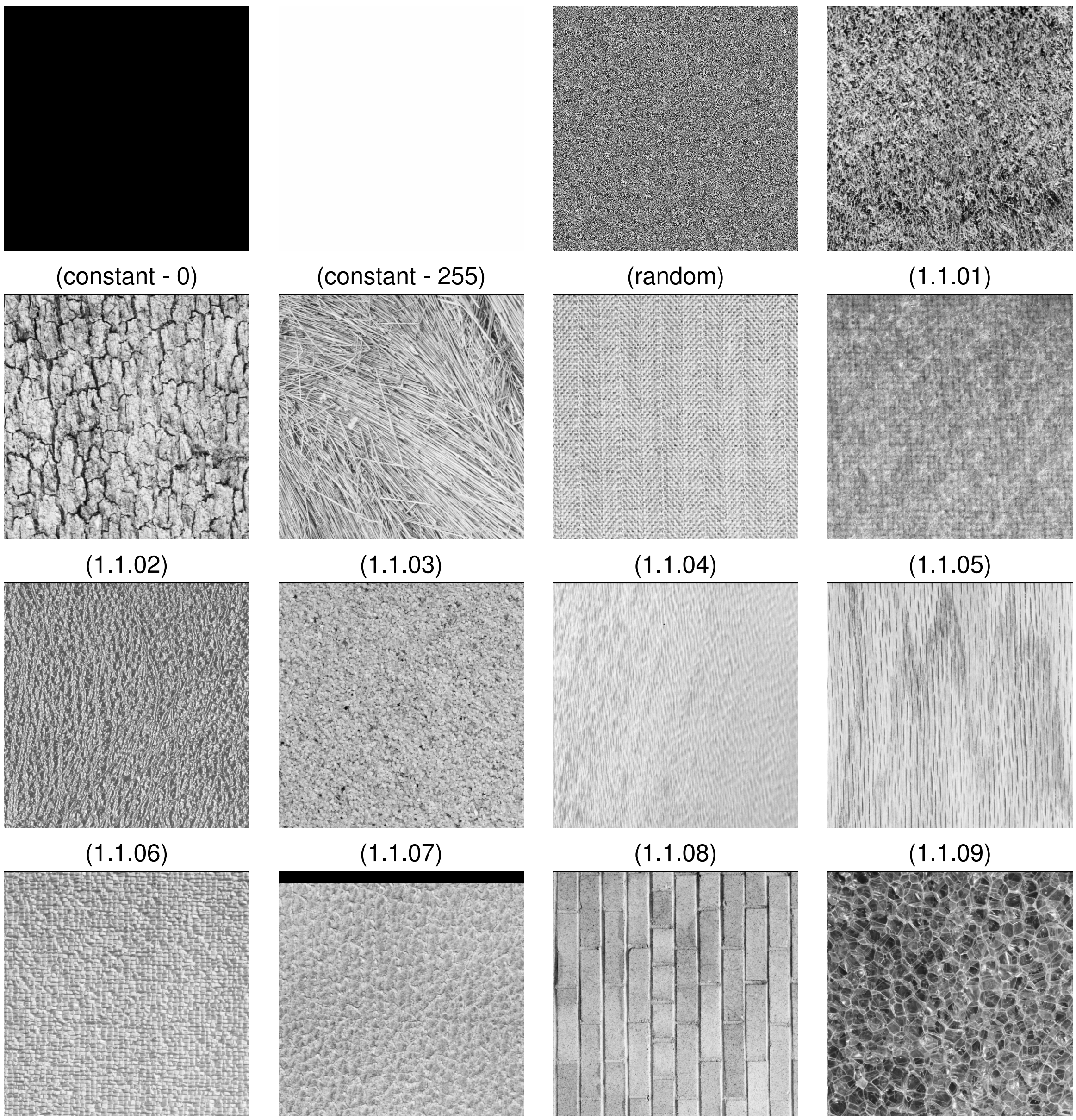

Figure 1: Sample Brodatz textures taken from the USC-SIPI texture database [1].

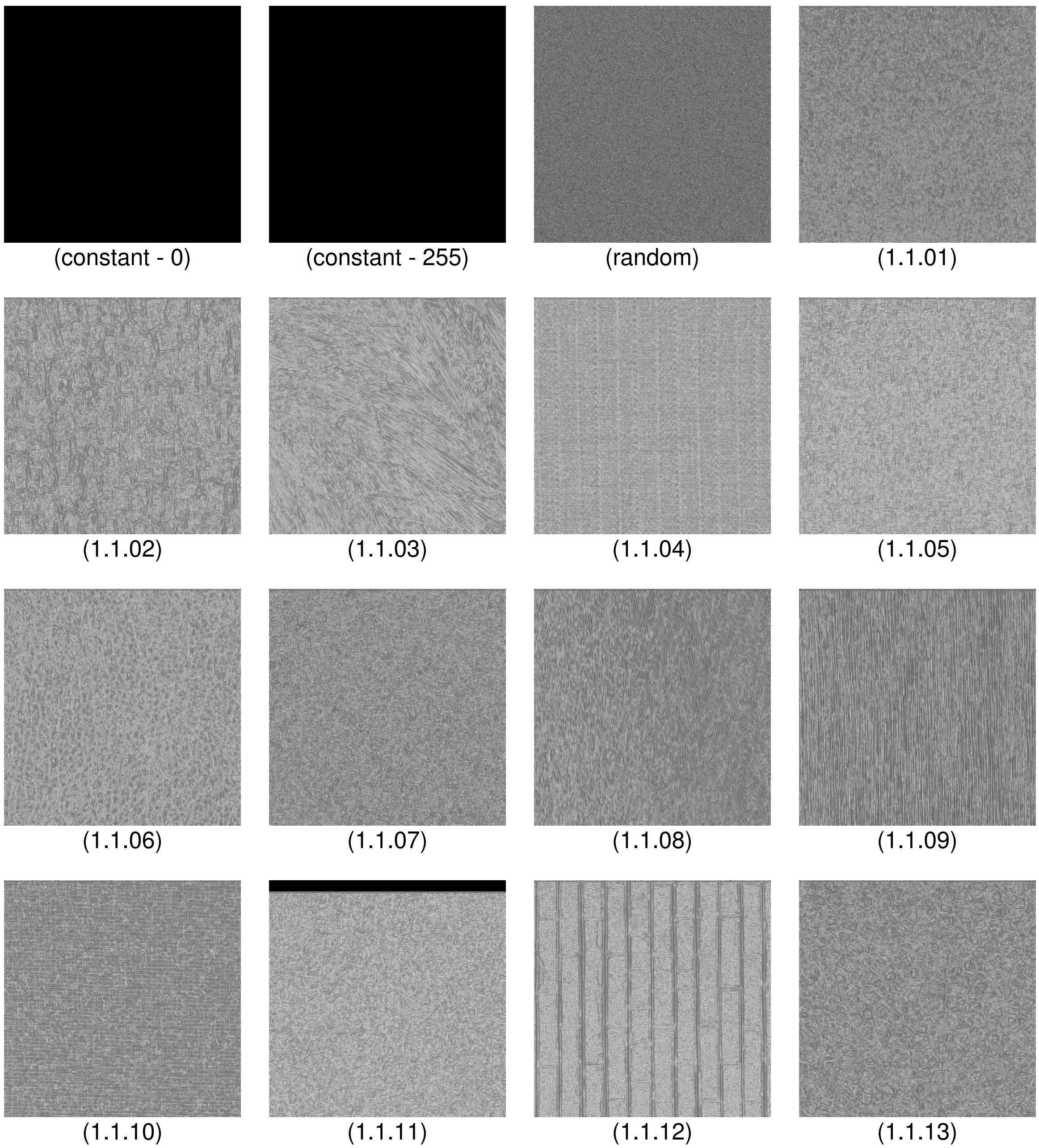

Figure 2: Fractal images of the sample Brodatz textures shown in Figure 1.

Figure 3: Fractal image of the Lenna image with radius = 3.