Programming Assignment 2: Fourier Transform

James Santiago

18 October 2007

IST 562 Prof: James Wang

Contents

Introduction

The goal for this assignment is to create a program that utilizes the Fourier Transform to discern signals from noise during image recognition. The task at hand is to count threads shown on an x-ray image of a painting. The Fourier Transform can benefit us in accomplishing this as its transformation of data into orthonormal functions allows us to analyze the results in terms of frequency and its underlying components such as the magnitude and phase. The threads in our images are similar to a signal transmission as there are strong patterns of pixels that are speckled with noise. The Fourier Transform can assist us in determining what is a thread and what is noise. It is in the hope of proper determination of a thread that the image can be analyzed so that we can determine where in the image there is an abnormal number of threads.

Microsoft's C# and the .NET framework will be used as the programming language to accomplish this project.

Assignment Steps

Step 1: find and reference a Fourier Transform framework

AForge.NET was chosen and tested as a suitable adaptation of basic Fourier Transform functions. The framework is an open source collaboration project that also has classes for image processing and other academic level functions.

Step 2: create application layout and apply Fourier Transform to uploaded image

The second step was to create the basic layout for the application and to ensure that the Fourier Transform function within the AForge framework could be properly applied to the supplied images. It was found that the image supplied must have a height and width that are both a power of two. A

method to adjust the size for any given image was implemented to resolve this issue. I also decided to apply the Fourier transform to horizontal cuts of the image that had a height of two pixels.

Step 3: analyze Fourier data

The next step was taken so that a proper strategy could be devised that could optimally count the threads in the image. I ended up deciding to remove low magnitude values instead of filtering the data by frequency.

Step 4: use transform data to count horizontal and vertical threads

After applying an inverse Fourier Transform on each cut a function called a blob counter was used from the AForge framework. The counter grouped pixels together that were not separated by any color darker than gray. This effectively counted all the "bright spots" which amounted to the number of threads. The process was done horizontally to count the vertical threads and vertically to count the horizontal threads.

Step 5: create a color overlay according to thread density

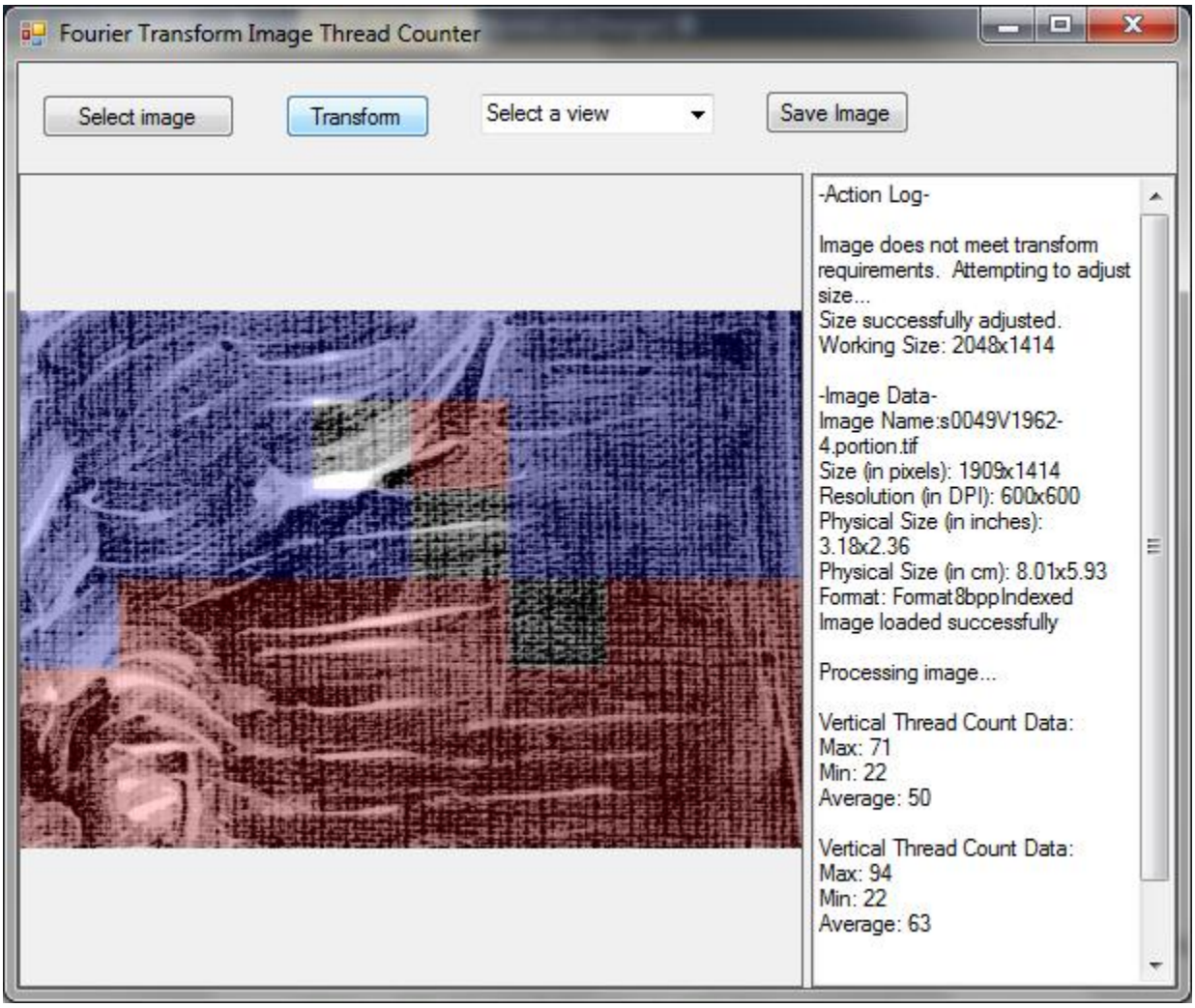
The last step involved breaking the image up into approximately 2cm x 2cm blocks. Then the threads in each block were compared to the average thread count of the entire picture. Above average thread counts were painted red, below average were painted blue and the rest were left alone.

How to Run the Program

The program is implemented with a graphical user interface. One the program is started an image will need to be selected. From there the transform button can be clicked to apply all the steps to the image. The resulting display will be the overlay image but other images that were created during the process can also be viewed and saved.

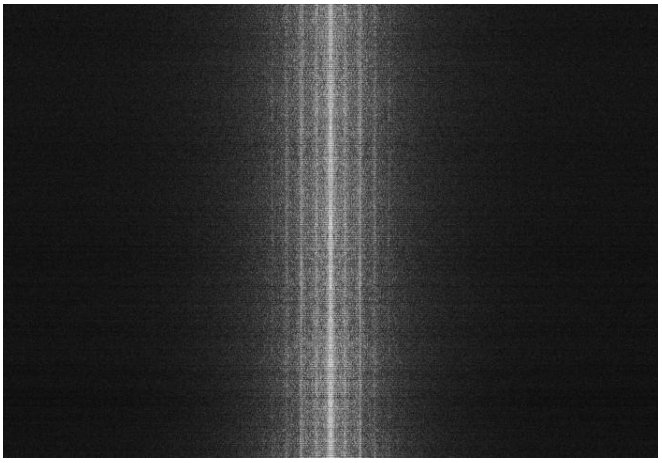
Example 1

Screenshot:



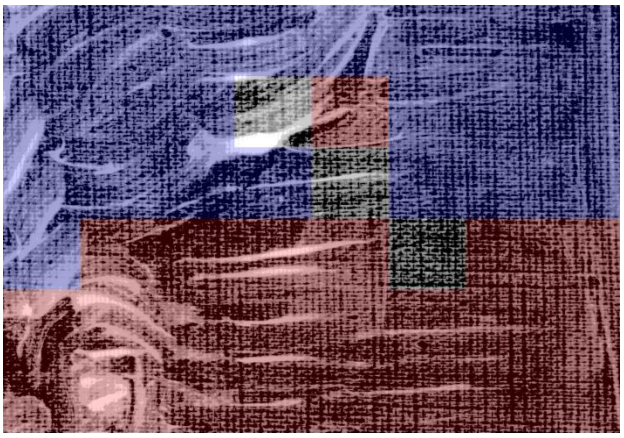
Starting Image:

Fourier Transform Image:



Inverse Fourier Transform Image:

Overlay Image:

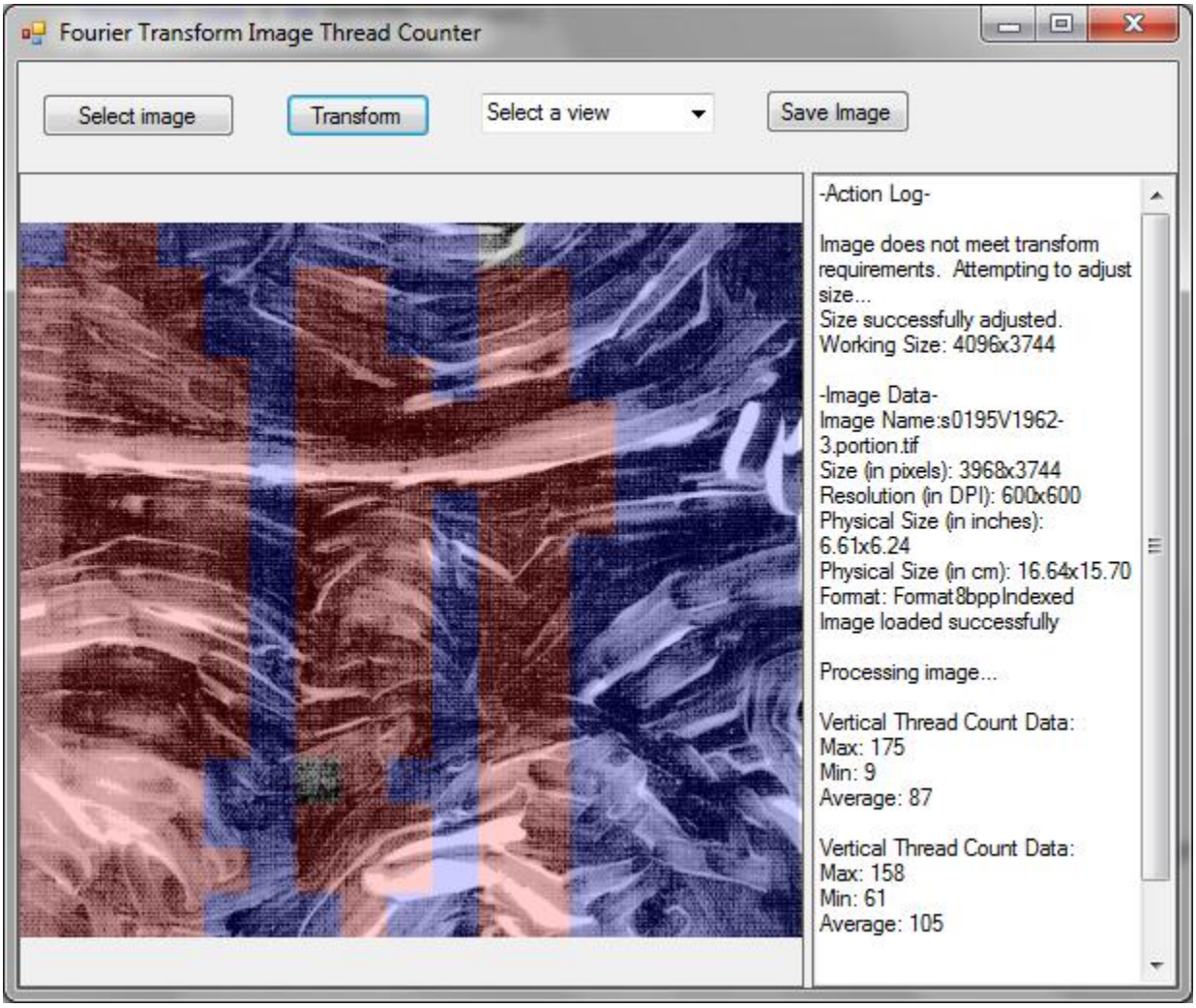


```
-Action Log-

Image does not meet transform requirements.  Attempting to adjust size...
Size successfully adjusted.
Working Size: 2048x1414

-Image Data-
Image Name:s0049V1962-4.portion.tif
Size (in pixels): 1909x1414
Resolution (in DPI): 600x600
Physical Size (in inches): 3.18x2.36
Physical Size (in cm): 8.01x5.93
Format: Format8bppIndexed
Image loaded successfully

Processing image...

Vertical Thread Count Data:
Max: 71
Min: 22
Average: 50

Vertical Thread Count Data:
Max: 94
Min: 22
```

```
Average: 63

Image successfully processed
```

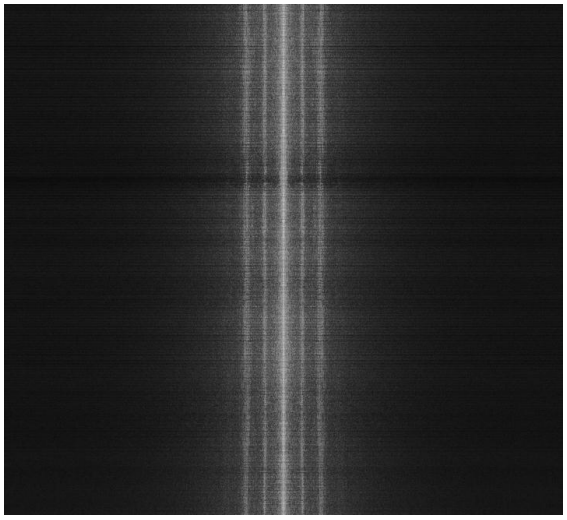
Example 2

Screenshot:



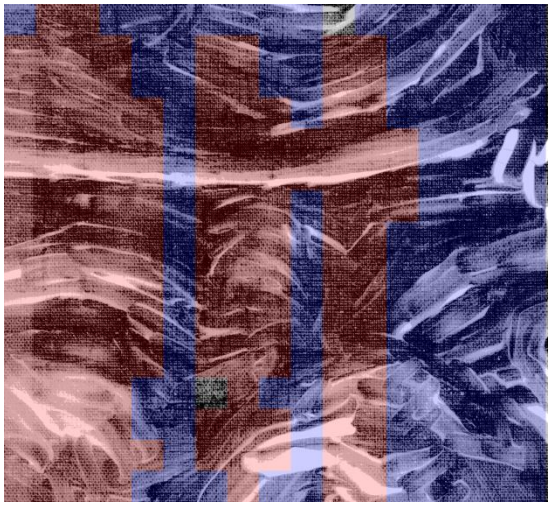
Starting Image:

Fourier Transform Image:



Inverse Fourier Transform Image:

Overlay Image:



```
-Action Log-

Image does not meet transform requirements.  Attempting to adjust size...
Size successfully adjusted.
Working Size: 4096x3744

-Image Data-
Image Name:s0195V1962-3.portion.tif
Size (in pixels): 3968x3744
Resolution (in DPI): 600x600
Physical Size (in inches): 6.61x6.24
Physical Size (in cm): 16.64x15.70
Format: Format8bppIndexed
Image loaded successfully

Processing image...

Vertical Thread Count Data:
Max: 175
Min: 9
```

```
Average: 87

Vertical Thread Count Data:
Max: 158
Min: 61
Average: 105

Image successfully processed
```

References

AForge Usage. (n.d.). Retrieved October 18, 2010, from http://www.codeproject.com/KB/GDI-plus/Image_Processing_Lab.aspx

AForge.NET Framework. (n.d.). Retrieved October 18, 2010, from http://www.aforgenet.com/framework/

Image Transforms - Fourier Transforms. (n.d.). Retrieved October 18, 2010, from http://homepages.inf.ed.ac.uk/rbf/HIPR2/fourier.htm