Comparison of Noise Filtering Algorithms

By: Michael Lang
February 6 2015

Introduction

Noise occurs in images taken on both digital and conventional cameras and can be caused by a number of reasons. Noise then must be removed either for artistic purposes or for technical purposes such as computer vision. There are multiple types of noise the most common being salt-and-pepper (impulse) noise and gaussian noise. The main tradeoffs of noise reducing algorithms is that they tend blur the image and obscure details such as the edges of lines however some algorithms are more susceptible to to those issues than others.

For the purpose of this experiment I have implemented 3 different noise filtering algorithms in Python; Median filter, Alpha-trimmed mean filter and the kuwahara filter.

Implementation

Median Filter:

The median filter is a type of non-linear digital filter technique most often used to remove noise from images. The median filter is a widely used algorithm in image processing since under the correct conditions it removes noise from the image while still preserving more of the edges than most filters. Median filter works by running through each pixel in the image and then replacing its value with the median of its neighbouring pixels. The neighbourhood containing the pixels around the current pixel being evaluated is known as the window.

Pseudo Code:

```
For each pixel p in image
    i = 0
    window = new Array[filter_size^2]
    for each pixel wp in the neighbourhood of p
        window[i++] = wp
    window.sort()
    p = window[i/2]
```

Alpha-Trimmed Mean Filter:

The Alpha-trimmed mean filter is a non-linear filter similar to the median filter. It also uses the window method for finding the neighbourhood around the current pixel. It works by sliding a window over every pixel to get its neighbourhood, and then sorts all the window pixels based on their value. The most extreme values (the minimum and maximum) are removed from the calculations based on the value of alpha. The filter then takes the mean of the remaining values and uses that for the value of the pixel.

Pseudo Code:

```
For each pixel p in image
        i = 0
        window = new Array[filter_size^2]
        for each pixel wp in the neighbourhood of p
                window[i++] = wp
        window.sort()
        window.trim(a)
        p = window.mean()
```

Kuwahara filter:

The kuwahara filter is another non-linear filter that uses the windowing method. The way the filter works is that it divides the window around the current pixel into 4 sub-windows. The edge values of each sub-window overlap. The filter then calculates the standard deviation of each of the sub-windows. The pixel in question then takes the value of the mean of the sub-window that has the lowest standard deviation. This allows the filter to find the area that is most homogenous. The algorithm is known for its ability to reduce noise in images, but also retain the edges in the image.

Pseudo Code:

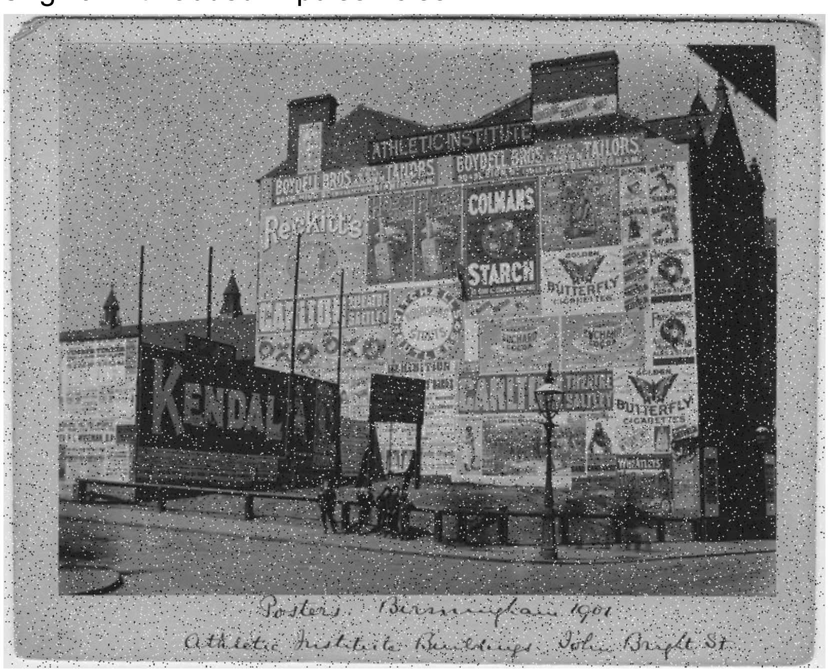
```
For each pixel p in image
        A = window.topleft()
        B = window.topright()
        C = window.bottomleft()
        D = window.bottomright()
        minMatrix = min( A.std(), B.std(), C.std(), D.std())
        p = minMatrix.mean()
```

Results

Image 1: 800 x 637 pixels

Original Image:

Original with added impulse noise:



Median filter applied (5x5 window):

Alpha-trimmed mean filter applied (5x5 window, alpha = 2):

Kuwahara filter applied (5x5 window):

Comparison of edges:

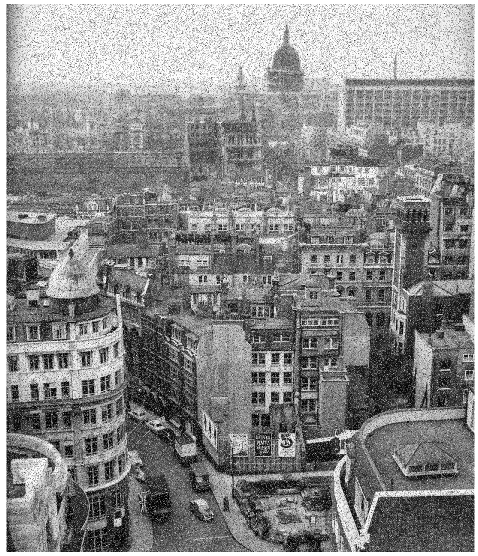
Kuwahara filter:

Median filter:

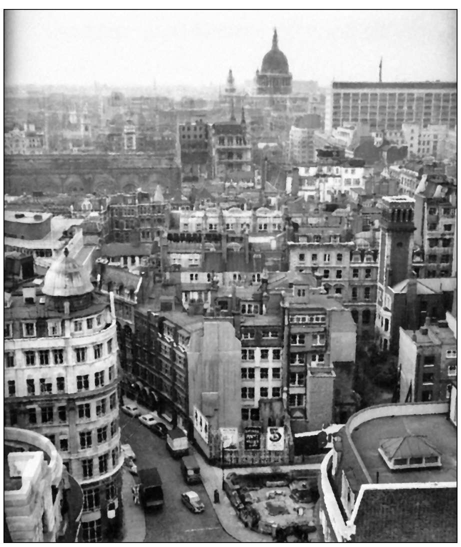
Image 2: 1091 X 1280 pixels

Original:

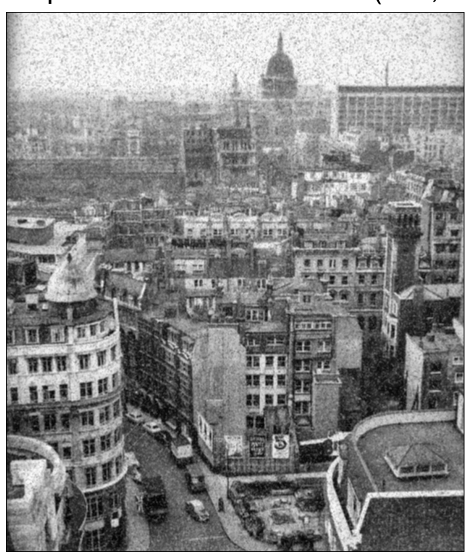
Impulse noise added:



Median filter (5x5):



Alpha-trimmed Mean filter (5x5, a = 2):



Kuwahara filter (5x5):

Analysis:

Image 1: 800x637 pixels

Filter execution time: Median: 7.1s
Alpha-trimmed mean: 14.2s
Kuwahara: 182.3s

Image 1 is a 800 x 637 image with Impulse noise added. When the 5x5 median filter is applied almost all of the noise disappears and the image appears slightly blurred. An almost identical result is returned from the 5x5 alpha-trimmed mean filter except there is still some visible noise in the result. The kuwahara filter gives us a different result. It blurs out the details of the image creating an artistic blocky effect however the edges are still well defined. In the close up of the edges you can see the edges from the median filter are somewhat muddy while the kuwahara edges are distinct.

Image 2: 1091x1280 pixels

Filter execution time: Median: 16.9s
Alpha-trimmed mean: 22.3s
Kuwahara: 290.1s

Image 2 has quite a bit longer execution time than image 1 due to the size of the image. The amount of noise in image 2 is also a lot higher than in image 1. In this experiment only the median filter was able to remove close to all of the noise, while the alpha-trimmed mean filter failed completely. The reason for this is that the mean filter is more sensitive to noise due to the fact that each pixel, every pixel in its neighbourhood influences its value. Alpha trimming removes some of the outliers but when there is too many noise pixels in the neighbourhood some don't get cut out and they end up influencing the value of the pixel. Median filter works well here because the extremeness of the values of the outliers do not affect the calculations for the pixel, so long as the neighbourhood is not almost completely comprised of noise. The kuwahara filter was able to remove more noise than the alpha-trimmed mean filter but less than the median, the reason being that the kuwahara filter is also based on the mean so it is more sensitive to noise than the median filter.

References

1. "Median Filter." *Spatial Filters*. N.p., n.d. Web. 06 Feb. 2015. <http://homepages.inf.ed.ac.uk/rbf/HIPR2/median.htm>.
2. "Kuwahara Filter." *Wikipedia*. Wikimedia Foundation, n.d. Web. 06 Feb. 2015. <http://en.wikipedia.org/wiki/Kuwahara_filter>.
3. Chernenko, Sergey. "Alpha-trimmed Mean Filter." — *Librow — Software*. Librow, n.d. Web. 06 Feb. 2015. <http://www.librow.com/articles/article-7>.
4. "Noise Reduction." *Wikipedia*. Wikimedia Foundation, n.d. Web. 06 Feb. 2015. <http://en.wikipedia.org/wiki/Noise_reduction>.