CHAPTER 3

REMOVAL OF NOISE USING WAVELET BASED FUZZY FILTER

3.1 PREPROCESSING

Noise removal is an important step for an image retrieval system to remove the unwanted information present in the image using filtering techniques for web based applications. Web images are often degraded by additive noise. The goal of smoothing the image is to remove the noise while retaining the image features such as color, texture, shape and so on. The denoising technique yields a better quality image. Denoising can be done through filtering which can either be linear filtering or non-linear filtering. Linear filters are applied by convolution with a low-pass filter convolution kernel. These do not eliminate additive noise as they have a tendency to blur the edges of an image. On the other hand, nonlinear filter is suitable for dealing with additive noise. It is also called as ordered filter, because it cannot be applied as a linear operator or convolution kernel. These filters operate on small size windows, and replace the value of the central pixel. Compared to other nonlinear techniques, wavelet based fuzzy filters have the ability to combine edge preservation and smoothing. In this chapter wavelet based fuzzy filter is used to filter the images and the result proved better in terms of MSE and PSNR when compared to linear, median and wiener filter.
3.1.1 Linear Smoothing Filter

Linear smoothing filters are good filters for removing Gaussian noise and in most cases the other types of noise as well. A linear filter is implemented using the weighed sum of the pixels in successive windows. Typically same pattern of weights is used in each window which means that the linear filter is spatially invariant and can be implemented using a convolution mask (Mitra Basu 2002).

3.1.2 Wiener Filter

Wiener filter is used to filter out noise that has corrupted an image. Wiener filter uses a local variance field of the distorted image that is based on statistical properties of the original image. If the variance is large, wiener performs little smoothing. If the variance is small, wiener performs more smoothing. This approach often produces better results than linear filtering (Dabov et al 2007). In addition, wiener filter requires more computation time than linear filtering. Wiener works best when the noise is constant-power ("white") additive noise, such as Gaussian noise.

3.1.3 Median Filter

A median filter is an example of a non-linear filter. The steps are followed as:

Step 1: Consider each pixel in the image
Step 2: Sort the neighboring pixels in order, based upon their intensities
Step 3: Replace the original value of the pixel with the median value from the list.
Mean filtering is similar to averaging filter, in which each output pixel is set to an average of the pixel values in the neighborhood of the corresponding input pixel. However, with median filtering, the value of an output pixel is determined by the median of the neighborhood pixels, rather than the mean. This median is much less sensitive than the mean of the extreme values (called outliers). Median filtering is therefore able to remove these outliers better than the mean filtering without reducing the sharpness of the image (Chen et al 1999).

3.1.4 Wavelet Based Fuzzy Filter

Wavelet based fuzzy filters provide promising result in image-processing tasks that cope with some drawbacks of other filters. It is capable of dealing with vague and uncertain information. Sometimes, it is required to recover a heavily noise corrupted image, where a lot of uncertainties are present. Wavelet based fuzzy filters perform both the edge preservation and smoothing effectively (Nejad et al 2006). The block diagram of wavelet based fuzzy filter is shown in Figure 3.1.

![Figure 3.1 Block diagram of wavelet based fuzzy filter](image)

A wavelet based fuzzy filter is used for the reduction of additive noise for digital color images. The filter consists of two stages namely wavelet filter and second fuzzy filter. In the first stage wavelet is used to distinguish between local variations. The second stage fuzzy filter is used to
enhance the first method by reducing the noise in the color components
differences without destroying the fine details of the image. This is realized
by calculating the local differences in the red, green and blue environment
separately. These differences are then combined to calculate the local
estimation of the central pixel (Schulte et al 2006).

3.1.4.1 Gaussian noise removal

RGB color model is represented by a 3 dimensional vector. Red,
Green and Blue are called as the primary components. Each element is
quantized to the range 0 to $2^m-1$, where $m$ is 8. A digital color image $C$ is
represented by a 2 dimensional array of vectors.

Images are corrupted by an additive Gaussian white noise following
a normal law defined by a zero mean and a known $\sigma^2$ variance, that is
$n \sim N(0, \sigma^2)$ and is expressed in Equation (3.1).

$$
N_{s,d}(I_{j,1}) N_{s,d} (I_{j,2}) N_{s,d} (I_{j,3}) = \left[ (C_{s,d}(I_{j,1})+\eta_1) (C_{s,d} (I_{j,2})+\eta_2)\right.
\left. (C_{s,d} (i,j,3)-\eta_3) \right]
$$

(3.1)

Where $C_{s,d}(i,j,1)$ - red component

$C_{s,d}(i,j,2)$ - green component

$C_{s,d}(i,j,3)$ - blue component

$N_{s,d}$ - Noisy wavelet coefficients of scale $s$ and orientation $d$ respectively.

$\eta_1, \eta_2, \eta_3$ - three separate randomly distributed Gaussian values
with means ($\mu_1, \mu_2$ and $\mu_3$) and standard deviations ($\sigma_1, \sigma_2$ and $\sigma_3$) respectively.
3.1.4.2 Wavelet filter

The steps for the wavelet filters are as follows:

**Step 1:** Choose window size of $(2K+1)^2$ for the current image pixel at position $(i,j)$.

**Step 2:** Obtain the feature using a nonlinear averaging filter in the wavelet sub bands of each single channel.

**Step 3:** Next, assign large weights to neighboring coefficients with similar magnitude and vice versa. The weights for the red, green and blue component at position $(i+k,j+l)$ are $w(i+k,j+l,1)$, $w(i+k,j+l,2)$ and $w(i+k,j+l,3)$ respectively.

**Step 4:** Calculate the fuzzy function of magnitude similarity and a fuzzy function of spatial similarity for the red, green and blue component which is defined in Equation (3.2) - (3.7).

\[
m(i + k, j + l, 1) = \exp \left[ - \left( \frac{N_{\text{diff}}(i+k,1)-N_{\text{diff}}(i+k+1,1)}{Thr} \right)^2 \right] \tag{3.2}
\]

\[
m(i + k, j + l, 2) = \exp \left[ - \left( \frac{N_{\text{diff}}(i+k,2)-N_{\text{diff}}(i+k+1,2)}{Thr} \right)^2 \right] \tag{3.3}
\]

\[
m(i + k, j + l, 3) = \exp \left[ - \left( \frac{N_{\text{diff}}(i+k,3)-N_{\text{diff}}(i+k+1,3)}{Thr} \right)^2 \right] \tag{3.4}
\]

\[
s(i + k, j + l, 1) = \exp \left[ - \left( \frac{(i+k)^2+(j+l)^2}{N} \right)^2 \right] \tag{3.5}
\]

\[
s(i + k, j + l, 2) = \exp \left[ - \left( \frac{(i+k)^2+(j+l)^2}{N} \right)^2 \right] \tag{3.6}
\]

\[
s(i + k, j + l, 3) = \exp \left[ - \left( \frac{(i+k)^2+(j+l)^2}{N} \right)^2 \right] \tag{3.7}
\]
Where $\text{Thr}=K \cdot \sigma_{n_c}$; $3 \leq K \leq 4$; $\sigma_{n_c}$ is estimate noise variance of channel c using median estimator and N is the number of coefficients in the local window; $k \in [-K \ldots K]$; $l \in [-L \ldots L]$.

According the three fuzzy functions, assign adaptive weight $w(i + k, j + l, 1)$ for each neighboring coefficient for red, green and blue component are expressed in Equations (3.8)-(3.10).

\[
w(i + k, j + l, 1) = m(i + k, j + l, 1) \ast s(i + k, j + l, 1) \tag{3.8}
\]

\[
w(i + k, j + l, 2) = m(i + k, j + l, 2) \ast s(i + k, j + l, 2) \tag{3.9}
\]

\[
w(i + k, j + l, 3) = m(i + k, j + l, 3) \ast s(i + k, j + l, 3) \tag{3.10}
\]

**Step 5:** Find the output image of the wavelet filter for the red, green and blue component. It is shown in Equations (3.11-3.13).

\[
F(i,j,1)=\frac{\sum_{k=-K}^{+K} \sum_{l=-L}^{+L} w(i+k,j+l,1) N(i+k,j+l,1)}{\sum_{k=-K}^{+K} \sum_{l=-L}^{+L} w(i+k,j+l,1)} \tag{3.11}
\]

\[
F(i,j,2)=\frac{\sum_{k=-K}^{+K} \sum_{l=-L}^{+L} w(i+k,j+l,2) N(i+k,j+l,2)}{\sum_{k=-K}^{+K} \sum_{l=-L}^{+L} w(i+k,j+l,2)} \tag{3.12}
\]

\[
F(i,j,3)=\frac{\sum_{k=-K}^{+K} \sum_{l=-L}^{+L} w(i+k,j+l,3) N(i+k,j+l,3)}{\sum_{k=-K}^{+K} \sum_{l=-L}^{+L} w(i+k,j+l,3)} \tag{3.13}
\]

**3.1.4.3 Fuzzy filter**

The process for the fuzzy filter is shown below:

**Step 1:** Calculate the gradients or derivatives for the red ($\text{LD}_R$) green ($\text{LD}_G$) and blue ($\text{LD}_B$) for each element of the window. It is shown in Equations (3.14-3.16).
\[ LD_R(k,l) = F(i+k, j+l, 1) - F(i,j,1) \]  
\[ LD_G(k,l) = F(i+k, j+l, 2) - F(i,j,2) \]  
\[ LD_B(k,l) = F(i+k, j+l, 3) - F(i,j,3) \]  

where \( k \in \{-K, \ldots, 0, \ldots, +K\} \) and \( l \in \{-L, \ldots, 0, \ldots, +L\} \)

**Step 2:** Calculate the correction terms \( c(k,l) \) for a 3x3 window (\( L = 1 \)). It is given by Equation (3.17) and shown in figure 3.2.

\[ c(k,l) = \frac{1}{3} (LD_R(k,l) + LD_G(k,l) + LD_B(k,l)) \]  

**Step 3:** Finally find the output of the fuzzy filter for red, green and blue components. It is shown in Equations (3.18-3.20).

\[ \text{Out} (i,j,1) = \sum_{k=-K}^{K} \sum_{l=-L}^{L} (F(i+k, j+l, 1) + c(k,l)) \frac{1}{2L+1} \]  
\[ \text{Out} (i,j,2) = \sum_{k=-K}^{K} \sum_{l=-L}^{L} (F(i+k, j+l, 2) + c(k,l)) \frac{1}{2L+1} \]  
\[ \text{Out} (i,j,3) = \sum_{k=-K}^{K} \sum_{l=-L}^{L} (F(i+k, j+l, 3) + c(k,l)) \frac{1}{2L+1} \]  

<table>
<thead>
<tr>
<th>j-1</th>
<th>j</th>
<th>j+1</th>
</tr>
</thead>
<tbody>
<tr>
<td>i-1</td>
<td>r_4</td>
<td>r_3</td>
</tr>
<tr>
<td>i</td>
<td>r_1</td>
<td>r_0</td>
</tr>
<tr>
<td>i+1</td>
<td>r_2</td>
<td>r_3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>j-1</th>
<th>j</th>
<th>j+1</th>
</tr>
</thead>
<tbody>
<tr>
<td>i-1</td>
<td>g_4</td>
<td>g_3</td>
</tr>
<tr>
<td>i</td>
<td>g_1</td>
<td>g_0</td>
</tr>
<tr>
<td>i+1</td>
<td>g_2</td>
<td>g_3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>j-1</th>
<th>j</th>
<th>j+1</th>
</tr>
</thead>
<tbody>
<tr>
<td>i-1</td>
<td>r_{4-} r_0</td>
<td>r_{3-} r_0</td>
</tr>
<tr>
<td>i</td>
<td>r_{1-} r_0</td>
<td>0</td>
</tr>
<tr>
<td>i+1</td>
<td>r_{2-} r_0</td>
<td>r_{3-} r_0</td>
</tr>
</tbody>
</table>

W_R  
W_B  
LD_R from W_R
<table>
<thead>
<tr>
<th></th>
<th>j-1</th>
<th>j</th>
<th>j+1</th>
</tr>
</thead>
<tbody>
<tr>
<td>i-1</td>
<td>g_{j-1} - g_0</td>
<td>g_{j-1} - g_0</td>
<td>g_{j-1} - g_0</td>
</tr>
<tr>
<td>i</td>
<td>g_{j} - g_0</td>
<td>0</td>
<td>g_{j} - g_0</td>
</tr>
<tr>
<td>i+1</td>
<td>g_{j+1} - g_0</td>
<td>g_{j+1} - g_0</td>
<td>g_{j+1} - g_0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>j-1</th>
<th>j</th>
<th>j+1</th>
</tr>
</thead>
<tbody>
<tr>
<td>i-1</td>
<td>b_{j-1} - b_0</td>
<td>b_{j-1} - b_0</td>
<td>b_{j-1} - b_0</td>
</tr>
<tr>
<td>i</td>
<td>b_j - b_0</td>
<td>0</td>
<td>b_j - b_0</td>
</tr>
<tr>
<td>i+1</td>
<td>b_{j+1} - b_0</td>
<td>b_{j+1} - b_0</td>
<td>b_{j+1} - b_0</td>
</tr>
</tbody>
</table>

The results of proposed and other filtering techniques are shown in Figure 3.3 and 3.4.

**Figure 3.2** Illustrating scheme for the calculation of the correction terms for a 3x3 window

**Figure 3.3** Continued
Figure 3.3 Results of applying different filtering techniques

(a) Sample images of ‘Bird 1’, ‘Tiger’, ‘Bear’ and ‘Elephant’ (b) using linear filter (c) using wiener filter (d) using median filter (e) using wavelet based fuzzy filter
Figure 3.4  Continued
3.2 PERFORMANCE EVALUATION

The proposed method has been used to remove the additive noise from the image using wavelet based fuzzy filter in color image processing. This filtering technique was applied to 1200 images and the output was compared with different filtering technique. The Mean Square Error (MSE) and Peak Signal to Noise Ratio (PSNR) are the two metrics used to compare the image quality.

Mean Square Error (MSE)

It represents the cumulative squared error. MSE value is the difference between noiseless images and the noisy image. The lower the value of MSE, the error is minimum. The MSE is calculated by Equation (3.21).

\[
MSE = \frac{\sum_{m=1}^{M} \sum_{n=1}^{N} (I_1(m,n)-I_2(m,n))^2}{(M-N)}
\]  
(3.21)
where, $I_1$ is the noiseless image, $I_2$ is the noisy image, $M$ and $N$ are the number of rows and columns in the input image respectively.

**Peak Signal to Noise Ratio (PSNR)**

It is used as a quality measurement between the noiseless and noisy image. The higher the PSNR, better the quality of the segmented image. It is calculated by using Equation (3.22).

$$
\text{PSNR} = 10 \log_{10} \left( \frac{R^2}{\text{MSE}} \right)
$$

Where $R$ is the maximum fluctuation in the input image

The evaluation of this method is given in Table 3.1, 3.2 and 3.3.

**Table 3.1 Performance evaluation of an images based on MSE values**

<table>
<thead>
<tr>
<th>Sample images</th>
<th>MSE(Mean Square Error) values</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Linear smoothing filter</td>
</tr>
<tr>
<td>Bird 1</td>
<td>133.92</td>
</tr>
<tr>
<td>Tiger</td>
<td>1198.90</td>
</tr>
<tr>
<td>Bear</td>
<td>1448.77</td>
</tr>
<tr>
<td>Elephant</td>
<td>616.92</td>
</tr>
<tr>
<td>Bird 2</td>
<td>624.84</td>
</tr>
<tr>
<td>Donkey</td>
<td>719.8</td>
</tr>
<tr>
<td>Deer</td>
<td>1012.43</td>
</tr>
<tr>
<td>Frog</td>
<td>653.96</td>
</tr>
</tbody>
</table>
Table 3.2 Performance evaluation of an images based on PSNR values

<table>
<thead>
<tr>
<th>Sample images</th>
<th>Linear smoothing filter</th>
<th>Wiener filter</th>
<th>Median filter</th>
<th>Wavelet based Fuzzy filter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bird 1</td>
<td>26.86</td>
<td>25.66</td>
<td>35.87</td>
<td>42.53</td>
</tr>
<tr>
<td>Tiger</td>
<td>17.34</td>
<td>21.59</td>
<td>25.28</td>
<td>35.47</td>
</tr>
<tr>
<td>Bear</td>
<td>16.52</td>
<td>22.13</td>
<td>25.42</td>
<td>41.25</td>
</tr>
<tr>
<td>Elephant</td>
<td>20.22</td>
<td>23.70</td>
<td>30.01</td>
<td>38.28</td>
</tr>
<tr>
<td>Bird 2</td>
<td>20.17</td>
<td>23.84</td>
<td>28.53</td>
<td>41.10</td>
</tr>
<tr>
<td>Donkey</td>
<td>19.56</td>
<td>23.21</td>
<td>29.08</td>
<td>42.58</td>
</tr>
<tr>
<td>Deer</td>
<td>18.07</td>
<td>23.15</td>
<td>27.85</td>
<td>41.66</td>
</tr>
<tr>
<td>Frog</td>
<td>19.97</td>
<td>23.47</td>
<td>30.47</td>
<td>40.37</td>
</tr>
</tbody>
</table>

Table 3.3 Comparative analyses of various Image Enhancement methods based on average PSNR and average MSE values

<table>
<thead>
<tr>
<th>Enhancement Technique</th>
<th>Average PSNR value</th>
<th>Average MSE value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear smoothing filter</td>
<td>19.84</td>
<td>801.19</td>
</tr>
<tr>
<td>Wiener filter</td>
<td>23.34</td>
<td>310.82</td>
</tr>
<tr>
<td>Median filter</td>
<td>29.06</td>
<td>99.71</td>
</tr>
<tr>
<td>Wavelet based fuzzy filter</td>
<td>40.41</td>
<td>6.96</td>
</tr>
</tbody>
</table>
Figure 3.5  Graphical representation of average MSE values for wavelet based fuzzy filter with existing filters

Figure 3.6  Graphical representation of average PSNR values for wavelet based fuzzy filter with existing filters
Graphical representation of MSE and PSNR for wavelet based fuzzy filter with existing filters is shown in Figure 3.5 and 3.6. The results obtained using wavelet based fuzzy filter technique ensures noise free and increase in quality of the image. Hence this method is more suitable than other filters available at present to remove noises and to enhance the image quality.

3.3 SUMMARY

In this chapter wavelet based fuzzy filtering technique for removing additive noise in color image is discussed. The result for the wavelet based fuzzy filter is compared with other filtering techniques such as linear smoothing filter, wiener filter and median filter. The performance results obtained using wavelet based fuzzy filtering technique has proven better results in terms of MSE and PSNR values. Color image segmentation technique is discussed in chapter 4.