KHWOPA ENGINEERING COLLEGE

COURSE CODE :BEG 475 IP

IMAGE PROCESSING AND PATTERN RECOGNTIION

Lab Report on
Bit Plane Slicing and Noise Filters

Author:
Rabi Raj Khadka

Roll Number:
700324

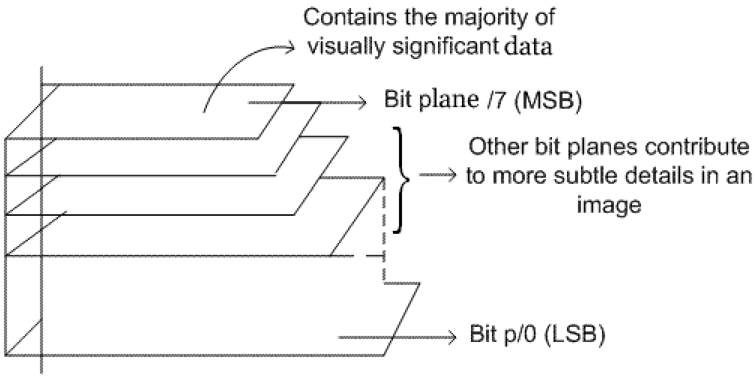
July 4, 2017

Contents

1 Theory

1.1 Bit Plane Slicing

The Bit-plane slicing highlights the contribution mode to total image appearance by specific bits.For the bit plane slicing, each pixel is represented by 8 bits. Since the image is composed of 8 bits, it contains 8 planes. Higher order bits contains the majority of the visual significance data whereas other bit planes contributes more suitable details of image. It is useful for analyzing the importance played by each bit of an image. The binary image for the bit plane 7 can be obtained by processing the input image with thresholding gray level transformations.



1.2 Noise and Noise Filters

Digital images are prone to a variety of types of noise. Noise is the result of errors in the image acquisition process that result in pixel values that do not reflect the true intensities of the real scene.In other words, noise in image, is any degradation in an image signal. There are several ways that noise can be introduced into an image, depending on how the image is created. For example:

- If the image is scanned from a photograph made on film, the film grain is a source of noise. Noise can also be the result of damage to the film, or be introduced by the scanner itself.
- If the image is acquired directly in a digital format, the mechanism for gathering the data (such as a CCD detector) can introduce noise.
- Electronic transmission of image data can introduce noise.

To simulate the effects of some of the problems listed above, the MATLAB toolbox provides the imnoise function, which we can use to add various types of noise to an image.

Image noise comes in many flavors, and as a consequence the appropriate model should be employed for robust image restoration. Noise may be additive, which can be expressed as

$$g(x, y) = f(x, y) + n(x, y)$$

where f is the original 2D signal (image), n is the noise contribution, and g is the corrupted image. Image noise can also be multiplicative, where

$$g(x, y) = f(x, y)n(x, y)$$

`Types of Noise`

▷ Salt and Pepper Noise
▷ Gaussian Noise
▷ Speckle Noise
▷ Uniform Noise

`Filtering Techniques`

Filtering Image data is standard process used in almost all image processing systems. Filters are used to remove noise from digital image while keeping the details of image preserved.The choice of filter is determined by

→ the nature of the task performed by filter
→ filter behavior
→ type of the data

Different Filtering techniques are linear and non-linear filters as below

⊡ Mean Filter

It is simple linear filter which replaces each pixel value in an image with the mean value of its neighbors, including itself and used to remove the impulse noise.

⊡ Gaussian Filter

Gaussian Filter is smoothing filter in the 2D convolution operation that is used to remove noise and blur from image. It is done by convolution each point in the input array with a gaussian kernel and then summing them all to produce the output array.

⊡ Median Filter

It is simple and powerful non-linear filter primarily used for reducing the amount of intensity variation between one pixel to other pixel. Pixel value are replaced with the median values.

⊡ Wiener Filter

Wiener Filter is based on statistical approach whuch is uded to filter out the noise that has the corrupted a signal and focuses on reducing mean square error.

1.3 Algorithm for Bit Plane Slicing

1. Start
2. Read the image
3. Convert the image into grayscale and then to double (x)
4. Convert the double image to binary image of 8 bit planes
5. Read the bit 0 plane using mod 2 on the double image i.e c0 = mod(x,2)
6. For each of the next bit plane, take the mod of floor value of half of x and 2. i.e.
x = floor(x/2) and mod(x,2)
7. The original image can be retrieved by
unit8((2*(2*(2*(2*(2*(2*(c7*2+c6)+c5)+c4)+c3)+c2)+c1))+c0)
8. End

1.4 Algorithm for Noise Filtering

1. Start
2. Read image and convert it into gray scale
3. Construct mean filter mask and apply it to the gray scale image
- meanf =[1 1 1;1 1 1;1 1 1]/9;
- result_one = imfilter(gray_img,meanf);
4. Construct weighted filter and apply it to gray scale image
- weightf=[1 2 1;2 4 2;1 2 1]/16;
- result_two = imfilter(gray_img,weightf);
5. Add different noises to the gray scale image
- a) Salt and pepper noise
– sp = imnoise(gray_img,'salt and pepper',0.1);
- b) Gaussian Noise
– g = imnoise(gray,'gaussian',0.1);
6. Remove the noises by applying the mean filter and weighted filter
7. Show all the filter imaged and noise image.
8. Plot the surface of the mean filter, median filter and gaussian filter as:
- a) Surface plot of gaussian filter
– surf(1:2*cutoff+1,1:2*cutoff+1,gaussf);title('Surface plot of gaussian Filter');
- b) Surface plot of mean filter
– freqz2(meanf); title('Mean filter response');
- c) Surface plot of meadian filter
– mf = medfilt2(g,[3,3]);
– figure; freqz2(mf); title('Median filter response');
9. Stop

2 Code Description

Program: Bit Plane Slicing

```
myimage = imread('\img\potrait.jpg');
gray_myimage = rgb2gray(myimage);
matrix_myimage=double(gray_myimage);
c0=mod(matrix_myimage,2); subplot(3,3,2); imshow(c0); title('Bit 0 Plane');
c1=mod(floor(matrix_myimage/2),2); subplot(3,3,3);imshow(c1);title('Bit 1 Plane');
c2=mod(floor(matrix_myimage/4),2);subplot(3,3,4);imshow(c2);title('Bit 2 Plane');
c3=mod(floor(matrix_myimage/8),2);subplot(3,3,5);imshow(c3);title('Bit 3 Plane');
c4=mod(floor(matrix_myimage/16),2);subplot(3,3,6);imshow(c4);title('Bit 4 Plane');
c5=mod(floor(matrix_myimage/32),2);subplot(3,3,7);imshow(c5);title('Bit 5 Plane');
c6=mod(floor(matrix_myimage/64),2);subplot(3,3,8);imshow(c6);title('Bit 6 Plane');
c7=mod(floor(matrix_myimage/128),2);subplot(3,3,9);imshow(c7);title('Bit 7 Plane');
figure; subplot(3,3,1);imshow(gray_myimage); title('GrayScale(Original) Image');impixelinfo;
original=(2*(2*(2*(2*(2*(2*(c7*2+c6)+c5)+c4)+c3)+c2)+c1))+c0; original=uint8(original);
figure;subplot(1,2,1);imshow(gray_myimage);title('Original Image');impixelinfo;
subplot(1,2,2);imshow(original);title('Recovered Image');impixelinfo;
```

Program: Noise and Noise Filter

```
myimage = imread('img\potrait.jpg');
gray_myimage = rgb2gray(myimage);
matrix_myimage=double(gray_myimage);
figure;
subplot(1,3,1);imshow(gray_myimage);title('Original Image');
meanf =[1 1 1;1 1 1;1 1 1]/9;
meanf_result = imfilter(gray_myimage,meanf);
subplot(1,3,2);imshow(meanf_result);title('Mean Filter on Original Image'
weightf=[1 2 1;2 4 2;1 2 1]/16;
weightf_result = imfilter(gray_myimage,weightf);
subplot(1,3,3);imshow(weightf_result);title('Weighted Filter on Original
sandp = imnoise(gray_myimage,'salt & pepper',0.1);
figure;
subplot(1,3,1);imshow(sandp);title('Salt and Pepper Noise on Original Ima
sp_remove_meanf_result = imfilter(sandp,meanf);
subplot(1,3,2);imshow(sp_remove_meanf_result);title('S&P noise removed by
sp_remove_weightf_result = imfilter(sandp,weightf);
subplot(1,3,3);imshow(sp_remove_weightf_result);title('S&P noise removed
```

```
gaussian_image = imnoise(gray_myimage,'gaussian',0.1);
figure;
subplot(2,3,1),imshow(gaussian_image),title('Gaussinan Noise on Original 
gaussian_remove_meanf_result = imfilter(gaussian_image,meanf);
subplot(2,3,2),imshow(gaussian_remove_meanf_result),title('Gaussinan Nois
gaussian_remove_weightf_result = imfilter(gaussian_image,weightf);
subplot(2,3,4),imshow(gaussian_remove_weightf_result),title('Gaussinan N
sigma =3;
cutoff = ceil(3*sigma);
gaussianf = fspecial('gaussian', 2*cutoff+1, sigma);
gaussian_remove_gaussianf_result = imfilter(gaussian_image, gaussianf);
subplot(2,3,6),imshow(gaussian_remove_gaussianf_result),title('Gaussinan 
wienerf= wiener2(gray_myimage,[5,5]);
subplot(2,3,5),imshow(wienerf),title('Gaussian Noise removed by Wiener F
medianf = medfilt2(gaussian_image,[3,3]);
subplot(2,3,3),imshow(medianf),title('Gaussian Noise removed by Median F
figure;
subplot(1,2,1); freqz2(meanf); title('Mean Filter Response');
subplot(1,2,2);freqz2(medianf); title('Median Filter Response');
```

3 Result and Discussion

The color image is converted to gray scale image as gray scale image is composed of only 8 bits pixel. So, it would be easy to separate the 8 bits of planes of an image for slicing.
The bit planes of an image are separated by taking the remainder when divided by 2 after converting the gray scale image to binary and get back the original image by multiplying each bit plane values by 2.
Outputs

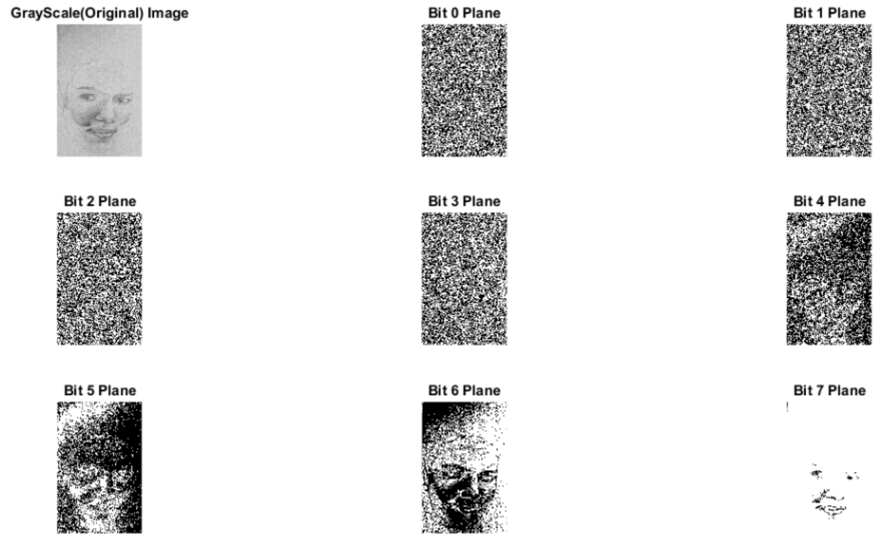


Figure 1: Bit Plane Sliced Images

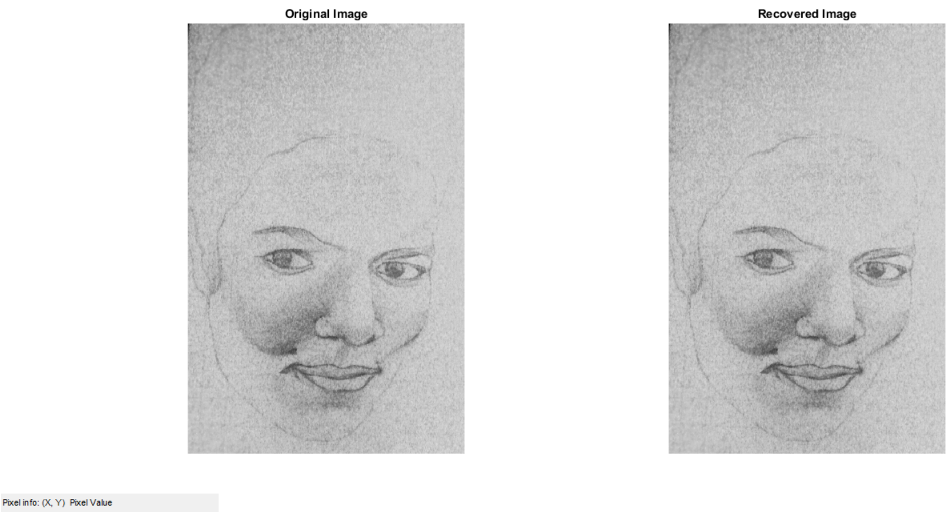


Figure 2: Original and Recovered Image from Slicing
For every image operation, the image must be converted to the gray scale image.

Then it will be easy for us to apply different filters into that mage as er our requirement.
The mean filter gives the image with blurred image. This can be overcome by the use of weighted filter.
The salt and pepper noise is added to the image to make it look old or make some dots in the image using the function

▷ imfilter(grayimage, 'salt and pepper', amount_of_ noise);

The noise is more reduced when we apply the weighted filter that that of the mean filter.
This applies same to the gaussian noise. The weighted filter gives clear result than that of using the built-in gaussian filter

▷ sigma =3;
▷ cutoff = ceil(3*sigma);
▷ gaussf = fspecial('gaussian', 2*cutoff+1, sigma);
▷ gresult2 = imfilter(g, gaussf);

The surface plot of different filters are obtained using the MATLAB function

▷ freqz2(meanf); title('Mean Filter response');
▷ mf = medfilt2(g,[3,3]);
▷ figure; freqz2(mf); title('Median Filter response');

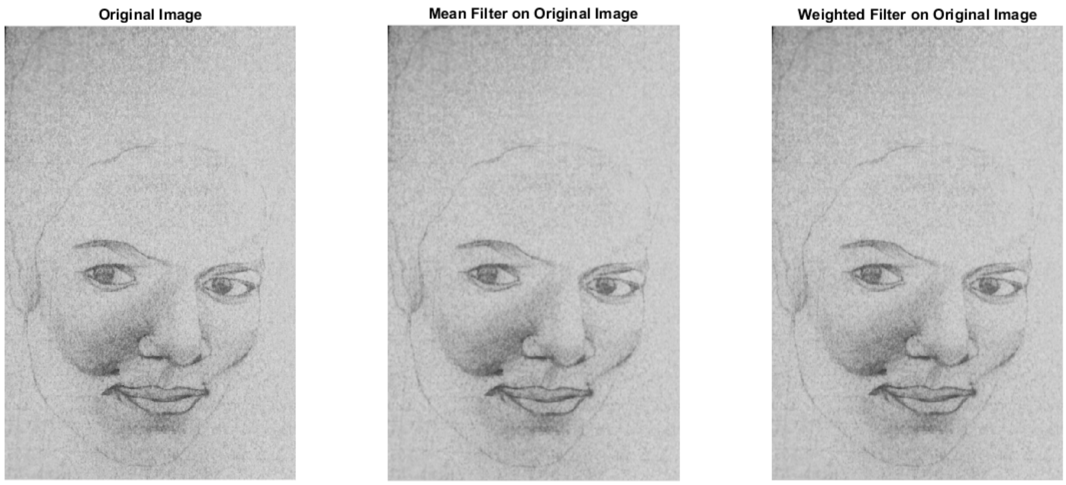


Figure 3: Mean and Weighted Filter

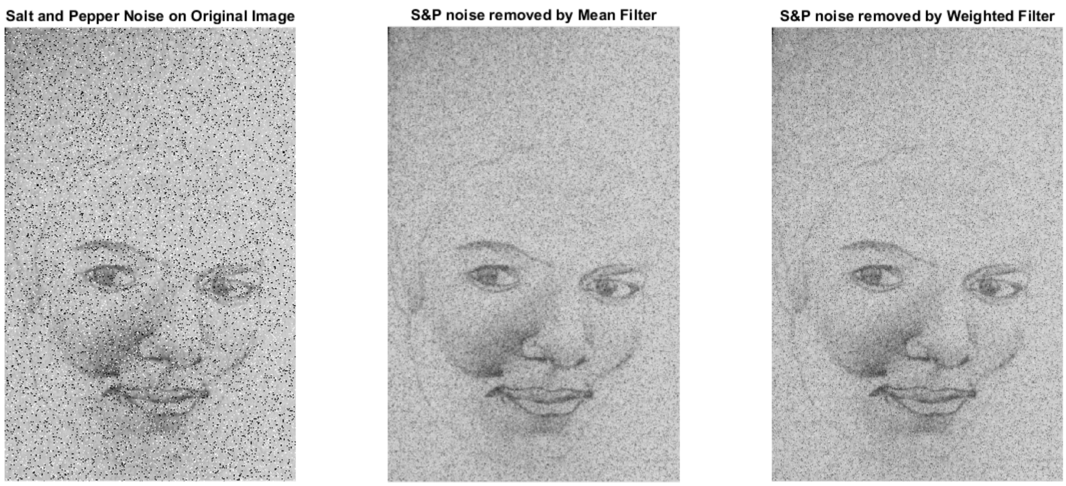


Figure 4: Salt and Pepper Noise

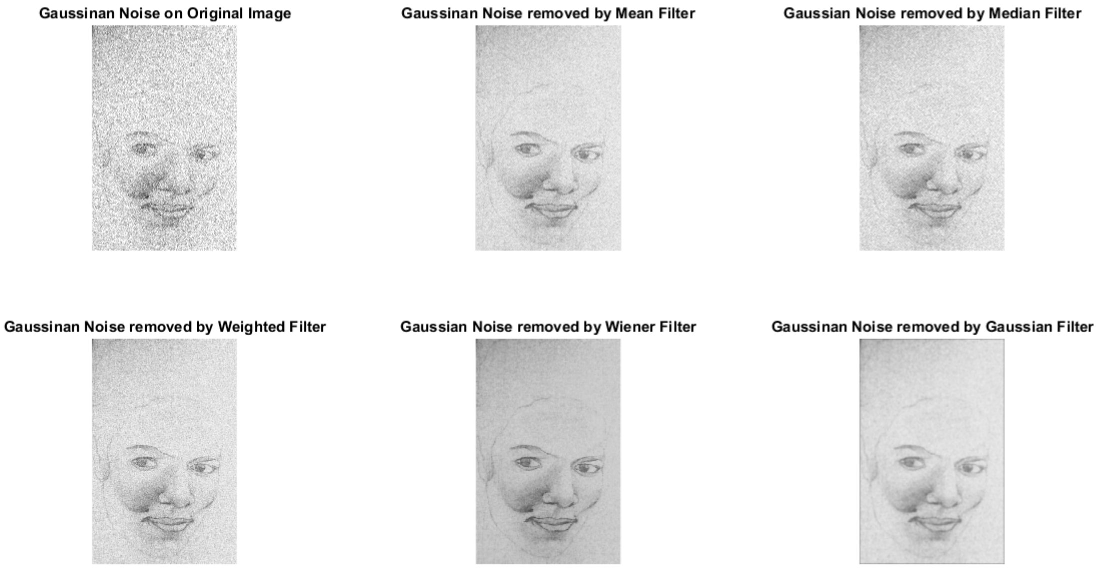


Figure 5: Gaussian Noise

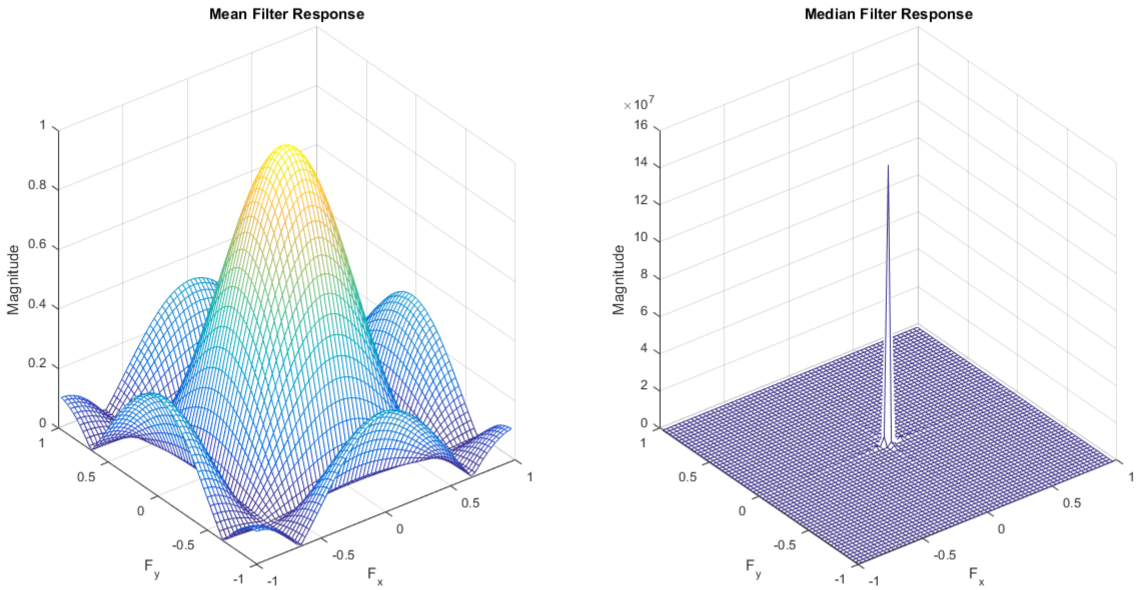


Figure 6: Plot of Mean and Median Filter Response

4 Conclusion

Hence,
We are familiarized with how the bit plane slicing works and various types of filters, its properties and effects on various types of noises of an image using the MATLAB application.