Module Code: CS3IA16

Assignment report Title: Image Enhancement

Student number: 30008602, 30006477, 30020691

Date (when the work completed): 27/10/2023

Actual hrs spent for the assignment: ~20 hours total

Assignment evaluation (3 key points):

- Removing noise in the frequency domain
- Removing noise in the spatial domain
- Enhancing images using image processing filters

Name	Contribution (in %)	Note (briefly explain the contribution)	Signature
Samual Jones	100%	Report + Matlab	S.Jones
Ben Bridgeman	100%	Report + Matlab	Ben Bridgeman
Jaweed Inayathulla	100%	Report + Matlab	Jaweed.I

Abstract

This project aims to show the fundamental methods in image analysis for frequency and spatial domain used in image enhancement. Frequency filters were used to cancel out the periodic noise in the frequency domain after Free-Fourier Transform (FFT). To cancel out the random noise in the spatial domain different filters were used. The distorted image can then be compared to the original image using the Mean Square Error (MSE), the lower the value the closer it is to the original image. Research throughout the methodoly were all through experimenting and comparison with the final decision being agreed by all group members.

Introduction

The main idea behind image enhancement is to process an image to improve its suitability for a particular use. The objective of this process was to demonstrate that using a variety of filters combined with transformation in both frequency and spatial domains, it can improve distorted images closer to their original quality. This document will explain the different trials and experiments we experienced. The methodology will mainly explain what we did to end on our final result whereas the results will explain more Indepth on the filters and enhancements which were tested but not agreed apon.

Methodology

All images can be shown in Figure 1 and demonstrate visual and MSE improvements after each filter application.

We initiated our image processing by applying a 3x3 median smoothing filter. This filter operates by evaluating a 3x3 grid of pixels surrounding the target pixel, determining the median value within this local neighbourhood, and then applying this median value to the target pixel. The primary aim of this step was to reduce noise in the distorted image. We chose the 3x3 grid size over a 5x5 grid because it offered a less aggressive approach, preserving edges and finer details. We found this very important in the early stages because any loss of detail would make it harder to restore the image and potentially worsen later results. The starting methodology gave more significant results on the overall quality.

Subsequently, we performed a Fast Fourier Transform (FFT) on the smoothed image to shift it from the spatial domain to the frequency domain. The Fourier transform breaks the image down into its frequencies so we can analyse the magnitude and patterns. FFT was chosen over DFT in this application for several reasons but mainly for its edge in efficiency and simplicity. In the frequency domain, we calculated the magnitude by taking the log of the absolute value at each point (Figure 2). We saved this transformed image to local files for further processing. To create masks around specific image features, we used Adobe Photoshop but any photo editing software would have worked just as well. Photoshop was employed to generate elliptical masks around each peak and the final image was then saved in the same folder as the script (Figure 3). The mask was then imported and applied to the FFT image to suppress most of the periodic noise. This was a more complex method than taking a threshold value but in testing, we found it was improved results and gives us better control. Afterwards, we performed an inverse FFT to switch back to the spatial domain. A 5x5 median filter was then applied, which yielded better results compared to a 3x3 grid this time and the random noise remaining was heavily reduced.

Next, we adjusted the contrast using the imadjust() function. This contrast adjustment method was preferred over histogram equalization as we found the post-adjusted images were more accurate and proven with lower MSE numbers. It enhanced the image by darkening dark sections, brightening brighter areas, improving object visibility, and delivering sharper edges. We found this adjustment to be the most impactful on the MSE taking it from 1436 down to 444, the largest jump in all the filters.

Finally, a Gaussian filter was applied. The Gaussian filter is a much more subtle smoothing effect, working by applying a weighted average to pixel values within the local neighbourhood. This technique, in contrast to the median filter, is adept at preserving the image's overall structure and details. This step further reduced the MSE to our final value of 417.

Additional testing found that the image would lose quality as well as increase the MSE so we found that this was the lowest and most accurate methodology we experimented with. In addition, whenever using parameters in functions, we ran various nested for loops which would try out different values and output the best results so we know that the included values weren't randomly chosen and yielded the best results.

Results and Discussion

One crucial piece of information we discovered during testing was that there seemed to be two layers of noise, one of which was periodic meaning that there was a pattern. The second layer was more random and appeared to be similar to a "salt & pepper" noise. We found better results when removing the periodic noise in the frequency domain and then removing the leftover noise in the spatial domain.

We did test different noise removal techniques including a bandpass filter but decided on a notch filter as it provided great results and allowed us to create accurate masks. The most complex part of doing it through our method was using external software to individually mask the intense points on the magnitude graph. This was useful for this task but in a situation where multiple images are used, a custom threshold or radius method might be better as it takes less manual adjustments.

We also found that smoothing the image slightly helped to provide a better post-masked image as the noise was already slightly reduced. This was accomplished through a median filter where we tried different values for the neighbourhood and settled on a 3x3. After the masking, we found a reduced MSE by once again smoothing with a median filter which cleaned up more excess noise but this time a 5x5 neighbourhood showed better results than a 3x3.

At this point, the image shows vast improvements with much less noise than what we started with, but the contrast of the image is far from the original. Through experimenting we found two methods that showed good results which were imadjust() and histeq(). Imadjust() was the chosen method as it not only showed better results to the human eye when compared to the original but also the MSE agreed that it was more accurate.

Lastly, we tried one last smoothing method which remained in the spatial domain which was the Gaussian blur. For this, we used a live script on Matlab and adjusted the input parameter using a numeric slider and watched to see where the MSE lowered. We tested sigma values 0-3 increasing at 0.1 intervals and found the best was when sigma = 1.2. Lastly, we ended with an MSE of 417.07 and found that more image enhancement methods lead to a higher MSE and loss of quality.

Conclusion

During our testing it was hard to conclude whether or not an image was more accurate than the original, as the human eye interprets images differently from one person to another. Also, just because the MSE of an image improves, doesn't mean that it looks better, as we found that on some occasions the MSE lowered, but some quality was lost. It was interesting to see which filters would visually improve the image and which would lower the MSE.

Appendix

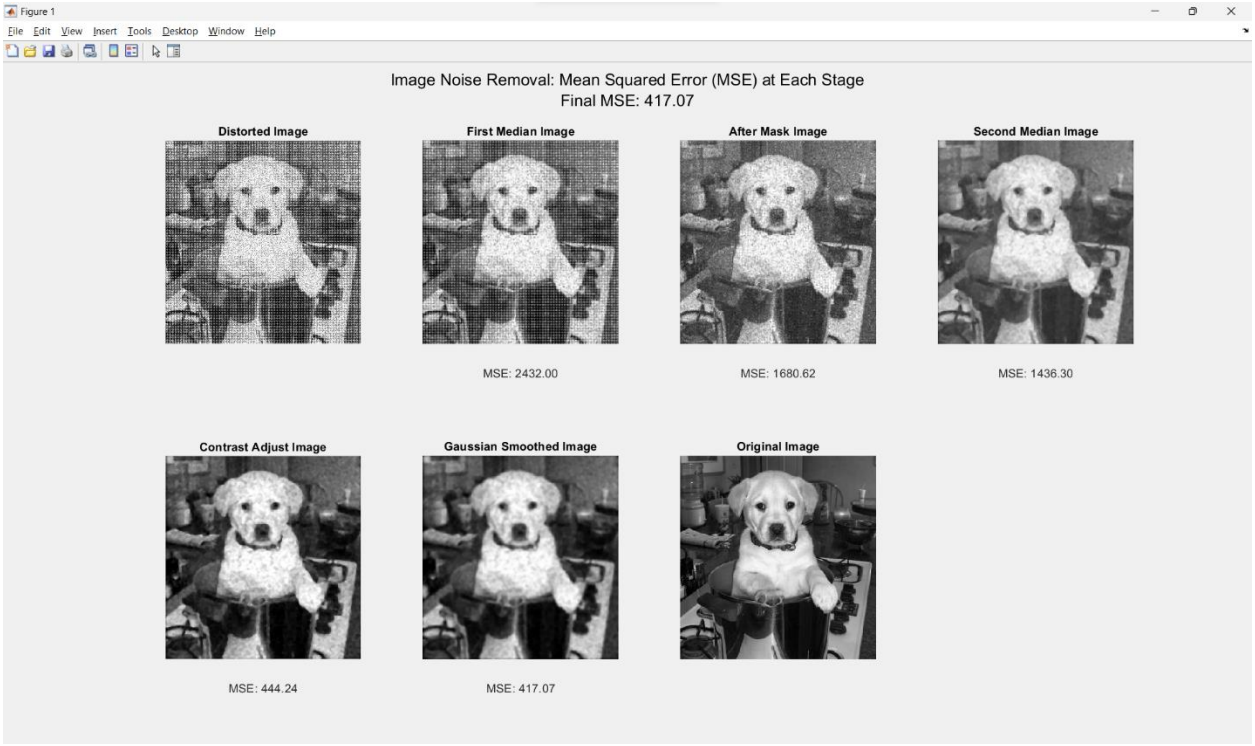

Figure 1 All Images throughout matlab restoration

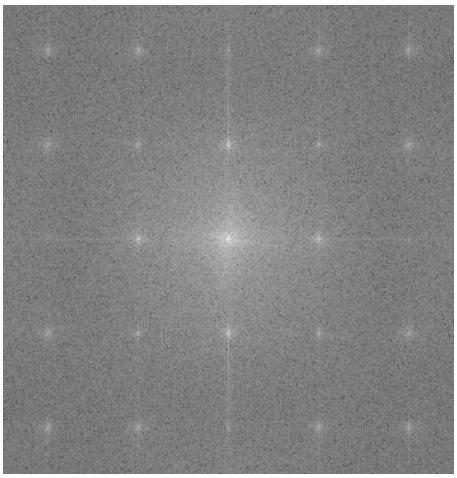

Figure 2 Amplitude graph before masking

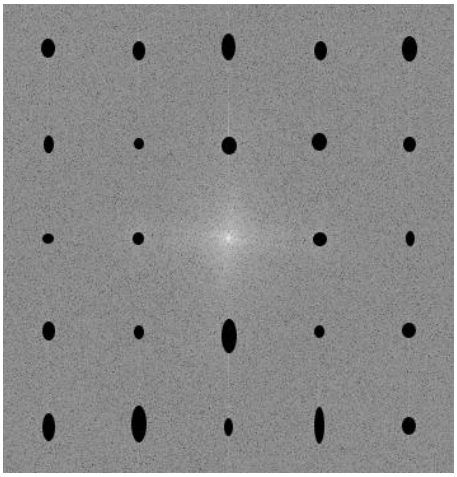

Figure 3 Masked amplitude graph using photo editing software

All resources can be accessed via GitHub: https://github.com/Joness07/Noise-removal-Matlab.git

Code

```
clc;
clear;
close all;

%load the images and create MSE array
orig = imread("dogOriginal.bmp");
dist = imread("dogDistorted.bmp");
mse_values = zeros(7, 1);

%apply median filter
med1 = medfilt2(dist, [3 3]);

%fft transform and intensity graph
freq = fftshift(fft2(double(med1)));
iGraph = log(abs(freq));
imwrite( iGraph/max(iGraph(:)), "intensityGraph.png");

%loads mask image (created outside)
mask = imread("mask.png");
mask = mask(:,:,1);
mask = double((mask>0));

%applies mask and inverses fft
maskApplied = mask .* freq;
postMask = real(ifft2(ifftshift(maskApplied)));
```

```
%second median filter
med2 = medfilt2(postMask, [5, 5]);

%adjusts constrast
adjust = imadjust(uint8(med2));

%applies gaussian filter
gauss = imgaussfilt(adjust, 1.2);

% Calculate all MSE values
mse_values(1) = immse(orig, uint8(med1));
mse_values(2) = immse(orig, uint8(postMask));
mse_values(3) = immse(orig, uint8(med2));
mse_values(4) = immse(orig, uint8(adjust));
mse_values(5) = immse(orig, uint8(gauss));


% Display all the images
figure;
subplot(2,4,1);
imshow(uint8(dist));
title("Distorted Image");

subplot(2,4,2);
imshow(uint8(med1));
title("First Median Image");
xlabel(sprintf("MSE: %.2f", mse_values(1)));

subplot(2,4,3);
imshow(uint8(postMask));
title("After Mask Image");
xlabel(sprintf("MSE: %.2f", mse_values(2)));

subplot(2,4,4);
imshow(uint8(med2));
title("Second Median Image");
xlabel(sprintf("MSE: %.2f", mse_values(3)));

subplot(2,4,5);
imshow(uint8(adjust));
title("Contrast Adjust Image");
xlabel(sprintf("MSE: %.2f", mse_values(4)));

subplot(2,4,6);
```

```
imshow(uint8(gauss));
title("Gaussian Smoothed Image");
mse_values(6) = immse(orig, gauss);
xlabel(sprintf("MSE: %.2f", mse_values(5)));

subplot(2,4,7);
imshow(uint8(orig));
title("Original Image");
mse_values(7) = 0;

titleText = sprintf("Image Noise Removal: Mean Squared Error (MSE) at Each Stage\nFinal MSE: %.2f",
mse_values(6));
sgtitle(titleText);
```

Refrences

Mathworks.com. (2019). *Image Processing Toolbox*. [online] Available at: https://uk.mathworks.com/products/image.html.

www.youtube.com. (n.d.). *Matlab Gaussian Noise Removal*. [online] Available at: https://www.youtube.com/watch?v=8lIqxk9tyf8& [Accessed 20 Oct. 2023].