Image Processing Using Mean, Median and Conservative Filters for Enhanching An Image

P. Manga Rao[1], M. Akash[2], P. Vijay Pal Reddy[3], M. Madhan Kumar[4], N. Tagore Reddy[5]
[1]*Assistant Professor, Dept of ECE, TKR College of Engineering and Technology*
[2,3,4] *Students, Dept of ECE, TKR College of Engineering and Technology*

***Abstract*—The project actually implements the image processing systems for the purpose of improving quality and identifying the defect in digital images. The combination of conservative, mean, as well as median, for contrasting enhancement. The PCB defect detection can enable for the purpose of transformation in order to perform digital images through noise reduction.**

***Index Terms*—Conservative Filtering", "Mean Filters", "PCB Defect Detection", "Median Filtering"**

I. INTRODUCTION

It is important to implement image processing techniques to obtain and improve valuable information contained in digital images, and the main purpose of this project is to implement various image filtering techniques that will remove noise and improve the image clarity of the overall image and create visually useful features. These filters include mean, median, adaptive, contrast enhancement, and PCB defect detection filters.

II. FILTER BACKGROUND

The quality and usability of images in image processing depend on filters as they are important digital instruments that reduce undesired noise and emphasize relevant content. Mean filters smooth images through averaging pixels and median filters eliminate salt-and-pepper noise [1]. An adaptive filter keeps edges in the output and contrast enhancement increases visibility.

Mean Filter

"Mean filtering" is simple and well-known method for image denoising, in which the pixel value in the image is replaced with the mean of all pixel values in a specified area (or window), yielding a smoothed image. When applying a mean filter, the mean pixel value is calculated from pixels within the "area constrained by a sliding window as the window traverses throughout the entire image. The central pixel value is then replaced with the average pixel value. Mean filtering is effective at reducing Gaussian noise where the noise is drawn from the normal distribution as it is simple to compute and can produce lower time requirements which can make it a more suitable option for use in real-time processing. The most significant drawback to mean filtering is although it reduces noise in images it simultaneously causes blurring of edges and features in the image. With macing all pixel information, it will cause sudden changes in intensity to present as smooth surfaces which leads to a loss of sharpness. Mean filtering is therefore particularly most suitable for applications where eliminating, or reducing, noise is prioritized over retaining finer detail in an image. Still, in many applications such as medical imaging, video processing, or even general image enhancement, mean filtering produces good results.

Conservative Filter

Conservative filtering, a denoising method, reduces noise while maintaining details and edges. Conservative filtering does not replace pixel values with the average of neighboring pixels, as in mean filtering, but rather modifies the pixel value if it falls below or above the minimum or maximum values of the range of neighboring pixels. This filter is applied by positioning a window around each targeted pixel, such as a "3x3" or "5x5 pixel window", to establish the maximum and minimum values based on neighboring pixels. If the pixel value is ranked above the maximum of the neighboring pixel values in the

window, the pixel value is modified based on the maximum. Conservative filtering is effective in reducing salt-and- pepper noise while preserving edge sharpness and detail. Due to its edge preservation, it is often utilized in fields that require fidelity. One of the main advantages of conservative filtering is the limited distortion it creates, even in images with complex textures or rapid intensity changes. Alternatively, if the pixel value is less than the minimum neighboring pixel value in the window, the pixel value is replaced with the minimum neighboring pixel value. If the pixel value lies between the minimum and maximum value, it is unchanged. Thus, we define the method in three steps: for each pixel copied, assess the minimum and maximum neighboring pixel values of the window; if greater than the maximum neighboring pixel value, replace with the maximum pixel value; if the pixel value is less than the minimum, then replace the pixel with the minimum; or if the original value is in the range, keep the original pixel value.

III. FILTER DEVELOPMENT

```
% Main script to apply Conservative Smoothing Filter to an image using path

% Image path (replace with your actual path)
image_path = 'D:/Matlab/IMG-20250316-WA0047.jpg';

% Load the image
img = imread(image_path);

% Define the window size for the conservative smoothing filter
window_size = 3;  % You can adjust this value

% Apply the conservative smoothing filter using the path image
filtered_image = conservative_smoothing_filter(img, window_size);

% Display the original and filtered images side by side
figure;
subplot(1, 2, 1);
imshow(img);
title('Original Image');

subplot(1, 2, 2);
imshow(filtered_image);
title('Filtered Image');

function output = conservative_smoothing_filter(input_image, window_size)
    % Convert the image to double for precision
    input_image = double(input_image);

    % Get the size of the image
    [rows, cols, channels] = size(input_image);

    % Initialize the output image
    output = input_image;
```

Figure 1: Conservative Smoothing Filter

The figure presents the MATLAB script which executes the Conservative Smoothing Filter onto an image. The program first retrieves the image from an identified file path before it applies the filter across a specific window area. The figure presents an original image displayed next to its filtered copy to show how the filter eliminates noise without distorting edge characteristics.

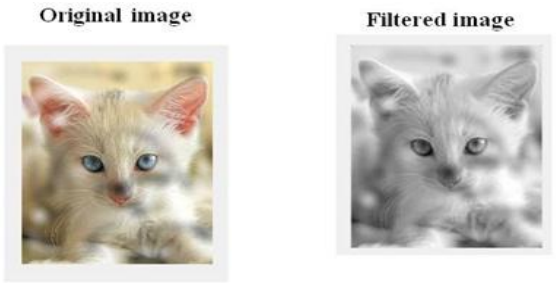

Figure 2: Original and Filtered Image

The implementation of the Conservative Smoothing Filter on the cat image appears in this image display. This figure presents the original image on the left side with its filtered Version shown on the right [2]. The conservative smoothing filter provides noise reduction that leads to image smoothing yet maintains critical shapes and edges to stop excessive blurring.

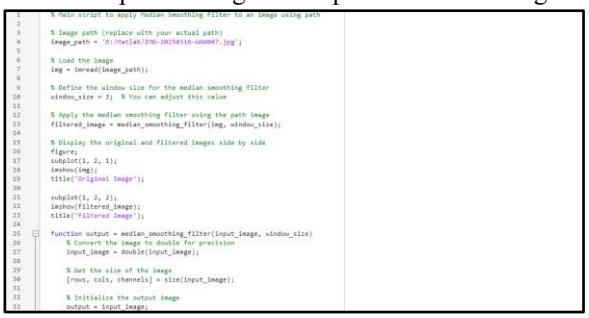

Figure 3: Median Smoothing Filter

A “Median Smoothing Filter processing” takes place within the script of this figure. A particular image path leads to the loading process while the filter window dimension receives a specific definition. A median filter reduces image noise prior to viewing both filtered and original picture side by side for comparison of denoising effects.

Figure 4: Original and Filtered Image

The picture reveals the unaltered cat image on the left followed by the right-hand side display of the filtered image after using the “Median Smoothing Filter”. Because of its effectiveness in removing salt-and-

pepper noise the median filter maintains clear edges which protects features such as fur and eyes from becoming distorted or blurred in the cat image.

```
% Main script to apply Mean Smoothing Filter to an image using path

% Image path (replace with your actual path)
image_path = 'D:/Matlab/IMG-20250316-WA0047.jpg';

% Load the image
img = imread(image_path);

% Define the window size for the mean smoothing filter
window_size = 3;  % You can adjust this value

% Apply the mean smoothing filter using the path image
filtered_image = mean_smoothing_filter(img, window_size);

% Display the original and filtered images side by side
figure;
subplot(1, 2, 1);
imshow(img);
title('Original Image');

subplot(1, 2, 2);
imshow(filtered_image);
title('Filtered Image');

function output = mean_smoothing_filter(input_image, window_size)
    % Convert the image to double for precision
    input_image = double(input_image);

    % Get the size of the image
    [rows, cols, channels] = size(input_image);

    % Initialize the output image
    output = input_image;
```

Figure 5: Mean Smoothing Filter

This figure in the MATLAB script shows the "Mean Smoothing Filter" applied to an image. The filter performs averaging of adjacent image pixels in defined windows [3]. In these display boxes, the original image is put next to the image processed by the mean filter, which allows users to see how mean filtering affects image quality by smoothing the image and reducing noise, although it might, of course, potentially eliminate some picture elements in the process.

Figure 6: Original and Filtered Image

IV. GUI DEVELOPMENT

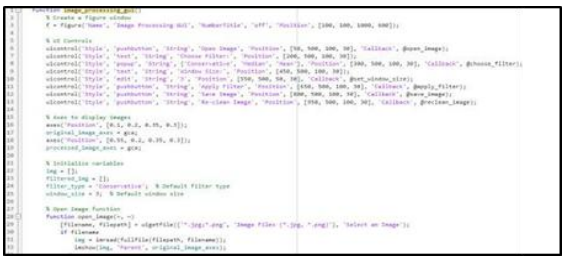

Figure 7: MATLAB Code for Image Processing

The figure provides MATLAB code that demonstrates how to use the "Conservative Smoothing Filter" for digital image processing [4]. The code first establishes the image path and then loads the image before the filter processing function runs along with the original and filtered image presentation side by side.

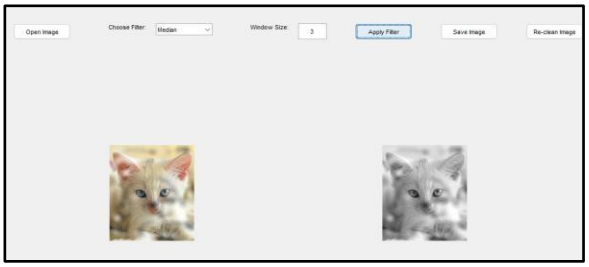

Figure 8: Original and Filtered Image - Median Filter

The cat image demonstrates the Median Filter's operational performance in this figure. The right side shows the filtered image in conjunction with the left side's original image. The median filter is effective in removing noise (particularly salt-and-pepper noise) while preserving the structural elements of the image.

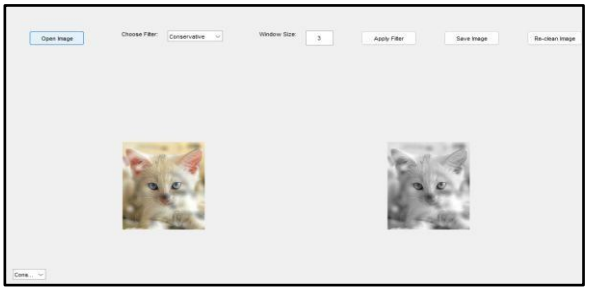

Figure 9: Original and Filtered Image - Conservative Smoothing Filter

The visual outcome of filtering the cat image with a Conservative Smoothing Filter appears in this illustration. The source image sits to the left while the filtered version takes position on the right side [5]. The filtering process smoothens images by minimizing noise and it retains significant edges together with important details in the image.

Figure 10: Original and Filtered Image – Mean Filter The cat image reveals its outcome after using a

Mean Filter based on this diagram. The figure displays the original image on the left side with its filtered version shown on the right. Noise reduction takes place through pixel value averaging in the mean filter which also causes blurring of certain image details.

V.ADDITIONAL WORK OF PERSONAL INTEREST

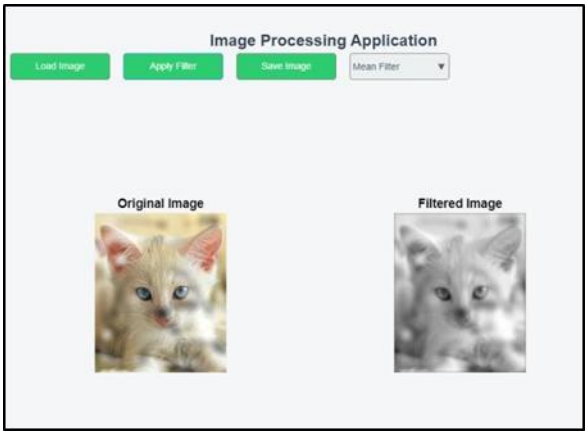

Figure 11: Mean Filter in Additional Work

The "Mean Filter" functions upon this cat image. The original image appears on the left side of the figure and the filtered image displays on the right side after the local window used for pixel averaging blurs the image to reduce noise [6]. This specific figure also demonstrates the filtering effect on the real image according to the changes done on each figure.

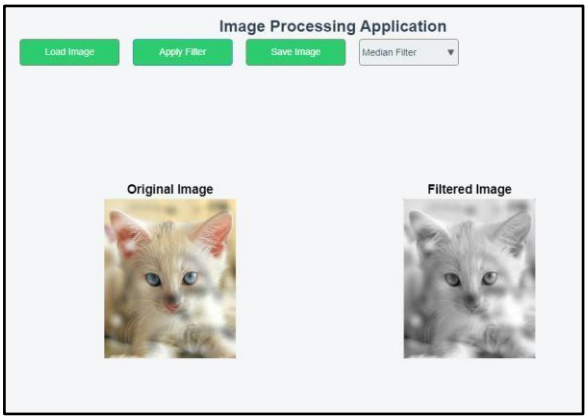

Figure 12: Median Filter in Additional Work

The result of the Median Filter application appears in this figure. Noise is successfully eliminated from the image, especially from salt-and-pepper noise but edges stay intact between the original image (left) and the filtered image (right).

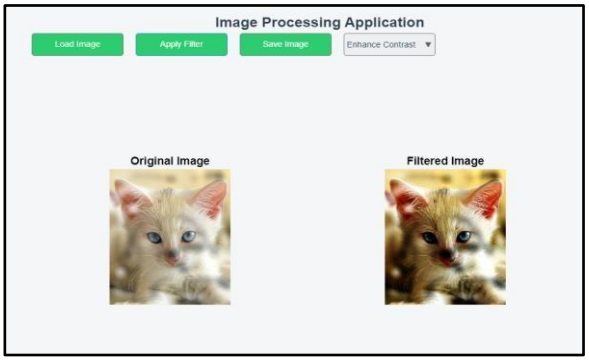

Figure 13: Contrast Enhancement in Additional Work The image has received Contrast Enhancement treatment according to this illustration [7]. The original image appears on the left side of this display alongside the right-side image which contains enhanced contrast details that stand out from the background.

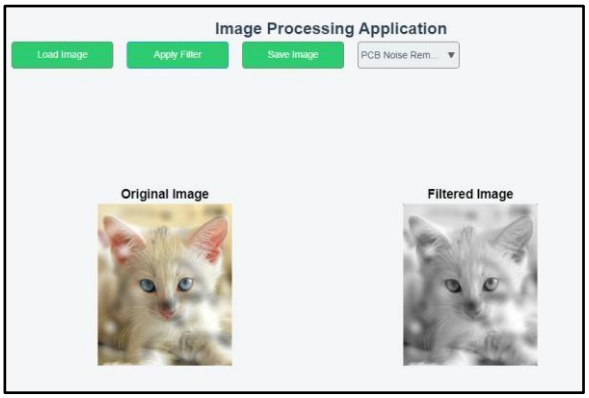

Figure 14: PCB Noise Removal in Additional Work The supplied figure demonstrates how PCB Noise Removal works on the image. The original image appears on the left with the right side showing a filtered version where a median filter led to noise reduction.

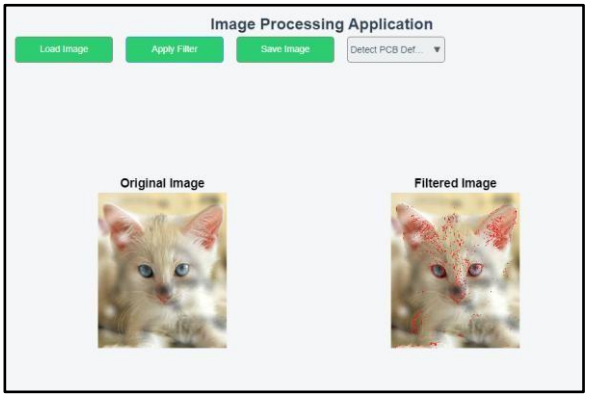

Figure 15: PCB Defect Detection in Additional Work The results from PCB Defect Detection

appear in this figure. The original image appears on the left side while the right side depicts the filtered version which utilizes edge detection to paint defects red although these might become faults.

VI. CONCLUSION

Multiple types of image filters result in better-quality images while simultaneously reducing their noise content. Visual clarity benefits from three rehabilitation techniques along with contrast enhancement which together produce effective results". The PCB defect detection filters serve as essential image processing tools because they detect important issues in visual data.

VII. ACKNOWLEDGMENTS

We are deeply thankful to the developers who created the graphical user interface (GUI) and image processing tools because their useful resources enabled advancement in this project. We express our gratitude to open-source developers alongside researchers who contribute to image processing which makes this system professionally operational.

REFERENCES

[1] Madhusudana, P.C., Birkbeck, N., Wang, Y., Adsumilli, B. and Bovik, A.C., 2022. Image quality assessment using contrastive learning. IEEE Transactions on Image Processing, 31, pp.4149-4161.

[2] Reinke, A., Tizabi, M.D., Sudre, C.H., Eisenmann, M., Rädsch, T., Baumgartner, M., Acion, L., Antonelli, M., Arbel, T., Bakas, S. and Bankhead, P., 2021. Common limitations of image processing metrics: A picture story. arXiv preprint arXiv:2104.05642.

[3] You, J. and Korhonen, J., 2021, September. Transformer for image quality assessment. In 2021 IEEE international conference on Image Processing (ICIP) (pp. 1389-1393). IEEE.

[4] Zhang, W., Ma, K., Zhai, G. and Yang, X., 2021. Uncertainty-aware blind image quality assessment in the laboratory and wild. IEEE Transactions on Image Processing, 30, pp.3474-3486.

[5] Ding, K., Ma, K., Wang, S. and Simoncelli, E.P., 2021. Comparison of full-reference image quality models for optimization of image processing systems. International Journal of Computer Vision, 129(4), pp.1258-1281.

[6] Yang, S., Wu, T., Shi, S., Lao, S., Gong, Y., Cao, M., Wang, J. and Yang, Y., 2022. Maniqa: Multi-dimension attention network for no-reference image quality assessment. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 1191-1200).

[7] Golestaneh, S.A., Dadsetan, S. and Kitani, K.M., 2022. No-reference image quality assessment via transformers, relative ranking, and self-consistency. In Proceedings of the IEEE/CVF winter conference on applications of computer vision (pp. 1220-1230).

[8] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 4th ed., Pearson Education, 2018.

[9] A. K. Jain, *Fundamentals of Digital Image Processing*, Prentice Hall, 1989.

[10] S. Jayaraman, S. Esakkirajan, and T. Veerakumar, *Digital Image Processing*, Tata McGraw-Hill Education, 2011.

[11] R. Szeliski, *Computer Vision: Algorithms and Applications*, Springer, 2010.

[12] D. Forsyth and J. Ponce, *Computer Vision: A Modern Approach*, 2nd ed., Pearson, 2012.

[13] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 4th ed., Pearson Education, 2018.

[14] A. K. Jain, *Fundamentals of Digital Image Processing*, Prentice Hall, 1989.

[15] S. Jayaraman, S. Esakkirajan, and T. Veerakumar, *Digital Image Processing*, Tata McGraw-Hill Education, 2011.

[16] R. Szeliski, *Computer Vision: Algorithms and Applications*, Springer, 2010.

[17] D. Forsyth and J. Ponce, *Computer Vision: A Modern Approach*, 2nd ed., Pearson, 2012.

[18] OpenCV Documentation, "Open-Source Computer Vision Library," Available: https://opencv.org/

[19] MATLAB Documentation, "Image Processing Toolbox," MathWorks, Available: https://www.mathworks.com/

[20] N. Otsu, "A Threshold Selection Method from Gray-Level Histograms," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.

[21] J. Canny, "A Computational Approach to Edge Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, no. 6, pp. 679–698, 1986.