A Shadow Removal Method for Tesseract Text Recognition

Huimin Lu[1] , Baofeng Guo[1], Juntao Liu[2], Xijun Yan[2]
[1]School of Automation, Hangzhou Dianzi University,Hangzhou, China
[2]China Academy of Launch Vehicle Technology, China Aerospace Science and Technology Corporation,Beijing, China

***Abstract*—For shadowed text images, the character recognition performance of Tesseract drops significantly. In this paper, we propose a new method to process the shadowed text images for the Tesseract's optical character recognition engine. First, a local adaptive threshold algorithm is used to transform the grayscale image into a binary image to capture the contours of texts. Next, to delete the salt-and-pepper noise in the shadow areas we propose a double-filtering algorithm, in which a projection method is used to remove the noise between texts and the median filter is used to remove the noise within characters. Finally, the processed binary image is fed into the Tesseract's optical character recognition engine. Experimental results show that the proposed method can achieve a better character recognition performance.**

Keywords- Tesseract; shadow; local adaptive threshold; projection denoising; median filter

I. INTRODUCTION

The Optical Character Recognition (OCR) technology has a wide range of applications in document processing. The text image information could be input to the computer quickly and accurately with an excellent OCR system [1]. In most cases, text images are printed as a black-on-white style. However, the recognition of degraded text images with uneven illuminations is still a great challenge for researchers.

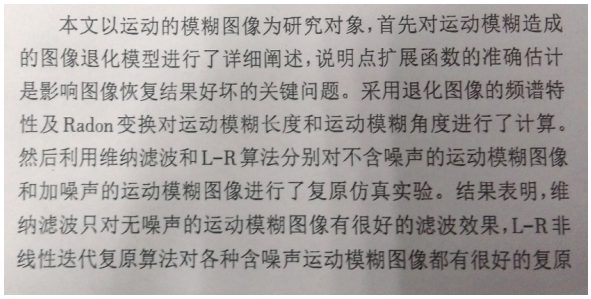
The Tesseract OCR engine is an open-source system that was developed originally in HP Labs between 1985 and 1995, shelved for 10 years, open-sourced in 2006 and now developed mostly at Google. It has gained in popularity due to its accuracy and support for a great many languages [2,3]. The project can be available now at http://code.google.com/p/tesseract-ocr. If a printed text image is not severely corrupted by noise, the recognition rate of Tesseract is fairly high. On the contrary, the performance of Tesseract will drop significantly if the input image contains much noise, e.g., the shadows. Fig. 1(a) shows a typical text image captured by a mobile phone, where shadows can be found on the right side of the picture. It is known that Tesseract employs the Otsu segmentation method to process the input image, so we transform Fig. 1(a) into a binary format using the Otsu method. The results are shown in Fig. 1(b). It is seen that the shadow area of Fig. 1(a) becomes an area of black in Fig. 1(b) after the Otsu segmentation. Consequently the OCR performance of Tesseract becomes very poor because Fig. 1(a) is not correctly pre-processed for Tesseract.

To improve the recognition accuracy for shadowed text images, some techniques have been proposed. Gamma correction, or often simply gamma, is a nonlinear operation usually used to encode and decode luminance or tristimulus values in video or still image systems. The gamma correction method can improve a little light distribution for text images, but its performance is not good [4]. Besides, the adaptive histogram equalization (AHE) and the improved algorithms based on AHE are presented to improve uneven illumination [5-7]. These approaches proved to be very effective for gray level images, but they are not preferred for text images. In most cases, the feature of text image is significantly different from the graphic image in frequency distribution. In order to obtain a good recognition result, the contrast of the text and background in a text image are supposed to be large enough, while the AHE scheme will increase the pixel value in the non-character information range, thus reducing the contrast of the image. What's more, the adaptive histogram equalization method produces a blocky effect, which is detrimental to subsequent processing.

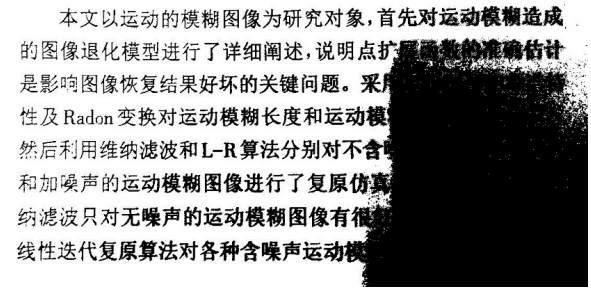
To solve the problem of Tesseract's weak performance for shadowed text images, we propose to generate binary images with less amount of noise before input into Tesseract. First of all, a local adaptive threshold processing is presented by replacing the traditional Otsu method, which can generate a binary image from a grayscale image and the binary image will contain a certain amount of salt-and-pepper noise. The noise is mainly distributed in the shadow areas. To remove the noise from the binary image, a double filter is further proposed in consideration of both denoising effect and computational cost. We first project the binary image in the horizontal direction and the vertical direction to remove the noise between the text lines. This step will remove the noise in the non-text area effectively. After removing the majority of the noise in the shadow areas, we employ the median filter to eliminate the noise within the regions of characters. Since the input of Tesseract can be color images, gray level images or binary images, we finally feed the binary image generated by our scheme into the Tesseract engine and evaluate its classification performance.

The rest of this paper is organized as follows. In Section II, we provide a detailed scheme of our shadow removal method for Tesseract text recognition. Experimental results and discussions are presented in Section III. We draw conclusions in Section IV.

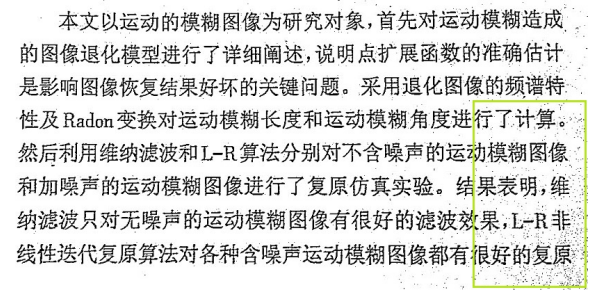
978-1-5386-1937-7/17/$31.00 ©2017 IEEE



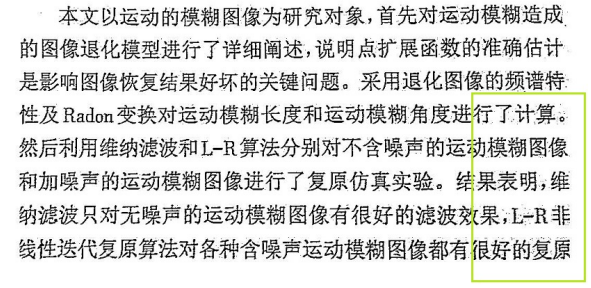
(a)



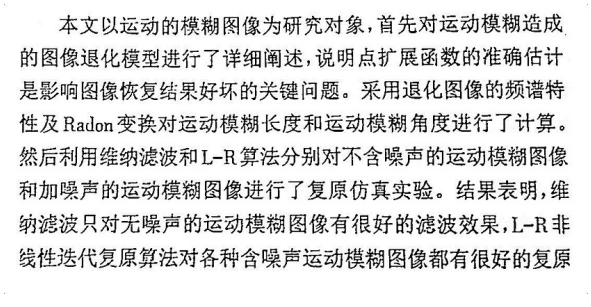
(b)



(c)



(d)



(e)

Fig. 1. Outcome of each step: (a) Original text image with shadows; (b)Segmentation using Otsu method; (c) Results using adaptive threshold; (d) Results using projection denoisng; (e) Results using median filtering

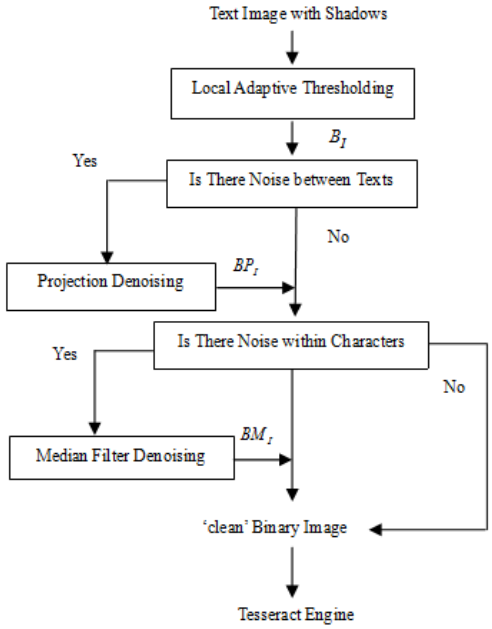


Fig. 2. Diagram of the proposed method

II. PROPOSED SCHEME FOR SHADOWED TEXT IMAGES PROCESSING

The proposed method basically consists of three steps, including (1) a local threshold processing, (2) a projection-based denoising and (3) a median filtering. The flowchart of the scheme is shown in Fig. 2. The original text images are degraded by shadows.

A. Local Threshold Processing

1) Graying: In this step, we first transform the input text image into a grayscale image using the following formula:

$$G_I = w1 \times R(i,j) + w2 \times G(i,j) + w3 \times B(i,j) \qquad (1)$$

where G_I is the output grayscale image, and R, G, B denote the pixel values of red, green and blue channel of the color image respectively. Coefficients $w1 - w3$ are weights of three channels and the sum of $w1 - w3$ must be 1.

2) Binarization with Adaptive Threshold: In this step, we aim to generate the binary image B_I from G_I. For a grayscale image G_I, we need to find a threshold T to separate the text area and background by the following equation [8]:

$$B_I(i,j) = \begin{cases} 1, & G_I(i,j) > T \\ 0, & G_I(i,j) \le T \end{cases} . \qquad (2)$$

where $1 \le i \le I$ and $1 \le j \le J$. I and J are the height and width of the original image respectively. If a pixel value in G_I is higher than T, we treat it as an object pixel and set its value to 1 in B_I. Otherwise we treat it as a background pixel and set its value to 0 in B_I. With a pixel-by-pixel comparison, we eventually obtain the binary image.When T is a constant that applies to the entire image, the process depicted in (2)

is called global thresholding; if the value of T changes on different locations on the image, the technique is called local thresholding thereby.

For shadowed text images, the global thresholding technique like Otsu method adopted by Tesseract does not work well, since it segments the whole image with a uniquely determined threshold. A stable threshold is able to separate the background from the image in some areas, but it does not work in other areas [9]. Therefore, we utilize an adaptive thresholding technique to segment G_I, which is proved effective for text images with uneven illumination. We choose the function cvAdaptiveThreshold() in the OpenCV library to process the image. OpenCV is an open-source computer vision library, which is written in C++ language and has achieved many common algorithms on image processing and computer vision [10,11]. We illustrated the adaptive threshold processing inside cvAdaptiveThreshold() as follows.

Let $f(i,j), 1 \leq i \leq I, 1 \leq j \leq J$ be the input image. For every pixel (i,j) in f, the mean m_{ij} and variance v_{ij} are calculated by:

$$m_{ij} = 1/(2p) \sum_{s=-p}^{p} \sum_{f=-p}^{p} f(i+s, j+t) \tag{3}$$

$$v_{ij} = 1/(2p) \sum_{s=-p}^{p} \sum_{f=-p}^{p} |f(i+s, j+t) - m_{ij}| \tag{4}$$

where $p \times p$ is the neighborhood centered with coordinates (i,j). Local threshold for pixel (i,j) is $t_{ij} = m_{ij} + v_{ij}$ for $v_{ij} > v_{min}$, and $t_{ij} = t_{i,j-1}$ for $v_{ij} \leq v_{min}$, where v_{min} is the minimum variance value. After every threshold t_{ij} for pixel (i,j) is calculated, we use formula (2) to generate the binary image B_I.

Fig. 1(c) shows the image B_I generated from Fig. 1(a), and obviously the quality of Fig. 1(c) is much better than that of Fig. 1(b), though some salt-and-pepper noise still exists in it. We mark the main noise area with green rectangle.

B. Projection Denoising

The image B_I usually contains a lot of noise, which will decrease the OCR performance significantly. The median filter is usually utilized to reduce the salt-and-pepper noise. However, if the window size of the median filter is small, the filtering performance is not very good because some outlier noises still remain after filtering; if the window size is large, the text contents will be damaged and computational cost will increase. Therefore, we first use a projection method to remove the majority noise in B_I, then the median filter with a window size of 3×3 is employed to reduce the rest isolated noise points.

For most of languages, all text lines are appear either horizontally or vertically [12]. Therefore, we can take horizontal text layout into consideration in this case.

1) Horizontal Projection: First, the text image B_I is horizontally projected to calculate the sum of pixel value of each row. Since the the line spacing of text images is generally equal [13], we are able to find a threshold T_r by analyzing the horizontal projection result $R(i), 1 \leq i \leq I$. $R(i)$ is calculated by:

$$R(i) = \sum_{j=1}^{J} B_I(i,j), 1 \leq i \leq I \tag{5}$$

For each particular i in $R(i)$, we use the following formula to remove the noise between texts:

$$B_I(i,j) = \begin{cases} B_I(i,j), & R(i) \leq T \\ 1, & R(i) > T \end{cases} . \tag{6}$$

where $1 \leq j \leq J$.

2) Vertical Projection: Secondly, B_I is vertically projected to the remove noise in the non-text areas on the left and right sides (if such areas exist). Similarly, a threshold T_c can be found by analyzing the vertical projection result $C(j), 1 \leq j \leq J$. $C(j)$ is calculated by:

$$C(j) = \sum_{i=1}^{I} B_I(i,j), 1 \leq j \leq J \tag{7}$$

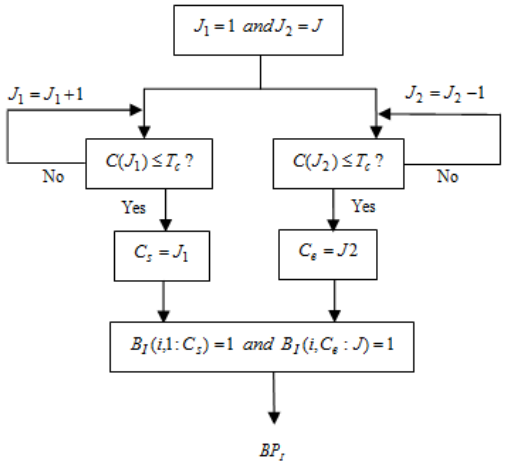


Fig. 3. Flowchart of vertical projection denoising

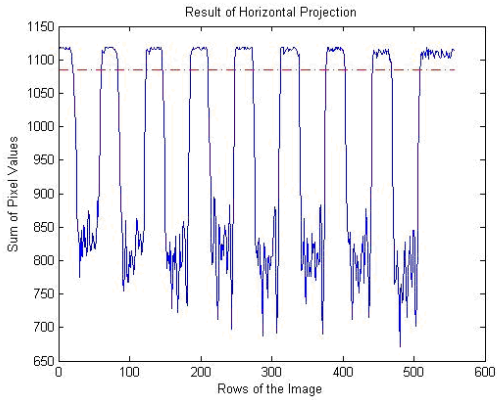
Fig. 3 demonstrates the vertical projection denoising process. Detailedly, we illustrate this process step by step as following:

(1) Set $j_1 = 1$ and $j_2 = J$ initially.

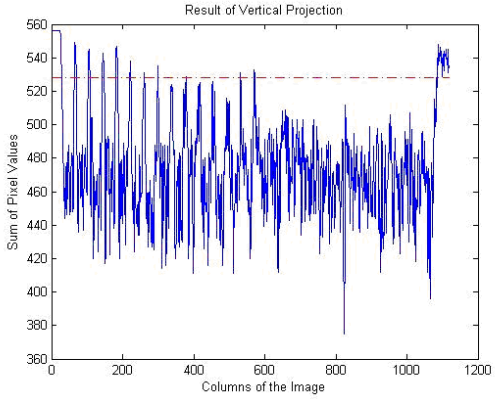
(2) If $C(j_1) \leq T_c$, set $C_s = j_1$ and go to (3); else, set $j_1 = j_1 + 1$ and repeat (2) until $j_1 = J$.

(3) If $C(j_2) \leq T_c$, set $C_e = j_2$ and go to (4); else, set $j_2 = j_2 - 1$ and repeat (3) until $j_2 = 1$.

(4) For j from 1 to C_s and from C_e to J, set $B_I(i,j) = 1$.



(a)



(b)

Fig. 4. (a) Horizontal projection result of Fig. 1(c). (b) Vertical projection result of Fig. 1(c)

Fig. 4(a) and (b) show the horizontal and vertical projection result of Fig. 1(c) respectively, and the threshold T_r and T_c with red dashed line. After the process depicted by Step 1 and Step 2, the image BP_I is generated from B_I and BP_I contains only a small amount of noise as shown in Fig. 1(d).

C. Median Filter Denoising

The last stage of our scheme is reducing the noise in the regions of characters with median filter. The median filter is working on binary images and is effective in eliminating isolated noise points [14,15]. The main idea of the median filter is to run through the image pixel by pixel, replacing each pixel with the median of neighboring pixels sorted numerically. The pattern of the neighbors is called the window, which slides, pixel by pixel, over the entire image. The output of 2D(two-dimensional) median filter can be obtained by:

$$g(i,j) = med\{f(i-k, j-l), (k,l \in W)\} \tag{8}$$

where $f(i,j)$ and $g(i,j)$ are the input image and output image respectively. W denotes the 2D template, which can be different shapes, and we use a square as a template in our scheme.

As discussed earlier, we set the median filter window size as 3×3 considering the recognition performance and computational cost. We finally obtain the image BM_I which contains only few noise points as shown in Fig. 1(e). BM_I is then sent to Tesseract engine in the experiments.

III. EXPERIMENTAL RESULTS AND DISCUSSIONS

In this section, we conduct the experiments to evaluate the performance of the proposed scheme. In details, we compare the Tesseract's OCR performance for shadowed text images of different size with and without the proposed preprocessing scheme. The shadows have different locations in the test images. Our program is written in C++ languages, with OpenCV 3.0 library installed. The Tesseract engine version adopted is 3.05.00dev.

The test images are segmented into binary images with the adaptive threshold processing. In this phase, the neighborhood size needs to be relatively large, and we set $p = 31$ in our experiments. The output images are projected horizontally and vertically to remove the noise between text lines and on the left and right non-text area. We set $T_r = 0.978 \times w$ and $T_c = 0.965 \times h$ according our experience at this stage, where w and h are the width and height of an original image. At last, these images are processed by median filter whose window size is set to 3×3. After these images are processed with three steps mentioned above, we input the relatively "clean" binary images into Tesseract engine to recognize the text information.

In order to show the superiority of the proposed scheme, the test images are also processed with AHE method and gamma correction in the experiments. The experimental results are shown in TABLE I. Fig. 5 displays the original test image T1 and the results processed with different schemes.

TABLE I
OCR ACCURACY COMPARISON

Test Images	No Preprocessing	AHE Method	Gamma Correction	Our Scheme
T1	78.2%	80.3%	81.6%	88.8%
T2	86.5%	86.5%	86.5%	94.3%
T3	86.9%	88.4%	89.6%	94.6%
T4	75.5%	73.2%	77.1%	84.2%
T5	92.0%	92.0%	92.7%	97.8%

In TABLE I, the OCR accuracy is given by

$$(u - \pi)/u \tag{9}$$

where u is the number of characters in a text image and π is the number of errors. From Fig. 5, we can draw conclusion that the AHE method does not take good effect on balancing illumination for text images, and the shadows are even more prominent. In addition, the gamma correction seems to work better in some degree, but it is far from an ideal solution because most shadows still exist. The experimental results in TABLE I show that our scheme can help improve Tesseract's OCR performance, with an average improvement of 8.1%.

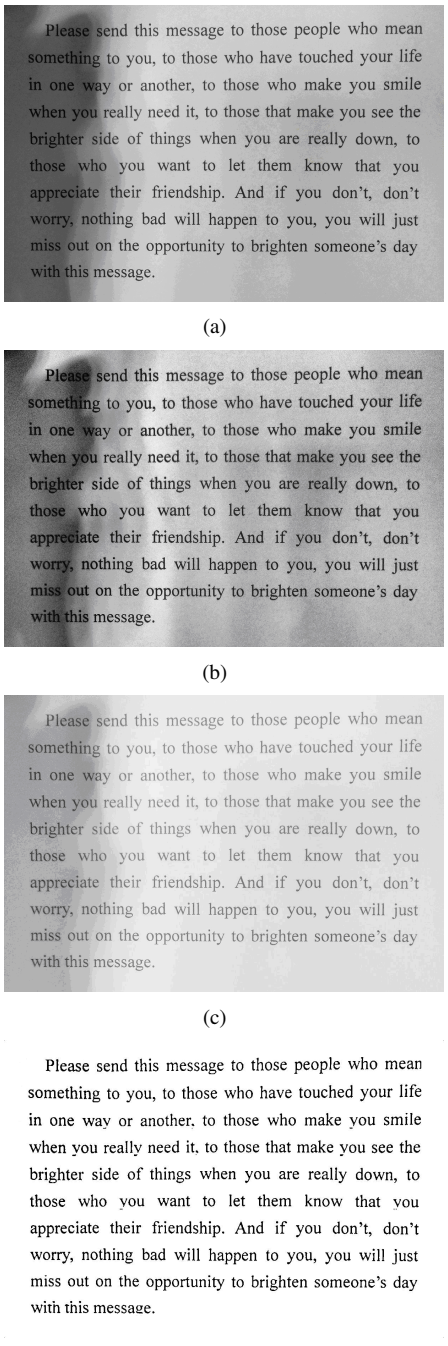


Fig. 5. Test image T1 and results of various schemes: (a) The original image; (b) Result with AHE; (c)Result with gamma correction ($\gamma = 0.40$); (d) Result with our scheme

IV. Conclusion

In view of Tesseract's weak OCR performance for shadowed text images, we have employed a novel technique to preprocess these images in order to obtain better performance. These images can be transformed into binary images with few noise points after using our scheme. According to the experimental results, the OCR accuracy can be increased. In our program, the values of and are important factors for preprocessing. In the future, we aim to research how to better set these coefficients.

Acknowledgment

This research is funded by the National Natural Science Foundation of China (NO.61375011) and the National Natural Science Foundation of Zhejiang province of China (LY13F030015).

References

[1] M. W. C. Reynaert, "Character confusion versus focus word-based correction of spelling and OCR variants in corpora." International Journal on Document Analysis & Recognition, vol.14, no.2, pp.173-187, 2011.
[2] R. Unnikrishna, and R. Smith. Combined script and page orientation estimation using the Tesseract OCR engine. ACM, 2009.
[3] R. Smith, "An Overview of the Tesseract OCR Engine." International Conference on Document Analysis and Recognition IEEE Computer Society, pp.629-633, 2007.
[4] J. Scott, and M. Pusateri, "Towards real-time hardware gamma correction for dynamic contrast enhancement." Applied Imagery Pattern Recognition Workshop IEEE, pp.1-5, 2009.
[5] J. Y. Kim, L. S. Kim, and S. H. Hwang, "An advanced contrast enhancement using partially overlapped sub-block histogram equalization." IEEE Transactions on Circuits & Systems for Video Technology, vol.11, no.4, pp.475-484, 2011.
[6] K. N. Chen, C. H. Chen, and C. C. Chang, "Efficient illumination compensation techniques for text images." Digital Signal Processing vol.22, no.5, pp.726-733, 2012.
[7] M. Sahani, S. K. Rout, L. M. Satpathy, and A. Patra, "Design of an embedded system with modified contrast limited adaptive histogram equalization technique for real-time image enhancement." International Conference on Communications and Signal Processing IEEE, pp.0332-0335, 2015.
[8] T. R. Singh, S. Roy, O. I. Singn, et al. "A New Local Adaptive Thresholding Technique in Binarization." International Journal of Computer Science Issues, vol.8, no.6, 2011.
[9] M. K. Kim, "Adaptive Thresholding Technique for Binarization of License Plate Images." Journal of the Optical Society of Korea, vol.14, no.4, pp.368-375, 2010.
[10] P. N. Druzhkov, V. L. Erukhimov, N. Y. Zolotykh, et al. "New object detection features in the OpenCV library." Pattern Recognition & Image Analysis, vol.21, no.3, pp.384-386, 2011.
[11] I. Culjak, D. Abram, T. Pribanic, H. Dzapo and M. Cifrek, "A brief introduction to OpenCV," 2012 Proceedings of the 35th International Convention MIPRO, Opatija, pp. 1725-1730, 2012.
[12] R. Smith, D. Antonova, and D. S. Lee. "Adapting the Tesseract open source OCR engine for multilingual OCR." Proceedings of the International Workshop on Multilingual OCR. ACM, 2009.
[13] F. F. Zeng, Y. Y. Gao, and X. L. Fu, "Application of denoising algorithm based on document image," Computer Engineering and Design, vol.33, pp.2701-2705, 2012.
[14] S. Esakkirajan, T. Veerakumar, A. N. Subramanyam, and C. H. Premchand, "Removal of High Density Salt and Pepper Noise Through Modified Decision Based Unsymmetric Trimmed Median Filter." IEEE Signal Processing Letters, vol.18, no.5, pp.287-290, 2011.
[15] M. Monajati, S. M. Fakhraie, and E. Kabir. "Approximate Arithmetic for Low-Power Image Median Filtering." Circuits Systems & Signal Processing, vol.34, no.10, pp.3191-3219, 2015.