Using Adaptive Threshold to Enhance Scanned Images
Algorithm Engineering 2022 Project Paper

Chengcheng Guo
Friedrich Schiller University Jena
Germany
chengcheng.guo@uni-jena.de

ABSTRACT
Scanned images may have noise due to uneven or poor lighting conditions. In this paper, I try different methods to enhance the scanned images. After comparing different methods. I found that the adaptive threshold method can best enhance the scanned image degradation caused by poor lighting conditions. I also used OpenMp to speed up the calculation process.

KEYWORDS
enhance scanned image, noise reduction, adaptive mean filter, contrast enhancement, histogram equalization

1 INTRODUCTION
1.1 Background
Scanned image reproduce paper document in an electronic, digital form. There are many benefits to having a digital version of document, such as being easier to carry around and view at any time, as well as being distributed via the Internet. Or we can use computer to process large volumes of digital documents to obtain useful information. Through scanned image enhancement, human vision can be improved. It can also prepare computers to process the image to obtain further information from the scanned images. Image processing is a method to perform some operations on an image, to get an enhanced image or to extract some useful information from it. The typical workflow in an image processing system is as follows:

Acquisition → Image preprocessing → Segmentation (for example, thresholding) → Feature extraction → Classification → Assertion.

But scanned documents can often have noise. The source of the noise may be uneven or too dark light when scanning, or poor quality paper, or signal interference in the scanning process. The purpose of this paper is to find the best and fastest way to enhance the scanned images.

1.2 Related Work
One of the methods to enhance background quality of gray scale images employs thresholding and binarization techniques. [1] Some resources divide thresholding techniques into two major groups. The methods in the first group use global algorithms which employ global image features to determine appropriate thresholds to divide image pixels into object or background classes. The second group uses local image information to calculate thresholds, similar to the locally adaptive thresholding method that uses neighborhood features such as the mean and standard deviation of pixels[3]

1.3 Our Contributions
This paper compares different methods of image enhancement and ends up with an executable portable and rapid code that can be used to enhance images.

1.4 Outline
This paper starts with introducing several image processing algorithms that can enhance contrast of scanned images and reduce noise. These methods are then experimented separately to evaluate whether they are effective in enhancing the scanned images. Then the most effective methods are accelerated. The purpose is to get the most effective and fastest method to enhance the scanned images.

2 THE ALGORITHM
2.1 Power-Law (Gamma) Transformations
The gamma correction algorithm improve the low contrast images. Gamma correction is, in the simplest cases, defined by the following power-law expression:

$$V_{out} = AV_{in}^{\gamma} \tag{1}$$

where the non-negative real input value V_{in} is raised to the power γ and multiplied by the constant A to get the output value V_{out}. Sometimes, a higher gamma makes the displayed image look better to viewers than the original because of an increase in contrast.[2]

2.2 Mean filter
To eliminate pixel values which are unrepresentative of their surroundings. We can use mean filter. The idea of mean filtering is to replace each pixel value in an image with the average of its neighbors, including itself. Mean filtering is usually implemented by convolutional operations. It is based around a kernel, which represents the shape and size of the neighborhood to be sampled when calculating the mean. Often a square kernel is used. In general, the arithmetic mean filters are well suited for random noise like Gaussian or uniform noise.[2]

2.3 Histogram linearisation
A significant mass of the histogram in a small interval of the range of intensity values means sub-optimal contrast. We can enhance the contrast with method of histogram equalization. This method usually increases the global contrast of images. Through this adjustment, the intensities can be better distributed on the histogram utilizing the full range of intensities evenly. This allows for areas of lower local contrast to gain a higher contrast. With histogram

linearisation, the quantisation characteristic is optimally adapted to the brightness values occurring in an image, i.e. areas with rare grey values are "moved closer together" in the histogram, areas with frequent grey values are stretched.

In order to linearise the histogram of an image, first cumulative the histogram:

$$h_c(I) = \sum_{i=0}^{I} h(i). \tag{2}$$

which gives the frequency of intensities below this gray value for each gray value I . Each pixel in the image with gray value I is then assigned a new gray value $I' = h_c(I)$ scaling the values of h_c to the range of gray values.

2.4 Binary thresholds

We can apply a thresholding operation to separate objects from the background in text analysis(OCR) and medical image segmentation. Usually, the thresholding methods binarize an initial image, two segments are formed, ideally separate the background and the searched objects. The assignment to the two segments (0 and 1) is made on the basis of a comparison of the gray value g of the pixel under consideration with the previously defined threshold value t . The resulting image can thus be calculated with very little computational effort, since only one simple comparison operation needs to be performed per pixel. The corresponding calculation rule of the image T_{global} is:

$$T_{\text{global}}(g) = \begin{cases} 0 & \text{falls } g < t \\ 1 & \text{falls } g \geq t \end{cases} \tag{3}$$

In the global thresholding method, a threshold value is selected globally for the entire image. The method is the easiest to calculate, but also very susceptible to brightness changes in the image.

2.5 Adaptive mean filter

In the previous method, we used one global value as a threshold. But this might not be good in all cases, e.g. if an image has different lighting conditions in different areas. In restauration, noise removal wants to consider the local image properties to optimally reduce noise such filter are called adaptive filter. Adaptive filters change their properties depending on the local image statistics in a neighbourhood S_{xy} around the current pixel at x, y. In that case, adaptive filter can help. Here, the algorithm determines the threshold for a pixel based on a small region around it. So we get different thresholds for different regions of the same image which gives better results for images with varying illumination. Here we apply an adaptive mean C filter, the threshold value T(x,y) is a mean of the **filter mask size × filter mask size** neighborhood of (x,y) minus C, where C is a constant.

This method follows the following steps:

(1) Convert original image to grayscale,the Gray conversation RGB image is defined as follows: [4]

$$(.) = 0.2989 \times r + O.5870 \times g + 0.1140 \times b \tag{4}$$

where r,g,b, are red, green and blue colors of the RGB image (.) correspondingly.

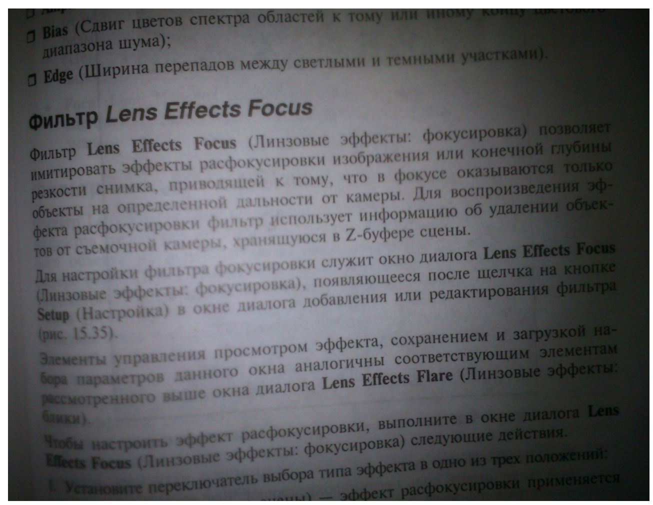
(2) Determine the parameters **filter mask size** and constant C, and **max value** in the result.
(3) Build the filter mask
(4) Zero padding the original image to ensure that the image does not get smaller after the convolution operation.
(5) Do the convolution operation with the filter mask and the image.
(6) • if : grayscale value of current pixel > T : result value in current pixel is **max value**
• else : result value in current pixel is **0**
where T is threshold, i.e. the result of the convolution operation minus **C**

2.6 Adaptive median filter

This method is similar to the adaptive mean filter, except that the filter mask is replaced with a median filter mask.

3 EXPERIMENTS

Figure 1 is the example of original scanned image that should be enhanced.



Bias (Сдвиг цветов спектра областей к тому или иному концу цветового диапазона шума);

Edge (Ширина перепадов между светлыми и темными участками).

Фильтр *Lens Effects Focus*

Фильтр **Lens Effects Focus** (Линзовые эффекты: фокусировка) позволяет имитировать эффекты расфокусировки изображения или конечной глубины резкости снимка, приводящей к тому, что в фокусе оказываются только объекты на определенной дальности от камеры. Для воспроизведения эффекта расфокусировки фильтр использует информацию об удалении объектов от съемочной камеры, хранящуюся в Z-буфере сцены.

Для настройки фильтра фокусировки служит окно диалога **Lens Effects Focus** (Линзовые эффекты: фокусировка), появляющееся после щелчка на кнопке **Setup** (Настройка) в окне диалога добавления или редактирования фильтра (рис. 15.35).

Элементы управления просмотром эффекта, сохранением и загрузкой набора параметров данного окна аналогичны соответствующим элементам рассмотренного выше окна диалога **Lens Effects Flare** (Линзовые эффекты: блики).

Чтобы настроить эффект расфокусировки, выполните в окне диалога **Lens Effects Focus** (Линзовые эффекты: фокусировка) следующие действия.

1. Установите переключатель выбора типа эффекта в одно из трех положений:

— эффект расфокусировки применяется

Figure 1

3.1 Power-Law (Gamma) Transformations

In Figure 2 we apply gamma correction with different value of γ on the scanned image. We can see when γ = 0.1 the shadows around the edges almost disappear, but the whole image looks brighter too, the text is not clear. When a larger gamma is selected, the text looks clearer, but the shadows around the edges are getting darker.

This method brightens or darkens all pixel points at the same time, it does not effectively enhance the scanned image.

3.2 Mean filter

Figure 3 shows the effect of a mean filter. After convolving the image with a mean filter the image does not change a lot, this method is not useful for our case.

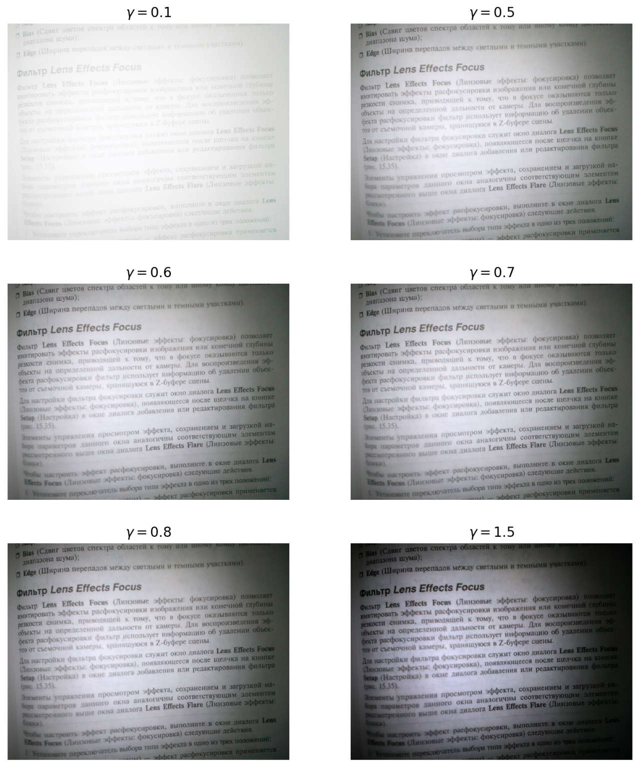


Figure 2: gamma correction on the scanned image with different γ

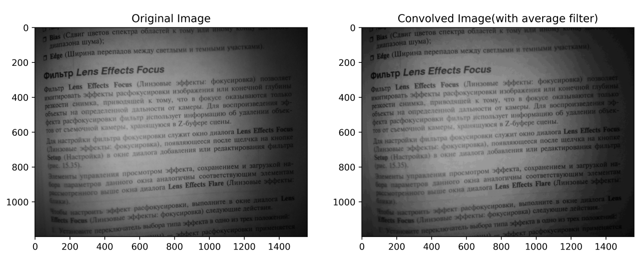


Figure 3: original scanned image compared with mean filter processed image

3.3 Histogram linearisation

Figure 4 shows the cumulative histogram before and after histogram linearisation. We notice that before histogram linearisation, the grayscale distribution of the images is not uniform. The overall picture looks greyish.(Figure 5 left)

After histogram linearisation, the image (Figure 5 right) seems brighter. The disadvantage is that the text in the center is not visible due to overexposure and the edges of the picture are even darker.

3.4 Binary thresholds

The previous methods all had a problem that the text didn't stand out enough. Besides the text (black) there was a lot of gray in the background. So next I tried the binary threshold.

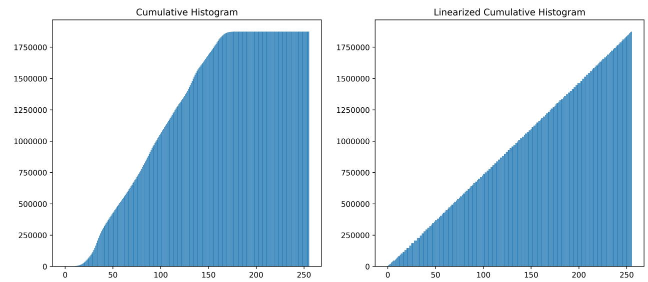


Figure 4: the cumulative histogram before and after histogram linearisation

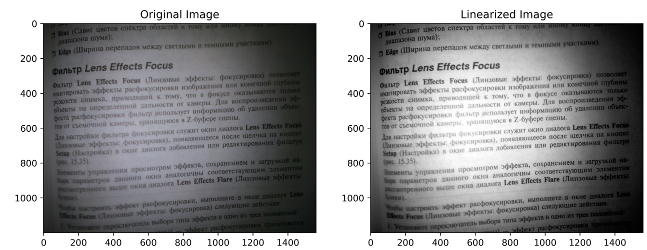


Figure 5: original scanned image compared with histogram linearisation processed image

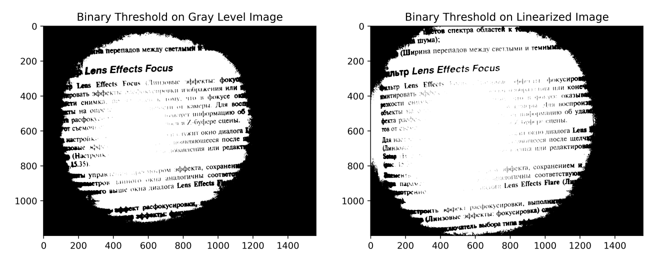


Figure 6: Binary thresholds method applied on original image and linearized image

Figure 6 shows the effect of binary thresholds applied on the original image and on the linearized image.

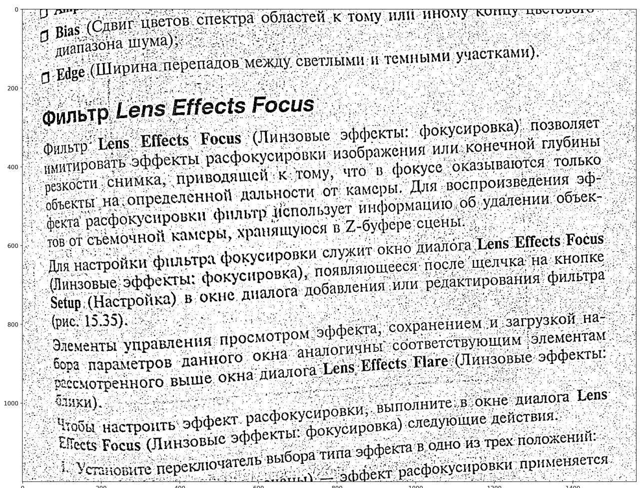
We can notice that although the image is not clear enough after histogram equalization, it has a larger range of visible text.

This method makes the text clearer, but since our scanned image lighting is stronger in the center and darker at the edges. So the text in the center is emphasized, but since the dark areas at the edges are as dark as the text in the center, the edges are all blacked out after binary processing.

Therefore we should split the image into smaller blocks and use a binary threshold segmentation in each block so that the different lighting at the edges and the center can be considered separately.

3.5 Adaptive mean filter

We try to use the threshold method with a neighborhood size of 11 and a C of 2. From Figure 7, we can notice a significant improvement in this method compared to the previous algorithms. Because in this algorithm the intensity is maximum greyscale when the current pixel value is greater than the neighborhood pixel intensity average minus C, and zero in all other cases. So there is no intermediate gray scale. The text and background are nicely separated. Only the background has obvious noise.


Bias (Сдвиг цветов спектра областей к тому или иному концу цветового
диапазона шума);
Edge (Ширина перепадов между светлыми и темными участками).

Фильтр Lens Effects Focus

Фильтр Lens Effects Focus (Линзовые эффекты: фокусировка) позволяет имитировать эффекты расфокусировки изображения или конечной глубины резкости снимка, приводящей к тому, что в фокусе оказываются только объекты на определенной дальности от камеры. Для воспроизведения эффекта расфокусировки фильтр использует информацию об удалении объектов от съемочной камеры, хранящуюся в Z-буфере сцены.

Для настройки фильтра фокусировки служит окно диалога Lens Effects Focus (Линзовые эффекты: фокусировка), появляющееся после щелчка на кнопке Setup (Настройка) в окне диалога добавления или редактирования фильтра (рис. 15.35).

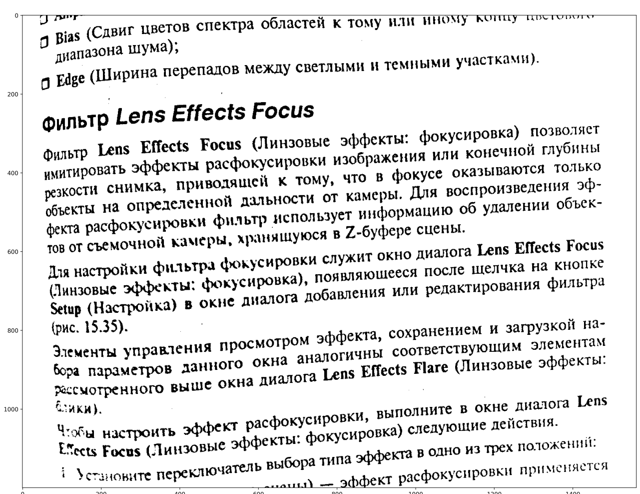
Элементы управления просмотром эффекта, сохранением и загрузкой набора параметров данного окна аналогичны соответствующим элементам рассмотренного выше окна диалога Lens Effects Flare (Линзовые эффекты: блики).

Чтобы настроить эффект расфокусировки, выполните в окне диалога Lens Effects Focus (Линзовые эффекты: фокусировка) следующие действия.

1. Установите переключатель выбора типа эффекта в одно из трех положений:
— эффект расфокусировки применяется

Figure 7: adaptive mean C filter (filter mask size = 11, C = 2)

By adjusting the parameters, we found that this approach can emphasize the scanned images very successfully. Figure 8 shows the effect of adaptive mean C method with **filter mask size** = 55, constant C = 8.


Bias (Сдвиг цветов спектра областей к тому или иному концу цветового
диапазона шума);
Edge (Ширина перепадов между светлыми и темными участками).

Фильтр Lens Effects Focus

Фильтр Lens Effects Focus (Линзовые эффекты: фокусировка) позволяет имитировать эффекты расфокусировки изображения или конечной глубины резкости снимка, приводящей к тому, что в фокусе оказываются только объекты на определенной дальности от камеры. Для воспроизведения эффекта расфокусировки фильтр использует информацию об удалении объектов от съемочной камеры, хранящуюся в Z-буфере сцены.

Для настройки фильтра фокусировки служит окно диалога Lens Effects Focus (Линзовые эффекты: фокусировка), появляющееся после щелчка на кнопке Setup (Настройка) в окне диалога добавления или редактирования фильтра (рис. 15.35).

Элементы управления просмотром эффекта, сохранением и загрузкой набора параметров данного окна аналогичны соответствующим элементам рассмотренного выше окна диалога Lens Effects Flare (Линзовые эффекты: блики).

Чтобы настроить эффект расфокусировки, выполните в окне диалога Lens Effects Focus (Линзовые эффекты: фокусировка) следующие действия.

1. Установите переключатель выбора типа эффекта в одно из трех положений:
— эффект расфокусировки применяется

Figure 8: adaptive mean C filter (filter mask size = 55, C = 8)

Next step, OpenMP is applied when generate the zero padding images, and then convolutional calculations to speedup the adaptive mean C function.

4 CONCLUSIONS

4.1 Best solution

From the analysis and the experiments, **adaptive mean filter** and **adaptive median filter** has the best enhancement effect. The binary threshold can also enhance the image, but it is not effective because of the uneven lighting of the original image.

4.2 About speed up

Table 1: Comparison of the running time of different method.

Language	Method	Running time (s)
C++	OpenMP	**0.083**
C++	–	0.783
Python	OpenCV	0.036
Python	–	52.7

As can be seen in table 1, the normal Python function is more than 60 times slower than the normal (unaccelerated) c++ function. The OpenMP accelerated C++ function is 9 times faster than the unaccelerated one. But the OpenMP accelerated C++ function is still 2.3 times slower than the OpenCV function in Python.

REFERENCES

[1] Atena Farahmand, Abdolhossein Sarrafzadeh, and Jamshid Shanbezadeh. 2013. Noise Removal and Binarization of Scanned Document Images. https://www.researchgate.net/publication/290530622_Document_Image_Noises_and_Removal_Methods
[2] Richard E. Woods Rafael C. Gonzalez. 2009. *Digital Image Processing.* Pearson Education.
[3] J. Sauvola and M. Pietikainen. 2000. *Adaptive document image binarization," Pattern Recognition.* Vol. 33 Issue 2, pp. 225-236.
[4] J. R. Mohanty Suresh Chandra Satapathy, Vikrant Bhateja and Siba K. Udgata. 2019. *Smart Intelligent Computing and Applications: Proceedings of the Third.* Springer.