Journal of Applied Science and Engineering, Vol. 19, No. 4, pp. 505–514 (2016) *DOI: 10.6180/jase.2016.19.4.14*

An Improved Brightness Balancing Method and its GPU Acceleration for Digital Images

Rui-Bin Zhao[1,3], Yan-Ling Zhang[2]*, Ming-Yong Pang[1] and Sheng-Hui Zhao[2]

[1]*Department of Education Technology, Nanjing Normal University, Nanjing, P.R. China*
[2]*School of Computer and Information Engineering, Chuzhou University, Chuzhou, P.R. China*
[3]*Anhui Center for Collaborative Innovation in Geographical Information Integration and Application, Chuzhou, P.R. China*

Abstract

Cloud and fog always lead to unbalance brightness in digital images, which limit information recognition and extraction. Based on an assumption that a cloud-fog covered image is an overlaying result of a normal ground image and a cloud-fog maskimage, this paper proposes an improved method for balancing brightness of digital image by removing cloud-fog effect through the following four steps: generating cloud-fog mask image, subtracting cloud-fog maskimage, choosing reference image, and locally adaptive enhancing. Additionally, in order to avoid time-consuming for large images, a parallel solution is introduced for accelerating the method based on graphics processing unit (GPU) acceleration. Finally, the method was tested by using different cloud-fog covered images, and the experiments verify that the method is effective at balancing brightness and its efficiency can be significantly improved through central processing unit and graphics processing unit (CPU-GPU) cooperative computing.

***Key Words*:** Digital Images, Brightness Balancing, Cloud-fog Removing, GPU Acceleration

1. Introduction

At present, more and more high-resolution digital images have been obtained, and widely used in many applications. However, due to the atmospheric environment and climate condition when capturing, theyare often contaminated by clouds or fog, and have uneven brightness [1]. As a result, the true ground information is difficult to recognize and extract, which strongly limits the use of the digital images [2]. Thus, balancing brightness to generate normal and high-quality images is an important issue.

Up to now, many methods have been proposed to solve this issue, which can be mainly classified into two categories: one is based on physical model, and another is based on image processing algorithm. The former usually remove cloud-fog effect and balance brightness according to physical models, which are constructed based on transmission and reflection principles of radiant solar energy, such as the models given in the literatures [3,4]. These methods are capable of generating a natural result. However, they often require extra information about real atmospheric and climate conditions to construct physical models, and the information is often unavailable. For this reason, some more feasible methods are proposed to balance brightness by employing image processing techniques, such as homomorphic filtering [5], Wallis transform [6], histogram transform [7], and Multi-Scale Fusion [8]. Among them,the homomorphic filtering and Wallis transformare the two most popular techniques owing to their relative simplicity and effectiveness.

*Corresponding author. E-mail: caughy@chzu.edu.cn

The main contributions of our work are summarized as followed: (1) present a new method for balancing brightness of digital images by combining multiple image processing algorithms; (2) introducea parallel solution for accelerating the proposed method by using a CPU-GPU cooperative computing model. The remainder of the paper is organized as follows. Section 2 describes the theory and main steps of the method. Section 3 describes the design and implementation of the parallel solution. Section 4 provides some experiments toverify the removing effectiveness and the accelerating performance. Finally, section 5 gives a conclusion.

2. The Proposed Method

In this paper, a cloud-fog covered image is considered as a blending result by overlaying a cloud-fog mask image on a normal ground image, and the cloud-fog mask image is assigned to represent the cloud-fog effects in the original image. Based on this, the proposed method balance an image through the following steps: (1) Generate the cloud-fog mask image through low-pass filtering of the original image; (2) Subtract the cloud-fog mask image from the original image and obtain a temporarily removed image; (3) Choose a proper reference image from the original image; (4) Enhance the temporarily removed image according to the reference image. The overview flowchart is shown in the Figure 1.

2.1 Generating the Cloud-fog Mask Image

As the main factors for uneven brightness of digital image, clouds and fog are complex atmospheric phenomena, and both of them are consisted of massive droplets and crystals suspended in the atmosphere. Usually, they cover a larger area in the sky, and their density change slowly and continuously, as shown in the original image presented in the Figure 2(a). Based on frequency analysis theory of image, it is obvious that cloud-fog effects are mainly contained in low frequency of an original image. So, the cloud-fog mask image of an original image can be obtained by low-pass filtering in frequency domain as the following formula:

$$I_{mask} = IFFT2D(FFT2D(I_{orig}) \times H) \quad (1)$$

where, the symbols I_{orig} and I_{mask} represent an original image and its cloud-fog mask image individually, $FFT2D$ represents two-dimensional fast Fourier transforms, H represents the Butterworth low-pass filter, and $IFFT2D$ represents two-dimensional inverse Fourier transforms. Firstly, the I_{orig} is transformed into its frequency domain by $FFT2D$. Then, its high-frequency component is filtered out and low-frequency component is reserved by applying the H. Finally, the filtered image is re-transformed into special domain, and is considered as the I_{mask}. Furthermore, the H can be given by:

$$H(u,v) = \frac{1}{1+[D(u,v)/D_0]^{2n}} \quad (2)$$

where, (u, v) represents one frequency in 2D frequency domain, $D(u, v) = \sqrt{u^2 + v^2}$ represents the distance from (u, v) to the origin, D_0 is the user-given cutoff frequency, and n is the order of the filter. For the original image shown in Figure 2(a), its cloud-fog mask image is given in Figure 2(b), which is obtained with the cutoff frequency $D_0 = 0.05$.

2.2 Subtracting the Cloud-fog Mask Image

As mentioned above, a cloud-fog covered image is considered as an overlaying result of a cloud-fog mask image and a normal ground image. So, the normal ground image can be obtained by subtracting the I_{mask} from the I_{orig} as the following formula (3). The Figure 3 presents an original image and its subtracted image. It is obvious

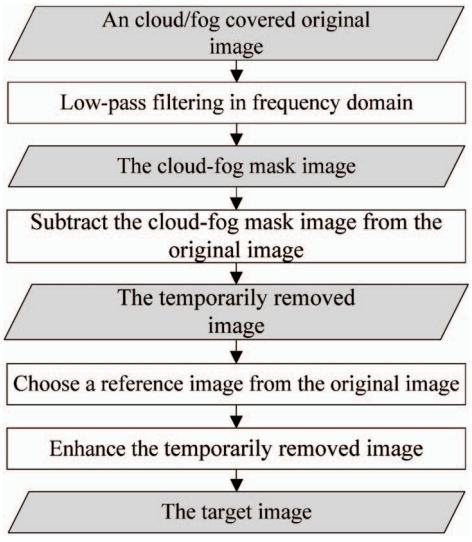
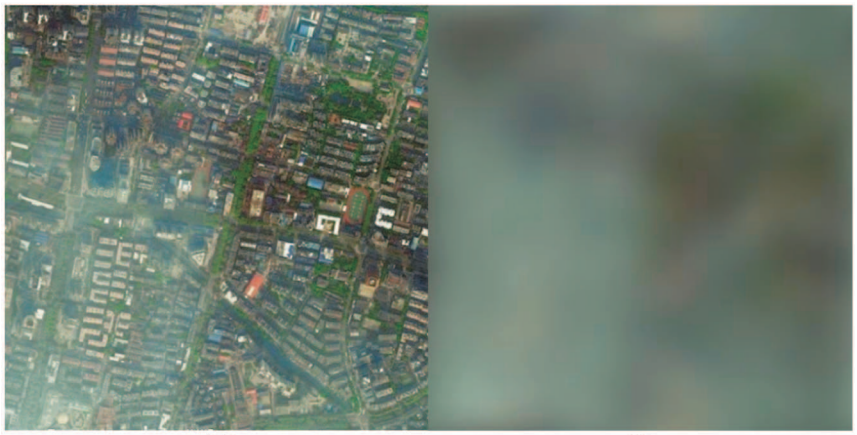


Figure 1. The flowchart of the proposed method.



(a) (b)

Figure 2. An original image (a) and its mask image (b).

(a) (b)

Figure 3. An original image (a) and its subtracted image (b).

that the main cloud-fog effects have been eliminated. However, due to low-pass filtering in frequency domain and subtracting the cloud-fog mask image, the obtained image often has low contrast, and the normal color has a little bias, which can be found by comparing the original image and its subtracted image. For the reason, the subtracted image just considered as a temporarily removed image, and marked as I_{remv}, which is needed to further enhance in this paper.

$$I_{remv}(x,y) = I_{orig}(x,y) - I_{mask}(x,y) + \bar{b} + offset \quad (3)$$

where, $I_{orig}(x, y)$, $I_{mask}(x, y)$ and $I_{remv}(x, y)$ represent the brightness values at the pixel (x, y) in the I_{orig}, I_{mask}, and I_{remv} individually, the symbol $\bar{b}$ represents the mean brightness value of all pixels in I_{orig}, and the symbol *offset* is a constant integer, which is used to adjust the overall brightness of the I_{remv}, a larger value will result in a brighter image, and vice versa.

2.3 Choosing a Reference Image

As described in the next section, the I_{remv} is needed to enhance by adjusting its mean brightness and standard deviation according to a reference image. So, a reference image, marked as I_{ref}, is required to choose firstly. In this paper, it is automatically chosen by extracting a sub-block from the I_{orig} according to the two criteria: (a) not covered by cloud or fog; (b) has a similar color histogram with the I_{orig}. After dividing the I_{orig} and the I_{remv} into many equal sized sub-blocks as shown in the Figure 4, we can find that the cloud-fog covered blocks in the I_{orig} often have a bigger mean brightness value than the correspondent sub-blocks in the I_{remv}, such as the sub-blocks

A and A', which can be easily explained that white cloud and fog certainly raise the brightness of their covered sub-blocks. So, for the criteria (a), the chosen sub-block should have a smaller mean brightness value than the correspondent sub-block in the I_{remv}. For the criteria (b), the chosen sub-block should have a bigger standard deviation, which is because a bigger standard deviation means that the sub-block has a more similar color histogram with the I_{orig}. Specifically, the I_{ref} is chosen through the following three steps:

(1) Divide the I_{orig} into many equal sized sub-blocks, marked as $\{B_i\}$. In the same way, the I_{remv} is also divided into many equal sized sub-blocks, and marked as $\{B_i'\}$.

(2) Calculate the mean brightness values of each B_i and B_i', marked as $\overline{b_l}$ and $\overline{b_l'}$ individually, and select sub-blocks from $\{B_i\}$, which satisfy the constraint condition $\overline{b_l} > \overline{b_l'}$, as the candidate sub-blocks.

(3) Calculate the standard deviation of each candidate sub-block B_i, and select the sub-block with the most standard deviation as the final reference image I_{ref}.

For the subtracted image shown in the Figure 4(c), the sub-block B of the original image is finally selected to be the reference image, which is shown in the Figure 4(b).

2.4 Locally Adaptive Enhancing

In this step, the proposed method further enhances the contrast and optimizes the brightness of the I_{remv} according to the I_{ref}. Specifically, for each pixel P of the I_{remv}, a $n \times n$ square neighborhood is defined firstly. Then, the target image I_{targ} is generated according to the locally adaptive filter as the following formula:

$$I_{targ}(x,y) = [I_{remv}(x,y) - m_g]\frac{cS_f}{cS_g + (1-c)S_f} + bm_f + (1-b)m_g \tag{4}$$

where, $I_{targ}(x, y)$ is the new brightness value at the pixel (x, y) of the I_{targ}, $I_{remv}(x, y)$ is the brightness value at the pixel (x, y) of the I_{remv}, m_g and S_g are the mean and standard deviation of the square neighborhood of the pixel (x, y), m_f and S_f are the mean and standard deviation of the I_{ref}, and $c \in [0,1]$ and $b \in [0,1]$ are two balancing factors.

Through the above adaptive filtering, the mean and standard deviation of the temporarily removed image are adjusted to be similar to the reference image. For the original image shown in the Figure 5(a), its temporarily removed image and final target image is given in the Figure 5(b) and (c). By comparing the temporarily removed image and the final target image, we can find that the latter has a closer color with the original image. Furthermore, in order to evaluate the effectiveness of locally adaptive enhancing quantitatively, the cloud-fog free sub-image, marked by the black frames in the Figure 5(a), is chosen from the original image for analyzing its brightness change during the removing process. In Figure 6, we show the RGB histograms of the sub-image in there different stages.

By comparing the RGB histograms shown in Figure 6(a) and (b), we can see that the contrast of the sub-image became lower after removing cloud-fog effects, and which is unavoidable due to the low-pass filtering in

Figure 4. The divided original image (a), the selected reference image (b), and the subtracted image (c).

the section 2.1. From the RGB histograms shown in Figure 6(c), we can see that the contrast of the sub-image is enhanced, and the histogram curves are similar with the Figure 6(a), which means the locally adaptive enhancing method is and effective and suitable.

3. Parallel Solution to the Proposed Method

As described in the above section 2, the proposed method balancing brightness through four steps and each step also includes multiple computing tasks. Unfortunately, some of the tasks are complex and time-consuming, such as *FFT2D* and *IFFT2D* in the step 1, and the locally adaptive enhancing in the step 4. For this reason, this paper introduced a CPU-GPU cooperative computing model for accelerating the method based on the CUDA.

3.1 GPU and CUDA

Traditionally, a graphics processing unit, or GPU for short, is used primarily to renders images, animations and video for the computer screen. However, with the rapid increase in the performance and improvements in programm ability in recent years [9], it has been using as a compelling platform for computationally demanding tasks in a wide variety of application domains [10,11], which is known as the general-purpose computation on GPU (GPGPU).

For performing general-purpose computation by employing the hundreds of processing cores of a GPU, NVIDIA developed a parallel computing platform and programming model, which is named the Compute Unified Device Architecture or CUDA for short [12]. Based on CUDA, an algorithm can be written as one or more CUDA kernel function, and each of them can be parallel executed by a number of threads on GPU.

In order to manage and schedule the multiple threads conveniently, CUDA organizes them into a hierarchy of grids of blocks as shown in Figure 7. Each block is composed by a group of threads, and all threads in the same block can share data through the shared local memory and synchronize their execution for coordinating accesses

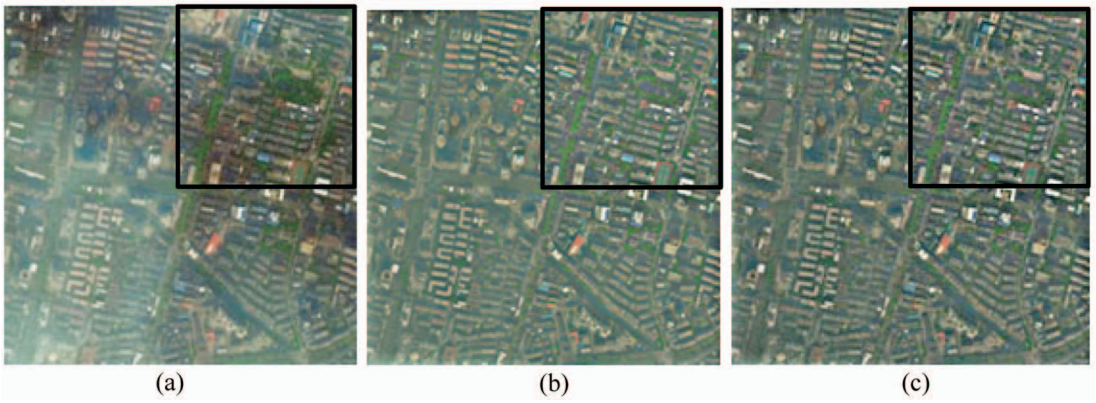


Figure 5. The cloud-fog covered original image (a), the temporarily removed image obtained (b), and the final target image (c), which is generated by locally adaptive enhancing according to the formula (4) with c = 1 and b = 1.

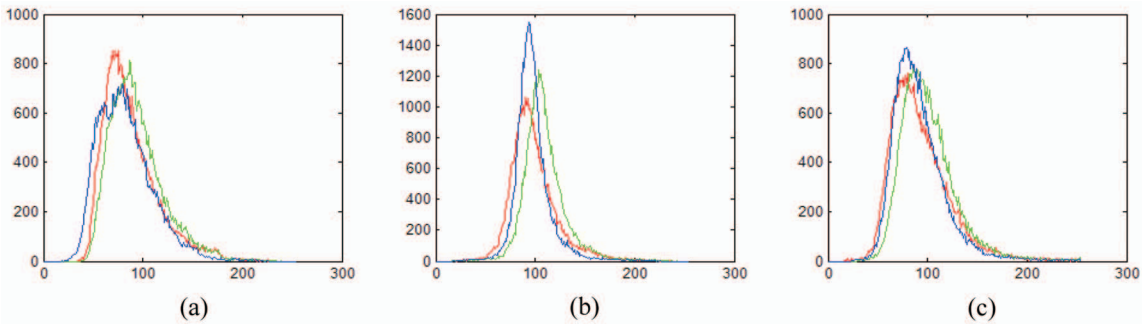


Figure 6. The RGB histograms of the original sub-image (a), the RGB histograms of the sub-image after removing cloud-fog effects (b), and the RGB histograms of the sub-image after locally adaptive enhancing (c).

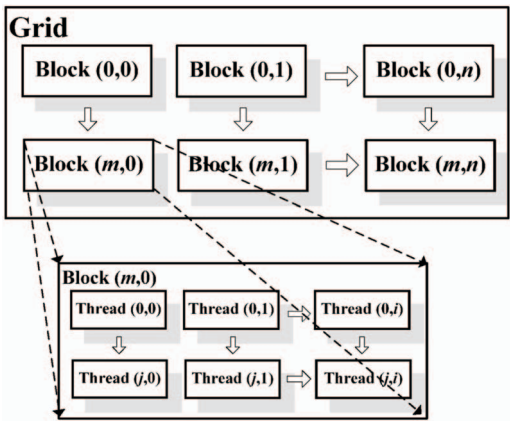


Figure7. Multiply threads and their organization in CUDA.

to memory. Each grid is a set of thread blocks, and the thread blocks in the same grid are applied to independently execute a same kernel in parallel. Furthermore, both of the block and grid can be organized as a 1D, 2D, or 3D array.

3.2 CPU-GPU Cooperative Computing Model

Firstly, the computation process of the proposed method is newly sectioned into six independent tasks, and each of them is written as a CUDA kernel. The detail description of them as following:

(1) Resize_kernel: it resizes the I_{orig} from its original size (w_0, h_0) to a new size (w_1, h_1) by using the bilinear interpolation algorithm. Before the Fourier transform, the I_{orig} is required to resize, and make its width and height are a power of two, such as (2028, 2048).

(2) FFT2D_kernel: it represents the fast Fourier transform, and is used to transform the resized I_{orig} into in the frequency domain. Its output is a (w_1, h_1) 2D array, and each element represents a particular frequency contained in the I_{orig} with a complex number.

(3) Filter_kernel: it represents the Butterworth low-pass filter, and is used to filter out high-frequency component and reserve low-frequency component through multiplying each frequency by the transfer function given in the formula (2).

(4) IFFT2D_kernel: it presents the inverse fast Fourier transform, and is used to re-transform the filtered I_{orig} into the spatial domain, and generate the cloud-fog mask image I_{mask}.

(5) Subtract_kernel: it is used to subtract the I_{mask} from the I_{orig} according to the formula (3), and generate the temporary removed image I_{remv}.

(6) Choose_kernel: it is used to choose the reference image from the I_{orig} according to the method described in the section 2.3.

(7) Enhance_kernel: it used to generate the target image I_{targ} through enhancing the contrast and optimizing the brightness of the I_{remv} according to the formula (4).

Then, a CPU-GPU cooperative computing model is proposed and implemented as shown in Figure 8. In this model, CPU is mainly responsible for controlling the execution flow and launching the CUDA kernels, and GPU is referred to as the co-processor and responsible for parallel executing each kernel with a large number of hardware threads. The flow-control algorithm run on CPU is described as follows.

Algorithm 1. The flow-control algorithm run on CPU

1: Read the I_{orig}, and define the new width and height (w_1, h_1) according to the original (w_0, h_0);
2: Allocate buffers in GPU global memory for saving the I_{orig}, I_{mask}, I_{targ}, and other temporary data;
3. Transfer the image data of the I_{orig} to GPU global memory;
4. Initialize the GPU, and define the CUDA parameters *blocksize* and *gridsize* for each kernel.
5. Launch the kernels in the order as given in Figure 4.
6. Free the allocated buffers in the GPU global memory.
7. Output and save the target image.

Before a kernel is executed, two important CUDA parameters, *blocksize* and *gridsize*, are required to specify firstly. The *blocksize* determines the number of threads in each block, and the *gridsize* determines the number of block in each grid. In this paper, the *blocksize* is specified experimentally by a special parameter $M \times M$, where M is required to be a power of eight, such as 16, 32 and 64. The *blocksize* is specified by (w/M, h/M), where w and h are the width and height of the processed image.

In the computing process, each kernel is executed by the all threads in a same grid, and the threads are distinguished from each other by the unique coordinates *blockId* and *threadId*, which are assigned automatically

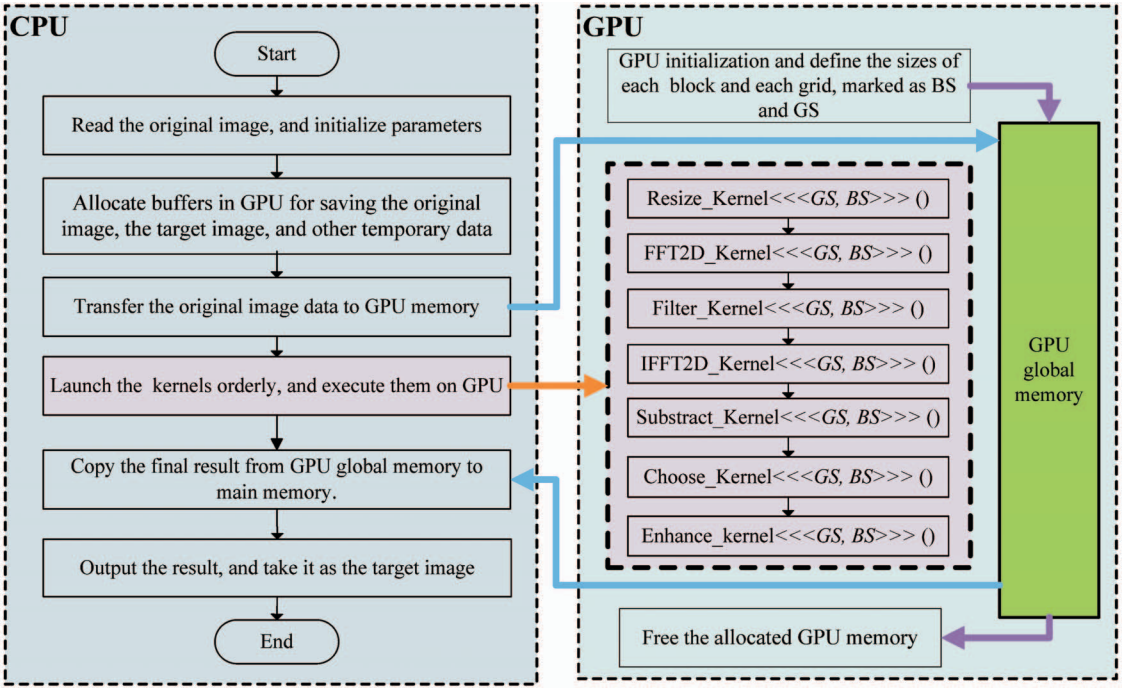


Figure 8. The CPU-GPU cooperative computing model of the proposed method.

by the CUDA runtime system. Meanwhile, each thread identifies the appropriate pixel of the image to process through the *blockId* and *threadId*. Based on above model, each kernel can be designed as the following, where (M, M) is the size of each block, (w_0, h_0) is the size of the image. Due to the similarity of the kernels in design, only the detail algorithm description of the Enhance_kernel is given in this paper.

Algorithm 2. Enhance_kernel
Dim2 blocksize (M, M);
Dim2 gridsize $(w_0/M, h_0/M)$;
__global__ void Enhance_kernel<<<gridsize, blocksize>>>( )
{

1. Specify the pixel (x, y) needed to process by this thread according to the blockIdx and threadIdx;
   $x \leftarrow$ blockIdx.x * blockDim.x + threadIdx.x
   $y \leftarrow$ blockIdx.y * blockDim.y + threadIdx.y
2. Create a neighborhood of the pixel (x, y);
3. Calculate the mean and standard deviation of the neighborhood;
4. Calculate the object value of the pixel (x, y) according to the formula (4);
5. Save the object result in the GPU global memory.

}

4. Experimental Results and Analysis

In this section, some experiments are presented to verify the effectiveness of the proposed method. On the one hand, the brightness balancing result of the proposed mothed is compared with other existing methods. On the other hand, the performance of GPU Acceleration is evaluated.

4.1 Balancing Result Analysis

In this paper, the brightness balancing result of the proposed method is compared with two other traditional balancing methods: the homomorphic filtering method and the Wallis transform based method. For the same original image shown in the Figure 9(a), balancing results of the homomorphic filtering based method, Wallis transform based method and our proposed method are given as following.

For the homomorphic filtering based method, the Figure 9(b) shows that it can remove cloud-fog effect efficiently, and generates a global balance result. How-

ever, it is not difficult to find that an obvious color bias is caused by comparing the result image (b) and the original image (a). For the Wallis transform based method, the Figure 9(c) shows that some block artifacts exist in the result image while it makes the result image with a relatively even distribution of brightness. From the Figure 9(d), we can find our method not only achieves a global brightness balance, but also has a closer color with the original image.

Furthermore, some other images, which include different levels or types cloud-fog effects, were tested and listed as the follows in Figure 10. Their results also indicated the effectiveness of the proposed method.

4.2 GPU-accelerating Experiments

To evaluate accelerating effectiveness, two solutions for this paper method were implemented. One of them is called GPU-accelerated solution, which is implemented according to the proposed CPU-GPU cooperative computing model by using C++ language and NVIDIA's CUDA 5.5 platform, and it is accelerated by a 1 GB NVIDIA GeForce 9800 GT graphics card. Another is called CPU-only solution, which is also implemented by using C++ language, but is only executed by the CPU. Both of the solutions have been tested in a same computer with Processor Inter Quad-Core i7 3820QM running at 2.70 GHz with 16 GB of RAM and the Windows operating system.

The timeconsumingof the two solutions are shown in Figure 11(a) and (b). In order to test the computation speed, eight images in different scales, ranging from 64 × 64 (3 bands, 12 KB) to 8192 × 8192 (3 bands, 192 MB), are tested. It can be clearly seen that in Figure 11(a) and (b), when the data size is 192 KB or smaller, the time consuming of the GPU-accelerated solution is more than the CPU-only solution. Because while handling the small data, the time consuming on image data transfer between CPU and GPU takes a greater proportion in the whole process, which including transferring data between CPU and GPU and processing data on GPU.

For the same data, the accelerating rate is defined as the time consumption of the CPU-only solution divided

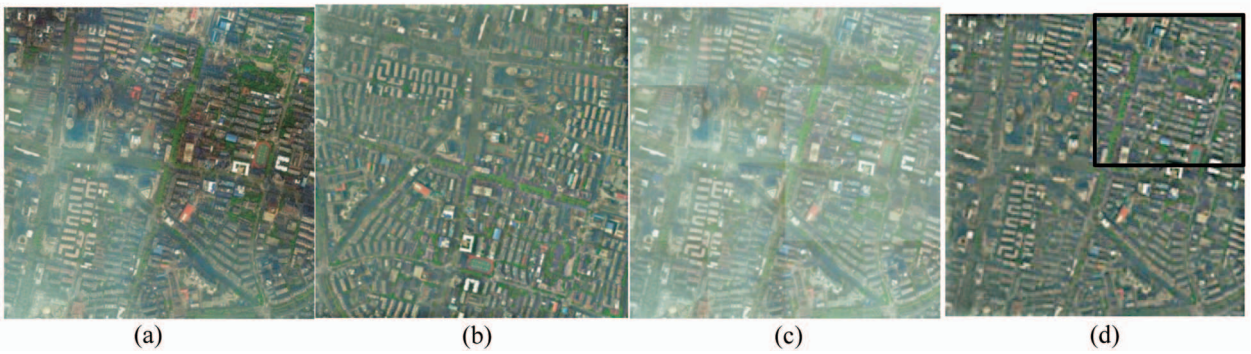


Figure 9. Brightness balancing results: (a) cloud-fog covered original image; (b) balancing result of the homomorphic filtering based method; (c) balancing result of the Wallis transform based method; (d) balancing result of the proposed method.

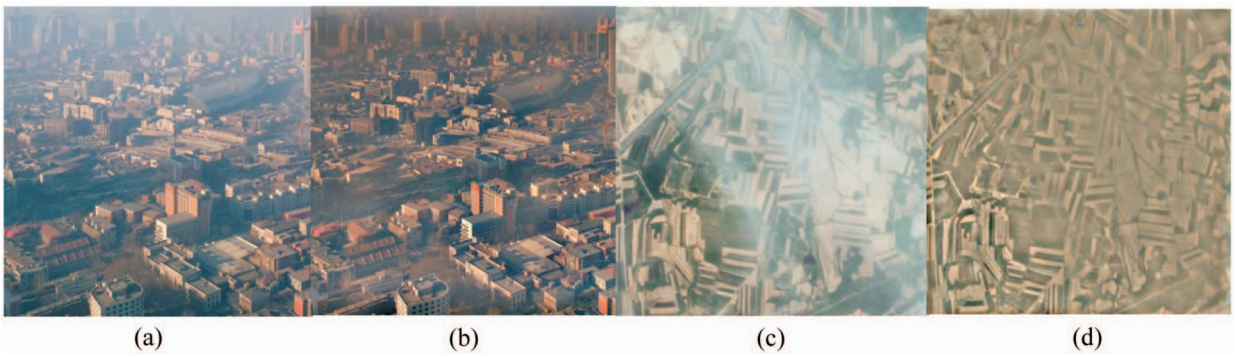


Figure 10. An original image (a) and its balancing results (b), and another original image (c) and its balancing results (d).

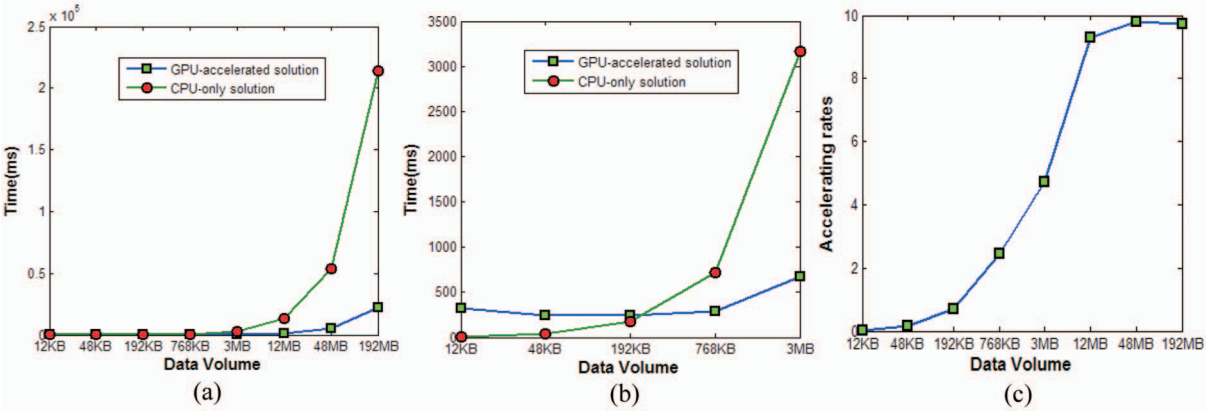


Figure 11. (a) Time consuming curves of the GPU-accelerated solution and the CPU-only solution; (b) Partial enlarged view of the time consumption curves shown in (a), and (c) Accelerating rate of the proposed method.

by the time consumption of the GPU-accelerated solution. The accelerating rates are shown in Figure 11(c). It can be seen that the greater the volume of data, the more the bigger the accelerating rate. When the image is very small, more time is spent on data transmission between GPU and CPU than data processing in the GPU. As a result, the GPU-accelerated solution consumes more time than the CPU-only solution, and the accelerating rate is less than 1. With an increase in data processing, the accelerating rate increases, and finally, it stabilizes.

5. Conclusions

In this paper, we propose a method for balancing brightness of digital images, which are mainly caused by cloud-fog effects. Meanwhile, for the time-consuming shortcoming of the method, a CPU-GPU cooperative computing model is introduced to accelerate the method. Some experiments have verified that not only a satisfactory brightness result can be obtained, but also the computation time can be decreased obviously. Nevertheless, some work remains to be done.

(1) Three important parameters, including the cutoff frequency, the neighborhood size, and the CUDA blocksize, are required to specify for the proposed method. In this paper, they are specified experimentally. Therefore, much work can be done for specifying them automatically.

(2) As CUDA supports parallel processing on multiple GPUs, the experiment can be implemented in GPU clusters for further efficiency improvement.

Acknowledgements

This work was supported by the Natural Science Foundation of Educational Commission of Anhui Province of China (Grant no. KJ2014A184), the Key Project of Anhui Center for Collaborative Innovation in Geographical Information Integration and Application (Grant no. 201116Z01), the Natural Science Foundation of Anhui Province of China (Grant no. 1408085MF126), and the Planning Project of Chuzhou University (Grant no. 2015GH21).

References

[1] Tseng, D. C., Tseng, H. T. and Chien, C. L., "Automatic Cloud Removal from Multi-temporal SPOT Images," *Applied Mathematics and Computation*, Vol. 205, No. 2, pp. 584–600 (2008). doi: 10.1016/j.amc.2008.05.050

[2] Cheng, Q., Shen, H., Zhang, L. and Zeng, C., "Cloud Removal for Remotely Sensed Images by Similar Pixel Replacement Guided with a Spatio-temporal MRF Model," *ISPRS Journal of Photogrammetry and Remote Sensing*, Vol. 92, No. 2, pp. 54–68 (2014). doi: 10.1016/j.isprsjprs.2014.02.015

[3] Oakley, J. P. and Satherley, B. L., "Improving Image Quality in Poor Visibility Conditions Using a Physical Model for Contrast Degradation," *IEEE Transactions on Image Processing*, Vol. 7, No. 2, pp. 167–179 (1998). doi: 10.1109/83.660994

[4] Sun, W., "A New Single-image fog Removal Algo-

rithm Based on Physical Model," *Optik-International Journal for Light and Electron Optics*, Vol. 124, No. 21, pp. 4770–4775 (2013). doi: 10.1016/j.ijleo.2013.01.097

[5] Xiao, L., Li, C., Wu, Z. and Wang, T., "An Enhancement Method for X-ray Image via Fuzzy Noise Removal and Homomorphic Filtering," *Neurocomputing*, Vol. 195, No. 24, pp. 56–64 (2016). doi: 10.1016/j.neucom.2015.08.113

[6] Cao, B., Zhu, B., Li, R., Yang, X. and Mo, D., "Wallis Algorithm for Single Image Dodging," *Journal of Geomatics Science and Technology*, Vol. 29, No. 5, pp. 373–377 (2012) (Chinese).

[7] Raju, A., Dwarakish, G. S. and Reddy, D. V., "A Comparative Analysis of Histogram Equalization based Techniques for Contrast Enhancement and Brightness Preserving," *International Journal of Signal Processing, Image Processing and Pattern Recognition*, Vol. 6, No. 5, pp. 353–366 (2013). doi: 10.14257/ijsip.2013.6.5.31

[8] Ancuti, C. O. and Ancuti, C., "Single Image Dehazing by Multi-scale Fusion," *IEEE Transactions on Image Processing*, Vol. 22, No. 8, pp. 3271–3282 (2013). doi: 10.1109/TIP.2013.2262284

[9] Owens, J. D., Luebke, D., Govindaraju, N., et al., "A Survey of General-Purpose Computation on Graphics Hardware," *Computer Graphics Forum*, Vol. 26, No. 1, pp. 80–113 (2007). doi: 10.1111/j.1467-8659.2007.01012.x

[10] Werff, H. M. A. and Bakker, W. H., "Implementation and Performance of a General Purpose Graphics Processing Unit in Hyperspectral Image Analysis," *International Journal of Applied Earth Observation and Geoinformation*, Vol. 26, No. 1, pp. 312–321 (2014). doi: 10.1016/j.jag.2013.08.009

[11] Kim, S. K. and Kim, Y. J., "GPGPU-Perf: Efficient, Interval-based DVFS Algorithm for Mobile GPGPU Applications," *Visual Computer*, Vol. 31, No. 6, pp. 1045–1054 (2015). doi: 10.1007/s00371-015-1111-1

[12] Che, S., Boyer, M., Meng, J., et al., "A Performance Study of General-purpose Applications on Graphics Processors Using CUDA," *Journal of Parallel and Distributed Computing*, Vol. 68, No. 10, pp. 1370–1380 (2008). doi: 10.1016/j.jpdc.2008.05.014

Manuscript Received: Jan. 26, 2016
Accepted: Jul. 13, 2016