Make Your Picture More Attractive – An Interactive System for Image Blur

Qingwei Wang

Abstract—**Image blurring is a very attractive feature of photography. Due to hardware constraint, it is nearly impossible for compact DC to produce such image with blurred background. In this paper, we propose a system to simulate background blurring function, which consists of two main steps. In foreground extraction step, firstly, we select meanshift based segmentation method to get over-segmented regions. Then we extract both texture and color features for each region. Furthermore, based on input stroke information, the foreground centroids are clustered and all regions are classified into foreground and background considering both feature value and coordinate value. In background blurring step, to make the simulation result more natural, we calculate the distance of each pixels in the background to the foreground based on distance transform and give different blurring effect according to the distance. The experimental results show that the system can produce stratifying result and gain relatively high subjective evaluation.**

Index Terms—**Image segmentation, foreground object extraction, distance transform, image blur.**

I. Introduction

One attractive feature of photography is the emphasis of the main theme against other areas of the picture by using a small depth of focus. An object or a face can spectacularly stand out against a blurred, homogeneous background. One of the reasons that the digital single lens reflex (DSLR) camera is so popular even with much more expensive price than compact camera, is that it can generate fantastic background blurring effect. Owners of compact cameras might have already encountered the problem that, in the majority of their photos, everything is sharp. A small depth of focus and the blurring of the background are very difficult to achieve. Such machines have a very small sensor with a small focal distance lens, and therefore, a large depth of focus. Even with a larger camera, you might feel that a photo you have taken needs its background blurred. So recently, some camera manufacturers have released some cameras which can simulate background blur functions, such as Panasonic, Casio, etc. Some professional image processing softwares also provide background blur function. Nowadays, smart phone and tablet computer are more and more popular and a new era of human computer interface (HCI) is started with the usage of multi touch screen. This new way of interaction makes it easy and friendly to manipulate the tablet computer to achieve some goals. In this paper, we propose a method for user to implement the background blur function for some photos just by simple interaction.

Manuscript received July 29, 2012; revised September 4, 2012.

Qingwei Wang is with the State Intellectual Property Office of PRC, China (e-mail: wqwszc@sohu.com).

II. Related Wroks

Researchers have made a lot of studies on the mechanisms of human visual system and found that it only selects some visual inputs by considering "salient regions", i.e. the most noticeable parts of scenes[1],[2] . So background blurring is to simulate the mechanism of human visual system to some extents. In some cases, image blurring is disadvantageous for application system. So there are a lot of works which are to estimate the image blurring extent by using different models. [3]–[5], and to remove the blurring effect in order to improve the accuracy of application system [6]–[8]. However, as aforementioned, from the point of view of photography, sometimes background blurring plays a very important role. The DSLR camera is so popular and attractive even with high price and heavy weight compared to compact camera, mainly because it can generate great blurred background. Some image processing software also provide such functions [9], but they need complicated operation and the users should have plentiful experience. In our system, with the user input of several strokes, the foreground will be extracted and kept while the background will be blurred.

In this system, the key step is to extract the foreground. Foreground extraction is widely researched for various purposes [10]-[13]. In [10], the author uses an iterative method which iteratively performs foreground extraction and 3D reconstruction in a manner similar to an EM algorithm on regions segmented in an initial stage. In [11], traditional GMM method is used to build the foreground/background model and improve its performance by introducing foreground pixel region growing method to check whether the foreground is changed. However, these traditional methods usually use continuous frames for model building. But in our system there is only one frame image. Considering this constraint, we use an interactive procedure which is similar to [13]. But we use different segmentation method and propose new method for foreground region growing based on the input stroke information.

To make the background blurring result more realistic, after foreground extraction, we use distance transform method to segment the background into several layers according to the distance to the foreground part.

III. System Structure

The system structure is shown in Fig.1. In this system, segmentation is an important initial processing step that aims at finding the foreground object. Input images are assumed to be natural photographic images, which are generally rich in

color and texture information. Because there is no apriori knowledge available on the image contents and composition, useful information is introduced by users with some input strokes. It is an interactive process which can improve the performance of the system. Because the system has some apriori knowledge by user stroke input, moderate over-segmentation is acceptable. The oversegmented data could be refined by using the stroke information. Once the foreground is extracted, all the pixels in foreground are set 0 and one distance transform algorithm is performed to compute the relative distance to the foreground of background pixels. For the pixels with different distance to the foreground we use different blurring filters to guarantee that the far pixel to the foreground has more blurry effect than the near pixel.

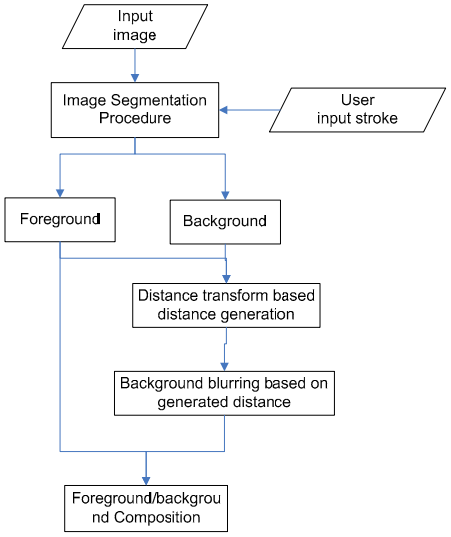


Fig. 1. Background blurring system structure.

A. Image Segmentation

This system is probably used in DC or other similar embedded system which has limited processing ability, the algorithm should be as simple as possible.

From 1965, more than 4500 articles related to image segmentation have been published [14]. Recently, a few approaches to segmentation stand out of the vast literature dedicated to this subject, such as efficient Graph-based (EGB) [15], JSEG [16], and EDISON[17], etc. Through their innovative and successful approaches, and the availability of source code to support their assertions, they have been extensively studied, and their major flaws have been dealt with in order to improve their results. After experiment, in our method MeanShift segmentation algorithm is chosen because it has merits like edge preserving and noise resisting.

An image is typically represented as a two-dimensional lattice of p-dimensional vectors (pixels). The space of the lattice is known as the spatial domain, while the gray level, color, or spectral information is represented in the range domain. In application of MeanShift algorithm, the location and the range vectors are concatenated in a joint spatial-range domain of dimension d. So for a gray image, $d = 1 + 2$, and for a color image, vector dimension , $d = 3 + 2$.

Define a symmetric kernel, usually we use normal kernel:

$$K_N(x) = (2\pi)^{-d/2} \exp(-\frac{1}{2}\|x\|^2) \qquad (1)$$

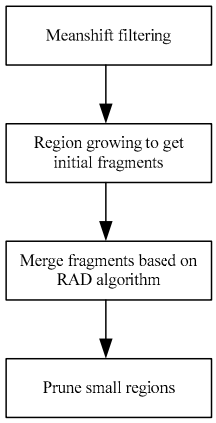


Fig. 2. Procedure of Meanshift based segmentation method.

In the application of image segmentation, a multi-variation kernel is defined:

$$K_{hs,hr}(x) = \frac{C}{hs^2 hr^p} k\left(\left\|\frac{x^s}{hs}\right\|^2\right) k\left(\left\|\frac{x^r}{hr}\right\|^2\right) \qquad (2)$$

where p is the dimension of the range domain (1 for gray image and 3 for color image).

The procedure of MeanShift segmentation is shown in Fig.2:

i. Meanshift based filtering

Let $\mathbf{x}_i$ and $\mathbf{z}_i$, $i = 1,...,n$ be the d-dimensional input and filtered image pixels in the joint spatial-range domain. For each pixel,

1) Initialize $j = 1$ and $\mathbf{y}_{i,1} = \mathbf{x}_i$.
2) Compute $\mathbf{y}_{i,j+1}$ according to the following equation until convergence, $\mathbf{y} = \mathbf{y}_{i,c}$:

$$\mathbf{y}_{j+1} = \frac{\sum_{i=1}^{n} \mathbf{x}_i g\left(\left\|\frac{\mathbf{x}-\mathbf{x}_i}{h}\right\|^2\right)}{\sum_{i=1}^{n} g\left(\left\|\frac{\mathbf{x}-\mathbf{x}_i}{h}\right\|^2\right)} \qquad (3)$$

3) Assign $\mathbf{z}_i = (\mathbf{x}_i^S, \mathbf{y}_{i,c}^r)$.

ii. Segmentation by region growing

The second step is region growing on the filtered image $\mathbf{z}$. If two adjacent pixels have the same color, they should be merged into one region. After this step, the original image will be divided into many small and fragmentary pieces for later processing.

iii. Merging fragments by RAD

The third step is to combine those pieces into regions with bigger sizes. An initial Region Adjacency Graph (RAG) is built for fragments generated at the second step and then it is used for graph contraction. During contraction, if the color difference between two adjacent nodes is lower than threshold T, these two nodes will be combined into one. This operation is conducted iteratively until convergence. Then we can get a segmentation result where regions are with acceptable size.

iv. Pruning small regions

In this step, a threshold for region size should be set and regions smaller than this threshold will be allocated to the nearest neighbor in the color space.

After all above steps, the original image is segmented into

several small segments. One example is shown in Fig. 3.

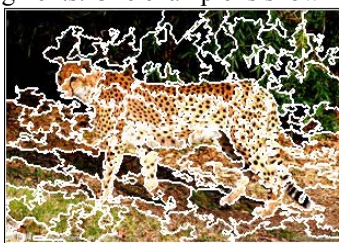


Fig. 3. Segmentation result using Meanshift.

B. Interactive Region Growing

As aforementioned, the problem of EGB is over segmentation of the object. To compensate the shortcoming, we use user input information to refine the foreground area. As shown in Fig.4, the red dot line represents the foreground area that the user wants to keep. What should be pointed out is that the user should include as many representative areas as possible and not across any background area.

Fig. 4. User input stroke example.

As hinted by several psycho visual studies, segmentation benefits from the combination of a number of features representing intensity, color, texture and edge information. Exploiting all available information has proved a successful strategy before, such as combining edge and color information (EDISON [17]), or color and texture (JSEG [16] and adaptive perceptual color segmentation [19]).

To grow the region obtained by segmentation step, as shown in Fig.3. We extract the following features for each region:

1) **Color**: the color information on the whole image is quantized into a set of 10-30 values which is thought enough to represent one image [16].

2) **Texture**: Local Binary Patterns (LBP) are chosen for their efficiency in representing texture information, and most important their high computational efficiency.

Using these two features, two histograms are built for every region: one for Local Binary Patterns and one with color clusters. Regions is relatively small in our target images (as little as 80 pixels per region on a 320x240 pixels image), so we have to carefully reduce the number of histogram bins, in order to avoid scarcity of data that would disturb the distance measures. Finally, every region is characterized by 2 normalized histograms, C for color and T for texture, of size (number of bins) B_c and B_t. The *b-th* bin of $C(T)$ is denoted $C_b(T_b)$, and satisfied with the following constraint,

$$\sum_1^{B_c} C_b = \sum_1^{T_c} T_b = 1 \tag{4}$$

In order to evaluate the distance between a pair of regions, Bhattacharyya distance is used as a metric between individual histograms. Given two histograms, H_1 and H_2, the distance is shown in formula (5). The higher the $d_{bhattachayya}(H_1, H_2)$ value, the more difference the two histograms.

$$d_{bhattachayya}(H_1, H_2) = \sqrt{1 - \sum_i \frac{\sqrt{H_1(i) \cdot H2(i)}}{\sum_i H_1(i) \cdot \sum_i H_2(i)}} \tag{5}$$

After extracting the features for each region, the region growing algorithm is as below,

1) Given all regions R we will firstly pick out all regions labeled by user stroke as R_f.

2) Use K-means method to generate several centroids R_c by using R_f.

3) Compute the difference between each region with R_c by considering both feature value and coordinate value. The similarity is computed using

$$D(R(i), R_c(j)) = \min(w_f \cdot d(R(i), R_c(j)) + w_d C(R(i), R_c(j))) \tag{6}$$

where $C(R(i), R_c(j))$ is normalized coordinate distance, the two diagonal vertexes distance is set 1. w_f and w_d is the weight for feature difference and distance difference. By comparing the region with all foreground centroids, we select the minimal value as its similarity. The smaller the value is, the more similar to foreground the region is.

4) Post process the growing result and remove some holes inside the foreground area using morphological method to get final result.

C. Background Blurring

In above step, we extract the foreground object. Because we lost the real distance information with using only one picture, to make the background blurring effect more vivid and make the foreground boundary area natural, we use distance transform method [20] to segment the background into several layers.

The distance transform of an image is defined as a new image in which every output pixel is set to a value equal to the distance to the nearest zero pixel in the input image. The distance from the foreground to the background can be explained in Fig.5.

4.5	4	3.5	3	3.5	4	4.5
4	3	2.5	2	2.5	3	4
3.5	2.5	1.5	1	1.5	2.5	3.5
3	2	1	0	1	2	3
3.5	2.5	1.5	1	1.5	2.5	3.5
4	3	2.5	2	2.5	3	4
4.5	4	3.5	3	3.5	4	4.5

Fig. 5. Distance from the center pixel (red circle) to neighbor pixels

By the distance transform we can calculate the distance to the foreground for each pixel in the background.

When we think of blurring we usually think of the Gaussian Blur filter. But the blur picture obtained from it tends to look fake. In our system, we generate different blurring effects for the pixels in background based on its real distance to foreground. The far from the foreground, the more blurred effect is generated. By this way, we can make the background blurring effect more realistic.

IV. Experimental Result

In our system, the main two parts are foreground object

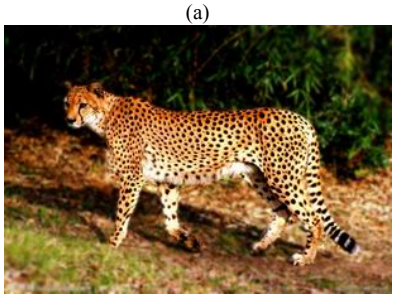
extraction and background blurring. Firstly, we select a batch of 200 pictures taken from both the Berkeley Segmentation dataset [18] and other natural images we took by camera, and resize them to 320x240. We run this program using the PC (Pentium IV 3.2 GHz, RAM 4Gb) on the test images and the results are shown in Table I. From this result, we can found that our method can achieve high processing speed which is very important for real-time application on hand-held devices.

TABLE 1: THE PROCESSING SPEED OF FOREGROUND EXTRACTION

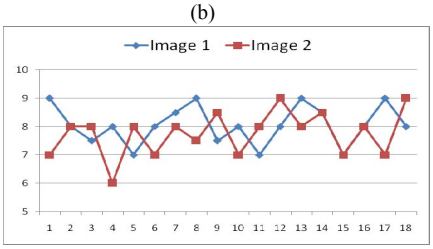
Algorithms	JSEG [16]	EDISON [17]	Our method
Processing time	2.3s	2.1s	1.1s

Because there is no quantitative metric to evaluate the effect of background blurring, we evaluate the system using subjective method. Firstly, we use the system to generate about 30 samples and invite 18 evaluation members to evaluate the effect. Each member picks out 15 samples randomly and gives the subjective score which is from 1 to 10, and the higher value means the better performance. One sample and their evaluation score are shown in Fig. 6,

(a)



(b)



(c)

Fig. 6. One example result of background blurring simulation result, (a) is original image, (b) is our system result, (c) subjective evaluation results.

The subjective evaluation result for these two images is shown in Fig.6(c). The results show that our system achieves satisfying simulation result. For all the test samples, the average evaluation score is 7.6 and the standard deviation is about 1.8. So we can conclude our system achieves good output and it is potential to be used in handhold device.

REFERENCES

[1] L. Zhaoping, "Theoretical understanding of the early visual processes by data compression and data selection," *Network: computation in neural systems*, 2006, vol. 17, pp.301–334.

[2] R. A. Khan, E. Dinet, and H. Konik, "Visual attention: effects of blur," in *Proc of 18th IEEE International Conference on Image Processing*, 2011, pp.3289-3291

[3] F. Zhang, G. Wang, J. Ye, and Q. Liu，"A new approach to estimate image blur extent based on wavelet module maximum," in *Proc of Intelligent Computing and Integrated Systems*, 2010 , pp. 193 – 196

[4] M. J. Chen and A. C. Bovik, "No-reference image blur assessment using multi-scale gradient," *EURASIP Journal on Image and Video Processing*, 2011.

[5] Y. Xu and G. Crebbin, "Image blur identification by using higher order statistic techniques," in *Proc of International Conference on Image Processing*, 1996, pp. 77-80.

[6] R. C. Luo, H. Potlapalli, and D. W. Hislop, "Defocusing blur restoration in natural scene images for fractal analysis," in *Proc of International Conference on Industrial Electronics, Control, and Instrumentation*, 1993, pp1377-1381.

[7] J. B. Huang and M. H. Yang, "Single image deblurring with adaptive dictionary learning," in *Proc of 17th IEEE International Conference on Image Processing*," 2010, pp. 1169-1172.

[8] Liang Wang, Siwei Luo, Zhe Wang, "Image deblur with regularized backward heat diffusion" in Proc of *17th IEEE International Conference on Image Processing",* 2010, pp. 1141-1144.

[9] Theresa Hernandez, Blurring the Background in Photoshop, http://www.digiscrappingtutorials.com/Tutorials/PSElements/Blur-Background-PSE3.pdf

[10] J. Kim, A. Park, and K. Jung, "Segment-based foreground extraction dedicated to 3D reconstruction," in *Proc of 20th IEEE International Conference on Pattern Recognition*," 2010, pp. 3587-3590.

[11] H. C. Zeng and S. H. Lai, "Adaptive foreground object extraction for real-time video surveillance with lighting variations," in *Proc of IEEE International Conference on Acoustics, Speech and Signal Processing*, 2007, pp.I.1201-I.1204.

[12] Y. S. Huang and F. H. Cheng, "Object-oriented foreground image extraction," in *Proc of IEEE International Conference on Innovative Computing, Information and Control*, 2007, pp.407.

[13] Z. Tang and Z. Miao, "Interactive foreground extraction for photo and video editing," in *Proc of IEEE International Conference on Signal Processing*, 2008, pp.1483-1486.

[14] Y. J. Zhang. "Advances in Image and Video Segmentation," Irm Press, 2006.

[15] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *International Journal of Computer Vision*, vol. 59, no. 2, pp.167–181, 2004

[16] Y. Deng and B. S. Manjunath, "Unsupervised segmentation of color-textureregions in images and video," *IEEE Transactions on Pattern Analysis and Machine Learning*, 2001, pp. 23, pp. 800–810.

[17] C. M. Christoudias, B. Georgescu, and P. Meer, "Synergism in low level vision," In *Proc of the 16th International Conference on Pattern Recognition*, 2002.

[18] D. Martin, C. Fowlkes, D. Tal, and J. Malik. "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," In *Proc of 8th International Conference on Computer Vision*, 2001, pp.416–423.

[19] J. Chen, T. N. Pappas, A. Mojsilovic, and E. Bernice "Adaptive perceptual color-texture image segmentation," *IEEE transactions on Image Processing*, 2005, pp.1524-1536.

[20] B. Borgefors, "Distance transformations in digital images," *Computer Vision, Graphics and Image Processing*, vol. 34, 1986, pp. 344-371.