Gradient Domain Image Blending and Implementation on Mobile Devices

Yingen Xiong and Kari Pulli

Nokia Research Center, Palo Alto, CA 94304, USA
firstname.lastname@nokia.com

Abstract. This paper presents an image blending approach which combines optimal seam finding and transition smoothing for merging a set of aligned source images into a composite panoramic image seamlessly. In this approach, graph cut optimization is used for finding optimal seams in overlapping areas of the source images to create a composite image. If the seams in the composite image are still visible, a gradient domain transition smoothing operation is used to reduce color differences between the source images to make them invisible. In the transition smoothing operation, a new gradient vector field is created using the gradients of source images and the seam information. A new composite image can be recovered from the new gradient vector field by solving a Poisson equation with boundary conditions.

Our approach presents several advantages. The use of graph cut optimization over the source images guarantees that optimal seams are found. The gradient domain transition smoothing operation allows smoothing out color differences globally and further improves image quality after merging with graph cut optimization. The final composite image is a global optimal solution. The approach is implemented in two ways called sequential image blending and global image blending. Sequential image blending allows us to use little memory in the whole blending process, which is very important for mobile devices. Global image blending guarantees a globally optimal solution. Experimental and application results in creating mobile image mosaics and mobile panorama are also given.

1 Introduction

As computational capabilities and memory of mobile devices increase and the camera quality of mobile devices improves, mobile image processing and mobile computational photography become increasingly interesting. Many applications which could only work on computer before can now be implemented on mobile devices, including mobile augmented reality [1,2], mobile image matching and recognition [3], and so on. Here, we are interested in creating high quality image mosaics and panoramic images on a mobile device. User can take an image sequence of a wide range of scenes with a mobile phone. Almost immediately after the capture the user can see a panoramic image on the device and share it with friends.

S. Ystad et al. (Eds.): MobiCASE 2009, LNICST 35, pp. 293–306, 2010.
© Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering 2010

Image blending is the final and a key step in creating high quality image mosaics and panoramic images. A simple copying and pasting of overlapping areas of source images usually produces visible artificial edges in the seam between images, due to differences in camera response (white balance, vignetting, etc.) and scene illumination, or due to geometrical alignment errors. Image blending can hide the seams and reduce color differences between source images.

In our mobile panorama system, two kinds of image blending algorithms are used. A fast and low quality blending algorithm is used to produce a quick result, so that user can preview it as soon as the image sequence is captured. If the user is satisfied with the image sequence, a more refined blending approach is used to produce a high-quality panoramic image. The fine blending approach is typically slower than the simple one, but its blending quality is much higher. Here we focus on the latter one.

In this work, we present a gradient domain image blending approach and its implementation on mobile devices to produce high-quality panoramic images. The approach combines processes of optimal seam finding and transition smoothing, so that it can make full use of their advantages and avoid their disadvantages. We use graph cut optimization to find optimal seams and gradient domain blending for smoothing the transitions. With graph cut optimization, we can create labeling for all pixels and find optimal seams in the overlapping areas of source images. A composite image can be created by copying pixels from corresponding source images with labeling information. As each pixel in the composite image comes only from one source image, the algorithm avoids most blurring and ghosting problems. If all seams in the composite image are invisible, then we use it as our final result. Otherwise, with gradient domain blending, we can reduce the color differences between source images and hide the seams. In the gradient domain image blending, a new gradient vector field is created by copying the gradient values from the constituent images. The gradients across seams are set to zero for smoothing color differences. A new composite image can be recovered from the gradient vector field by solving a Poisson equation with boundary conditions. The composite image is a globally optimal solution, where the color transition is smoothed over the whole image.

1.1 Related Work

Existing methods for image stitching in the literature can be categorized into two main classes[4,5]: optimal seam finding and transition smoothing. Each class has its own advantages and disadvantages.

Optimal seam finding algorithms [4,6,7,8,9] search for a seam in the overlapping area where the differences between source images are minimal. All pixels of the composite image are labeled based on the seams so that for each output pixel there is a unique input pixel. The composite image is produced by copying corresponding pixels from the source images using labeling information. This kind of approach works well when the source images are similar enough. However, when they are too dissimilar for the algorithms to find ideal seams, artifacts may still be created.

Transition smoothing algorithms reduce the color differences between source images for hiding the seam. Alpha blending [10] is one of the simplest and fastest transition smoothing approaches. It uses weighted combination to create the composite image. The main disadvantage of alpha blending is that moving objects will cause ghosting and small registration errors can cause blurring of high frequency details. In order to solve this problem, Burt and Adelson [11] presented a multi-band blending algorithm. They blend low frequencies over a large spatial range, and high frequencies over a short spatial range. Recently, gradient domain image blending approaches [9,5,12,13,14,15,16] have been applied to image stitching and editing. A new gradient vector field is created with the gradients of source images and a new composite image can be recovered from the gradient vector field by solving a Poisson equation. This kinds of algorithms can adjust the color differences due to illumination changes and variations in camera gains for the composite image globally. It can produce high quality composite images. However, the memory and computational cost is also high.

1.2 Organization of the Paper

In Section 2, we introduce the work flow of our approach. The details of the gradient domain image blending approach are described in Section 3. An implementation of the approach on mobile devices is explained in Section 4. Applications and result analysis are discussed in Section 5. A summary of the paper is given in Section 6.

2 Summary of Our Approach

The approach can be divided into two parts, optimal seam finding and transition smoothing processes. Figure 1 shows the work flow of the approach. The input source images are already aligned and warped. We need to stitch them together seamlessly to create a composite image.

The first part of our approach is to find optimal seams in the overlapping areas of the source images. An objective function is created by combining two items: a data property of a pixel and the color differences between corresponding pixels in the overlapping areas. The objective function is minimized to obtain optimal seams by graph cut optimization. Using the results, we create labeling for all pixels. The composite image is created by copying corresponding pixels from source images according to labeling information. If all seams in the composite image are invisible, we use it as our final result. Otherwise, further processing in the next part is needed.

The second part of our approach is to reduce the differences of the source images to make the seams minimally visible. We perform gradient domain blending to smooth the differences. A gradient vector field is created by copying gradient values from the constituent source images with labeling information. We set the gradient values across seams to zero for smoothing color transition. A divergence vector $divG$ is created from the gradient vector field and is used as a guidance

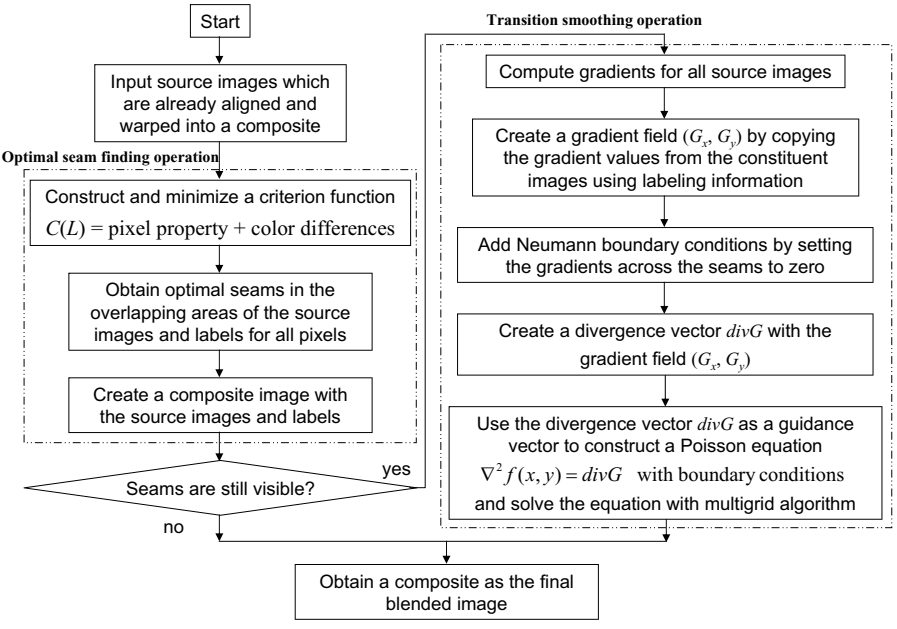


Fig. 1. A work flow of the gradient domain image blending approach

vector to construct a Poisson equation $\nabla^2 f = divG$. A new composite image can be recovered from the gradient vector field by solving the Poisson equation with Neumann boundary conditions. Since the discrete Poisson equation is an over-constrained linear system, we solve for the least-squares optimal results. In order to speed up the performance, we employ a multigrid algorithm for the linear solver.

3 Gradient Domain Image Blending

We use graph cut optimization to find optimal seams and gradient domain image blending to reduce color differences between source images for hiding visible seams.

3.1 Labeling with Graph Cut Optimization

As Figure 2 shows, we assume that the source images $S_0, S_1, \ldots, S_n$ are already registered and warped into a composite image I_b. We apply graph cut optimization to find optimal seams $m_0, m_1, ..., m_{n-1}$, using which a mapping or labeling between pixels in image I_b and the source images can be created. With the labeling, we can copy corresponding pixels from the source images to image I_b.

Considering implementation on mobile devices, we create a simple and efficient objective function which includes two items: pixel property P_p and color

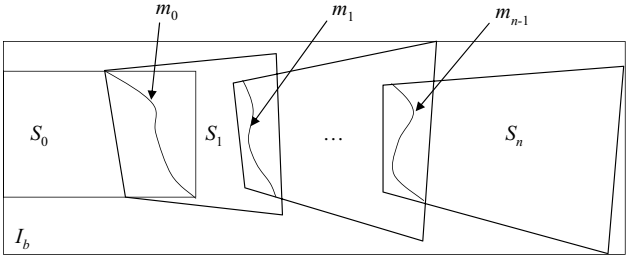


Fig. 2. Optimal seam finding with graph cut optimization

differences D_p between corresponding pixels, both of which depend on pixel labeling L.

$$O(L) = \sum_i P_p(i, L(i)) + \sum_{i,j} D_p(i, j, L(i), L(j)) \tag{1}$$

where

$P_p(i, L(i))$ depends on the property of pixel i;
$D_p(i, j, L(i), L(j))$ is the color difference.

The pixel property item, $P_p(i, L(i))$ is set it to a very large number when the pixel is invalid, which means that the seam is not allowed to go to invalid areas. Otherwise, we set it to zero, i.e.,

$$P_p(i, L(i)) = \begin{cases} N & \forall i \in \Phi \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

where

N is a large number;
Φ is an invalid area.

The invalid areas are created in warping and interpolation processes after registration. Figure 3 shows an example. The black regions shown in (a) and (b) are invalid areas. The seam shown in (c) between the two source images is not allowed to go into these areas.

The color differences D_p are defined by the Euclidean distances in RGB space of all pairs of neighboring pixels i, j, i.e.,

$$D_p(i, j, L(i), L(j)) = \|S_{L(i)}(i) - S_{L(j)}(i)\| + \|S_{L(i)}(j) - S_{L(j)}(j)\| \tag{3}$$

where

S_k is source image k;
$L(i)$ is the label of pixel i.

We could add other items into the objective function to consider other properties when cutting the source images, such as an item to consider edge information,

(a) (b) (c)

Fig. 3. An example of graph cut results with source images

and so on, but it would require more computation and memory, and this definition has been sufficient.

The optimal seam finding or labeling problem can be cast as a binary graph cut problem. We apply a max-flow approach [17] to solve it. The approach uses "alpha expansion" to minimize the objective function and obtain a globally optimal solution.

3.2 Transition Smoothing in the Gradient Domain

When source images to be stitched are too dissimilar for the optimal seam finding process to find ideal seams, the seams remain visible in the composite image. Figure 3 (c) shows an example. In this case, transition smoothing is needed to reduce the differences between the source images for hiding the seams.

We apply gradient domain transition smoothing to improve blending quality. In this operation, we create a color gradient vector field (G_x, G_y) by copying the color gradients of corresponding pixels in source images using the same labels which are created by optimal seam finding process. With the color gradient vector field, a divergence vector $div(G)$ can be computed. Suppose $f(x, y)$ is the composite image. We use the divergence vector as a guidance vector to construct a Poisson equation

$$\nabla^2 f(x, y) = div(G) \tag{4}$$

where

∇^2 is the Laplacian operator such that $\nabla^2 f(x, y) = \frac{\partial^2 f(x,y)}{\partial x^2} + \frac{\partial^2 f(x,y)}{\partial y^2}$;
$div(G)$ is the divergence vector field with $div(G) = \frac{\partial G_x}{\partial x} + \frac{\partial G_y}{\partial y}$.

Equation 4 is a linear partial differential equation. In order to solve this equation, we must first specify boundary conditions. In our case, we use the Neumann boundary conditions,

$$\nabla f \cdot \overrightarrow{n} = 0, \tag{5}$$

where

∇f is the gradient of $f(x, y)$;
$\overrightarrow{n}$ is the normal on the boundary.

Equation 5 tells us that the gradient in the direction normal to the boundary is zero. The discrete form will be used in actual implementation.

In the color gradient vector field (G_x, G_y), we set gradient values across seams to zero for smoothing color transition between two source images and gradient values on the boundaries to zero for fitting the boundary conditions.

Finally, a new composite image f_b can be recovered from the color gradient vector field (G_x, G_y) by solving the Poisson equation with boundary conditions. We use the result as our final blended image I_b.

4 Implementation

4.1 Discretization of the Poisson Equation

Since the Poisson equation is linear, we use standard finite differences to approximate each item.

For the left hand side,

$$\nabla^2 f(x,y) = \frac{\partial^2 f(x,y)}{\partial x^2} + \frac{\partial^2 f(x,y)}{\partial y^2}, \tag{6}$$

we apply central difference

$$\frac{\partial^2 f(x,y)}{\partial x^2} \approx f(x+1,y) - 2f(x,y) + f(x-1,y) \tag{7}$$

and

$$\frac{\partial^2 f(x,y)}{\partial y^2} \approx f(x,y+1) - 2f(x,y) + f(x,y-1). \tag{8}$$

where we use a unit step size, i.e.,

$$\begin{cases} \triangle x = 1 \\ \triangle y = 1 \end{cases} \tag{9}$$

Now the Laplacian operation for $f(x,y)$ is approximated as

$$\nabla^2 f(x,y) \approx f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) - 4f(x,y) \tag{10}$$

For the right hand side, we first use standard finite differences to approximate the gradients of the source images.

$$\begin{cases} G_{sx}(x,y) \approx S(x+1,y) - S(x,y) \\ G_{sy}(x,y) \approx S(x,y+1) - S(x,y) \end{cases} \tag{11}$$

where $S(x,y)$ is the source image.

As described in Section 3.2, we create a gradient vector field $(G_x(x,y), G_y(x,y))$ using the gradients of source images and labels obtained from the optimal seam

finding process. With the gradient vector field, we construct a divergence vector field $div(G)$ as follows:

$$div(G) \approx G_x(x, y) - G_x(x-1, y) + G_y(x, y) - G_y(x, y-1) \tag{12}$$

So the discrete form of the Poisson equation can be described as

$$\begin{aligned} &f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1) - 4f(x, y) \\ &= G_x(x, y) - G_x(x-1, y) + G_y(x, y) - G_y(x, y-1) \end{aligned} \tag{13}$$

For each color channel, we solve the Poisson equation with Neumann boundary conditions and obtain a recovered composite image for this channel. Finally, we combine these single channel results into a final blended image I_b .

We have two implementations for the approach presented in this paper on mobile devices. We call the first implementation sequential image blending and the second one global image blending. In the following sections, we describe these two implementations in detail.

4.2 Implementation 1: Sequential Image Blending

In this implementation, we select a base image S_{base} from the source image sequence $S_0, S_1, \ldots, S_n$ and blend others onto the base image sequentially. The blending order is decided by sorting the offsets of the source images.

Figure 4 shows the process. First, we sort the offsets of the source images to create an order for the blending operation. The size of the final blended image I_b

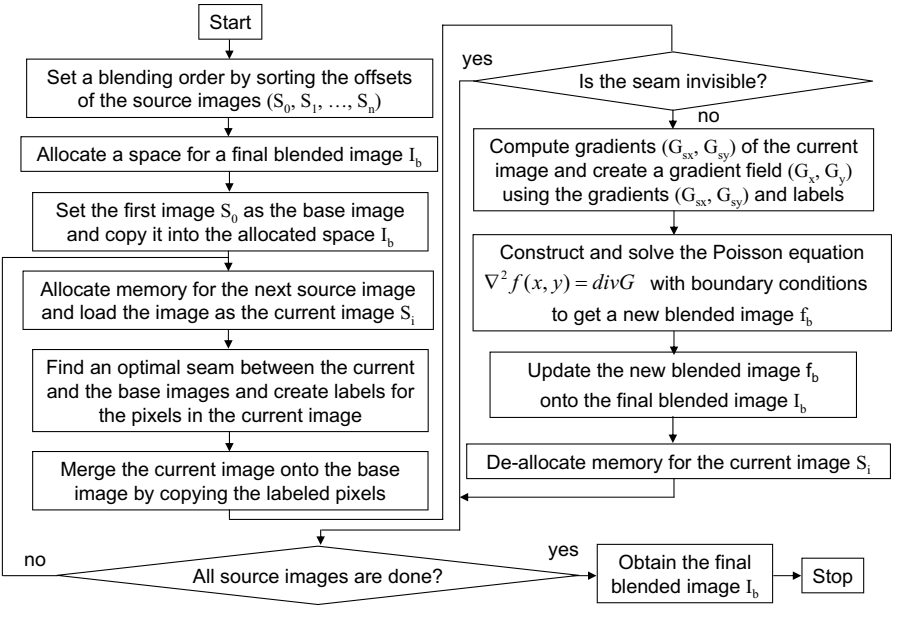


Fig. 4. The work flow of sequential image blending

can be computed by the sizes and offsets of the source images. For convenience, we set the first image S_0 as the base image and put it into the space of the final blended image I_b. Then, the next source image S_i is loaded as the current image. With graph cut optimization, we compute the optimal seam between image I_b and the current image S_i and merge S_i onto image I_b. At the same time, we create labels for all pixels in the overlapping area.

If the seam of the merged image I_b is invisible, the blending process for the current image S_i is done. We can load next image S_{i+1} to replace the current image S_i and repeat the blending process.

Otherwise, if the seam is still visible, then gradient domain smoothing is needed for further processing. In this case, we compute gradients (G_{sx}, G_{sy}) of the current image and create a gradient vector field (G_x, G_y) using labeling information. Then we construct and solve a Poisson equation to obtain a new blended image. We update it onto image I_b. The blending operation for the current image S_i is done, and we continue with the next source image S_{i+1}, until all source images $S_0, S_1, \ldots, S_n$ have been blended into the final image I_b.

One of the main advantages of this implementation is low memory consumption, which is very important for running on mobile devices. There is a disadvantage, though. When more than two images overlap in the same area, the area may be blended several times. Usually, we use this implementation in one dimensional stitching, where only at most two images can overlap at a given pixel.

4.3 Implementation 2: Global Image Blending

In this implementation, we store all source images S_0,S_1, ..., S_n into memory, find optimal seams for all overlapping areas of the source images, and merge and blend them together in one time.

Figure 5 shows the process of this implementation. First, we load all source images $S_0, S_1, \ldots, S_n$ into memory and create a space for the final blended image I_b. Then we compute optimal seams for all overlapping areas of the source images and create labels with graph cut optimization. With the labeling information, we merge the source images together to create a composite image I_c. If the seams in the composite image are invisible, we use it as our final blended image I_b.

If the seams are still visible, we apply the gradient domain blending operation to improve the quality of the composite image I_c. In this case, we compute gradients (G_{sx}, G_{sy}) for all source images and create a gradient vector field (G_x, G_y) to construct a Poisson equation. A new composite image can be recovered from the gradient vector field (G_x, G_y) by solving the Poisson equation with boundary conditions, and we use it as our final blended image I_b.

Since the optimal seam finding operation is performed globally, the obtained optimal seams and labeling are global solutions. The whole composite image only needs to be blended once and the transition smoothing operation is performed globally. It can smooth color differences on the whole composite image. The main disadvantage is that all source images need to be loaded into memory at same time, which is problematic on mobile devices with limited amount of RAM.

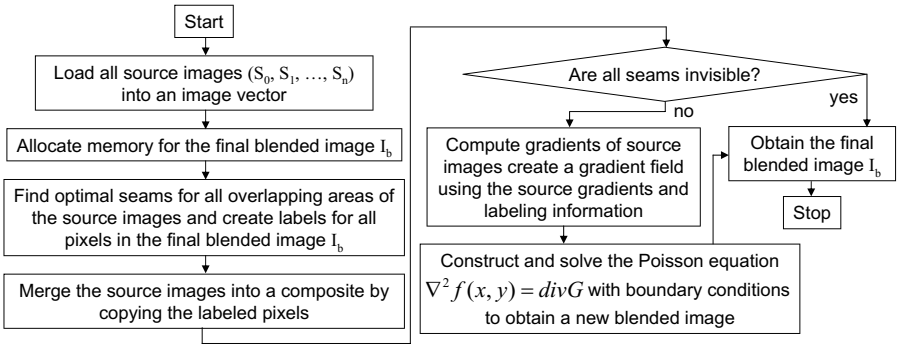


Fig. 5. The work flow of global image blending

4.4 Solving the Poisson Equation

The partial differential equation 4 can be solved after specifying the boundary conditions. As mentioned in Section 3.2, we employ Neumann boundary conditions. In the case of image blending, the boundary conditions are equivalent to dropping any equations involving pixels outside the boundaries of the image.

The approximation of finite differences shown in equation 13 produces a large system of linear equations. One equation corresponds to a pixel in the final blended image. The large system of linear equations is over-constrained. We solve for least-squares optimal vector f using the Full Multigrid Algorithm [18] as our final result.

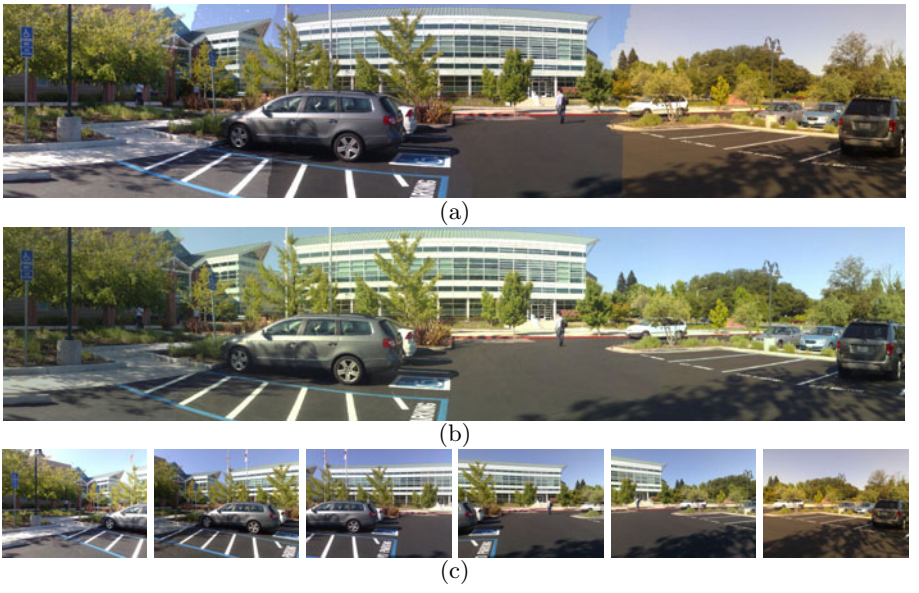
5 Applications and Result Analysis

We have implemented these algorithms on mobile devices and used them to produce high-quality panoramic images in our mobile panorama system. Here we describe some example applications and results obtained by running them on Nokia N95 8GB mobile phones with an ARM 11 332 MHz processor and 128 MB RAM. It can also be run on other mobile devices. The results are shown by figures. In each figure, the top shows the blending result and the bottom shows the source images. In these applications, the size of source images is 1024×768. Since the seam finding process with graph cut optimization needs more memory and computation, we down-sample the source images with scale factor 0.25 to solve the optimization problem to obtain labeling and scale the results back for the final composite image.

We use different kinds of image sequences captured in different conditions to test the approach and verify its performance.

5.1 Applications to Panorama Stitching for Outdoor Scenes

We apply the approach to create panoramic images for outdoor scenes. Figure 6 shows an example result with 6 source images in the image sequence. First, we



(a) (b) (c)

Fig. 6. An outdoor panoramic image created with 6 source images

apply graph cut optimization to find optimal seams and create labeling for the composite image. Since the color and luminance between the source images are very different, the seams are still visible as shown in Figure 6 (a). The graph cut operation takes 323.76 seconds when the source images with the original size are used. However, it only takes 16.4 seconds when the down-sampled source images are used.

(a) (b) (c)

Fig. 7. An indoor panoramic image created with 10 source images

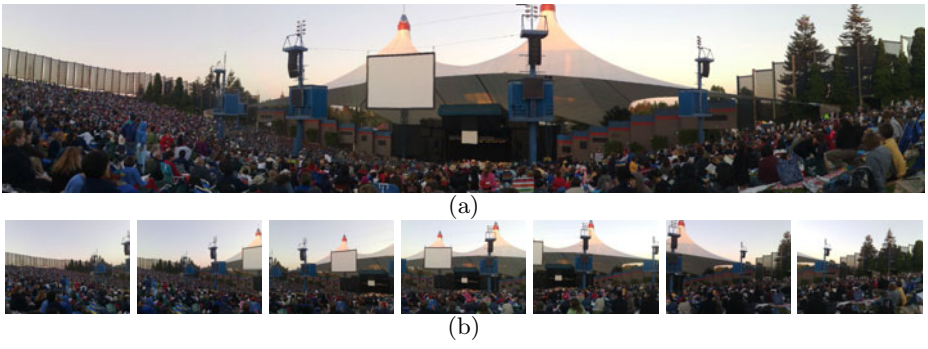


Fig. 8. A panoramic image created with 7 source images captured after sunset

We perform gradient domain image blending to further smooth the color transition between the source images. Figure 6 (b) shows the final result. From the result we can see that all merging artifacts are removed. The color transition between source images is smoothed. All seams are invisible after the gradient domain blending. The gradient domain blending operation takes 54.4 seconds.

5.2 Applications to Panorama Stitching for Indoor Scenes

We applied the approach to panorama stitching for indoor image sequences. Figure 7 shows the source images and results. Figure 7(a) shows the result created by the optimal seam finding process. It takes 15.92 seconds with down-sampled source images. Since the color differences between source images shown in Figure 7(c) are small, some seams are invisible. We apply the gradient domain blending operation to further smooth the color transitions. Figure 7(b) shows the blended panoramic image. From the result we can see that all stitching artifacts are removed. The blending operation takes 80 seconds.

5.3 Applications to Panorama Stitching for Low Light Scenes

We apply the approach to panorama stitching for the scene where the environment light is not sufficient. Figure 8 shows a result created with 7 source images which are captured after sunset. From the panoramic image we can see that the approach still works fine. In panorama stitching, the seam finding process takes 15.92 seconds and the gradient domain operation takes 52.2 seconds.

From the result of this application we can also see another property of the approach. In this case, there are many moving objects in the scene and the approach can find proper seams to avoid ghosting artifacts.

The approach has been tested with many image sequences and applied to different scenes. Figure 9 shows more panoramic images created by the approach. The results are satisfying.

Fig. 9. More example results

6 Conclusions and Discussion

We have presented a gradient domain image blending approach and implemented it on mobile devices. It can be applied to image stitching and image editing for producing high quality mobile panoramic images and composite images.

The approach combines optimal seam finding and transition smoothing processes to provide operations for different source images which are in different conditions. Usually, camera responses and environmental illumination changes when capturing pictures. If the change is small, it may suffice to only use optimal seam finding operation to merge the source images, and the result will be good enough. When the source images are too dissimilar, the seams between the source images in overlapping areas may still be visible. In this case the algorithm also uses the transition smoothing process to further improve the quality of the result image. In this approach, a gradient domain image blending is used for transition smoothing.

We have presented examples in panorama stitching for indoor scenes, outdoor scenes, low light scenes, and others. From the applications we can see that the approach works fine on mobile devices. The results are satisfying.

The main disadvantage of the approach is that memory consumption and computational costs are high. Especially, the optimal seam finding process with graph cut optimization is slow when large source images are used without downsampling.

The future work includes saving memory and speeding up the algorithm. OpenGL ES and parallel processing implementation will be considered.

References

1. Chen, W.C., Xiong, Y., Gao, J., Gelfand, N., Grzeszczuk, R.: Efficient extraction of robust image features on mobile devices. In: ISMAR 2007: Proceedings of the 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality, Washington, DC, USA, pp. 1–2. IEEE Computer Society, Los Alamitos (2007)
2. Takacs, G., Chandrasekhar, V., Gelfand, N., Xiong, Y., Chen, W.C., Bismpigiannis, T., Grzeszczuk, R., Pulli, K., Girod, B.: Outdoors augmented reality on mobile phone using loxel-based visual feature organization. In: MIR 2008: Proceeding of the 1st ACM international conference on Multimedia information retrieval, pp. 427–434. ACM, New York (2008)
3. Chen, D., Tsai, S.S., Chandrasekhar, V., Takacs, G., Singh, J., Girod, B.: Robust image retrieval using multiview scalable vocabulary trees. In: Rabbani, M., Stevenson, R.L. (eds.) Visual Communications and Image Processing 2009. SPIE, vol. 7257, 72570V (2009)
4. Gracias, N., Mahoor, M., Negahdaripour, S., Gleason, A.: Fast image blending using watersheds and graph cuts. Image Vision Comput. 27, 597–607 (2009)
5. Levin, A., Zomet, A., Peleg, S., Weiss, Y.: Seamless image stitching in the gradient domain. In: Pajdla, T., Matas, J.G. (eds.) ECCV 2004. LNCS, vol. 3024, pp. 377–389. Springer, Heidelberg (2004)
6. Milgram, D.L.: Computer methods for creating photomosaics. IEEE Trans. Comput. 24, 1113–1119 (1975)
7. Efros, A.A., Freeman, W.T.: Image quilting for texture synthesis and transfer. In: SIGGRAPH 2001: Proceedings of the 28th annual conference on Computer graphics and interactive techniques, pp. 341–346. ACM, New York (2001)
8. Davis, J.: Mosaics of scenes with moving objects. In: CVPR 1998: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Washington, DC, USA, p. 354. IEEE Computer Society, Los Alamitos (1998)
9. Agarwala, A., Dontcheva, M., Agrawala, M., Drucker, S., Colburn, A., Curless, B., Salesin, D., Cohen, M.: Interactive digital photomontage. ACM Trans. Graph. 23, 294–302 (2004)
10. Uyttendaele, M., Eden, A., Szeliski, R.: Eliminating ghosting and exposure artifacts in image mosaics. In: Computer Vision and Pattern Recognition (CVPR 2001), pp. 509–516. IEEE Computer Society, Los Alamitos (2001)
11. Burt, P.J., Adelson, E.H.: A multiresolution spline with application to image mosaics. ACM Trans. Graph. 2, 217–236 (1983)
12. Szeliski, R., Uyttendaele, M., Steedly, D.: Fast poisson blending using multi-splines. Technical Report MSR-TR-2008-58, Microsoft Research (2008)
13. Kazhdan, M., Hoppe, H.: Streaming multigrid for gradient-domain operations on large images. ACM Trans. Graph. 27, 1–10 (2008)
14. Fattal, R., Lischinski, D., Werman, M.: Gradient domain high dynamic range compression. In: SIGGRAPH 2002: Proceedings of the 29th annual conference on Computer graphics and interactive techniques, pp. 249–256. ACM, New York (2002)
15. Pérez, P., Gangnet, M., Blake, A.: Poisson image editing. ACM Trans. Graph. 22, 313–318 (2003)
16. Jia, J., Sun, J., Tang, C.K., Shum, H.Y.: Drag-and-drop pasting. In: SIGGRAPH 2006: ACM SIGGRAPH 2006 Papers, pp. 631–637. ACM, New York (2006)
17. Kolmogorov, V., Zabih, R.: What energy functions can be minimized via graph cuts. IEEE Transactions on Pattern Analysis and Machine Intelligence 26, 65–81 (2004)
18. Briggs, W.L., Henson, V.E., McCormick, S.F.: A Multigrid Tutorial. The Society for Industrial and Applied Mathematics, New York (2001)