Eliminating Ghosting and Exposure Artifacts in Image Mosaics

Matthew Uyttendaele
Microsoft Research

Ashley Eden
Harvard University

Richard Szeliski
Microsoft Research

Abstract

As panoramic photography becomes increasingly popular, there is a greater need for high-quality software to automatically create panoramic images. Existing algorithms either produce a rough "stitch" that cannot deal with common artifacts, or require user input. This paper presents methods for dealing with two artifacts that often occur in practice. Our first contribution is a method for dealing with objects that move between different views of a dynamic scene. If such moving objects are left in, they will appear blurry and "ghosted". Treating such regions as nodes in a graph, we use a vertex cover algorithm to selectively remove all but one instance of each object. Our second contribution is a method for continuously adjusting exposure across multiple images in order to eliminate visible shifts in brightness or hue. We compute exposure corrections on a block-by block basis, then smoothly interpolate the parameters using a spline to get spatially continuous exposure adjustment. Our enhancements, combined with previously published techniques for automatic image stitching, result in a high-quality automated stitcher that exhibits far fewer artifacts than existing software.

1. Introduction

As panoramic photography becomes increasingly popular, there is a greater need for software to create panoramic images. Ideally, the image stitching process should be completely automatic, requiring no user information in calculating the panorama [1,10,11,12]. This not only applies to registering the images, but also to fixing irregularities typical to amateur photography. Two such irregularities discussed in this paper are movement of objects within a scene, and differences in exposure between images.

One of the problems in automatic image stitching is that of de-ghosting. When the images are taken, there is no guarantee that objects in the image stay stationary from one image to the next. This becomes a problem in the areas of overlap between images. When the images are stitched together, we take a composite of the overlapping images in order to create a smooth transition between neighbors. However, if regions of the scene are not stationary, the overlap image will be slightly different in each image contributing to the overlap. Thus, those regions of the composited image will contain combinations of pixel values from entirely different parts of the scene. For example, if a person moves his head in an area of overlap, the region containing his head in the stitched image will be a combination of head and background from both images. It will give the head a ghosted look, not to mention that this ghosted head will appear in two locations.

In order to get around this problem of ghosting, we need to display the stitched image as if nothing in the scene moved. Thus, when regions of the scene do have movement, we would like to use pixel values from only one of the contributing images for that region. In order to accomplish this, however, we need to determine a) where the movement occurred, and b) which image to use.

Several methods have been proposed to eliminate ghosts. Shum and Szeliski [10] propose a method for deghosting small misregistrations based on computing optic flow and then doing a multi-way morph. Median filters are often used and are effective when more than half the images contain consistent pixels [3, 9]. This is not the case for mosaics created from a relatively small number of images. Davis [2] proposes cutting images between regions of movement, finding the best cut with Dijkstra's algorithm. However, it is not clear how to generalize this to mosaics created from many overlapping images. We need a novel algorithm to tackle the more complicated problem of multiple overlapping regions of movement.

Another problem in automatic image stitching is exposure differences between images. Exposure differences are a common occurrence, especially with digital photographs. If the differences are not corrected, the panorama will appear to have seams, even when the images are blended in overlapping regions. Additionally our difference-based de-ghosting algorithm could interpret exposure differences as moving objects. Thus, we must find a way to equalize the exposure of each image based on the information in neighboring images while retaining local smoothness.

Previous work in this area uses a large number of images of the same scene to do a radiometric calibration of the camera [4,5,6]. In our work we don't assume a calibration step. In work by Hasler *et al* [7], the image registration and the camera's parametric Opto-Electronic Conversion Function are simultaneously computed for pairs of images. This assumes a known parametric model for the camera and it is not clear how to generalize this to multiple overlapping images. Burt and Adelson [13] use multi-resolution splines to perform spatial blending between different images. However, since their method depends on band-pass image pyramids, it is not clear how to apply it to the irregularly shaped images present in general image mosaics. Furthermore, current stitching techniques [1,10] already use large "feathering" regions, so multi-resolution splining may not help. In our work we compute corrections on a block-by-block basis, and then smoothly interpolate the parameters using a spline to get spatially continuous exposure adjustment.

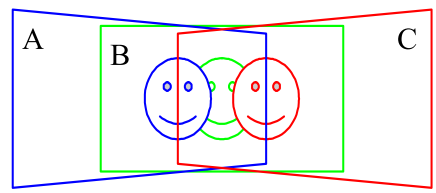
2. Ghosting Artifacts

2.1 Where Does Movement Occur?

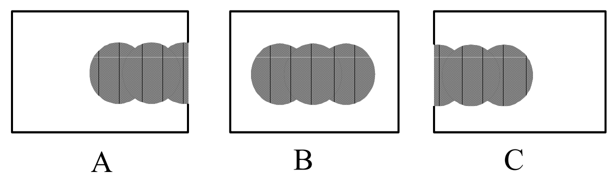
The first step in our de-ghosting process is to determine which regions in the input images are not static and thus differ across images. We limit the search for regions of difference (ROD) to the areas of overlap between input images. To identify RODs, a map is computed for each input image by flagging pixels which differ by more than a certain threshold from pixels in overlapping images. To smooth the difference maps, a morphological erode and dilate step is then applied. Next, a region extraction algorithm is applied to identify and label contiguous regions. Figure 1 depicts the construction of difference maps for a simple mosaic.

Our overall goal is to use information from only one image for each ROD. Thus, we must group corresponding regions across images, keep information from one of the images, and ignore corresponding information in the other images. But how do we group corresponding regions? Not all corresponding regions are the same size. Because more than one image may overlap with a given image, and each area of overlap is not the same size or location, an image may have different sized RODs than its overlapping neighbors' Figure 1. We define RODs in different images to be corresponding if they have any overlap at all.

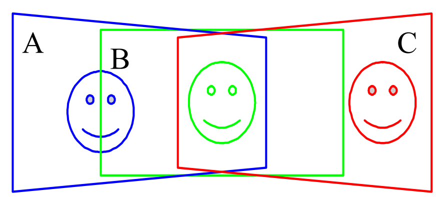
Note that corresponding RODs are not defined to be the same object in different images. In figure 1a the moving face has slight motion. This motion induces the RODs shown in figure 1b. In figure 1c the face has more motion and does not happen to overlap in overlapping images. In this case the algorithm presented will map the face into separate RODs as shown in figure 1d and these corresponding RODs will be handled separately.



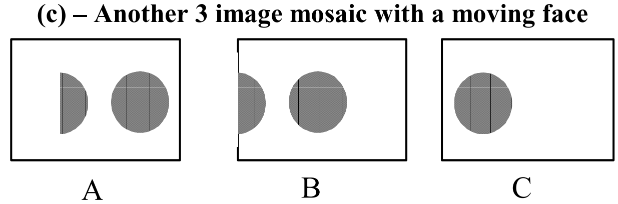
(a) – A 3 image mosaic with a moving face



(b) – Corresponding RODs caused by motion in (a)



(c) – Another 3 image mosaic with a moving face



(d) – Corresponding RODs caused by motion in (c)

Figure 1

2.2 Which Image to Use?

Assume, for a moment, that RODs are vertices in a graph and that corresponding RODs are linked by an edge. We know by definition that non-corresponding RODs will not overlap, and vice versa. Thus, each chain of corresponding regions will be its own graph. The graph formed by the RODs in Figure 1 is shown in the figure below.

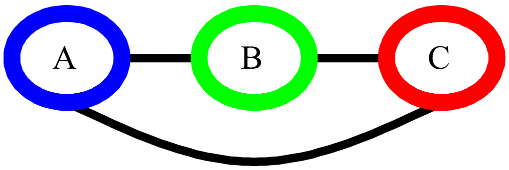


Figure 2 – Graph of corresponding ROD in figure 1b

Now, consider the vertex cover of such a graph. The vertex cover of an undirected graph **G = (V, E)** is a subset **V'** $\subseteq$ **V** such that each edge in **E** is incident on at least one of the vertices in **V'**. In other words, each pair of overlapping regions has at least one of the regions in the vertex cover. If we were to identify the vertex cover and remove these vertices from the graph, we would be left with a set of vertices that have no edges in common, i.e. do not overlap. Thus, if we were to remove the cover, we would be left with a set of non-corresponding regions. Next if we eliminate the contribution of pixel data corresponding to the vertices in the cover, we would have no conflicting regions, and no ghosting.

A problem with this is that regions are removed relatively randomly. AB, BC and AC are all minimum solutions to the vertex cover in the graph of Figure 2. To fix this we use the following idea to help select regions. To avoid having object discontinuities arising from selecting a ROD (vertex) at an image boundary, we give RODs (vertices) a weight proportional to its size and proximity to the center of its image. Larger and more central RODs receive a higher weight. The weight is computed by summing the feather weights, described in [10], within the ROD.

Figure 3 depicts the advantages of using this heuristic. In this figure, an oval object is moving during the acquisition of 3 images that form a mosaic. Two simple graphs arise from this configuration. If we were to solve the basic vertex cover problem, either vertex in either graph would be a reasonable solution. However by weighting vertices that correspond to more central regions higher and eliminating the minimum weight vertex from the graph, we choose the region in the center image. In this case B1 is weighted higher than A and B2 higher than C, so A and C are the minimum weight vertices to be removed. This avoids the object discontinuities that would have arisen from selecting either of the side images.

Thus, we must compute the minimum weight vertex cover of the graph. The weighted vertex cover problem is summarized as follows. Given an undirected graph **G = (V, E)** and a positive integer weight function **w: V —** $\mathbf{Z}^+$ on the vertices, find the cover **V'** $\subseteq$ **V** that minimizes the weight **w (V')**. The solution of this is known to be NP complete. In our implementation we run an exhaustive solver for graphs with 8 or less vertices and run a randomized approximation algorithm [8] for larger graphs.

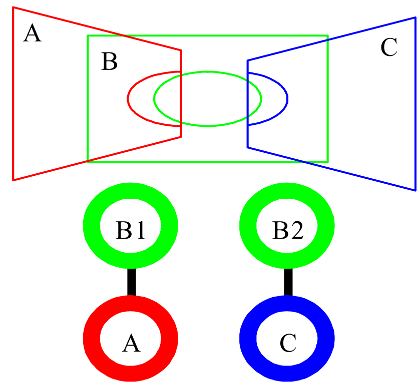


Figure 3 – simple mosaic containing a moving oval shape and its associated ROD graphs.

In mosaics that contain many mutually overlapping images, these graphs may become arbitrarily complex as shown in Figure 4. There are several points worth noting in these graphs. There may be a group of images that do not all overlap with each other, but all overlap with a ROD of at least one of the images (Figure 4). Similarly, it is possible to have two RODs from the same image in the same graph. However, by definition, they do not overlap, and thus do not share an edge. Therefore, edges connect only regions of different images.

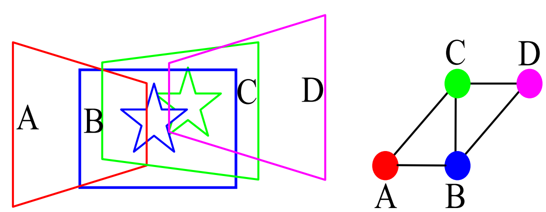


Figure 4 – Mosaic containing moving star shape plus associated ROD graph.

The method of minimum weight vertex cover is not failsafe, however. The solutions to the weighted vertex cover for the graph in Figure 4 might be CB, ABD or ACD. If CB were chosen as the solution, this would eliminate pixel data for images C and B in their RODs, thus causing a hole in the mosaic wherever the RODs for A or D overlap C and D. In practice however we find that this situation is rare. Due to the high weight placed on large and central RODs the chosen solution is usually one of ABD or ACD. To eliminate this artifact we detect when it occurs and in those regions we fall back to the fully composited mosaic.

3. Exposure Artifacts

When capturing images to create mosaics, automatic cameras often change their exposure settings. Even if the images are blended in the areas of overlap, if the exposure difference between images is too high, it will give the panorama a creased look around the edges of the overlap. Feathering [10] the effect of the blend into the rest of the image will help remove creases, but the exposure change will still be too abrupt to appear natural (Figure 7).

Exposure differences can also confuse the previously described de-ghosting algorithm, causing it to interpret differences across images as moving objects. Thus, it is important to have an exposure compensation step prior to de-ghosting. Ideally we would do a full radiometric calibration of the camera; however given limited input data this is infeasible for many mosaics.

To eliminate exposure adjustment artifacts in mosaics, we need to apply a transfer function to an input image to make it look more like its neighbors. This becomes complex as the number of overlapping neighbors increases. To alleviate this complexity, we have developed a block-based exposure adjustment technique. This allows us to vary the image's transfer function as it overlaps with differently exposed neighbors.

Each image is divided into blocks. The block size is variable in our implementation, but we have found that 32x32 usually gives good results. Within each block, we compute a quadratic transfer function that in a least-squares sense best maps, in luminance, this image block to the composite of images overlapping this block. We may then iterate and use the adjusted composite as the reference for the next step. In our implementation we find the 3 iterations yields good results.

As with many block-based algorithms the results of simply applying the transfer function in isolation within the block are too "blocky" (Figure 7-B). To smooth the variation in the transfer function distributions, we use a combination of two techniques. First, we average the functions in each patch with those of their neighbors. In our implementation, we use an iterated separable kernel of (¼, ½, ¼), and typically use 2 iterations.

Second, for each pixel, we blend the results of applying the transfer function from neighboring patches. This can be done using bilinear or biquadratic interpolators as shown in Equation 1. The resulting pixel values are the same as if we had interpolated a separate function for each pixel, but since we implement the transfer function using a look-up table, the computational cost is much smaller.

$$p(x,y)=\sum_{v=-1}^{v=1} C_v\left(\frac{y}{N}\right)\sum_{u=-1}^{u=1} C_u\left(\frac{x}{N}\right) f_{m+u,\,n+v}(p(x,y))$$

$\mathrm{p(x, y)}$: pixel value at block location x, y

N : block size

$C_n(x)$: bilinear or bicubic coefficient

$f_{u,v}(x)$: transfer function for block u, v

Equation 1 – interpolated transfer function

Of course, exposure is not the only reason images might be different. Other effects such as vignetting may also cause image differences. An advantage of our method is that rather than attempting to solve a parametric model for such complex camera characteristics, the mapping of pixel values varies smoothly across the image and will locally compensate for the previously mentioned effects.

4. Results

Figures 5 and 6 show the results of our de-ghosting algorithm. In both cases, the algorithm was set with the same parameters and ran completely automatically. In figure 5 a four image mosaic is shown. Notice how the individual images that comprise the mosaic contained moving objects. When a basic image registration and blending algorithm [1] is applied (figure 5-A) the moving objects appear as "ghosts". Since they did not remain in the same position between images they are blended with the background. Figure 5-B is the result of applying our de-ghosting algorithm. Notice that the image of the person in the far right was ghosted in the original while our algorithm removes this ghost in the result.

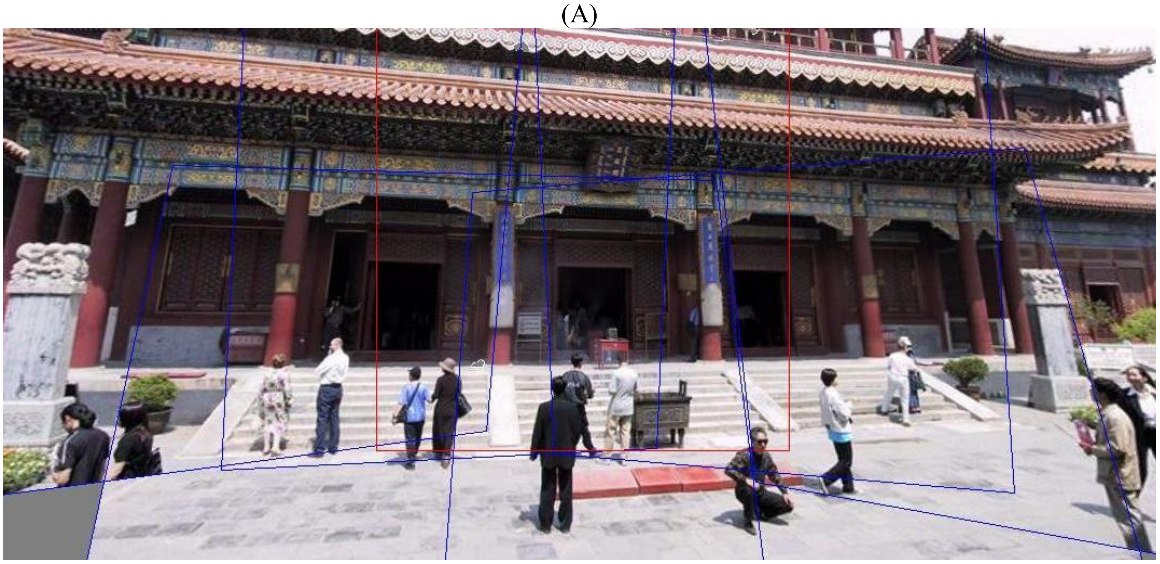
In figure 6, the algorithm is applied to a much more difficult sequence. Notice that many moving objects are contained in this scene in which many images have significant overlap. Figure 6-A shows the results of a

(A)

(B)

Figure 5 – (A) Ghosted mosaic. (B) Result of de-ghosting algorithm.

(A)



(B)

Figure 6 – (A) Ghosted mosaic. (B) Result of de-ghosting algorithm.

simple registration and blend. Applying our algorithm to this mosaic yields the result in Figure 6-B. Again notice the significant reduction in ghosting.

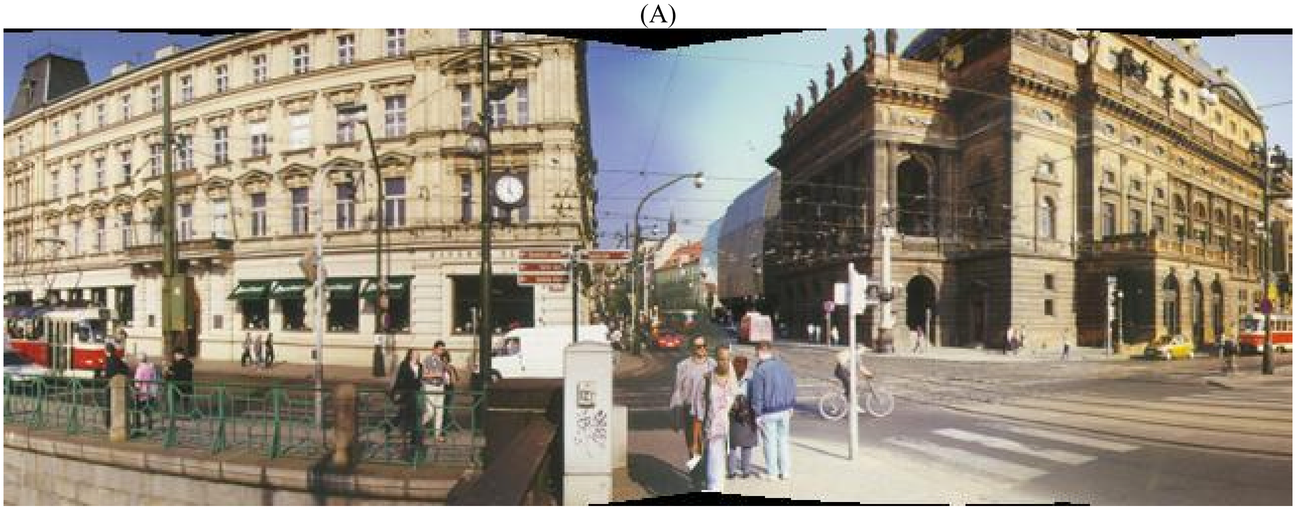
In Figure 7, we demonstrate the results of the exposure compensation algorithm. We used a block size of 32x32 for all exposure compensation results presented here. Two images with significant exposure differences are registered and feathered as shown in Figure 7-A. Notice the sharp transition between images in the center of the mosaic. If we simply do block based exposure adjustment we get the result in 7-B. After applying the smoothing of the transfer functions we get the result in Figure 7-C.

Our final example, shown in Figure 8, contains both exposure differences and moving objects. In the mosaic in Figure 8-A notice the sharp transition in the sky caused by exposure differences, as well as the people moving in the scene. Figure 8-B shows the result of applying exposure compensation followed by the de-ghosting algorithm. Notice that both types of artifacts are removed, resulting in a much better looking mosaic.

(A) (B) (C)

Figure 7 – (A) 2 image mosaic with exposure differences. (B) Blocky result before smooth interpolation of transfer functions (C) Result of the exposure compensation algorithm.

(A)



(B)

Figure 8 (A) Mosaic which contains both ghost and exposure artifacts. (B) Result of applying exposure compensation followed by ghost removal.

5. Conclusion

Panoramic photography presents several problems in creating a seamless mosaic. In this paper we have presented novel algorithms for fixing two such common problems. In order to remove ghost effects caused by moving objects, we use a weighted vertex cover algorithm. Although there are still cases of duplicate images and holes, using the minimum cost directly minimizes these effects.

In order to obtain a good difference metric for use in de-ghosting, and for creating a smooth, natural looking image, exposure adjustment was also needed. We presented an algorithm of block-based adjustment, which alters the pixel values using a weighted average of lookup tables from nearby parts of the image. By altering the image in blocks, it is possible to adjust scenes where a single change in exposure would result in an under or overexposed image. In addition, taking into account local and overlap information for every pixel allows for smooth transitions not only between overlapping images, but within an image containing multiple regions of overlap.

6. References

[1] R.Szeliski and H. Y. Shum., *Creating Full View Panoramic Image Mosaics and Environment Maps,* Proc. SIGGRAPH'97 pg. 251-258

[2] J. Davis, *Mosaics of Scenes with Moving Objects,* IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'98) pg. 354-360

[3] T. Echigo, R. Radke, P. Ramadge, H. Miyamori, S. Iisaku, *Ghost Error Elimination and Superimposition of Moving Objects in Video Mosaicing,* IEEE International Conference on Image Processing 1999
[4] T. Mitsunaga, S. Nayar, *Radiometric Self Calibration,* IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'99) pg. 374-380
[5] P. Debevec, J. Malik, *Recovering High Dynamic Range Radiance Maps from Photographs,* Proc. SIGGRAPH'97 pg. 369-378
[6] S. Mann, R. Piccard, *Being 'Undigital' with Digital Cameras: Extending the Dynamic Range by Combining Differently Exposed Pictures ,* Proc. Of ISTs 48th Annual Conf. pg. 422-428, May 1995
[7] D. Hasler, S. Süsstrunk, *Colour Handling in panoramic photography,* Proceedings IS&T/SPIE Electronic Imaging 2001: Videometrics and Optical Methods for 3D Shape Measurements, Vol. 4309
[8] R. Motwani, *Lecture Notes on Approximation Algorithms Volume I*, Technical Report Dept. of CS, Stanford, 1992
[9] M. Irani, P. Anandan, J. Bergen, R. Kumar, S. Hsu, *Efficient Representations of Video Sequences and Their Applications,* Signal Processing: Image Communication Volume 8, 1996 pg. 327-351
[10] H.-Y. Shum, R. Szeliski, *Construction of Panoramic Image Mosaics with Global and Local Alignment*, International Journal of Computer Vision, Vol. 36(2), pp. 101-130, Feb. 2000
[11] F. Dufaux, F. Moscheni, *Background Mosaicking for Low Bit Rate Video Coding*, IEEE International Conference on Image Processing 1996
[12] H. Sawhney, S. Ayer, *Compact representations of videos through dominant and multiple motion estimation*, IEEE Trans. Pattern Analysis and Machine Intelligence, pp. 814--830, Aug. 1996.
[13] P. J. Burt and E. H. Adelson, *A multiresolution spline with applications to image mosaics*, ACM Transactions on Graphics, 2(4), pp. 217-236, October 1983.