Fast Image Blending using Watersheds and Graph Cuts

Nuno Gracias*, Art Gleason, Shahriar Negahdaripour,
and Mohammad Mahoor
Department of Electrical and Computer Engineering
University of Miami
Coral Gables, FL-33124, USA
ngracias@isr.ist.utl.pt

Abstract

This paper presents a novel approach for combining a set of registered images into a composite mosaic with no visible seams and minimal texture distortion. To promote execution speed in building large area mosaics, the mosaic space is divided into disjoint regions of image intersection based on a geometric criterion. Pair-wise image blending is performed independently in each region by means of watershed segmentation and graph cut optimization. A contribution of this work – use of watershed segmentation to find possible cuts over areas of low photometric difference – allows for searching over a much smaller set of watershed segments, instead of over the entire set of pixels in the intersection zone.

The proposed method presents several advantages. The use of graph cuts over image pairs guarantees the globally optimal solution for each intersection region. The independence of such regions makes the algorithm suitable for parallel implementation. The separated use of the geometric and photometric criteria frees the need for a weighting parameter. Finally, it allows the efficient creation of large mosaics, without user intervention. We illustrate the performance of the approach on image sequences with prominent 3D content and moving objects.

1 Introduction

Image blending is the final and often very important step in producing high quality mosaics. Radiometric variations in overlapping views and violation of certain scene assumptions commonly made – rigidity, stationary, and (or) planarity – lead to geometric misalignments and photometric differences. Upon blending, these usually result in degrading artifacts, such as blurry regions or artificial seams.

In this paper we are interested in developing a image blending algorithm capable of producing seamless 2D mosaics and preserving the appearance and clarity of object textures while dealing with misalignments resulting from strong 3D content. A primary

*This work has been partially supported by DoD/DOE/EPA SERDP Proj. CS – 1333 and by the Portuguese Fundação para a Ciência e Tecnologia BPD/14602/2003.

BMVC 2006 doi:10.5244/C.20.49

motivation for this work is the creation of large–area underwater habitat mosaics capable of being interpreted by a human expert. This application stresses the need to preserve the consistency of textures which are of large importance in the successful recognition of benthic structures. We favor blending using contributions from a single image for each mosaic point, while minimizing the intensity discrepancies along the boundary lines of overlapping images. Additionally, we are interested in obtaining and comparing fast methods that could be applied in near real time.

1.1 Background

The watershed transform [12] is a region-based segmentation approach whose intuitive idea is that of a topographic relief flooded by water: watersheds are the divide lines of the domains of attraction of rain falling over the region. The image processing literature provides a large number of application examples of watersheds, such as 2D/3D region and surface segmentation [3] and in contour detection [10, 9]. However, to the best of our knowledge, it has not been used in the context of mosaic blending. Recently, Li et al. [9] illustrated the advantage of using clusters of pixels to reduce the algorithmic complexity of finding approximate object contours in images, given a coarse user input. This paper aims at the same benefit, but in the different domain of automated mosaic blending.

Many of the problems that arise in early vision can be naturally expressed in terms of energy minimization. In the last few years, a new approach to solving these problems has gained wide acceptance, based on graph cuts from combinatorial optimization. The classical use of graph cuts in computer vision is to solve pixel-labelling problems. The input is a set of pixels P and a set of labels L. The goal is to find a labelling f (i.e., a mapping from P to L) which minimizes an energy function in the standard form

$$E(f) = \sum_{p \in P} D_p(f_p) + \sum_{p,q \in N} V_{p,q}(f_p, f_q), \tag{1}$$

where $N \subset P \times P$ is a neighborhood system on pixels. $D_p(f_p)$ defines the cost of assigning the label f_p to the pixel p, while $V_{p,q}(f_p, f_q)$ represents the cost of assigning the labels f_p, f_q to the adjacent pixels p and q (used to impose spatial smoothness). For the case of binary labels, Equation 1 is a particular case of the *Ising* model, and the global minimum can be found over a single graph cut computation [7].

1.2 Related work

The approaches to image stitching in the literature can be divided into two main classes [1]: Transition smoothing and optimal seam finding.

Transition smoothing methods, commonly referred to as feathering or alpha blending, take the locations of seams between images as a given and attempt to minimize the visibility of seams by smoothing. A traditional approach is multiresolution splining by Burt and Adelson [4]. Recent examples include gradient domain blending [1, 2], which reduces the inconsistencies due to illumination changes and variations in the photometric response of the cameras. Gradient domain methods have the advantage that dissimilarities in the gradients are invariant to the average image intensity, but require recovering the blended image from a gradient description. On a general case, there is no image that exactly matches the gradient field. A least-squares solution can be found by solving a discrete Poisson equation, at a high computational cost.

Optimal seam finding methods, in contrast, place the seam between two images where intensity differences in their area of overlap are minimal [6, 5]. The method proposed in this paper fits in this class, and is therefore related to previous work. Uyttendaele et al. [11] search for regions of difference (ROD) among images using thresholds over the image difference. Each ROD is assigned to just one image by computing the minimum weight vertex cover over a graph representation of weighted RODs. This is done exhaustively for 8 or less vertices, and by a randomized approximation for more. However there is little control over the shape or sizes of the RODs, and it is not clear how the quality of the results scales with the number or RODs. Agarwala et al. [2] use graph cuts to find the contribution regions among several images where each pixel is treated independently. Pixel labelling is performed in general terms, by minimizing over all images at the same time. To find a solution, an iterative alpha–expansion graph cut algorithm was used. The application of multi-label graph cuts requires a potentially large number of graph-cut iterations (which grows with the number of labels). In contrast, our approach constrains the problem by dividing the mosaic space into large disjoint regions using a geometric criterion of distance to camera centers. We independently solve a single binary labelling problem for each region, releasing the need for iterative approximations. Since only one graph–cut is performed per region, the total optimization time for our method is in the order of a single multi-label alpha-expansion iteration. Section 4 demonstrates that our approach is more suited to the processing of large sets of images without human intervention.

2 Image Blending with Watersheds and Graph Cuts

The watershed/graph cut approach divides the regions of image intersection into sets of disjoint segments then finds the labelling of the segments that minimizes intensity differences along the seam. By *labelling* we refer to the association of each watershed segment to one of the images. By *seam* we refer to the combined path that separates neighboring segments that have different labels.

2.1 Segmentation of the intersection region

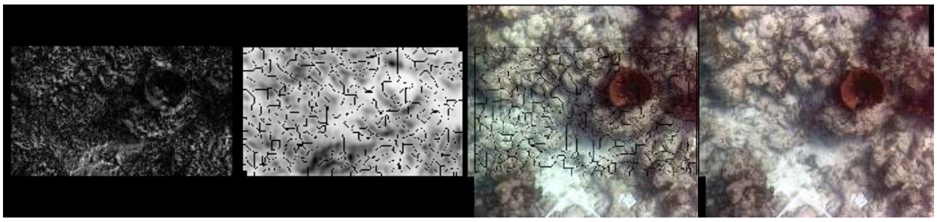
The watershed transform divides an input surface into a set of disjoint regions around local minima. When used on a similarity surface, created from the intersection of a given pair of images, it aggregates the areas where the images are least similar. These are the areas to be avoided when looking for the best seam between the images.

This paper uses the absolute value of the grey–level differences as a similarity criterion. Direct application of the watershed algorithm to the grey–level image difference generally results in over–segmentation, i.e. the creation of a large number of very small contiguous regions. To avoid over–segmentation the image is smoothed prior to the application of the watershed algorithm. For all the image sets of this paper, good results were achieved using a Gaussian low pass filter with a standard deviation of 1.4 pixels.

An example of watershed segmentation and blending using two registered images from an underwater sequence of a coral patch is shown in Figures 1 (a) and (b). Blending using simple geometric criteria is inadequate; the average image (Figure 1(c)) is blurry, and filling pixels with the contribution from the image with the closest center produces a visible seam (Figure 1(d)). Figure 2 presents the absolute image difference and the watershed result over the low pass filtered difference. At this point we could blend the images by simply associating each watershed segment to the closest image center (Figure

(a) (b) (c) (d)

Figure 1: Original images used for the watershed blending example (a and b) and examples of purely geometric blending – average over intersection (c) and closest to image center (d).



(a) (b) (c) (d)

Figure 2: Absolute value of the grey–level image difference (a), watershed segmentation over the inverted, low–pass filtered difference (b), and segmentation outline over the closest–to–center blending (c). Simple watershed blending obtained by associating each segment to the closest image center (d).

2(d)). Although not perfect, it improves greatly over the simple geometric algorithms (Figure 1(c,d)).

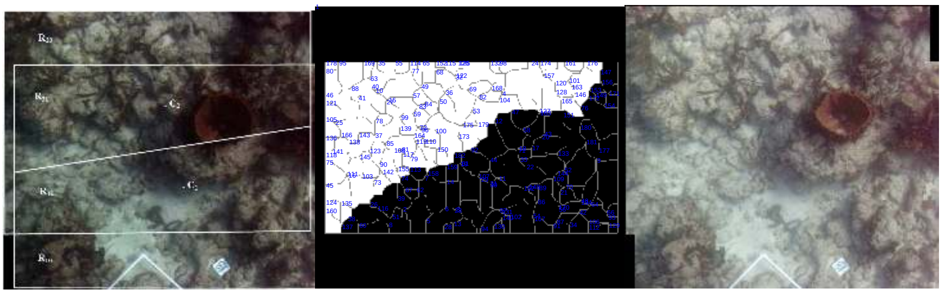
2.2 Graph cut labelling

The visibility of the seams can be further reduced by penalizing the photometric difference along the seams, and using graph cuts to assign the watershed segments. Let L be a binary label vector of size n, where n is the number of watershed segments. Let S_i be the binary mask that defines segment i.

Let D_1 and D_2 be vectors containing the costs of assigning each segment to each image. Let V be the $n \times n$ matrix such that $V(i,j)$ contains the cost of having S_i and S_j associated with different images.

The costs are found as follows. Let I_{1w} and I_{2w} be the images to be blended, already warped into a common (mosaic) reference frame. Let R_{12} be the mosaic frame region were the mosaic points are closer to the center of image 1 and second closer to image 2. Let R_{21} be the opposite. The union of R_{12} and R_{21} completely defines the intersection of I_{1w} and I_{2w}. Let R_{10} and R_{20} be the areas where outside the intersection region where the mosaic points are closer to the center of image 1 and 2 respectively. R_{12}, R_{21}, R_{10} and R_{20} are mutually exclusive, i.e., have no intersection. These regions are illustrated in Figure 3 (a). We denote $D_{diff}(S_i, S_j)$ as the vector of intensity differences between $I1w$ and $I2w$ along the *common boundary* of regions S_i and S_j. If S_i and S_j are not neighbors (i.e. no common boundary) then $D_{diff}(S_i, S_j)$ is null.

The assignment costs penalize the segments that are neighbors to R_{10} and are attributed to image 2 and vice–versa, whereas the interaction costs penalize the dissimilarity



(a) (b) (c)

Figure 3: Example of graph cut labelling over the watershed segments – Regions involved in computing the cost function (a), optimal labelling (b) and resulting blending (c).

along common boundaries,

$$D_1(i) = \left\|D_{diff}(S_i, R_{20})\right\|_p; \quad D_2(i) = \left\|D_{diff}(S_i, R_{10})\right\|_p; \quad V(i,j) = \left\|D_{diff}(S_i, S_j)\right\|_p$$

where $\|\cdot\|_p$ is the $p-norm$. We define a cost function as

$$C(L) = \sum_i \left(D_1(i) \cdot \overline{L}(i) + D_2(i) \cdot L(i)\right) \; + \; \sum_{i,j} V(i,j) \cdot (L(i) \otimes L(j))$$

where $\overline{L}(i) = 1 - L(i)$ and $\otimes$ is the exclusive OR.

The labelling problem above can be cast as a binary graph–cut problem. An efficient algorithm exists based on a max–flow approach[1], which guaranties global minimization [8]. The main condition for applicability and guarantee of global minimum is that the cost function be *regular*, defined as

$$V_{ij}(0,0) + V_{ij}(1,1) \leq V_{ij}(0,1) + V_{ij}(1,0)$$

where $V_{ij}(l_i, l_j), l_i, l_j \in \{0,1\}$ refers to the costs of each combination of having segments i and j attributed to each of the images. Our cost function is regular since $V_{ij}(0,0) = V_{ij}(1,1) = 0$, $V_{ij}(0,1) \geq 0$ and $V_{ij}(1,0) \geq 0$ for all i, j.

Figure 3 illustrates the outcome of the process using $p = 1$. Comparing to the simple watershed blending result from the previous section (Figure 2 (d)), two main improvements are noticeable: (1) Photometric criterion helps to preserve the most prominent scene features such as the circular sponge on the right; (2) Use of boundary conditions defined by R_{10} and R_{20} eliminated the seams at the limits of the image intersection areas.

2.3 Dealing with multiple images

Sections 2.1 and 2.2 described the watershed/graph cut algorithm operating on a pair of images. Extension to any number of images assumes known image–to–mosaic coordinate transformations, and requires dividing the mosaic space in disjoint *regions of image intersection* (ROII). These regions are obtained from the first and second closest maps. We refer to the *first closest map* as the two–dimensional array that, for each element (u,v), contains the index of the image whose center is the closest to (u,v). Conversely, the *second closest map* contains the indicies to the second closest image. Let R_{ij} denote the

[1] The C++ code for constructing the graph and finding the minimal cut is available in the internet [8].

mosaic region where, simultaneously, image i is the closest image and image j is the second closest. Every pair of overlapping images i and j will create a ROII, which is defined as $ROII_{i,j} = R_{ij} \cup R_{ji}$. Both closest maps and the ROIIs are defined only by geometric (registration) parameters and can be computed very efficiently.

Once the ROIIs are defined, then pair–wise image blending is performed independently in each region, as described previously for the case of two images.

From an implementation point of view it should be noted that we are using geometric and photometric criteria separately — the initial computation of the ROIIs is purely geometric while the posterior watershed blending is purely photometric. This separation allows for a very compact memory usage. All the required information is stored in just four arrays: the first and second closest maps, and their corresponding image texture mosaics. These arrays have the dimensions of the final mosaic. Such compact storage is of great importance when processing large data sets and large mosaics.

3 Results

The performance of the approach is illustrated on two distinct sequences.

The first data set is a panoramic sequence of an outdoor scene[2], captured under rotation, with multiple moving pedestrians. It was initially used by Uyttendaele et al. [11] and more recently by Aggarwal et al. [2]. The sequence is available as a stack of 7 warped images, already transformed into the mosaic frame. Figure 4 contains a sub–set of the original images and the resulting watershed mosaic. The mosaic shows no visible cuts over the people, except for the cases where a cut is unavoidable (example – feet of a man on the lower right, for which there is no possible cut that could either include or exclude him totally).

The second sequence contains 10 underwater images of a coral reef patch (Figure 5). The image motions were estimated based on a planar model for the environment, resulting in registration inaccuracies over the areas of strong 3-D structure. The watershed/graph cut blending provided a realistic rendering of the scene, by cutting around the prominent benthic structures (such as rocks and coral heads).

4 Comparison to pixel–level graph cut blending

A central idea in this paper is that watershed segmentation greatly reduces the search space for finding contribution boundaries without affecting the seam quality, when compared to searching over all individual pixels in the intersection zone. Searching over individual pixels would allow for an arbitrarily shaped seam, whereas our method imposes the seam to be formed by the boundaries of the watershed segments. Therefore it is relevant to compare both approaches in terms of execution speed and image difference along the seams. For this purpose, a pixel–level equivalent of our method was implemented, using 8–connectivity to compute the neighboring costs for each pixel.

Using the mosaic of Figure 5 of 1172×795 pixel, comparative results were obtained for several values of σ (the standard deviation of the low pass Gaussian filter), in the range $\sigma \in [0.8 \quad 5]$ pixel. The size of the watershed segments grows approximately linearly with σ and ranges from 46 to 920 pixels.

[2] This set is available at http://grail.cs.washington.edu/projects/photomontage

Figure 4: 4 original images from the outdoor panoramic sequence (top) and watershed blending result (down), cropped over the area containing moving people.

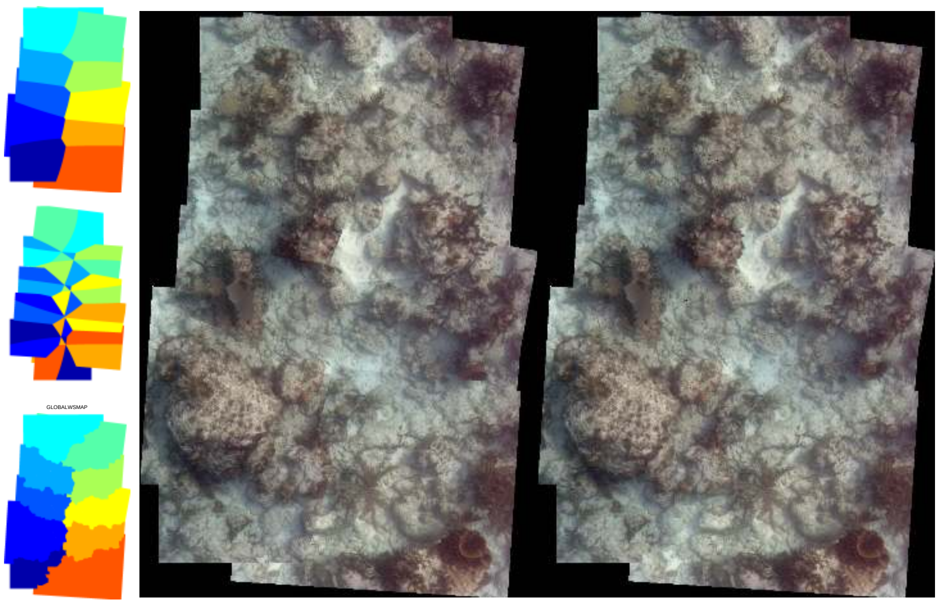


Figure 5: Underwater sequence – First closest map (left top), second closest map (left middle) and the watershed blending contribution map (left bottom). Mosaic from the closest contribution (center) and mosaic from watershed blending (right).

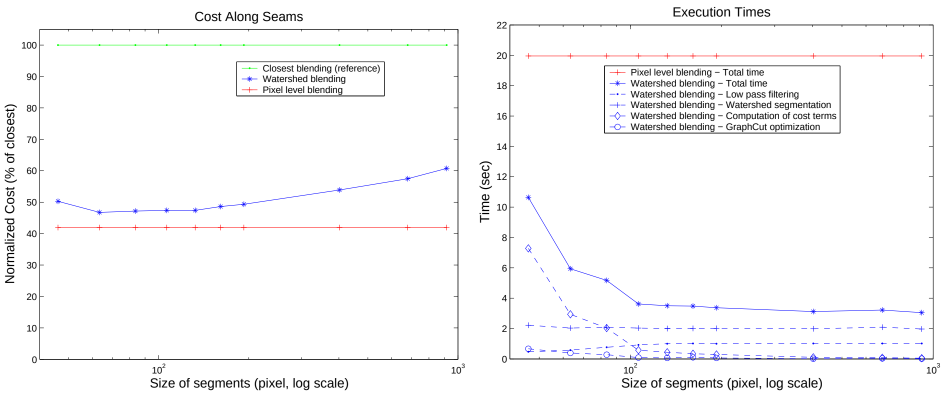


Figure 6: Effect of the average size of segments on the cost (left) and on the execution times (right). The partial execution times of the watershed blending are represented as dotted lines.

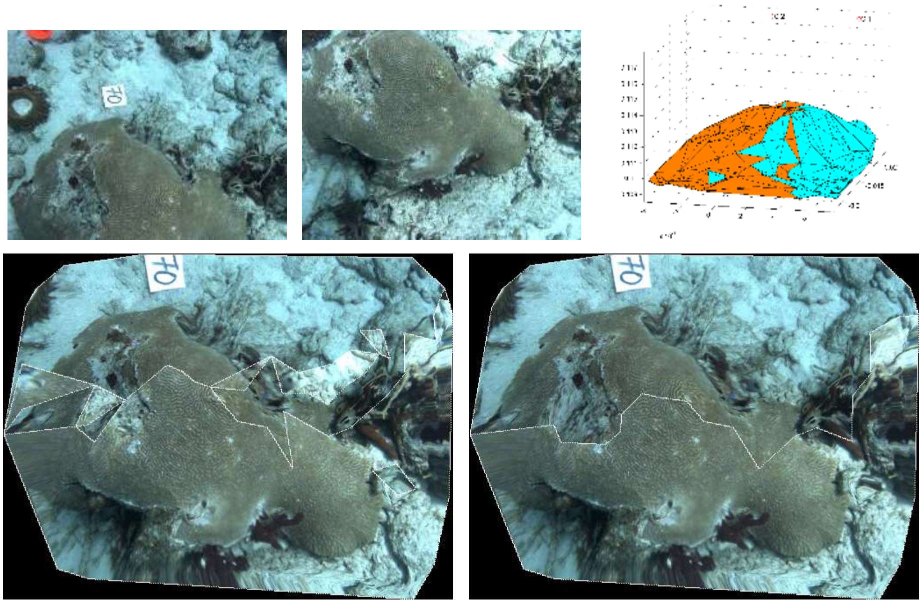


Figure 7: Example of image blending over a 3D model – The upper row shows two original images (left and center) selected to provide texture to the faceted model for the 3D structure (top right). The faces are color–coded according to the geometric criterion of minimum angle between face normals and camera centers (represented as red dots). The lower row contains a ortho–rectified view of the scene using just the geometric criterion (left) and the graph-cut solution combining geometric and photometric criteria (right). The seams are marked by dotted white lines.

Figure 6 illustrates the effect of varying the average segment size on the total execution times and the seam costs for both methods. The execution time for the watershed blending has a significant drop for segments less than 100 pixels and is approximately constant above than value, where it is roughly 6 times faster that the pixel–level blending. The seam cost is defined as the sum of absolute image differences along the seam. As a base–line for comparison, we consider the cost associated with using the closest-contribution (Figure 5 (center)), and normalize the results by it. The seam cost is approximately 6% higher for segments less than 100 pixels and grows up to 34% for segments around 900 pixels.

The right hand side of figure 6 illustrates the execution times of the several components of the watershed method. For segments less than 80 pixels the computation of the cost terms is the dominant term. This term primarily depends on the accumulated length of the watershed boundaries, which in turn depends on the number of segments and their smoothness.

A good compromise between execution times and seam cost is obtained for watershed segments of around 100 pixels. This is achieved without any noticeable effects on the seam quality. Although the threshold for detecting visible seams may vary from person to person, in this example the seams only became apparent for normalized costs of more than 80%. In conclusion, the speed increase of the graph-cut minimization easily offsets the added computational cost of the watershed segmentation, even for small segments of tens of pixels.

5 Extension to image blending in 3–D

The approach in this paper can be suitably extended to image blending over 3–D surfaces. This section outlines an extension, and provides an illustrative example using a 3D relief model of a coral head.

The surface model was estimated from matched point projections over a set of 6 underwater images, using standard structure–from–motion techniques. This resulted in a planar patch approximation to the real surface, comprising 273 triangles. Two images were selected to provide texture to the 3D model. The problem was cast an optimization problem, to balance both geometric and photometric criteria. The chosen geometric criterion is the minimal angle between the normal to each triangle and the vector uniting the center of the triangle to the camera optical center. It promotes minimum texture distortion, by choosing the least slanted image to contribute to each 3D triangle. As a photometric criterion, the difference of intensities along common edges of the triangles was used [3]. The top row of Figure 7 show a view of the 3D model of the scene, where the faces are color–coded to illustrate the geometric criterion. The roughness of the surface results in a small number of faces being separated from the two main regions. The creation of a ortho–rectified view of the scene using just the geometric criterion leads to visible seams (Figure 7, lower–left). The visibility of the seams is greatly reduced by combining the photometric criterion and obtaining a graph cut solution (Figure 7, lower–right).

[3] Given the small size of the triangles, the watershed segmentation was not performed for this example.

6 Conclusions

This paper presented a new approach for automated blending of registered images to create a mosaic. A novel aspect is the use of watershed segmentation in the context of image blending. Instead of optimizing over the entire set of pixels in the intersection zone, preprocessing with the watershed transform led to a reduction of the search space for finding the boundaries between images while still ensuring that the boundaries of each segment would be along low difference areas. Results were presented for 3 challenging image sets, with moving objects and unaccounted 3D structure. An extension of the approach was also illustrated for the case of image blending over 3D surfaces.

The proposed method has several advantages for automated mosaic creation. The use of graph cuts over image pairs guarantees the globally optimal solution for each intersection region. The independence of such regions makes the algorithm suitable for parallel implementation. The separated use of the geometric and photometric criteria eliminates the need for a weighting parameter. Finally, the simple input requirements (closest and second closest image index and texture maps) is memory-efficient, enabling this technique to scale to large mosaics.

References

[1] S. Peleg A. Levin, A. Zomet and Y. Weiss. Seamless image stitching in the gradient domain. In *Proc. of the European Conference on Computer Vision (ECCV04)*, Prague, Czech Republic, May 2004.

[2] A. Agarwala, M. Dontcheva, M. Agrawala, S. Drucker, A. Colburn, B. Curless, D. Salesin, and M. Cohen. Interactive digital photomontage. In *Proc. SIGGRAPH04*, August 2004.

[3] A. Baumberg. Blending images for texturing 3d models. In *Proc. of the British Machine Vision Conference*, Cardiff, UK, September 2002.

[4] P. Burt and E. Adelson. A multiresolution spline with application to image mosaics. *ACM Trans. Graph.*, 2(4):217–236, 1983.

[5] J. Davis. Mosaics of scenes with moving objects. In *Proc. of the Conference on Computer Vision and Pattern Recognition*, Santa Barbara, CA, USA, June 1998.

[6] A. Efros and W. Freeman. Image quilting for texture synthesis and transfer. *Proceedings of SIGGRAPH 2001*, pages 341–346, August 2001.

[7] D. Greig, B. Porteous, , and A. Seheult. Exact maximum a posteriori estimation for binary images. *J. Royal Statistical Soc.*, 51(2):271–279, 1989. Series B.

[8] V. Kolmogorov and R. Zabih. What energy functions can be minimizedvia graph cuts? *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(2):147–159, 2004.

[9] Y. Li, J. Sun, C.-K. Tang, and H.-Y. Shum. Lazy snapping. *ACM Trans. Graph.*, 23(3):303–308, 2004.

[10] L. Najman and M. Schmitt. Geodesic saliency of watershed contours and hierarchical segmentation. *Proc. IEEE Trans. Pattern Analysis and Machine Intelligence*, 18(12):1163–1173, Dec. 1996.

[11] M. Uyttendaele, A. Eden, and R. Szeliski. Eliminating ghosting and exposure artifacts in image mosaics. In *Proc. Int. Conf. on Comp. Vision and Patt. Recog. (CVPR2001)*, pages 509–516, 2001.

[12] L. Vincent and P. Soille. Watersheds in digital spaces: an efficient algorithm based on immersion simulations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(6):583–598, 1991.