Image and Vision Computing xxx (2008) xxx–xxx

Contents lists available at ScienceDirect

Image and Vision Computing

journal homepage: www.elsevier.com/locate/imavis

# Fast image blending using watersheds and graph cuts ☆

Nuno Gracias [a,*], Mohammad Mahoor [b], Shahriar Negahdaripour [b], Arthur Gleason [c]

[a] EIA Department, Ed. PIV, University of Girona, Girona 17003, Spain
[b] ECE Department, University of Miami, Coral Gables, FL 33124, USA
[c] RSMAS – MGG, University of Miami, Miami, FL 33149-1098, USA

ARTICLE INFO

*Article history:*
Received 29 January 2007
Received in revised form 12 December 2007
Accepted 24 April 2008
Available online xxxx

*Keywords:*
Image mosaicing
Watershed segmentation
Graph cuts
3-D texture blending

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 on image differences 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. Watershed transform seeks areas of low difference when creating boundaries of each segment. Constraining the overall cutting lines to be a sequence of watershed segment boundaries results in significant reduction of search space. The solution is found efficiently via graph cut, using a photometric criterion. 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 leads to reduced memory requirements and a compact storage of the input data. Finally, it allows the efficient creation of large mosaics, without user intervention. We illustrate the performance of the approach on image sequences with prominent 3-D 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 an 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 3-D content. A primary 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.

The paper is organized as follows. The rest of this section overviews relevant work on image blending and provides some background on two techniques that are central to this paper, the watershed transform and graph cut optimization. Section 2 introduces the watershed blending approach for the case of two images, followed by the extension to multiple images. Section 3 illustrates the application on selected data sets. Sections 4 and 5 present relevant quantitative comparisons of our method against both pixel-level and multi-label graph cut approaches. Section 6 extends the method to the blending of textures over 3-D surfaces. Finally Section 7 provides a summary and conclusions.

### 1.1. Related work and background

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

☆ This work has been partially supported by DoD/DOE/EPA SERDP Proj. CS – 1333, by the Portuguese Fundação para a Ciência e Tecnologia BPD/14602/2003, and by the Ramon y Cajal Program of the Spanish Ministry of Science.

* Corresponding author. Tel.: +34 972418013; fax: +34 972418976.
*E-mail address:* ngracias@isr.ist.utl.pt (N. Gracias).

0262-8856/$ - see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.imavis.2008.04.014

by smoothing. A traditional approach is multiresolution splining by Burt and Adelson [2], where the images are decomposed into a set of band-pass components, and joined using a weighted average over a transition zone which is inversely proportional in size to the band frequency. Multiresolution splining has been extensively used for blending images in 2-D panoramas [3] and 3-D models [4]. Recent transition smoothing approaches include the multiresolution blending using wavelets [5] and the gradient domain blending [1,6]. Gradient domain methods reduce the inconsistencies due to illumination changes and variations in the photometric response of the cameras since dissimilarities in the gradients are invariant to the average image intensity. However, they require recovering the blended image from a gradient description. For 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 [7,8]. The method proposed in this paper fits in this class, and is therefore related to previous work. Davis [8] describes an image blending method for sequences with moving objects over a static background. It computes the relative photometric difference between two images, and searches for the dividing boundary along the low intensity of the difference image using Dijkstra's algorithm. No details are given on how the starting and ending points of the boundaries are defined. These points are required for the use of Dijkstra's algorithm. Uyttendaele et al. [9] 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 of RODs. Agarwala et al. [6] use graph cuts to find the contribution regions among several images where each pixel is treated independently. Pixel labeling 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 labeling 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 graph-cut operation inside each multi-label alpha-expansion iteration. Section 5 demonstrates that our approach is more suited to the processing of large sets of images, without user intervention.

#### 1.1.1. Watershed transform

The watershed transform [10] 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 reader is referred to [11] for an extended review of several existing definitions of the watershed transform and associated algorithms, with both sequential and parallel implementations.

The image processing literature provides a large number of application examples of watersheds, such as 2-D/3-D region and surface segmentation [4] and in contour detection [12,13]. Nguyen et al. [14] address the issue of lack of smoothness control of the segmentation result. The authors establish a connection between the watershed segmentation and energy-based segmentation methods. Using a distance-based definition for the watershed line, they derive an energy function whose minimization is equivalent to the watershed segmentation. The main advantage is to take into account a priori knowledge of boundary smoothness or shape. Li et al. [13], 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. Our paper aims at the same benefit, but in the different domain of automated mosaic blending. In [15], an algorithm is developed for the compositing of grayscale images based on several morphological operators leading to the definition of watershed lines over a correlation image. As presented, the algorithm is limited to the blending of 2 images, which restricts its applicability. Related to this, Soille [16] proposes a method based on the segmentation of the gradient intensity, which is suited to process an arbitrary number of large multispectral images, provided that none of the image domains is completely included in the union of the domains of all others. Both methods in [15,16] use a marker-controlled segmentation approach to propagate labels until a dividing line is found. The label propagation paradigm uses local intensity or intensity gradients to guide the dividing lines towards object contours. This condition promotes sensitivity to strong radiometric differences, caused for example by transient objects, non-rigid structure or inaccuracies in the registration. It also stresses the relevance of removing spurious or dynamic features [16], either automatically or through user interaction. This contrasts with the approach in our paper where we explicitly define the cost associated with the visibility of the seam. A global minimum is found for each pair of overlapping images, thus strongly reducing the sensitivity to radiometric and registration artifacts. Additionally, we show that this approach can be easily extended to deal with blending over 3-D surfaces.

#### 1.1.2. Graph cuts

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 basic technique is to construct a specialized graph representing for the energy function, such that the minimum cut on the graph also minimizes the energy (either globally or locally). The minimum cut, in turn, can be computed very efficiently by maximum flow algorithms. These methods have been successfully used for a wide variety of vision problems, including image restoration [17,18], stereo and motion [19–21], image synthesis [22] and more recently in image blending [6,23].

The classical use of graph cuts in computer vision is to solve pixel-labeling problems. The input is a set of pixels $P$ and a set of labels $L$. The goal is to find a labeling $f$ (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). Graph cut techniques often provide solutions with important theoretical properties, namely the global minimum [17,24,25,20], or a local minimum that is within a known factor of the global minimum. The later is also referred as a local minimum in a strong sense [18]. For the case of binary labels, Eq. 1 is a particular case of the *Ising* model, and the global minimum can be found over a single graph cut computation [24].

Please cite this article in press as: N. Gracias et al., Fast image blending using watersheds and graph cuts, Image Vis. Comput. (2008), doi:10.1016/j.imavis.2008.04.014

## 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 labeling of the segments that minimizes intensity differences along the seam. By *labeling* 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.

Direct application of the watershed algorithm to an image of intensity difference generally results in over-segmentation, i.e., the creation of a large number of very small regions due to the presence of many surface minima. 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 $\sigma$ of 1.4 pixels. The effect of varying $\sigma$ on the performance of the algorithm is presented and discussed in Section 4.

An example of watershed segmentation and blending using two registered images from an underwater sequence of a coral patch is shown in Figs. 1 and 2. Blending using simple geometric criteria is inadequate; the average image (Fig. 1(c)) is blurry, and filling pixels with the contribution from the image with the closest center produces a visible seam (Fig. 1(d)). Fig. 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 (Fig. 2(d)). Although not perfect, it improves greatly over the simple geometric algorithms (Fig. 1(c and d)).

### 2.2. Graph cut labeling

The visibility of the seams can be significantly reduced by penalizing the photometric difference along the seams. A suitable way to reduce such visibility is to cast the assignment of the watershed segments as an energy minimization problem and use graph cuts to find a solution.

For the simple case of 2 images, we start by considering 4 regions illustrated in Fig. 3(a), associated with the image domains and their intersection. 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 pixels are closer to the center of image 1 and secondly closer to image 2. Region $R_{21}$ is defined as the opposite. The union of $R_{12}$ and $R_{21}$ completely defines the intersection of $I_{1w}$ and $I_{2w}$. We refer to $R_{10}$ and $R_{20}$ as the areas outside the intersection region where the mosaic points are closer to the center of image 1 and 2 respectively. All regions $R_{12}$, $R_{21}$, $R_{10}$ and $R_{20}$ are mutually exclusive, i.e., have no intersection.

A given watershed segment $i$ is represented by a binary image mask $S_i$. A possible labeling solution is represented by the vector $L$ of $n$ binary labels (where $n$ is the number of watershed segments), that associates each segment with one of the images. Therefore we have the equivalences $L(i) = 0 \Longleftrightarrow S_i \in U_1$ and

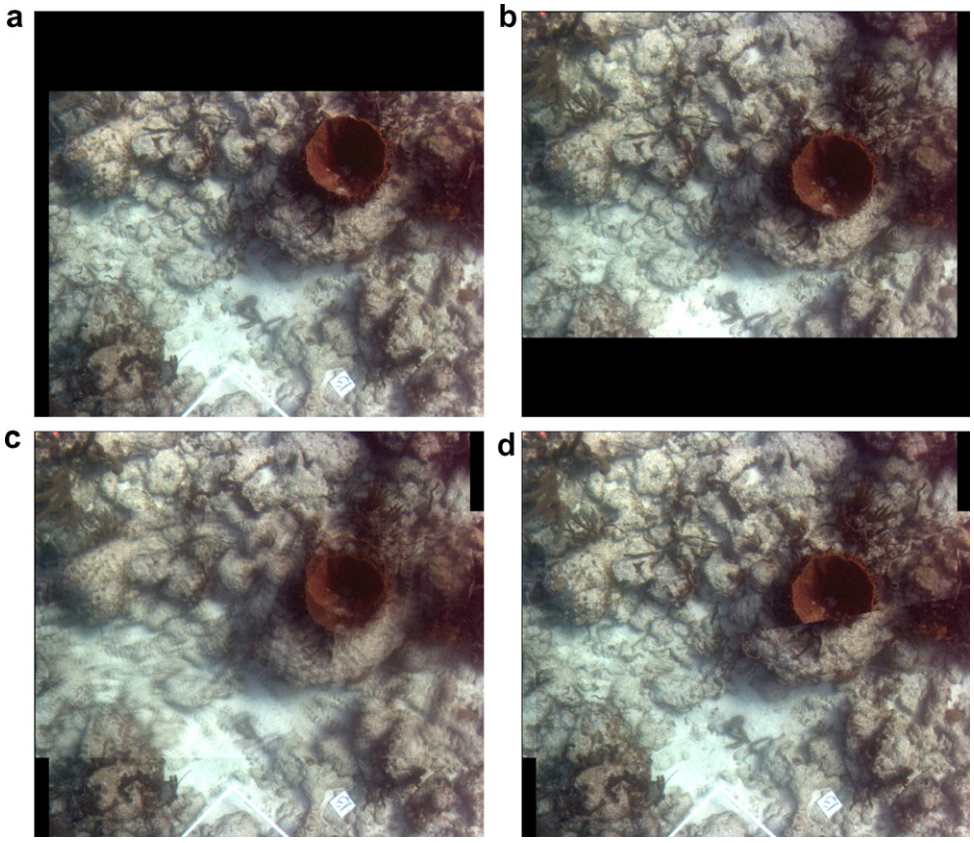


**Fig. 1.** Original images used for the watershed blending example (a and b) and examples of purely geometric blending – average over intersection area (c) and closest to image center (d).

Please cite this article in press as: N. Gracias et al., Fast image blending using watersheds and graph cuts, Image Vis. Comput. (2008), doi:10.1016/j.imavis.2008.04.014

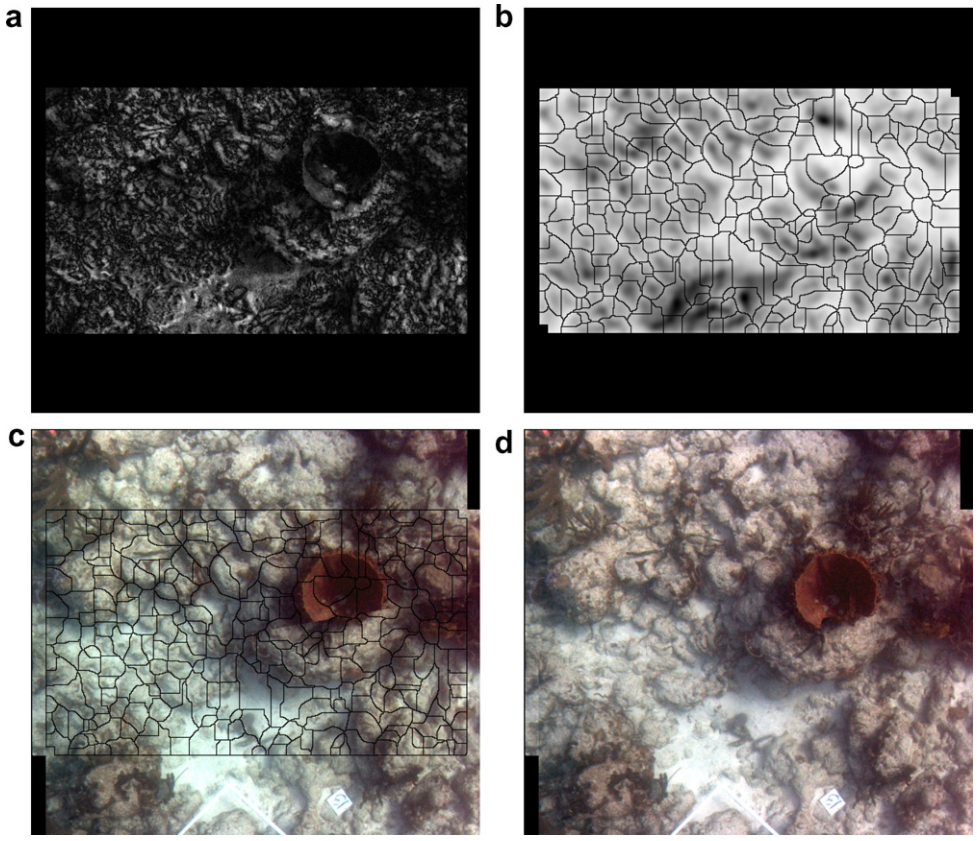


**Fig. 2.** Absolute value of the grey-level image difference (a), watershed segmentation over the inverted, low-pass filtered difference (b), and segmentation outline over the closest-to-center blending (c). Simple watershed blending obtained by associating each segment to the closest image center (d).

$L(i) = 1 \iff S_i \in U_2$, where $U_1$ and $U_2$ represent the sets of segments that belong to image 1 and 2, respectively.

We denote $D_1$ and $D_2$ as the $n$-vectors containing the costs of assigning each segment to each image. Conversely, $V$ is the $n \times n$ matrix such that $V(i,j)$ contains the cost of having $S_i$ and $S_j$ associated with different images. We define $D_{diff}(S_i, S_j)$ as the vector of intensity differences between $I_{1w}$ and $I_{2w}$ along the *common boundary* of regions $S_i$ and $S_j$. For color images, each element of $D_{diff}$ contains the largest difference over all channels. If $S_i$ and $S_j$ are not neighbors (i.e., no common boundary) then $D_{diff}(S_i, S_j)$ is null.

Having introduced the notation, we define the cost as follows. 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 along common boundaries,

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

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

$$C(L) = \sum_i (D_1(i) \cdot \bar{L}(i) + D_2(i) \cdot L(i)) + \sum_{i,j} V(i,j) \cdot (L(i) \otimes L(j)) \qquad (2)$$

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

The cost function above is in the form of Eq. 1. An efficient algorithm exists based on a max-flow approach, which guaranties global minimization [26]. The condition for applicability and guarantee of global minimum is that the cost function be *regular*, defined as

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

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 $V_{ij}(0,0) = V_{ij}(1,1) = 0$, $V_{ij}(0,1) \geqslant 0$ and $V_{ij}(1,0) \geqslant 0$ for all $i, j$.

Fig. 3 illustrates the outcome of the process using $p = 1$. Comparing to the simple watershed blending result from the previous section (Fig. 2(d)), two main improvements are noticeable: (1) photometric criterion helps to preserve the most prominent scene features such as the 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 indices to the second closest image. Fig. 5 (top left and top center) provides a graphical representation of these regions for one of our test sequences.

Let $R_{ij}$ denote the 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

Please cite this article in press as: N. Gracias et al., Fast image blending using watersheds and graph cuts, Image Vis. Comput. (2008), doi:10.1016/j.imavis.2008.04.014

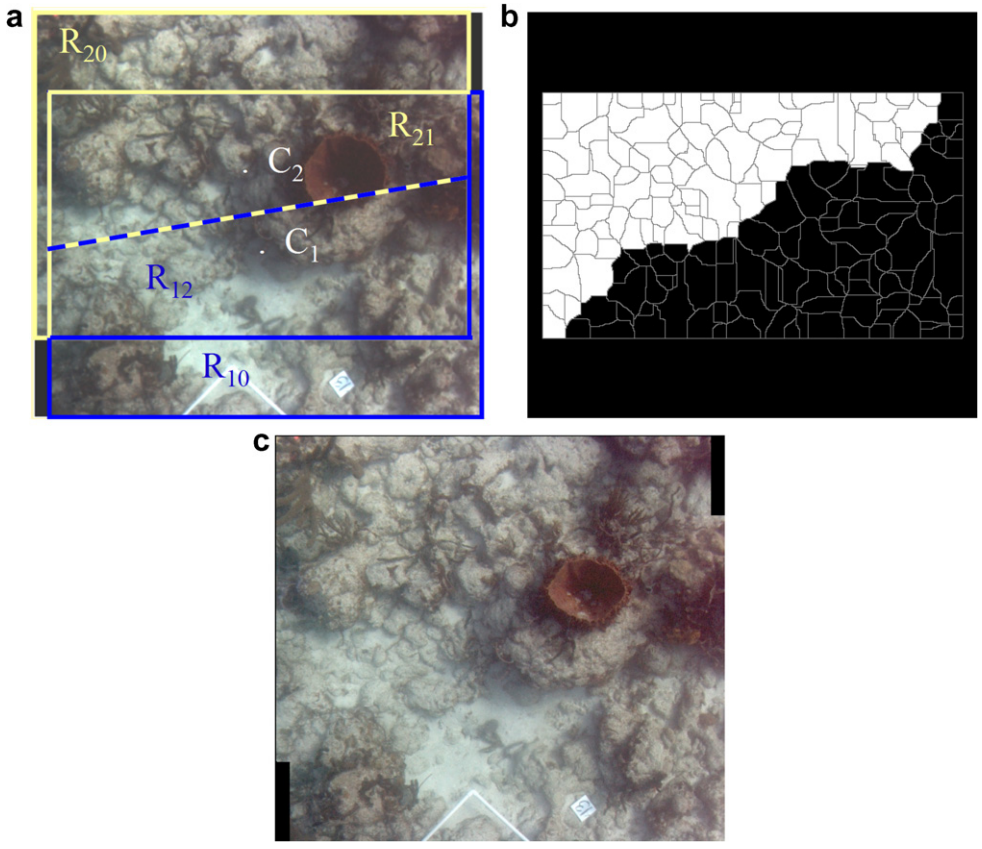


**Fig. 3.** Example of graph cut labeling over the watershed segments – regions involved in computing the cost function (a), optimal labeling (b) and resulting blending (c).

**Fig. 4.** Four of the original images from the outdoor panoramic sequence (top row) and watershed blending result (bottom image), cropped over the area containing moving people.

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 input data for the watershed blending 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 major importance when processing large data sets and large mosaics.

## 3. Results

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

Please cite this article in press as: N. Gracias et al., Fast image blending using watersheds and graph cuts, Image Vis. Comput. (2008), doi:10.1016/j.imavis.2008.04.014

The first data set is a panoramic sequence of an outdoor scene,[1] captured under rotation, with multiple moving pedestrians. It was initially used by Uyttendaele et al. [9] and more recently by Agarwala et al. [6]. The sequence is available as a stack of seven images, already warped into the mosaic frame. Since the location of the image center was not readily available for computing the closest and second closest maps, it was estimated by computing the homography that un-warps each image domain into a standard image rectangle of 3:2 proportions. The central point of the rectangle was considered as the image center. Alternatively one could use a distance transform directly over the binary mask of the image domain, for the same purpose. Fig. 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. An example of this is the area over the 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. The motion between pairs of images was estimated based on a planar model for the environment [27,28]. The use of this model resulted in registration inaccuracies over the areas of strong 3-D structure. The first closest map, second closest map and the watershed blending contribution map are shown in the upper row of Fig. 5. The lower row shows the mosaics resulting from the closest contribution and watershed segmentation. The watershed/graph cut blending provides a realistic rendering of the scene, by cutting around the prominent benthic structures (such as rocks and coral heads). The areas where the improvement is most visible are the center and lower left regions.

## 4. Comparison to pixel-level binary 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 Fig. 5 of $1172 \times 795$ pixel, comparative results were obtained for several values of $\sigma$ (the standard deviation of the low pass Gaussian filter), in the range $\sigma \in [0.85]$ pixel. Fig. 6 shows the effect of $\sigma$ on the number of watershed segments and their average size. The curves are related by their product

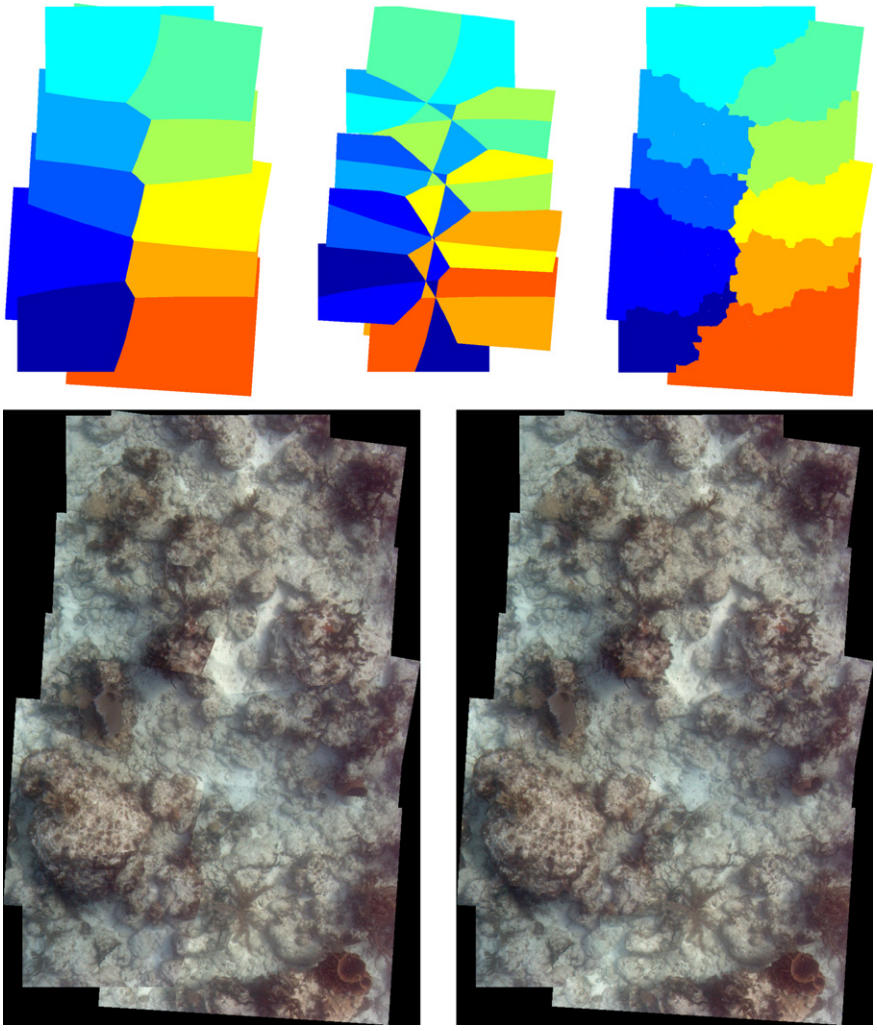

**Fig. 5.** Underwater sequence – first closest map (top left), second closest map (top center) and the watershed blending contribution map (top right). Mosaic from the closest contribution (bottom left) and mosaic from watershed blending (bottom right). The areas of most visible improvement are over the center and lower left.

Please cite this article in press as: N. Gracias et al., Fast image blending using watersheds and graph cuts, Image Vis. Comput. (2008), doi:10.1016/j.imavis.2008.04.014

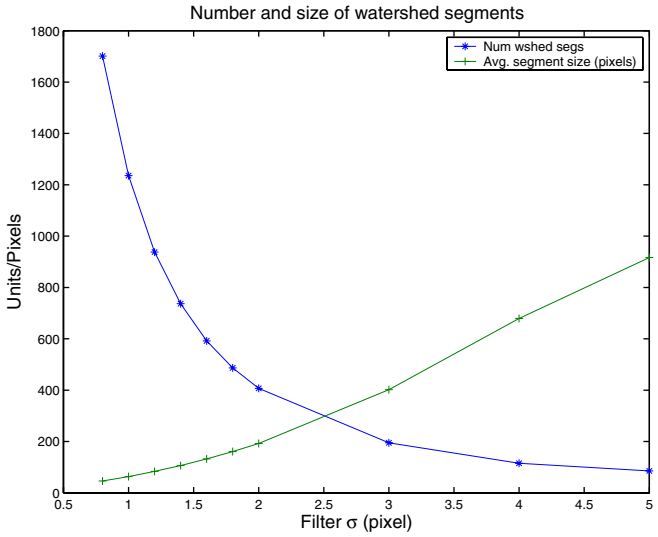


**Fig. 6.** Effect of $\sigma$, the standard deviation in pixels of the Gaussian low pass filter used to smooth the difference images, on the average size and number of segments used in the watershed blending.

being equal to the size of the mosaic. The size of the watershed segments grows approximately linearly with $\sigma$ and ranges from 46 to 920 pixels.

Fig. 7 illustrates the effect of varying the average segment size on the seam costs and total execution times for both methods. 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 (Fig. 5, lower left), and normalize the results by it. The seam cost is approximately 6% higher for segments less than 100 pixels and grows up to 18% for segments around 900 pixels.

The execution time for the watershed blending decreases significantly with increasing segment sizes up to 100 pixels and is approximately constant above that value, where it is roughly 6 times faster than the pixel-level blending. 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. The combined computation time for low pass filtering, watershed segmentation and graph cut optimization is approximately constant and invariant with the segment size.

All execution times refer to a 3.0 GHz Intel Pentium processor running Matlab 6.5. The components of the watershed/graph cut blending method were implemented in C, with the exception of the low pass filtering and the watershed segmentation for which efficient implementations exist in Matlab. C++ code for constructing the graphs and finding the minimal cuts is available on the internet [26].

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.

One may ask if the final boundary from pixel labeling is generally somewhere near the watershed result. Having studied this issue, we have determined that the watershed boundary can be close to the pixel-level boundary for some cases and relatively far away for others. However, for all cases, the cost function of the watershed seam pixel labeling is typically flat in the vicinity of the minimum, suggesting the presence of different solutions with relatively similar costs for the seam placement problem.

## 5. Comparison to multi-label graph cut blending

In the context of image blending, the use of multi-label graph cuts was introduced recently by Agarwala et al. [6]. Pixel labeling is performed by optimizing simultaneously over all images.

This section presents a comparison of our method against multi-label graph cut optimization. For this comparison we build upon an efficient implementation recently used for comparing and evaluating methods for energy minimization in Markov random fields [29,30]. Source code is available on the web [29]. The required adaptation dealt only with the definition of the assignment and

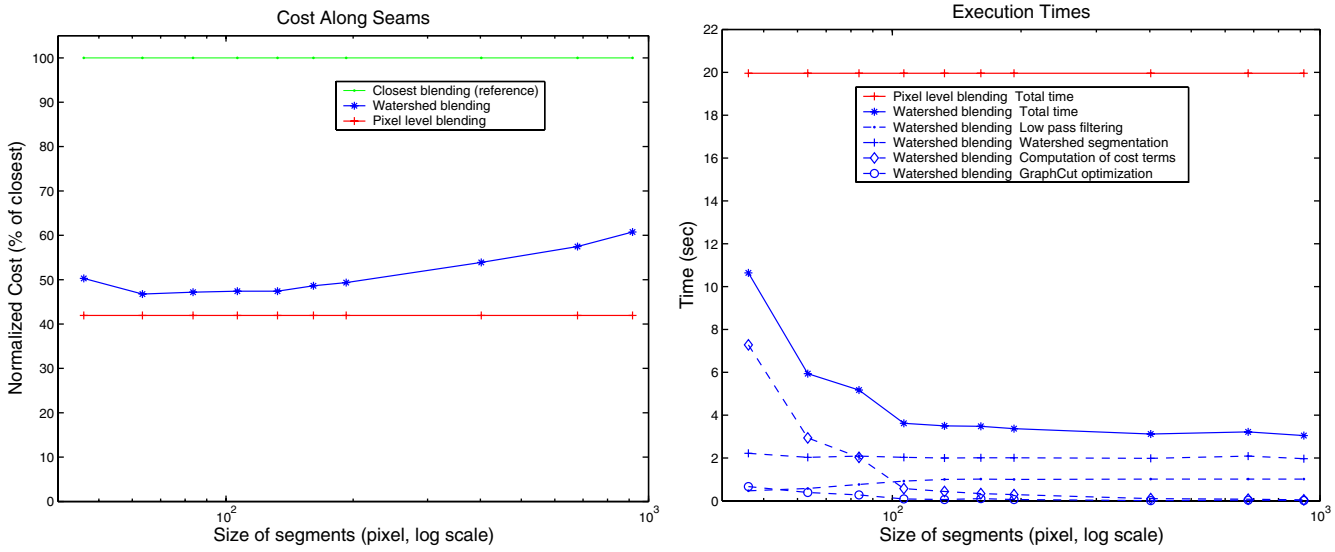


**Fig. 7.** Effect of the average size of segments on the cost (left) and on the execution times (right). The partial execution times of the watershed blending are represented as dotted lines.

Please cite this article in press as: N. Gracias et al., Fast image blending using watersheds and graph cuts, Image Vis. Comput. (2008), doi:10.1016/j.imavis.2008.04.014

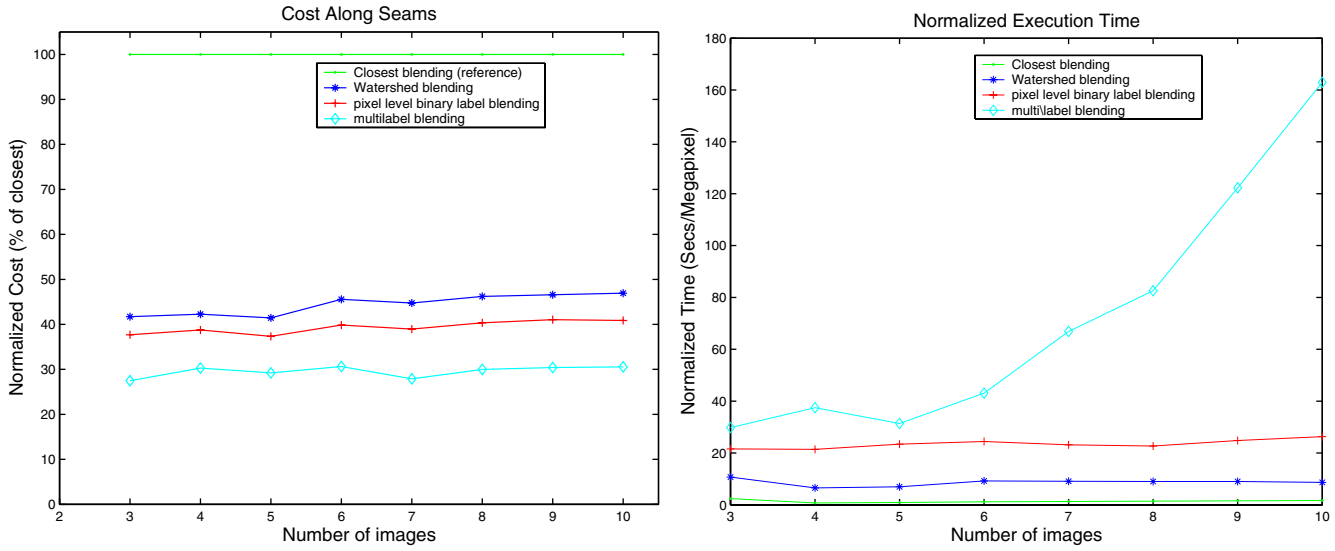


**Fig. 8.** Performance comparison of different approaches, under increasing number of images, on the normalized cost (left) and on the total execution time (right).

interaction costs. These costs mirror the"seam objective of matching colors" as described in [6]. This objective assigns a cost based on the sum of absolute color intensity differences along the seam, and attributes a large penalty to pixels from invalid images. Similarly, an alpha-expansion algorithm was employed, which terminates when no expansion is able to reduce the cost.

To illustrate the performance of both approaches under similar conditions, we conducted experiments using the two image sequences of Section 3. The results of each approach were similar for both sequences. Due to space limitations, we present detailed quantitative results only for the underwater sequence, since it comprises a larger number of images. The test consisted of running the four different methods (closest blending, pixel-level binary graph cut, multi-label graph cut and our approach) over subsets ranging from 3 to 10 images. The results were evaluated in terms of cost and execution time.

The results regarding normalized cost are plotted on the left hand side of Fig. 8. The multi-label graph cut method obtains the best results with an average slightly above 30%, followed by the pixel-level binary graph cut (39%) and the watershed method (44%). This is due to the fact that the optimization over multiple labels is less constrained than the other two methods, which optimize over the 2 closest images. All three methods are significantly below the empirical visibility threshold of 80%.

The right hand side plot in Fig. 8 presents the total execution times which are normalized by the area of the mosaic. The execution times for the pixel-level binary graph cut and for our approach include the creation of the closest and second closest maps, since these are pre-requisites of both methods.

The most distinctive feature is the steep increase of the computation time for the multi-label graph cut method. This is justified by the fact that the increase on the number of images (labels) results in the increase of the number of required alpha-expansions. For $N$ labels, each expansion implies solving $N$ elementary graph cut operations [26]. Since each label competes against all others, the elementary graph cut operations are performed in large graphs, proportional to the area of the mosaic. This contrasts with the proposed scheme of using binary graph cuts (both watershed and pixel-level), where the increase of the execution time is significantly lower. For the 10 image case, the ratio of execution speeds between the multi-label graph cut and the watershed graph cut method is more than ten fold. This example illustrates the appropriateness of the proposed method for dealing with the blending of large image sets.

## 6. 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 describes an extension, and provides an illustrative example using a 3-D relief model of a coral colony.

The surface model was estimated from matched point projections over a set of six underwater images, using standard structure-from-motion techniques. This resulted in a planar patch approximation to the real surface, comprising 273 triangles (Fig. 9, top).

Two images were selected to provide texture (Fig. 9, middle row). Prior to blending, the images were independently projected into the 3-D surface. The two resulting textured surfaces were flattened [31] by orthographic projection onto a plane (Fig. 9, bottom row). This plane was defined as being perpendicular to the optical axis of the most central image (from the set of six), and passing through the centroid of the 3-D points. The resulting flattened textures are close approximations to ortho-rectified views since, for this dataset, the plane is approximately parallel to the average plane of the sea bottom.

Watershed segmentation was applied to the difference of the flattened textures, as described in Section 2. The normal of the 3-D surface was computed at the centroid of each segment.

The 3-D blending problem was cast an optimization problem, to balance both geometric and photometric criteria. The chosen geometric criterion was the minimal angle between the normal of the surface at the centroid of each segment and the vector uniting the centroid to the camera optical center. It promotes minimum texture distortion, by choosing the least slanted image to contribute to each segment. As a photometric criterion, the difference of intensities along common borders of the segments was used.

Fig. 10 shows the results of blending over the 3-D surface and the improvement obtained by combining both criteria. The top row illustrates the assignment using the geometric criterion alone. The roughness of the surface results in a small number of faces being separated from the two main regions and in visible seams. It can be argued that the sharp edges and the relatively small amount of triangles contribute to the visibility of the seam. To test

Please cite this article in press as: N. Gracias et al., Fast image blending using watersheds and graph cuts, Image Vis. Comput. (2008), doi:10.1016/j.imavis.2008.04.014

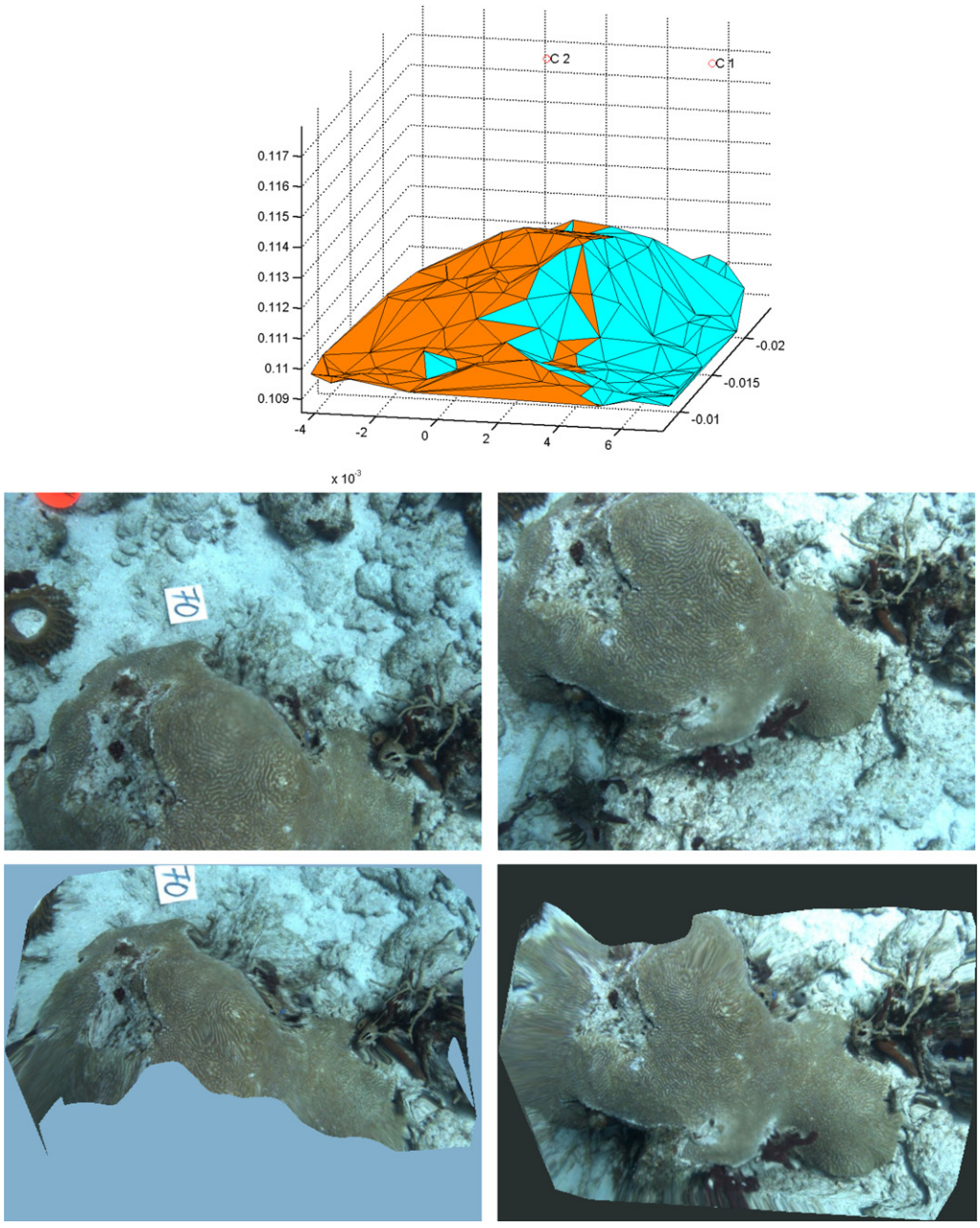


**Fig. 9.** Image blending over a 3-D model – the upper figure illustrates the 3-D faceted surface displayed as an oblique view. The faces are color-coded according to the geometric criterion of minimum angle between face normals and the two camera centers (marked as $C_1$ and $C_2$). Two images were selected to provide the texture (middle row). Prior to blending, the textures were flattened onto a common planar surface, which approximates ortho-rectified views (lower row).

this, the blending result in the middle row was created, using watershed segments to break down the size of the triangles. Similarly to the top row, only the geometric criterion was used to assign the segments to each image. Although the visibility decreases considerably, the seams are still noticeable in the top right part of the texture maps and on the central region of the coral colony. In the lower row, the visibility of the seams is eliminated by combining both criteria.

## 7. 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 and graph cuts 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 2 challenging image sets, with moving objects and unaccounted 3-D structure.

A central idea in this paper is that watershed segmentation greatly reduces the search space for finding contribution boundaries, when compared to searching over all individual pixels in the intersection zone. To support this we presented quantitative comparisons of this method against pixel-level blending, for both cases of binary or multi-label graph cut optimization. Our approach compares very favorably in terms of time complexity, without noticeable degradation of the seam quality.

Please cite this article in press as: N. Gracias et al., Fast image blending using watersheds and graph cuts, Image Vis. Comput. (2008), doi:10.1016/j.imavis.2008.04.014

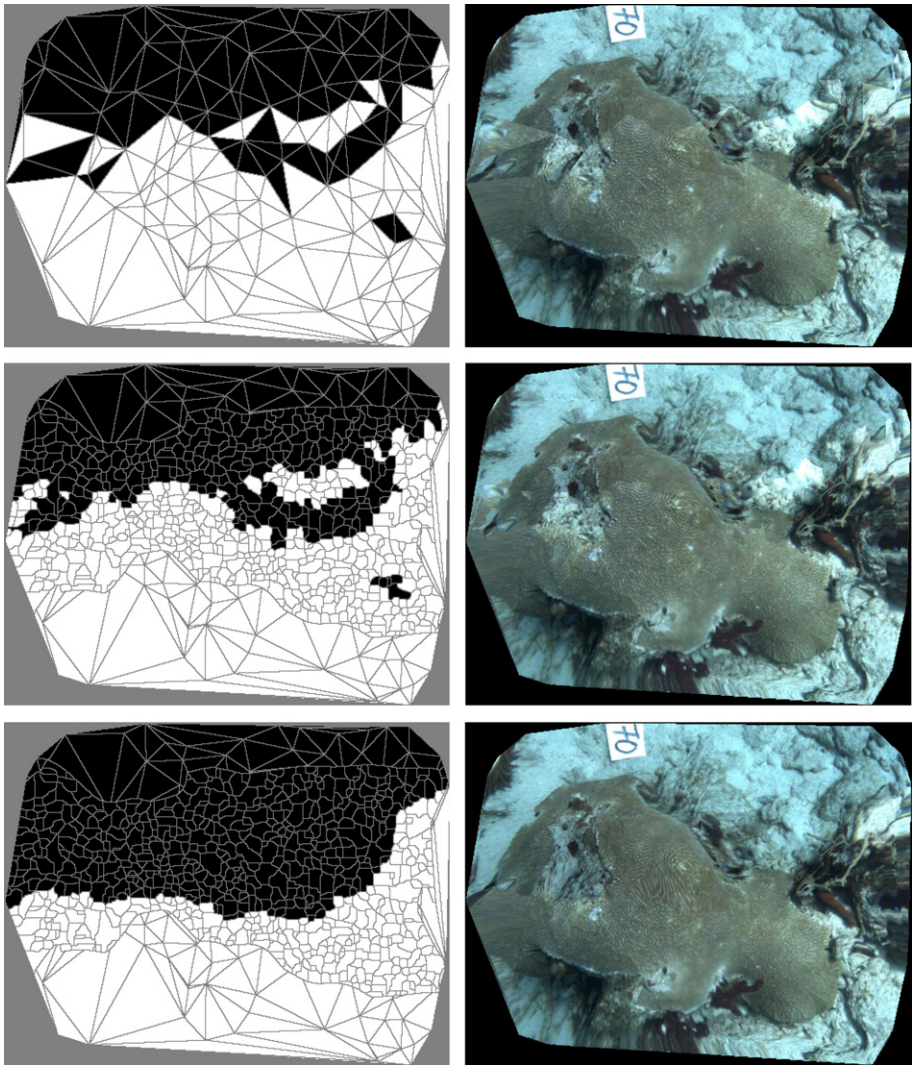

**Fig. 10.** Example of image blending over a 3-D model – the upper row shows the result of applying the only the geometry criterion over the original faceted model. The middle row also uses the geometric criterion alone, but the common area visible in both images was further divided using watershed segmentation applied to the image difference. The lower row contains the graph-cut solution combining geometric and photometric criteria. For all results, the diagrams of segment assignments are shown on the left and the corresponding blending results on the right.

An extension of the approach was proposed for the case of image blending over 3-D surfaces. A result using 3-D structure illustrated the versatility of the method for integrating distinct geometric and photometric criteria.

The proposed method has several advantages for automated mosaic creation. The division of the mosaic space into ROIIs, where each pair of images can be treated independently, makes the algorithm suitable for parallel implementation. The use of graph cuts over image pairs guarantees an optimal solution over each intersection region. Finally, the separated use of the geometric and photometric criteria leads to a very compact memory usage. All the input data for the watershed graph cut blending is stored in just four arrays (closest and second closest image index, and corresponding texture maps). Such memory efficiency associated with execution speed, enables this technique to scale to large mosaics.

## References

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

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

[3] M. Brown, D.G. Lowe, Recognising panoramas. in: ICCV'03: Proceedings of the Ninth IEEE International Conference on Computer Vision, IEEE Computer Society, Washington, DC, USA, 2003, p. 1218.

[4] A. Baumberg, Blending images for texturing 3D models, in: Proceedings of the British Machine Vision Conference, Cardiff, UK, September 2002.

[5] M. Su, W. Hwang, K. Cheng, Analysis on multiresolution mosaic images, IEEE Trans. Image Process. 13 (7) (2004) 952–959. July.

[6] A. Agarwala, M. Dontcheva, M. Agrawala, S. Drucker, A. Colburn, B. Curless, D. Salesin, M. Cohen, Interactive digital photomontage, in: Proceedings of SIGGRAPH04, August 2004.

[7] A. Efros, W. Freeman, Image quilting for texture synthesis and transfer, in: Proceedings of SIGGRAPH 2001, August 2001, pp. 341–346.

[8] J. Davis, Mosaics of scenes with moving objects, in: Proceedings of the Conference on Computer Vision and Pattern Recognition, Santa Barbara, CA, USA, June 1998.

[9] M. Uyttendaele, A. Eden, R. Szeliski, Eliminating ghosting and exposure artifacts in image mosaics, in: Proceedings of the International Conference on Computer Vision and Pattern Recognition (CVPR2001), 2001, pp. 509–516.

[10] L. Vincent, P. Soille, Watersheds in digital spaces: an efficient algorithm based on immersion simulations, IEEE Trans. Pattern Anal. Mach. Intell. 13 (6) (1991) 583–598.

[11] J. Roerdink, Arnold Meijster, The watershed transform: definitions, algorithms and parallelization strategies, Fundam. Inform. 41 (1–2) (2000) 187–228.

[12] L. Najman, M. Schmitt, Geodesic saliency of watershed contours and hierarchical segmentation, Proc. IEEE Trans. Pattern Anal. Mach. Intell. 18 (12) (1996) 1163–1173. December.

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

Please cite this article in press as: N. Gracias et al., Fast image blending using watersheds and graph cuts, Image Vis. Comput. (2008), doi:10.1016/j.imavis.2008.04.014

[14] H. Nguyen, M. Worring, R. van den Boomgaard, Watersnakes: energy-driven watershed segmentation, IEEE Trans. PAMI 25 (3) (2003) 330–342.
[15] F. Araujo Jr., N. Leite, A morphological algorithm for photomosaicking, in: Proceedingsof the Eighth European Signal Processing Conference (EURASIP), Trieste, Italy, 1996, pp. 181–186.
[16] P. Soille, Morphological image compositing, IEEE Trans. Pattern Anal. Mach. Intell. 28 (5) (2006) 673–683.
[17] Y. Boykov, O. Veksler, R. Zabih, Markov random fields with efficient approximations, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 1998, pp. 648–655.
[18] Y. Boykov, O. Veksler, R. Zabih, Fast approximate energy minimization via graph cuts, Proc. IEEE Trans. Pattern Anal. Mach. Intell. 23 (11) (2001) 1222–1239.
[19] M.H. Lin, Surfaces with Occlusions from Layered Stereo, PhD thesis, Stanford University, December 2002.
[20] S. Roy, Stereo without epipolar lines: a maximum flow formulation, Int. J. Comput. Vis. 1 (2) (1999) 1–15.
[21] S. Roy, I. Cox, A maximum-flow formulation of the *n*-camera stereo correspondence problem, in: International Conference on Computer Vision, 1998.
[22] V. Kwatra, A. Schodl, I. Essa, G. Turk, A. Bobick, Graphcut textures: image and video synthesis using graph cuts, in: ACM Trans. Graphics, Proceedings of the SIGGRAPH 2003, July 2003.
[23] F. Gu, Y. Rzhanov, Optimal image blending for underwater mosaics, in: Proceedings of the IEEE/MTS OCEANS 2006, Boston, MA, USA, September 2006.
[24] D. Greig, B. Porteous, A. Seheult, Exact maximum a posteriori estimation for binary images, J. R. Stat. Soc. Ser. B 51 (2) (1989) 271–279.
[25] H. Ishikawa D. Geiger, Occlusions, discontinuities, and epipolar lines in stereo, in: European Conference on Computer Vision, 1998, pages 232–248.
[26] V. Kolmogorov, R. Zabih, What energy functions can be minimized via graph cuts?, IEEE Trans Pattern Anal. Mach. Intell. 26 (2) (2004) 147–159.
[27] N. Gracias, S. Zwaan, A. Bernardino, J. Santos-Victor, Mosaic based navigation for autonomous underwater vehicles, J. Ocean. Eng. 28 (4) (2003). October.
[28] D. Lirman, N. Gracias, B. Gintert, A. Gleason, R.P. Reid, S. Negahdaripour, P. Kramer, Development and application of a video-mosaic survey technology to document the status of coral reef communities, Environ. Monit. Assess. 159 (2007) 59–73.
[29] R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappen, C. Rother, A comparative study of energy minimization methods for Markov random fields, in: Proceedings of Ninth European Conference on Computer Vision (ECCV 2006), vol. 2, Graz, Austria, May 2006, pp. 16–29.
[30] Y. Boykov, V. Kolmogorov, An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision, Proc. IEEE Trans. Pattern Anal. Mach. Intell. 26 (9) (2004) 1124–1137. September.
[31] Z. Jankó, G. Kós, D. Chetverikov, Creating entirely textured 3d models of real objects using surface flattening, Mach. Graph. Vis. Int. J. 14 (4) (2005) 379–398.

Please cite this article in press as: N. Gracias et al., Fast image blending using watersheds and graph cuts, Image Vis. Comput. (2008), doi:10.1016/j.imavis.2008.04.014