Interactive Digital Photomontage

Aseem Agarwala[1] Mira Dontcheva[1] Maneesh Agrawala[2] Steven Drucker[2] Alex Colburn[2]
Brian Curless[1] David Salesin[1,2] Michael Cohen[2]

[1]University of Washington [2]Microsoft Research

Abstract

We describe an interactive, computer-assisted framework for combining parts of a set of photographs into a single composite picture, a process we call "digital photomontage." Our framework makes use of two techniques primarily: graph-cut optimization, to choose good seams within the constituent images so that they can be combined as seamlessly as possible; and gradient-domain fusion, a process based on Poisson equations, to further reduce any remaining visible artifacts in the composite. Also central to the framework is a suite of interactive tools that allow the user to specify a variety of high-level image objectives, either globally across the image, or locally through a painting-style interface. Image objectives are applied independently at each pixel location and generally involve a function of the pixel values (such as "maximum contrast") drawn from that same location in the set of source images. Typically, a user applies a series of image objectives iteratively in order to create a finished composite. The power of this framework lies in its generality; we show how it can be used for a wide variety of applications, including "selective composites" (for instance, group photos in which everyone looks their best), relighting, extended depth of field, panoramic stitching, clean-plate production, stroboscopic visualization of movement, and time-lapse mosaics.

CR Categories: I.3.6 [Computer Graphics]: Methodology and Techniques—Interaction Techniques; I.4.9 [Image Processing and Computer Vision]: Applications

Keywords: Interactive image editing, image compositing, user-guided optimization

1 Introduction

Seldom does a photograph record what we perceive with our eyes. Often, the scene captured in a photo is quite unexpected — and disappointing — compared to what we believe we have seen. A common example is catching someone with their eyes closed: we almost never consciously perceive an eye blink, and yet, there it is in the photo — "the camera never lies." Our higher cognitive functions constantly mediate our perceptions so that in photography, very often, what you get is decidedly *not* what you perceive. "What you get," generally speaking, is a frozen moment in time, whereas "what you perceive" is some time- and spatially-filtered version of the evolving scene.

In this paper, we look at how digital photography can be used to create photographic images that more accurately convey our subjective impressions — or go beyond them, providing visualizations or a greater degree of artistic expression. Our approach is to utilize multiple photos of a scene, taken with a digital camera, in which some aspect of the scene or camera parameters varies with each photo. (A film camera could also be used, but digital photography makes the process of taking large numbers of exposures particularly easy and inexpensive.) These photographs are then pieced together, via an interactive system, to create a single photograph that better conveys the photographer's subjective perception of the scene. We call this process *digital photomontage*, after the traditional process of combining parts of a variety of photographs to form a composite picture, known as *photomontage*.

The primary technical challenges of photomontage are 1) to choose good seams between parts of the various images so that they can be joined with as few visible artifacts as possible; and 2) to reduce any remaining artifacts through a process that fuses the image regions. To this end, our approach makes use of (and combines) two techniques: *graph-cut optimization* [Boykov et al. 2001], which we use to find the best possible seams along which to cut the various source images; and *gradient-domain fusion* (based on Poisson equations [Pérez et al. 2003; Fattal et al. 2002]), which we use to reduce or remove any visible artifacts that might remain after the image seams are joined.

This paper makes contributions in a number of areas. Within the graph-cut optimization, one must develop appropriate cost functions that will lead to desired optima. We introduce several new cost functions that extend the applicability of graph-cuts to a number of new applications (listed below). We also demonstrate a user interface that is designed to encourage the user to treat a stack of images as a single, three-dimensional entity, and to explore and find the best parts of the stack to combine into a single composite. The interface allows the user to create a composite by painting with high-level goals; rather then requiring the user to manually select specific sources, the system automatically selects and combines images from the stack that best meet these goals.

In this paper, we show how our framework for digital photomontage can be used for a wide variety of applications, including:

Selective composites: creating images that combine all the best elements from a series of photos of a single, changing scene — for example, creating a group portrait in which everyone looks their best (Figure 1), or creating new individual portraits from multiple input portraits (Figures 6 and 10).

Extended depth of field: creating an image with an extended focal range from a series of images focused at different depths, particularly useful for macro photography (Figure 2).

Relighting: interactively lighting a scene, or a portrait, using portions of images taken under different unknown lighting conditions (Figures 4 and 11).

Stroboscopic visualization of movement. automatically creating stroboscopic images from a series of photographs or a video sequence in which a subject is moving (Figure 5).

http://grail.cs.washington.edu/projects/photomontage/

Figure 1 From a set of five source images (of which four are shown on the left), we quickly create a composite family portrait in which everyone is smiling and looking at the camera (right). We simply flip through the stack and coarsely draw strokes using the *designated source* image objective over the people we wish to add to the composite. The user-applied strokes and computed regions are color-coded by the borders of the source images on the left (middle).

Time-lapse mosaics: merging a time-lapse series into a single image in which time varies across the frame, without visible artifacts from movement in the scene (Figure 7).

Panoramic stitching: creating panoramic mosaics from multiple images covering different portions of a scene, without ghosting artifacts due to motion of foreground objects (Figure 8).

Clean-plate production: removing transient objects (such as people) from a scene in order to produce a clear view of the background, known as a "clean plate" (Figures 9 and 12).

1.1 Related work

The history of photomontage is nearly as old as the history of photography itself. Photomontage has been practiced at least since the mid-nineteenth century, when artists like Oscar Rejlander [1857] and Henry Peach Robinson [1869] began combining multiple photographs to express greater detail. Much more recently, artists like Scott Mutter [1992] and Jerry Uelsmann [1992] have used a similar process for a very different purpose: to achieve a surrealistic effect. Whether for realism or surrealism, these artists all face the same challenges of merging multiple images effectively.

For digital images, the earliest and most well-known work in image fusion used Laplacian pyramids and per-pixel heuristics of salience to fuse two images [Ogden et al. 1985; Burt and Kolczynski 1993]. These early results demonstrated the possibilities of obtaining increased dynamic range and depth of field, as well as fused images of objects under varying illumination. However, these earlier approaches had difficulty capturing fine detail. They also did not provide any interactive control over the results. Haeberli [1994] also demonstrated a simplified version of this approach for creating extended depth-of-field images; however, his technique tended to produce noisy results due to the lack of spatial regularization. He also demonstrated simple relighting of objects by adding several images taken under different illuminations; we improve upon this work, allowing a user to apply the various illuminations locally, using a painting interface.

More recently, the texture synthesis community has shown that representing the quality of pixel combinations as a Markov Random Field and formulating the problem as a minimum-cost graph-cut allows the possibility of quickly finding good seams. Graph-cut optimization [Boykov et al. 2001], as the technique is known, has been used for a variety of tasks, including image segmentation, stereo matching, and optical flow. Kwatra *et al.* [2003] introduced the use of graph-cuts for combining images. Although they mostly focused on stochastic textures, they did demonstrate the ability to combine two natural images into one composite by constraining certain pixels to come from one of the two sources. We extend this approach to the fusion of multiple source images using a set of high-level image objectives.

Gradient-domain fusion has also been used, in various forms, to create new images from a variety of sources. Weiss [2001] used this basic approach to create "intrinsic images," and Fattal *et al.* [2002] used such an approach for high-dynamic-range compression. Our approach is most closely related to Poisson image editing, as introduced by Perez *et al.* [2003], in which a region of a single source image is copied into a destination image in the gradient domain. Our work differs, however, in that we copy the gradients from many regions simultaneously, and we have no single "destination image" to provide boundary conditions. Thus, in our case, the Poisson equation must be solved over the entire composite space. We also extend this earlier work by introducing discontinuities in the Poisson equation along high-gradient seams. Finally, in concurrent work, Levin *et al.* [2004] use gradient-domain fusion to stitch panoramic mosaics, and Raskar *et al.* [2004] fuse images in the gradient domain of a scene under varying illumination to create surrealist images and increase information density.

Standard image-editing tools such as Adobe Photoshop can be used for photomontage; however, they require mostly manual selection of boundaries, which is time consuming and burdensome. While interactive segmentation algorithms like "intelligent scissors" [Mortensen and Barrett 1995] do exist, they are not suitable for combining multiple images simultaneously.

Finally, image fusion has also been used, in one form or another, in a variety of specific applications. Salient Stills [Massey and Bender 1996] use image fusion for storytelling. Multiple frames of video are first aligned into one frame of reference and then combined into a composite. In areas where multiple frames overlap, simple per-pixel heuristics such as a median filter are used to choose a source. Image mosaics [Szeliski and Shum 1997] combine multiple, displaced images into a single panorama; here, the primary technical challenge is image alignment. However, once the images have been aligned, moving subjects and exposure variations can make the task of compositing the aligned images together challenging. These are problems that we address specifically in this paper. Akers *et al.* [2003] present a manual painting system for creating photographs of objects under variable illumination. Their work, however, assumes known lighting directions, which makes data acquisition harder. Also, the user must manually paint in the regions, making it difficult to avoid artifacts between different images. Shapetime photography [Freeman and Zhang 2003] produces composites from video sequences that show the closest imaged surface to the camera at each pixel. Finally, in microscopy and macro photography of small specimens such as insects and flowers, scientists

Figure 2 A set of macro photographs of an ant (three of eleven used shown on the left) taken at different focal lengths. We use a global *maximum contrast* image objective to compute the graph-cut composite automatically (top left, with an inset to show detail, and the labeling shown directly below). A small number of remaining artifacts disappear after gradient-domain fusion (top, middle). For comparison we show composites made by Auto-Montage (top, right), by Haeberli's method (bottom, middle), and by Laplacian pyramids (bottom, right). All of these other approaches have artifacts; Haeberli's method creates excessive noise, Auto-Montage fails to attach some hairs to the body, and Laplacian pyramids create halos around some of the hairs.

struggle with a very limited depth of field. To create focused images of three-dimensional specimens, it is common for scientists to combine multiple photographs into a single extended-depth-of-field image. The commercial software package Auto-Montage [Syncroscopy 2003] is the most commonly used system for this task.

Thus, most of the applications we explore in this paper are not new: many have been explored, in one form or another, in previous work. While the results of our framework compare favorably with — and in certain cases actually improve upon — this previous work, we believe that it is the convenience with which our framework can produce comparable or improved output for such a wide variety of applications that makes it so useful. In addition, we introduce a few new applications of image fusion, including selective composites and time-lapse mosaics.

In the next section, we present our digital photomontage framework. Sections 3 and 4 discuss the two main technical aspects of our work: the algorithms we use for graph-cut optimization, and for gradient-domain fusion, respectively. Section 5 presents our results in detail, and Section 6 suggests some areas for future research.

2 The photomontage framework

The digital photomontage process begins with a set of source images, or *image stack*. For best results, the source images should generally be related in some way. For instance, they might all be of the same scene but with different lighting or camera positions. Or they might all be of a changing scene, but with the camera locked down and the camera parameters fixed. When a static camera is desired, the simplest approach is to use a tripod. Alternatively, in many cases the images can be automatically aligned after the fact using one of a variety of previously published methods [Szeliski and Shum 1997; Lucas and Kanade 1981].

Our application interface makes use of two main windows: a *source window*, in which the user can scroll through the source images; and a *composite window*, in which the user can see and interact with the current result. New, intermediate results can also be added to the set of source images at any time, in which case they can also be viewed in the source window and selected by the various automatic algorithms about to be described.

Typically, the user goes through an iterative refinement process to create a composite. Associated with the composite is a *labeling*, which is an array that specifies the source image for each pixel in the composite.

2.1 Objectives

After loading a stack, the user can select an *image objective* that can be applied *globally* to the entire image or *locally* to only a few pixels through a "painting"-style interface. The image objective at each pixel specifies a property that the user would like to see at each pixel in the designated area of the composite (or the entire composite, if it is specified globally). The image objective is computed independently at each pixel position p, based on the set of pixel values drawn from that same position p in each of the source images. We denote this set the *span* at each pixel.

The general image objectives that may be applied in a variety of applications include:

Designated color: a specific desired color to either match or avoid (Figures 4 and 11).

Minimum or ***maximum luminance***: the darkest or lightest pixel in the span (Figures 4 and 11).

Minimum or ***maximum contrast***: the pixel from the span with the lowest or highest local contrast in the span (Figures 2 and 9).

Minimum or ***maximum likelihood***: the least or most common pixel value in the span (subject to a particular histogram quantization function, Figures 5 and 12).

Eraser: the color most different from that of the current composite (Figure 12).

Minimum or ***maximum difference***: the color least or most similar to the color at position p of a specific source image in the stack

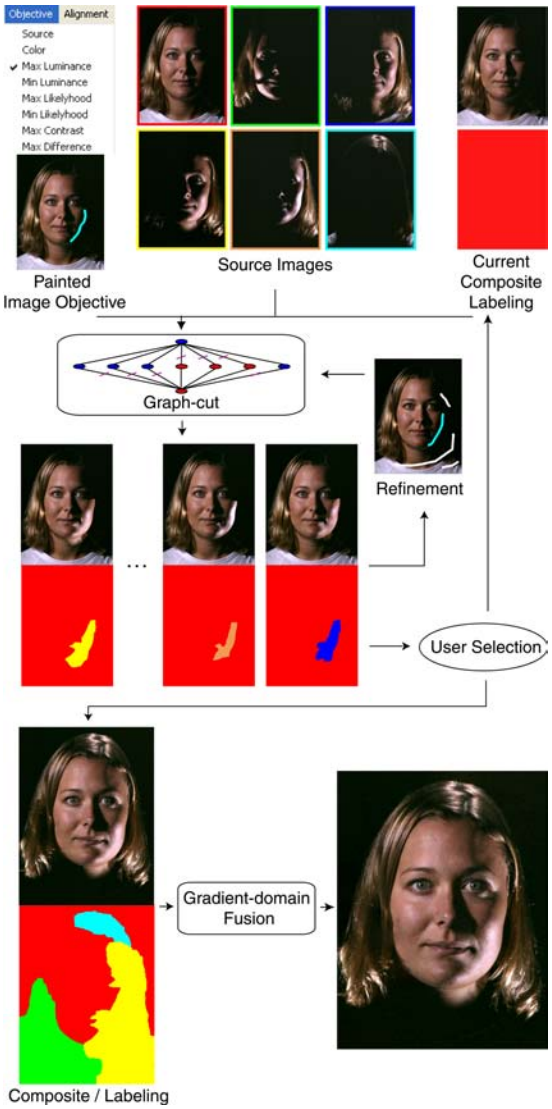

Figure 3 *A flow diagram of a typical interaction with our photomontage framework.* In this example, the user starts with a set of six source images (top) taken under different lighting conditions and attempts to create a final composite with attractive lighting. Initially, the first source image is used as the current composite and labeling (the labeling, shown beneath the composite, is initially constant everywhere since all pixels of the composite come from the same source image). The user then modifies the current composite and labeling by iteratively painting with the single-image brush (described in Section 2.2) with various image objectives. Finally, gradient-domain fusion is applied in order to remove any remaining visible seams.

(Figure 5).

Designated image: a specific source image in the stack (Figures 1, 5, 6, and 10).

For some photomontage applications we design custom image objectives (e.g., Figures 7 and 8). There is also a second type of objective that can be specified by the user, which we call a *seam objective*. The seam objective measures the suitability of a seam between two image regions. Unlike the image objective, it is always specified globally across the entire image. The types of seam objectives we have implemented include:

Colors: match colors across seams (Figures 5, 6, and 10).

Colors & gradients: match colors and color gradients across seams (Figures 2, 7, 8, 9, and 12).

Colors & edges: match colors across seams, but prefer seams that lie along edges (Figures 1, 4, and 11).

The *designated image* objective was originally introduced by Kwatra *et al.* [2003] to interactively merge image regions. Many of the other objectives that we introduce do not require the selection of a specific source, thus allowing the user to specify high-level goals for which the system automatically selects the most appropriate sources. The *colors* and *colors & edges* seam objectives were also introduced by Kwatra *et al.*.

The relative importance of the image vs. seam objectives must be chosen by the user. However, the effect of this importance can be seen very naturally in the result. A relatively low-importance seam objective results in many small regions, and thus many seams. A higher-importance seam objective results in fewer, larger regions. Since different applications call for different amounts of spatial regularization vs. adherence to the image objective, we have found the ability to control this trade-off to be quite useful in practice.

Typically, a seam objective is chosen just once for the entire iterative process, whereas a new image objective is specified at each iterative step. Once an image objective and seam objective are chosen, the system performs a graph-cut optimization that incorporates the relative importance of the image and seam objectives.

2.2 Brushes

If the image objective is specified globally across the space of the composite, then the optimization considers all possible source images and fuses together a composite automatically. To specify an image objective locally, the user can paint with one of two types of brushes. If the *multi-image brush* is used, then the framework proceeds as if a global objective had been specified: all possible source images are fused together in order to best meet the requirements of the locally-specified image objective and the globally-specified seam objective. However, this operation can take significant time to compute.

Thus, more frequently the *single-image brush* is used; in this case graph-cut optimization is performed between the current composite and each of the source images independently. This process is depicted in Figure 3. After painting with the brush, a subset of the source images that best satisfy the locally-specified image objective is presented to the user, along with a shaded indication of the region that would be copied to the composite, in a new window, called the *selection window*. The source images are ordered in this window according to how well they meet the image objective. As the user scrolls through the source images in the selection window, the composite window is continually updated to show the result of copying the indicated region of the current source to the current composite. The user can "accept" the current composite at any time.

Alternatively, the user can further refine the automatically selected seam between the selected source image and the current composite in one of two ways. First, the user can enlarge the portion of the selected source image by painting additional strokes in the results window. In this case, any pixels beneath these new strokes are required to come from the source image. Second, the user can adjust an "inertia" parameter: the higher the inertia setting, the smaller the region of the source image that is automatically selected. Since graph-cuts are a global optimization method, a brush stroke can have surprising, global effects. This parameter allows the user to specify that only pixels close to the brush stroke should be affected by the stroke.

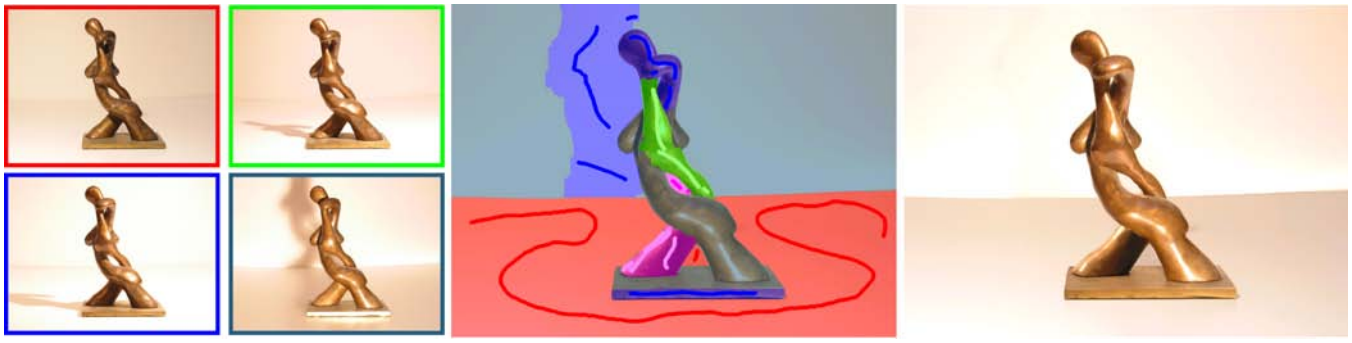

Figure 4 Four of a set of five images (left) of a bronze sculpture under different lighting conditions (taken by waving a desk lamp in front of the sculpture). The user begins with a single source image, and then creates the lighting of the final composite by painting a series of strokes with various image objectives. Strokes using the *maximum* and *minimum luminance* objectives are used to remove and add both highlights and shadows. Strokes using the *designated color* objective are used to select the color of the statue's base, and to create an evenly lit table surface (the colors of the strokes shown indicate the source image chosen by the image objective). The supplemental video shows the process of creating this result.

This painting process is repeated iteratively. At each step, the user can choose a new image objective and apply it either globally or locally, by painting new strokes. Finally, in some cases the resulting composite may have a few, lingering visible artifacts at seams. If so, the user can perform a final gradient-domain fusion step to improve the result.

The next two sections describe the graph-cut optimization and gradient-domain fusion steps in detail.

3 Graph-cut optimization

We use graph-cut optimization to create a composite that satisfies the image and seam objectives specified by the user.

Suppose we have n source images $S_1, \ldots, S_n$. To form a composite, we must choose a source image S_i for each pixel p. We call the mapping between pixels and source images a *labeling* and denote the label for each pixel $L(p)$. We say that a *seam* exists between two neighboring pixels p, q in the composite if $L(p) \neq L(q)$.

Boykov *et al.* [2001] have developed graph-cut techniques to optimize pixel labeling problems; we use their approach (and software). The algorithm uses "alpha expansion" to minimize a cost function. Although a full description of this algorithm is beyond the scope of this paper, it essentially works as follows. The t'th iteration of the inner loop of the algorithm takes a specific label α and a current labeling L_t as input and computes an optimal labeling L_{t+1} such that $L_{t+1}(p) = L_t(p)$ or $L_{t+1}(p) = \alpha$. The outer loop iterates over each possible label. The algorithm terminates when a pass over all labels has occurred that fails to reduce the cost function. If the cost function is a metric, the labeling computed is guaranteed to be within a factor of two of the global minimum.

In our case, we define the *cost function* C of a pixel labeling L as the sum of two terms: a *data penalty* C_d over all pixels p and an *interaction penalty* C_i over all pairs of neighboring pixels p, q:

$$C(L) = \sum_p C_d(p, L(p)) + \sum_{p,q} C_i(p, q, L(p), L(q)) \quad (1)$$

For our application, the data penalty is defined by the distance to the image objective, whereas the interaction penalty is defined by the distance to the seam objective.

Specifically, we define the data penalty $C_d(p, L(p))$ in the following ways as selected by the user:

Designated color (most or least similar): the Euclidean distance in *RGB* space of the source image pixel $S_{L(p)}(p)$ from a user-specified target color. We supply a user interface for the selection of a pixel in the span that is used as the color target.

Minimum (maximum) luminance: the distance in luminance from the minimum (maximum) luminance pixel in a pixels span.

Minimum (maximum) likelihood: the probability (or one minus the probability) of the color at $S_{L(p)}(p)$, given a probability distribution function formed from the color histogram of all pixels in the span (the three color channels are histogrammed separately, using 20 bins, and treated as independent random variables).

Eraser: the Euclidean distance in *RGB* space of the source image pixel $S_{L(p)}(p)$ from the current composite color.

Minimum (maximum) difference: the Euclidean distance in *RGB* space of the source image pixel $S_{L(p)}(p)$ from $S_u(p)$, where S_u is a user-specified source image.

Designated image: 0 if $L(p) = u$, where S_u is a user-specified source image, and a large penalty otherwise.

Contrast: a measure created by subtracting the convolution of two Gaussian blur kernels computed at different scales [Reinhard et al. 2002].

We define the seam objective to be 0 if $L(p) = L(q)$. Otherwise, we define the objective as:

$$C_i(p, q, L(p), L(q)) = \begin{cases} X & \text{if matching "colors"} \\ Y & \text{if matching "gradients"} \\ X+Y & \text{if matching "colors \& gradients"} \\ X/Z & \text{if matching "colors \& edges"} \end{cases}$$

where

$$\begin{aligned} X &= \|S_{L(p)}(p) - S_{L(q)}(p)\| + \|S_{L(p)}(q) - S_{L(q)}(q)\| \\ Y &= \|\nabla S_{L(p)}(p) - \nabla S_{L(q)}(p)\| + \|\nabla S_{L(p)}(q) - \nabla S_{L(q)}(q)\| \\ Z &= E_{L(p)}(p,q) + E_{L(q)}(p,q)) \end{aligned}$$

and $\nabla S_z(p)$ is a 6-component color gradient (in R, G, and B) of image z at pixel p, and $E_z(p,q)$ is the scalar edge potential between two neighboring pixels p and q of image z, computed using a Sobel filter.

Note that all of these seam objectives are metrics, except X/Z which is a semi-metric since it does not always satisfy the triangle inequality. When this seam penalty is used, many of the theoretical guaran-

Figure 5 To capture the progression of time in a single image we generate this stroboscopic image from a video sequence. Several video frames are shown in the first column. We first create a background image using the *maximum likelihood* objective (second column, top) and then add it to the stack. Then, we use the *maximum difference* objective to compute a composite that is maximally different from the background (second column, bottom). A lower weight for the image objective results in fewer visible seams but also fewer instances of the girl (third column, top). Beginning with the first result, the user removes the other girls by brushing in parts of the background and one of the sources using the *designated source* objective (third column, bottom) to create a final result (right).

tees of the "alpha expansion" algorithm are lost. However, in practice we have found it still gives good results. Kwatra *et al.* [2003] also successfully use alpha expansion with this interaction penalty.

Finally, the "inertia" control, described in the previous section, is implemented by calculating an approximate Euclidean distance map [Danielsson 1980] $D(p)$ that describes the distance from the painted area at each point p. (This distance is 0 within the area.) A weighted version of this distance is added to the overall cost function being minimized by the graph-cut optimization whenever $L_{t+1}(p) \neq L_t(p)$. The "inertia" parameter is precisely this weight. Thus, the higher the inertia, the less likely the graph-cut is to select a new label for regions that are far from the painted area.

4 Gradient-domain fusion

For many applications the source images are too dissimilar for a graph-cut alone to result in visually seamless composites. If the graph-cut optimization cannot find ideal seams, artifacts may still exist.

In these cases, it is useful to view the input images as sources of color gradients rather than sources of color. Using the same graph-cut labeling, we copy color gradients to form a composite vector field. We then calculate a color composite whose gradients best match this vector field. Doing so allows us to smooth out color differences between juxtaposed image regions. We call this process *gradient-domain fusion*.

The composite color image is computed separately in the three color channels. The details of this task have been well described by other researchers [Fattal et al. 2002; Pérez et al. 2003]. As they noted, unless the gradient field is conservative, no image exists whose gradient exactly matches the input. Instead, a best-fit image in a least-squares sense can be calculated by solving a discretization of the Poisson equation.

For a single color channel, we seek to solve for the pixel values $I(x,y)$. We re-order these values into a vector v, but, for convenience here, we still refer to each element $v_{x,y}$ based on its corresponding (x,y) pixel coordinates. An input gradient $\nabla I(x,y)$ specifies two linear equations, each involving two variables:

$$v_{x+1,y} - v_{x,y} = \nabla I_x(x,y) \tag{2}$$

$$v_{x,y+1} - v_{x,y} = \nabla I_y(x,y) \tag{3}$$

Like Fattal *et al.* [2002], we employ *Neumann boundary conditions*, equivalent to dropping any equations involving pixels outside the boundaries of the image. In addition, because the gradient equations only define v up to an additive constant, we ask the user to choose a pixel whose color will be constrained to the color in its source image and then add this constraint to the linear system.

The resulting system of equations is over-constrained. We solve for the least-squares optimal vector v using conjugate gradients applied to the associated normal equations [Meyer 2000]. This algorithm can be slow; however, we generally compute this image only once as a final stage of the process. As discussed by Fattal *et al.* [2002] and others, a solution can be computed very quickly using multigrid methods, at the cost of a more complex implementation.

One additional complication can arise when using gradient-domain fusion with a seam cost that cuts along high-gradient edges. Blending across these seams may result in objectionable blurring artifacts, since strong edges may be blended away. We solve this problem by simply dropping the linear constraints wherever the edge strength, as measured by a Sobel operator, exceeds a certain threshold.

5 Results

We now demonstrate how our system can be applied to a variety of tasks. Depending on the specific task, a different set of image and seam objectives will be used to achieve the goal. Some results (Figures 2, 7, 8,and 9) do not require painting by the user; they are computed automatically by applying an image objective globally over the whole image. Other results are created by user-painted image objectives (Figures 1, 4, 6, 10, and 11). Finally, the user may choose to begin with a globally-computed composite, and then interactively modify it (Figures 5 and 12).

Selective composites. Photomontage allows us to interactively select and assemble the best fragments of a set of images. Photographs of a group of people are a good example; it is difficult to capture a group portrait without a few closed eyes or awkward expressions. In Figure 1, we merge several group portraits into one that is better than any of the originals. Even portraiture of a single

Figure 6 We use a set of portraits (first row) to mix and match facial features, to either improve a portrait, or create entirely new people. The faces are first hand-aligned, for example, to place all the noses in the same location. In the first two images in the second row, we replace the closed eyes of a portrait with the open eyes of another. The user paints strokes with the *designated source* objective to specify desired features. Next, we create a fictional person by combining three source portraits. Gradient-domain fusion is used to smooth out skin tone differences. Finally, we show two additional mixed portraits.

person can be challenging and can be aided by assembling a composite (Figure 6). We can also push the photomontage paradigm to create entirely fictional but surprisingly realistic (and sometimes funny) composite portraits of different people (Figures 6 and 10).

Image-based relighting. Photography is the art of painting with light; our next application of photomontage demonstrates this point. Studio portraiture (Figure 11) and product photography (Figure 4) typically involve a complex lighting setup. Instead, we allow a user to literally paint light, and simulate complex lighting with just a few brush strokes. After capturing a set of images taken under various lighting conditions, we quickly relight a subject, both with realistic and non-physically realizable lighting. This allows photographers to experiment with a wide range of lighting possibilities after a shoot, instead of carefully planning a desired effect in advance. Note that several of the image objectives can be thought of as highlight or shadow brushes that indicate the desired placement of these lighting effects in a scene. The user can paint with the *color* image objective using a bright, highlight color or a dark, shadow color to find highlights or shadows in a specific location in the set of source images. Alternatively, the *maximum* or *minimum luminance* objectives can also be used to find highlights or shadows, respectively, and add them to the composite.

Extended depth-of-field. In microscopy and macro photography of small specimens such as insects and flowers, scientists struggle with a very limited depth of field. To create focused images of three-dimensional specimens, it is common for scientists to combine multiple photographs into a single, extended depth-of-field image. The commercial software package Auto-Montage [Syncroscopy 2003] is the most commonly used system for this task. This problem has also been addressed using Laplacian pyramids [Ogden et al. 1985; Burt and Kolczynski 1993]. Finally, Haeberli [1994] demonstrates a simplification of this approach that uses per-pixel heuristics of contrast to select a source. We similarly use a measure of contrast to select sources, but do so with the spatial consistency afforded by graph-cut optimization. This application requires no user input since the objectives are global over the full image. To demonstrate our system we obtained a stack of images of an ant from an entomologist. Figure 2 shows a comparison of the results using previous algorithms to our approach. Our result has fewer seam artifacts and more regions in focus.

Image mosaics. Image mosaics [Szeliski and Shum 1997] combine multiple, displaced images into a single panorama; here, the primary technical challenge is image alignment. However, once the images have been aligned, moving objects and exposure variations can make the task of compositing the aligned images together challenging. If people in a scene move between images, simple linear blending of source images results in ghosts (Figure 8). An ideal composite will integrate these different moments of time into one, natural still. Davis [1998] addresses this problem by finding optimal seams with Dijkstra's algorithm; however it cannot handle many overlapping images. Uyttendaele *et al.* [2001] use a vertex cover algorithm and exposure compensation to compute mosaics. We have tested our approach on the same data set used in their paper in Figure 8, and our results compare favorably.

Background reconstruction. In many cases it can be difficult to capture an image of a scene free of unwanted obstructions, such as passing people or power lines. We demonstrate the application of our system to reconstructing a background image free of obstruction. Figure 12 shows how we can remove people from a crowded town square in front of a cathedral by merging several images of the scene taken on a tripod. We also show that our results improve upon common alternate approaches. In Figure 9, we use multiple photographs taken from offset positions of a mountain scene that is obstructed with power lines, to create an image free of wires.

Visualizing motion. Artists have long used a variety of techniques to visualize an interval of time in a single image. Photographers like Jules-Etienne Marey [Braun 1992] and Eadward Muybridge [1955] have created compelling stroboscopic visualizations of moving humans and other types of motion. Traditional stroboscopy depicts multiple instances, or sprites, of a moving subject at regular intervals of time. Using graph-cut optimization and a video sequence as input, we are able to produce a similar effect (Figure 5). The optimization goes beyond what is possible with regular intervals by choosing sprites that appear to flow seamlessly into each other. This effect would be very difficult to create manually. Simply averaging the selected images would result in a series of ghosts superimposed on the background.

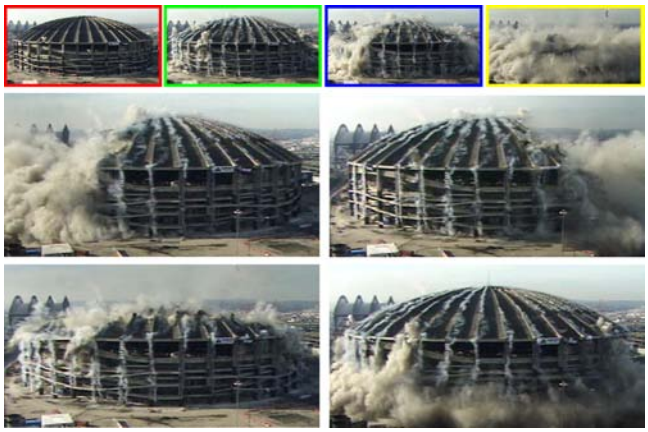

Figure 7 A few frames of a video sequence depicting the collapse of a building (top row). Using a custom image objective that encourages linear time variation across the space of the composite, we depict time flowing right to left, and then left to right (middle row). Time can also flow bottom to top, or top to bottom (bottom row).

Time-lapse mosaics. Time-lapse photography allows us to witness events that occur over too long an interval for a human to perceive. Creating a time-lapse image can be challenging, as there is typically large-scale variation in lighting and transient scene elements in a time-lapse sequence. Our photomontage framework is well suited for this task as shown in Figure 7.

6 Conclusions and future work

We have presented a framework that allows a user to easily and quickly create a digital photomontage. We have demonstrated a system that combines graph-cut optimization and a gradient domain image-fusion algorithm with an intuitive user interface for defining local and global objectives. Our work on digital photomontage suggests a number of areas for future work.

Our system has been shown to be applicable to a wide range of photomontage applications; however, we strongly suspect that there are many more. In the process of conducting this research we have discovered that a surprising array of beautiful, useful, and unexpected images can arise from an image stack. An exciting opportunity for future work is to discover more applications of photomontage. This would involve finding new types of image stacks than the ones presented here, and possibly new image objectives that would be used to find the best parts of a stack to retain in a composite.

Several of the applications we present require user guidance in selecting the best parts of images (e.g., image-based relighting and selective composites); more sophisticated image objectives could be defined that automatically find the best parts of images. For example, the group portrait in Figure 1 could created by automatically finding the best portrait of each person.

Finally, it may be possible to apply our approach to other types of data, such as 2.5D layered images, video sequences, 3D volumes, or even polygonal models. Such data sets would probably require new image and seam objectives that consider the relationships between the additional dimensions of the data, and an interface for controlling optimization in these higher dimensional spaces.

Acknowledgments: The ant images (Figure 2) are courtesy John Longino, Evergreen College. We also thank Paul Debevec for the Light Stage data, Rick Szeliski for the panoramic images, and Vladimir Kolmogorov for the graph-cut optimization software.

References

AKERS, D., LOSASSO, F., KLINGNER, J., AGRAWALA, M., RICK, J., AND HANRAHAN, P. 2003. Conveying shape and features with image-based relighting. In *IEEE Visualization*, 349–354.

BOYKOV, Y., VEKSLER, O., AND ZABIH, R. 2001. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence 23*, 11, 1222–1239.

BRAUN, M. 1992. *Picturing Time: The Work of Etienne-Jules Marey*. The University of Chicago Press.

BURT, P., AND KOLCZYNSKI, R. 1993. Enhanced image capture through fusion. In *International Conference on Computer Vision (ICCV 93)*, 173–182.

DANIELSSON, P.-E. 1980. Euclidean distance mapping. *Computer Graphics and Image Processing 14*, 227–248.

DAVIS, J. 1998. Mosaics of scenes with moving objects. In *Computer Vision and Pattern Recognition (CVPR 98)*, 354–360.

FATTAL, R., LISCHINSKI, D., AND WERMAN, M. 2002. Gradient domain high dynamic range compression. *ACM Transactions on Graphics 21*, 3, 249–256.

FREEMAN, W. T., AND ZHANG, H. 2003. Shape-time photography. In *Conference on Computer Vision and Pattern Recognition (CVPR 03)*, 151–157.

HAEBERLI, P. 1994. *Grafica Obscura web site*. http://www.sgi.com/grafica/.

KWATRA, V., SCHÖDL, A., ESSA, I., TURK, G., AND BOBICK, A. 2003. Graphcut textures: Image and video synthesis using graph cuts. *ACM Transactions on Graphics 22*, 3, 277–286.

LEVIN, A., ZOMET, A., PELEG, S., AND WEISS, Y. 2004. Seamless image stitching in the gradient domain. In *European Conference on Computer Vision (ECCV 04)*, (to appear).

LUCAS, B. D., AND KANADE, T. 1981. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence (IJCAI '81)*, 674–679.

MASSEY, M., AND BENDER, W. 1996. Salient stills: Process and practice. *IBM Systems Journal 35*, 3&4, 557–574.

MEYER, C. 2000. *Matrix Analysis and Applied Linear Algebra*. Society for Industrial and Applied Mathematics.

MORTENSEN, E. N., AND BARRETT, W. A. 1995. Intelligent scissors for image composition. In *Proceedings of SIGGRAPH 95*, Computer Graphics Proceedings, Annual Conference Series, 191–198.

MUTTER, S., AND KRAUSE, M. 1992. *Surrational Images: Photomontages*. University of Illinois Press.

MUYBRIDGE, E. 1955. *The human figure in motion*. Dover Publications, Inc.

OGDEN, J. M., ADELSON, E. H., BERGEN, J., AND BURT, P. 1985. pyramid-based computer graphics. *RCA Engineer 30*, 5, 4–15.

PÉREZ, P., GANGNET, M., AND BLAKE, A. 2003. Poisson image editing. *ACM Transactions on Graphics 22*, 3, 313–318.

RASKAR, R., ILIE, A., AND YU, J. 2004. Image fusion for context enhancement and video surrealism. In *NPAR 2004: Third International Symposium on Non-Photorealistic Rendering*, (to appear).

REINHARD, E., STARK, M., SHIRLEY, P., AND FERWERDA, J. 2002. Photographic tone reproduction for digital images. *ACM Transactions on Graphics 21*, 3, 267–276.

REJLANDER, O., 1857. Two ways of life. Photograph.

ROBINSON, H. P. 1869. *Pictorial Effect in Photography: Being Hints on Composition and Chiaroscuro for Photographers*. Piper & Carter.

SYNCROSCOPY, 2003. Auto-montage.

SZELISKI, R., AND SHUM, H.-Y. 1997. Creating full view panoramic mosaics and environment maps. In *Proceedings of SIGGRAPH 97*, Computer Graphics Proceedings, Annual Conference Series, 251–258.

UELSMANN, J., AND COLEMAN, A. D. 1992. *Jerry Uelsmann: Photo Synthesis*. University Press of Florida.

UYTTENDAELE, M., EDEN, A., AND SZELISKI, R. 2001. Eliminating ghosting and exposure artifacts in image mosaics. In *Conference on Computer Vision and Pattern Recognition (CVPR 01)*, 509–516.

WEISS, Y. 2001. Deriving intrinsic images from image sequences. In *International Conference On Computer Vision (ICCV 01)*, 68–75.

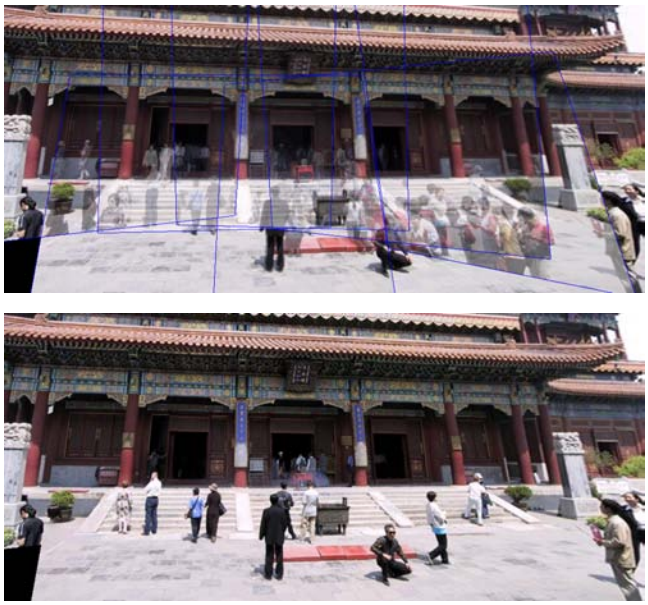

Figure 8 Simple linear blending of panoramic source images (top, with each registered source outlined in blue) can result in ghosting artifacts if the scene contains movement; our framework can eliminate this ghosting (bottom). Each aligned source image is extended to cover the full panorama; pixels not originally present in the source image are marked as invalid. A custom image objective is designed that returns a large penalty for pixel labels that point to invalid source pixels, and zero otherwise. Finally, gradient-domain fusion is very effective at compensating for differences in brightness and hue caused by exposure variation between stack images.

Figure 9 Three of a series of nine images of a scene that were captured by moving the camera to the left, right, up, and down in increments of a few feet. The images were registered manually to align the background mountains. A *minimum contrast* image objective was then used globally to remove the wires.

Figure 10 Fictional researchers created by combining portraits of graphics pioneers Andries Van Dam and Jim Kajiya, using the same approach used in Figure 6.

Figure 11 We apply our relighting tools to an image stack (first row) acquired with the aid of a spherical gantry or a light stage. With just a few strokes of the *luminance* objectives we simulate complex studio lighting. To create a glamorous Hollywood portrait (middle row, left) we simulate a key light to the left of the camera, two fill lights for the left cheek and right side of the hair, and an overhead light to highlight the hair (bottom row, left). The graph-cut introduces a slight artifact in the hair; gradient domain fusion removes the sharp edges (middle row, middle). Note this result is quite distinct from simply averaging the source images (bottom row, middle). We create a completely different and frightening look (middle row, right) by adding a key light from below and a fill light from above. Using the multi-image brush we quickly produce a strong specular halo around her head (bottom row, right).

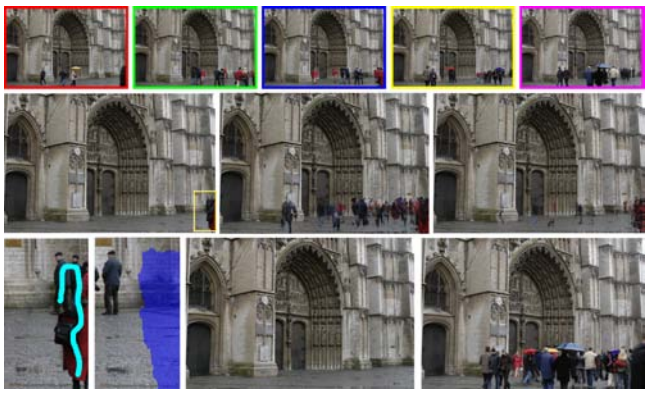

Figure 12 From a set of five images (top row) we create a relatively clean background plate using the *maximum likelihood* objective (middle row, left). The next two images to the right show that our result compares favorably to a per-pixel median filter, and a per-pixel maximum likelihood objective, respectively. An inset of our result (bottom row, left) shows several remaining people. The user paints over them with the *eraser* objective, and the system offers to replace them with a region, highlighted in blue, of the fourth input image. The user accepts this edit, and then applies gradient-domain fusion to create a final result (bottom row, middle). Finally, using a *minimum likelihood* image objective allows us to quickly create a large crowd (bottom right).