Edge-Based Image Coarsening

Raanan Fattal
Hebrew University of Jerusalem, Israel
Robert Carroll
University of California, Berkeley
Maneesh Agrawala
University of California, Berkeley

This paper presents a new dimensionally-reduced linear image space that allows a number of recent image manipulation techniques to be performed efficiently and robustly. The basis vectors spanning this space are constructed from a scale-adaptive image decomposition, based on kernels of the bilateral filter. Each of these vectors locally binds together pixels in smooth regions and leaves pixels across edges independent. Despite the drastic reduction in the number of degrees of freedom, this representation can be used to perform a number of recent gradient-based tonemapping techniques. In addition to reducing computation time, this space can prevent the bleeding artifacts which are common to Poisson-based integration methods. In addition, we show that this reduced representation is useful for energy-minimization methods in achieving efficient processing and providing better matrix conditioning at a minimal quality sacrifice.

Categories and Subject Descriptors: I.3.3 [**Computer Graphics**]: Picture/Image Generation, Display algorithms

General Terms: Algorithms

Additional Key Words and Phrases: image representation, bilateral filtering, gradient domain techniques

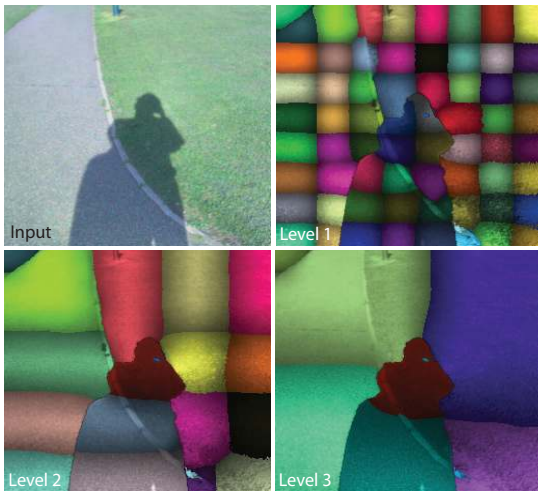

Fig. 1. An input image with three levels of our scale-adaptive coarsened representation. This representation breaks the image into overlapping kernels that bind together smooth regions but are also shaped by strong edges. This visualization shows the dominant kernel for each pixel.

1. INTRODUCTION

Edges and gradients carry most of the visually important information in images. A wide variety of recent image editing techniques are based on this insight and provide powerful tools for manipulating images based on their edges and gradients. These techniques fall into two main classes; *bilateral filter based methods* and *gradient based manipulation techniques.*

The bilateral filter [Tomasi and Manduchi 1998] is a nonlinear filter that locally gathers information from pixels that are similar to one another (i.e. in intensity, color, etc.), while excluding pixels that lie across nearby edges. Recently the bilateral filter has become a practical building block [Durand and Dorsey 2002], replacing earlier PDE-based methods that decomposed images into a piecewise-smooth base layer and one or more residual detail layers [Tumblin and Turk 1999]. Since edges are preserved in the base layer, techniques based on the bilateral image decomposition avoid the haloing artifacts common to linear filter image decomposition techniques [Burt and Adelson 1987]. Moreover, the bilateral filter is robust and efficient to compute [Chen et al. 2007]. As a result, bilateral image decompositions have proven useful for many tasks including dynamic range compression [Durand and Dorsey 2002], flash/no-flash enhancement [Eisemann and Durand 2004; Petschnigg et al. 2004], tone management [Bae et al. 2006], nonphotorealistic relighting [Fattal et al. 2007] and operator upsampling [Kopf et al. 2007].

The gradient domain provides a natural setting for image manipulation techniques, including dynamic range compression [Fattal et al. 2002], seamless image stitching [Levin et al. 2004], image editing [Pérez et al. 2003], alpha matte extraction [Sun et al. 2004], and shadow removal [Finlayson et al. 2006; Xu et al. 2006]. Solving the Poisson equation amounts to performing an L_2 minimization in which the image gradients are weighted uniformly in space. More recent gradient based methods such as colorization [Levin et al. 2004], interactive tone mapping [Lischinski et al. 2006] and alpha matting [Levin et al. 2006], propagate local image editing operations throughout the image according to the underlying gradient field. These approaches require solving a similar optimization problem, but in this case the output image gradients are weighted in a spatially-dependent manner.

In this paper we develop a new dimensionally-reduced linear image space that is based on the bilateral filter and designed to support many of these recent image manipulation techniques. The key idea of bilateral filter based methods is that similar pixels should be treated similarly and independently of pixels separated by a discontinuity. We apply this idea to construct a coarse image representation consisting of elementary basis functions that are derived from the bilateral filter kernels, shown in Figure 1. Together these functions act as a spanning basis for the output image.

Our coarse representation forces pixels that are similar in the input to change collectively and smoothly, but permits strong modifications along edges. Although this reduction limits the possibilities for manipulating images, we show that it is well-suited for many applications. By defining two projection operators onto this coarse representation, one based on pixel intensities and the other based on pixel differences, our representation naturally expresses gradient domain techniques and with a simple change of variables it can express gradient-based energy functionals.

We show that projection into this space significantly reduces the bleeding artifacts common to gradient-based operations formulated in the original pixel-based image space. Also, our coarsened representation allows us to closely approximate the results obtained via pixel-based optimization formulations while achieving efficient processing and better matrix conditioning. However, the basis functions we construct are tailored to the edges of the input image and therefore operations that alter the geometry of the edges, such as seamless pasting, cannot be performed in this representation.

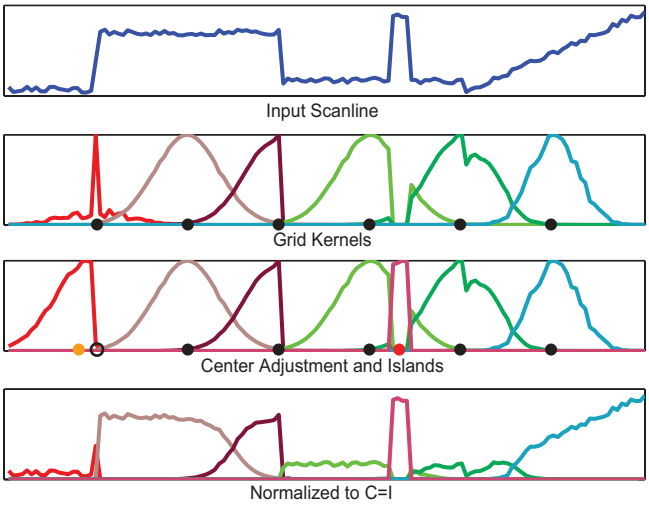

Fig. 2. 1D kernel construction steps. (top) Input image scanline. (second row) Kernels computed on regular grid with centers shown as block dots. (third row) Kernel centers are shifted away from edges (open dot moves to orange dot) and island kernels are added (red dot). (bottom row) Kernels normalized to input image.

2. IMAGE COARSENING

Our coarsened edge-based image representation is *data dependent*. Given an input image we construct a basis derived from bilateral filter kernels. This basis spans a linear subspace of images in which similar pixels (i.e. pixels in smooth image regions) are bound together. More formally, given an input image I we construct a set of kernels whose support is composed of pixels with similar intensity (or color) as measured by the bilateral [Tomasi and Manduchi 1998] pixel-pair affinity function

$$S(\mathbf{x}, \mathbf{y}) = g_s(\|\mathbf{x} - \mathbf{y}\|) \cdot g_r(|I(\mathbf{x}) - I(\mathbf{y})|),$$

where $g_s = \exp(-x^2/\sigma_s^2)$ is a spatial weighting function with σ_s defining a spatial scale, and $g_r = \exp(-x^2/\sigma_r^2)$ is an edge-stopping function with σ_r defining a scale in the intensity range. Other functions can be used for spatial weighting and edge stopping as discussed by Durand and Dorsey [2002].

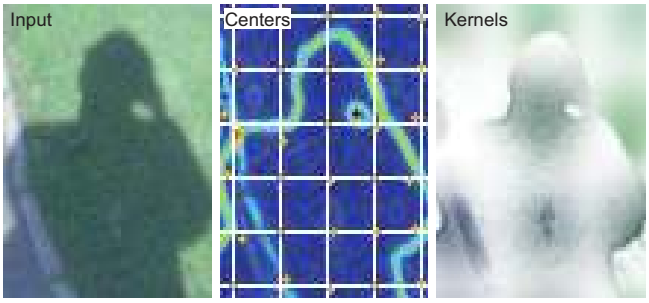

Fig. 3. Image kernels. (left) Input image, (middle) kernel centers are marked in orange and island centers in red. (right) Visualization of shape of non-overlapping kernels. Note that kernels do overlap in our representation but to clarify their shape we are only visualizing a subset of them.

We use this affinity function to define the following set of normalized kernels

$$K_i(\mathbf{x}) = \frac{C(\mathbf{x})S(\mathbf{x}_i, \mathbf{x})}{\sum_{j=1}^{n} S(\mathbf{x}_j, \mathbf{x})}, \tag{1}$$

where $C(\mathbf{x})$ is a per-pixel normalization factor which we describe shortly, and $\mathbf{x}_i$ for $i = 1...n$ are the kernel centers.

To compute the kernel centers we first uniformly downsample the input pixels by a factor k (in our examples k ranges from 4 to 8 pixels.) Since we are interested in binding together pixels in homogeneous regions, we locally shift each kernel center $\mathbf{x}_i$ to a pixel where the input image has a minimal gradient norm $\|\nabla I(\mathbf{x})\|$ within a small window of $k-2$ by $k-2$ pixels around the original uniformly downsampled center. This local shift places kernel centers in smooth areas and avoids edge pixels. By construction these kernels overlap one another in smooth regions but are disjoint across edges (see figures 2 and 3).

The local shift of the kernel centers can cause some pixels to be far away from any kernel center. In addition, some pixels may be very different in intensity from any nearby kernel center. Such pixels may not be well represented by our construction. Therefore, we add *island kernels* to ensure adequate coverage of such pixels. We build these additional kernels by iteratively scanning the image for pixels $\mathbf{x}$ where $\sum_i S(\mathbf{x}_i, \mathbf{x}) < \tau$ and use this location as the new kernel's center. In our examples, we set the island creation threshold, τ, to be 0.3 unless stated otherwise.

We use these kernels to define a set of overlapping basis functions that are smooth where the input image is smooth and discontinuous along edges. More specifically the kernels modulate a set of *construction polynomials* (CPs) given by

$$\begin{aligned} P_i(\mathbf{x}) = a_{i0} + a_{i1}(x - x_i) + a_{i2}(y - y_i) + a_{i3}(x - x_i)(y - y_i) \\ + a_{i4}(x - x_i)^2 + a_{i5}(y - y_i)^2 + ... \end{aligned}$$

The degree of the polynomial (usually 0, 1 or 2) is chosen depending on the application. We call the linear image subspace $\mathcal{J}$ spanned by these polynomials the *Bilateral Image Coarsening* (BIC) space and it is given by

$$\mathcal{J} = \Big\{ \sum_{i=1}^{n} K_i(\mathbf{x}) P_i(\mathbf{x}) \ : \ a_{ij} \in \mathbb{R} \Big\}.$$

The coefficients of these polynomials a_{ij} are the degrees of freedom that parameterize this space. For example, for first-order CPs every image $J \in \mathcal{J}$ is given by

$$J(\mathbf{x}) = \sum_{i=1}^{n} K_i(\mathbf{x})\big(a_{i0} + (x - x_i)a_{i1} + (y - y_i)a_{i2}\big), \tag{2}$$

for some choice of a_{ij} where $j = 0..2$.

The normalization function $C(\mathbf{x})$ in equation (1) determines the sum of the kernels at each pixel, i.e., $\sum_i K_i(\mathbf{x}) = C(\mathbf{x})$, and is used to shape $\mathcal{J}$. By setting $a_{i0} = 1$ and $a_{ij} = 0$ for $j > 0$ we see that $\mathcal{J}$ must contain $C(\mathbf{x})$. Intuitively, the normalization function serves as an "origin" point for $\mathcal{J}$. We show that there are two useful options for setting $C(\mathbf{x})$ depending on the application. One choice is to set $C(\mathbf{x}) = I(\mathbf{x})$ so that $\mathcal{J}$ is a subspace "surrounding" the input image. Another choice is to set $C(\mathbf{x}) = 1$ so that $\mathcal{J}$ becomes a space of piecewise-smooth functions which is useful for representing the parameters of some image editing operations such as colorization.

2.1 Scale-Adaptive Coarsening

Thus far we have only considered binding pixels at a single scale, giving us kernels of roughly the same size. In this setting large homogeneous regions are covered by many small kernels (see level 1 in Figure 1). Alternatively, we define a scale-adaptive coarsening which gives us kernels that vary in size depending on image content.

To build these scale-adaptive kernels we first compute kernels at a single scale, which are a combination of kernels centered on a regular grid and island kernels. The island kernels are added to the list of scale-adaptive island kernels and the grid kernels K^l at level l are treated analogously to pixels in the single scale construction and define an affinity function between kernels as

$$S(K_i, K_j) = g_s(\|\bar{\mathbf{x}}_i - \bar{\mathbf{x}}_j\|) \cdot g_r(|\bar{K}_i - \bar{K}_j|),$$

where

$$\bar{\mathbf{x}}_i = \sum_{\mathbf{x}} \mathbf{x} K_i(\mathbf{x}) / \sum_{\mathbf{x}} K_i(\mathbf{x}),$$

is the kernel centroid, and

$$\bar{K}_i = \sum_{\mathbf{x}} I(\mathbf{x}) K_i(\mathbf{x}) / \sum_{\mathbf{x}} K_i(\mathbf{x}),$$

is the kernel's average intensity (or color). We then define scale-adaptive kernels as

$$K_i^{l+1} = \sum_{j=1}^{m} K_j^l \frac{S(K_i^l, K_j^l)}{\sum_{k=1}^{n} S(K_j^l, K_k^l)}, \tag{3}$$

where $K_{i'}^{l+1}$ is at the same spatial location as K_i^l. Given m grid kernels at level l we coarsen the grid sampling by a factor k and obtain m/k^2 grid kernels at level $l+1$. We typically use $4 \le k \le 8$ at the first level and set $k = 2$ for $l > 1$. At each level we increase σ_s by the grid coarsening factor (we use $\sigma_s = 2^l k$ in our testing) and

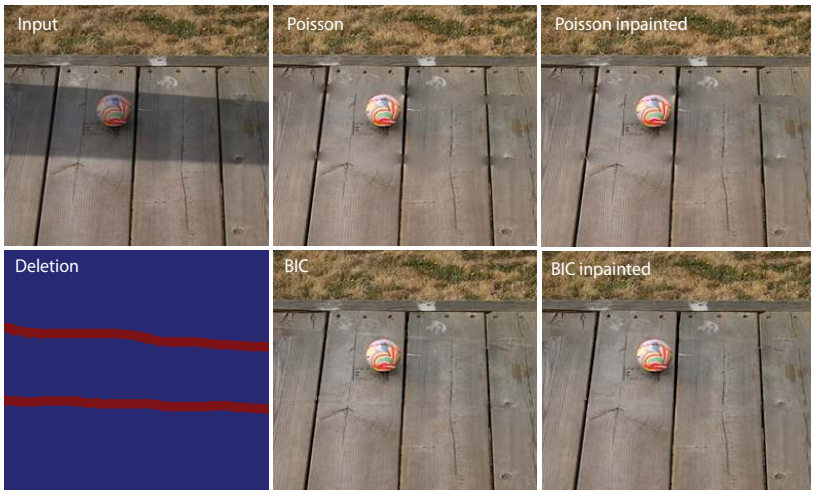

Fig. 4. Shadow removal. (top-left) Shows the input and below, and below it we see the edges deleted in red. (top-center) Shows this edge-deleted gradient field after it is integrated using Poisson and below it we see its projection onto the BIC space. (right) Shows the corresponding results of performing an additional step of inpainting.

optionally specify a different σ_r for the higher levels $l > 1$. As in the first level, we also construct island kernels at each of the subsequent levels and add them to the list of scale-adaptive kernels. Thus, the final result is a set of kernels consisting of those from the coarsest level and the island kernels found at each level. Altogether, by its construction, this set of kernels covers all of the pixels in the image.

In fact, the list of island kernels, denoted by Ω, can also be adaptively coarsened in order to avoid the creation of small and similar island kernels. However, unlike the grid kernels, these kernels do not lie on a rectangular grid and therefore we cannot use a predetermined subsampling pattern. Instead, we construct a coarser subset Ω^C as following: once the kernels at level l are computed, we run through the current list of island kernels, Ω, and if we encounter a kernel K that is poorly covered by Ω^C, i.e., as before $\sum_j S(K_j, K) < \tau$ where $j \in \Omega^C$, then we add it to Ω^C. Thus, each island kernel that is missing from Ω^C is close enough to one (or more) of the kernels in Ω^C. Finally, we use this subset of island kernels to define coarser scale-adaptive island kernels K_i^C for $i \in \Omega^C$ by

$$K_i^C = \sum_{j \in \Omega} K_j \frac{S(K_i, K_j)}{\sum_{k \in \Omega} S(K_j, K_k)}.$$

3. PROJECTION OPERATIONS

We propose the following paradigm for manipulating images in our coarsened representation. Given an input image I we build a corresponding BIC space representation $\mathcal{J}$. Rather than computing or manipulating a_{ij} directly, we apply the desired operation on the pixels or gradients of the input image to build an intermediate image Q. We then project this image or its gradient field onto $\mathcal{J}$ to obtain

the closest image *within* $\mathcal{J}$. We will show that we can use this new representation to implement existing gradient-based image manipulation methods with minimal adaptation.

3.1 Image Projection

In order to project the image Q into $\mathcal{J}$ we search for the image within $\mathcal{J}$ that best matches Q as measured by the L_2 norm. Thus, we compute the coefficients a_{ij} by solving

$$\min_{a_{ij}} \| \sum_i K_i(\mathbf{x}) P_i(\mathbf{x}) - Q \|^2.$$

The quadratic objective function is optimized by solving a system of linear equations. This mapping from Q to $J = \sum_i K_i(\mathbf{x}) P_i(\mathbf{x})$ defines an orthogonal projection $\mathbb{R}^N \rightarrow \mathcal{J}$ (see [Strang 2003]).

For example, in the case of a zero-order basis, $P_i(\mathbf{x}) = a_{i0}$, this minimization amounts to solving $A\mathbf{a} = \mathbf{r}$, where the matrix A is given by $A_{ij} = \langle K_i, K_j \rangle$, the vector $\mathbf{a}$ is composed of a_{i0}, and the vector $\mathbf{r}$ is given by $\mathbf{r}_i = \langle K_i, Q \rangle$. The dot product $\langle \cdot, \cdot \rangle$ operates on images and is defined by

$$\langle I_1, I_2 \rangle = \sum_{\mathbf{x}} I_1(\mathbf{x}) I_2(\mathbf{x}).$$

Using higher order CPs is equivalent to adding kernels modulated by higher order monomials, $x^p y^q K_i(\mathbf{x})$. We do not store these kernels but compute them on the fly when constructing the linear system. The dimension of this system equals the number of CP coefficients times the number of kernels n.

Example Application: Shadow Removal. Finlayson et al. [2006] and Xu et al. [2006] remove shadows from an image by deleting gradients from the log intensity image along the shadow edges and then integrating this modified gradient field using a Poisson equation. In Figure 4 we see that the image resulting from the Poisson integration is blurry along the deleted strips of shadow edge, and there are spurious dark regions at the intersections of the shadow edges and the gaps in the deck. These artifacts are known as *bleeding artifacts* [Pérez et al. 2003].

We can reduce these artifacts by taking an additional step of projecting the resulting image onto $\mathcal{J}$, which is constructed based on the original image I using zero-order construction polynomials and the normalization factor C set to I. As shown in Figure 4, we recover most of the details along the deleted strips and eliminate most of the bleeding artifacts. This successful restoration results from the fact that $\mathcal{J}$ consists only of images that are locally brightened and darkened versions of the original image, and hence $\mathcal{J}$ is unable to produce blurriness or bleeding effects. As we are primarily concerned with the ability to reconstruct an image given a modified field, we deleted the shadow edges manually in this example. Figure 4 also shows that we can go further and use texture-synthesis based inpainting [Efros and Leung 1999] after the shadow removal to reduce remaining artifacts. However, as shown in the example, inpainting alone does not fully eliminate the artifacts when using a Poisson-based integration.

3.2 Gradient Projection

Many recent image manipulation methods operate on image gradients rather than pixels [Finlayson et al. 2002; Fattal et al. 2002; Pérez et al. 2003; Levin et al. 2004; Sun et al. 2004; Xu et al. 2006]. These methods manipulate the gradient field of an input image and try to solve for the corresponding image, but in general this manipulated gradient field does not correspond to an image. Like the inconsistency illustrated in Escher's Waterfall, the manipulated gradient field does not integrate to zero along closed loops and hence does not correspond to a potential function – an image. Instead, these methods search for an image whose gradients are closest to the manipulated gradient field under the L_2 norm by solving a Poisson equation. As we saw in the shadow removal example in the previous section, this minimization under the L_2 norm tends to spread errors, resulting as bleeding artifacts.

The BIC image space is designed to avoid such artifacts because similar pixels are bound together so that flat regions remain flat and edge contours remain unaltered. Thus, we formulate the gradient integration procedure over the BIC space as

$$\min_{a_{ij}} \|\nabla \sum_i K_i(\mathbf{x}) P_i(\mathbf{x}) - \big(Q^x(\mathbf{x}), Q^y(\mathbf{x})\big)\|^2,$$

where $\nabla = (D_x, D_y)$, D_x and D_y denote the forward (or backward) pixel difference operators along the two axes, and (Q^x, Q^y) denotes the manipulated gradient field. In the case of zero-order CPs we minimize this equation by solving $L\mathbf{a} = \mathbf{r}$, where the Laplacian matrix L is given by

$$L_{ij} = \langle K_i^x, K_j^x \rangle + \langle K_i^y, K_j^y \rangle,$$

with

$$K_i^x = D_x K_i, \qquad \text{and} \qquad K_i^y = D_y K_i,$$

$\mathbf{a}$ is the a_{i0} vector, and the right hand side divergence vector $\mathbf{r}$ is given by

$$r_i = \langle K_i^x, Q^x \rangle + \langle K_i^y, Q^y \rangle.$$

Connection to the Finite Element Method. Our coarsened BIC image space with these projection operations bears some resemblance to the *Finite Element Method* (FEM) [Zienkiewicz and Taylor 2000]. In its traditional form the FEM consists of decomposing the domain into localized sets and defining smooth functions (usually polynomial-based) over them. When solving linear boundary problems using the FEM, the right-hand-side function is also projected onto the discrete space and the solution is obtained by solving a linear system based on an element-to-element dot product matrix known as the *stiffness* matrix. While our construction similarly breaks an image into localized kernels on which we define polynomial basis functions, we use our prior knowledge about the solution to construct our space. Since we expect the solution to have the same smooth regions and edges as the input image, we shape the support of the basis functions in a data-dependent manner based on the input image.

4. RESULTS

Our BIC space is a general representation of images useful for a variety of image manipulations. We demonstrate here how our approach improves the efficiency and

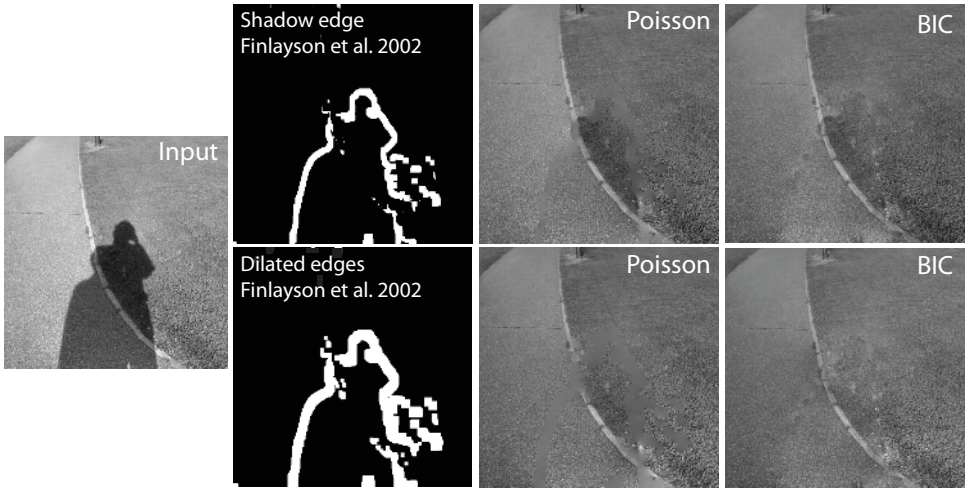

Fig. 5. Gradient based shadow removal. (left) Input image, (top, left to right) we show the shadow edges deletion map (taken from Finlayson et al. 2002) and the results of integrating this shadow-free gradient field via Poisson and over the BIC space. (bottom row) Shows the same operations performed after a 5 pixel dilation of the original shadow edge map.

quality of several applications.

Shadow Removal. We begin with the shadow removal problem. In Section 3.1 we integrated the deleted shadow-edge gradient field using Poisson integration followed by an image projection onto the BIC space. Here we reconstruct an image directly from the gradient field using the BIC gradient projection operator. We formulate the problem as

$$\min_{\mathbf{a}} \|\nabla \log I(\mathbf{x}) + \nabla \sum_i K_i(\mathbf{x}) a_i - (Q_x, Q_y)\|^2$$

where (Q_x, Q_y) is the gradient field of the log image intensities with the shadow edges deleted. In this case the output image is given by $\exp\left(\log I(\mathbf{x}) + \nabla \sum_i K_i(\mathbf{x}) a_i\right)$ where $\exp\left(\nabla \sum_i K_i(\mathbf{x}) a_i\right)$ represents the multiplicative change in illumination. Shadows consist of a constant change in illumination and therefore a natural choice is to compute the kernels based on $\log I$, set $C \equiv 1$, and use first-order construction polynomials which can better capture (and eliminate) gradual change illumination than zero-order representation.

In Figure 5 we apply this to an example gradient field taken from Finlayson et al. [2002]. Note that we pick this specific example, because Finlayson in his original paper did not completely delete all the shadow edges in the modified gradient field, and therefore this example tests how well our approach can prevent bleeding artifacts. As we already saw in Section 3.1, the Poisson-based integration produces blurriness along the deleted shadow edges and bleeding artifacts wherever the shadow edges are not completely deleted. When we integrate the *same* gradient field using the BIC we preserve more detail along the deleted edge. While there are newer and better methods for removing shadows, such as [Finlayson et al. 2006], here we show that by simply switching to the BIC space we can improve the results

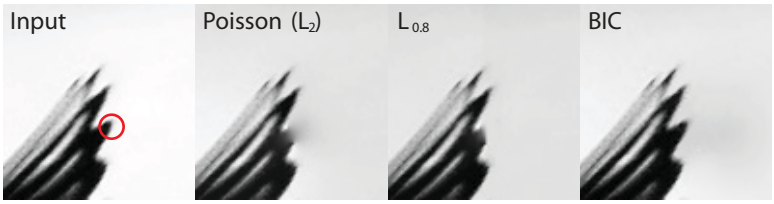

Fig. 6. Integration using L_2, $L_{0.8}$, and the BIC.

obtained by a *given* gradient-domain method. In fact, we can further exploit the robustness of the BIC representation and reduce the artifacts even more by naively dilating the original shadow edge map. We dilate the map by 5 pixels and delete all gradients in this extra band of pixels. This approach is likely to eliminate any remaining shadow edges in the modified gradient field and as shown in Figure 5 the bleeding artifacts are significantly reduced in both the Poisson-integration and BIC results. However, the Poisson-integration result contains more blurriness than before while no blurriness is added to the BIC result.

Sparse-Error Norms. Unlike the L_2 norm, when minimizing under L_α for $\alpha < 1$, errors tend to localize rather than spread. In this norm, when edge gradients are deleted a new relatively sharp edge is formed, hence reducing the bleeding artifacts. Setting $\alpha = 1$ gives the regularization term in the Total Variation Method [Rudin et al. 1992] commonly used in the context of edges preserving noise removal. Lower values of α (such as ≈ 0.8) are used in several image restoration techniques [Mallat 1989; Tappen et al. 2001; Levin et al. 2007] as an image prior probability model that favors images with sparsely distributed derivatives. In Figure 6 we test these image integration alternatives on an edge where some portion of its gradients are deleted. As expected, Poisson-integration yields bleeding artifacts, while integration under the $L_{0.8}$ norm reconstructs a sharper edge, which is a straight line. In the BIC space edges cannot be blurred or change their shape, hence the edge contour is more accurately reconstructed. This result is not surprising because the BIC kernels are built using the input image which contains the edge we wish to maintain. The BIC formulation is designed to exploit information in the original image to avoid bleeding artifact, while the pixel-based integration does not use information in the original image and works only with the modified gradient field. Thus, in cases where the input image is given and the desired image manipulation does not involve altering edge contours, the BIC space outperforms integration under non-quadratic norms in its ability to produce faithful results. Working in the BIC space also avoids the need to solve non-linear equations, involved in non-quadratic optimization problems.

Preconditioning Energy Functionals. Some recent image editing problems such as colorization [Levin et al. 2004], guided tone mapping [Lischinski et al. 2006], and alpha matting [Levin et al. 2006] are formulated using the following weighted derivatives least-squares optimization problem

$$\min_J \sum_{\mathbf{x}} w_x(\mathbf{x})\big(D_x J(\mathbf{x})\big)^2 + w_y(\mathbf{x})\big(D_y J(\mathbf{x})\big)^2, \tag{4}$$

where the weights $w_x(\mathbf{x})$ and $w_y(\mathbf{x})$ are related inversely to the magnitude of the input image gradient vectors. The solution to this quadratic optimization problem

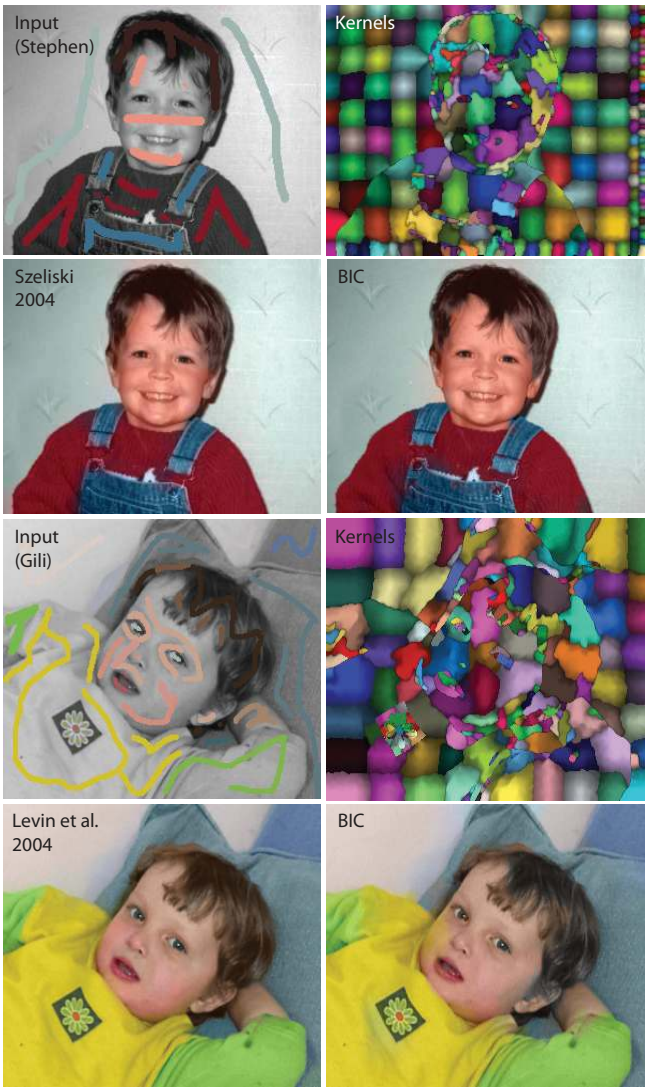

Fig. 7. Image colorization. (first and third rows) Input image, to its right we visualize the BIC kernels. (second and fourth rows) We see the result obtained over the image pixels and the BIC space. Images taken from Szeliski 2007 and Levin et al. 2004.

is computed by solving a spatially-inhomogeneous Laplace equation whose condition number greatly depends on the ratios between these weights. Solving such linear systems is a well-studied topic in numerical computing [Chen 2005] and more recently in computer graphics [Grady et al. 2005; Szeliski 2006]. For example, Szeliski [2006] builds a hierarchical preconditioning basis based on the algebraic structure of the Laplacian matrix, effectively collapsing smooth regions into a single variable.

The functions spanning the BIC space bind together similar pixels and allow

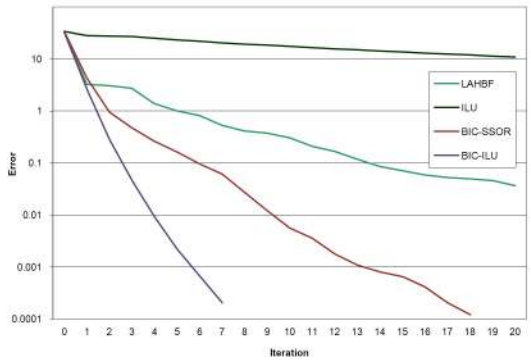

Image	Num. Dofs	Cond. Num.	SSOR	ILU
Stephen	289 66752	73 93543	32 3112	3.2 3628
Gili	158 84800	14 19114	2.6 4772	1.3 470

Fig. 8. Plot shows the RMS error at each conjugate gradient iteration using ILU [Saad 2003], LAHBF [Szeliski 2006], and over the BIC using SSOR and ILU [Hackbusch 1994]. The table shows the number of kernels and pixels, following by the condition numbers of the linear systems resulting from the BIC (left) and regular pixel formulations (right).

independence across edges. Thus, these function are analogous to Szeliski's hierarchal preconditioning basis. We solve the optimization problem in the BIC space by substituting J with $\sum_i K_i(\mathbf{x})P_i(\mathbf{x})$ in (4). The advantage of this formulation is that there are far fewer kernels than pixels, resulting in linear systems that have fewer variables and belong to different scales from the BIC space. For these reasons we expect a well-conditioned system when solving in the BIC space. By working in the BIC space we *first reduce* the model and *then solve* the optimization whereas preconditioning operates on the full linear system. Thus, we inherently obtain a small well-conditioned system and do not have to rely on the performance of a preconditioning scheme to detect independent components. However this approach does not prevent us from using simple preconditioners, such as symmetric successive overrelaxation (SOR) or incomplete LU (ILU) factorization [Hackbusch 1994; Saad 2003], when solving the small linear system resulting in the BIC space.

In Figure 7 we show the results produced using the BIC space for image colorization [Levin et al. 2004]. For this example we constructed the kernels based on the gray-scale input image. For images with pixel values ranging between zero and one, we set $\sigma_r = 0.1$ and use a sampling rate $k = 4$. Since we expect the image colors to be almost piecewise constant, we use a constant normalization factor $C \equiv 1$ and zero-order CPs. Our scale-adaptive coarsening produced only 158 kernels in order to represent the Gili image (bottom example in Figure 7), which is made of 84800 pixels. The resulting small linear system converged after 7 iterations of successive overrelaxation to a root mean squared error of 10^{-7}. In the table given in Figure 8 we compare the condition number of the systems resulting from using standard preconditioners and formulating the colorization in the BIC space. The plot in this figure shows the root mean squared error progression and compares between the BIC and Szeliski's hierarchal preconditioning basis. The error values are computed

against an exact solution in the original pixel-space image representation. The initial errors of the different methods are identical since they use the same initial guesses, a zero image.

We used *LASPack* [Skalicky 1995] as our iterative linear solver; however, the matrices resulting from the BIC space formulation, including the matrices for the examples given here, are small enough as to be solved efficiently using a direct solver. In terms of visual accuracy, the result produced by the BIC on the Stephen image is nearly identical to [Levin et al. 2004]. In the Gili example, this drastic level of coarsening generates kernels that over-cluster pixels matching regions of different color, such as at the right cheek and the yellow shirt. It should be noted that Szeliski's preconditioning applies to any Laplacian matrix resulting from any image manipulation, including ones that alter the edges geometry.

Fig. 9. Alpha matting. (top-left) Shows the input image including a minimal user input specifying the foreground (white circles) and background (blue circles). (top-right) Shows the kernels generated on this image and (bottom-left) shows the result by Levin et al. [2007]. (bottom-right) Shows the resulting matte obtained by the Min-Cut algorithm applied to the BIC kernels rather than image pixels.

Alpha Matting. Levin et al. [2006] use a functional of the form of (4) to extract alpha mattes that separate a foreground object from the background. This quadratic cost function attributes variations in the image to the alpha channel. Hence the alpha matte derivatives are weighted inversely to the magnitude of the input image gradient vectors. In Levin et al. [2007] this cost function is not minimized directly, but rather the alpha mattes are computed based on the few eigenvectors that correspond to the lowest eigenvalues of the minimizing Laplacian matrix.

The coefficients of these eigenvectors are made positive by rotating them within the linear sub-space they define to obtain valid fuzzy matting components. These components are then used to construct mattes either as building blocks for user construction or in an unsupervised fashion. Our scale-adaptive basis functions capture smooth image regions and therefore produce low scores for the matting cost functional. Hence our basis functions reside in the lower part of the Laplacian spectrum of the matting Laplacian and also correspond to linear combinations of the lowest eigenvectors.

In Figure 9 we show the results obtained using the BIC kernels to produce an alpha matte. Given a minimal user input defining the foreground and the background, we propagate this selection using the Min-Cut algorithm of Boykov et al. [2001]. We set the inter-kernel graph weights to $\langle K_i, K_j \rangle^2$, which measures the overlap between the two kernels. In this comparison we see that our results less accurately capture some of the fine details, e.g., the woman's hair. However, our approach is much faster than Levin et al.'s [2007] approach which computes eigenvectors and solves non-linear system of equation. In our approach, the BIC kernels are computed explicitly and it takes about three seconds to compute the BIC representation for the 200 by 200 pixel image shown in Figure 9. Additional running times for computing the BIC representation are given at the end of this section. The parameters we used to generate this example are $k = 8$, $\sigma_r = 0.1$ at the first scale and $\sigma_r = 0.05$ for the adaptive coarsening.

Joint Bilateral Upsampling. Kopf et al. [2007] use bilateral kernels to upsample image operations performed at a low-resolution. While our construction resembles this approach, it can better handle data at multiple scales. When computing the scale-adaptive coarsening, we form large kernels as well as small islands, depending on the image contents. Bilateral upsampling fails when the sampling rate exceeds the minimum size of features in the input image. In these cases the bilateral kernels consist of pixels that differ considerably from the pixel to be interpolated.

In Figure 10 we show this deficiency when colorizing an image that contains small features and compare the results to the ones obtained in the BIC space. We perform colorization using bilateral upsampling as follows. We first downsample the image by a factor of 8 to 4096 pixels, then colorize this smaller image and finally apply the joint bilateral upsampling to produce the result. In this figure, we also show the result obtained by colorizing this image using the BIC formulation as we described earlier. Our scale-adaptive kernel construction generated, on this quarter-megapixel image, 957 kernels. In Figure 10 we see that the results generated by joint bilateral upsampling show some prominent colorization artifacts compared to ones obtained from the BIC. These errors occur despite the fact that the optimization problem it used consists of four times as many variables as in the BIC.

Kopf et al.[2007] show that joint bilateral upsampling can be used for panoramic image stitching [Agarwala et al. 2004]. By treating weights in the bilateral kernels as votes for a labeling in the high resolution, they propagate a Min-Cut labeling computed at a coarse resolution to the full resolution. We use the BIC kernels for stitching in a similar way. We compute a labeling on kernels, produced with $C \equiv 1$, $k = 8$, $\sigma_r = 0.4$ and $\sigma_r = 0.05$ at higher levels, and use their weights in the voting mechanism. Since the labeling assigns each kernel to one of the two input images,

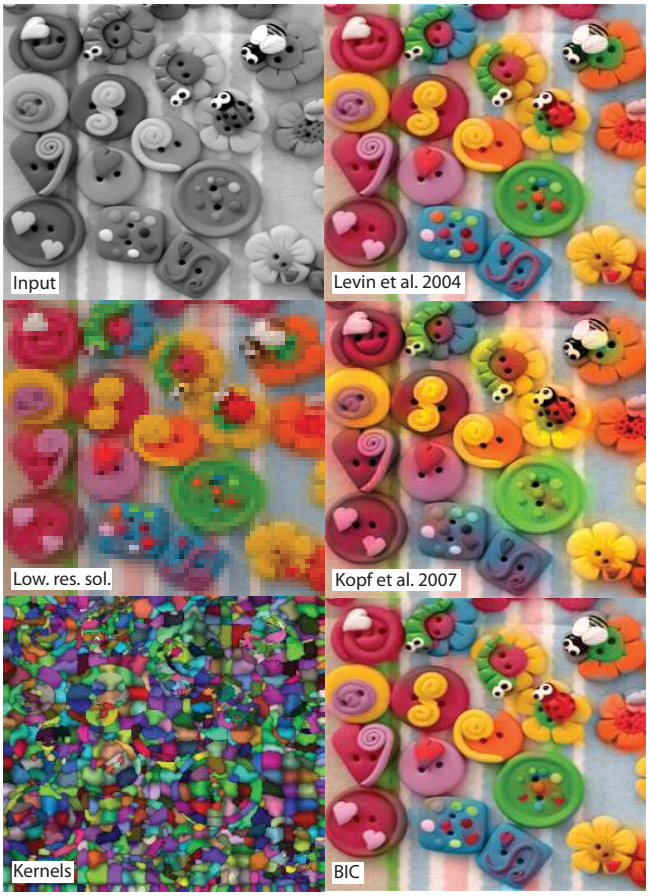

Fig. 10. Colorization using Bilateral Upsampling, Kopf et al. [2007].

we construct a single set of kernels based on the two images by changing the pixel affinity to

$$S(\mathbf{x},\mathbf{y}) = g_s(|\mathbf{x}-\mathbf{y}|)\cdot g_r(|I_1(\mathbf{x})-I_1(\mathbf{y})|)\cdot g_r(|I_2(\mathbf{x})-I_2(\mathbf{y})|).$$

The resulting kernels are shaped by edges from both I_1 and I_2. We construct the labeling graph with a node for each kernel and define the cost along the graph edges as

$$C(K_i,K_j) = (\langle K_i, I_{dif}\rangle + \langle K_j, I_{dif}\rangle)\cdot\langle K_i \star g, K_j \star g\rangle$$

where $I_{dif}(\mathbf{x}) = |I_1(\mathbf{x}) - J_2(\mathbf{x})|$, g is a 2D gaussian, and $\langle K_i \star g, K_j \star g\rangle$ is a measure of the shared boundary between two kernels. This graph has many fewer nodes than pixels, yet the edge weights it uses take into account mismatches between the registered images at the fine-resolution. In Figure 11 we show the results obtained by using the BIC for image stitching.

In [Li et al. 2004] image coarsening combined with a Graph-Cut labeling is used to provide an interactive cutout tool. While our kernels fade gradually and overlap

Fig. 11. Stitching using Min-Cut. Image taken from Kopf et al. 2007

at smooth regions, the Lazy Snapping tool results in a hard segmentation. Lazy Snapping is inadequate for operations that require smooth transitions such as alpha matting and image colorization.

Running Times. Constructing 4169 kernels computed at a single scale from a 512-by-512 pixel image (with $k = 8$) takes less than two seconds on a 3GHz Intel Xeon Pentium machine. Constructing the linear system for gradient projection and solving it takes another 2.7 seconds. A four level scale-adaptive construction yielding 162 kernels takes 2.3 seconds to compute, and only half a second for the gradient projection.

Studying the Parameters Effect. Our construction involves several parameters which affect the resulting set of scale adaptive kernels. Here we demonstrate the effects of these parameters on several examples by reporting the number of BIC kernels our construction produces and by evaluating the effectiveness of our representation for image colorization. In Figure 12 we see the outcomes of setting $\sigma_r = 0.2$ and $\sigma_r = 0.05$ (while in both cases the colorization equations are constructed with $\sigma_r = 0.07$). The figure shows that this parameter controls the kernels separation given the dissimilarity in the grayscale image. Using $\sigma_r = 0.2$ produces only 206 kernels, but it is too high as there is strong color bleeding along some of the edges, such as the edges around the silhouettes of the the pyramid and the camel. The stricter construction, with $\sigma_r = 0.05$, results in 664 kernels and produces a much better result which is comparable and even superior in certain regions to solving the full linear system of Levin et al. [2004]. We also compare two different values of τ, the island-construction threshold. The permissive value $\tau = 0.05$ produces only 274 kernels but shows some artifacts such as the blue sky color penetrating the tip of the pyramid and the camel's nose. At coarse enough scales, these convex regions are not surrounded by enough kernels of a similar color, thus allowing the sky kernels to have a significant contribution when kernels are merged in further coarsening. The stricter value of $\tau = 0.3$ disconnects these kernels before reaching these coarse grid resolutions.

In Figure 13 we see the effect of choosing different values for the sampling factor k. The value $k = 6$ resulted in 121 kernels, $k = 7$ in 96 kernels, and $k = 8$ in

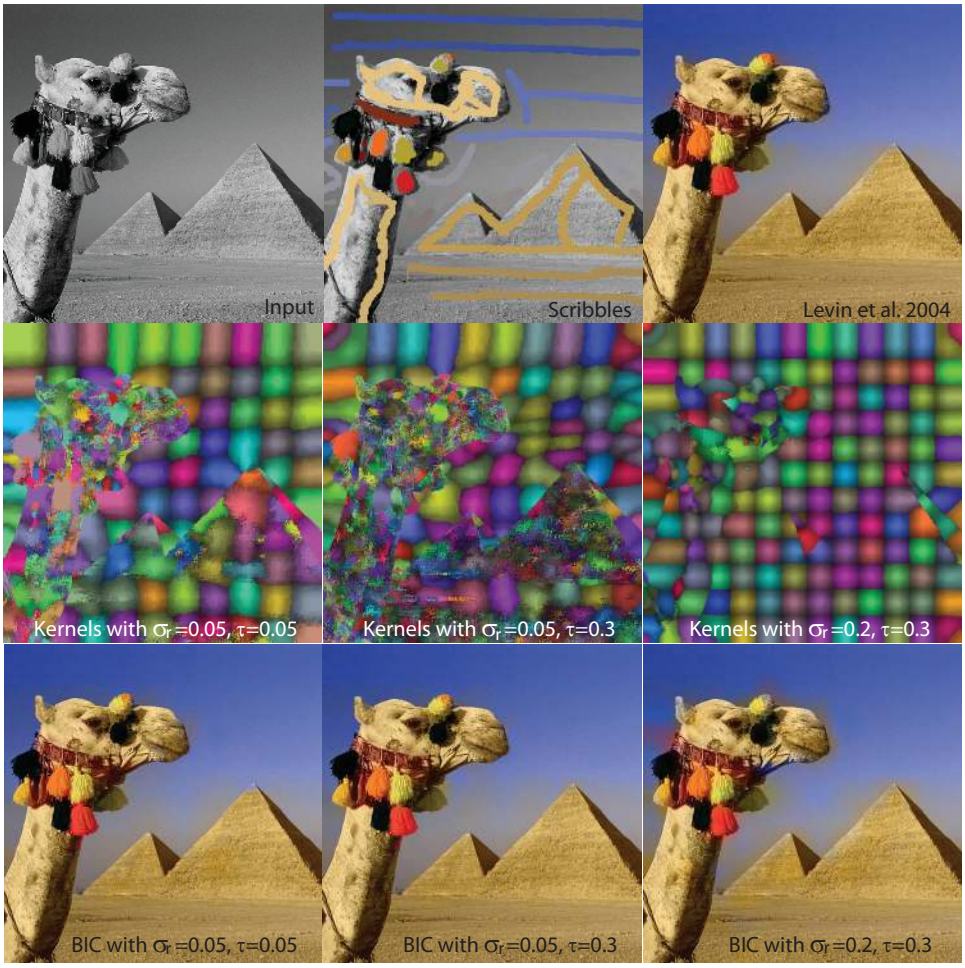

Fig. 12. Varying the threshold for island kernels creation τ and the rage scale, σ_r.

187 kernels. These numbers of kernels are somewhat arbitrary as our construction procedure continues until the multi-scale termination criterion is met. Note that for the quarter megapixel image shown, these numbers correspond to less than 1% of the original number of degrees of freedom. The spatial scale parameter, σ_s, determines the overlap of neighboring kernels (in smooth regions). The value $\sigma_s = 2^l k$ allows the kernel to cover their surrounding and decay before reaching the centers of adjacent kernels. We use this value in all our testing and it appears to be application independent.

Limitations. The bilateral image coarsening space is not well-suited to image processing that consists of scale separations. The kernels we produce either fully contain the image data (when $C \equiv I$), or lack the fine scale detail (when $C \equiv 1$). Thus our method is inappropriate for tasks such as detail enhancement and dynamic range compression. Much like image segmentation algorithms, our scale-adaptive

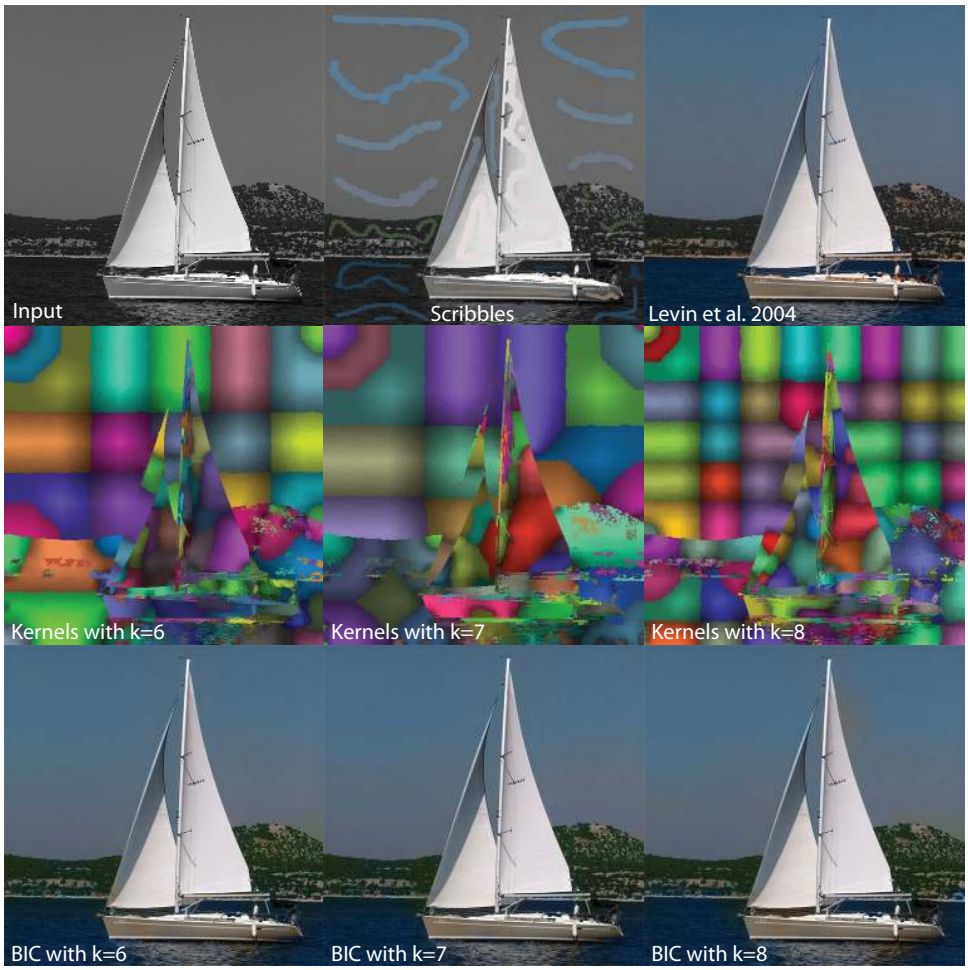

Fig. 13. Varying the sampling rate k from 6 on the left to 8 on the right.

kernel construction may often over or under cluster pixels, depending on the values chosen values for the parameters σ_r and τ. The choice of these parameters usually boils down to a tradeoff between undesirable kernel bindings and the efficiency gain by variable reduction. We intend to improve this binding strategy, as future work, using high-order visual cues to better meet user intentions.

5. CONCLUSIONS

We presented an efficient scheme for constructing a new dimensionally-reduced image space which is data-dependent. In this linear space, pixels are bound together according to the edge contents of the image using bilateral filter kernels. We showed that this reduced representation is useful for many gradient-based image applications where it avoids artifacts, lessens computation, and yields systems that are better conditioned.

Throughout the paper we gave several example applications that benefit from the BIC representation. We believe that this image coarsening can be better tailored to specific applications and achieve better results. For example, we would like to find a pixel-pair affinity function that better matches the one described in Levin et al. [2006] in order to better capture fine details in alpha matting.

REFERENCES

Agarwala, A., Dontcheva, M., Agrawala, M., Drucker, S., Colburn, A., Curless, B., Salesin, D., and Cohen, M. 2004. Interactive digital photomontage. In *ACM Transactions on Graphics*. Vol. 23. 294–302.

Bae, S., Paris, S., and Durand, F. 2006. Two-scale tone management for photographic look. In *ACM Transactions on Graphics*. Vol. 25. 637–645.

Boykov, Y., Veksler, O., and Zabih, R. 2001. Fast approximate energy minimization via graph cuts. *IEEE Trans. Pattern Anal. Mach. Intell. 23,* 11, 1222–1239.

Burt, P. J. and Adelson, E. H. 1987. The laplacian pyramid as a compact image code. 671–679.

Chen, J., Paris, S., and Durand, F. 2007. Real-time edge-aware image processing with the bilateral grid. In *ACM Transactions on Graphics*. Vol. 26. 103.

Chen, K. 2005. *Matrix Preconditioning Techniques and Applications*. Number 19. Cambridge Press.

Durand, F. and Dorsey, J. 2002. Fast bilateral filtering for the display of high-dynamic-range images. In *ACM Transactions on Graphics*. Vol. 21. 257–266.

Efros, A. A. and Leung, T. K. 1999. Texture synthesis by non-parametric sampling. In *International Conference on Computer Vision (ICCV 99)*. 1033–1038.

Eisemann, E. and Durand, F. 2004. Flash photography enhancement via intrinsic relighting. In *ACM Transactions on Graphics*. Vol. 23. 673–678.

Fattal, R., Agrawala, M., and Rusinkiewicz, S. 2007. Multiscale shape and detail enhancement from multi-light image collections. In *ACM Transactions on Graphics*. Vol. 26. 51.

Fattal, R., Lischinski, D., and Werman, M. 2002. Gradient domain high dynamic range compression. In *ACM Transactions on Graphics*. Vol. 21. 249–256.

Finlayson, G. D., Hordley, S. D., Drew, M. S., and Tj, E. N. 2002. Removing shadows from images. *ECCV 2002: European Conference on Computer Vision*, 823–836.

Finlayson, G. D., Hordley, S. D., Lu, C., and Drew, M. S. 2006. On the removal of shadows from images. *IEEE Transactions on Pattern Analysis and Machine Intelligence 28,* 1, 59–68.

Grady, L., Tasdizen, T., and Whitaker, R. T. 2005. A geometric multigrid approach to solving the 2d inhomogeneous laplace equation with internal dirichlet boundary conditions. In *Proceedings of ICIP 2005*. Vol. 2. 642–645.

Hackbusch, W. 1994. *Iterative Solution of Large Sparse Systems of Equations*. Springer-Verlag, Berlin.

Kopf, J., Cohen, M. F., Lischinski, D., and Uyttendaele, M. 2007. Joint bilateral upsampling. In *ACM Transactions on Graphics*. Vol. 26. 96.

Levin, A., Fergus, R., Durand, F., and Freeman, W. T. 2007. Image and depth from a conventional camera with a coded aperture. *26,* 3, 70.

Levin, A., Lischinski, D., and Weiss, Y. 2004. Colorization using optimization. In *ACM Transactions on Graphics*. Vol. 23. 689–694.

...

Levin, A., Lischinski, D., and Weiss, Y. 2006. A closed form solution to natural image matting. In *Computer Vision and Pattern Recognition (CVPR 06)*. IEEE Computer Society, 61–68.

Levin, A., Rav-Acha, A., and Lischinski, D. 2007. Spectral matting. In *Computer Vision and Pattern Recognition (CVPR 07)*. 1–8.

Levin, A., Zomat, A., Peleg, S., and Weiss, Y. 2004. Seamless image stitching in the gradient domain. In *European Conference on Computer Vision (ECCV 04)*.

Li, Y., Sun, J., Tang, C.-K., and Shum, H.-Y. 2004. Lazy snapping. In *ACM Transactions on Graphics*. Vol. 23. 303–308.

Lischinski, D., Farbman, Z., Uyttendaele, M., and Szeliski, R. 2006. Interactive local adjustment of tonal values. In *ACM Transactions on Graphics*. Vol. 25. 646–653.

Mallat, S. G. 1989. A theory for multiresolution signal decomposition: The wavelet representation. *IEEE Trans. Pattern Anal. Mach. Intell. 11,* 7, 674–693.

Pérez, P., Gangnet, M., and Blake, A. 2003. Poisson image editing. In *ACM Transactions on Graphics*. Vol. 22. 313–318.

Petschnigg, G., Agrawala, M., Hoppe, H., Szeliski, R., Cohen, M., and Toyama, K. 2004. Digital photography with flash and no-flash image pairs. In *ACM Transactions on Graphics*. Vol. 23. 664–672.

Rudin, L. I., Osher, S., and Fatemi, E. 1992. Nonlinear total variation based noise removal algorithms. *Phys. D 60,* 1-4, 259–268.

Saad, Y. 2003. *Iterative Methods for Sparse Linear Systems*. Society for Industrial and Applied Mathematics.

Skalicky, T. 1995. Package for solving large sparse systems of linear equations. *http://www.mgnet.org/mgnet/Codes/laspack/html/laspack.html*.

Strang, G. 2003. *Introduction to Linear Algebra, 3rd Edition*. Wellesley Cambridge.

Sun, J., Jia, J., Tang, C.-K., and Shum, H.-Y. 2004. Poisson matting. In *ACM Transactions on Graphics*. Vol. 23. 315–321.

Szeliski, R. 2006. Locally adapted hierarchical basis preconditioning. In *ACM Transactions on Graphics*. Vol. 25. 1135–1143.

Tappen, M. F., Russell, B. C., and Freeman, W. T. 2001. Exploiting the sparse derivative prior for super-resolution and image demosaicing. *3rd Intl. Workshop on Statistical and Computational Theories of Vision*.

Tomasi, C. and Manduchi, R. 1998. Bilateral filtering for gray and color images. In *International Conference on Computer Vision (ICCV 98)*. IEEE Computer Society, 839.

Tumblin, J. and Turk, G. 1999. Lcis: a boundary hierarchy for detail-preserving contrast reduction. In *SIGGRAPH '99: Proceedings of the 26th annual conference on Computer graphics and interactive techniques*. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 83–90.

Xu, L., Qi, F., and Jiang, R. 2006. Shadow removal from a single image. In *ISDA '06: Proceedings of the Sixth International Conference on Intelligent Systems Design and Applications (ISDA'06)*. IEEE Computer Society, 1049–1054.

Zienkiewicz, O. and Taylor, R. 2000. *The Finite Element Method*. Butterworth-Heinemann.

...