Poisson Image Editing

Patrick Pérez* Michel Gangnet† Andrew Blake‡

Microsoft Research UK

Abstract

Using generic interpolation machinery based on solving Poisson equations, a variety of novel tools are introduced for seamless editing of image regions. The first set of tools permits the seamless importation of both opaque and transparent source image regions into a destination region. The second set is based on similar mathematical ideas and allows the user to modify the appearance of the image seamlessly, within a selected region. These changes can be arranged to affect the texture, the illumination, and the color of objects lying in the region, or to make tileable a rectangular selection.

CR Categories: I.3.3 [Computer Graphics]: Picture/Image Generation—Display algorithms; I.3.6 [Computer Graphics]: Methodology and Techniques—Interaction techniques; I.4.3 [Image Processing and Computer Vision]: Enhancement—Filtering;

Keywords: Interactive image editing, image gradient, guided interpolation, Poisson equation, seamless cloning, selection editing

1 Introduction

Image editing tasks concern either global changes (color/intensity corrections, filters, deformations) or local changes confined to a selection. Here we are interested in achieving local changes, ones that are restricted to a region manually selected, in a seamless and effortless manner. The extent of the changes ranges from slight distortions to complete replacement by novel content. Classic tools to achieve that include image filters confined to a selection, for slight changes, and interactive cut-and-paste with cloning tools for complete replacements. With these classic tools, changes in the selected regions result in visible seams, which can be only partly hidden, subsequently, by feathering along the border of the selected region.

We propose here a generic machinery from which different tools for seamless editing and cloning of a selection region can be derived. The mathematical tool at the heart of the approach is the Poisson partial differential equation with Dirichlet boundary conditions which specifies the Laplacian of an unknown function over the domain of interest, along with the unknown function values over the boundary of the domain. The motivation is twofold.

First, it is well known to psychologists [Land and McCann 1971] that slow gradients of intensity, which are suppressed by the Laplacian operator, can be superimposed on an image with barely noticeable effect. Conversely, the second-order variations extracted by the Laplacian operator are the most significant perceptually.

Secondly, a scalar function on a bounded domain is uniquely defined by its values on the boundary and its Laplacian in the interior. The Poisson equation therefore has a unique solution and this leads to a sound algorithm.

So, given methods for crafting the Laplacian of an unknown function over some domain, and its boundary conditions, the Poisson equation can be solved numerically to achieve seamless filling of that domain. This can be replicated independently in each of the channels of a color image. Solving the Poisson equation also has an alternative interpretation as a minimization problem: it computes the function whose gradient is the closest, in the L_2-norm, to some prescribed vector field — the *guidance* vector field — under given boundary conditions. In that way, the reconstructed function interpolates the boundary conditions inwards, while following the spatial variations of the guidance field as closely as possible. Section 2 details this guided interpolation.

We will examine a number of possible choices for the guidance vector field. We show in particular that this interpolation machinery leverages classic cloning tools, both in terms of ease of use and capabilities. The resulting cloning allows the user to remove and add objects seamlessly. By mixing suitably the gradient of the source image with that of the destination image, it also becomes possible to add transparent objects convincingly. Furthermore, objects with complex outlines including holes can be added automatically without the need for painstaking cutting out. These different cloning facilities are presented in Section 3.

As shown in Section 4, the same machinery can also be used to modify the appearance of an image within a restricted domain, while avoiding visible discontinuities on the domain boundary. In particular, the color, the texture, or the illumination of an object can easily be modified without any need for precise delineation of object boundaries. Also, a rectangular image region can be made seamlessly tileable.

Related work The Poisson equation has been used extensively in computer vision. It arises naturally as a necessary condition in the solution of certain variational problems. In the specific context of image editing applications three previous pieces of work are related to the use of the Poisson equation proposed here.

In [Fattal et al. 2002], the gradient field of a High Dynamic Range (HDR) image is rescaled non-linearly, producing a vector field that is no longer a gradient field. A new image is then obtained by solving a Poisson equation with the divergence of this vector field as right-hand-side and under Neumann boundary conditions specifying that the value of the gradient of the new image in the direction normal to the boundary is zero. In contrast, the method we are proposing here can be applied to arbitrary patches selected from an image, not just to the entire image. In order to do this, Neumann boundary conditions on a rectangular outline must be replaced by Dirichlet conditions on an arbitrary outline. A further generalization is to extend the range of nonlinear operations applied to gradients, to include maximum operations and suppression of small gradients, both of which have useful editing functions.

In [Elder and Goldberg 2001], a system is introduced to edit an image via a sparse set of its edge elements (edgels). To suppress an

*e-mail: pperez@microsoft.com
†e-mail:mgangnet@microsoft.com
‡e-mail:ablake@microsoft.com

object, associated edgels are removed; to add an object, associated edgels as well as color values on both sides of each of these edgels are incorporated. The new image is then obtained by interpolating smoothly the colors associated to the new set of edgels. This amounts to solving a Laplace equation (a Poisson equation with a null right hand side) with Dirichlet boundary conditions given by colors around edgels. Editing edgels and associated colors is not always simple. In addition, image details are lost when converting to and from the contour domain, which might be undesirable. The sparse edgel-based representation is indeed incomplete, as opposed to the related representation based on wavelet extrema [Mallat and Zhong 1992], which are complete but less adapted to manual editing.

In [Lewis 2001], spots are removed from fur images by separating out the brightness component from details in a selected region and replacing the brightness by harmonic interpolation (solving a Laplace equation) of the brightness at the selection boundary.

In terms of image editing functionalities, two existing techniques achieve seamless cloning as the basic instance of our system does. The first one is Adobe© Photoshop©7's Healing Brush [Adobe© 2002]. To the best of our knowledge, the technique used by this tool has not been published. Therefore, we don't know whether or not it uses a Poisson solver.

The second technique is the multiresolution image blending proposed in [Burt and Adelson 1983]. The idea is to use a multiresolution representation, namely a Laplacian pyramid, of the images of interest. The content of the source image region is mixed, within each resolution band independently, with its new surrounding in the destination image. The final composite image is then recovered by adding up the different levels of the new composite Laplacian pyramid thus obtained. The technique results in multiresolution mixing where finest details are averaged very locally around the boundary of the selection, while lower frequencies are mixed over much larger distances around these boundaries. This fast technique achieves an approximate insertion of the source Laplacian in the destination region (on the first level of the Laplacian pyramid) whereas we perform this Laplacian insertion exactly via the solution of a Poisson equation. More importantly, multiresolution blending incorporates data from distant source and destination pixels, via the upper levels of the pyramid, within the final composite image. This long range mixing, which might be undesirable, does not occur in our technique. In addition, our system offers extended functionality besides opaque seamless cloning, see Sections 3 and 4.

Finally, whereas we propose a guided interpolation framework, with the guidance being specified by the user, e.g., in the form of a source image in the case of seamless cloning, various interpolation methods have been proposed to fill in image regions automatically using only the knowledge of the boundary conditions. A first class of such approaches is composed of inpainting techniques [Ballester et al. 2001; Bertalmio et al. 2000] where PDE-based interpolation methods are devised such as to continue the isophotes hitting the boundary of the selected region. The PDEs to be solved are more complex than the Poisson equation, and work only for bridging fairly narrow gaps in relatively texture-free regions. Example-based interpolation methods [Barret and Cheney 2002; Bornard et al. 2002; Efros and Leung 1999] where the new image region is synthesized using an arrangement of many small patches are an interesting alternative to inpainting. These methods handle large holes and textured boundaries in a more convincing way. Moreover, they can also be used to import textures as shown in [Efros and Freeman 2001; Hertzmann et al. 2001].

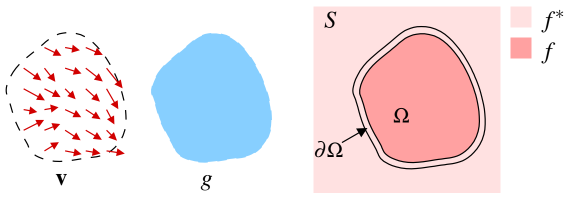

Figure 1: **Guided interpolation notations**. Unknown function f interpolates in domain Ω the destination function f^*, under guidance of vector field $\mathbf{v}$, which might be or not the gradient field of a source function g.

2 Poisson solution to guided interpolation

Guided Interpolation In this section, we detail image interpolation using a guidance vector field. As it is enough to solve the interpolation problem for each color component separately, we consider only scalar image functions. Figure 1 illustrates the notations: let S, a closed subset of $\mathbb{R}^2$, be the image definition domain, and let Ω be a closed subset of S with boundary $\partial\Omega$. Let f^* be a known scalar function defined over S minus the interior of Ω and let f be an unknown scalar function defined over the interior of Ω. Finally, let $\mathbf{v}$ be a vector field defined over Ω.

The simplest interpolant f of f^* over Ω is the membrane interpolant defined as the solution of the minimization problem:

$$\min_f \iint_\Omega |\nabla f|^2 \text{ with } f|_{\partial\Omega} = f^*|_{\partial\Omega}, \tag{1}$$

where $\nabla . = [\frac{\partial .}{\partial x}, \frac{\partial .}{\partial y}]$ is the gradient operator. The minimizer must satisfy the associated Euler-Lagrange equation

$$\Delta f = 0 \text{ over } \Omega \text{ with } f|_{\partial\Omega} = f^*|_{\partial\Omega}, \tag{2}$$

where $\Delta . = \frac{\partial^2 .}{\partial x^2} + \frac{\partial^2 .}{\partial y^2}$ is the Laplacian operator. Equation 2 is a Laplace equation with Dirichlet boundary conditions. For image editing applications, this simple method produces an unsatisfactory, blurred interpolant, and this can be overcome in a variety of ways. One is to use a more complex differential equation as in the "inpainting" technique of [Bertalmio et al. 2000]. The route proposed here is to modify the problem by introducing further constraints in the form of a guidance field as explained below.

A *guidance field* is a vector field $\mathbf{v}$ used in an extended version of the minimization problem (1) above:

$$\min_f \iint_\Omega |\nabla f - \mathbf{v}|^2 \text{ with } f|_{\partial\Omega} = f^*|_{\partial\Omega}, \tag{3}$$

whose solution is the unique solution of the following Poisson equation with Dirichlet boundary conditions:

$$\Delta f = \operatorname{div}\mathbf{v} \text{ over } \Omega, \text{ with } f|_{\partial\Omega} = f^*|_{\partial\Omega}, \tag{4}$$

where $\operatorname{div}\mathbf{v} = \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}$ is the divergence of $\mathbf{v} = (u, v)$. This is the fundamental machinery of Poisson editing of color images: three Poisson equations of the form (4) are solved independently in the three color channels of the chosen color space. All the results reported in this paper were obtained in the RGB color space, but similar results were obtained in CIE-Lab for instance.

When the guidance field $\mathbf{v}$ is conservative, i.e., it is the gradient of some function g, a helpful alternative way of understanding what Poisson interpolation does is to define the correction function $\tilde{f}$ on Ω such that $f = g + \tilde{f}$. The Poisson equation (4) then becomes the following Laplace equation with boundary conditions:

$$\Delta \tilde{f} = 0 \text{ over } \Omega, \ \tilde{f}|_{\partial\Omega} = (f^* - g)|_{\partial\Omega}. \tag{5}$$

Therefore, inside Ω, the additive correction $\tilde{f}$ is a membrane interpolant of the mismatch $(f^* - g)$ between the source and the destination along the boundary $\partial\Omega$. This particular instance of guided interpolation is used for seamless cloning in Section 3.

Discrete Poisson solver The variational problem (3), and the associated Poisson equation with Dirichlet boundary conditions (4), can be discretized and solved in a number of ways.

For discrete images the problem can be discretized naturally using the underlying discrete pixel grid. Without loss of generality, we will keep the same notations for the continuous objects and their discrete counterparts: S, Ω now become finite point sets defined on an infinite discrete grid. Note that S can include all the pixels of an image or only a subset of them. For each pixel p in S, let N_p be the set of its 4-connected neighbors which are in S, and let $\langle p,q\rangle$ denote a pixel pair such that $q \in N_p$. The boundary of Ω is now $\partial\Omega = \{p \in S\setminus\Omega : N_p \cap \Omega \neq \emptyset\}$. Let f_p be the value of f at p. The task is to compute the set of intensities $f|_\Omega = \{f_p,\ p \in \Omega\}$.

For Dirichlet boundary conditions defined on a boundary of arbitrary shape, it is best to discretize the variational problem (3) directly, rather than the Poisson equation (4). The finite difference discretization of (3) yields the following discrete, quadratic optimization problem:

$$\min_{f|_\Omega} \sum_{\langle p,q\rangle\cap\Omega\neq\emptyset} (f_p - f_q - v_{pq})^2, \text{ with } f_p = f_p^*, \text{for all } p \in \partial\Omega, \quad (6)$$

where v_{pq} is the projection of $\mathbf{v}(\frac{p+q}{2})$ on the oriented edge $[p,q]$, i.e., $v_{pq} = \mathbf{v}(\frac{p+q}{2})\cdot\vec{pq}$. Its solution satisfies the following simultaneous linear equations:

$$\text{for all } p \in \Omega, \quad |N_p|f_p - \sum_{q\in N_p\cap\Omega} f_q = \sum_{q\in N_p\cap\partial\Omega} f_q^* + \sum_{q\in N_p} v_{pq}. \quad (7)$$

When Ω contains pixels on the border of S, which happens for instance when Ω extends to the edge of the pixel grid, these pixels have a truncated neighborhood such that $|N_p| < 4$. Note that for pixels p interior to Ω, that is, $N_p \subset \Omega$, there are no boundary terms in the right hand side of (7), which reads:

$$|N_p|f_p - \sum_{q\in N_p} f_q = \sum_{q\in N_p} v_{pq}. \quad (8)$$

Equations (7) form a classical, sparse (banded), symmetric, positive-definite system. Because of the arbitrary shape of boundary $\partial\Omega$, we must use well-known iterative solvers. Results shown in this paper have been computed using either Gauss-Seidel iteration with successive overrelaxation or V-cycle multigrid. Both methods are fast enough for interactive editing of medium size color image regions, e.g., 0.4 s. per system on a Pentium 4 for a disk-shaped region of 60,000 pixels. As demonstrated in [Bolz et al. 2003], multigrid implementation on a GPU will provide a solution for much larger regions.

3 Seamless cloning

Importing gradients The basic choice for the guidance field $\mathbf{v}$ is a gradient field taken directly from a source image. Denoting by g this source image, the interpolation is performed under the guidance of

$$\mathbf{v} = \nabla g, \quad (9)$$

and (4) now reads

$$\Delta f = \Delta g \text{ over } \Omega, \text{ with } f|_{\partial\Omega} = f^*|_{\partial\Omega}. \quad (10)$$

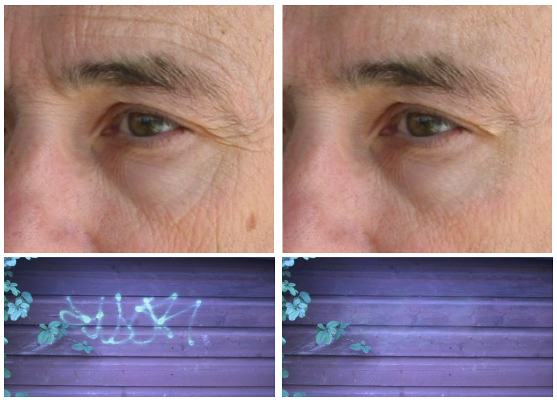
Figure 2: **Concealment**. By importing seamlessly a piece of the background, complete objects, parts of objects, and undesirable artifacts can easily be hidden. In both examples, multiple strokes (not shown) were used.

As for the numerical implementation, the continuous specification (9) translates into

$$\text{for all } \langle p,q\rangle,\ v_{pq} = g_p - g_q, \quad (11)$$

which is to be plugged into (7).

The seamless cloning tool thus obtained ensures the compliance of source and destination boundaries. It can be used to conceal undesirable image features or to insert new elements in an image, but with much more flexibility and ease than with conventional cloning, as illustrated in Figs. 2-4. From the perspective of user input, most tasks will simply require very loose lasso selections, as shown for instance in Fig. 3. However, when features of the source have to be aligned with corresponding features in the destination, as in the fence example in Fig. 2 (bottom row) or the face example in Fig. 4 (top row), the positioning of the source and destination regions must be more precise. Finally, in situations where seamless cloning involves mostly pieces of texture, as in the face touch-up example in Fig. 2 (top row) the texture swap example in Fig. 4 (bottom row) applying repeatedly broad brush strokes is the more effective way.

Up to global changes induced by the interpolation process, the full content of the source image is retained . In some circumstances, it is desirable to transfer only part of the source content. The most common instance of this problem is the transfer of the intensity pattern from the source, not the color. A simple solution is to turn the source image monochrome beforehand, see Fig. 5.

Mixing gradients With the tool described in the previous section, no trace of the destination image f^* is kept inside Ω. However, there are situations where it is desirable to combine properties of f^* with those of g, for example to add objects with holes, or partially transparent ones, on top of a textured or cluttered background.

An example is shown in Fig.6, in which a text layer is to be peeled off the source image and applied to the destination image, without the need for complex selection operations. One possible approach is to define the guidance field $\mathbf{v}$ as a linear combination of source and destination gradient fields but this has the effect of washing out the textures, see Fig. 6.

However, the Poisson methodology allows non-conservative guidance fields to be used, which gives scope to more compelling effect. At each point of Ω, we retain the stronger of the variations in f^* or in g, using the following guidance field:

$$\text{for all } \mathbf{x}\in\Omega,\ \mathbf{v}(\mathbf{x}) = \begin{cases} \nabla f^*(\mathbf{x}) & \text{if } |\nabla f^*(\mathbf{x})| > |\nabla g(\mathbf{x})|, \\ \nabla g(\mathbf{x}) & \text{otherwise.} \end{cases} \quad (12)$$

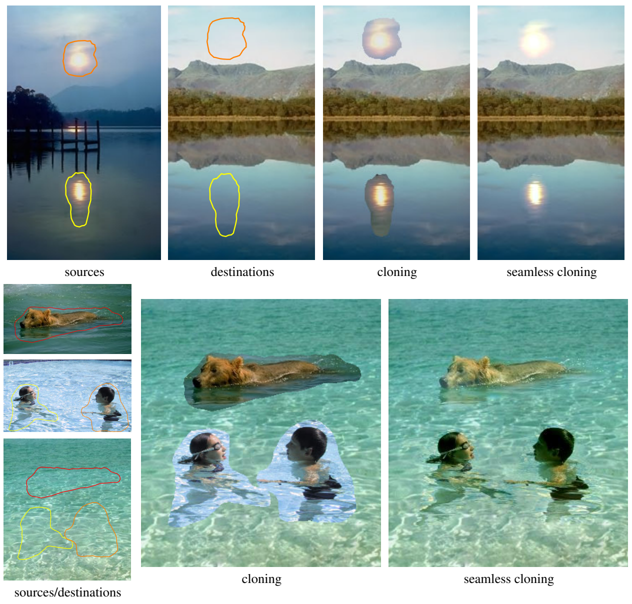

Figure 3: **Insertion**. The power of the method is fully expressed when inserting objects with complex outlines into a new background. Because of the drastic differences between the source and the destination, standard image cloning cannot be used in this case.

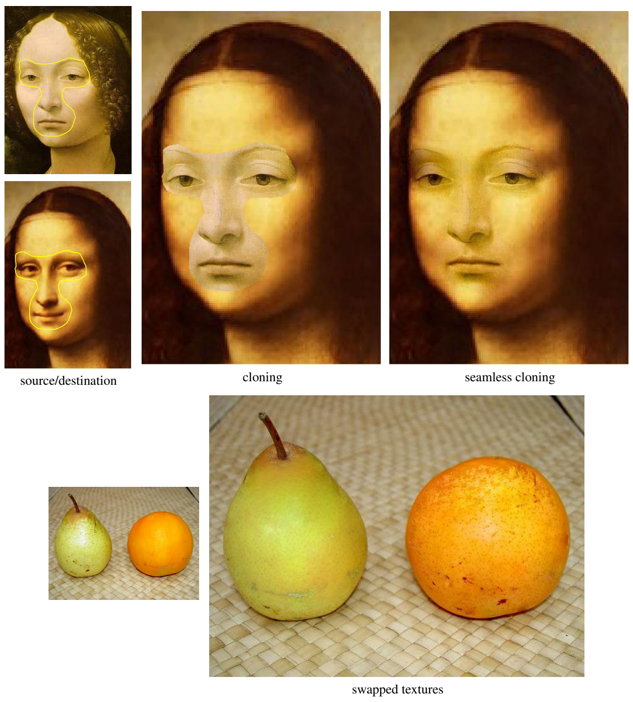

Figure 4: **Feature exchange**. Seamless cloning allows the user to replace easily certain features of one object by alternative features. In the second example of texture swapping multiple broad strokes (not shown) were used.

The discrete counterpart of this guidance field is:

$$v_{pq} = \begin{cases} f_p^* - f_q^* & \text{if } |f_p^* - f_q^*| > |g_p - g_q|, \\ g_p - g_q & \text{otherwise,} \end{cases} \tag{13}$$

for all $\langle p, q \rangle$. The effect of this guidance field is demonstrated in Figs. 6 and 7.

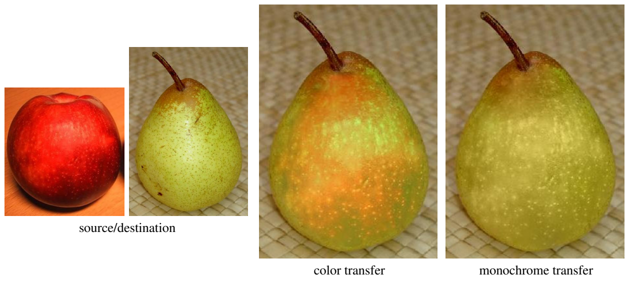

Figure 5: **Monochrome transfer**. In some cases, such as texture transfer, the part of the source color remaining after seamless cloning might be undesirable. This is fixed by turning the source image monochrome beforehand.

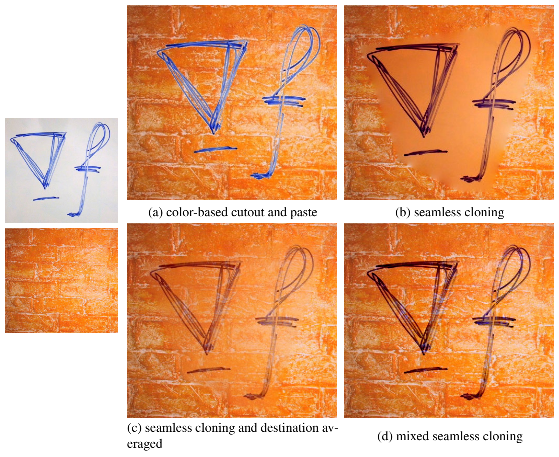
(a) color-based cutout and paste
(b) seamless cloning
(c) seamless cloning and destination averaged
(d) mixed seamless cloning

Figure 6: **Inserting objects with holes**. (a) The classic method, color-based selection and alpha masking might be time consuming and often leaves an undesirable halo; (b-c) seamless cloning, even averaged with the original image, is not effective; (d) mixed seamless cloning based on a loose selection proves effective.

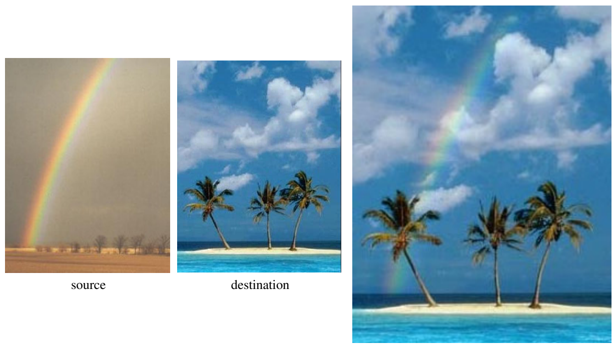

Figure 7: **Inserting transparent objects**. Mixed seamless cloning facilitates the transfer of partly transparent objects, such as the rainbow in this example. The non-linear mixing of gradient fields picks out whichever of source or destination structure is the more salient at each location.

This mixed seamless cloning is also useful when adding one object from a source image very close to another object in the destination image, see Fig. 8.

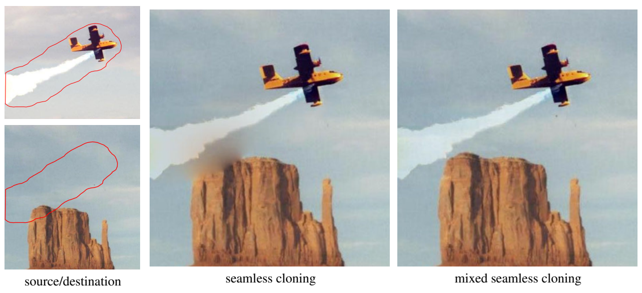

Figure 8: **Inserting one object close to another**. With seamless cloning, an object in the destination image touching the selected region Ω bleeds into it. Bleeding is inhibited by using mixed gradients as the guidance field.

4 Selection editing

In the two previous sections, the guidance field depended, partly or wholly, on the gradient field of a source image g. Alternatively, in-place image transformations can be defined by using a guidance field depending entirely on the original image. Based on this idea, this section details texture flattening, spatially selective illumination changes, background or foreground color modifications, and seamless tiling. The first two effects rely on non-linear modifications of the original gradient field ∇f^* in the selected region. The latter effects rely on in-place seamless cloning after the original image has been modified either inside the domain, providing a new source image, or outside, providing new boundary conditions.

Texture flattening The image gradient ∇f^* is passed through a sparse sieve that retains only the most salient features:

$$\text{for all } \mathbf{x} \in \Omega,\ \mathbf{v}(\mathbf{x}) = M(\mathbf{x})\nabla f^*(\mathbf{x}), \tag{14}$$

where M is a binary mask turned on at a few locations of interest.

A good choice for M is an edge detector, in which case the discrete version of (14), to be plugged into (7), is:

$$v_{pq} = \begin{cases} f_p - f_q & \text{if an edge lies between } p \text{ and } q, \\ 0 & \text{otherwise,} \end{cases} \tag{15}$$

for all $\langle p,q \rangle$. As shown in Fig. 9, the content of the selection Ω gets a flattened appearance, with small grain details washed out, and the main structure preserved. The extent of this effect depends obviously on the sparsity of the sieve. The more selective the edge detector, the sparser the edge map, and the more pronounced the effect.

Note that this instance of Poisson editing has strong connections with the contour-domain editing system of Elder and Goldberg [Elder and Goldberg 2001]. The difference is that we specify approximately the gradient vectors at edge locations through sparse guidance (14), whereas their system relies on an exact specification of color values on both sides of each edgel.

Local illumination changes As pointed out by the authors, the method of [Fattal et al. 2002] is not limited to HDR images and can be applied to ordinary images in order to modify smoothly their dynamic range. First, the gradient field of the logarithm of the image is transformed in order to reduce the large gradients and to increase the small ones. The transformed vector field $\mathbf{v}$ is then used to reconstruct the logarithm of the image, f, by solving the Poisson equation $\Delta f = \text{div}\mathbf{v}$ over the whole image domain under the Neumann boundary conditions.

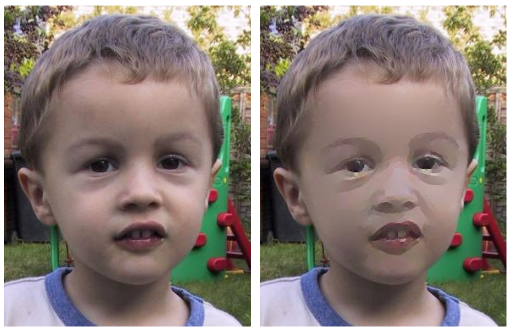

Figure 9: **Texture flattening**. By retaining only the gradients at edge locations, before integrating with the Poisson solver, one washes out the texture of the selected region, giving its contents a flat aspect.

A natural extension is to restrict the correction to a selected region Ω, using appropriate Dirichlet conditions on ∂Ω. Using a simplified version of the Fattal *et al.* transformation [Fattal et al. 2002], the guidance field is defined in the log-domain by:

$$\mathbf{v} = \alpha^{\beta} |\nabla f^*|^{-\beta} \nabla f^*, \tag{16}$$

with $\alpha = 0.2$ times the average gradient norm of f^* over Ω, and $\beta = 0.2$. As shown in Fig. 10, this tool can be used for instance to correct an under-exposed object of interest, or to reduce specular reflections.

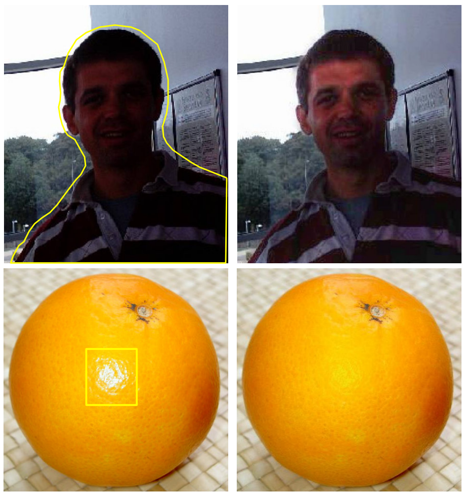

Figure 10: **Local illumination changes**. Applying an appropriate non-linear transformation to the gradient field inside the selection and then integrating back with a Poisson solver, modifies locally the apparent illumination of an image. This is useful to highlight under-exposed foreground objects or to reduce specular reflections.

Local color changes Poisson editing is also a powerful tool for manipulating colors. Given an original color image and a selection Ω, two differently colored versions of this image can be mixed seamlessly: one version provides the destination function f^* outside Ω, the other one provides the source function g to be modified within Ω according to (10).

For example, the task of turning everything in an image monochrome except some object of interest would classically be

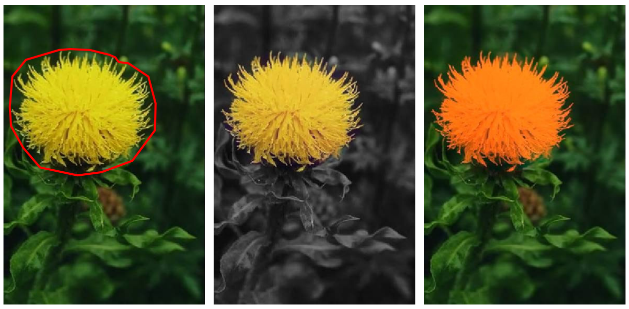

Figure 11: **Local color changes**. Left: original image showing selection Ω surrounding loosely an object of interest; center: background decolorization done by setting g to the original color image and f^* to the luminance of g; right: recoloring the object of interest by multiplying the RGB channels of the original image by 1.5, 0.5, and 0.5 respectively to form the source image.

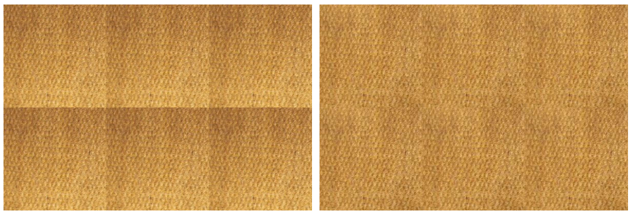

Figure 12: **Seamless tiling**. Setting periodic boundary values on the border of a rectangular region before integrating with the Poisson solver yields a tileable image.

performed by precisely selecting an object and then setting its complement to monochrome. In contrast, Poisson editing frees the user from the tedium of precise selection: given a source color image g, (a) the destination image f^* is set to be the luminance channel from g, (b) the user selects a region Ω containing the object, and this may be somewhat bigger than the actual object, and (c) the Poisson equation (10) is solved in each color channel. An example is presented in Fig. 11. Note that, although the result seems to offer also a precise segmentation of the object for free, this is not actually the case as there is some residual contamination of the destination image outside the object.

Conversely Poisson image editing can be used to modify the color of a loosely selected object. Before solving the Poisson equation (10), the original image is copied to the destination f^* and a version with modified colors is copied to the source g, see Fig. 11.

Seamless tiling When the domain Ω is rectangular, its content can be made tileable by enforcing periodic boundary conditions with the Poisson solver. The source image g is the original image, and the boundary conditions are derived from the boundary values of g, such that opposite sides of the rectangular domain correspond to identical Dirichlet conditions. In Fig. 12, we have chosen $f^*_{\text{north}} = f^*_{\text{south}} = 0.5(g_{\text{north}} + g_{\text{south}})$, and similarly for the east and west borders.

5 Conclusion

Using the generic framework of guided interpolation, we have introduced a variety of tools to edit in a seamless and effortless manner the contents of an image selection. The extent of possible changes ranges from replacement by, or mixing with, another source image region, to alterations of some aspects of the original image inside the selection, such as texture, illumination, or color. An important common characteristic of all these tools is that there is no need for precise object delineation, in contrast with the classic tools that address similar tasks. This is a valuable feature, whether one is interested in small touch-up operations or in complex photomontages.

Although not illustrated in this paper, it is clear that the cloning facilities described in Section 3 can be combined with the editing ones introduced in Section 4. It is for instance possible to insert an object while flattening its texture to make it match the style of a texture-free destination.

Finally, it is worth noting that the range of editing facilities derived in this paper from the same generic framework could probably be extended further. Appearance changes could for instance also deal with the sharpness of objects of interest, thus allowing the user to make apparent changes of focus.

Image credits Two landscapes and swimming bear in Fig. 3, flower in Fig.11: from Corel Professional Photos, copyright ©2003 Microsoft Research and its licensors, all rights reserved; rainbow in Fig. 7 courtesy Professor James B. Kaler, University of Illinois.

References

ADOBE©. 2002. *Photoshop© 7.0 User Guide*. Adobe Systems Incorporated.

BALLESTER, C., BERTALMIO, M., CASELLES, V., SAPIRO, G., AND VERDERA, J. 2001. Filling-in by Joint Interpolation of Vector Fields and Gray Levels. *IEEE Trans. Image Processing 10*, 8, 1200–1211.

BARRET, A., AND CHENEY, A. 2002. Object-Based Image Editing. *ACM Transactions on Graphics 21*, 3, 777–784.

BERTALMIO, M., SAPIRO, G., CASELLES, V., AND BALLESTER, C. 2000. Image Inpainting. In *Proceedings of ACM SIGGRAPH 2000*, ACM Press / ACM SIGGRAPH, New-York, E. Fiume, Ed., Computer Graphics Proceedings, Annual Conference Series, ACM, 417–424.

BOLZ, J., FARMER, I., GRINPSUN, E., AND SCHRÖDER, P. 2003. Sparse Matrix Solvers on the GPU: Conjugate Gradients and Multigrid. *ACM Transactions on Graphics*. to appear.

BORNARD, R., LECAN, E., LABORELLI, L., AND CHENOT, J.-H. 2002. Missing Data Correction in Still Images and Image Sequences. In *Proc. ACM International Conference on Multimedia*.

BURT, P., AND ADELSON, E. 1983. A Multiresolution Spline with Application to Image Mosaics. *ACM Transactions on Graphics 2*, 4, 217–236.

EFROS, A., AND FREEMAN, W. 2001. Image Quilting for Texture Synthesis and Transfer. In *Proceedings of ACM SIGGRAPH 2001*, ACM Press / ACM SIGGRAPH, New-York, E. Fiume, Ed., Computer Graphics Proceedings, Annual Conference Series, ACM, 341–346.

EFROS, A., AND LEUNG, T. 1999. Texture Synthesis by Non-Parametric Sampling. In *Proc. Int. Conf. Computer Vision*, 1033–1038.

ELDER, J., AND GOLDBERG, R. 2001. Image Editing in the Contour Domain. *IEEE Trans. Pattern Anal. Machine Intell. 23*, 3, 291–296.

FATTAL, R., LISCHINSKI, D., AND WERMAN, M. 2002. Gradient Domain High Dynamic Range Compression. *ACM Transactions on Graphics 21*, 3, 249–256.

HERTZMANN, A., JACOBS, C., OLIVER, N., CURLESS, B., AND SALESIN, D. 2001. Image Analogies. In *Proceedings of SIGGRAPH 2001*, ACM Press / ACM SIGGRAPH, New-York, E. Fiume, Ed., Computer Graphics Proceedings, Annual Conference Series, ACM, 327–340.

LAND, E., AND MCCANN, J. 1971. Ligthness and Retinex Theory. *J. Opt. Soc. Amer. 61*, 1–11.

LEWIS, J., 2001. Lifting Detail from Darkness. SIGGRAPH 2001 Tech Sketch.

MALLAT, S., AND ZHONG, S. 1992. Characterization of Signals from Multi-Scale Edges. *IEEE Trans. Pattern Anal. Machine Intell. 14*, 710–732.