Published in Image Processing On Line on 2016–11–18.
Submitted on 2016–01–23, accepted on 2016–09–12.
ISSN 2105–1232 © 2016 IPOL & the authors CC–BY–NC–SA
This article is available online with supplementary materials,
software, datasets and online demo at
https://doi.org/10.5201/ipol.2016.163

Poisson Image Editing

J. Matías Di Martino[1], Gabriele Facciolo[2], Enric Meinhardt-Llopis[2]

[1] Facultad de Ingeniería, Universidad de la República, Uruguay (matiasdm@fing.edu.uy)
[2] CMLA, ENS Cachan, France ({facciolo,enric.meinhardt}@cmla.ens-cachan.fr)

Communicated by Luis Álvarez *Demo edited by* Enric Meinhardt-Llopis

Abstract

The gradient of images can be directly edited to perform useful operations; this is called gradient-based image processing or Poisson editing. For example operations such as seamless cloning, contrast enhancement, texture flattening or seamless tiling can be performed in a very simple and efficient way by combining/modifying the image gradients. In the present work we will describe the Poisson image editing method, and review the contributions that have been made since it was proposed in 2003. In addition the integration problem will be discussed and analyzed, both from the theoretical and numerical points of view. Two different numerical implementations will be discussed, the first one uses discrete versions of differential operators to convert the problem into a sparse linear system of equations, while the second one is based on Fourier transform properties.

Source Code

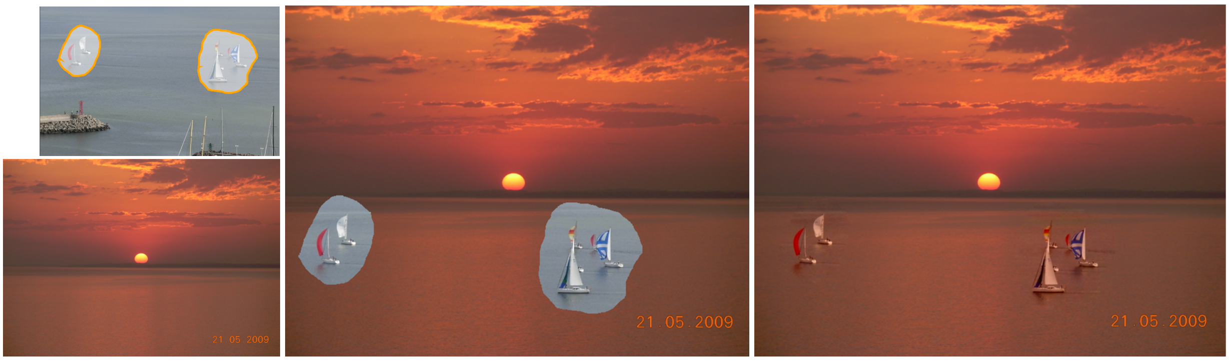
The Octave/Matlab source code, the code documentation, and the online demo are accessible at the IPOL web page of this article[1] and usage instruction are included in the README.txt file of the compressed archive.

Keywords: Poisson editing; image gradient; integration; Poisson equation; seamless cloning; image filtering

1 Introduction

Methods based on the manipulation of image gradients are a powerful tool for processing or combining images. For example operations such as *seamless cloning*, *local illumination changes*, *texture flattening* or *seamless tiling* can be performed in a very simple and efficient way by combining/modifying the image gradients as illustrated in Figure 1. In addition, many practical applications allow to retrieve the gradient field of different physical quantities of interest; for example, Photometric Stereo (PS) [33], Shape from Shading (SfS) [15] and Differential 3D (D3D) [11], retrieve the gradient field

[1] https://doi.org/10.5201/ipol.2016.163



Input Images Cloning Cloning in the gradient domain

Figure 1: Example of the results that can be obtained by mixing the gradients of two source images. In the following sections we will explain in detail how this is achieved.

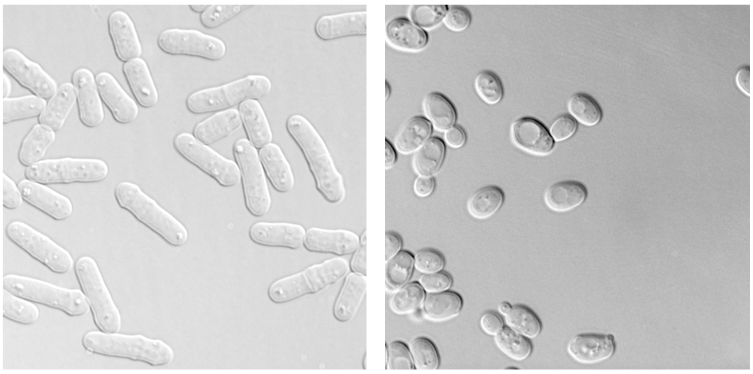


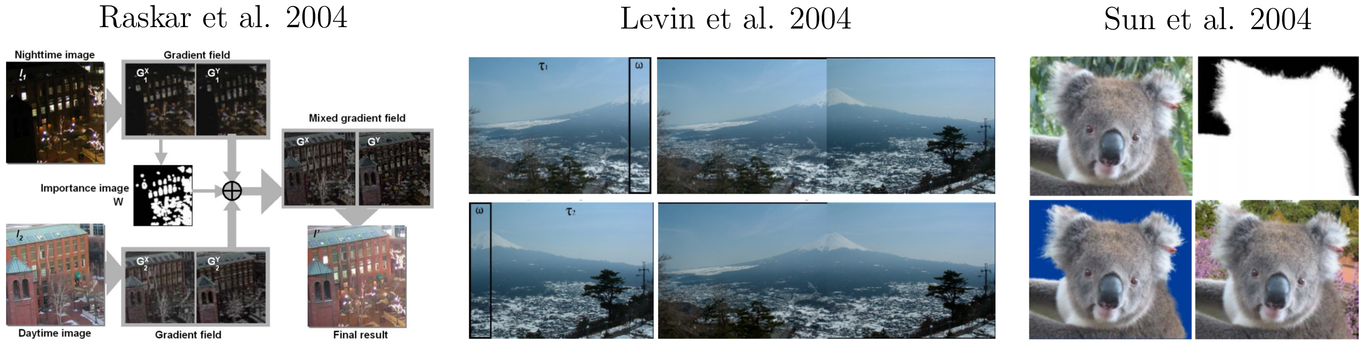
Figure 2: Directional partial derivative of the sample phase obtained by DIC technique (objective 100×), Left: *Schizosaocharomyces pombe* specie, Right: *Saccharomyces cereviseiae* specie [10].

of the depth map of a scene by measuring the variations on the local illumination. Another example are phase gradient methods such as Differential Phase Contrast [14] or Differential Interference Contrast (DIC) [23, 5] in which features of biological samples can be obtained by measuring the phase gradient of a light wavefront. Figure 2 illustrates empirical data obtained by commercial differential microscopes.

Due to the numerous applications and methods that work with the gradient of different quantities, the problem of processing and integrating a given vector field is a very relevant problem. Gradient-based image editing is usually called “Poisson Editing” because the last step involves solving the Poisson equation $\Delta f = \text{div}\,(\mathbf{v})$, where $\mathbf{v}$ is the processed gradient we want to integrate. In the present work we will describe in depth the theory of Poisson Image Editing, and review the contributions that have been made since it was proposed in 2003 [24]. In addition, the integration problem will be discussed and analyzed, both from the theoretical and numerical points of view. Two different numerical implementations will be discussed. The first one uses discrete versions of differential operators to convert the problem into a sparse linear system of equations, while the second one is based on the Fourier transform properties.

1.1 Review of Related Work

Poisson Image Editing was first proposed in 2003 by Pérez et al. [24]. Even though some ideas of combining image gradients have been proposed before 2003, (e.g. Fattal et al. [13] combine the gradient of images taken under different exposure times to achieve a single High Dynamic Range


Raskar et al. 2004

Levin et al. 2004

Sun et al. 2004

Figure 3: Illustrative images of a selected set of references (images extracted from the original articles). Left: flowchart of the technique proposed by Raskar et al. [25]; importance image is derived from only the night time image. Mixed gradient field is created by linearly blending intensity gradients; (image extracted from [25] with permission of the authors). Center: example of the results achieved by Levin et al. [16] for image stitching; On the left are the input images, on top right is a simple pasting of the input images and on the bottom right is the result of the proposed algorithm; (image extracted from [16] with permission of the authors). Right: an example of the results achieved by Sun et al. [29]; top left: a complex natural image for existing matting techniques where the color background is complex, top right: a high quality matte generated by Poisson matting, bottom left: a composite image with the extracted koala and a constant color background, and bottom right: a composite image with the extracted koala and a different background; (image extracted from [29] with permission of the authors)

image), Pérez et al. were the first to formulate the problem in variational terms and to present the most common applications of the Poisson equation to image processing. In the present work we will review most of the ideas proposed by Pérez et al., complementing them with other interpretations and numerical solutions to the problem.

Since 2003 numerous applications and image processing techniques (see Figure 3) have taken advantage of the benefits of working in the gradient domain [19]. For example Raskar et al. [25] proposed in 2004 a technique in which several images taken along the day are combined to exploit different illumination conditions. Their approach was based on a gradient domain technique that preserves important local perceptual cues while avoiding traditional problems such as aliasing, ghosting and haloing. Another example is the work of Sun et al. [29] who proposed, also in 2004, to formulate the problem of natural image matting as one of solving the Poisson equation with the matte gradient field. The authors call this approach Poisson Matting, and show that their new technique is capable of producing good results for many complex images problematic to previous natural image matting methods. The same year Levin et al. [16] proposed an approach for image stitching which combines several individual images that have some overlapped regions; in order to do that they defined a set of cost functions in which the similarity of the images and visibility of the seam were defined in the gradient domain. The authors showed that by combining the image gradients rather than the intensities, global inconsistencies were reduced and the effects of illumination and camera photometric response were minimized. Another alternative to Poisson equation (2) is given by the *linear osmosis equation* $\Delta f(x) = \operatorname{div}(\mathbf{v}(x) f(x))$, which can be used in the same context and has many desirable contrast-invariant properties [32].

In 2005 Tumblin, Agrawal and Raskar [31] showed explicitly the benefits of working in the gradient domain in their work entitled "Why I want a Gradient Camera", in which they proposed a camera that measures static gradients instead of static intensities. To that end they proposed an electronic setup that quantizes sensed intensity differences between adjacent pixel values, and permits an ordinary A/D converter to measure detailed high contrast (HDR) scenes. Afterward, Agrawal et al. proposed

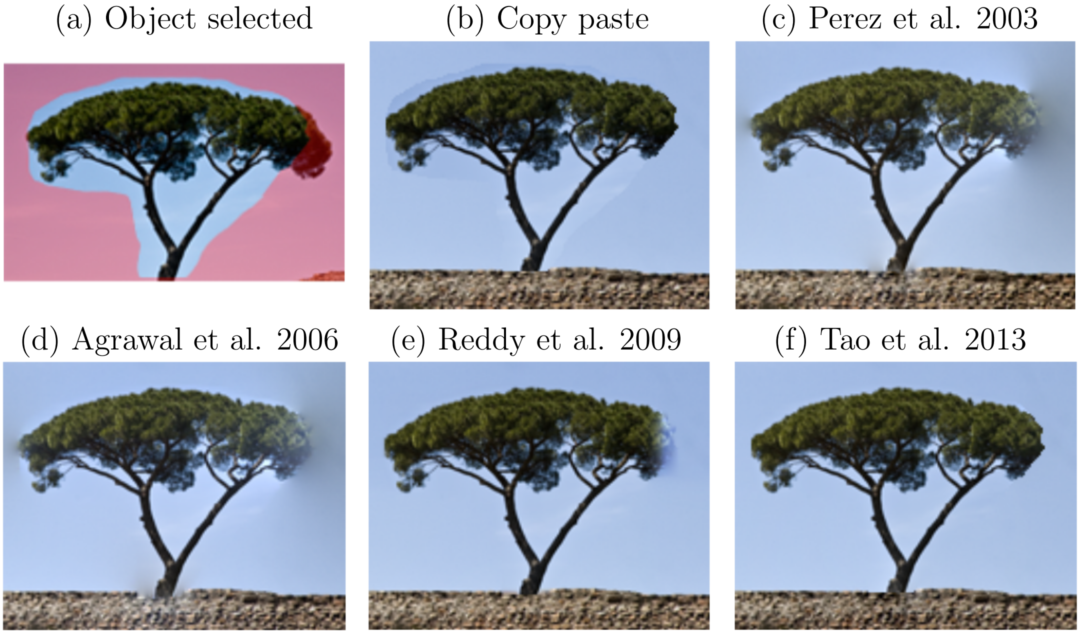


Figure 4: (a) Object selected. (b) Result of copy and paste, (c) Poisson compositing [24]. (d) Result of integrating the gradient field using a diffusion anisotropic tensor [3], (e) L^1 norm minimization [26], and (f) result of applying boundary adjustment and a weighted integration step [30]. Images extracted from [30] with permission of the authors.

an algebraic approach to surface reconstruction from gradient fields (in the context of Shape from Shading (SfS) and Photometric Stereo (PS) techniques) [2, 3]; this work plus Agrawal's PhD thesis [1] are a complete review of different robust integration approaches and explain the main problems that must be faced when working with empirical—noisy—gradient fields (such as the obtained by SfS and PS techniques). In the same direction other works also studied the problem of integrating a *non-integrable field*[2]. For example Du et al. [12] proposed a robust surface reconstruction algorithm by minimizing the absolute error (i.e. the L^1 norm) between the input and the solution gradient field. Reddy et al. [26] proposed a similar approach by finding the gradient field that best fits the input gradient field in the L^1 sense, and presented an exhaustive analysis of the properties of the L^1 solution for gradient field integration using linear algebra and graph analogy. More recently Tao et al. [30] proposed an error-tolerant compositing technique that reduces bleeding artifacts where the composited image regions do not match. The key ideas were, first to define the boundary gradients so that the produced gradient field becomes nearly integrable, and second, to control the integration process to concentrate residuals where they are less conspicuous. To that end the authors modified the Poisson equation in order to locate the residuals in regions where they are less observable, i.e. in regions rich in texture. Figure 4 illustrates some of the common problems of integrating a modified gradient field and the defects that can appear.

Some recent works mix Poisson Editing ideas with patch-based techniques: one example is the work of Shen et al. [28] where a gradient-based image completion algorithm was presented. The authors reconstructed the region of removal in two phases: firstly, they completed the gradient maps in the removed area through a patch-based filling algorithm. And secondly, the image was reconstructed from the gradient maps by solving the Poisson equation. In that work the authors also proposed a matching criterion that uses both the gradient and the color information to achieve a better image completion. Another example is the work of Arias et al. [4] where a variational framework for non-local image inpainting was proposed. The authors derived in a unified way four

[2] i.e. a non-irrotational field.

schemes: *patch NonLocal-means*, *NL-medians*, *NL-Poisson* and *NL-gradient medians*, corresponding to similarity criteria and they used both L^1 and L^2 norms between patches or their gradients to synthesize the inpainted image. More recently Sadek et al. [27] also proposed a variational model for gradient-based video editing. In this work optical flow was used to guide the propagation of the information in the gradient domain. The authors showed that this method was temporally consistent and able to handle fast and abrupt illumination changes in time as well as smooth illumination changes in space. Their experimental results showed that the proposed model was able to handle a large number of frames and relatively complex sequences.

Bhat et al. studied a wide variety of applications of Poisson Equation and proposed a set of techniques and filters using modified versions of it. In 2008 [7] they proposed a variational formulation that combines a data term with a gradient term, the combination of these two kinds of terms allowed them to derive several kinds of filters such as *Sharpening*, *Flattening* and *de-blocking*. For example, a sharpening filter is proposed as a minimization problem where the objective function trades off two terms: a fidelity term to the original image against a fidelity term to the amplified gradients. The image that minimizes this objective function presents boosted high frequencies while its intensity values keeps close to the input image. In this work the authors also present an analysis of the proposed functionals in the Fourier domain, and show that the minimization can be performed in a direct, exact and efficient way in this domain. The work previously described was extended by the same authors in [8, 9]. In these works the authors described in a more general way their proposed framework which combines zero order and first order (i.e. first derivative) fidelity terms. In addition they introduced spatially varying weights over the constraints to approximate the L^1 norm and extend some of their previous ideas to video sequences by adding temporal constraints.

More recently Morel et al. [22] proposed a Fourier implementation of Poisson image editing method, which allows an exact and fast solution of the Poisson equation even when the region of interest has a complex shape. In addition the authors presented an automatic method for selecting the region of interest and applied it to a classic local contrast enhancement principle. The main idea behind this work is to solve the Poisson equation over the entire domain using the modified gradient map inside the region of interest, and the gradient of the background image outside it. In addition Dirichlet boundary conditions are replaced by Neumann conditions (at the edges of the entire image). Even though the proposed (Fourier) implementation solves the Poisson equation over the entire domain of the image (instead of the selected sub-domain), Morel et al. empirically showed that the solutions of both (the exact and the *Fourier based*) implementations are perceptually equivalent outside the sub-domain for a varied set of images. In a similar direction, Limare et al. [18] proposed an implementation in the Fourier domain of the Retinex[3] algorithm as a Poisson Equation with Neumann boundary conditions. The method Limare et al. proposed consists in setting to zero those gradients whose modulus is below certain threshold, and then compute the image that corresponds to the modified gradient map by solving the Poisson equation. Thus, in the new image the small details and shades have been removed, which, according to Retinex theory models the property of our perception of constant colors regardless of their shading. This method is very similar to one of the applications presented in Pérez et al. work [24] as *Texture flattening*, in which only the gradients at edge locations are kept before integrating with the Poisson solver. Finally, in a related work by Morel et al. [21] a functional containing a first order term and a zero order term was used with the aim of removing the effect of nonuniform illumination while preserving the fine details of the image. The proposed approach is similar to the one presented by Bhat et al. in [7], the main difference lies in the data (zero order) term, while Bhat et al. adjust the solution to an input image, Morel et al. adjust the solution to be close to it's own mean value.

[3] Retinex theory was formulated in 1964 by Edwin H. Land, and was the first quantitative attempt to simulate how the human visual system perceives color.

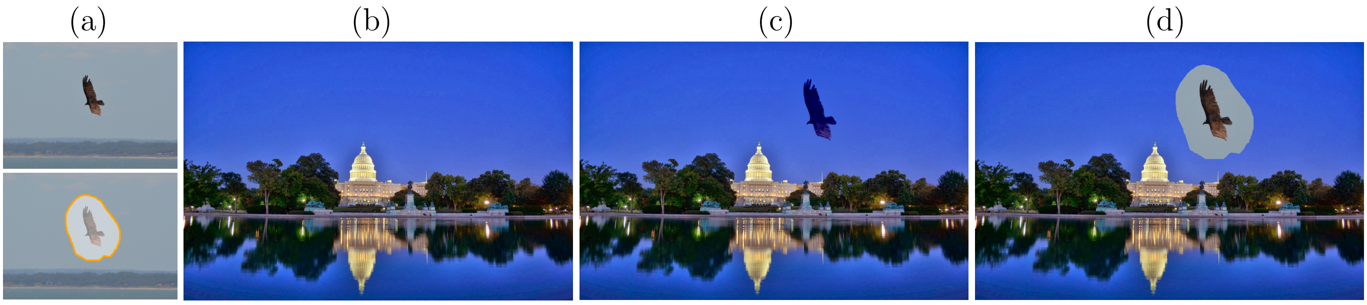


Figure 5: Seamless cloning example. (a) Input image and a selected region of interest, (b) input *Background* image, (c) Poisson editing result, and (d) copy and paste result.

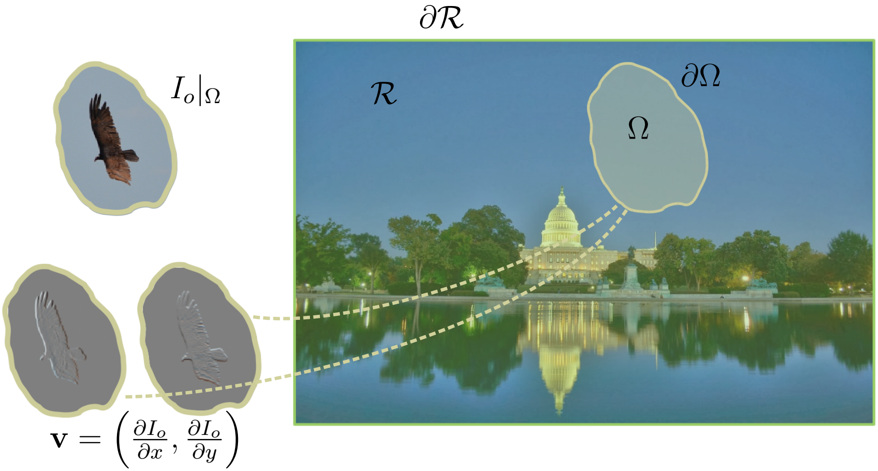

$\partial\mathcal{R}$
$I_o|_\Omega$
$\mathcal{R}$
$\partial\Omega$
Ω
$\mathbf{v} = \left(\frac{\partial I_o}{\partial x}, \frac{\partial I_o}{\partial y}\right)$

Figure 6: Illustration of the relationship between the input data (illustrated in Figure 1) and the notation introduced in Equation (1).

2 Poisson Editing Theory

Let us begin by introducing a classic example (illustrated in Figure 5) that will motivate and clarify the theory we are about to present.

In this example the objective is to seamlessly clone a region selected from an image (Figure 5(a)) over a background image (Figure 5(b)). This problem can be formulated as the variational problem [24]

$$\min_{\substack{f \,\in\, \mathscr{C}^2(\mathcal{R}) \\ \text{st. } f|_{\mathcal{R}\setminus\Omega} = f^*|_{\mathcal{R}\setminus\Omega}}} \int_\Omega \|\nabla f - \mathbf{v}\|^2 \, dx, \tag{1}$$

where $\mathcal{R}$ is a closed subset of $\mathbb{R}^2$ and represent the image domain, $\mathscr{C}^2(\mathcal{R})$ represents the set of real functions twice differentiable over the interior of $\mathcal{R}$, $\Omega \subset \mathcal{R}$, f^* represents the destination (background) image, and finally $\mathbf{v}$ is a differentiable gradient field obtained from the *selected region* as it is illustrated in Figure 6. The solution of Equation (1) must satisfy the Euler-Lagrange equation (see Appendix)

$$\Delta f(x) = \operatorname{div}(\mathbf{v}(x)) \quad \text{for all } x \in \Omega, \text{ and } f|_{\partial\Omega} = f^*|_{\partial\Omega}, \tag{2}$$

and of course outside $\bar{\Omega}$ the solution is trivial and takes the same values of f^*. Intuitively Equation (2) allows us to fill-in the region of interest following the variations (of intensity and color) that correspond to the scene we are inserting, and at the same time it creates a solution which is coherent with the intensities of the background image. See in the example illustrated in Figure 5(c). Equation (2) is called the Poisson equation.

This mathematical approach is very versatile as we shall see in Section 4, where a wide variety of applications and methods based on Poisson Equation are discussed. As we will demonstrate, very different results can be obtained with slight and simple modifications on the definition of the gradient field $\mathbf{v}$ and the region Ω.

3 Numerical Solution

In this section, two numerical methods to solve the Poisson Equation will be presented and discussed. The first approach we will discuss is a *finite difference method* similar to the original solution proposed by Pérez et al. [24]. The second method called *Fourier method* [22] is based on Fourier transform properties to achieve an efficient and simple solution to the problem.

Input Data and Notation

To focus on the numerical solution of the Poisson equation we will assume that the input data consists of: a destination image I, a sub-domain Ω, and a guidance vector field $\mathbf{v}$ defined over Ω. In the numerical solution we will assume that all the inputs have the same rectangular grid structure, i.e. I is an $H \times W$ $(= \mathcal{R})$ gray-level[4] image, Ω an $H \times W$ binary image, and $\mathbf{v}$ composed by two images v^x and v^y (containing x and y partial derivatives respectively) each of them also $H \times W$ images (recall that $\mathbf{v}$ is relevant only inside the domain $\Omega \cup \partial\Omega$ so one can fill v^x and v^y images with arbitrary values outside this domain).

3.1 Finite Difference Approach

A classic numerical approach to solve Equation (2) consists in approximating the differential operators using finite differences between adjacent pixels. Then the numerical solution of Equation (2) can be obtained by solving a sparse linear system $Ax = b$.

This requires defining discrete counterparts of the continuous formalism introduced above. We will assume that the destination image is defined on a regular rectangular grid of size $H \times W (= \mathcal{R})$ (for color images the solution is computed for each channel independently). The subset Ω is an arbitrary set of pixels in $\mathcal{R}$ that may or may not intersect the edge of the rectangular domain, in addition the set $\partial\Omega$ is defined as the set of pixels in $\mathcal{R}/\Omega$ for which at least one of its four neighbors (in the up, down, left and right directions) belongs to Ω, as illustrated in Figure 7. It is important to notice that in those cases in which the set Ω intersects $\partial\mathcal{R}$, Dirichlet boundary conditions are replaced by Neumann boundary conditions on $\Omega \cup \partial\mathcal{R}$ (i.e. $\frac{\partial Z}{\partial n} = 0$ where n represents the outward orthogonal direction on $\partial\mathcal{R}$). This is the case for example for the 5^{th}, 6^{th} and 7^{th} pixels of the last column in the illustration given in Figure 7.

For an image Z of size $H \times W$ we will define the vector z ($N \times 1$, with $N = HW$) obtained by concatenating each column of Z under the previous column (as illustrated in Figure 8). Partial derivatives will be computed using finite differences e.g. we can use a forward scheme[5] to define x and y partial derivatives, i.e. $\mathcal{D}_x z = (Z_{x+1,y} - Z_{x,y})$ and $\mathcal{D}_y z = (Z_{x,y+1} - Z_{x,y})$ where $\mathcal{D}_{x/y}$ are sparse $N \times N$ matrices. In particular, for the edges $x = W$ and $y = H$ we define the x and y partial derivative as zero respectively. Analogously a sparse matrix $\mathcal{L}$ can be defined to compute the Laplacian

$$\mathcal{L}z = -4Z_{x,y} + Z_{x+1,y} + Z_{x-1,y} + Z_{x,y+1} + Z_{x,y-1}. \tag{3}$$

[4] The extension to color images is trivially performed by processing each color channel independently.

[5] In the next subsection the impact of choosing different schemes e.g. Backward or Centered differences will be analyzed.

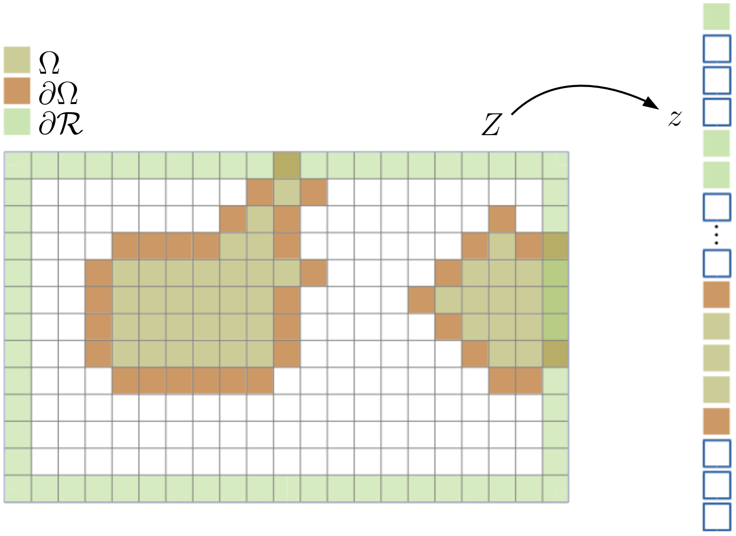


Figure 7: Vectorization of the discrete domain for the numerical scheme.

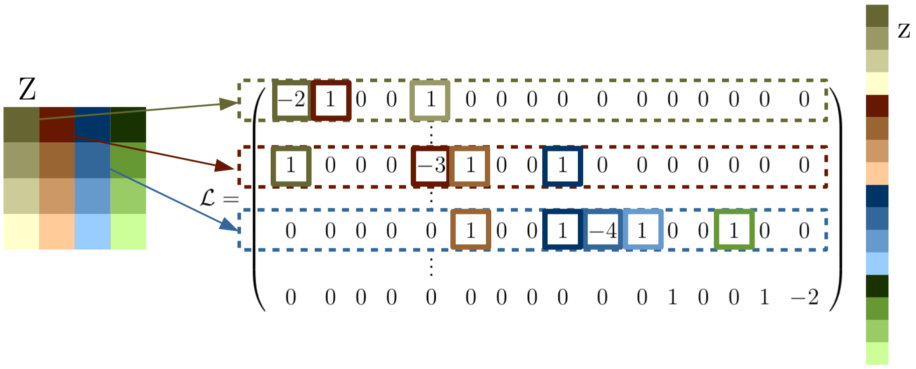


Figure 8: Illustration of the definitions of discrete operators.

Neumann conditions at $\partial\mathcal{R}$ are imposed implicitly by defining the Laplacian operator as illustrated in Figure 8, where the derivatives in the outward direction on $\partial\mathcal{R}$ are assumed to be zero.

From now on, we will refer to v^x, v^y, $\mathbf{1}_\Omega$, $\mathbf{1}_{\partial\Omega}$ and I as the column vectors ($HW = N \times 1$) associated to each respective matrix. The vectors are defined from the original matrices as illustrated in Figure 8.

The last step before proceeding with the solution of Equation (2) consists in defining the following projection matrices:

- A projection matrix for the domain Ω defined as

$$\mathcal{P}_\Omega(i,j) = \begin{cases} \mathbf{1}_\Omega(i) & \text{if } i = j, \quad i = 1 \dots N \\ 0 & \text{otherwise,} \end{cases} \quad (N \times N \text{ sparse matrix}), \tag{4}$$

  where $\mathcal{P}_\Omega$ is a $N \times N$ sparse matrix that contains in its diagonal the vectorized elements of Ω. Recall that Ω is a binary vector of lenght N containing 1 in those pixels in the subset Ω and 0 otherwise.

- A projection matrix for $\partial\Omega$ is defined as

$$\mathcal{P}_{\partial\Omega}(i,j) = \begin{cases} \mathbf{1}_{\partial\Omega}(i) & \text{if } i = j, \quad i = 1 \dots N \\ 0 & \text{otherwise.} \end{cases} \quad (N \times N \text{ sparse matrix}), \tag{5}$$

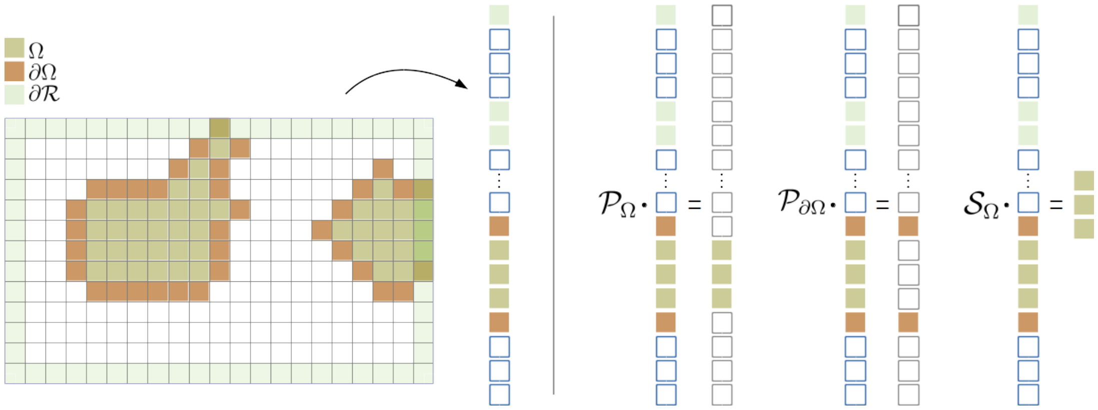

Ω
∂Ω
∂R
$\mathcal{P}_\Omega \cdot$ =
$\mathcal{P}_{\partial\Omega} \cdot$ =
$\mathcal{S}_\Omega \cdot$ =

Figure 9: Illustration of the definition of projection and sampling matrices.

- A sampling matrix that samples the pixels that lie on Ω is defined as

$$\mathcal{S}_\Omega(i,j) = \begin{cases} 1 & \text{if } j = \texttt{idx}(i), \quad i = 1 \dots n \\ 0 & \text{otherwise,} \end{cases} \quad (n \times N \text{ sparse matrix}), \tag{6}$$

where the variable n corresponds to the number of nonzero elements on Ω and `idx` is an $n \times 1$ vector that contains the position (index) in the vector Ω of each nonzero element.

To illustrate the purpose of these projection matrices, let us consider an input image U ($W \times H$ matrix) and the vector u ($WH \times 1$) created by concatenating each column of U, then the vector $u_\Omega = \mathcal{P}_\Omega u$ will have the same values as u for those pixels on Ω and zero in all the other entries. Analogously $u_{\partial\Omega} = \mathcal{P}_{\partial\Omega} u$ will have the same values of u for those pixels on ∂Ω and 0 otherwise. On the other hand, $s_\Omega = \mathcal{S}_\Omega u$ will be a vector with less elements than u containing just the pixels of u on Ω. The products $\mathcal{P}_{\partial\Omega} u$, $\mathcal{P}_{\partial\Omega} u$ and $\mathcal{S}_\Omega u$ are graphically illustrated in Figure 9.

At this point we have all the ingredients to write in an easy and intuitive way the discretization of the Poisson equation

$$\Delta f(x) = \operatorname{div}(\mathbf{v}(x)) \quad x \in \Omega, \text{ and } f|_{\partial\Omega} = I|_{\partial\Omega}. \tag{7}$$

We start by replacing continuous differential operators by their discrete counterparts,

$$\mathcal{L}f = \mathcal{D}_x v^x + \mathcal{D}_y v^y. \tag{8}$$

The previous equation needs to be restricted to those pixels that belong to the discrete set Ω. Notice that to compute $\mathcal{L}f$ over Ω, only the pixels in $\Omega \cup \partial\Omega$ are involved, then we can replace f by $(\mathcal{P}_\Omega + \mathcal{P}_{\partial\Omega})f$ and the equations associated to pixels in Ω will remain unchanged

$$\begin{aligned} &\mathcal{L}((\mathcal{P}_\Omega + \mathcal{P}_{\partial\Omega})f) = \mathcal{D}_x v^x + \mathcal{D}_y v^y \\ &\Rightarrow \mathcal{L}\mathcal{P}_\Omega f + \mathcal{L} \underbrace{\mathcal{P}_{\partial\Omega} f}_{= \mathcal{P}_{\partial\Omega} I} = \mathcal{D}_x v^x + \mathcal{D}_y v^y \\ &\Rightarrow \mathcal{L}\mathcal{P}_\Omega f = \mathcal{D}_x v^x + \mathcal{D}_y v^y - \mathcal{L}\mathcal{P}_{\partial\Omega} I. \end{aligned} \tag{9}$$

The last equation is still a system of N linear equations with N unknowns, the last step consists in keeping only the n relevant equations and unknowns (those that correspond to pixels in the interior of Ω). To that end we can simply keep the set of equations and unknowns as

$$\mathcal{S}_\Omega \mathcal{L} \mathcal{P}_\Omega \mathcal{S}_\Omega^T x = \mathcal{S}_\Omega[\mathcal{D}_x v^x + \mathcal{D}_y v^y - \mathcal{L}\mathcal{P}_{\partial\Omega} I], \tag{10}$$

where x is an $n \times 1$ column vector with the unknowns pixels in Ω. This final equation solves the Poisson equation over the domain Ω imposing Dirichlet boundary conditions $f|_{\partial\Omega} = I|_{\partial\Omega}$ plus Neumann boundary conditions in those regions in which Ω intersects the boundary of the domain $\mathcal{R}$ (if it does). In our implementation the linear system (10) is solved using the Matlab *backslash* function, i.e. $x = A\backslash b$.

The main steps of the finite difference approach are summarized in Algorithm 1.

Algorithm 1: Finite differences approach to gradient-based image manipulation

```
input  : Destination image $I$ and edited vector field $\mathbf{v}$ defined inside $\Omega \cup \partial\Omega$
output: Edited image $\widetilde{I}$
Define sparse matrices: $\mathcal{L}$, $\mathcal{D}_x$, $\mathcal{D}_y$, $\mathcal{P}_\Omega$, $\mathcal{P}_{\partial\Omega}$, and $\mathcal{S}_\Omega$.
Compute $(n \times n)$ matrix $A = \mathcal{S}_\Omega \mathcal{L} \mathcal{P}_\Omega \mathcal{S}_\Omega^T$ and vector $b = \mathcal{S}_\Omega[\mathcal{D}_x v^x + \mathcal{D}_y v^y - \mathcal{L}\mathcal{P}_{\partial\Omega} I]$.
Solve $Ax = b$
$\widetilde{I} \leftarrow I$
$\widetilde{I}(\Omega) \leftarrow x$
```

3.2 Fourier Approach

The discrete Fourier transform can be computed very efficiently for rectangular and regular domains, by transforming the problem

$$\Delta f(x) = \operatorname{div}(\mathbf{v}(x)) \quad x \in \Omega, \text{ and } f|_{\partial\Omega} = I|_{\partial\Omega}, \tag{11}$$

into an approximate problem,

$$\Delta f(x) = \operatorname{div}(\mathbf{V}(x)) \quad x \in \mathcal{R}, \text{ and } \nabla f \cdot \nu|_{\partial\mathcal{R}} = 0, \tag{12}$$

where instead of solving the problem restricted to the domain Ω the solution is computed in the entire (rectangular) domain $\mathcal{R}$, and Dirichlet boundary conditions $\partial\Omega$ are replaced by Neumann boundary conditions $\partial\mathcal{R}$. In order to have a proper solution of the problem the gradient field in the extended domain is defined as

$$\mathbf{V}(x) = \begin{cases} \mathbf{V}(x) = \mathbf{v}(x) & \text{if } x \in \Omega \\ \mathbf{V}(x) = \nabla I(x) & \text{otherwise} \end{cases} \tag{13}$$

as illustrated in Figure 10.

To solve Equation (12) we will start by recalling the definition of the Discrete Fourier Transform (DFT) for a discrete bidimensional signal Z of size $(H \times W)$

$$\hat{Z}(u,v) = \sum_{x=0}^{W-1} \sum_{y=0}^{H-1} Z(x,y) e^{-2\pi i \left(\frac{xu}{W} + \frac{yv}{H}\right)} \tag{14}$$

and its inverse,

$$Z(x,y) = \frac{1}{WH} \sum_{u=0}^{W-1} \sum_{v=0}^{H-1} \hat{Z}(u,v) e^{2\pi i \left(\frac{xu}{W} + \frac{yv}{H}\right)}. \tag{15}$$

Differentiating the previous equations we have

$$\widehat{\frac{\partial Z}{\partial x}} = \left(\frac{2\pi i}{W} u\right) \hat{Z} \quad \text{and} \quad \widehat{\frac{\partial Z}{\partial y}} = \left(\frac{2\pi i}{H} v\right) \hat{Z}. \tag{16}$$

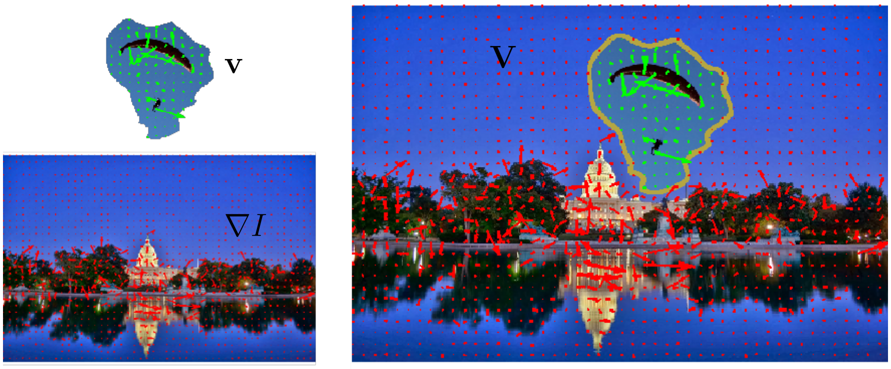


Figure 10: Extension of the guidance vector field $\mathbf{V} = (V^x,\, V^y)$ to complete the domain $\mathcal{R}$

By computing the DFT on both sides of Equation (12) and using Equation (16) we obtain

$$\left[\left(\frac{2\pi i}{W}\right)^2 u^2 + \left(\frac{2\pi i}{H}\right)^2 v^2\right]\hat{f} = \left(\frac{2\pi i}{W}u\right)\widehat{V^x} + \left(\frac{2\pi i}{H}v\right)\widehat{V^y}. \tag{17}$$

Hence the DFT coefficients of the solution of Equation (12) can be obtained as

$$\hat{f}_{uv} = \begin{cases} \dfrac{\left(\frac{2\pi i}{W}u\right)\widehat{V^x}_{uv} + \left(\frac{2\pi i}{H}v\right)\widehat{V^y}_{uv}}{\left[\left(\frac{2\pi i}{W}\right)^2 u^2 + \left(\frac{2\pi i}{H}\right)^2 v^2\right]} & u = 1\cdots(W-1)\ ,\ v = 1\cdots(H-1) \\ & \\ f_{00} & u = 0,\, v = 0 \end{cases}, \tag{18}$$

where the coefficient $\hat{f}_{00}$ can by fixed to the mean value of the output solution f. This additional degree of freedom is because the minimization problem given by Equation (2) is invariant up to a constant value when we replace Dirichlet by Neumann boundary conditions (which has to be done to solve the problem in the extended domain).

Even though we can obtain a solution of $\Delta f = \operatorname{div}(\mathbf{V})$ as $f = \mathtt{DFT}^{-1}(\hat{f})$ (with $\hat{f}$ given by Equation (18)), there is no guarantee that the obtained solution verifies the Neumann boundary conditions $\nabla f \cdot \nu|_{\partial\mathcal{R}} = 0$. To meet this condition we extend the input gradient field $\mathbf{V}$ so that the components V^x and V^y are antisymmetric with respect to the x and y axis respectively, i.e. $V^x_{ext}(x,y) = \mathtt{sg}(x)V^x(x,y)$ and $V^y_{ext}(x,y) = \mathtt{sg}(y)V^y(x,y)$ ($\mathtt{sg}$ denotes the sign function). By extending the original input images symmetrically, we obtain extended images which are four times bigger and become symmetric and periodic. The Neumann condition is ensured because by uniqueness the solution is symmetric across the boundaries. Then we solve Equation (18) for the extended gradient field, take the inverse DFT and finally restrict the solution to the original domain.

The key steps of the Fourier approach are summarized in Algorithm 2.

3.3 Remarks

In this section we will discuss some properties of the presented methods, and their relationship with the original formulation presented by Pérez et al. [24].

Choice of the gradient discretization. In a continuous setting if we take a region I_Ω out of a given image I, compute its gradient field $\mathbf{v} = \nabla I_\Omega$ and solve the Poisson equation $\Delta u = \operatorname{div}(\mathbf{v})$ in

```
Algorithm 2: Fourier approach to gradient-based image manipulation
  input  : Destination image $I$ and edited vector field $\mathbf{v}$ defined inside $\Omega \cup \partial\Omega$
  output: Edited image $\widetilde{I}$
1 Compute the gradient field $\mathbf{V}$ combining $\mathbf{v}$ over $\Omega$ and $\nabla I$ over $\mathcal{R}/\Omega$
2 $\mathbf{W} \leftarrow$ Extend the gradient field $\mathbf{V}$ by symmetrization
3 $\widehat{\mathbf{W}} \leftarrow$ Compute the DFT for each component of $\mathbf{W}$
4 $\widehat{\mathbf{f}} \leftarrow$ Compute the coefficients of the solution according to Equation (18)
5 $\widetilde{I} \leftarrow$ Compute the inverse DFT of $\widehat{\mathbf{f}}$ and restrict the solution to the original domain
```

Ω with boundary conditions from I we recover exactly the original image. However, in a discrete setting this may not be the case for many discretization choices of ∇ and Δ.
Indeed, in Figure 11 are shown the Poisson editing results obtained with different methods for computing the gradients, and different methods for solving the problem (i.e. finite difference and Fourier). Note that the difference is zero only when the two discretizations match.

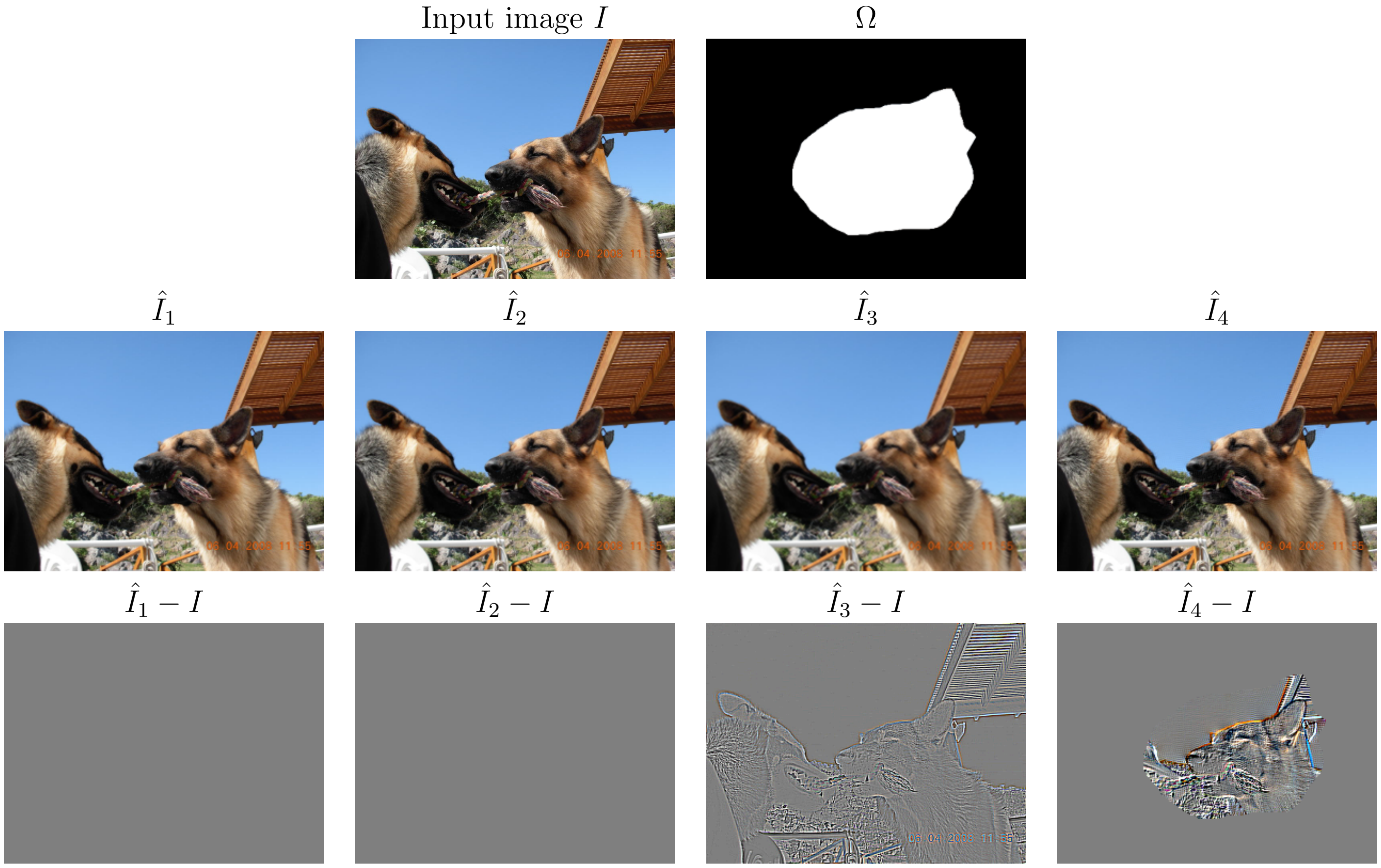
To understand these results, we will introduce the notation $\mathcal{D}^+_{x/y}$ to refer to the differential operator that computes x/y-partial derivative using forward finite differences, analogously we will refer to $\mathcal{D}^-_{x/y}$ and $\mathcal{D}^c_{x/y}$ to the operators that compute the derivative by backward and centered differences respectively, and $\mathcal{D}^F_{x/y}$ when the partial derivatives are computed in the Fourier domain.

Recalling the above definition (3) of the Laplacian operator $\mathcal{L}z = -4Z_{x,y}+Z_{x+1,y}+Z_{x-1,y}+Z_{x,y+1}+Z_{x,y-1}$, it is easy to see that $\mathcal{L} = \mathcal{D}^+_x\mathcal{D}^-_x + \mathcal{D}^+_y\mathcal{D}^-_y$. Hence if we compute the gradient field of an input image as $\mathbf{v} = \nabla I = (\mathcal{D}^-_x I, \mathcal{D}^-_y I)$ we will recover exactly the same image by solving the Poisson equation with the proposed finite differences scheme i.e. $\mathcal{L}f = \mathcal{D}^+_x I_x + \mathcal{D}^+_y I_y = (\mathcal{D}^+_x\mathcal{D}^-_x + \mathcal{D}^+_y\mathcal{D}^-_y)I = \mathcal{L}I = \mathcal{L}f \to \mathcal{L}(I - f) = 0$, and hence, considering Lemma 1, I is numerically identical to f (up to a global constant).

Lemma 1. *If two vectors x_1 and x_2 verify the equation: $\mathcal{L}(x_1 - x_2) = 0$, then $x_1 = x_2 + c$ for some scalar constant c.*
To prove this Lemma, we will first prove that a constant vector $\vec{c} = [c, \dots, c]^T$ is an eigenvector of the laplacian matrix $\mathcal{L}$ and the eigenvalue associated is 0. To that end, recall the definition of the matrix $\mathcal{L}$ illustrated in Figure 8, naming l_{ij} the elements of the matrix $\mathcal{L}$, we know that all the rows of the matrix sum zero, hence the i^{th} element of the vector $\mathcal{L}\vec{c} = \sum_j l_{ij}c_j = c\sum_j l_{ij} = 0$ for all i, and therefore a constant vector is an eigenvector of $\mathcal{L}$ with eigenvalue equal to zero. Secondly, as $\mathcal{L}$ is a diagonal dominant block irreducible matrix, the kernel space is a one dimensional space [20]. Hence if $\mathcal{L}(x_1 - x_2) = 0$, $(x_1 - x_2)$ must be a constant vector, i.e. x_1 and x_2 may differ only by a global constant.

On the other hand, if the input gradient field is computed by any other numerical scheme, we will recover a solution very similar but strictly different to the original image; e.g. if $\mathbf{v} = \nabla I = (\mathcal{D}^c_x I, \mathcal{D}^c_y I)$ we have $\mathcal{L}f = \mathcal{D}^+_x I_x + \mathcal{D}^+_y I_y = (\mathcal{D}^+_x\mathcal{D}^c_x + \mathcal{D}^+_y\mathcal{D}^c_y)I \to f \ \approx I$. Analogously, if we use the Fourier method to solve the Poisson equation, we may compute the input gradient field in the Fourier domain in order to recover exactly the same numerical values of the original image. In general, if the guide field is computed with a scheme $\mathcal{D}$ and we denote by $\mathcal{D}^*$ its adjoint operator (transposed conjugated), then the requirement is that the Laplacian is computed as $\mathcal{L} = \mathcal{D}^*_x\mathcal{D}_x + \mathcal{D}^*_y\mathcal{D}_y$.

In practice, if the objective is to integrate a given gradient field (and we do not have information about how it was computed) we can use any of the presented methods to solve the Poisson equation. As shown in Figure 11 all the methods are robust and achieve satisfactory results in any of the cases.



finite-diff gradients, finite-diff solution

Fourier gradients, Fourier solution

finite-diff gradients, Fourier solution

Fourier gradients, finite-diff solution

Figure 11: Test of the identity property. Solutions of the Poisson equation with different methods for computing the gradients, and different methods for solving the problem. Notice that when the gradients and the solution model are consistent to each other, the initial image is recovered exactly.

Relation with the original formulation of Pérez et al. The formulation presented by Pérez et al. [24] slightly differs from the one presented in Section 3.1; while Pérez et al. discretize the energy, we compute the Euler-Lagrange equation in the continuous domain and discretize the resulting partial differential equation. In addition, in the original formulation an oriented gradient map is defined for each pixel p considering its neighbors N_p. When N_p is the set of four-connected neighbors the two formulations are equivalent, since Equation (7) in [24] can be rewritten to match Equation (10) in Section 3.1.

4 Experiments and Examples

Seamless Cloning

Seamless cloning is one of the most impressive applications of the Poisson equation. The inputs are: a source image I_1 (or a set of images) from which we want to clone some regions, a destination image I_2, a selected region Ω, and finally the location of Ω on the destination image I_2.

Then we just need to define the guidance vector field $\mathbf{v}$ in order to proceed with the solution of

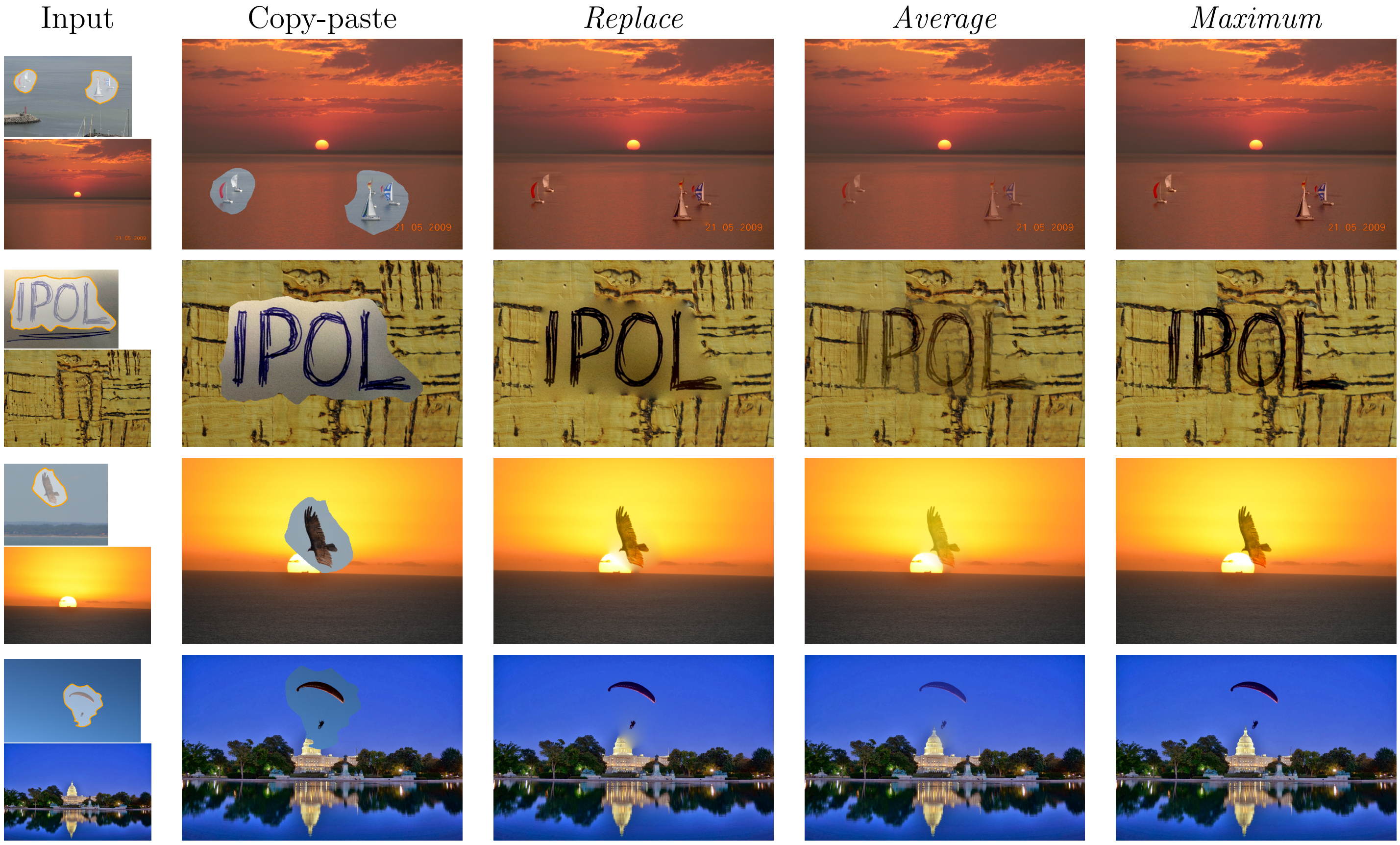


Figure 12: The first column shows the input images and the selected domain, the second column shows the result of copy-pasting the selected region over the destination image. The third column shows the result of replacing the gradients ($\mathbf{v} = \nabla I_1$), the fourth column the result of averaging source and destination gradients i.e. $\mathbf{v} = 1/2(\nabla I_1 + \nabla I_2)$, and finally the last column corresponds to keeping the maximum gradient between the source and destination images. These results were obtained using the *finite difference numerical method*.

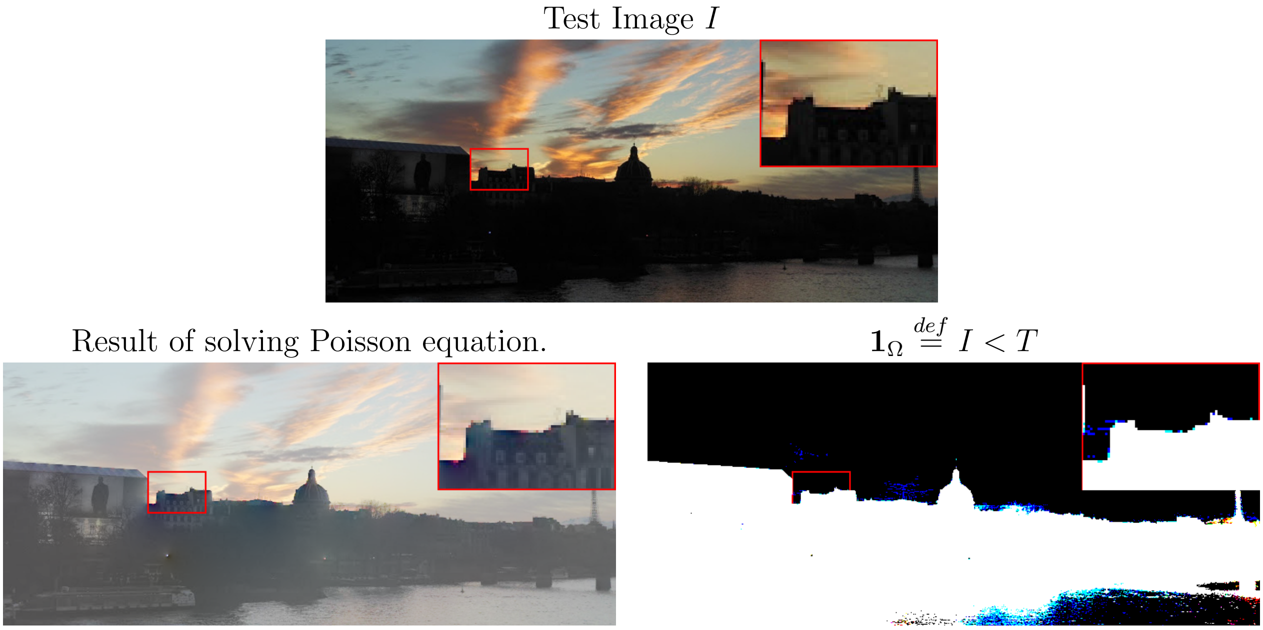
Equation (2). For example we can define the guidance vector field as

$$\begin{aligned} \mathbf{v} &= \nabla I_1 && \text{Replace} \\ \mathbf{v} &= \frac{1}{2}(\nabla I_1 + \nabla I_2) && \text{Average} \\ \mathbf{v} &= \begin{cases} \nabla I_1 \text{ if } |\nabla I_1| \geq |\nabla I_2| \\ \nabla I_2 \text{ otherwise} \end{cases} && \text{Maximum.} \end{aligned} \tag{19}$$

Figure 12 illustrates the differences between these alternative definitions of $\mathbf{v}$ inside the domain Ω.

Contrast Enhancement

Another possible application of the Poisson equation is to enhance the contrast of an image through a selective gradient amplification as proposed in [22]. In this case the input is a single image I, and we will define the region of interest Ω as: $\Omega = \{x \in \mathcal{R} \ / \ I(x) < T\}$ where T is a threshold whose value is considered an input parameter. The guidance vector field $\mathbf{v}$ will be defined as: $\mathbf{v} = \alpha \nabla I$, where α is a constant coefficient that determines the amount of amplification (applied inside the region Ω). Figure 13 shows a test input image (upper image), the set of underexposed regions that



Test Image I

Result of solving Poisson equation.

$\mathbf{1}_\Omega \overset{def}{=} I < T$

Figure 13: Sample test image and the domain Ω defined as $\Omega = \{x \in \mathcal{R} \mathrel{/} I(x) < T\}$. In this example the value of the threshold was set to $T = 60$. The bottom left image corresponds to the indicator function of the set Ω.

define the set Ω as explained previously (bottom right image), and the result of solving Poisson equation (bottom left image). The guidance vector field $\mathbf{v}$ is obtained through the amplification of the original gradient field on the underexposed pixels of the input image. In this experiment, finite difference method with Dirichlet boundary conditions was used to solve the Poisson equation. As can be seen in this example, the contrast in the dark regions of the image is increased as it is expected. However, this comes at the expense of contracting the dynamic range on the bright portions of the image. Furthermore, the abrupt modification of the original gradient field leads to visual artifacts (e.g. see the areas around the edges on the result shown in Fig. 13), which are obtained because the modified field has a non-irrotational component as explained in Sec. 1.1. In addition, these artifacts also produce a reduction of the effective dynamic range of the resulting image. To mitigate the previous problem at least two approaches prove to be effective. Firstly, a smooth amplification of the gradient field can be performed, i.e. we keep the original gradient field in those pixels that are close to $\partial\Omega$ and we gradually amplify the gradient field as we move to the interior points of the set Ω. The smooth amplification allows to reduce the irrotational component of the guidance vector field and also prevents the amplification of large gradients that may exist close to an edge of the image. Secondly, as the solution of the Poisson equation will have a different dynamic range (compared to the original input image), a normalization strategy must be chosen in order to remap the resulting image to the display range (typically in [0 255]). To that end, the simplest color balance [17, 21] proved to be a good normalization method as it can handle a few outliers that may be introduced due to the residual irrotational components of the guidance field. The simplest color balance algorithm simply stretches as much as it can, the values of the three channels Red, Green, Blue (R, G, B), so that they occupy the maximal possible range by applying an affine transform $ax + b$ to each channel. Since artifacts may produce a few aberrant pixels, the simples color balance method saturates a small percentage of the pixels with the highest values to 255 and a small percentage of the pixels with the lowest values to 0, before applying the affine transform.

The smooth amplification of the gradient field mentioned above is performed in two steps. First we produce a smooth version of the indicator function $\mathbf{1}_\Omega$ by convolving it with a Gaussian kernel.

Then a sigmoid function $s(x)$ is applied to it, mapping the interval [0.5 1] to the values $[1\ \alpha]$ in a smooth way. The result of these two steps is a smooth amplification map $\boldsymbol{\alpha}(x,y) = s(\mathbf{1}_\Omega * K_\sigma)$ where $*$ denotes the two dimensional convolution and K_σ a symmetric Gaussian kernel with standard deviation σ. In the present experiments σ was set to three pixels, the sigmoid function was defined according to eq. (20). Finally, the percentage of pixels saturated by the simplest color balance method was set to 1%.

$$s(x) = (\alpha - 1)\frac{1}{1 + e^{-20x+15}} + 1. \tag{20}$$

Figure 14 compares the result obtained by: (a) an homogeneous amplification ($\mathbf{v} = \alpha \nabla I$), and (b) an smooth amplification ($\mathbf{v} = \boldsymbol{\alpha}(x,y)\nabla I$). In this first two experiments the images are normalized setting the minimal value to 0 and the maximum value to 255, in the last two experiment (c-d) presented the normalization was performed following the simples color balance algorithm. Figure 15 show some additional experiments following the smooth amplification procedure described above plus the simples color balance method.

Amplifying the gradient field of an image by a factor α at points where the intensity values are below a threshold T, is mathematically equivalent to directly remapping intensity levels of the image with the function $h(u)$ defined as

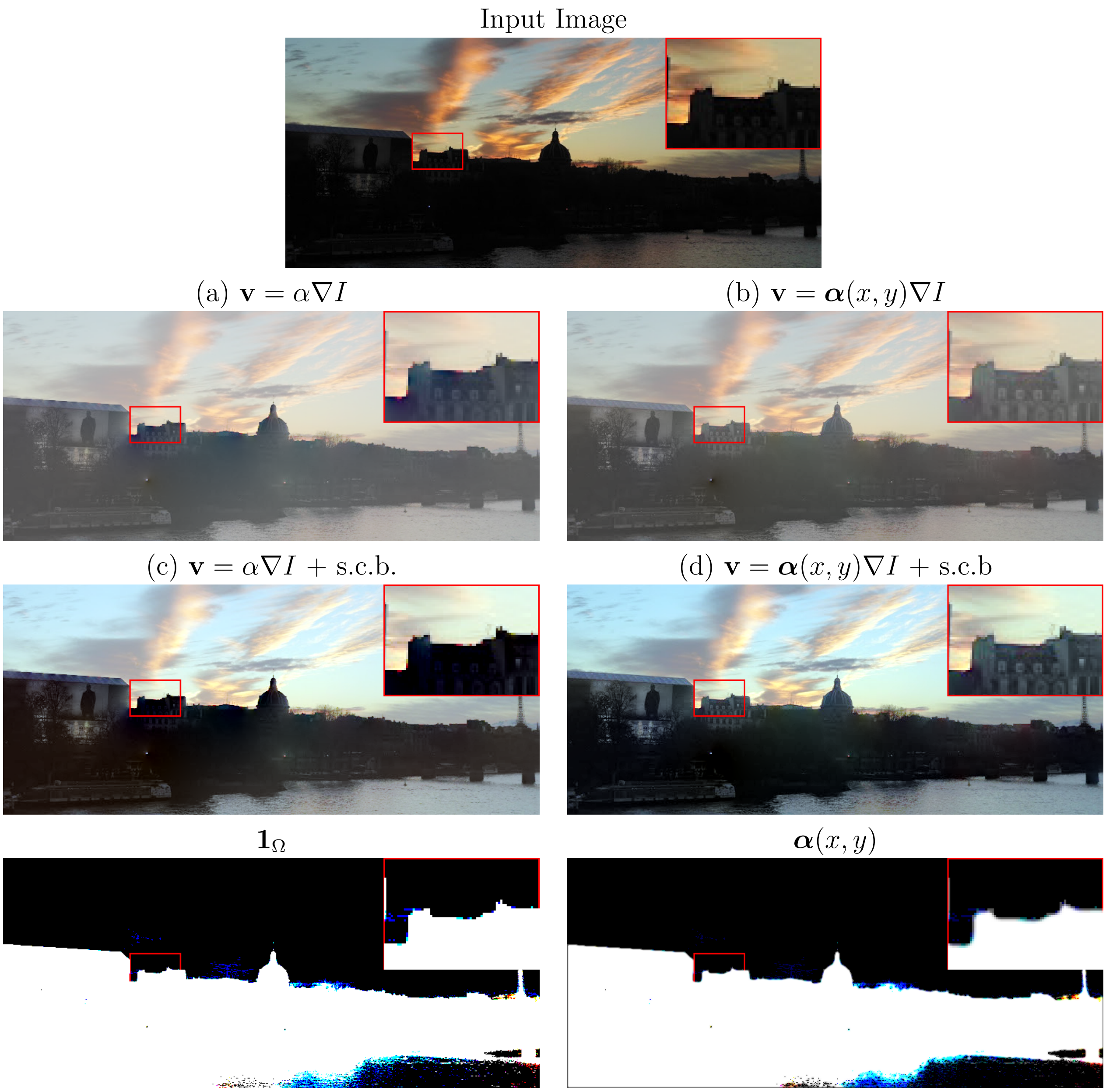
$$h(u) = \begin{cases} \alpha u & \text{if } u < T, \\ u + (\alpha - 1)T & \text{if } u \geq T. \end{cases} \tag{21}$$

The equivalence can be easy proved in the continuous domain, for instance, let us define the remapped image $g(\mathbf{x}) = h(f(\mathbf{x}))$ where f denotes the input image. Then we have that $\nabla g = h'(f(\mathbf{x}))\nabla f(\mathbf{x})$ so the gradient field $\nabla g(\mathbf{x}) = \alpha \nabla f(\mathbf{x})$ if $f(\mathbf{x}) < T$ and $\nabla g(\mathbf{x}) = \nabla f(\mathbf{x})$ if $f(\mathbf{x}) > T$. For those regions in which $f(\mathbf{x}) = T$ the derivative of the mapping function h is not defined, however this does not represent a problem as long as the set $\{\mathbf{x} \in \mathcal{R}\ /\ f(\mathbf{x}) = T\}$ is of measure zero. Figure 16 shows the result obtained by integrating the set of amplified gradient (result of Poisson equation with guidance vector $\mathbf{v} = \boldsymbol{\alpha}(x,y)\nabla I$) and the result of directly remapping the intensity values of the image according to equation (21). In order to display and compare the output images, both results were rescaled to the interval [0 255] using the simplest color balance algorithm. While in the continuous domain both approaches are equivalent, in the discrete domain both methods may produce different results. In particular, as the gradient field is computed using finite differences, the gradient at the boundary of the domain may be associated to an interior pixel of the set Ω or an exterior one (depending of the position of the pixel relative to $\partial\Omega$). In practice, the direct mapping of the intensity values achieves very similar results to the ones obtained trough the gradient amplification, while being a much simpler and faster approach.

Texture Flattening (Retinex)

Similar to the previous method, one can remove those gradients on the image whose modulus is below a given threshold. This produces texture flattening and can be used, for example, to remove shadows and smooth illumination transitions (e.g. to simulate Retinex theory). The idea is again to consider an input image I and to define the region of interest as $\Omega = \{x \in \mathcal{R}\ /\ |\nabla I| < T\}$. Then, the guidance vector field is set as $\mathbf{v} = 0$ inside Ω. An example is shown in Figure 17.

As in the previous method, we complete the processing step applying a normalization to the output. The normalization is performed by setting the mean and variance of the solution of Poisson equation to match the mean and variance of the original image, this is the same normalization performed in [18], and allows to recover the mean value of the solution (which is undefined when Fourier solver is used) and give a result that can be easily compared with the input image.



Input Image

(a) $\mathbf{v} = \alpha \nabla I$

(b) $\mathbf{v} = \boldsymbol{\alpha}(x, y) \nabla I$

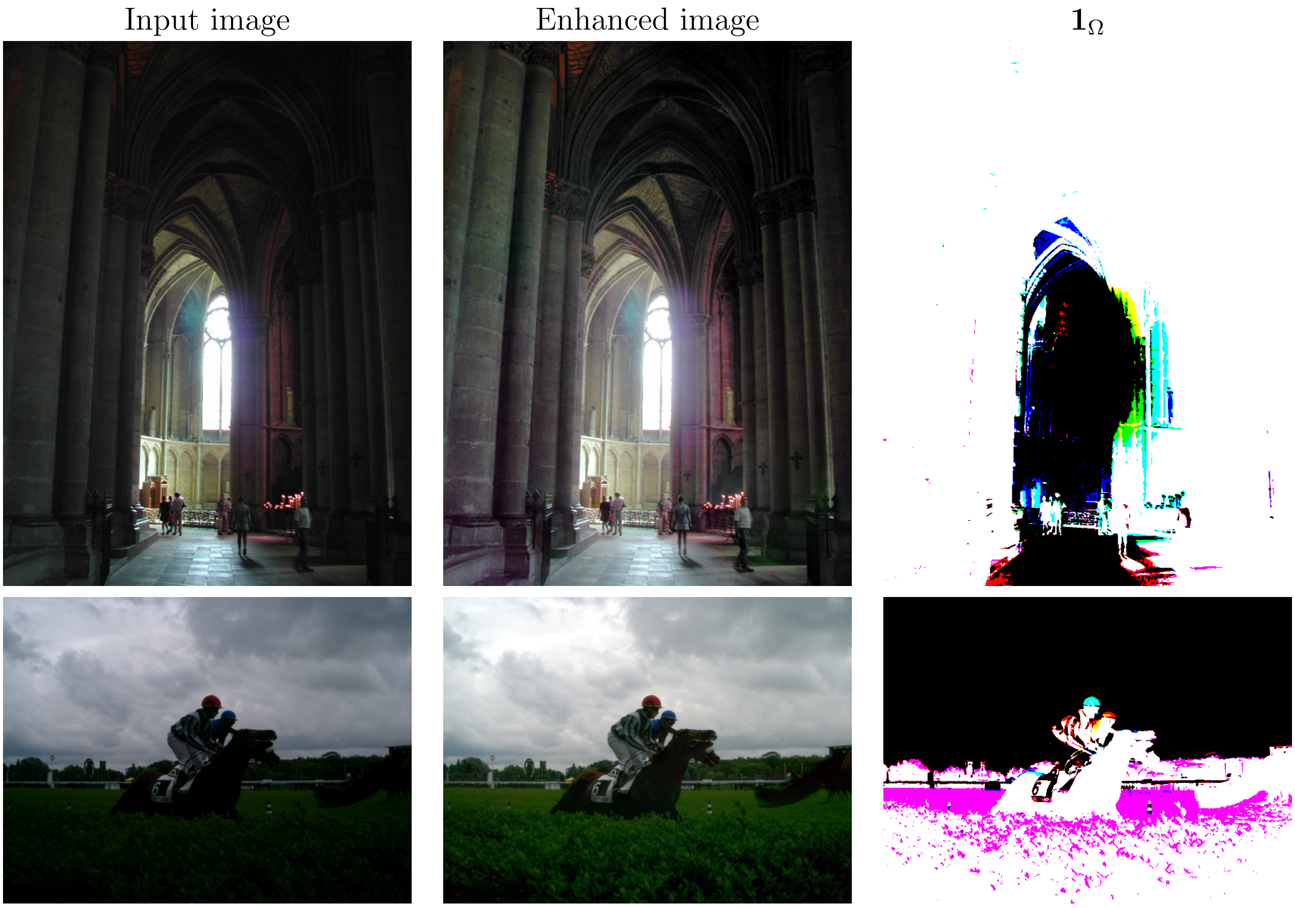
(c) $\mathbf{v} = \alpha \nabla I$ + s.c.b.

(d) $\mathbf{v} = \boldsymbol{\alpha}(x, y) \nabla I$ + s.c.b

$\mathbf{1}_{\Omega}$

$\boldsymbol{\alpha}(x, y)$

Figure 14: Results with homogeneous and smooth gradient amplification, and with the simples color balance (s.c.b.) normalization method.

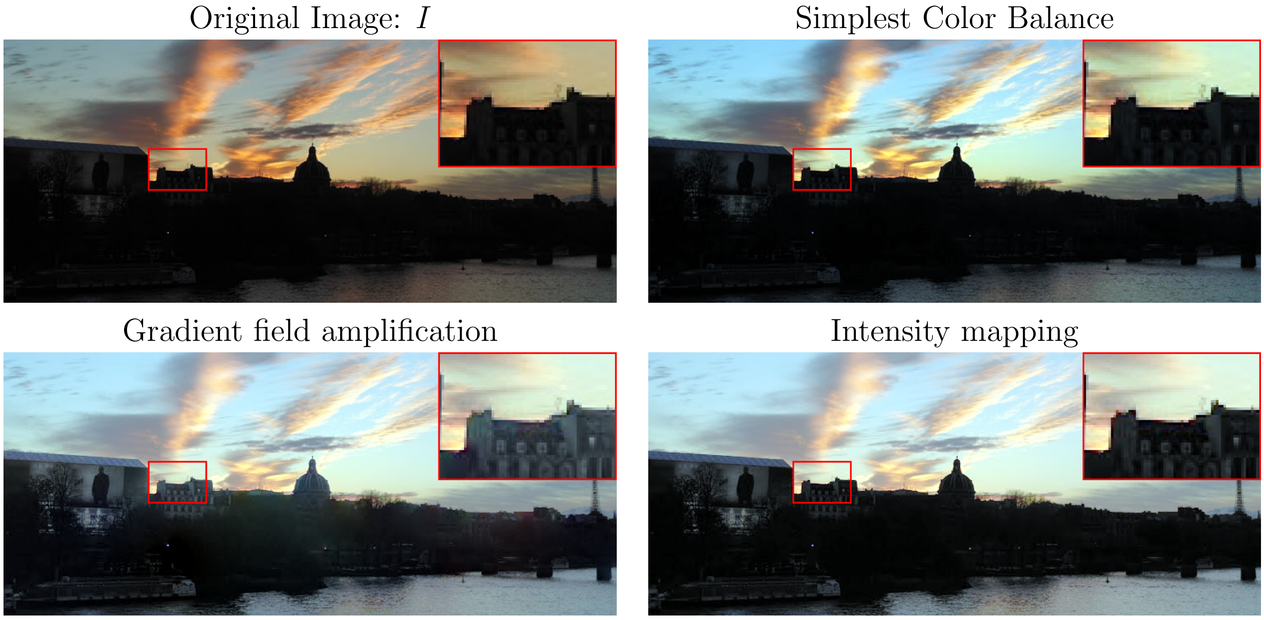


Input image

Enhanced image

$\mathbf{1}_\Omega$

Figure 15: Contrast enhancement results obtained using the *Finite differences numerical method*. The region Ω is computed for each color channel independently. In the third column the images display the red, green and blue components of Ω.



Original Image: I

Simplest Color Balance

Gradient field amplification

Intensity mapping

Figure 16: Upper left: original image, upper right: original image renormalized with the simplest color balance method. Bottom left: result of Poisson method (smooth amplification and s.c.b. normalization), bottom right: result of remapping the intensity values of the input image according to equation (21) and then normalizing the result using s.c.b. method.

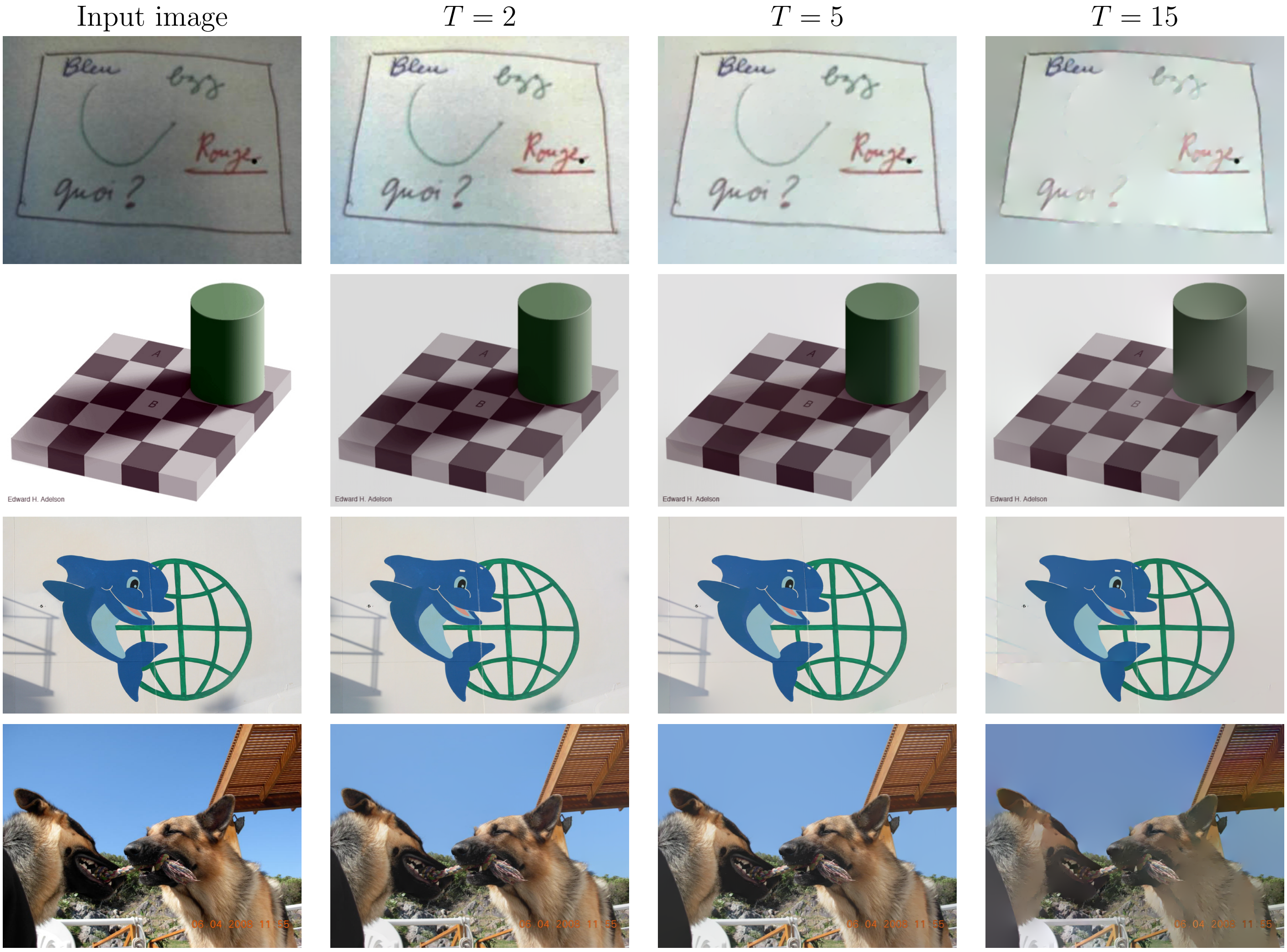


Figure 17: Results obtained by setting to zero those gradients whose modulus was below the threshold T. These results were obtained using the *Fourier numerical method*.

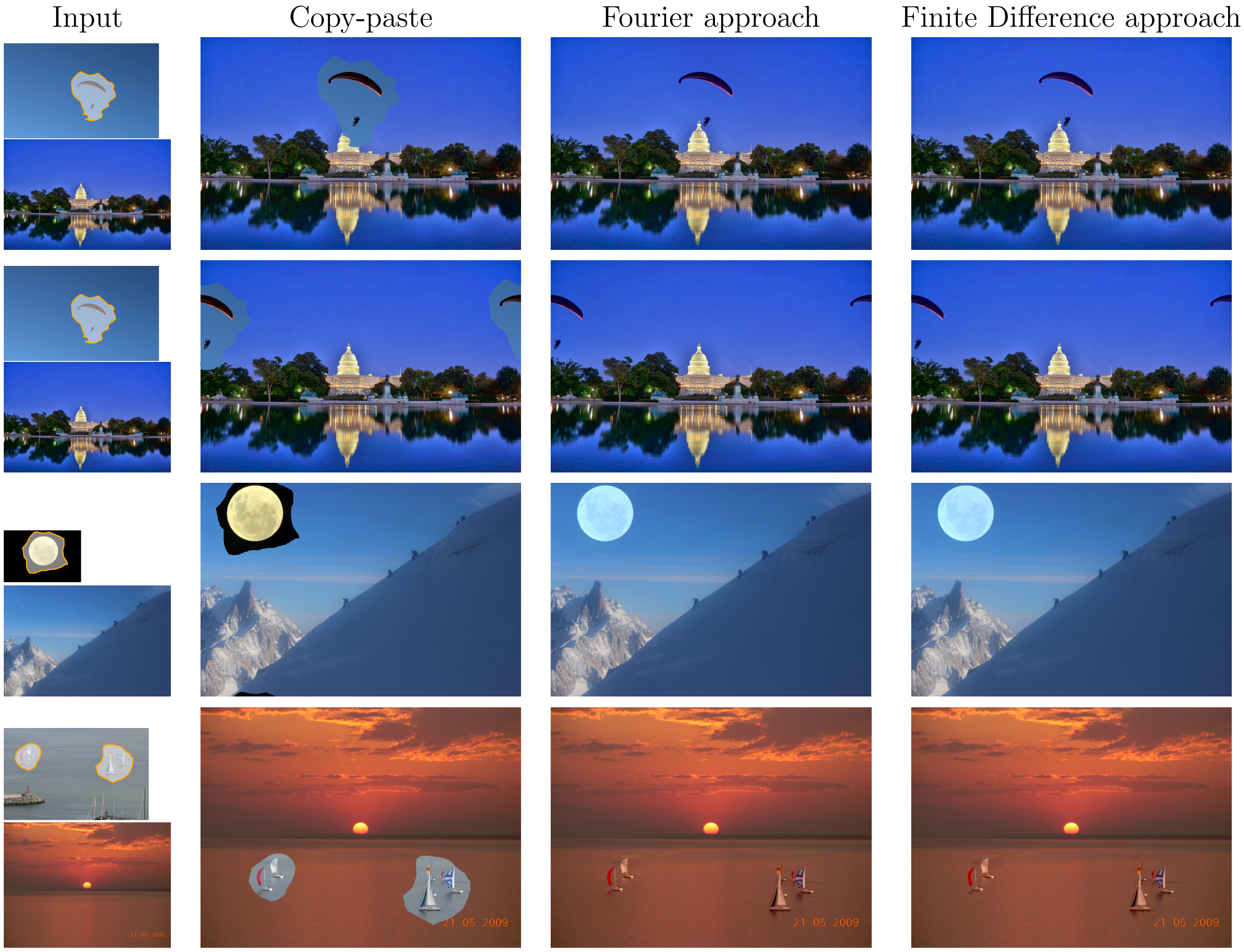


Figure 18: Examples of seamless cloning with the described numerical methods.

Comparison of Fourier and Finite Difference Approaches

Figure 18 shows the results obtained using both the Fourier and the finite difference approaches previously described. In the first experiment different seamless cloning situations were tested, e.g. the first row shows the results for an example in which Ω is a convex set that does not intersect the edges of the image, the second and third rows show examples in which Ω reaches the edge of the image, and the last row shows an example where Ω is composed by two separate regions included in the interior of the image. We empirically found that for our Matlab implementations the Fourier approach was approximately twice as fast as the finite difference approach. Moreover for a color image of approximately $900 \times 800 \times 3$ pixels the Fourier approach takes on average one second (Matlab implementation running on a machine with an Intel Core i5 @ 2.27GHz processor and 8GB of RAM).

In a second set of experiments (shown in Figure 19) we tested both numerical methods for contrast enhancement and texture flattening. In these examples the set Ω was obtained by thresholding the input image (or its gradient) as explained above. As we can see in the zoomed details on the third row of Figure 19, when $\partial\Omega$ intersects regions of the image with high variations, annoying artifacts can appear during the integration step. The Fourier approach produces ringing artifacts while the finite difference approach produces color artifacts in the neighborhood of $\partial\Omega$. As was explained in Section 1.1 these defects occur because the modified gradient map we are integrating is a non-integrable field (i.e. it is not irrotational). As a general rule, we found that the finite difference

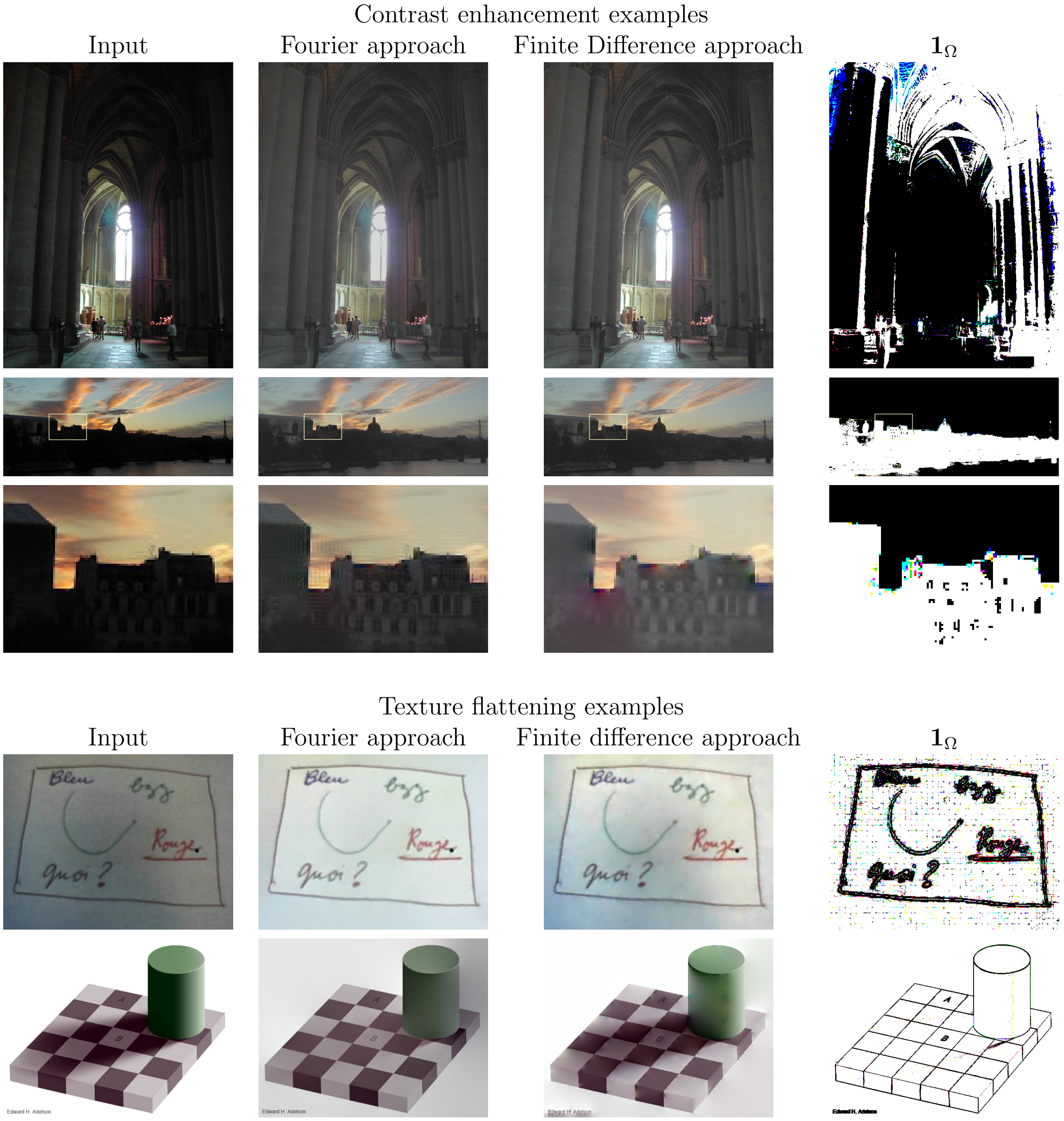


Figure 19: Examples of contrast enhancement and texture flattening with the described numerical methods.

approach works very well in most seamless cloning examples and its solutions is exact in the sense that the destination image is not modified outside Ω. On the other hand for contrast enhancement and texture flattening we found that the Fourier approach gives visually more satisfactory results.

5 Conclusion

In the present work the Poisson image editing method was described in detail and an up-to-date review of the latest contributions on the subject was provided. In addition the integration problem was discussed and analyzed, both from a theoretical and numerical point of view. Two different numerical implementations were described, the first one (finite difference approach) uses discrete versions of differential operators to convert the problem into a linear system of equations; the second one (Fourier approach) is based on Fourier transform properties and solves the Poisson equation over an extended domain. Both Octave/Matlab implementation and an on-line demo to test them are publicly available at the IPOL web page of this article[6].

Acknowledgment

The authors want to thank Jean-Michel Morel for fruitful discussions, and Miguel Colom and Martin Rais for their technical support and assistance with the on-line demo of the article. Thanks to Michael Tao, Jian Sun and Adrian Ilie for allowing us to use images shown in Figures 4 and 3. Work partly founded by the European Research Council (advanced grant Twelve Labours n. 246961) and the Office of Naval research (ONR grant N00014-14-1-0023).

Appendix A: Euler-Lagrange Equation

The solution $\hat{f}$ of the minimization problem,

$$\hat{f} = \underset{\substack{f \,\in\, \mathscr{C}^2(\mathcal{R}) \\ \text{st. } f|_{\mathcal{R}\setminus\Omega} = f^*|_{\mathcal{R}\setminus\Omega}}}{\arg\min} \left\{ \int_\Omega \|\nabla f - \mathbf{v}\|^2 \, dx \right\}, \tag{22}$$

is the minimum of the functional,

$$F[f] = \left\{ \int_\Omega \|\nabla f - \mathbf{v}\|^2 \, dx \right\}, \tag{23}$$

subject to the condition $f|_{\partial\Omega} = f^*|_{\partial\Omega}$. If we define the set $S_0 = \{g \in \mathscr{C}^2(\mathcal{R}) \ / \ g|_{\partial\Omega} = 0\}$, it is easy to see that the solution $\hat{f}$ must verify the condition

$$\lim_{\epsilon\to 0} \frac{F[f+\epsilon g] - F[f]}{\epsilon} = 0 \text{ , for all } g \,\in\, S_0. \tag{24}$$

Since the perturbation functions g are in S_0, all the functions defined as $f + \epsilon g$ also verify the boundary conditions (if f satisfies it), i.e. $(f + \epsilon g)|_{\partial\Omega} = f|_{\partial\Omega} + \epsilon g|_{\partial\Omega} = f^*|_{\partial\Omega}$.

[6] https://doi.org/10.5201/ipol.2016.163

Equation (24) can be written as

$$\begin{aligned} 0 &= \lim_{\epsilon \to 0} \frac{1}{\epsilon} \left\{ \int_\Omega \left(\|\nabla(f+\epsilon g) - \mathbf{v}\|^2 - \|\nabla f - \mathbf{v}\|^2 \right) dx \right\} \\ &= \lim_{\epsilon \to 0} \frac{1}{\epsilon} \left\{ \int_\Omega \left(2\epsilon \nabla g \cdot (\nabla f - \mathbf{v}) + \epsilon^{\mathbf{2}} \|\nabla \mathbf{g}\|^{\mathbf{2}} \right) dx \right\} \\ &= \int_\Omega \left(2\nabla g \cdot (\nabla f - \mathbf{v}) \right) dx. \end{aligned} \tag{25}$$

Using Green's identity [6, pag. 62][7] for each spatial coordinate, the previous equation can be transformed into

$$0 = \int_\Omega \left(2\nabla g \cdot (\nabla f - \mathbf{v}) \right) dx = -2 \int_\Omega g \left(\Delta f - \operatorname{div}(\mathbf{v}) \right) dx + \int_{\partial\Omega} g \left(\nabla f - \mathbf{v} \right) \cdot \nu \, ds. \tag{27}$$

The second term of the right expression is equal to zero as $g \in S_0$, and hence in order to verify

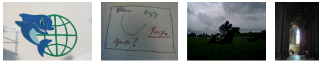
$$\int_\Omega g \left(\Delta f - \operatorname{div}(\mathbf{v}) \right) dx = 0 \text{ , for all } g \in S_0, \tag{28}$$

the objective function must verify the Poisson equation

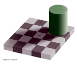
$$\Delta f(x) = \operatorname{div}(\mathbf{v}(x)) \text{ , for all } x \in \Omega \text{ , with } f|_{\partial\Omega} = f^*|_{\partial\Omega}. \tag{29}$$

Image Credits

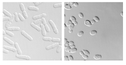
The authors CC-BY



Ipol.im[8]



Edward H. Adelson[9]



YRC Public Image Repository[10]

References

[1] A. Agrawal, *Scene Analysis under Variable Illumination using Gradient Domain Methods*, PhD thesis, University of Maryland, 2006.

[2] A. Agrawal, R. Chellappa, and R. Raskar, *An algebraic approach to surface reconstruction from gradient fields*, Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1, 1 (2005). http://dx.doi.org/10.1109/ICCV.2005.31.

[7] *Let u, $v \in \mathscr{C}^1(\bar{\Omega})$, and ν the outward unit normal of $\partial\Omega$. Then*

$$\int_\Omega u_{x_i} v \, dx = -\int_\Omega u v_{x_i} \, dx + \int_{\partial\Omega} u v \nu^i \, ds. \quad (i = 1, 2). \tag{26}$$

[8] http://www.ipol.im/pub/art/2011/lmps_rpe/ and http://www.ipol.im/pub/art/2011/llmps-scb/

[9] http://web.mit.edu/persci/people/adelson/checkershadow_illusion.html

[10] http://images.yeastrc.org

[3] A. Agrawal, R. Raskar, and R. Chellappa, *What is the Range of Surface Reconstructions from a Gradient Field?*, European Conference on Computer Vision, (2006), pp. 578–591. http://dx.doi.org/10.1007/11744023_45.

[4] P. Arias, G. Facciolo, V. Caselles, and G. Sapiro, *A variational framework for exemplar-based image inpainting*, International Journal of Computer Vision, 93 (2011), pp. 319–347. http://dx.doi.org/10.1007/s11263-010-0418-7.

[5] M. R. Arnison, K. G. Larkin, C. J. R. Sheppard, N. I. Smith, and C. J. Cogswell, *Linear phase imaging using differential interference contrast microscopy*, Journal of Microscopy, 214 (2004), pp. 7–12. http://dx.doi.org/10.1111/j.0022-2720.2004.01293.x.

[6] G. Aubert and P. Kornprobst, *Mathematical Problems in Image Processing: Partial Differential Equations and the Calculus of Variations (second edition)*, vol. 147 of Applied Mathematical Sciences, Springer-Verlag, 2006.

[7] P. Bhat, B. Curless, M. Cohen, and C. L. Zitnick, *Fourier analysis of the 2D screened poisson equation for gradient domain problems*, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 5303 LNCS (2008), pp. 114–128. http://dx.doi.org/10.1007/978-3-540-88688-4_9.

[8] P. Bhat, C. L. Zitnick, M. Cohen, and B. Curless, *GradientShop : A Perceptually-Motivated Optimization-Framework for Image and Video Processing*, Technical report, University of Washington, (2008).

[9] ——, *GradientShop: A Gradient-Domain Optimization Framework for Image and Video Filtering*, ACM Transactions on Graphics, 29 (2010), pp. 1–14. http://dx.doi.org/10.1145/1731047.1731048.

[10] M. Bonomi, R. Pellarin, S. J. Kim, D. Russel, B. A. Sundin, M. Riffle, D. Jaschob, R. Ramsden, T. N. Davis, E. Muller, et al., *Determining protein complex structures based on a bayesian model of in vivo Förster resonance energy transfer (FRET) data*, Molecular & Cellular Proteomics, 13 (2014), pp. 2812–2823. http://dx.doi.org/10.1074/mcp.M114.040824.

[11] J. M. Di Martino, A. Fernández, and J. Ferrari, *One-shot 3D gradient field scanning*, Optics and Lasers in Engineering, 72 (2015), pp. 26–38. http://dx.doi.org/10.1016/j.optlaseng.2015.04.001.

[12] Z. Du, A. Robles-Kelly, and F. Lu, *Robust surface reconstruction from gradient field using the L1 norm*, in Digital Image Computing Techniques and Applications, 9th Biennial Conference of the Australian Pattern Recognition Society on, IEEE, 2007, pp. 203–209. http://dx.doi.org/10.1109/DICTA.2007.4426797.

[13] R.n Fattal, D. Lischinski, and M. Werman, *Gradient domain high dynamic range compression*, ACM Transactions on Graphics, 21 (2002), pp. 249–256. http://dx.doi.org/10.1145/566654.566573.

[14] D. K. Hamilton and C. J. R. Sheppard, *Differential phase contrast in scanning optical microscopy*, Journal of Microscopy, 133 (1984), pp. 27–39. http://dx.doi.org/10.1111/j.1365-2818.1984.tb00460.x.

[15] B. K. P. Horn and M. J. Brooks, eds., *Shape from Shading*, MIT Press, Cambridge, MA, USA, 1989. ISBN 0262081830.

[16] A. Levin, A. Zomet, S. Peleg, and Y. Weiss, *Seamless Image Stitching in the Gradient Domain*, European Conference on Computer Vision, 4 (2004), pp. 377–389. http://dx.doi.org/10.1007/978-3-540-24673-2_31.

[17] N. Limare, J.L. Lisani, J-M. Morel, A.B. Petro, and C. Sbert, *Simplest Color Balance*, Image Processing On Line, 1 (2011). http://dx.doi.org/10.5201/ipol.2011.llmps-scb.

[18] N. Limare, A. B. Petro, C. Sbert, and J-M. Morel, *Retinex Poisson Equation: a Model for Color Perception*, Image Processing On Line, 1 (2011), pp. 1–12. http://dx.doi.org/10.5201/ipol.2011.lmps_rpe.

[19] J. McCann and N. S. Pollard, *Real-time gradient-domain painting*, ACM Transactions on Graphics, 27 (2008), p. 1. http://dx.doi.org/10.1145/1360612.1360692.

[20] R. Merris, *Laplacian matrices of graphs: a survey*, Linear algebra and its applications, 197 (1994), pp. 143–176.

[21] J-M. Morel, Petro A. B., and C. Sbert, *Screened Poisson Equation for Image Contrast Enhancement*, Image Processing On Line, 4 (2014), pp. 16–29. http://dx.doi.org/10.5201/ipol.2014.84.

[22] J-M. Morel, A. B. Petro, and C. Sbert, *Fourier implementation of Poisson image editing*, Pattern Recognition Letters, 33 (2012), pp. 342–348. http://dx.doi.org/10.1016/j.patrec.2011.10.010.

[23] G. Normanski, *Microinterferometre differential a ondes polarisees*, Journal of Physics Radium, Paris, 16 (1955).

[24] P. Pérez, M. Gangnet, and A. Blake, *Poisson image editing*, ACM Transactions on Graphics, 22 (2003), p. 313. http://dx.doi.org/10.1145/882262.882269.

[25] R. Raskar, A. Ilie, and J. Yu, *Image Fusion for Context Enhancement and Video Surrealism*, in International Symposium on Non-Photorealistic Animation and Rendering, 2004, pp. 85–152. http://dx.doi.org/10.1145/987657.987671.

[26] D. Reddy, A. Agrawal, and R. Chellappa, *Enforcing integrability by error correction using l1-minimization*, 2009 IEEE Conference on Computer Vision and Pattern Recognition, (2009). http://dx.doi.org/10.1109/CVPR.2009.5206603.

[27] R. Sadek, G. Facciolo, P. Arias, and V. Caselles, *A Variational Model for Gradient-Based Video Editing*, International Journal of Computer Vision, 103 (2013), pp. 127–162. http://dx.doi.org/10.1007/s11263-012-0597-5.

[28] J. Shen, X. Jin, C. Zhou, and C. C. L. Wang, *Gradient Based Image Completion by Solving the Poisson Equation*, Computers & Graphics, 31 (2007), pp. 119–126. http://dx.doi.org/10.1016/j.cag.2006.10.004.

[29] J. Sun, J. Jia, C-K. Tang, and H-Y. Shum, *Poisson matting*, ACM Transactions on Graphics, 23 (2004), p. 315. http://dx.doi.org/10.1145/1015706.1015721.

[30] M. W. Tao, M. K. Johnson, and S. Paris, *Error-tolerant image compositing*, International Journal of Computer Vision, 103 (2013), pp. 178–189. http://dx.doi.org/10.1007/s11263-012-0579-7.

[31] J. Tumblin, A Agrawal, and R. Raskar, *Why I Want a Gradient Camera*, IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), 1 (2005), pp. 103–110. http://dx.doi.org/10.1109/CVPR.2005.374.

[32] J. Weickert, K. Hagenburg, M. Breuss, and O. Vogel, *Linear osmosis models for visual computing*, in Energy Minimization Methods in Computer Vision and Pattern Recognition, Springer, 2013, pp. 26–39.

[33] R. J. Woodham, *Photometric method for determining surface orientation from multiple images*, Optical Engineering, 19 (1980), p. 191139. http://dx.doi.org/10.1117/12.7972479.