TOWARDS UNIFYING DIFFUSION AND EXEMPLAR-BASED INPAINTING

Emmanuel d'Angelo, Pierre Vandergheynst

Ecole Polytechnique Fédérale de Lausanne (EPFL)
Signal Processing Laboratory

ABSTRACT

A novel framework for image inpainting is proposed, relying on graph-based diffusion processes. Depending on the construction of the graph, both flow-based and exemplar-based inpainting methods can be implemented by the same equations, hence providing a unique framework for geometry and texture-based approaches to inpainting. Furthermore, the use of a variational framework allows to overcome the usual sensitivity of exemplar-based methods to the heuristic issues by providing an evolution criterion. The use of graphs also makes our framework more flexible than former non-local variational formulations, allowing for example to mix spatial and non-local constraints and to use a data term to provide smoother blending between the initial image and the result.

Index Terms— Inpainting, Non-local methods, Graphs

1. INTRODUCTION

Image inpainting, also known as image completion, is the problem of finding missing parts of an image using only the available content and some regularization constraints. The disoccluded areas should cause few, if none, visual artifacts, which makes inpainting a difficult image processing problem involving knowledge about image models and regularization techniques. While it has been a long time problem for painting restoration, it has gained in importance with the growth of the digital photography market, since it allows users to remove disturbing elements from their pictures or repair damages such as visible dust on the sensor of the camera or scratches in an old digitized photograph.

Traditional approaches to inpainting try to find a good continuation of the surroundings of the holes, effectively propagating lines inside. Hence, these techniques are also known as *geometry-driven* algorithms. Isophote propagation is obtained by solving partial differential equations such as the heat flow of the Navier-Stokes equation as in [1]. Since this tend to produce blurred estimates, edge-preserving techniques such as Total Variation minimization [2] or curvature motion [3] were used as an alternative. However, these models still lack a support for texture information and are consequently unable to reproduce it. Geometry-based methods can thus hardly be applied to large area inpainting without noticeable visual disturbance.

Since textures, and especially the structured ones like brick walls, are hard to model due to their high yet constrained variability, people working in the texture synthesis field applied successfully alternative exemplar-based techniques, which tackle the lack an explicit mathematical texture model. Starting with the work of Efros and Leung [4], exemplar-based techniques take as input a source image containing the desired texture. Then, random parts of the source are extracted and used to fill the larger target picture, with a special treatment to avoid visual discontinuities between neighbouring overlapping patches (see [5, 6] for instance). These techniques were quickly applied to texture inpainting [7]. They are however sensitive to the order in which gaps are filled, that can create disturbing subjective contours. Hence, the authors of [8] defined careful heuristics for choosing which pixels to process first, and managed to successfully inpaint large areas. This type of approach can even be extended to the case of video inpainting [9].

Few attempts were made however to conciliate geometry-driven and exemplar-based techniques in an unified framework. Simultaneous geometry and texture inpainting algorithms that were proposed until here, such as the work in [10], separate inpainting into two distinct steps dedicated to geometry diffusion and texture synthesis respectively. This approach requires first to decompose the image into a geometric part, or sketch, and a textural part, which is still an arduous task [11]. Specifically, the texture function spaces used contain only very regular oscillating patterns, which is seldom the case for real life textures such as the bricks aforementioned.

In this work, our main goal is to take advantage of the so-called *non-local* approach, successfully introduced in [12] for image denoising, to develop a general variational framework for image inpainting, hence embedding both geometry and texture based approaches in a single framework. While the work in [13] is also based on the non-local approach, the constraints it provides cannot be adapted to handle both geometric and textural inpainting in a single framework. The remaining of this paper is organized as follows: in section 2, we restate the inpainting task as a non-local variational problem and present a general framework defined for both local and non-local processing tasks. Section 3 describes the corresponding algorithm and its implementation. We present some experimental results in section 4, along with a discussion, before concluding in section 5.

2. NON-LOCAL VARIATIONAL INPAINTING

The recent trend in non-local methods is due to the success of the Non-Local Means denoising filter [12], although its authors trace back their inspiration to the work of Efros and Leung [4] and Yaroslavsky [14]. Their intuition is simple : similar-looking patches are more likely to represent the same phenomenon, and hence should be used when averaging pixels without considerations to their spatial distance to obtain an efficient denoising process that also respects image textures.

To turn the initial patch-based formulation of the non-locality into a variational problem, the authors of [15] proposed to use graph differential geometry. Since graphs can model any kind of inter-data relationship, depending only on the way connections are made, they are good candidates to derive both local and non-local regularization frameworks.

We first build a non-local graph of all the available image data

using an image self-similarity measure. Vertices originating inside the hole will exhibit different connections than others because of their visual dissimilarity, and hence will cause a non-local singularity. We apply an iteration of heat flow on this non-local graph to reduce this singularity. Then, patches are back-projected to the image domain to obtain a novel image where the information inside the hole is now closer to the remaining of the image. This procedure is iterated several times after updating the graph connectivity to diffuse the non-local information into the gaps.

2.1. Non-local inpainting formulation

Let I be an image defined on a domain $\Omega \subset \mathbb{R}^2$. $\Phi(x)$ and $\Phi(y)$ are square patches of area d centered in x and y respectively, and also denote the corresponding vectors obtained by stacking the values inside the patch in lexicographic order[1]. We want to infer missing information in a subdomain $\omega \subset \Omega$ in a plausible way. Note that we did not make any assumptions about the shape or topology of ω, which is of arbitrary shape and can contain holes or several disconnected parts.

There is a natural one-to-one correspondence between each patch and the corresponding image pixel. Hence, $\Phi : \Omega \to \mathbb{R}^d$ defines an embedding of the image data in $\mathbb{R}^d$. In this embedding, the distance between two points is obtained by the following patch similarity measure :

$$w(x,y) = w(y,x) = \exp\left(-\frac{\|\Phi(x) - \Phi(y)\|_\sigma^2}{h}\right), \qquad (1)$$

where $\|\cdot\|_\sigma^2$ is the sum of squared differences between the two patches filtered by a Gaussian neighbourhood function of standard deviation σ :

$$\|\Phi(x) - \Phi(y)\|_\sigma^2 = \left(\sum_{i=1}^{d} \left(\Phi_i(x) - \Phi_i(y)\right)^2\right) \star G_\sigma. \qquad (2)$$

The parameter h in Eq. (1) acts as a selectivity parameter. Large values of h will gather dissimilar looking patches, while smaller values will be more selective. This parameter thus provides a way to infer the missing data despite the presence of noise, which may not be desirable, since it leads to blurry estimates. The Gaussian G_σ is introduced to represent the finite spatial support of a patch. Finally, note that w does not take into account the positions of x and y in the image, and hence is non-local.

2.2. Graph-based regularization

A weighted graph $\mathcal{G} = (V, E, w)$ is made of a finite set of *vertices* $V = \{v_1, \dots v_N\}$ linked by *edges* taken in $E \subset V \times V$ with associated *weights* computed using a similarity function $w_\mathcal{G} : V \times V \to \mathbb{R}_+$. The vertices correspond to the actual data points. Two vertices u and v are connected by an edge $e = (u, v)$ if $w_\mathcal{G}(u, v) > 0$, and we write $u \sim v$. There is an immediate one-to-one mapping between image pixels and graph vertices, and the function $w(x, y)$ defined in Eq. (1) is positive and tends to 0 when patches are highly dissimilar, so we use it without modifications to compute the graph weights ($w = w_\mathcal{G}$). Since the property $w(u, v) = w(v, u)$ holds, $\mathcal{G}$ is an undirected graph.

We build a non-local graph $\mathcal{G}$ as described above to obtain the final version of the embedding $\Phi : V \to \mathbb{R}^d$ that we want to regularize. Φ is d-dimensional: there is one dimension per component of a patch. The vertices coming from ω disrupt the non-local smoothness of Φ measured using $\mathcal{G}$, since the corresponding image patches differ from the patches of $\Omega\backslash\omega$. The non-local inpainting formulation can be restated to enforce the smoothness of Φ, given the immutable data outside ω :

$$\min_{\Phi : V \to \mathbb{R}^d} \frac{1}{2} \sum_{i=1}^{d} \sum_{v \in V} |\nabla_w \Phi_i(v)|^2 + \frac{\lambda}{2} \sum_{i=1}^{d} \|\Phi_i - \Phi_i^0\|_2^2. \qquad (3)$$

The first term $|\nabla_w \Phi_i(v)|^2$ ensures that we are not introducing visual dissimilarities by controlling the smoothness of Φ. The data term can be used to enforce some proximity with the initial image, typically along the borders of the hole, but λ will be set to 0 inside the holes to allow the full replacement of their content.

Eq. (3) can be solved component-wise using gradient descent after defining differential calculus operators on a graph. Given a weighted graph $\mathcal{G}$, one can define the *directional derivative* of Φ_i at the vertex v in the direction given by the edge (u, v) by :

$$\partial_u \Phi_i(v) = \sqrt{w(u,v)}\left(\Phi_i(v) - \Phi_i(u)\right). \qquad (4)$$

The *weighted gradient operator* of Φ_i at a given vertex v is then defined as the vector of all the directional derivatives measured at v:

$$\nabla_w \Phi_i(v) = \left[\partial_u \Phi_i(v) : u \sim v\right]^T. \qquad (5)$$

The norm of the gradient defines the *local variation* of Φ_i at v, and measures the regularity of Φ_i with respect to the graph connections. It is computed in the usual way :

$$|\nabla_w \Phi_i(v)| = \sqrt{\sum_{u \sim v} (\Phi_i(u) - \Phi_i(v))^2}. \qquad (6)$$

3. NONLOCAL INPAINTING ALGORITHM

3.1. Algorithm

We solve Eq. (3) by an iterative procedure, and denote iteration indices by superscripts. The initial solution then writes Φ_i^0 for each component of Φ^0. Solving Eq. (3) is equivalent to minimizing the energy:

$$\mathcal{E}_i^w\left(\Phi_i, \Phi_i^0, \lambda\right) = \frac{1}{2} \sum_{v \in V} |\nabla_w \Phi_i(v)|^2 + \frac{\lambda}{2} \|\Phi_i - \Phi_i^0\|_2^2, \qquad (7)$$

for each component $i \in [1, d]$. Since $\mathcal{E}_i^w$ is convex, we need to solve the system of equations :

$$\frac{\partial \mathcal{E}_i^w(\Phi_i, \Phi_i^0, \lambda)}{\partial \Phi_v} = 0, \ \forall v \in V, \qquad (8)$$

which can be done using Gauss-Jacobi iterations. Finally, the update iterations write (see [15] for a detailed derivation) :

$$\begin{cases} \Phi_i^{(0)} = \Phi_i^0 \\ \Phi_i^{t+1}(v) = \frac{\lambda \Phi_i^0(v) + \sum_{u \sim v} 2w(u,v) \Phi_i^t(u)}{\lambda + \sum_{u \sim v} 2w(u,v)}. \end{cases} \qquad (9)$$

This leads to the following iterative algorithm, where the superscript depicts the index of the current iteration :

1. initialize Φ^0 from I ;
2. construct the non-local graph $\mathcal{G}_{nl}^t$ and the current embedding Φ^t from the image I^t;

[1] For color images, we stack the RGB or LAB values the same way, hence multiplying the patch size by 3.

3. compute the regularized embedding $\hat{\Phi}^t$ with respect to $\mathcal{G}_{nl}^t$ by applying Eq. (9) ;
4. back project the patches according to $\hat{\Phi}^t$ to form the image $\hat{I}^{t+1}$;
5. compute the updated solution I^{t+1} as I^0 in $\Omega \backslash \omega$ (outside the hole) and $\hat{I}^{t+1}$ inside ω ;
6. go back to step 2 until done.

Remark that, by replacing the non-local weights of $\mathcal{G}_{nl}$ by fixed values depending on pixel locations, we fall back to a geometry-based diffusion process.

3.2. Implementation details

The update iterations described in Eq. (9) are straightforward to implement. The back-projection step at the end of each iteration leads to overlapping patches. In this case, we simply average the corresponding values weighted by the position of the current pixel in each of the overlapping patch. The weight of each contribution is given by the Gaussian G_σ that defines the spatial extension of a patch.

Using a full non-local graph requires to compute and store the weights between all the patches of an image. To overcome this computational burden, we did not build the full non-local graph, but instead a k-nearest neighbour version of it, i.e. a vertex can have at most k neighbours that are the nearest according to the function w. In all the experiments, k is kept small (between 1 and 10), and the neighbours are searched in a restricted (yet large) area to avoid the exploration of the full image. This restriction is discussed in the following section.

The choice of the parameter h is of interest. From Eq. (1), it is clear that smaller values of h are of better choice since higher selectivity produces less averaging between patches. In the limit, h equals to 0 corresponds to copying only identical patches. This case can be approximated in the implementation by looking for the nearest neighbour of a given patch, and assigning to it a weight of 1 (see also [13] for a similar argument).

All operations are done pixel or patch-wise. Hence, it is immediate to derive a parallel implementation of the iterations in Eq. (9) to reduce the computation time.

4. INPAINTING RESULTS

4.1. Validation

First, we validate that a perfect non-local graph contains enough information for the inpainting. In this experiment (Fig. 1), we build the graph before removing some image parts and let the non-local diffusion iterate. As can be seen, an isotropic diffusion defined on a non-local graph can correctly inpaint textures.

4.2. Real images

In all our experiments we chose $h = 0$ and k between 1 and 10, and proceed with 10 to 50 iterations of the proposed algorithm. Note that the nearest neighbour of a given patch is always defined. In the worst case, if a patch is very singular, it will not be updated at once. Instead, it will stay black for several iterations and will start evolving when the surrounding patches will also get closer to the remaining of the image. We did not use any additional information, such as a confidence map [8], to improve the results, since our primary goal is to explore the effects of the non-local diffusion.

Fig. 1. Inpainting assuming perfect knowledge of the non-local graph. Left : input image with holes. Right : detail of the reconstruction.

Patches need to be chosen large enough to contain an entire example of the texture to reproduce. In our experiments, we fixed the width of the spatial Gaussian σ to values between 1 and 3. There is however a trade-off to be found here : bigger patches will allow to repeat larger texture patterns, but will be averaged (in the non-local space) to produce a flatter estimate. Hence, images containing high frequency textures should be inpainted using small patches. This is the case, for example, of the bungee jumper image (Fig. 2), where we took patches of 5-by-5 pixels. Conversely, the statue image (Fig. 3) required wider (25-by-25 pixels) patches in order to correctly reproduce the vegetation.

Fig. 2. Bungee jumper image. The inpainting mask is the same as [1]. Note that some texture has been inpainted on the water. The pink band is not a real mistake : the algorithm chose to repeat a pattern from the legs and the shoes that was close to its initial guess.

The proposed algorithm is iterative. Hence, the final result depends on the initial solution, here the input image. This explains why our algorithms sometimes produces visually smooth but physically impossible or unexpected results. In Fig. 2, the bungee knot was replaced by a texture taken from the shoes and legs because of its initial colour and shape. This behaviour is more disturbing in Fig. 4, where the dark dominant colour mislead the non-local nearest-neighbour search. Consequently, the fence pattern, which is darker than the other textures, was used to fill almost the entire designated area and smoothly mixed to its surroundings. In a real graphics software,

Fig. 3. Statue image (user designated area in red). Note that the algorithm successfully reproduced complex foliage texture.

such an image should be inpainted using one step for each different texture (ground, fence, sake bottles) to circumvent this problem.

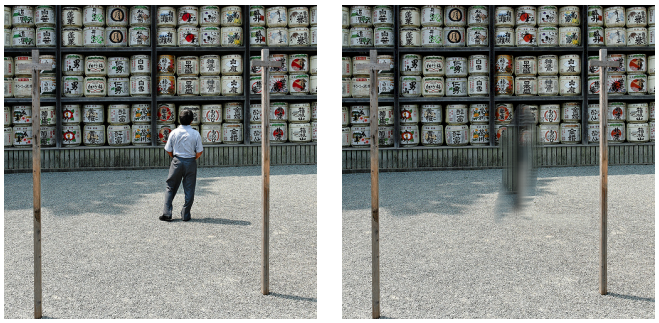

Fig. 4. Example of failure. Since initial colours were dark, the darker texture was extended and smoothly mixed to the surrounding areas.

4.3. Is full non-locality always desirable ?

Since building the non-local graph is computationally expensive, we restricted the search of a non-local neighbour in a smaller vicinity around the hole. Using principal component analysis and kd-tree sorting, it is possible to speed-up the whole process. However, our first experiments did not show any improvement. Preliminary investigation showed that many hole pixels were mislead by the dominant colour and chose the same neighbour, producing a flat image.

5. CONCLUSION AND FUTURE WORK

In this work, we proposed to use a graph-based variational framework to tackle the problem of texture inpainting. In contrast with previous methods, this algorithm can handlenaturally both geometric-based and exemplar-based approaches for inpainting, which comes from the use of graphs to implement the underlying diffusion processes. Hence, graphs seem a natural tool to extend the non-local methods to more complex problems than denoising, simply by re-using previously existing diffusion-based methods.

As a future work, the described inpainting method would obviously require a few refinement, such as using a confidence map. However, as pointed out in the experiments, it is important to make a deeper study of the non-local space induced by the patches. This will be made possible by using fast nearest neighbour methods, such as the PCA approach mentioned. An interesting study would be to find an optimal patch size, or an adequate way of combining patches, that would be suitable for both large and sharp texture synthesis. Building a patch scale-space, where patches get larger but contain less and less high-frequency components, seems also promising.

6. REFERENCES

[1] M. Bertalmio, A.L. Bertozzi, G. Sapiro, et al., "Navier-stokes, fluid dynamics, and image and video inpainting," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2001, vol. 1.

[2] T.F. Chan and J. Shen, "Mathematical models for local nontexture inpaintings," *SIAM Journal on Applied Mathematics*, vol. 62, no. 3, pp. 1019–1043, 2001.

[3] T.F. Chan and J. Shen, "Nontexture inpainting by curvature-driven diffusions," *Journal of Visual Communication and Image Representation*, vol. 12, no. 4, pp. 436–449, 2001.

[4] A.A. Efros and T.K. Leung, "Texture synthesis by non-parametric sampling," in *International Conference on Computer Vision*. Corful, Greece, 1999, vol. 2, pp. 1033–1038.

[5] A.A. Efros and W.T. Freeman, "Image quilting for texture synthesis and transfer," in *Proceedings of SIGGRAPH 2001*. Los Angeles, CA, 2001, pp. 341–346.

[6] V. Kwatra, A. Schodl, I. Essa, G. Turk, and A. Bobick, "Graph-cut textures: Image and video synthesis using graph cuts," *ACM Transactions on Graphics*, vol. 22, no. 3, pp. 277–286, 2003.

[7] P. Harrison, "A non-hierarchical procedure for re-synthesis of complex textures," in *Proc. Int. Conf. Central Europe Comp. Graphics, Visua. and Comp. Vision*. Citeseer, 2001.

[8] A. Criminisi, P. Perez, and K. Toyama, "Region filling and object removal by exemplar-based image inpainting," *IEEE Transactions on Image Processing*, vol. 13, no. 9, pp. 1200–1212, 2004.

[9] Y. Wexler, E. Shechtman, and M. Irani, "Space-time completion of video," *IEEE transactions on pattern analysis and machine intelligence*, vol. 29, no. 3, pp. 463, 2007.

[10] M. Bertalmio, L. Vese, G. Sapiro, and S. Osher, "Simultaneous structure and texture image inpainting," *IEEE Transactions on Image Processing*, vol. 12, no. 8, pp. 882–889, 2003.

[11] L.A. Vese and S.J. Osher, "Modeling textures with total variation minimization and oscillating patterns in image processing," *Journal of Scientific Computing*, vol. 19, no. 1, pp. 553–572, 2003.

[12] Antoni Buades, Bartolomeu Coll, and Jean-Michel Morel, "A review of image denoising algorithms, with a new one," *SIAM Multiscale Modeling and Simulation*, vol. 4, no. 2, pp. 490–530, 2005.

[13] P. Arias, V. Caselles, and G. Sapiro, "A variational framework for non-local image inpainting," *Proc. of EMMCVPR. Springer, Heidelberg*, 2009.

[14] L. P. Yaroslavsky, *Digital picture processing. An introduction.*, Springer Verlag, 1985.

[15] A. Elmoataz, O. Lezoray, and S. Bougleux, "Nonlocal discrete regularization on weighted graphs: a framework for image and manifold processing," *IEEE Transactions on Image Processing*, vol. 17, no. 7, pp. 1047–1060, 2008.