A smarter exemplar-based inpainting algorithm using local and global heuristics for more geometric coherence

Maxime Daisy, Pierre Buyssens, David Tschumperlé, Olivier Lézoray

To cite this version:
Maxime Daisy, Pierre Buyssens, David Tschumperlé, Olivier Lézoray. A smarter exemplar-based inpainting algorithm using local and global heuristics for more geometric coherence. IEEE International Conference on Image Processing (ICIP 2014), Oct 2014, Paris, France. hal-01080029

HAL Id: hal-01080029
https://hal.archives-ouvertes.fr/hal-01080029
Submitted on 4 Nov 2014

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
A SMARTER EXEMPLAR-BASED INPAINTING ALGORITHM USING LOCAL AND GLOBAL HEURISTICS FOR MORE GEOMETRIC COHERENCE

M. Daisy, P. Buyssens, D. Tschumperlé, O. Lézoray

GREYC (UMR 6072) - CNRS, Université de Caen Basse-Normandie, ENSICAEN - Image Team
6, Bd. Maréchal Juin - 14000 Caen, FRANCE

ABSTRACT

In this paper, we propose two major improvements to the exemplar-based image inpainting algorithm, initially formulated by Criminisi et al. [1]. First, we introduce a structure-tensor-based data-term for a better selection of pixel candidates to fill in based on priority. Then, we propose a new lookup heuristic in order to locate the best source patches to copy/paste to these targeted points. These two contributions clearly make the inpainting algorithm reconstruct more geometrically coherent images, as well as speed up the process drastically. We illustrate the great performances of our approach compared to existing state-of-the-art methods.

Index Terms— Exemplar-based image inpainting, Structure tensor analysis, Patch lookup strategy.

1. INTRODUCTION

Inpainting is the process of reconstructing unknown (masked) image regions such that the resulting image looks as natural as possible. Since 2000, this kind of algorithm has raised a wide interest in the image processing community due to the various applications it is able to deal with, such as object removal, or image repairing and interpolation. The literature on inpainting algorithms, reviewed in [2], exhibits mainly two kinds of approaches: geometry-based, and patch-based methods.

- **Geometry-based** methods [3, 4, 5] aim at reconstructing image regions using various semi-local geometry-aware interpolations of the known (noncorrupted) image data. These methods usually provide interesting results in terms of local geometry consistency. However, they mostly fail at reconstructing wide-scaled textures and often lead to flat-looking images particularly when large holes have to be filled.
- On the other hand, **patch-based** methods [1, 6, 7, 8, 9] are the logical sequel of previous algorithms originally proposed for texture synthesis [10]. They focus on finding patterns which are present in the known part of the image and fit visually well with the local data lying on the boundaries of the missing regions. Such similar local pieces of images (i.e. patches) are then copied inside the masked region to fill in, iteratively. This kind of methods has proven to work remarkably well in case of quite large holes to fill, due to its ability to synthesize large-scale textures respectful of the image content. Nonetheless, such methods often lack a good global geometry analysis of the image. This leads to some unrealistic reconstruction of geometrical structures (often with block artefacts for instance). Although hybrid methods have been more recently proposed in the literature [11, 12] to take advantages of both geometric and texture approaches, most of these visual artefacts remain. In this paper we focus on the so-called exemplar-based inpainting algorithm initiated by Criminisi et al. in [1], which has been quickly considered as a reference method in the field of patch-based inpainting. This paper proposes two important adjustments to the original method and show that they both clearly improves the geometric coherence of the inpainted results: we first reformulate the so-called data-term, in order to better select the pixel candidates to fill in based on priority. Then, we introduce a new lookup heuristic that is able to locate source patches to be copied back to the masked hole, in a very efficient way (both in terms of visual quality and execution speed). These two geometry-guided contributions greatly improve the visual quality of inpainted results when complex geometrical structures are involved. We illustrate the effectiveness of our approach with difficult reconstruction examples of real images.

2. CONTEXT AND CONTRIBUTIONS

**Examplar-based inpainting:** Back in 2004, Criminisi et al. proposed a greedy inpainting algorithm [1] aiming at reconstructing a missing region \( \Omega \) of an image \( I \) piece by piece, with a priority scheme. The order of pixel reconstruction was defined by a priority function \( P_p \) estimated at each \( p \in \delta \Omega \) (\( \delta \Omega \) being the boundary of \( \Omega \)). The priorities \( P_p = C_p \times D_p \) are defined over \([0, 1]\) i.s.t. the confidence term \( C_p \) that only depends on the local topology of \( \Omega \) (the more pixels of \( \Omega \) in the neighborhood of \( p \) means a lower confidence), while the data term \( D_p \) also depends on local image structures:

\[
D_p = \frac{\langle \nabla \hat{I}_p, \hat{m}_p \rangle}{\alpha} \text{ with } \nabla \hat{I}_p = \{ \nabla \hat{I}_q \mid \text{arg max}_{q \in (\mathcal{I} - \Omega) \cap \Psi_p} \| \nabla \hat{I}_q \| \}
\]

This research was supported by French national grant Action 3DS.
is the normal vector to the mask at \( p \) is the normal vector to the mask at \( p \), and \( \alpha \) a constant normalization factor (that can be in fact ignored). \( D_p \) is intended to favor the reconstruction of local linear structures orthogonally crossing \( \Omega \) at \( p \). Once all priorities \( P_p \) have been computed, the pixel \( p \) with the highest priority is chosen as the target pixel. For this \( p \), the patch \( \psi_p = \{ \psi_q \mid q \in \{ I - \Omega \} \} \) is sought all over the image (or generally into a square lookup region of fixed size centered at \( p \)) and is drawn at \( p \) in \( \Psi_p \cap \Omega \). The confidences are then updated with \( \forall q \in \Psi_p \cap \Omega, C_q = C_p \) as well as the boundary \( \delta \Omega \). The whole process is repeated until \( \delta \Omega = \emptyset \). This algorithm has been actually praised for the overall quality of the image reconstructions, and the acceptable execution time it provides. The heuristics proposed for defining the priorities \( P_p \) were actually good and quite hard to improve (some recent attempts in that direction has been tried in [2, 13]). Anyway, the algorithm has some drawbacks that we propose to manage hereafter. Note that, as the inpainting process is iterative, and as each patch \( \Psi_p \) copy-pasted in \( \Omega \) at one iteration depends mostly on the local image geometry (i.e. on what happened during all previous iterations), it is not so surprising improving even slightly one or two aspects of the initial algorithm leads finally to great visual ameliorations in the results.

**A better priority term:** The data-term \( D_p \) must tell about whether linear structures are present or not at a point \( p \) and with which angle they eventually cross the region \( \Omega \). Unfortunately, as \( p \in \delta \Omega \) on an point having no data, the gradient \( \vec{\nabla} I_p \) cannot be precisely estimated at this point. In [1], it is suggested to approximate \( \vec{\nabla} I_p \) as the maximum value of the image gradient in the known neighborhood \( \Psi_p \cap I \) of \( p \) (1). This approximation is actually a bit too coarse to get a locally accurate geometry analysis: when the patch size \( N \) is large, \( D_p \) will be high not only on the exact location of an eventual image contour crossing \( \Omega \), but also for every target candidates no more distant than \( N \) to this contour. This dilation effect of \( D_p \) gives too much importance to these neighboring points and one of them can be unluckily selected as having the highest priority (instead of the contour point itself), particularly if its confidence term \( C_p \) is higher. Fig.1(a)-(b) illustrates this assertion: here, \( D_p \) is high for all target points around the sand-sea frontier, and the final selected target patch to reconstruct (in hatched purple) will not be centered at the contour. A patch containing only sand will be likely pasted there, breaking seriously this local image structure.

To overcome this issue, we propose a more geometry-aware approach by using structure tensors [14] to modelize the image variations inside a candidate patch \( \Psi_p \). First, this has the interest of better managing local image structures using a channel-correlated approach (R,G,B channels will be taken into account simultaneously). Second, we take advantage of the algebraic properties of the tensor sum, to allow patches containing structures with multiple orientations (typically textures) to score favorably whatever the local normal vector \( \vec{n}_p \) to \( \delta \Omega \) is. We propose to define \( D_p \) as:

\[
\hat{D}_p = \|G_p \vec{\nabla} I_p\| \quad \text{with} \quad G_p = \sum_{q \in \Psi_p \cap (I - \Omega)} w_q \vec{\nabla} I_q \vec{\nabla} I_q^T \quad (2)
\]

\( G \) is a weighted average of structure tensors estimated on non-masked parts \((I - \Omega)\) of the target patch \( \Psi_p \) and \( w_p \) is a normalized 2d gaussian function centered at \( p \), with a given constant standard deviation. Thus, these properties hold:

- When \( G \) is anisotropic \( (G \approx \vec{n} \vec{n}^T) \), one main contour oriented along \( \vec{n}^\perp \) lies inside the patch \( \Psi_p \). (2) is almost equal to \( | \vec{\nabla} I_p \| \) and will be indeed high when the contour crosses \( \Omega \) orthogonally. Note that \( D_p \) will be higher for target points \( p \) that are precisely on the image contour (thanks to the weights \( w \) which will be maximal at the center of \( \Psi_p \)). No more dilation effects occur, and the contour point itself will have the highest data-term priority (Fig.1c).
- When \( G \) is isotropic (i.e. \( G \approx \lambda \text{Id} \)), (2) is almost equal to \( \lambda \). As expected, it is low for homogeneous regions. But, it remains high when \( \Psi_p \) contains a lot of variations with

---

**Fig. 1.** Illustration of the impact of the data terms (1) (original) or (2) (ours) for selecting the priority points to inpaint.
different directions (λ high) whatever the orientation of the mask normal $\vec{n}_p$ is (on the contrary, the original formulation (1) can be low there if the maximum gradient and the normal vector $\vec{n}_p$ are orthogonal by misfortune).

By using (2), we improve the finding of the point to reconstruct in priority. This has actually a tremendous incidence on the overall quality of the reconstructed images, as the iterative nature of the inpainting algorithm propagates reconstruction errors (Fig.3(b)-(c) show an example of differences).

**A better patch lookup procedure:** Our second contribution consists in the proposal of an improved patch search technique for the examplar-based inpainting algorithm [1]. We actually borrow ideas from the famous Patchmatch algorithm [15] and its recent extensions [16, 9]. These algorithms actually propose to efficiently compute an approximate Nearest Neighbor Field (NNF) for each point $p$ of the image, which quickly returns the most similar patch to the one centered at $p$. This has been successfully used within multiscale pixel-based inpainting approaches, but it becomes far less efficient when dealing with purely patch-based methods, as it requires an update of the NNF each time a new patch is pasted, which is time-consuming in practice.

Our proposed extended search scheme relies on a similar use of offsets to reduce the patch lookup space. Let $p_N$ be the center of the partially unknown target patch $\Psi_{p_N}$ to fill in, $\mathcal{A} = \{p_0, \ldots, p_{N-1}\}$ the set of centers of the patches $\{\Psi_{p_0}, \ldots, \Psi_{p_{N-1}}\}$ already pasted during the previous iterations of the inpainting algorithm. $NN(p_i) = \hat{p}_i$ the center of the best patch $\psi_{\hat{p}_i}$ found for $\Psi_{p_i}$, and $NNF = \{\hat{p}_0, \ldots, \hat{p}_{N-1}\}$ the set centers of the best patches.

We first define the offset vector between $p_N$ and the points of $\mathcal{A}$ as $\Delta_{p_N} = p_N - p_i, \forall i \in \{0, \ldots, N - 1\}$. Let $S$ be the set of points $p_i$ of $\mathcal{A}$ for which $\|\Delta_{p_N}^f\| < T$. Our proposed search scheme consists in extending the search space by considering several lookup sub-windows centered around each point $NN(q) - \Delta_{p_N}^q, \forall q \in S$. This scheme allows to search for the best patch candidate in potentially more interesting windows w.r.t. the already pasted patches. Note that a search window centered on $p_N$ is still considered for the lookup. In order to reduce the number of overall lookup candidates, we set the size $sz_N$ of all these sub-windows as $sz_N = sz / \sqrt{Card(S)}$, where $sz$ is the size of the initial search window (fixed parameter, used when no patches have been already reconstructed in a neighbor of the target point $p \in \delta$). The rationale behind this size is to keep a similar complexity regardless the number of visited sites. A schematic view of the process is illustrated on Fig. 2, where the target patch $\Psi_{p_0}$ is the dotted square inside $\Omega$, and the two search windows are depicted by the two large dotted squares centered on the initial guesses (obtained by the offsets $\Delta_{p_1}$ and $\Delta_{p_2}$ w.r.t. $\hat{p}_1$ and $\hat{p}_2$). For clarity purposes, the search window centered on the $p_3$ is not shown. Note that to inpaint a patch such as $\Psi_0$, the search window is bigger and centered on its center $p_0$ since it has no direct already reconstructed neighbors (i.e. $S = \emptyset$). The advantages of our scheme are:

- As lookup sub-windows may overlap (they do it often in practice), the number of lookup candidates is always lower than the one defined by the initial searching scheme [1], hence we accelerate the whole process by a non-negligible order of magnitude (see Fig.3(a)-(d)).
- As the size of the search space is reduced, the approximate nearest neighbor patch may be not optimal (in the sense of the SSD), but in fact it is often visually better. Lookup sub-windows are indeed centered at geometrically coherent locations w.r.t. the previously pasted patches. As a consequence, our modified inpainting algorithm becomes less sensitive to the patch size: in practice, one can recover larger coherent image structures and textures while considering smaller patches sizes than the original examplar-based algorithm [1].

**3. RESULTS AND CONCLUSION**

Fig. 3(a)-(d) illustrates the successive improvements of our two contributions to the original examplar-based inpainting algorithm [1]. The large size of the mask as well as the very complex structures involved in this image make it a very difficult case for an image inpainting algorithm. It clearly appears that our two improvements make the inpainted image more respectful of the initial image content. Fig. 3 shows that our final inpainting algorithm clearly competes with other state-of-the-art methods [17] (modified examplar-based inpainting) and [8, 16] (base of the Photoshop content-aware filling algorithm). For each result, an additional spatial patch blending step has been used to slightly improve the visual coherence (this is one of our previous work published in [18]). To ease the reproducibility of our research work, we have released the C++ sources and binaries of this inpainting algorithm within the G’MIC open-source framework [19].
Fig. 3. First row: Illustration of the successive contributions presented in this paper. Then from top to bottom, comparisons with state-of-the-art inpainting algorithms: Masked images, results from Lemeur et al. [17], results from Wexler et al. [8] (using Patchmatch [16]), and results from our inpainting algorithm. We have tried to find optimal parameters to generate each of these results.
4. REFERENCES


