Fast and Improved Exemplar-Based Inpainting Techniques for Natural Images

Lingni Ma  Luat Do  Peter H.N. de With

Eindhoven University of Technology
Department of Electrical Engineering
P.O. Box 513, 5600MB Eindhoven, the Netherlands

{L.Ma.1, Q.L.Do, P.H.N.de.With}@tue.nl

Abstract

Image inpainting is an image completion technique that has a wide range of applications such as image restoration, object removal and occlusion filling in view synthesis. In this paper, two novel techniques are proposed to enhance the performance of Criminisi’s algorithm, which inpaints images with an exemplar-based approach. First, a gradient-based searching is developed, which drastically lowers the computational complexity of global searching. Second, the patch matching process is modified with a distance-dependent criterion, such that the accuracy of the best matching candidate is enhanced. The experimental results have shown that with our proposed technique, computational cost is substantially reduced and the inpainting quality is also improved. For large images of 1024 × 768 pixels, our inpainting algorithm is almost 5 times faster than the original algorithm.

1 Introduction

Sensing of the environment with natural images is an established technique for geo-referenced and surveillance imaging, but this has sometimes imperfections due to object changes in the environment. For further processing, these undesirable objects need to be removed, which produces holes in those images. These holes can be filled in by image completion techniques, also known as image inpainting.

In literature, there are mainly two classes of inpainting algorithms. One class is based on Partial Differential Equations (PDE) [1], where the key feature is to propagate structures into holes via diffusion. The other class of algorithms is based on exemplars, employing the principle to generate textures by sampling and copying the known pixels for filling the missing area. One major drawback of the PDE-based approach is the blurring effect due to diffusion, which becomes especially noticeable for large holes. Therefore, in this paper we have concentrated on the exemplar-based algorithms [2–8].

Pioneering work for exemplar-based algorithms has been developed by Criminisi et al. [2,3], where a priority function is proposed to determine the inpainting order, such that linear structures are propagated correctly to connect broken lines. Although Criminisi’s algorithm performs well in most situations, there is a need to improve its efficiency and accuracy. For this purpose, several techniques have been developed. A robust priority function is proposed in [4], which enhances the accuracy for inpainting of large circular holes. To lower the computational cost, Chen et al. [5] suggest to use a fixed window size for searching candidate patches. However, no solution has been given to optimize the window size. An alternative technique to limit the search window size is developed in [6], where the authors propose to correlate the search window size with the hole size. Since this algorithm does not consider the texture information, its hole filling results are not sufficient for all situations. Recently, more advanced techniques
have been developed to improve the priority computation and the patch matching [7,8]. These algorithms improve the quality, but meanwhile also demand complex analysis and thereby a higher computational cost.

In this study, we aim at improving the efficiency and accuracy with two new techniques. First, a Gradient-Guided Search (GGS) is developed to limit the window size for searching candidate patches, which reduces the computational cost without sacrificing the inpainting performance too much. Second, a Distance-Dependent Patch Matching (DDPM) is proposed to enhance the patch matching accuracy. From our experiments, we have observed that the execution time of our proposed algorithm becomes substantially lower and the inpainting quality is also improved. The sequel of this paper is organized as follows. Section 2 gives the background information for exemplar-based inpainting algorithms with the emphasis on Criminisi’s algorithm. Section 3 explains our proposed technique to improve the performance of Criminisi’s algorithm. Section 4 shows the experimental results and evaluations and Section 5 presents the final conclusions.

2 Background of the Exemplar-Based Algorithm

In this section, we mainly focus on the discussion of Criminisi’s algorithm, which is one of the most widely used exemplar-based approach and also forms the basis for our development. The algorithm is first briefly explained, followed by the analysis of its limitations.

2.1 Criminisi’s Algorithm

It has been observed that the exemplar texture synthesis is capable to extend linear structures to connect broken lines. However, to achieve perceptually satisfying results, a correct inpainting order is crucial. It is desirable to first inpaint patches along the continuation of edges, which ensures the propagation of linear structures prior to the synthesis of similar textures.

Criminisi’s algorithm provides a good solution to the above requirement with a well designed priority function. The algorithm is an iterative process, which consists of three main steps. In the first step, a target patch with the highest priority is determined. Given the notation in Fig. 1, the priority for a patch $\Psi_{p}$ centered at pixel $p$, is computed by

$$ P(p) = C(p) \cdot D(p), $$

Fig. 1: Diagram for explaining notations: a target patch $\Psi_{p}$ centered at pixel $p$ is marked with a black frame. The variables $\Phi$ and $\Omega$ denote the known and missing regions in the image, respectively. The parameter $\delta \Omega$ shows the filling front. The vector $\nabla I_{p} \perp$ points to the edge direction and $\mathbf{n}_{p}$ is orthonormal to $\delta \Omega$.
where the confidence term $C(p)$ and the date term $D(p)$ are specified as

$$C(p) = \frac{\sum_{q \in \Psi_p \cap (\Phi + \delta \Omega)} C(q)}{|\Psi_p|}, \quad (2a)$$

$$D(p) = \frac{|\nabla I_p \perp \cdot n_p|}{|p|}. \quad (2b)$$

The term $C(p)$ and $D(p)$ measure the amount of known information and the relative position between the edge and the filling front $\delta \Omega$, respectively. In the second step, the algorithm performs a global searching to find a best matching candidate according to the texture distance $d(\Psi_p, \Psi_q)$, which is the sum of squared errors between the target and the candidate patch. In the last step, the image is updated by copying the candidate patch to fill the target patch.

### 2.2 Limitations of Criminisi’s Algorithm

Criminisi’s algorithm performs well in most situations, however, it suffers from two main drawbacks that degrade its performance. First, the computational cost increases drastically with the image size. Since the algorithm demands a global patch searching, the execution time becomes quite lengthy and thereby impractical for many applications. Second, the algorithm can produce noticeable artifacts, because its patch matching is merely dependent on texture. Since only known pixels are compared, the candidate can contain undesirable information, which can lead to a degradation of the successive inpainting steps. In the next section, we will propose two main techniques to reduce these limitations.

### 3 Improvements for Criminisi’s Algorithm

In this section, we describe our modified version of Criminisi’s algorithm. Fig. 2 shows the modified diagram for one iteration. In the first step, we select a proper target patch by computing the priority similar to Criminisi’s algorithm. In the second step, we calculate the search window size for candidate patches based on the gradient. In the third step, an optimal candidate is determined according to our modified criterion which is dependent on the location distance in addition to the texture similarity. In the last step, the image is updated by copying the corresponding pixels in the candidate patch. Let us now describe our proposed techniques in detail.

**Fig. 2**: Principal steps of our proposed algorithm for one iteration.

#### 3.1 Fast Gradient-Guided Search

The first improvement is based on the observation that the magnitude of the image gradient is an indication for the variation of the texture or the intricate details. When the gradient magnitude is high, it indicates the presence of abrupt changes, such as edges. In this case, it is desirable to search in a larger region, in order to find a good patch that matches the intricate details. Alternatively, when the gradient magnitude is relatively low, the surrounding texture is rather smooth, so that the best matching
patch is very likely to be in the neighborhood of the target patch. Based on this assumption, we choose the search window size adaptively: we define the window size as a function of the gradient magnitude. The higher the gradient magnitude, the larger the search window for candidate patches.

**Fast searching algorithm**

Given a patch $\Psi_p$ centered at pixel $p$, we compute the gradient magnitude $g$ of the patch as the highest gradient magnitude of neighboring known pixels. Then the search window radius $r$ is defined by

$$r = R^{\alpha g + \beta},$$

where $R$ is the user-specified maximum search window radius for the entire image. The variables $\alpha$ and $\beta$ are coefficients constrained by two conditions such that $R^{\alpha G_L + \beta} = R_L$ and $R^{\alpha G_U + \beta} = R_U$, where $R_L$ and $R_U$ are the lower and upper bound for the search radius, respectively. Likewise, variables $G_L$ and $G_U$ are the lower and upper bound for the gradient magnitude, respectively.

The mapping curve is presented in Fig. 3 (right column) together with the cumulative probability distribution of the gradient magnitude. The patch size is $9 \times 9$ pixels and the image size is $400 \times 300$ pixels. From our experiments, we set the parameters $G_L = 50$, $G_U = 100$, $R_L = 5$ and $R_U = 120$. The results show that with this mapping, the search radius for the majority of the patches is highly reduced, thereby a lower computational complexity is achieved.

The large benefit of the gradient-based searching is that it substantially reduces the computational cost with little sacrifice of the performance, since in most cases the target patch resides in low-frequency textures and searching in a large window is not necessary. In our algorithm, the search window size is correlated with the texture variation, while the approach in [6] correlates the search window size with the hole.
In contrast to our algorithm, the later approach does not work well for small holes surrounded by intricate details and does not reduce the computational cost for large holes residing in smooth texture.

3.2 Improvement by Distance-Dependent Patch Matching

As explained in Section 2.2, the texture distance is not always sufficient to find the best matching patch. From our early experiments, we have observed that a patch is more likely to resemble its neighboring patches than its far away counterparts. It is therefore logical to favor patches from neighboring locations for filling holes.

This assumption motivates our proposal for the distance-dependent patch matching technique, which involves the selection of candidate patches by adding the location distance as a penalty criterion to the patch matching process. Our modified technique consists of three steps.

1. Rank the candidate patches according to their texture distance.
2. Select the first $N$ patches with the smallest distance.
3. From these $N$ patches, add the location distance as a penalty to their texture distance and rank them again.

We define the location distance by

$$D(\Psi_p, \Psi_q) = \gamma \cdot (|x_p - x_q| + |y_p - y_q|),$$

(4)

where $\gamma$ is the weight and $(x_p, y_p), (x_q, y_q)$ are the centers of patch $\Psi_p$ and $\Psi_q$, respectively. The patch with the smallest modified distance is selected as the candidate.

The modified patch matching produces better results in two situations. First, when several candidates have the same texture distance, the location distance helps to select the correct patch. Moreover, in situations where a candidate with minimum texture distance is located further away from the target, a higher penalty reduces the risk of including undesirable details.

4 Experimental Results and Discussion

In this section, we discuss the experimental results of our proposed algorithm. We have performed two series of experiments and compare them to Criminisi’s algorithm, which we have also evaluated for the same input images.

In the first series of experiments, we compare the computational cost of these two algorithms by examining the time consumption for the inpainting process. The test set consists of images with sizes varying from $320 \times 240, 640 \times 320, 768 \times 576, 800 \times 600$ to $1024 \times 768$ pixels. For each image size, 10 independent pictures are tested and the average iteration cycle time is compared. The result in Fig. 4 shows that the gradient-based searching drastically improves the execution time of the inpainting process, especially for images with a high resolution. Specifically, Fig. 4 indicates that our proposed method is almost 5 times faster than the standard Criminisi’s algorithm for images of $1024 \times 768$ pixels.

Evidently, we have to evaluate the obtained gain in computational efficiency against the possible visual degradation of the quality. For this reason, in a second experiment we have also compared the visual performance of Criminisi’s and our new algorithms of which the results are shown in Fig. 5. The objective of the experiment is to inpaint the images after removal of the dark birds and the person. It can be observed that our algorithm performs equally well with object inpainting and propagates linear structures to connect broken lines similarly to Criminisis algorithm. In addition, our proposed algorithm produces fewer noticeable artifacts in less structured areas, such as clouds and water (e.g. look to the birds shadow removal and the bridge reflection).
<table>
<thead>
<tr>
<th>Image Size (pixels)</th>
<th>Patch Size (pixels)</th>
<th>GS per iteration (ms)</th>
<th>GBS per iteration (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>320 × 240</td>
<td>09 × 09</td>
<td>34.2</td>
<td>11.6</td>
</tr>
<tr>
<td>640 × 480</td>
<td>11 × 11</td>
<td>86.6</td>
<td>24.4</td>
</tr>
<tr>
<td>768 × 576</td>
<td>11 × 11</td>
<td>123.3</td>
<td>29.9</td>
</tr>
<tr>
<td>800 × 600</td>
<td>13 × 13</td>
<td>129.0</td>
<td>30.5</td>
</tr>
<tr>
<td>1024 × 768</td>
<td>15 × 15</td>
<td>200.7</td>
<td>34.8</td>
</tr>
</tbody>
</table>

Fig. 4: Comparison of computational cost between Global Searching (GS) used in Criminisi’s algorithm and Gradient-Guided Searching (GGS) used in our improved algorithm.

Fig. 5: Inpainting results of Criminisi’s algorithm and our proposed algorithm. The objective is to inpaint the image after removal of certain objects, which in our case are the birds and the person.
5 Conclusions

In this paper, we have proposed two novel techniques to improve the inpainting performance of natural images based on Criminisi’s algorithm. We have improved this algorithm in two major aspects. First, we have made the search window for finding candidate patches adaptive to the magnitude of the local image gradient. This modification improves the execution time of the algorithm with a factor up to five, depending on the image size. The speed enhancement is especially noticeable for large images. Second, we have introduced an algorithm, called distance-dependent patch matching, which helps to select a more appropriate candidate patch, leading to a reduction of the artificial edges. It has been shown by visual evaluation experiments that the obtained quality is even improved despite the strongly reduced execution time. This is explained as follows: the distance-dependent search guides the algorithm towards local candidate patches which on the average have a better correlation than the patches result in global search. These advantages make our proposed algorithm more practical and attractive for inpainting of natural images.

References


