Context Aware Exemplar-based Image Inpainting using Irregular Patches

C. Fotsing¹ and D. Cunningham¹

¹Brandenburgische Technische Universität Cottbus-Senftenberg, Chair of Graphic Systems, Germany

Abstract

We propose a new exemplar-based image inpainting method in this paper. Our method is based on the Criminisi pipeline. We focused on three main stages of the pipeline: calculation of priorities, construction of patches, and the search for the best match. To assign a high priority to patches constructed from the edge pixels, we use the ability of segmentation algorithms to divide an image into different texture blocks. The patches built from pixels located at the border between several texture blocks receive a high priority. Unlike most patch-based image inpainting methods which use regular patches (rectangle, square), the shape and size of our patches depend on the textural composition around the original pixel. The patches are built using a region growing principle in the different texture blocks around the original pixel. The search for the best match is done contextually. We search for the best match of the patch with the highest priority in a similar environment to its neighborhood around the target zone. The method is simple and easy to implement. The experiments show that our method obtains more plausible results than the basic method of Criminisi and its improved version Amoeba in most cases.

Keywords: Inpainting, Exemplar-based, Color segmentation, Region growing

1. Introduction

Digital image inpainting is an important task in computer vision that aims to restore damaged or missing parts of an image. It is used in several applications ranging from the restoration of artworks (paintings, films, photographs) [BBS01, KMF95] to the removal of unwanted artifacts from images and videos (object removal) [CPT04a, LMWY13]. Unlike other image restoration problems like denoising where the pixels to be processed contain both the correct information and the bias, in image inpainting, we do not have any information on the value of pixels to be inpainted. The inpainting problem is described as follows: Let I be an Image of size $m \times n$. Mathematically, I can be defined as a two-dimensional function

$$I : M \times N \rightarrow R^k$$

where $k$ is the number of color channels (1 for gray images and 3 for RGB images), $M = \{1, 2, \ldots, m\}$ and $N = \{1, 2, \ldots, n\}$. Within the framework of the inpainting problem, $\exists \Omega \subseteq M \times N, \forall (x,y) \in \Omega \quad I(x,y) = ?$. In the literature, $\Omega$ is called the target zone, $M \times N \setminus \Omega$ the source zone denoted by $\Phi$, and the set of border pixels between the two regions is denoted by $\delta \Omega$. The objective of inpainting is to reconstruct the unknown pixels in a way that is not detectable to observers, i.e., the result should appear natural to the human eye and should be as physically plausible as possible [CPT04a].

To solve inpainting problems, researchers proposed a wide range of solutions based on different approaches [GL14]. One of the simplest and most effective solutions is the exemplar-based algorithm proposed in [CPT04a]. It is a simple-well-defined pipeline that gradually and iteratively fills the vacuum area $\Omega$. Once $\Omega$ and $\Phi$ are defined, the algorithm performs 4-main steps at each iteration: 1- it calculates the priorities of the candidate patches, 2- it searches for the best match for the candidate patch with the highest priority, 3- it propagates structures and textures into $\Omega$, and finally 4- it updates $\Omega$, $\Phi$, and $\delta \Omega$. The method’s effectiveness primarily depends on the reliability and precision of the processes used at each stage of the pipeline. Our main objectives in this work are: (1) to propose a new image inpainting method based on this pipeline. (2) to compare the results of our method with those of other methods.

The research has three main contributions. Firstly, we propose a new formula for calculating the priority of candidate patches. We use the reliability of image segmentation algorithms (ability to determine the borders between the different texture blocks of an image) to assign high priorities to candidate patches constructed from the texture border pixels (edge pixels). This ensures the continuity of the image structures in $\Omega$. Secondly, instead of using geometrically regular patches (squares, rectangles) like in traditional exemplar-based methods, we execute a local region growing process to build up the patches efficiently. This approach reduces the likelihood of copying inconsistent pixels. Finally, once the candi-
date patch with the highest priority is determined, the search for
the best match in $\Phi$ is performed contextually, i.e., we are not only
looking for the best match of the patch, but the best match in a con-
text similar to the one surrounding the patch in the target area. This
contextual approach improves the consistency between $\Phi$ and
the new texture created to fill $\Omega$.

The paper is organized as follows. In section 2, we review some
relevant works on image inpainting. We present the new method
in section 3. Section 4 describes the experimental procedures and
analyzes the results. The conclusions of the research are presented
in section 5.

2. Related works
Restoration of paintings is an activity as old as art. It consists of
restoring damaged paintings "by hand" (loss of paint, weakened
canvas, tears, water damage, or fire) so that the result is as similar as
possible to the original work or so that the changes are imper-
ceptible and consistent [CRG¹³]. It is a tedious task and requires
an artistic background and a lot of concentration. Researchers have
recently developed several digital solutions that attempt to virtually
replicate the basic techniques used by professionals to recover the
damaged parts of images [PPM12]. These solutions can be classi-
fied into two algorithmic classes: sequential algorithms [GL14] and
deep learning [SWFY20, PKD¹⁶].

2.1. Sequential methods
Depending on the used approach, sequential methods can be
grouped into three sub-classes: diffusion-based methods, exemplar-
based methods, and methods using both approaches.

2.1.1. Diffusion-based inpainting
Diffusion-based methods use principles similar to physical prop-
agation phenomena to locally propagate information from $\Phi$ into
$\Omega$. The basic principle consists in exploiting information from the
neighborhoods of the border pixels $\partial \Omega$ to smoothly extend $\Phi$ into
$\Omega$ while taking care to preserve the orientation of the isophote lines
[GL14]. Various mathematical models have been used in the frame-
work of diffusion-based image inpainting. They include Partial Differential Equations (PDE) [Sch¹⁵], Fourier Transform [MT¹⁹],
wavelets [CDNL98, DJL¹²], Cahn-Hilliard Equation [BHS09,
BEG07], statistical and stochastic modeling [GG84, LZW03].

In recent decades, the diffusion-based methods that have re-
ceived the most attention are those based on partial differential
equations. The pioneering work in the field is the method proposed by Bertalmío et al. [BSCB00]. After identification of the $\Omega$ area
by the user, they used an anisotropic diffusion model to smoothly
propagate neighborhood information from the border pixels $\partial \Omega$
into $\Omega$. To ensure the continuity of the isophote lines, the propaga-
tion was done according to the normal of the gradient vector of the
pixels along the border $\partial \Omega$. The main drawbacks of this technique
are its slowness and the difficulty to restore large textured regions.
In order to improve the computational time, Telea [Tel04] presented
a Fast Marching Method for image inpainting. They diffused an image smoothness estimator along the image gradient, similar to
[BBS01]. The image smoothness was estimated as a weighted aver-
age over a known image neighborhood of the pixel to be inpainted.
To maintain the isophote lines direction, other PDE-based models
have been suggested, including Total Variational (TV) [SC02] and
Curvature-Driven Diffusion [CS01].

Diffusion-based methods are suitable for completing straight
lines, curves and for inpainting small areas and, they avoid having
unconnected edges. However, they are not well suited to recover
the texture of large surfaces [GL14].

2.1.2. Exemplar-based inpainting
The general principle behind this approach is based on the idea
that it is possible to consistently restore damaged parts of an image
by filling unknown pixels with color values from the source area. Pix-
els can be restored in blocks (patch-based inpainting) [BDTL15] or
individually (pixel-based inpainting) [WL00a, DSC04].

Giving priority to edge pixels, Qiang et al. [QHX17] restored $\Omega$
pixel by pixel. After selecting border pixels with the highest pri-
ority at each iteration, they constructed (searched for) a subset of
candidate patches for each selected pixel $P_i$ similar to the patch
centered at $P_i$. The value of the center pixel of each candidate patch
is a possible value of $P_i$. The median method is adopted to select
the best filling value of $P_i$. As in other pixel-based methods, this
solution suffers from high computational costs and has difficulty
restoring large textures made up of many small objects [GL14].

To speed up the restoration process, an obvious solution is to
proceed by copying sub-regions of pixels (patches) at the same
time. One of the most promising patch-based methods was pro-
posed by Criminisi et al. [CPT04a]. As discussed above, it is a
pipeline made up of four main steps that restore $\Omega$ by gradually
sampling and copying sub-regions of color values from the source
region $\Phi$. To ensure the propagation of the image structures in $\Omega$,
they assigned a high fill order priority to patches containing edges.
The hindrance of this method is the propagation of synthetic er-
rors, i.e., copying a few unreliable pixels in one step is enough to
make the result inconsistent and implausible. Over the past two
decades, many improvements have been proposed. In most solu-
tions, the general idea of the pipeline remains unchanged. Only the
approaches used at one or more stages of the pipeline are improved
or modified [FZ18, OLK19].

To overcome the problem of discontinuous structures and inco-
stistent textures, several authors proposed improvements. Xu and
Sun [XS10] designed a patch structure sparsity function to assign
high priorities to patches located at the image structure. Instead
of directly copying the patch with the best match, they built the
patch to be filled as the sparse linear combination of candidate
patches under the local patch consistency constraint in a framework
of sparse representation. Lu et al. [LHL10] suggested using adap-
tive patch sizes according to structure and the local texture. After
calculating the best match, Castillo et al. [CCWB18] used a region
growing process (amoeba) to extract only consistent and reliable
pixels instead of copying all the pixels in the patch.

2.1.3. Hybrid methods
Hybrid methods aim to combine the two previous approaches by
simultaneously taking advantage of the capacity of the diffusion-

© 2021 The Author(s)
based methods to preserve the structure of images and the exemplar-based methods to reconstruct large textures. The idea behind is to separate images into several components (e.g., structure, texture), then to restore them individually using the most suitable approach and combine the results, or combine the different approaches into a global function.

Bertalmio et al. [BVSO03] suggested to decompose images into the sum of two functions (structure and textures), then to reconstruct each of the components separately using a suitable filling-in algorithm, and finally combine the two output components to have the result. Following the same pipeline, Grossauer [Gro04] proposed to fill-in the structure component with the PDE-based solution of [GS03]. He employed the texture synthesis algorithm in [WL00b] to restore the texture component. Wu and Ruan [JQ08] used a bi-directional diffusion PDE to inpaint the structure after separating the structure part from the texture part with the total variation equation. The texture was restored using an exemplar-based inpainting solution constrained by a cross-isophote diffused data term.

Unlike the previous solutions, which separated the decomposition and filling-in stages, Elad et al. [ESQD05] combined the two stages in a single task. The separation was done using the morphological component analysis (MCA) algorithm proposed in their previous work to decompose the image into texture and cartoon layers. They modeled the inpainting problem as an optimization. Bugeau et al. [BBCS10] proposed to combine the texture synthesis term, the diffusion term, and a third term (coherence term) into an energy function. The restoration was executed by minimizing the energy function.

2.2. Deep learning techniques

With the recent evolution in deep learning, several learning models have been proposed to restore damaged parts of digital images. The deep learning methods used for image inpainting are mainly based on traditional Convolutional Neural Networks (CNN) or Generative Adversarial Networks (GAN).

[KSSH14, CSL*17, KY15] proposed to first train a neural network model to automatically map the unknown pixels without user intervention (blind inpainting). The model suggested by Cai et al. [CSL*17] is a CNN with three convolutional layers, which takes as input a damaged image, identified the corrupted or unknown pixels, and automatically restores them. The model was trained using Stochastic gradient descent with standard backpropagation. After extracting the missing regions, Alliou and Yaghmaee [KY15] sorted them according to their size and then applied a pre-trained Generalized Regression Neural Network (GRNN) model for restoration. To restore shape with sharp structures and fine-detailed textures Yan et al. [YLL*18] incorporated a special shift-connection layer with guidance loss to the U-Net architecture. They took into account the shape of the missing region as an important parameter in the recovering process.

Introduce by Goodfellow et al. [GPAM*14] to perform generative modeling, GANs have been used by several authors to solve image inpainting problems [LYL*18, VSB19]. Pathak et al. [PKD*16] proposed an unsupervised visual feature learning algorithm driven by context-based pixel prediction to preserve the appearance and the semantics of visual structures in images. They trained a CNN model to predict missing pixels based on their surrounding context. Their model is a simple encoder-decoder pipeline. The role of the encoder is to extract a latent feature representation of the image and then using these features the decoder restores the unknown regions. To make the prediction look real, they used an adversarial loss function similar to [GPAM*14]. Focusing on both the local and global consistency of the inpainted image, Iizuka et al. [ISSI17] presented a learning model for image completion. Their architecture consists of a completion network and two auxiliary context discriminator networks used only for training the completion network. During the training process, the discriminators’ role is to check if the inpainted image generated by the completion network is real, while the completion network is trained to fool both discriminator networks.

3. Method

We present a new image inpainting method based essentially on the [CPTO4b] pipeline in this section. The method focuses on three main aspects: the calculation of the priorities, the construction of the patches, and the search for the best match. The main idea is to use the abilities of image segmentation algorithms (boundary determination between different texture blocks of an image) to assign high priority to patches constructed from border pixels located on the edges. Then, we use a region growing principle to build the pixel subset (patch), which has to be extended in Ω. Instead of searching only for the best patch match in Φ, we search for the best match in an area similar to the initial neighborhood of the patch around the target zone.

3.1. Priorities calculation

To ensure the continuity of image structures in Ω, it is necessary to give a high priority of fill order to the patch containing edges [CPTO4b]. Based on this observation, we proposed a new formula to calculate priorities. Let \( p_i(x_p, y_p) \) be a pixel belonging to the border \( \partial \Omega \), the priority of the patch constructed from \( p_i \) is

\[
P(p_i) = N(p_i) \times \text{Distance}(p_i, \text{Center}) \times \text{Rate}(p_i) / \text{Conf}(p_i) \tag{1}
\]

where:

\( N(p_i) \) corresponds to the number of texture blocks surrounding the pixel \( p_i \). It is determined using the color image segmentation process [GGGD14]. As shown in Figure 1(c) \( N(p_1) = 3, N(p_2) = 2, N(p_3) = 1 \). This implies that around pixels \( p_1, p_2, \) and \( p_3 \) there are 3, 2, and 1 texture blocks, respectively. \( N(p_i) \) assigns a high priority to the patches built from border pixels located at the edges. This ensures the propagation of image structures in Ω.

\[
\text{Distance}(P_i, C_i) \text{ is the Euclidean distance between } P_i \text{ and } C_i.
\]

In our approach, we have divided Ω into a disjoint sub-sets \( \Omega_1, \Omega_2, \ldots, \Omega_l \) as illustrated in Fig 1(b). Let \( C_i(X, Y) \) be the center of the subset \( \Omega_i \), and \( p_i(x_v, y_v) \) be any point in \( \Omega_i \); \( X = \frac{1}{l} \sum_{i=1}^{l} x_i, Y = \frac{1}{l} \sum_{i=1}^{l} y_i \). Assume that \( P_i \) is at the border of \( \Omega_i \). Then, \( \text{Distance}(P_i, C_i) \) is the Euclidean distance between \( P_i \) and

\[
\begin{align*}
\text{Distance}(P_i, C_i) &= \sqrt{(x_p - x_c)^2 + (y_p - y_c)^2} \\
&= \sqrt{(X - x_p)^2 + (Y - y_p)^2}
\end{align*}
\]
Once the pixel with the highest priority has been determined, we construct the subset of the known pixels that must be extended into $\Omega$. Unlike traditional methods that propose building a regular patch (circle, square, rectangle) centered around the border pixel, we suggest constructing an irregular patch using a region-growing technique. Let $P$ be the border pixel with the highest priority. The principle is to grow a region in each texture block around $P$ and finally merge the resulting blocks into one set (see Figure 3). The growth process in a block starts from a pixel $C$, a neighboring pixel of $P$ belonging to the corresponding block. The basic principle of the growing process is inspired by the morphological amoeba algorithm [LDM05]. Algorithm 1 describes the main idea behind the process. It takes as input an image to restore $GrayIm$, the central pixel from which the growth is made $P_0$, a cumulative difference threshold $\beta$ which is used to manage the luminosity variation in the texture block, a rate $\lambda$ used to control the patch extent according to the image size, and the Mask. We start by marking all the pixels of the image as unprocessed. $P_2$ is marked as belonging to the growing region $SegBlock$. Each of the neighboring pixels to $P_0$ (we denote them $P_i$) belonging to $\Phi$ receives as cumulative difference $P_i, cum\_diff = |GrayIm(P_i) - GrayIm(P_0)|$. In Algorithm 1, these steps are computed by the functions ComputeCum_diff( Neighbour($P_i$)). Then, all the $P_i$ are stacked in a stack $Stk$ and marked as treated. As long as $Stk$ is not empty, we unstack a pixel $P_i$. If $P_i, cum\_diff < \beta$ and $|P_i.x - P.x| <= \lambda * GrayIm.Width$ and $|P_i.y - P.y| <= \lambda * GrayIm.height$, $P_i$ is put into $SegBlock$. All the untreated neighboring pixels of $P_i$ belonging to $\Phi$ (we note them $P_j$) receive a value $P_j, cum\_diff = P_i, cum\_diff + |GrayIm(P_i) - GrayIm(P_j)|$. Each $P_i$ is then marked as processed and stacked in $Stk$. Once the process is applied to all the texture blocks in the direct neighborhood of $P$, the final patch is the union of the resulting $SegBlocks$.

3.3. Best match determination

The search for the best match is performed contextually. We find the best match in an environment similar to the home region (a neighborhood similar to the one surrounding the patch around $\Omega$). This avoids copying regions surrounded by neighborhoods different from the original one. We define a context with a predefined dimension around the template (the match). Note that the context size may be reduced depending on the location of the template. For example, templates located at the borders of the image frame can only be surrounded by a smaller context. As shown in Figure 4, we see that the best match which does not take into account the en-

Figure 1: Illustration of a segmentation case. a) $24 \times 24$ generated image, b) mask, c) segmented image

Figure 2: Known pixels priority box: rate($P_1$) > rate($P_2$) (see Equation 1)
environment (Match1), would lead to an incoherent extension of the green texture block into the gray metal bar.

Another context with the same dimension is defined around the patch in its original environment. As shown in Figure 5, the area delimited by $\Gamma$ contains both the patch $P$ and the neighborhood $V$. $\Gamma = P \cup V$, $P \cap V \cap \Omega = \emptyset$ and $||\Gamma|| = ||\Gamma'||$. The best match is the one that minimizes $SSD(P, P') + SSD(V, V') / |V|$. With $V \neq \emptyset$ and $SSD()$ referring to the sum of squared differences.

3.4. Patch extension and update

Once the best match is found, we perform region growing from the correspondents of the different $C_i$ pixels in the best match area (see Figure 6(b)). The union of the different resulting growing regions corresponds to an extension of the patch. Then, $\Omega$ is filled with the subset of the excess pixels (see Figure 6(c)). $\Omega$, $\Phi$ and $\delta \Omega$ are then updated. Let $P$ be the current border pixel from which the patch is built and, $R = \{P_1, P_2, ..., P_n\}$ the set of the border pixel generated after partially filling $\Omega$. $\forall P_i \in R, Conf(P_i) = Conf(P) + \ldots$

**Algorithm 1 Region growing: GrayIm, Mask, P, $P_k$, $\beta > 0$, $\lambda > 0$**

1: Stk $\leftarrow \emptyset$
2: Segblock $\leftarrow \{P_k\}$
3: $W$ = GrayIm.width
4: $H$ = GrayIm.height
5: $N \leftarrow \text{ComputeCum_diffNeighbour}(P_k, \text{GrayIm, Mask})$
6: $\text{MarkAsTreated}(N)$
7: $\text{Stack}(\text{Stk}, N)$
8: while $\text{Stk} \neq \emptyset$ do
9: $P_i \leftarrow \text{Unstack}(\text{Stk})$
10: if $P_i$.cum_diff $< \beta$ && $|P_i.x - P.x| < \lambda \times W$ && $|P_i.y - P.y| < \lambda \times H$ then
11: $\text{Segblocks} \leftarrow \text{Segblock} \cup \{P_i\}$
12: $N \leftarrow \text{ComputeCum_diffNeighbour}(P_i, \text{GrayIm, Mask})$
13: $\text{MarkAsTreated}(N)$
14: $\text{Stack}(\text{Stk}, N)$
15: end if
16: end while
17: return Segblock

![Figure 3](image-url) **Figure 3**: Patch growing. a) $24 \times 24$ generated image with target area $\Omega$. b) growing regions: $P$ is the border pixel with the highest priority, $R_1, R_2$ and $R_3$ have been grown from $P_1, P_2$ and $P_3$ respectively. c) Patch $= R_1 \cup R_2 \cup R_3$

![Figure 4](image-url) **Figure 4**: Illustration of contextual and non-contextual matching: Match2 is obtained using the context and Match1 without the context.

![Figure 5](image-url) **Figure 5**: Best match with context: $\Gamma = P \cup V$, $\Gamma' = P' \cup V'$, $P \cap V \cap \Omega = \emptyset$, $P' \cap V' \cap \Omega = \emptyset$, and $||\Gamma|| = ||\Gamma'||$. 
The process is repeated until we have no more border pixels. Figure 7 presents the progress of the restoration at some stages.

4. Experiments

To demonstrate the effectiveness of our method, we carried out several experiments. The datasets used in our experiments are images from [CCWB18]. This benchmark is known for containing images with attributes that present challenges in preventing artifact generation and preserving content and structure.

Instead of defining patches using a regular shape as in most Criminisi-based inpainting methods, we controlled the patch extent using two parameters \( \lambda \) and the maximal cumulative difference \( \beta \) (see algorithm 1). We submitted 16 incomplete images to restore (see Fig 8) to three exemplar-based inpainting methods; the original method of Criminisi et al. [CPT04b], the modified version (amoeba) proposed by Castillo et al. [CCWB18], and to our method. For the Criminisi algorithm, 19 restorations were performed on each image using different patch sizes ranging from 2 to 20. The restorations with the amoeba method were performed using the maximum amoeba distance \( TH \) set to 20 and a physical distance \( PD = 1 \). We tested different radii ranging from 1 to 10 for each image. For our method, the maximal cumulative distance \( \beta \) was set to 0.15 in the \( LAB \) color space. For each image, we carried out 20 reconstructions with a random value of \( \lambda \) selected between 0.01 and 0.1. The context was a \( 3 \times \lambda \times W \times 3 \times \lambda \times H \) rectangle centered on the border pixel with the highest priority (\( W \) and \( H \) are the width and the height of the image, respectively). Fig 8 shows images of the benchmark.

4.1. Results and validation

The main goal of inpainting algorithms is to improve image filling quality so that any imperfection is not noticeable by a person who is not familiar with the original image. The ideal is to let the results be appreciated by human observers in order to judge the performance of the algorithms. To get around this tedious task, we used three metrics: Peak Signal-to-Noise Ratio (PSNR) [MM98], Edge Histogram (EH) [WPP02] and Structural Similarity Index (SSIM) [WB02].

Peak Signal-to-Noise Ratio (PSNR) is a measure of distortion used in image processing to quantify the performance of image restoration or compression algorithms. It is an estimate of the restored image quality compared with the original image. Let \( A \) be a restored image and \( B \) the corresponding original image, both with size \( M \times N \).

\[
PSNR(A, B) = 10 \log \left( \frac{S^2}{MSE(A, B)} \right)
\]  

where \( S = 255 \) for an 8-bit image and

\[
MSE(A, B) = \frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j=1}^{N} [A(i, j) - B(i, j)]^2
\]

Edge Histogram: The EH descriptor represents the distribution of 5 types of edges (vertical, horizontal, 45-degree diagonal, 135-degree diagonal, and non-directional edges) in each local area called a sub-image. The image is divided into a grid of 4 \( \times \) 4 blocks. Whatever the size of the image, the final descriptor is an edge histogram with 150 bins (80 bins (local) + 5 bins (global) + 65 bins
Let A and B be two images, the distance $D(A, B)$ between their edge histogram can be measured by

$$
D(A, B) = \sum_{i=0}^{29} |(\text{Local}_A[i] - \text{Local}_B[i])| + 5 \times \sum_{i=0}^{4} |(\text{Global}_A[i] - \text{Global}_B[i])| + \sum_{i=0}^{64} |(\text{Semi}_A[i] - \text{Semi}_B[i])|
$$

Structural Similarity Index (SSIM) is a human visual system (HVS) based metrics introduced by Wang and Bovik [WB02] to assess the human visibility similarity between a restored image and the original. SSIM measures the similarity of the combination of contrast and luminance [ANC12]. Let A and B be the original image and the restored one. Both images are first divided into blocks of size $8 \times 8$ and converted into vectors. Let $x = \{x_1, x_2, \ldots, x_T\}$ and $y = \{y_1, y_2, \ldots, y_T\}$ be two corresponding vectors from A and B respectively:

$$
\text{SSIM}(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}
$$

where $C_1$ and $C_2$ are constants. $\mu_x$, $\mu_y$ are the mean values of the vectors $x$ and $y$. $\sigma_x^2$ and $\sigma_y^2$ the variances and, $\sigma_{xy}$ the covariance between $x$ and $y$.

Let $L$ be the number of local windows over the images. So $A = \{A_1, A_2, \ldots, A_L\}$ and $B = \{B_1, B_2, \ldots, B_L\}$ then,

$$
\text{SSIM}(A, B) = \frac{1}{L} \sum_{i=1}^{L} \text{SSIM}(A_i, B_i)
$$

Table 1 presents the values of parameters that produce the best result for each image of the benchmark for each of the three metrics. Since EH reflects the distance between the original image and the inpainted one, the optimal restoration is the result that generates...
Table 1: Best metric values obtained by the methods are presented in bold font: The parameters that produce the best results are indicated in the brackets (best radii for Criminisi and Amoeba and, λ × 10^3 for our method). For the EH metric, the smaller values indicate better reconstructions while for PSNR and SSIM larger values indicate better reconstructions.

<table>
<thead>
<tr>
<th>Image</th>
<th>Size (W × H)</th>
<th>Criminisi PSNR</th>
<th>Criminisi SSIM</th>
<th>Criminisi EH</th>
<th>Amoeba PSNR</th>
<th>Amoeba SSIM</th>
<th>Amoeba EH</th>
<th>Our method PSNR</th>
<th>Our method SSIM</th>
<th>Our method EH</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bicycle</td>
<td>460 × 300</td>
<td>24.13(10)</td>
<td>0.84(3)</td>
<td>18.71(9)</td>
<td>24.5(5)</td>
<td>0.95(4)</td>
<td>14.62(11)</td>
<td>24.69(15)</td>
<td>0.96(30)</td>
<td>15.61(25)</td>
</tr>
<tr>
<td>Twobirds</td>
<td>600 × 450</td>
<td>16.45(5)</td>
<td>0.7(1)</td>
<td>39.77(16)</td>
<td>17.25(12)</td>
<td>0.75(1)</td>
<td>27.27(12)</td>
<td>22.41(25)</td>
<td>0.94(25)</td>
<td>25.78(25)</td>
</tr>
<tr>
<td>BattleShip</td>
<td>1024 × 756</td>
<td>22.16(11)</td>
<td>0.85(6)</td>
<td>46.01(13)</td>
<td>23.02(8)</td>
<td>0.87(4)</td>
<td>44.20(8)</td>
<td>23.4(45)</td>
<td>0.84(15)</td>
<td>47.91(45)</td>
</tr>
<tr>
<td>Blueman</td>
<td>1024 × 681</td>
<td>18.5(7)</td>
<td>0.91(2)</td>
<td>17.53(9)</td>
<td>18.67(11)</td>
<td>0.94(6)</td>
<td>18.25(16)</td>
<td>19.02(30)</td>
<td>0.95(30)</td>
<td>17.98(15)</td>
</tr>
<tr>
<td>BrickHouse</td>
<td>1024 × 683</td>
<td>24.43(15)</td>
<td>0.89(5)</td>
<td>31.86(13)</td>
<td>24.28(6)</td>
<td>0.84(8)</td>
<td>38.55(6)</td>
<td>24.5(35)</td>
<td>0.91(35)</td>
<td>31.82(35)</td>
</tr>
<tr>
<td>Bungee</td>
<td>206 × 308</td>
<td>17.06(4)</td>
<td>0.80(8)</td>
<td>38.41(5)</td>
<td>16.78(19)</td>
<td>0.81(20)</td>
<td>35.06(19)</td>
<td>17.08(25)</td>
<td>0.82(25)</td>
<td>29.78(25)</td>
</tr>
<tr>
<td>Castle</td>
<td>1024 × 768</td>
<td>23.87(9)</td>
<td>0.91(11)</td>
<td>33.72(9)</td>
<td>23.89(13)</td>
<td>0.94(9)</td>
<td>39.24(2)</td>
<td>23.84(40)</td>
<td>0.97(35)</td>
<td>32.80(60)</td>
</tr>
<tr>
<td>Cat</td>
<td>1024 × 683</td>
<td>23.2(18)</td>
<td>0.97(9)</td>
<td>42.74(15)</td>
<td>23.64(3)</td>
<td>0.98(1)</td>
<td>35.48(14)</td>
<td>23.59(10)</td>
<td>0.98(25)</td>
<td>38.75(25)</td>
</tr>
<tr>
<td>Child</td>
<td>1024 × 680</td>
<td>25.63(5)</td>
<td>0.95(13)</td>
<td>30.85(16)</td>
<td>25.73(13)</td>
<td>0.97(1)</td>
<td>28.29(2)</td>
<td>25.64(25)</td>
<td>0.96(55)</td>
<td>29.68(20)</td>
</tr>
<tr>
<td>Eagle</td>
<td>600 × 402</td>
<td>28.57(20)</td>
<td>0.94(2)</td>
<td>27.26(17)</td>
<td>28.46(7)</td>
<td>0.95(1)</td>
<td>19.91(20)</td>
<td>29.36(15)</td>
<td>0.95(10)</td>
<td>29.78(55)</td>
</tr>
<tr>
<td>fish</td>
<td>1024 × 768</td>
<td>24.14(5)</td>
<td>0.84(15)</td>
<td>30.51(16)</td>
<td>24.16(2)</td>
<td>0.91(6)</td>
<td>28.29(2)</td>
<td>24.5(30)</td>
<td>0.91(10)</td>
<td>30.66(25)</td>
</tr>
<tr>
<td>matrioska</td>
<td>1440 × 971</td>
<td>19.56(13)</td>
<td>0.74(11)</td>
<td>27.69(16)</td>
<td>19.58(7)</td>
<td>0.89(1)</td>
<td>25.84(2)</td>
<td>19.34(45)</td>
<td>0.85(15)</td>
<td>29.43(30)</td>
</tr>
<tr>
<td>mochizuki</td>
<td>547 × 346</td>
<td>24.04(18)</td>
<td>0.89(1)</td>
<td>19.89(19)</td>
<td>22.47(9)</td>
<td>0.82(1)</td>
<td>16.71(17)</td>
<td>25.27(10)</td>
<td>0.94(50)</td>
<td>18.80(25)</td>
</tr>
<tr>
<td>mountains</td>
<td>512 × 683</td>
<td>27.53(9)</td>
<td>0.94(17)</td>
<td>8.32(15)</td>
<td>28.81(17)</td>
<td>0.97(15)</td>
<td>10.43(4)</td>
<td>28.5(15)</td>
<td>0.97(20)</td>
<td>9.80(15)</td>
</tr>
<tr>
<td>penguins</td>
<td>615 × 461</td>
<td>19.71(13)</td>
<td>0.86(15)</td>
<td>18.96(19)</td>
<td>19.31(11)</td>
<td>0.98(1)</td>
<td>18.11(1)</td>
<td>19.61(10)</td>
<td>0.92(25)</td>
<td>17.71(15)</td>
</tr>
<tr>
<td>car</td>
<td>500 × 375</td>
<td>24.08(17)</td>
<td>0.91(3)</td>
<td>8.34(3)</td>
<td>24.19(8)</td>
<td>0.95(2)</td>
<td>10.44(3)</td>
<td>25.06(15)</td>
<td>0.96(10)</td>
<td>8.32(15)</td>
</tr>
</tbody>
</table>

Table 2: Average values of metrics over all of our experiments.

<table>
<thead>
<tr>
<th></th>
<th>Criminisi PSNR</th>
<th>Amoeba PSNR</th>
<th>Our method PSNR</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSNR</td>
<td>22.69</td>
<td>22.79</td>
<td>23.48</td>
</tr>
<tr>
<td>SSIM</td>
<td>0.87</td>
<td>0.90</td>
<td>0.92</td>
</tr>
<tr>
<td>EH</td>
<td>27.53</td>
<td>25.71</td>
<td>25.91</td>
</tr>
</tbody>
</table>

Obtained for each of the metrics on the benchmark is presented in Table 2. We note that our method wins over the two others according to the PSNR and SSIM metrics. However, the amoeba method is slightly better for the EH metric. In general, these results reflect an improvement in the restoration quality obtained with our method compared to those of Amoeba and Criminisi.

The best results (subjective judgment) generated by each of the three methods for some images are presented in Figure 9. These results do not correspond to those indicated by the metrics in all cases. We can see that the results obtained by our method are the most plausible in the majority of cases.

4.2. Conclusion

A new exemplar-based inpainting method was presented in this work. Based on Criminisi’s pipeline, our method proposes several improvements at each stage of the restoration process. A new priority function was defined. To give a high order priority to the patches constructed from the pixels located on the edges, we used the ability of color segmentation algorithms to subdivide images to determine the patches covering the largest number of texture blocks. This approach allowed for a considerable improvement in the reconstruction of continuous structures in the image. Unlike in most exemplar-based inpainting methods where patches are usually defined by regular structures (square, rectangle, ...), in the new method, the patches are built using a region-growing algorithm in the different texture blocks surrounding the border pixel with the highest priority. The search for the best match is done contextually. We look for the best match in a neighborhood similar to the one surrounding the patch around the target area.

To demonstrate the effectiveness of our method we used a benchmark of 16 images. A comparison of our results with those obtained by the Criminisi and Amoeba algorithms shows a clear improvement in the quality of the restoration obtained by our method.
results have fewer inconsistent artifacts, and the restored images are more plausible.

The innovations proposed in this research (the priority function, the construction of the patch, and the search for the best match) can be used in any other modular version of the exemplar-based inpainting method based on the Criminisi pipeline. Although having a complexity close to that of the basic Criminisi algorithm, a naïve implementation of our method can quickly increase the time of the restoration considerably.
Acknowledgments

The authors would like to thank all the TikiLab members for their support.

References


C. Foting & D. Cunningham / Context Aware Exemplar-based Image Inpainting using Irregular Patches


