

Constrained Texture Restoration

A. Rares

Information and Communication Theory Group, Mediamatics Department, Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology, 2628 CD Delft, The Netherlands

Division of Image Processing, Radiology Department, Leiden University Medical Center, 2333 ZA Leiden, The Netherlands
Email: a.rares@lumc.nl

M. J. T. Reinders

Information and Communication Theory Group, Mediamatics Department, Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology, 2628 CD Delft, The Netherlands

Email: m.j.t.reinders@ewi.tudelft.nl

J. Biemond

Information and Communication Theory Group, Mediamatics Department, Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology, 2628 CD Delft, The Netherlands

Email: j.biemond@ewi.tudelft.nl

Received 11 December 2004; Revised 19 May 2005; Recommended for Publication by Mauro Barni

A method is proposed for filling in missing areas of degraded images through explicit structure reconstruction, followed by texture synthesis. The structure being reconstructed represents meaningful edges from the image, which are traced inside the artefact. The structure reconstruction step relies on different properties of the edges touching the artefact and of the areas between them, in order to sketch the missing edges within the artefact area. The texture synthesis step is based on Markov random fields and is constrained by the traced edges in order to preserve both the shape and the appearance of the various regions in the image. The novelty of our contribution concerns constraining the texture synthesis, which proves to give results superior to the original texture synthesis alone, or to the smoothness-preserving structure-based restoration.

Keywords and phrases: image restoration, edge structure reconstruction, sequentiality, nonparametric texture synthesis.

1. INTRODUCTION

This paper addresses the problem of image restoration in situations where areas of the image are completely missing. This type of information loss takes place in old photographs, films, drawings, paintings, and so forth [1, 2, 3]. The problem of information loss does not affect old media only. Even the most recent types of media are affected by similar problems. Packet losses during the transmission of streaming digital video (especially live broadcasts, or other broadcasts where no retransmission is possible) result in corrupted image areas, or even the complete loss of one or more consecutive frames [4]. The visual appearance of these errors can be quite disturbing.

The work presented here was carried out in the context of archived film restoration. Film restoration is usually

performed by means of temporal or spatiotemporal algorithms. However, when pathological motion occurs (i.e., when the objects in the image are difficult to track), the temporal restoration algorithms fail due to inaccurate motion vectors. Several solutions have been proposed [5, 6, 7], which discard the temporal information when pathological motion is detected, since this type of information is unreliable. This paper continues the aforementioned restoration efforts and represents a solution that also uses only spatial information during the restoration.

In spatial restoration, the missing image information has to be recovered using the remaining valid portions of the image. The approaches taken so far for solving the problem at hand can basically be classified into two categories: smoothness-preserving algorithms and texture synthesis ones.

The algorithms belonging to the first category range from simple isophote connection [8] to PDE-based or variational inpainting models using isophotes, gradients, curvatures, [9, 10, 11, 12, 13, 14, 15, 16, 17] normalized convolution [18],

or inpainting guided by explicitly sketched edges [7, 19, 20, 21]. Although the term “inpainting” was mostly used in association with the aforementioned PDE-based or variational methods, in the following we will use the terms “inpainting,” “restoration,” and “interpolation” interchangeably.

The second category comprises a number of different approaches for texture synthesis. The algorithms presented in [5, 22, 23] are nonparametric and are based on Markov random fields (MRF), while the algorithm presented in [24] is parametric and is based on a Bayesian, 2D autoregressive model. The approach presented in [25] uses partial differential equations and Gabor filters, and in [26], spatial and frequency information is combined in a framework of projection onto convex sets.

Smoothness-preserving algorithms are good at reconstructing piecewise flat (or relatively flat) areas, but they do not reconstruct texture satisfactorily. On the other hand, the texture synthesis methods tend to neglect the image structure. While the appearance of the restored texture may be pleasant, the object edges suffer sometimes deformations that may be visually disturbing. To overcome these drawbacks, some first algorithms for hybrid structure and texture interpolation are reported in [27, 28, 29, 30], in which various approaches for combined structure and texture interpolation are used. The approach presented in [27] by Rane et al. classifies complete 8×8 artefact blocks either as flat or texture areas, based on their surrounding uncorrupted blocks. The blocks classified as flat areas are restored using the smoothness-preserving inpainting method described in [12], while the textured blocks are restored using the texture synthesis algorithm described in [22]. In [28], Bertalmio et al. approach the problem differently by decomposing the image into two completely overlapping images of the same size, one with no texture, and one containing only texture, whose sum represents the original image. These are then restored in parallel, using the same aforementioned algorithms for inpainting and texture synthesis, and then combined back into one image by simply adding them up. In [29, 30], Jia and Tang describe a novel technique based on tensors. Here, edge structure is first reconstructed, followed by texture synthesis. Both steps use adaptive tensor voting.

All the aforementioned combining approaches have advantages and disadvantages. While the first one benefits from the distinction between different parts of the image, it is hampered by the fact that the structure/texture separation is done on a block basis. The second one benefits from the power of handling smooth texture changes (especially shadows), but it may fail when the structures (i.e., the prominent edges) resulting from the two parallel interpolations are different. The third approach benefits from the power of tensor-based texture reconstruction, but the edge pairing approach is rather basic.

We have opted in our approach for a hybrid, structure and texture restoration, in which different areas of the artefact are restored independently by means of texture synthesis. The structure reconstruction part of the algorithm is based on the one we proposed in [7], while for the texture

restoration part we use a modified version of the algorithm described in [5]. The advantage of the new approach over these two methods lies in the fact that the algorithm described in [7] cannot handle textural contents, while the algorithm described in [5] does not preserve edge smoothness generally and cannot handle overlapped structures. The proposed method is able to reproduce texture, while preserving the smoothness of object edges, and has the ability to deal with crossing structures, if needed.

In the algorithm that we propose, we separate different areas of the artefact and interpolate them independently, so we do not run the risk of “double edges,” as in [28]. Moreover, our approach is superior to the one presented in [27] because it explicitly constructs the image skeleton (i.e., the object edges) and then uses it to *constrain* the texture restoration in a precise manner during the reconstruction step. The constrained texture synthesis approach is preferable to a simple one, since the latter does not use superior knowledge for preserving the object boundaries, thus being prone to “spilling texture” over the edges. Our method also has the advantage that it is not confined to square artefacts, and it is actually able to adaptively split them into smaller pieces, according to the recovered structure. The algorithm we propose also benefits from an edge connection scheme which is more elaborate than the one used in [29, 30], being thus capable of recovering more complex structures.

1.1. Algorithm overview

The spatial restoration algorithm that we propose consists of three main steps, depicted in Figure 1:

- (1) segmentation and feature extraction;
- (2) structure reconstruction;
- (3) texture synthesis.

The input to our algorithm is an image and an artefact mask.¹ Without loss of generality, in the remainder of this paper we consider that the mask consists of only one artefact. The general assumption that we make is that there is enough redundant information in the input image, outside the artefact, which allows us to restore what is missing inside it.

In the first step (Section 2), a simple segmentation takes place which separates areas with distinct properties. Ideally, this results in a separate mask for each object. The second step of the algorithm (Section 3) builds the structure of the image inside the artefact. Hereto, we try to reconstruct the missing object boundaries inside the artefact (see the dashed lines in Figure 1(d)) as continuations of the object boundaries from outside the artefact. The outer boundaries separate the areas segmented in the first step. Finally, in the third step (Section 4), the reconstructed structure is used to guide a texture synthesis procedure which fills in the artefact area. Here, a key aspect of our method contributes to its strength. Namely, the sampling neighborhood for a certain

¹We assume that the artefact mask is detected by another algorithm.

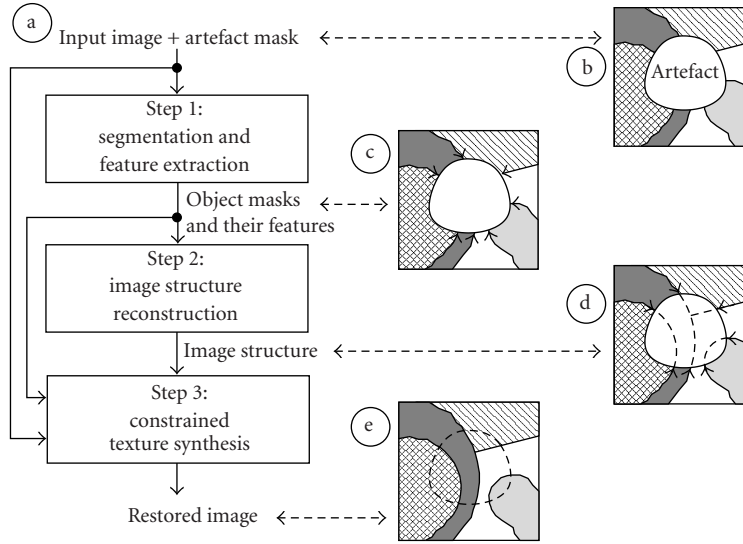


FIGURE 1: General algorithm outline (left) and an illustration of the inputs/outputs for each stage (right). (a) Algorithm steps; (b) artefact mask; (c) detected object mask and extracted boundaries (thick arrows); (d) reconstructed structure inside the artefact (dashed lines); (e) restored artefact using constrained texture synthesis.

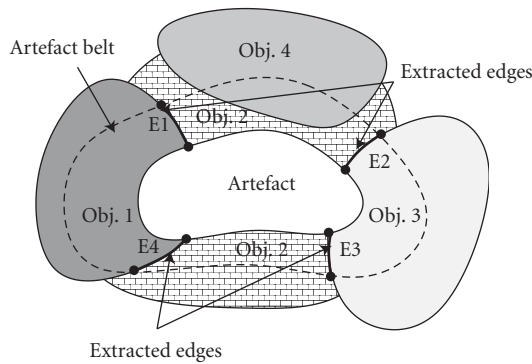


FIGURE 2: Objects lying close to the artefact may influence the feature values.

(reconstructed) area inside the artefact is taken *only* from areas around the artefact that are belonging to the same object mask.

2. SEGMENTATION AND EXTRACTION OF OBJECT EDGE FEATURES

The segmentation procedure is an essential step which allows us to identify distinct areas around the artefact. We have used the segmentation algorithm described in [31]. Ideally, we should be able to distinguish between all objects within an image. This is, however, not a trivial task, and no current algorithm is able to output a segmentation mask which is very similar to the ground truth (i.e., a manually segmented image), except in some restricted contexts. This is a limitation we have assumed for our current experiments. In the

following, we consider that the segmentation mask we get is accurate enough.

Once the segmentation is performed, the edges which separate different objects are extracted. In addition, for every area between consecutive edges, a number of characteristic features are computed. These features will help us later to identify pairs of edges with similar properties (e.g., pairs $E_1 - E_4$ and $E_2 - E_3$ in Figure 2). These pairs of edges originate from the same object border, after a part of it was occluded by the artefact. The segments are considered up to a certain distance from the artefact (20 pixels in our case), forming a *belt* around the artefact.

The features that describe the segments are the intensity histogram, measured on the original image intensity, as well as the gradient angle histogram (in radians) within these segments, measured on the image gradient. The gradient angle histogram shows the dominant gradients and is calculated from the angle of each gradient vector, as shown in Figure 3. The angle interval $[0, \dots, 2\pi]$ is divided into N "pie slices," corresponding to the N bins of the histogram. Each gradient vector contributes with *one* unit to the bin that corresponds to its angle. Since each gradient vector contributes with only one unit to the histogram, the gradient magnitude information is discarded, making the histogram insensitive to contrast changes. One needs to pay attention to the value of N . If N is too small, the bins are too coarse, covering a wide angle range and giving rise later to false histogram matches. If N is too big, the histograms may become sparse, giving rise to false histogram mismatches. In our experiments, N was chosen to be 100. However, one should adapt this number to the situation at hand, since histogram sparsity may also arise if N is too big with respect to the numbers of pixels belonging to the segments.

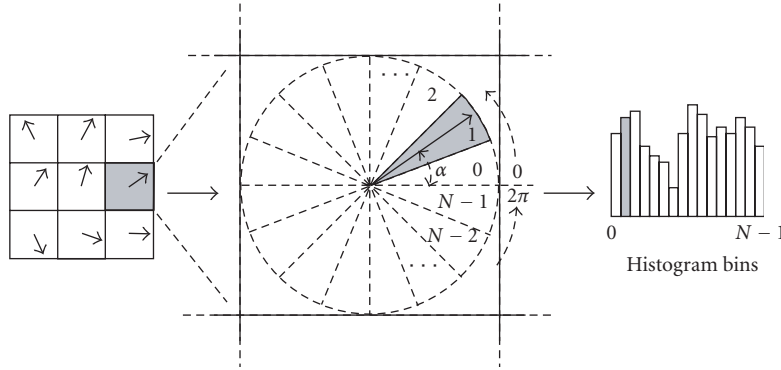


FIGURE 3: The gradient angle histogram is calculated from the gradient vectors. The angle interval $[0, \dots, 2\pi]$ is divided into N “pie slices,” and each gradient vector contributes with one unit to the bin corresponding to its angle.

The feature that describes the likelihood that two object edges belong to the same object boundary is calculated by measuring how well they fit the same circle. We rely here on the observation that most object edges have a rather constant curvature, locally.

The extracted histograms may get disturbed by various factors. For example, if an object was missed during the segmentation process and lies close to the artefact (such as object 4 in Figure 2), then the histogram of the corresponding segment (e.g., the upper part of object 2) may display significant perturbations, resulting later on in histogram mismatches. Other examples of feature perturbations include the presence of shading along an object, the bending of textured objects (which affects the gradient angle histograms), and so forth. In all these cases, those parts of the segments (from the same objects) which lie close to the artefact will have histograms more similar to each other than those parts which lie further away from the artefact. For this reason, we calculated weighted histograms, instead of normal ones. As such, a pixel contributes to a histogram bin with a quantity given by a Gaussian weight proportional with the distance to the closest artefact point (the standard deviation used is the belt width). In the end, the weighted histograms are normalized by the sum of the histogram bins. All these operations are applied in order to reduce histogram discrepancies between similar segments with different numbers of pixels and to avoid the influence of noninformative areas.

At this point it is important to mention that we do not work with segments in our structure reconstruction step—rather, with the edges which separate them. Accordingly, each edge will be described by five features: four individual features and a shared one. The individual features are the histograms of the intensity and gradient angles, on both the right and left sides of the edge. The fifth, shared feature, is the shape fitting cost.

Additionally, there is a sixth feature, *sequentiality*, which is not calculated only for edges (or edge pairs). Rather, it is calculated for the entire configuration of edge connections. For every potential configuration that we tentatively construct, this feature will measure the degree to which the edge

connections in the configuration at hand lie in consecutive order. Alternatively, it may be seen as the degree to which the edge connections do not cross each other. Since this feature is only computed after pairs of edges have been connected to each other, it will be described later in this paper.

3. IMAGE STRUCTURE RECONSTRUCTION

The structure reconstruction step is crucial to our proposed restoration scheme, since the explicit image structure that is recovered represents the “skeleton” of the restoration process. The input to this step represents a list of *edges* coming into the artefact, in clockwise order. The output of this step will be a list of *edge couples* arranged in *groups* of couples, and a list of *spare edges*. An *edge couple* is composed of two edges that are considered to represent the same object border. These two edges will be connected in order to recover the object border inside the artefact. A *group* of couples is a collection of edge couples which lie in strict consecutive order, that is, they do not cross each other, and no edge from a couple that does not belong to the current group lies between two couples of this group. The groups will be formed after the edge couples are constructed. The *spare edges* are edges for which there was no other edge that could be matched. Hence, it is assumed that a spare edge represents a *T-junction* (e.g., the upper right edge in Figure 1(d)) or a *fading edge* (an edge that gradually dissolves).

The procedure for reconstructing the image structure inside the artefact is based upon our approach presented in [7]. However, in the current approach, the feature description of the edges (in particular the histogram features) is different. We briefly describe here the procedure and the cost functions involved, but more details can be found in [7].

The basic idea is that we build several particular configurations (edge couples, groups, and spare edges), test how well they match the measured features, and then select the one which scores the best. The overall cost c^{cfg} of a particular configuration is computed from the five features shared by the two edges within each edge couple (c^{cp}), and from the additional feature characterizing a global property of

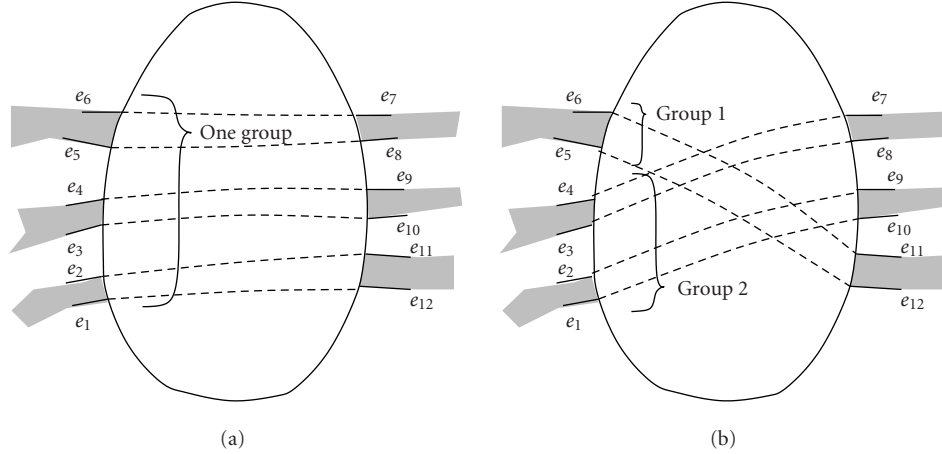


FIGURE 4: Contribution of the *sequentiality* parameter. (a) Configuration given preference by the sequentiality parameter; (b) configuration penalized by the sequentiality parameter.

the configuration (c^{seq}):

$$c^{\text{cfg}}(\mathbf{Z}) = \frac{5 \times c^{\text{cpl}}(\mathbf{Z}) + c^{\text{seq}}(\mathbf{Z})}{6}, \quad (1)$$

where \mathbf{Z} represents the configuration of (groups of) edge couples. c^{seq} is the cost associated with the *sequentiality* feature, defined later in this section. The cost c^{cpl} is defined in (2) and expresses how well the two edges in every couple match each other, that is, how smoothly they can be continued into each other, and how similar are the regions they separate. All aforementioned costs (c^{cfg} , c^{cpl} and c^{seq}) have values between 0 and 1, with 0 indicating a perfect match and 1 indicating a complete mismatch. c^{cpl} contributes with 5/6 of the final configuration score c^{cfg} because it is computed from 5 features. For the same reason, c^{seq} contributes with 1/6 of c^{cfg} .

The couple-related cost is calculated as follows:

$$c^{\text{cpl}}(\mathbf{Z}) = \sum_i \sqrt{\frac{\beta_i^C (\chi_i^{\lambda,C} + \chi_i^{\gamma,C}) + \beta_i^A (\chi_i^{\lambda,A} + \chi_i^{\gamma,A}) + \omega_i^2}{2 * \beta_i^C + 2 * \beta_i^A + 1}}, \quad (2)$$

where χ_i is the mean square error (MSE) operation applied on the difference of the histograms belonging to the two edges in edge couple \mathbf{q}_i . This histogram difference is a simple one-to-one subtraction of the arrays holding the histograms. Since all histograms are normalized to sum up to 1, the χ_i 's always return values between 0 and 1. Each χ_i is calculated for a certain feature (λ for the histogram of intensities, and γ for the histogram of gradient angles) and on a certain side of the edge (C for the clockwise side, and A for the anticlockwise side, these sides being considered with respect to only one of the two edges in the couple). If the next edge on a certain side of a couple is a *spare edge* (see Figure 1), or if it belongs to an edge couple which intersects the current couple (see Figure 4), then the shared features on that side of the current

couple become irrelevant. For this reason, we have used the binary flags $\beta_i \in \{0, 1\}$ to switch off cost contributions of the respective features, when needed.

ω_i is the cost of fitting a certain shape to both edges of couple \mathbf{q}_i and returns values between 0 and 1. We have chosen to fit circles because we assume that object boundaries have constant curvature, locally. ω_i is computed based on three measurements: the spatial deviation of the edge pixels from the fitted circle; the angular proximity of the points where the two edges touch the artefact (computed with respect to the center of the fitted circle); and a formula that checks the spatial order of edge pixels, making sure that the two edges stretch in opposite directions with respect to the artefact (see [7] for more details).

In both formulas from (1) and (2) we combined the feature costs through addition rather than multiplication for several reasons. First of all, in case of multiplication, a feature cost that is close to zero would cancel the contribution of all other feature costs. This is not a desirable behavior, since these feature costs cannot describe any possible edge configuration. Rather, they are chosen to describe configurations that are encountered most often. In case of a valid configuration that is not properly modeled by one of our feature costs, the final cost should not be influenced too much by the faulty feature cost. Even when the edge configuration is properly modeled by the chosen features, the cost calculations may have imperfections (e.g., due to faulty segmentation masks). As such, some costs may again get lower than normal, which could strongly influence the final configuration score if multiplication was used instead of addition.

The sequentiality cost c^{seq} tries to measure a special property of object edges around the artefacts. Namely, for each edge that touches an artefact on one side, there is usually a second edge on the other side, belonging to the same object. Moreover, consecutive objects (i.e., object margins) which are occluded by the same artefact give rise to consecutive,

- (1) Calculate c^{pl} for all possible edge pairs with (2) by setting all β flags to 1 (at this point we do not know which edges are spare ones).
- (2) Eliminate from further consideration those couples whose costs c^{pl} are too big.
- (3) FOR each subset of couples from the remaining ones, and FOR each couple from the current subset, build potential Z_j in a greedy fashion:
 - (a) Start a new group with the current couple.
 - (b) Incrementally add neighboring couples to the group, on both sides of the initial couple, until one reaches a couple that intersects any couple from the current group.
 - (c) Start another group with the intersecting couple and add again neighboring couples from the remaining ones.
 - (d) Repeat the above steps until no more couples are left.
 - (e) All remaining edges are considered spare edges.
- (4) Choose the configuration with the minimal cost: $Z = Z_k, k = \arg \min_j c^{\text{fg}}(Z_j)$ (the β flags are now enabled according to the existing spare edges).

PSEUDOCODE 1: Pseudocode for the structure reconstruction procedure.

nonoverlapping edge couples (see Figure 4a). c^{seq} returns a cost of 0 when a maximum number of couples is formed with a minimum number of groups (i.e., one group), and a gradually increasing cost for less “sequential” configurations. The sequentiality feature is a very useful property of the edges around artefacts, since it is extremely robust against noisy data.

It is worth pointing out that the sequentiality parameter does not forbid a configuration containing crossing edges—rather, it penalizes it. If the evidence coming from the other features strongly indicates a crossing, the edge couples are formed accordingly (resulting in a configuration such as the one in Figure 4b).

Three problems arise when determining the sequentiality of a configuration. First, we must find a way to express it as a number. Secondly, despite the fact that it is used to calculate the configuration cost, we can measure it only *after* the configuration of edge couples has been formed. Thirdly, the sequentiality does not represent a measurement of each edge couple alone—rather, it is a measurement of the complete configuration, which is an ensemble of edge couples.

The sequentiality of a configuration Z was defined as follows:

$$c^{\text{seq}}(Z) = \begin{cases} 1 - \frac{\sum_{i=1}^{N^G} (\|G_i\| - 1)}{[N^E/2] - 1}, & \left\lfloor \frac{N^E}{2} \right\rfloor \geq 2, \sum_{i=1}^{N^G} \|G_i\| \geq 1, \\ 0, & \left\lfloor \frac{N^E}{2} \right\rfloor = 1, \sum_{i=1}^{N^G} \|G_i\| = 1, \\ 1, & \text{otherwise,} \end{cases} \quad (3)$$

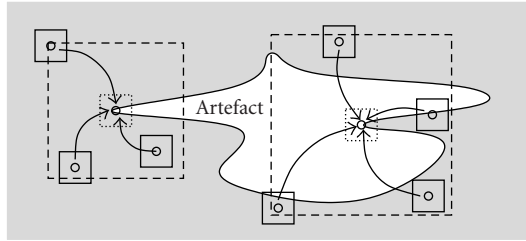
where $G_i, i = 1, \dots, N^G$, are the groups of edge couples in configuration Z , $\|G_i\|$ is the number of edge couples in group G_i , and $[N^E/2]$ represents the maximum number of edge couples that can be achieved out of the N^E edges detected around the artefact.

The algorithm for building the final configuration of groups of edge couples is described in Pseudocode 1. The

computational complexity of this structure reconstruction step is (in a worst-case scenario) $O(2^n \times n^3)$, since, in step 3 from the pseudocode, trying every subset from the selected edges is $O(2^n)$, then building a different configuration starting from every edge is $O(n)$, and building every single configuration is $O(n^2)$ (adding couples to the current group is $O(n)$, and the step of checking couple crossing for each newly added couple is also $O(n)$, assuming that the amount of couples in the current subset is $O(n)$). Although the complexity of this step may seem high at first glance, in reality, however, the computational demands can be reduced drastically. We severely eliminate unlikely edge couples right from the beginning. As a result, only a fraction of the possible couples are left for each edge, in general. This reduces the search space for making subsets of couples and for checking couple crossing. Additionally, it is not necessary to check all crossings. We stop at the first crossing that is found. Finally, one should limit the maximal number of initial edges (to about 15) because generally only the main structure within the artefact area needs to be reconstructed.

To reconstruct the missing structure, all edge couples are traced inside the artefact according to the circle fitted to the couple. Finally, the spare edges are continued as straight lines within the artefact area (possibly stopping into an already reconstructed object boundary inside the artefact). Spare edges are traced as straight lines because circle fitting proved to be much less reliable for single edges, as opposed to edge couples. All contiguous areas lying between consecutive sketched edges, inside and outside the artefact, are distinctly labeled, resulting in disjoint masks of objects.

In the case where we have more groups of edge couples (i.e., crossing edge couples as in Figure 4b), each group is reconstructed separately. We have to assume, however, that one group lies in front of the others. Since the information extracted so far provides no guidelines as to which one is in the front and which one in the background, this choice is made arbitrarily. Only groups consisting of a single edge couple (e.g., a horizon line) are “pushed” to the background, since their reconstruction in the foreground may obliterate all other groups.






-  Pixel to be interpolated and its neighborhood
-  Search area
-  Sampling pixel and its neighborhood

FIGURE 5: Search for candidates during texture synthesis.

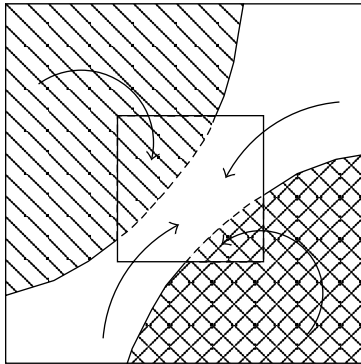


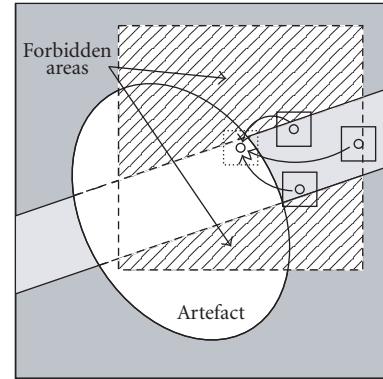
FIGURE 6: Key idea for the constrained texture synthesis: the artefact pixels are interpolated based only on pixels from the same object mask.

4. TEXTURAL INPAINTING

4.1. Texture synthesis

The structure reconstruction step builds only a virtual sketch of the missing areas. The process of filling these areas must take care that object appearance inside the artefact area is the same as outside it. This will be achieved by means of texture synthesis, constrained by the current edge configuration.

We have chosen to apply the texture reconstruction algorithm of Bornard et al. presented in [5], modified as described in Section 4.2. This algorithm is able to reconstruct both regular and irregular textures. It stems from Efros and Leung's algorithm for texture reconstruction [22]. In contrast to Efros and Leung's algorithm, the algorithm of Bornard et al. is based on nonstationary Markov random fields (MRF) and imposes coherence constraints on the texture interpolation.






-  Pixel to be interpolated and its neighborhood
-  Search area
-  Sampling pixel and its neighborhood

FIGURE 7: Texture interpolation near the object edges. Preference will be implicitly given for the sampling pixels lying next to the same edge (e.g., the top candidate).

The underlying statistical model of the algorithm assumes that each pixel's probability distribution function (PDF) is independent of the rest of the image. The neighborhood of the pixel is assumed—for the time being—to be a square window centered around the pixel which must be synthesized. The algorithm is nonparametric in the sense that the PDF is not imposed, nor constructed explicitly. Rather, it is approximated from a sample region of the image which has the same size as the aforementioned pixel neighborhood. It should be large enough to capture the texture characteristics, but not too large, or else we run the danger of falling out of the textured area.

The algorithm goes as follows. For each artefact pixel, a search is started in an area around that pixel (we used a search area size of 41×41 pixels in our experiments) in order to find valid pixels (nonartefact pixels, or pixels which were already interpolated) whose neighborhoods are similar enough to the neighborhood of the current artefact pixel (our choice for neighborhood size was again 41×41 pixels). Out of the set of candidates that were found, one is randomly drawn and its value is pasted into the current artefact pixel (see Figure 5). When computing the neighborhood similarity, the L^2 norm is used. Since pixels in the closer vicinity are more relevant than pixels lying further away, Gaussian weights are imposed upon calculating the norm. The artefact pixels that have not been synthesized yet and happen to lie in the neighborhood windows are not taken into account for the norm calculation.

The aforementioned nonstationarity of the MRF is modeled by using an adaptive window size [5], rather than a fixed

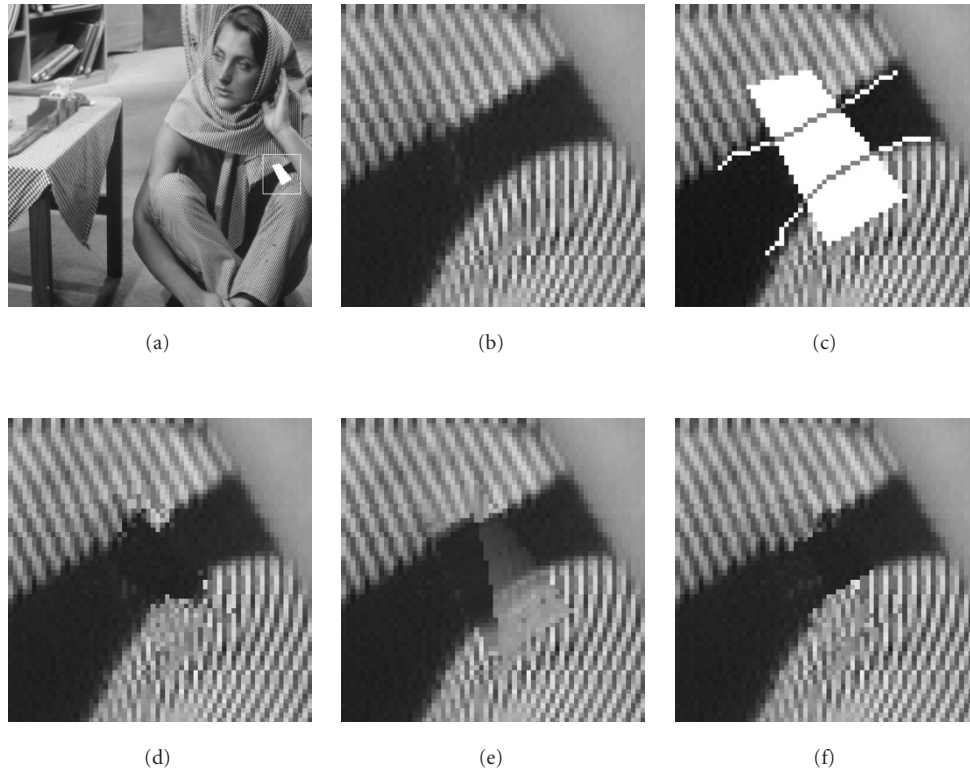


FIGURE 8: Comparison between different types of restoration (all images are zoom-ins on the artefact area, except the first one). (a) Original image with artefact overlay; (b) original artefact contents; (c) structure reconstructed in the proposed method; (d) texture-only restoration; (e) smooth restoration based on edges; (f) proposed method.

size (as used in Efros and Leung’s approach [22]). Whenever there are less than a given number of valid pixels in the neighborhood of the artefact pixel (1500 pixels in our case), the neighborhood is enlarged until we have the requested number of pixels. As such, each pixel can have different conditional PDFs.

A second improvement introduced by Bornard et al. [5] speeds up the algorithm considerably without sacrificing the quality of the restoration. For this purpose, it imposes coherence constraints on the candidate search. If an artefact pixel to be synthesized has one or more neighbors that have already been synthesized, then the candidates used for restoring these neighbors are also used to generate candidates for the current pixel. Namely, the shifts between the current pixel and the previously synthesized neighbors are used to shift their respective candidates in order to generate the replacement candidates for the current pixel. From the generated candidates whose scores stay above a certain threshold, the best scoring one is selected for restoring the current pixel. If no shifted candidate has a score above the chosen threshold, a complete search is initiated as previously described. It should be noted that the coherence constraint can only be imposed after pixels from the outermost layer of the artefact have been already synthesized.

4.2. Integration of structure and texture reconstruction

The mask created in the structure reconstruction step consists of distinctly labeled areas, corresponding to different objects (see Figure 1(e)). Each object’s mask consists of a part lying inside the artefact, and one lying outside it, namely, in the *belt* area (see Figure 2; for simplicity reasons we will not represent the belt in the following figures). We disregard those parts of the masks lying farther away than the belt. Since the texture of an area inside the artefact should resemble the texture of the *same* object outside the artefact, we can use the computed masks to constrain the texture synthesis scheme of Bornard et al. such that the sampling step takes place only in the belt area covered by the same object mask (see Figure 6).

Nevertheless, when computing the difference between the neighborhood of a pixel inside the artefact and the neighborhood of a pixel from the corresponding belt subset, the neighborhood window is allowed to fall outside the current object mask. In practice, for artefact pixels lying along object edges (especially straight edges), this gives priority to candidate pixels lying along the same edge, effectively reconstructing the effect of transition between two objects in an image (see Figure 7).

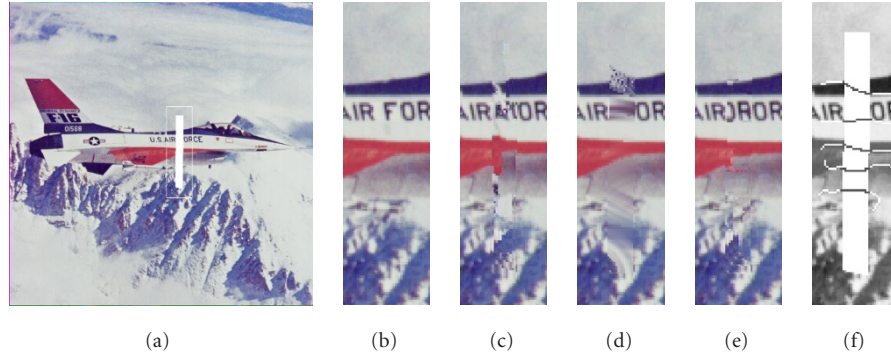


FIGURE 9: Comparison between different types of restoration (all images are zoom-ins on the artefact area, except the first one). (a) Original image with artefact overlay; (b) original artefact contents; (c) texture-only restoration; (d) smooth restoration based on edges; (e) proposed method; (f) structure reconstructed in the proposed method.

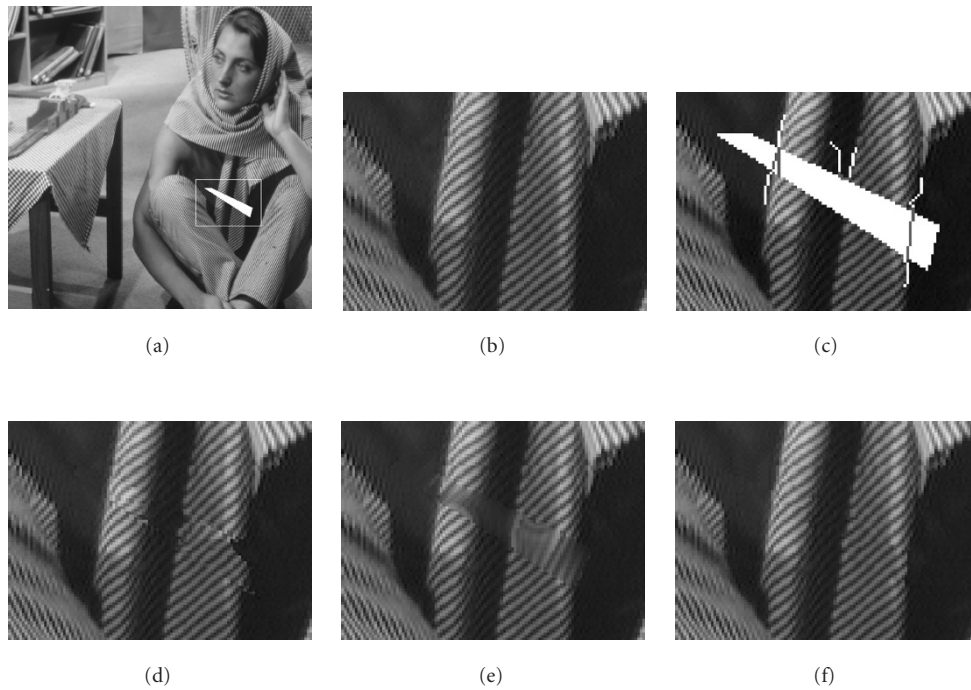


FIGURE 10: Comparison between different types of restoration (all images are zoom-ins on the artefact area, except the first one). (a) Original image with artefact overlay; (b) original artefact contents; (c) structure reconstructed in the proposed method; (d) texture-only restoration; (e) smooth restoration based on edges; (f) proposed method.

5. EXPERIMENTS AND RESULTS

This section describes the performance of our proposed restoration algorithm. In Figures 8, 9, 10, 11, 12, 13, and 14 we demonstrate it by some visual examples of restoration based on the current algorithm, compared to the algorithms from [5, 7]. All images have a size of 512×512 pixels, except the image from Figure 14, which has a size of 200×200 pixels.

In general, the structure-based algorithm from [7] was affected by the failure of the edge detection in the textured areas and by the smooth interpolation approach used therein. At times, the edge detection failure manifested itself not only as missed edges, but also as supplementary, fake edges, as was the case in Figure 8e. On its turn, the texture-based algorithm from [5] was not able to accurately render the object edges. Only the combined approach managed to preserve

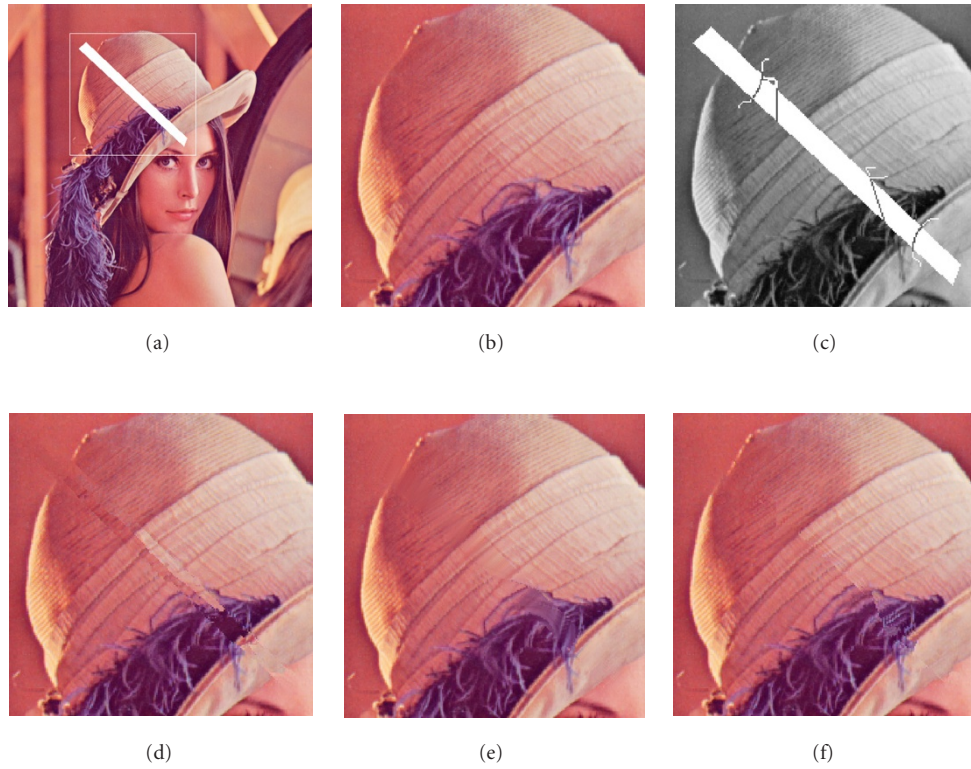


FIGURE 11: Comparison between different types of restoration (all images are zoom-ins on the artefact area, except the first one). (a) Original image with artefact overlay; (b) original artefact contents; (c) structure reconstructed in the proposed method; (d) texture-only restoration; (e) smooth restoration based on edges; (f) proposed method.

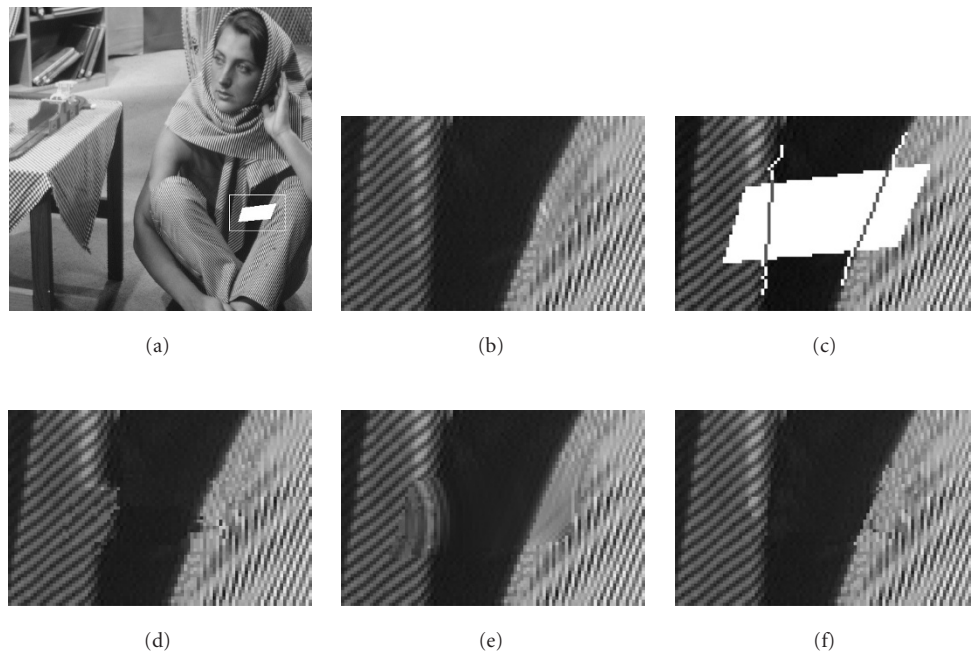


FIGURE 12: Comparison between different types of restoration (all images are zoom-ins on the artefact area, except the first one). (a) Original image with artefact overlay; (b) original artefact contents; (c) structure reconstructed in the proposed method; (d) texture-only restoration; (e) smooth restoration based on edges; (f) proposed method.

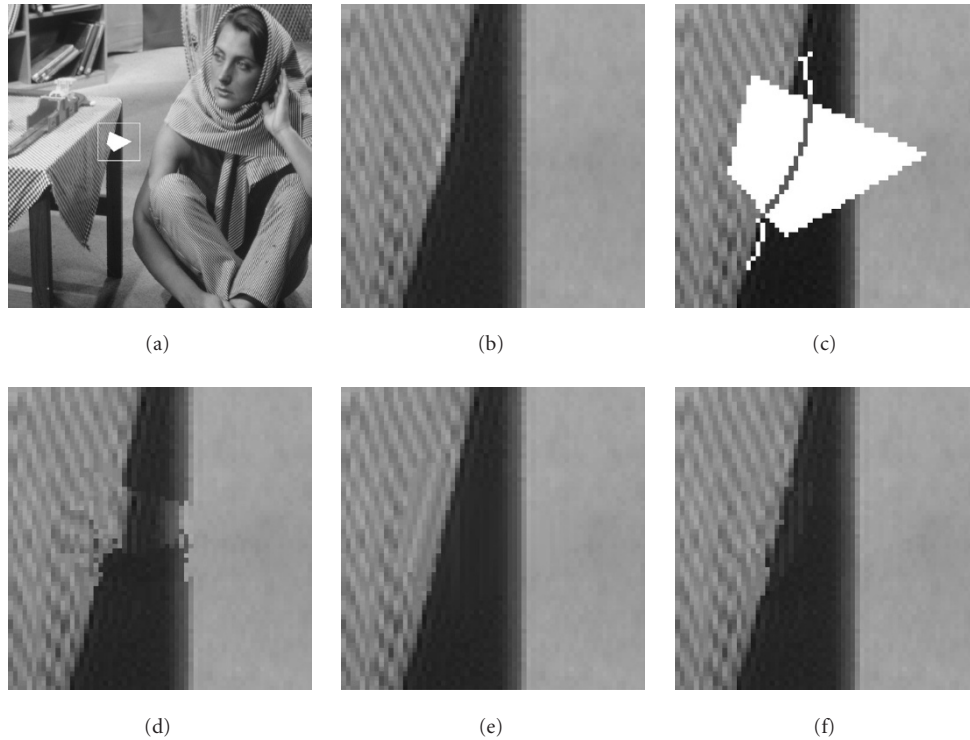


FIGURE 13: Comparison between different types of restoration (all images are zoom-ins on the artefact area, except the first one). (a) Original image with artefact overlay; (b) original artefact contents; (c) structure reconstructed in the proposed method; (d) texture-only restoration; (e) smooth restoration based on edges; (f) proposed method.

both object borders and their texture (see Figures 8, 9, 10, 11, 12, and 13).

It is interesting to note that the two main reconstruction steps, the structure reconstruction and the texture synthesis, generally help each other, resulting in better restoration results than each of the steps separately. As such, a wrong texture interpolation (which tends to give irregular object borders) is “straightened” by the structure within which it must fit. Conversely, errors which take place sometimes in the structure reconstruction step may get corrected by the texture restoration algorithm, as shown in Figures 10 and 11.

As opposed to the texture synthesis approach, the proposed method is able to handle situations where overlapping structures are involved. This is the case in Figure 14, where two grey bars cross each other. The texture synthesis algorithm could only reconstruct one of the bars, while the other one looks “interrupted,” since the reconstruction of the first one brought along some neighborhood color (the black background). The proposed method correctly reconstructed the overlapped bars. One can clearly see in Figure 14e that two groups of edge couples were formed that cross each other.

6. DISCUSSION AND CONCLUSIONS

We have presented here a restoration algorithm which combines structure reconstruction and texture synthesis. Both

algorithms make use of spatial information only. Based on edge information coming from an object segmentation process, the structure reconstruction step recovers the structure of the image (the object borders) inside the artefact. The texture reconstruction step is then used to paint in the missing areas of the objects with their respective textures.

One of the main advantages of our method (over the texture synthesis method alone) is that the nonstationarity of the MRF is modeled not only by using an adaptive window size. It is actually modeled by the various object masks the texture restoration process is confined to. In this way, the actual sampling windows are shaped by the object masks, thus getting more accurate conditional PDFs for each pixel. Of course, questions may be raised about the dependence on the object segmentation results. Indeed, at times, this may influence negatively the structure reconstruction step. In our opinion, however, an object segmentation is certainly preferable to a grid-like segmentation into square blocks that would be independent of the image content, followed by a “blind” texture synthesis that would sample indiscriminately from the entire neighborhood, as is the case in [27].

Despite the fact that the segmentation used is among the best available ones, the current state of the art in object segmentation is still far from perfect. Many relevant edges are still missed, and nonexistent ones get erroneously detected. Although the texture synthesis process helps in recovering from some of these cases, both types of errors may still affect

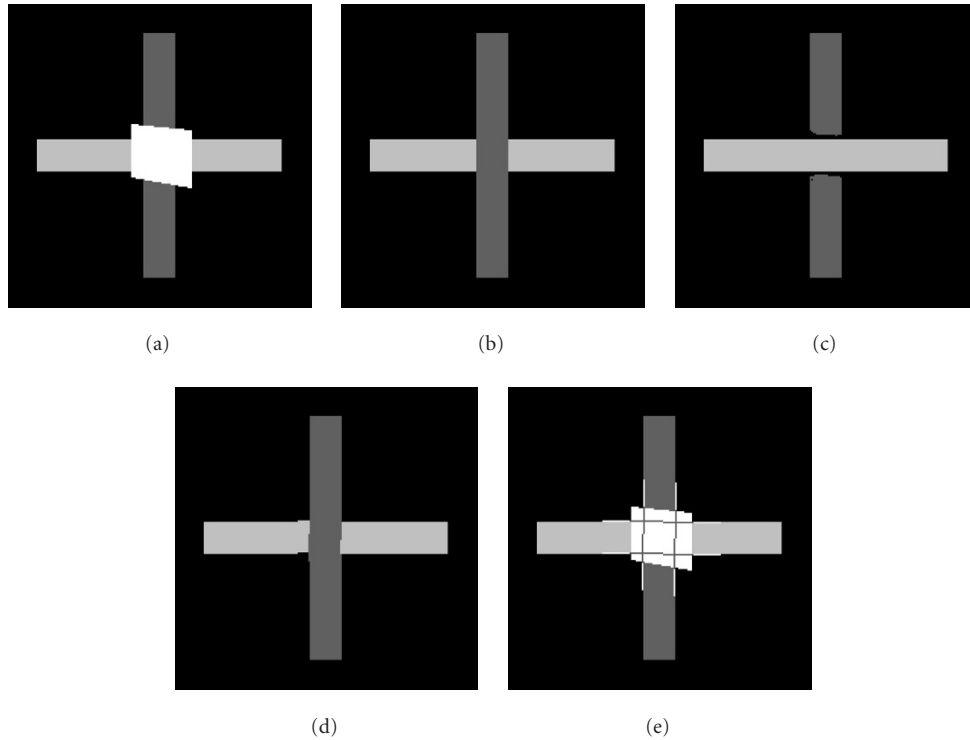


FIGURE 14: Example of crossing structures. (a) Original image with artefact overlay; (b) original image contents; (c) texture-only restoration; (d) proposed method; (e) structure reconstructed in the proposed method.

the results negatively. Nevertheless, our proposed restoration algorithm showed its superiority even in such far from ideal conditions.

The same holds for the texture synthesis algorithm. While the one we used is among the top ones, it may go (partially) wrong at times. For example, it does not handle shadings smoothly. When the opposite sides of an artefact belong to the same object and differ in intensity, a sudden luminosity change can be seen in the middle of the restored artefact area, where the iteratively synthesized sides meet each other. Imposing continuity constraints on the low frequency contents within each object mask will help overcome these problems.

The work presented here can be improved in several ways. One of these is by assigning local certainties to the object segmentation masks (and consequently to the locally extracted features). These certainties could then be used to avoid building structure in areas with low-certainty features or to trigger a process for improving the segmentation masks.

The structure reconstruction step can be enhanced by a thorougher analysis of the shapes of the extracted edges and their coupling. For example, having a measure of the smoothness of transition between adjacent objects can contribute more information to the process of making edge couples.

One of the implicit assumptions made in this paper is that the artefact masks do not have holes. Indeed, the overwhelming majority of artefacts from old films do not have

holes. When they do have them, a few solutions could be applied. The simplest one is to consider that the artefact does not have holes, restore it in the way presented in our paper, and then paste the original content of the artefact holes back into the image (thus overriding a part of the restoration result). This, of course, neglects the structure that may be present inside the artefact holes, which might help in guiding the structure reconstruction process. In some cases, the information present in the artefact holes may even be used to decide which group gets painted in the foreground. Another solution would be to split the artefact mask conveniently such that no resulting submask contains any holes, and then proceed with the normal restoration algorithm.

Currently, in case several groups have been formed, the order of inpainting is arbitrary, except for the groups consisting of one edge couple, which are pushed to the background. More clues may actually be used to decide the inpainting order. If depth information is available (e.g., from range sensors, or from the analysis of the object motion in neighboring frames of an image sequence), this would help in establishing the right inpainting order.

When tracing the object boundaries inside the artefacts, we made simple assumptions about their shapes, considering them to have constant curvature locally (i.e., being either circular or straight). These assumptions do not hold in all cases. However, it would be very hard and even hazardous to try to detect more complex shapes. The largest

radical improvement of object boundary reconstruction can take place in the context of temporal restoration of image sequences. Here, tracking an object along several frames can give us precious information about the actual object shape.

The results of the algorithm we present here are very promising. They indicate that the current algorithm inherits the strengths of the two algorithms it combines, while avoiding (some of) their weaknesses. The algorithm's main strength comes from the fact that the structure reconstruction and the texture synthesis algorithms help each other to achieve better results. As such, the *constrained* texture synthesis that we propose shows robustness against errors of the modules upon which it is built.

ACKNOWLEDGMENTS

This work was partly funded by the EU's IST research and technological development program. Portions of it were carried out within the *Brava* project (Broadcast Archives Restoration Through Video Analysis) [32]. We would like to thank the project partners from INA (Institut National de L'Audiovisuel), Paris, France, namely, Raphaël Bornard, Emanuelle Lecan, Louis Laborelli, and Jean-Hugues Chenot, for providing the source code of their texture synthesis algorithm. We would also like to thank the authors of the segmentation algorithm [31] we used, Y. Deng and B. S. Manjunath, who made their code publicly available on the Internet [33].

REFERENCES

- [1] A. C. Kokaram, *Motion Picture Restoration: Digital Algorithms for Artefact Suppression in Degraded Motion Picture Film and Video*, Springer, Berlin, Germany, 1998.
- [2] P. M. B. van Roosmalen, *Restoration of archived film and video*, Ph.D. thesis, Delft University of Technology, Delft, the Netherlands, 1999.
- [3] <http://brava.ina.fr/brava-public/impairments.list.en.html>.
- [4] M. E. Al-Mualla, C. N. Canagarajah, and D. R. Bull, *Video Coding for Mobile Communications*, Academic Press, Boston, Mass, USA, 2002.
- [5] R. Bornard, E. Lecan, L. Laborelli, and J.-H. Chenot, "Missing data correction in still images and image sequences," in *Proc. 10th ACM International Conference on Multimedia (ACM MM '02)*, pp. 355–361, Juan les Pins, France, December 2002.
- [6] A. Rareş, M. J. T. Reinders, and J. Biemond, "Image sequence restoration in the presence of pathological motion and severe artifacts," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '02)*, vol. 4, pp. 3365–3368, Orlando, Fla, USA, May 2002.
- [7] A. Rareş, M. J. T. Reinders, and J. Biemond, "Edge-based image restoration," to appear in *IEEE Trans. Image Processing*.
- [8] S. Masnou and J.-M. Morel, "Level lines based disocclusion," in *Proc. IEEE International Conference on Image Processing (ICIP '98)*, vol. 3, pp. 259–263, Chicago, Ill, USA, October 1998.
- [9] C. Ballester, M. Bertalmio, V. Caselles, G. Sapiro, and J. Verdera, "Filling-in by joint interpolation of vector fields and gray levels," *IEEE Trans. Image Processing*, vol. 10, no. 8, pp. 1200–1211, 2001.
- [10] C. Ballester, V. Caselles, J. Verdera, M. Bertalmio, and G. Sapiro, "A variational model for filling-in gray level and color images," in *Proc. 8th IEEE International Conference on Computer Vision (ICCV '01)*, vol. 1, pp. 10–16, Vancouver, British Columbia, Canada, July 2001.
- [11] C. Ballester, V. Caselles, and J. Verdera, "A variational model for disocclusion," in *Proc. IEEE International Conference on Image Processing (ICIP '03)*, vol. 3, pp. 677–680, Barcelona, Spain, September 2003.
- [12] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester, "Image inpainting," in *Proc. International Conference on Computer Graphics and Interactive Techniques (ACM SIGGRAPH '00)*, pp. 417–424, New Orleans, La, USA, July 2000.
- [13] M. Bertalmio, A. L. Bertozzi, and G. Sapiro, "Navier-Stokes, fluid dynamics, and image and video inpainting," in *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '01)*, vol. 1, pp. 355–362, Kauai Marriott, Hawaii, USA, December 2001.
- [14] T. F. Chan and J. Shen, "Mathematical models for local non-texture inpaintings," *SIAM Journal on Applied Mathematics*, vol. 62, no. 3, pp. 1019–1043, 2001.
- [15] 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.
- [16] T. F. Chan, S. H. Kang, and J. Shen, "Euler's elastica and curvature-based inpainting," *SIAM Journal on Applied Mathematics*, vol. 63, no. 2, pp. 564–592, 2002.
- [17] S. Masnou, "Disocclusion: a variational approach using level lines," *IEEE Trans. Image Processing*, vol. 11, no. 2, pp. 68–76, 2002.
- [18] H. Knutsson and C.-F. Westin, "Normalized and differential convolution: methods for interpolation and filtering of incomplete and uncertain data," in *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '93)*, pp. 515–523, New York, NY, USA, June 1993.
- [19] L. Atzori and F. G. B. De Natale, "Error concealment in video transmission over packet networks by a sketch-based approach," *Signal Processing: Image Communication*, vol. 15, no. 1–2, pp. 57–76, 1999.
- [20] L. Atzori and F. G. B. De Natale, "Reconstruction of missing or occluded contour segments using Bezier interpolations," *Signal Processing*, vol. 80, no. 8, pp. 1691–1694, 2000.
- [21] L. Atzori, F. G. B. De Natale, and C. Perra, "A spatio-temporal concealment technique using boundary matching algorithm and mesh-based warping (BMA-MBW)," *IEEE Trans. Multimedia*, vol. 3, no. 3, pp. 326–338, 2001.
- [22] A. A. Efros and T. K. Leung, "Texture synthesis by non-parametric sampling," in *Proc. 7th IEEE International Conference on Computer Vision (ICCV '99)*, vol. 2, pp. 1033–1038, Kerkyra, Corfu, Greece, September 1999.
- [23] A. Criminisi, P. Pérez, and K. Toyama, "Object removal by exemplar-based inpainting," in *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '03)*, vol. 2, pp. 721–728, Madison, Wis, USA, June 2003.
- [24] A. C. Kokaram, "Parametric texture synthesis for filling holes in pictures," in *Proc. IEEE International Conference on Image Processing (ICIP '02)*, vol. 1, pp. 325–328, Rochester, NY, USA, September 2002.
- [25] S. T. Acton, D. P. Mukherjee, J. P. Havlicek, and A. C. Bovik, "Oriented texture completion by AM-FM reaction-diffusion," *IEEE Trans. Image Processing*, vol. 10, no. 6, pp. 885–896, 2001.

- [26] A. N. Hirani and T. Totsuka, "Combining frequency and spatial domain information for fast interactive image noise removal," in *Proc. International Conference on Computer Graphics and Interactive Techniques (ACM SIGGRAPH '96)*, pp. 269–276, New Orleans, La, USA, August 1996.
- [27] S. D. Rane, G. Sapiro, and M. Bertalmio, "Structure and texture filling-in of missing image blocks in wireless transmission and compression applications," *IEEE Trans. Image Processing*, vol. 12, no. 3, pp. 296–303, 2003.
- [28] M. Bertalmio, L. Vese, G. Sapiro, and S. Osher, "Simultaneous structure and texture image inpainting," *IEEE Trans. Image Processing*, vol. 12, no. 8, pp. 882–889, 2003.
- [29] J. Jia and C.-K. Tang, "Image repairing: robust image synthesis by adaptive ND tensor voting," in *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '03)*, vol. 1, pp. 643–650, Madison, Wis, USA, June 2003.
- [30] J. Jia and C.-K. Tang, "Inference of segmented color and texture description by tensor voting," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 26, no. 6, pp. 771–786, 2004.
- [31] Y. Deng and B. S. Manjunath, "Unsupervised segmentation of color-texture regions in images and video," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 23, no. 8, pp. 800–810, 2001.
- [32] <http://brava.ina.fr/>.
- [33] <http://vision.ece.ucsb.edu/segmentation/jseg>.

A. Rareş was born in Bucharest, Romania. He received the B.S. and M.S. degrees in computer science from Politehnica University of Bucharest in 1996 and 1997, respectively. In 2004 he received the Ph.D. degree in electrical engineering from Delft University of Technology, The Netherlands. From 1996 to 1999 he was a Teaching Assistant in the Faculty of Automatic Control and Computers at Politehnica University of Bucharest. From 1999 to 2003 he was a Ph.D. student in the Information and Communication Theory Group, Mediamatics Department of the Faculty of Electrical Engineering, Mathematics and Computer Science at Delft University of Technology. Since 2003, he has been working as a researcher in the Laboratory for Clinical and Experimental Image Processing at Leiden University Medical Center, The Netherlands. His research interests are in image and video processing, including restoration, object tracking, motion estimation, data compression, and medical image analysis.



M. J. T. Reinders received his M.S. degree in applied physics and a Ph.D. degree in electrical engineering from Delft University of Technology, The Netherlands, in 1990 and 1995, respectively. Recently he became a Professor in bioinformatics in the Mediamatics Department of the Faculty of Electrical Engineering, Mathematics and Computer Science at the Delft University of Technology. The background of Professor Reinders is in pattern recognition. Besides studying fundamental issues, he applies pattern recognition techniques to the areas of bioinformatics, computer vision, and context-aware recommender systems. His special interest goes towards understanding complex systems (such as biological systems) that are severely undersampled.



J. Biemond was born in De Kaag, The Netherlands. He received the M.S. and Ph.D. degrees in electrical engineering from Delft University of Technology, Delft, The Netherlands, in 1973 and 1982, respectively. Currently, he is a Professor in the Information and Communication Theory Group and Head of the Department of Mediamatics at the Faculty of Electrical Engineering, Mathematics and Computer Science at Delft University of Technology. His research interests include image and video processing (restoration, compression, content-based retrieval, motion estimation) with applications in digital TV, 3D TV, HDTV, multimedia, digital libraries, scan-rate conversion, and computer vision. He has published extensively in these fields and supervised more than 25 Ph.D. theses covering these fields. He was a Visiting Professor at Rensselaer Polytechnic Institute, Troy, NY, and at Georgia Institute of Technology, Atlanta, Ga, and recipient of the Dutch Telecom Award "Vederprijs" for his contributions in the area of digital image processing, in particular in image restoration and subband coding. He is a Fellow of the IEEE. He served as the General Chairman of the 1997 Visual Communication and Image Processing Conference (VCIP-97), San Jose, Calif, and as General Cochair of the 2005 IEEE International Conference on Multimedia & Expo (ICME-2005) in Amsterdam, The Netherlands.

