Christine Guillemot and Olivier Le Meur

Image Inpainting

Overview and recent advances

Image inpainting refers to the process of restoring missing or damaged areas in an image. This field of research has been very active over recent years, boosted by numerous applications: restoring images from scratches or text overlays, loss concealment in a context of impaired image transmission, object removal in a context of editing, or disocclusion in image-based rendering (IBR) of viewpoints different from those captured by the cameras. Although earlier work dealing with disocclusion has been published in [1], the term *inpainting* first appeared in [2] by analogy with a process used in art restoration.

Image inpainting is an ill-posed inverse problem that has no well-defined unique solution. To solve the problem, it is therefore necessary to introduce image priors. All methods are guided by the assumption that pixels in the known and unknown parts of the image share the same statistical properties or geometrical structures. This assumption translates into different local or global priors, with the goal of having an inpainted image as physically plausible and as visually pleasing as possible.

The first category of methods, known as *diffusion-based inpainting*, introduces smoothness priors via parametric models or partial differential equations (PDEs) to propagate (or diffuse) local structures from the

Digital Object Identifier 10.1109/MSP.2013.2273004
Date of publication: 5 December 2013

exterior to the interior of the hole (as shown in Figure 1, where $\mathbf{U}$ denotes the unknown part or the hole to be filled in, and $\mathbf{S}$ the source or known part of the image). Many variants exist using different models (linear, nonlinear, isotropic, or anisotropic) to favor the propagation in particular directions or to take into account the curvature of the structure present in a local neighborhood. These methods are naturally well suited for completing straight lines, curves, and for inpainting small regions. They, in general, avoid having unconnected edges that are perceptually annoying. However, they are not well suited for recovering the texture of large areas, which they tend to blur.

The second category of methods is based on the seminal work of Efros and Leung [3] and exploits image statistical and self-similarity priors. The statistics of image textures are assumed to be stationary (in the case of random textures) or homogeneous (in the case of regular patterns). The texture to be synthesized is learned from similar regions in a texture sample or from the known part of the image. Learning is done by sampling, and by copying or stitching together patches (called *examplar*) taken from the known part of the image. The corresponding methods are known as examplar-based techniques.

With the advent of sparse representations and compressed sensing, sparse priors have also been considered for solving the inpainting problem. The image (or the patch) is in this case assumed to be sparse in a given basis [e.g., discrete cosine transform (DCT), or wavelets]. Known and unknown parts of the image are assumed to share the same sparse representation. Examplar-based and sparse-based methods are better suited than diffusion-based techniques for filling large texture areas. Hybrid solutions have then naturally emerged, which combine methods dedicated to structural (geometrical) and textural components.

This article surveys the theoretical foundations, the different categories of methods, and illustrates the main applications.

THE INPAINTING PROBLEM

An image $\mathbf{I}$ can be mathematically defined as

$$\mathbf{I}: \begin{vmatrix} \Omega \subset \mathcal{R}^n \rightarrow \mathcal{R}^m \\ \mathbf{x} \rightarrow \mathbf{I}(\mathbf{x}), \end{vmatrix} \tag{1}$$

where $\mathbf{x}$ represents a vector indicating spatial coordinates of a pixel $p_{\mathbf{x}}$, which in the case of a two-dimensional (2-D) image ($n = 2$), is defined as $\mathbf{x} = (x, y)$. In the case of a color image, each pixel carries three color components ($m = 3$) defined in the $(\mathrm{R}, \mathrm{G}, \mathrm{B})$ color space. Each cth image color channel of $\mathbf{I}$ is denoted $\mathbf{I}^c : \Omega \rightarrow \mathcal{R}$. In the inpainting problem, the input image $\mathbf{I}$ (i.e., each color channel of the image) is assumed to have gone through a degradation operator, denoted $\mathbf{M}$, which has removed samples from the image. As a result, the generic definition domain Ω of the input image $\mathbf{I}$ can be seen as composed of two parts: $\Omega = \mathbf{S} \cup \mathbf{U}$, $\mathbf{S}$ being the known part of $\mathbf{I}$ (source region) and $\mathbf{U}$ the unknown part of $\mathbf{I}$, which we search to estimate. The observed degraded version $\mathbf{F}$ of the image can also be expressed as $\mathbf{F} = \mathbf{MI}$.

The goal of inpainting is to estimate the color components of the pixels $p_{\mathbf{x}}$ located at each position $\mathbf{x}$ in the unknown region $\mathbf{U}$, from the pixels located in $\mathbf{S}$ the known region, to finally construct the inpainted image. The objective in terms of quality is that the recovered region looks natural to the human eye, and is as physically plausible as possible. Typical inpainting artifacts are unconnected edges, blur, or inconsistent pieces of texture (also called texture garbage).

DIFFUSION-BASED INPAINTING USING SMOOTHNESS PRIORS

The term *diffusion* comes from the idea of propagating local information with smoothness constraints, by analogy with physical phenomena like heat propagation in physical structures. These phenomena can be formalized with PDEs, and diffusion is therefore performed using PDE-based regularization. Inpainting using diffusion smoothly propagates local image structures from the exterior to the interior of the hole, "imitating" the gesture of professional painting restorators. The considered data are assumed to satisfy smoothness constraints and are iteratively regularized, producing a continuous sequence of smoother images.

Image regularization can be defined locally as the diffusion of pixel values using PDEs, or formulated as the minimization of a functional measuring a global image variation. Moreover, to preserve edges, the regularization (or smoothing) must follow directions given by the local image structure. If the pixel is located on an image contour, the smoothing must be performed along the contour direction and not across boundaries. If the pixel is located in an homogeneous region, the smoothing can be performed in all directions. The first step is therefore to retrieve the local image geometry and then use PDEs or variational methods to describe continuous evolutions of the image and of its structures.

RETRIEVING IMAGE LOCAL GEOMETRY

Local image geometry is in general retrieved by computing gray level lines (also called *isophotes*) or structure tensors.

ISOPHOTES

Isophotes are lines of constant intensity within an image as shown by the red lines in Figure 1. Their directions at a given pixel $p_{\mathbf{x}}$ (located on the front line) are given by the normal to the discretized gradient vector computed at this point, which will be mathematically expressed as $\nabla \mathbf{I}^{\perp}$ (instead of $\nabla \mathbf{I}^{\perp}_{p_{\mathbf{x}}}$ for simplifying the notations). In the case of 2-D color images $\mathbf{I}$ with three color channels, the derivatives are first computed separately for each color channel and then added to produce the final color gradient in the pixel location $p_{\mathbf{x}}$. The discretized gradient vector gives the direction of largest spatial changes, while its 90° rotation is the direction of smallest spatial changes, hence, of the isophotes.

STRUCTURE TENSORS

Another method used to retrieve local geometry is based on the computation of the spectral elements of the structure tensor, also called the *Di Zenzo matrix* [4]. The structure tensor for a scalar image (with only one color channel) is computed at each point $p_{\mathbf{x}}$ as $\mathbf{G} = \nabla \mathbf{I} \nabla \mathbf{I}^T$, where the term $\nabla \mathbf{I}$ denotes

the image spatial gradient, and the term $\nabla \mathbf{I}^T$ denotes the transpose of the image gradient. In the case of color images with three color channels, the structure tensor is computed for each color channel and the results are then added as $\mathbf{G} = \sum_{c=1}^{3} \nabla \mathbf{I}^c \nabla \mathbf{I}^{cT}$ to produce the structure tensor of the image $\mathbf{I}$. By using the spectral decomposition, the structure tensor $\mathbf{G}$ can be expressed as $\mathbf{G} = \mathbf{R}\mathbf{D}\mathbf{R}^T$, where the columns of $\mathbf{R}$ are the eigenvectors ($\mathbf{v}_1$ and $\mathbf{v}_2$) of $\mathbf{G}$ and $\mathbf{D}$ is the diagonal matrix whose entries are the corresponding eigenvalues λ_1 and λ_2. The orientation of the eigenvector $\mathbf{v}_2$ corresponding to the smallest eigenvalue λ_2 is the orientation with the lowest fluctuations (i.e., of the isophote; see Figure 1). The orientation of the eigenvector $\mathbf{v}_1$ corresponding to the highest eigenvalue λ_1 gives the gradient direction (i.e., of the normal to the level curve at this point). The retrieved local image geometry can be used to control directions along which pixel values are propagated in the inpainting process, as well as the regularization strength.

In the following paragraphs, we first recall the basics of PDE-based regularization and then present how PDE-based regularization is used for inpainting. We finally give the main ideas of image regularization and inpainting with variational methods.

BASICS OF PDE-BASED REGULARIZATION OR DIFFUSION

The simplest formulation concerns isotropic regularization (or diffusion), which stems from the linear heat (or heat flow) equation

$$\begin{cases} \mathbf{I}_{(t=0)} = \mathbf{F} \\ \dfrac{\partial \mathbf{I}}{\partial t} = \Delta \mathbf{I}, \end{cases} \tag{2}$$

where $\mathbf{F}$ is a degraded version of the original picture $\mathbf{I}$. The term $\Delta \mathbf{I}$ denotes the image Laplacian. The PDE evolution is parameterized with a time variable t, which describes the continuous progression of the function $\mathbf{I}$. Isotropic diffusion, which minimizes these variations in all directions, acts as a low-pass linear filtering suppressing high frequencies in the image. For this reason, the method suffers from blur close to edges and contours. More general formalisms (using nonlinear PDEs), first used for describing physical and fluid dynamics phenomena, have been introduced to better preserve edges and sharpness.

NONLINEAR ISOTROPIC DIFFUSION

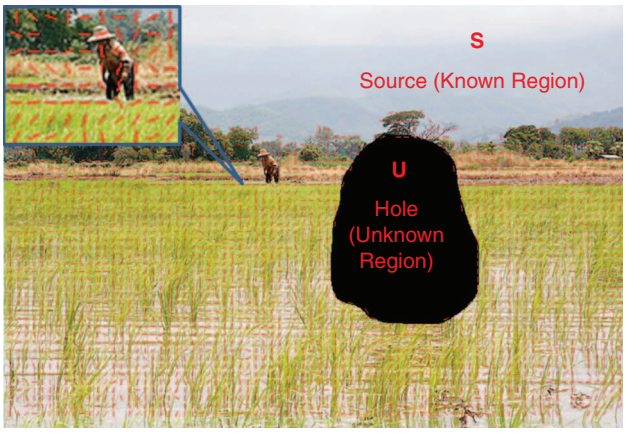
The heat equation can be rewritten in a divergence form as

$$\frac{\partial \mathbf{I}}{\partial t} = \Delta \mathbf{I} = \operatorname{div}(\nabla \mathbf{I}), \tag{3}$$

where the notation $\operatorname{div}(.)$ stands for the divergence operator which measures how fast a vector field (here image intensities) is changing in x and y directions. This led Perona and Malik in [5] to introduce a nonlinear extension of the heat equation, as

$$\frac{\partial \mathbf{I}}{\partial t} = \operatorname{div}(g(\| \nabla I \|^2) \nabla \mathbf{I}) \tag{4}$$

by introducing a diffusion coefficient (also called conductivity coefficient), which is a scalar in $[0, 1]$ returned by a decreasing



[FIG1] Isophotes or lines (in red) of constant intensity represented for one out of ten pixels with a zoom on one region of the image. (Original image courtesy of [39].)

function $g(.)$ of the image gradient. The goal of the diffusion coefficient is to limit diffusion around edges and avoid smoothing across region boundaries. Two functions have been proposed in [5] for computing the diffusion coefficient

$$g(\| \nabla \mathbf{I} \|) = e^{-(\| \nabla \mathbf{I} \|/\alpha)^2} \tag{5}$$

and

$$g(\| \nabla \mathbf{I} \|) = \frac{1}{1 + (\| \nabla \mathbf{I} \|/\alpha)^2}, \tag{6}$$

where α is a constant chosen experimentally to control the sensitivity to edges. The function $g(.)$ vanishes in the neighborhood of steep edges (high gradients) and returns a value close to 1 on flat regions (low gradients).

The method is often referred to as *anisotropic diffusion* even though it is rather a nonlinear diffusion. Indeed, the filter applied locally is isotropic, but its response is adapted locally and varies in space so that the diffusion process is lessened near edges and is strong in homogenous areas. A good review of this model can be found in [6].

ANISOTROPIC DIFFUSION USING TENSOR FIELDS

One step further has been to introduce locally adapted filters for performing truly anisotropic diffusion close to image structures such as edges [7]. Anisotropic regularization (or diffusion) refers to a smoothing in privileged spatial directions with different weights. Weickert in [7] used a field of diffusion tensors to find privileged directions of diffusion.

The diffusion tensor $\mathbf{D}$ (symmetric and positive-definite matrix) is derived in each point $\mathbf{x} = (x, y)$ of the image from the spectral elements of the structure tensor $\mathbf{G} = \nabla \mathbf{I} \nabla \mathbf{I}^T$ as [7]

$$\mathbf{D} = \lambda_1 \mathbf{v}_1 \mathbf{v}_1^T + \lambda_2 \mathbf{v}_2 \mathbf{v}_2^T, \tag{7}$$

where $\mathbf{v}_1$ and $\mathbf{v}_2$ are the eigenvectors (λ_1 and λ_2 are the corresponding eigenvalues) of the structure tensor $\mathbf{G}$). Pixel values are therefore anisotropically smoothed along local directions given by the eigenvectors $\mathbf{v}_1$ and $\mathbf{v}_2$, with weights (or smoothing strength) given by the corresponding eigenvalues. In other words, the eigenvalues determine the diffusivities

in the directions of the eigenvectors. The diffusion is governed by the PDE

$$\frac{\partial \mathbf{I}}{\partial t} = \mathrm{div}(\mathbf{D}\nabla \mathbf{I}). \tag{8}$$

Note that, if $\mathbf{D}$ equals the identity matrix, we obtain the heat equation, i.e., isotropic diffusion. Note also that the tensor can be regularized with Gaussian kernels, which leads to a smoothing of the image with small elongated kernels along edges and with Gaussian-like kernels in homogeneous regions. An edge-enhancing diffusion (EED) is further proposed in [7], where the diffusivity decreases with the increased contrast in the direction perpendicular to edges. The effect of the varying diffusivity of EED can be seen in Figure 2. A coherence-enhancing diffusion (CED) is also proposed in [8], which increases the diffusivity along the coherent direction given by $\mathbf{v}_2$ when the coherence measured by $(\lambda_1 - \lambda_2)^2$ increases.

ANISOTROPIC DIFFUSION USING ORIENTED LAPLACIANS

Image smoothness (and its variation) can also be measured by a discretization of the 2-D Laplacian of the image, instead of using gradient or divergence operators. Anisotropic diffusion models are in this case expressed as

$$\frac{\partial \mathbf{I}}{\partial t} = \lambda_1 \mathbf{I}_{\mathbf{v}_1\mathbf{v}_1} + \lambda_2 \mathbf{I}_{\mathbf{v}_2\mathbf{v}_2}, \tag{9}$$

where the terms $\mathbf{I}_{\mathbf{v}_1\mathbf{v}_1}$ and $\mathbf{I}_{\mathbf{v}_2\mathbf{v}_2}$ are the image Laplacians, i.e., the second derivatives of $\mathbf{I}$ in the directions given by the vectors $\mathbf{v}_1$ and $\mathbf{v}_2$ that, as above, are derived from the local image structures (gradients, isophotes, or tensor fields). Here again, the above equations can be applied on each color channel of the image separately or on vectors formed by the three components (R,G,B) of color images to smooth multivalued images.

The diffusion is controlled by the knowledge of the smoothing directions $\mathbf{v}_1$ and $\mathbf{v}_2$, and the corresponding weights λ_1 and λ_2. This can be seen as heat flows oriented along orthonormal directions given by $\mathbf{v}_1$ and $\mathbf{v}_2$.

INPAINTING USING DIFFUSION PDEs

The use of regularization or diffusion for image inpainting was pioneered by Bertalmio et al. [2] in 2000. All PDE-based image regularization methods (isotropic or anisotropic) described in the previous sections can be used for inpainting. The authors in [2] use an anisotropic model that propagates image Laplacians [as formalized by (9)] from the surrounding neighborhood into the interior of the hole. The directions of the propagation are given by the directions of the isophotes estimated by the perpendicular to the image gradient in each point. The algorithm numerically solves the equation

$$\frac{\partial \mathbf{I}}{\partial t} = \nabla(\Delta \mathbf{I})\nabla \mathbf{I}^{\perp} \tag{10}$$

for the image intensity $\mathbf{I}$ inside the hole, until a steady state solution ($\nabla(\Delta \mathbf{I})\nabla \mathbf{I}^{\perp} = 0$), which means that the image Laplacians ($\Delta \mathbf{I}$) remain constant (no variations captured by the gradient ∇) in the directions $\nabla \mathbf{I}^{\perp}$ of the isophote. The term $\nabla(\Delta \mathbf{I})\nabla \mathbf{I}^{\perp}$ is the derivative of $\Delta \mathbf{I}$ in the direction $\nabla \mathbf{I}^{\perp}$, leading to a smooth continuation of available information (the Laplacians) inside the region to be inpainted. In other words, the image information is propagated inside the missing region in a way that aims at preserving the isophote directions. The analogy between this propagation of image intensity along smooth level curves and the transport of vorticity in fluid dynamics formalized with the Navier–Stokes equations has been established in [9]. The image intensity is seen as a stream in fluid dynamics and isophote lines are seen as flow streamlines.

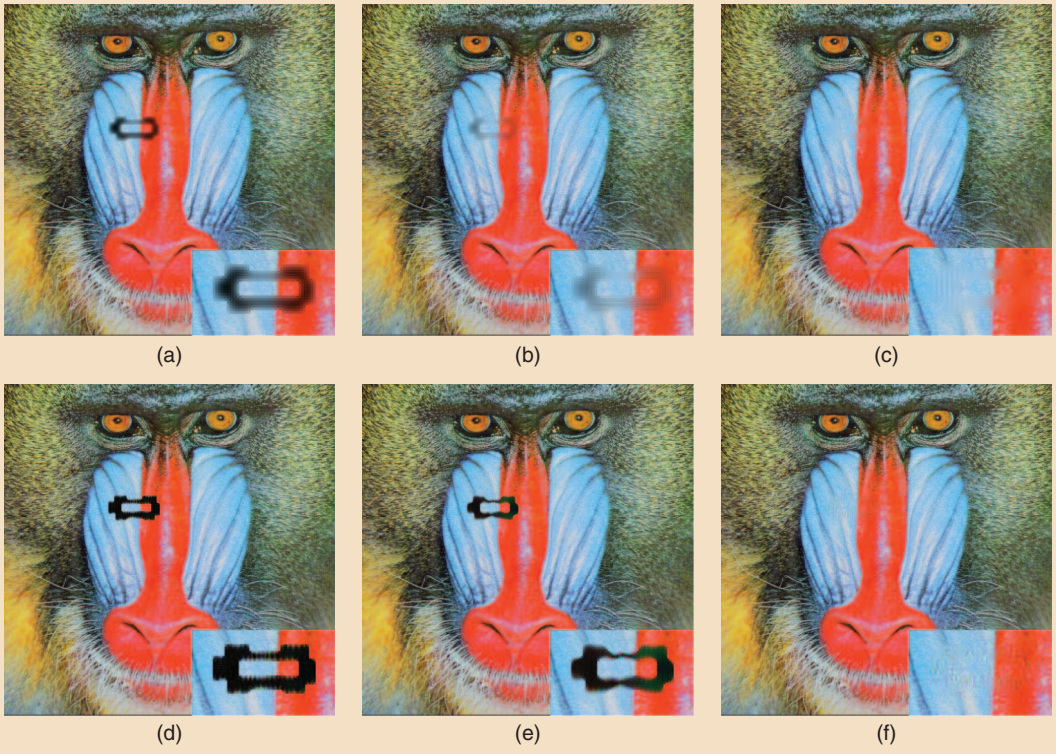
Variants have then been proposed in the use of anisotropic diffusion for image inpainting, for either reducing complexity or better preserving structures. PDE-based methods require implementing iterative numerical methods that are generally quite slow. A fast marching technique is described in [10], which estimates the unknown pixels in one pass using weighted means of already calculated pixels. A trace-based PDE model is proposed in [11] to regularize multivalued images (with multiple color channels). It was observed in [11] that Gaussian behavior inherent to the use of tensors for defining orientation and strength of the diffusion degrades the reconstruction of curved image structures such as corners. This observation led the author in [11] to use heat flows constrained on integral curves to better preserve curvatures in image structures.

Diffusion methods tend to prolong structures (e.g., isophotes) arriving at the border of the region to be filled in. These methods are hence successful for piecewise smooth images, for propagating strong structures, or for filling small gaps. However, they are not well suited for textured images, especially if the region to be completed is large. Although intended to preserve edges, after a few diffusion iterations, the inpainted region appears smooth with a lot of blur when the missing region is large. Figure 2 shows inpainting results after a few iterations when using two different diffusion equations: isotropic diffusion with heat equation, and edge-enhancing diffusion (EED) [7]. Isotropic diffusion introduces a blur over the entire region to be filled in. On the contrary, EED (and similarly for CED) introduces less smoothing across edges. Figure 3 illustrates typical artifacts of diffusion-based inpainting methods when the hole to be filled in is large.

VARIATIONAL INPAINTING

Image regularization, hence image inpainting, may also be formulated as a variational problem where the image is seen as a function of bounded variation (BV). The total variation (TV) image model, first proposed for denoising and deblurring applications in [12], has been applied to image inpainting in [13]–[15]. The energy function is based on the total variational norm, hence the name of the TV image model.

Keeping the goal of a smooth propagation of image intensity, the idea consists in searching for a function $\hat{\mathbf{I}}(\mathbf{x})$ of BV on the set Ω ($\mathrm{BV}(\Omega)$) on which the input image $\mathbf{I}$ is defined, which minimizes a TV energy (defined as the integral of the gradient magnitude) of the image inside the hole, under some



[FIG2] The diffusion process with isotropic diffusion after (a) ten, (b) 20, and (c) 600 iterations. Edge-enhancing diffusion [7] after (d) ten, (e) 50, and (f) 600 iterations.

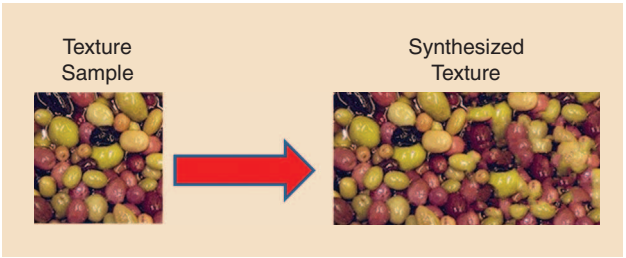
constraint of fidelity to image observations. The minimization problem is written as

$$J_{\mathrm{TV}}(\hat{\mathbf{I}}) = \int_{\Omega} |\nabla \hat{\mathbf{I}}(\mathbf{x})| \, d\mathbf{x} + \lambda \int_{\Omega \setminus \mathbf{U}} (\mathbf{I}(\mathbf{x}) - \hat{\mathbf{I}}(\mathbf{x}))^2 d\mathbf{x}. \qquad (11)$$

The first integral term represents the regularization term, whereas the second term is a data term measuring the fidelity of the reconstructed image to the input image for the known samples. The difference with the denoising problem in [12] resides in the fact that the second integral is computed over $\Omega \setminus \mathbf{U}$ instead of the entire domain Ω. The regularization strength can be seen as spatially varying, where $\lambda(\mathbf{x}) = 0$ in $\mathbf{U}$. It can also vary locally according to the image geometry or to given image models.

Note that a variational formulation has been first applied to the disocclusion problem in [1] where the image is reconstructed from a set of level lines defined as BV functions. The level lines within the hole are geodesic paths joining isophotes arriving at the boundary of the hole. This method is the first variational approach where the detection of the isophotes is followed by their variational continuation across the hole.

[FIG3] Typical blurring effects of (a) and (b) diffusion-based methods (example with the method in [10]) when the hole to be filled in is large. [(a) courtesy of [39]. (b) generated by Christine Guillemot and Olivier Le Meur.]



[FIG4] The texture synthesis problem. The goal is to produce a texture larger than the input sample with a similar visual appearance. (Figure used with permission from the Massachusetts Institute of Technology.)

TV regularization is an effective inpainting technique that is capable of recovering sharp edges with, however, some problems of connectivity. To further improve the edge connectivity, the TV inpainting model has been extended by introducing in the regularization term energy functionals taking into account curve structures, via curvature-driven diffusion models [16], or the Euler elastica functional [17].

EXAMPLAR-BASED METHODS

A second family of inpainting methods has appeared in the last decade based on seminal work on texture synthesis [3], [18] with the aim of better recovering the texture of the missing area. The texture synthesis problem is slightly different from image inpainting. The goal of texture synthesis is to create a texture from a given sample as shown in Figure 4, in such a way that the produced texture is larger than the source sample with a similar visual appearance. This problem is also referred to as *sample-based texture synthesis*. There is a vast body of work on texture synthesis, using either local region growing or a global optimization.

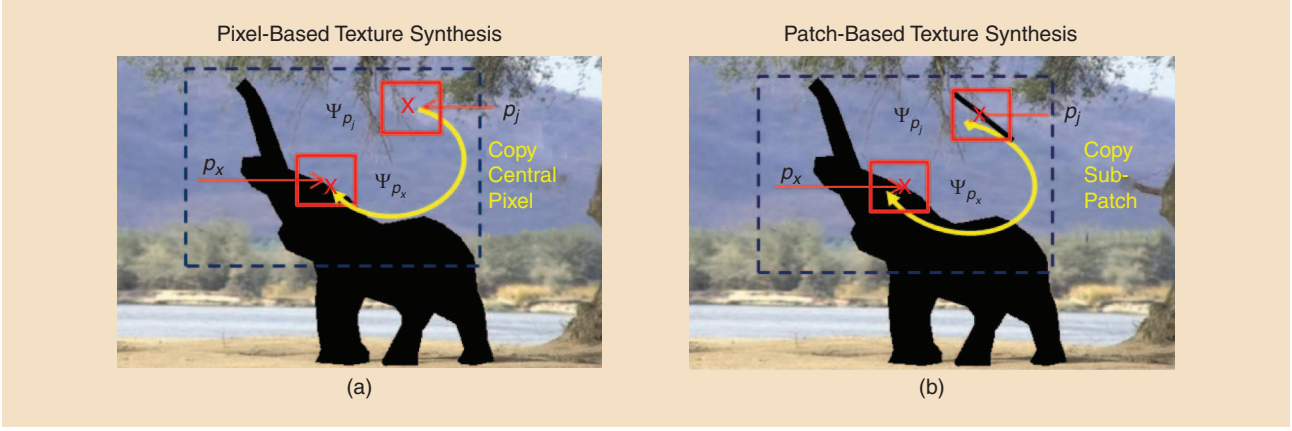
Examplar-based inpainting has been, for a large part, inspired by local region-growing methods that grow the texture one pixel or one patch at a time, while maintaining coherence with nearby pixels.

Most local pixel-based synthesis techniques rely on Markov random fields (MRFs) modeling of textures. Instead of running a complex probabilistic inference on the graphical model of the MRF for learning the missing pixels from the input sample, simpler, yet efficient, approximate solutions have been proposed in [3]. Exploiting both locality (the color of a pixel being assumed to depend on its local neighborhood only) and stationarity (the dependency is independent of the pixel location), the missing pixels are learned by sampling and copying the central pixel of a patch from the sample texture that best matches the known neighborhood of the input pixel to be synthesized, according to a certain distance. Similarly, in [18], the output image is generated pixel-per-pixel in a raster scan order choosing at each step a pixel from the sample image which neighborhood is most similar to the currently available neighborhood in the texture being synthesized. Texture synthesis methods have evolved from pixel-based to patch-based techniques, with recent enhancements relying on elaborated blending and quilting strategies, using, e.g., graph-cut techniques to minimize energy terms along a seam. We come back to this issue in the section "Patch Stitching with Blending and Quilting."

Texture synthesis methods directly apply to the inpainting problem where the known part of the image can be seen as the input texture sample from which the missing pixels can be learned [as illustrated in Figure 5(a)].

The simple pixel-based texture synthesis technique in [3] proceeds as follows. Let $p_{\mathbf{x}}$ be a pixel located at position $\mathbf{x}$ in the image $\mathbf{I}$, and $\Psi_{p_{\mathbf{x}}}$ be the patch centered on the pixel $p_{\mathbf{x}}$. This patch has a known part ($\Psi^S_{p_{\mathbf{x}}}$) and an unknown part ($\Psi^U_{p_{\mathbf{x}}}$). The idea is to search for the patch Ψ_{p_j} (centered on p_j) the most similar to the input patch $\Psi_{p_{\mathbf{x}}}$. The central pixel p_j having a neighborhood most similar to the known neighborhood of $p_{\mathbf{x}}$ is then copied to recover $p_{\mathbf{x}}$. Image information is therefore pixel-per-pixel propagated from the known part to the unknown part of the image.

This pixel-per-pixel recovery algorithm suffers from a high computational cost, even if its complexity can be reduced by constraining the search for best matching patches among the candidates of the neighboring pixels that have been already inpainted [19]. Another limitation is the



[FIG5] The principle of examplar-based methods: search for the patch the most similar to the known part of the input patch to be completed and copy the central pixel for (a) pixel-based approaches or (b) a set of pixels for patch-based approaches. (Figure used with permission from [24].)

difficulty for this type of approach to synthesize textures that are not frontal (with some perspective transformations) and to fill in large and dispersed holes. Moreover, although performing better than diffusion methods on textured areas, the pixel-based synthesis techniques often suffer from synthesis errors propagation and from repetitive patterns, which look unnatural especially in the case of stochastic textures. They also run into difficulties when synthesizing texture formed by an arrangement of small objects.

Approaches synthesizing entire patches rather than only one pixel at a time have then emerged to cope with the drawbacks just mentioned. Instead of synthesizing the missing region pixel per pixel, the idea of patch-based solutions is to recover entire patches in one step by sampling and copying texture patterns (entire patches) from the source [20]. The first step for estimating the pixels in the unknown part $\Psi_{p_x}^U$ of the patch again consists in searching for the patch Ψ_{p_j} (centered on p_j), which is the most similar to the patch $\Psi_{p_x}^S$. But this time, all the pixels from Ψ_{p_j} which are located at the same position as $\Psi_{p_x}^U$ are copied to estimate the unknown pixels of the input patch, as shown on the right side of Figure 5.

The terminology *examplar-based inpainting* now mostly refers to these methods that synthesize entire patches by learning from patches in the known part of the image. Since they synthesize entire patches at once, these methods are faster than pixel-based approaches. Many variants have then been introduced to optimize patch-based methods. These variants concern:

- distance metrics for finding best matching patches
- fast search of best matching patches
- patch processing order
- global spatial coherence via constrained search or global optimization
- patch stitching with blending and quilting
- methods to learn unknown pixels from best matching patches
- multiscale refinement.

These variants are discussed in the next section.

ISSUES AND VARIANTS OF EXAMPLAR-BASED INPAINTING

DISTANCE METRICS

Several metrics exist for measuring similarity between images or between image patches. The most widely used metrics can be classified in the following categories: pixel-based metrics measuring the similarity in terms of difference or cross-correlation between pixel color values and statistics-based metrics measuring the similarity between probability distributions of pixel color values in patches. The sum of squared differences (SSD), the L_p norm, as well as the normalized cross-correlation belong to the first category. Statistics-based metrics include the Bhattacharyya distance, the normalized mutual information (NMI), and the Kullback–Leibler divergence.

The most widely used metric to search for similar patches is the SSD. However, as observed in [21], the SSD introduces some bias towards uniform regions. In other words the SSD favors the copy of pixels from uniform regions.

A weighted Bhattacharya distance has been proposed in [21] ($d_{(\mathrm{SSD,\,BC})}$) to cope with this limitation. This metric is computed as

$$d_{(\mathrm{SSD,\,BC})}(\Psi_{p_x}, \Psi_{p_j}) = d_{\mathrm{SSD}}(\Psi_{p_x}, \Psi_{p_j}) \times d_{\mathrm{BC}}(\Psi_{p_x}, \Psi_{p_j}), \quad (12)$$

where the term $d_{\mathrm{SSD}}(\Psi_{p_x}, \Psi_{p_j})$ is the SSD between the two patches Ψ_{p_x} and Ψ_{p_j}. The term $d_{\mathrm{BC}}(\Psi_{p_x}, \Psi_{p_j})$ is a modified Bhattacharya distance given by $d_{\mathrm{BC}}(\Psi_{p_x}, \Psi_{p_j}) = \sqrt{1 - \sum_k \sqrt{h_1(k) h_2(k)}}$, with h_1 and h_2 representing the histograms of patches Ψ_{p_x} and Ψ_{p_j}, respectively.

However, when two patches have the same distribution, their distance d_{BC} is zero, hence the weighted Bhattacharya distance is also zero, even if one patch is a geometrically transformed version of the other one. A variant of the weighted Bhattacharya distance addressing this problem is proposed in [22] where the SSD is multiplied by $(1+d_{\mathrm{BC}})$. When two patches have the same distribution, the distance value is equal to the SSD between the two patches.

Note that other distance metrics also exist that account for geometrical transformations [23]. Transformed (rotated, scaled, and mirror) versions of existing patches are, in this case, included as possible match candidates [24].

FAST SEARCH OF BEST MATCHING PATCHES

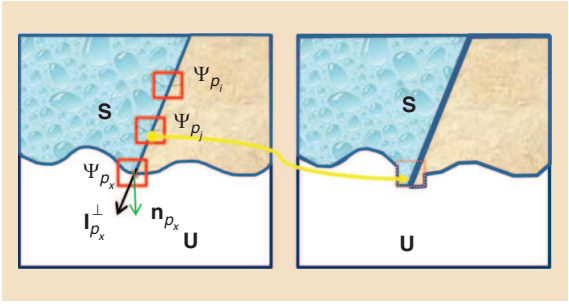
Examplar-based inpainting methods first search for K-nearest neighbors (K-NNs) within the known part of the image. A naive solution to the NN search problem is to compute the distance from the query patch to all possible candidate patches, treating each patch as a point in a high-dimensional space. Faster and approximate NN search methods exist that organize the candidates in specific space-partitioning data structures, such as the k-dimensional trees (kd-trees) [25] or the vantage point trees (vp-trees) [26], according to their distribution in the search space. The kd-trees are a special case of binary space partitioning trees that divide the space along different coordinates. The set of data points is bisected at the median of all points in a selected dimension to build a binary tree. The vp-tree, instead of splitting along coordinate values, splits the set of data points according to their distance to a specific point called the vantage point. The NN search can then be efficiently done by using the tree properties to quickly eliminate large portions of the search space and check only a small portion of candidates. The kd-tree-based matching is one of the most widely used algorithms for finding the nearest patch. However, its number of searched nodes increases exponentially with the space dimension. When the dimension is large (e.g., higher than 15), its search speed becomes very slow. Several NN search algorithms are assessed in [27] for finding similar patches in images.

The tree-based approximate NN (ANN) search methods treat the queries separately. A randomized patch search algorithm called PatchMatch was introduced in [28], which exploits dependency among the queries to perform some collaborative search. The idea relies on the assumption that images are coherent. Once a pair of similar patches has been

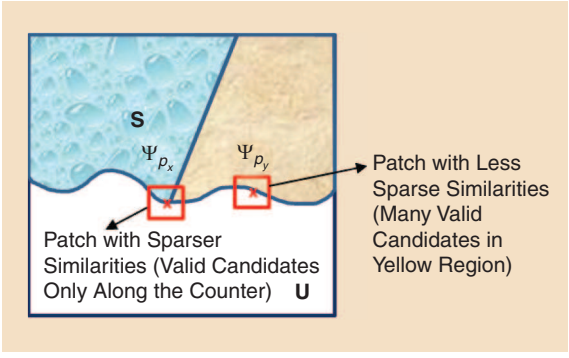
[TABLE 1] PATCH FILLING ORDER.

THE DATA TERM $D(p_x)$ USED FOR FAVORING PATCHES WITH STRONGER STRUCTURES CAN TAKE DIFFERENT FORMS.

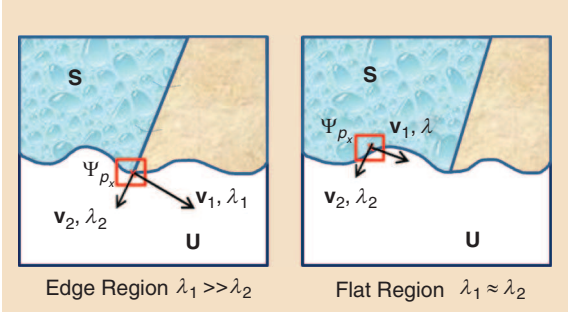
1) GRADIENT-BASED DATA TERM [31]: $D(p_x) = (|\nabla \mathbf{I}^{\perp}_{p_x} . \mathbf{n}_{p_x}|)/\alpha$, WHERE $\mathbf{n}_{p_x}$ IS THE NORMAL TO THE FRONT LINE AT THE POSITION OF THE PIXEL p_x, AND THE PERPENDICULAR TO THE GRADIENT $\nabla \mathbf{I}^{\perp}_{p_x}$ REPRESENTS THE ISOPHOTE.



2) SPARSITY-BASED STRUCTURE CONFIDENCE TERM [32]: $D(p_x) = \|\mathbf{w}_{p_x}\|_2 \times \sqrt{N_s/N}$, WHERE N_s AND N ARE THE NUMBERS OF VALID AND CANDIDATE PATCHES IN THE SEARCH WINDOW.



3) TENSOR-BASED PRIORITY [8], [33]: $D(p_x) = \alpha + (1-\alpha)exp(-(\eta)/(\lambda_1 - \lambda_2)^2)$, WHERE $\eta \geq 0$ ($\eta = 8$) AND $\alpha \in [0,1]$ ($\alpha = 0.01$). WHEN $\lambda_1 \approx \lambda_2$, FLAT REGION IMPLIES LOW PRIORITY ($D(p_x)$ SMALL). WHEN $D(p_x)$ $\lambda_1 >> \lambda_2$, PRESENCE OF STRUCTURE IMPLIES HIGH PRIORITY (TENDS TO BE "1").



found in two images, then their neighbors in the image plane (patches shifted by a few pixels) are likely to be also similar. Therefore, the matching result of a query patch can be propagated to the nearby queries, providing a good initial guess that is then updated by some randomly sampled candidates.

PatchMatch is a fast algorithm for computing dense approximate NN correspondences between patches of two image regions. The set of approximate correspondences is called an NN field (NNF). The method searches for the NNF by proceeding as follows. The NNF is first initialized either by random values or prior information. Initializing the NNF with a random guess is likely to provide few good guesses. The NNF is then iteratively refined by interleaving two operations called propagation and random search at the patch level. The propagation step updates a patch offset with the known offsets of its causal neighborhood, exploiting the image coherency. At even iterations, offsets are propagated from up and left patches, whereas at odd iterations, offsets are propagated from right and bottom patches. The second operation performs a local random search to seed the patch matches, and these matches are then propagated by iterating a small number of times.

The algorithm was shown to be much faster than kd-trees with, however, less accuracy. It can be trapped in a local optimum due to the short-distance propagation. To improve the NN search accuracy, the random search step of PatchMatch is replaced with a hashing scheme in [29]. Good matches are propagated to nearby patches as well as to similar patches that were hashed to the same value (i.e., which are similar in appearance). The algorithm therefore runs faster and turns out to be more accurate. The PatchMatch algorithm has been generalized in [30] to find K-NNs instead of one, and to extend the search space with transformations (rotations, scaling).

PATCH PROCESSING ORDER

The missing regions in an image may be, in general, composed of textures and structures. It has been observed [31] that it was important to separate these two components and start by first recovering the structures. This led to proposing patch processing orders that are defined in such a way that patches on structures are filled in first. In general, the processing order is given by a patch priority measure defined as the product of two terms ($P(p_x) = C(p_x)D(p_x)$). The first term accounts for the amount of known pixels versus unknowns in the input patch [this is a so-called confidence term $C(p_x)$] and the second term $D(p_x)$, called data term, reflects the presence of some structure in the patch. This data term can take several forms (see Table 1).

A gradient-based data term is proposed in [31], which favors patches in which the isophote is perpendicular to the front line at pixel p_x. The data term is defined as the absolute value of the inner product between the isophote direction (perpendicular to the gradient $\nabla \mathbf{I}^{\perp}_{p_x}$) and the normal $\mathbf{n}_{p_x}$ to the front line as

$$D(p_x) = \frac{|\nabla \mathbf{I}^{\perp}_{p_x} . \mathbf{n}_{p_x}|}{\alpha}. \tag{13}$$

Thanks to this priority term, unknown pixels at the edge of an object have higher priority than pixels located on flat image areas (see Table 1).

A sparsity-based data term has been proposed by Xu et al. [32] to measure the structure confidence of patches. The structure confidence is based on the sparseness of nonzero patch similarities. Structural patches are assumed to have sparser

nonzero similarities with its neighboring patches compared to textural patches. This assumption stems from the observation that structures (edges, corners) are, in general, sparsely distributed in the image. A similarity weight w_{p_x,p_j} (i.e., proportional to the similarity between the two patches centered on p_x and p_j) is computed for each pair of patches. A vector $\mathbf{w}_{p_x}$ is formed with the similarity weights between all the pairs (p_x, p_j) where $\mathbf{j} = (i_1, j_1), \ldots (i_N, j_N)$, with N being the number of all candidate patches within a search window. The sparsity-based data term is defined as

$$D(p_x) = \|\mathbf{w}_{p_x}\|_2 \times \sqrt{\frac{N_s}{N}}, \tag{14}$$

where N_s and N are the numbers of valid (all their pixels are known) and candidate patches in the search window centered on the pixel p_x.

A large value of the structure sparsity term $D(p_x)$ means sparse similarity with neighboring patches—a good confidence that the input patch is on some structure—whereas a small value indicates that the current input patch is highly predictable by many candidates, hence is likely to be a textural patch. This is illustrated in Table 1, where the patch Ψ_{p_x} has sparser similarity than the patch Ψ_{p_j} for which a larger number of matching patches can be found in the uniform yellow region. This term, compared to the one defined on isophotes, better distinguishes structural from textural patches.

A tensor-based data term is proposed in [33] based on Di Zenzo's structure tensor [4] computed on the color components (R,G,B) as $\mathbf{J} = \sum_{c=1}^{m} \nabla \mathbf{I}^c \nabla \mathbf{I}^{cT}$. As explained above, the eigenvectors and eigenvalues of the structure tensor give an indication of the local geometry. Based on the discrepancy of the eigenvalues, the degree of anisotropy of a local region can be evaluated. The tensor-based data term is therefore written as

$$D(p_x) = \alpha + (1-\alpha)\exp\left(-\frac{\eta}{(\lambda_1 - \lambda_2)^2}\right), \tag{15}$$

where $\eta \geq 0$ (e.g., $\eta = 8$) and $\alpha \in [0,1]$ (e.g., $\alpha = 0.01$). On flat regions ($\lambda_1 \approx \lambda_2$), no direction is favored for the propagation (isotropic filling order). When $\lambda_1 >> \lambda_2$, which indicates the presence of a structure, the data term $D(p_x)$ tends to be "1."

The patch processing order has a strong impact on the quality of the inpainted image: giving a higher priority to structural patches leads to a better recovery of object boundaries, avoiding, e.g., unconnected boundaries in the inpainted image, as shown in Figure 6. Studies of vision psychology show that human observers are quite annoyed by unconnected edges [34].

GLOBAL IMAGE COHERENCE VIA CONSTRAINED SEARCH OR GLOBAL OPTIMIZATION

The approaches in [3], [24], and [31] progress patch per patch in a greedy fashion; for this reason, they do not ensure a global image coherence. The visual quality of the inpainted image can be improved by maximizing similarity between the synthesized patch and original patches in the known part of the image. This can be achieved in pixel-based synthesis approaches by searching for candidate pixels in the neighborhood of those already used to inpaint the neighbors of the input pixel to be estimated [19].

The authors in [35] constrain the search for candidate patches by using a global coherence measure $\mathrm{Coh}()$ defined as

$$\mathrm{Coh}(\Psi^U_{p_x}) = \min_{p_j \in S} \{d_{\mathrm{SSD}}(\Psi^U_{p_x}, \Psi^U_{p_j})\}. \tag{16}$$

This coherence measure is used to favor patches giving a synthesized part $\Psi^U_{p_x}$ most similar to the pixels $\Psi^U_{p_j}$ of the known part S of the image. Among the best candidate patches, the algorithm keeps the one that minimizes the above coherence measure. This prevents from pasting in the unknown region a texture that would be too different from original textures. This measure therefore limits texture garbage. Examples of texture garbage are visible in Figure 6.

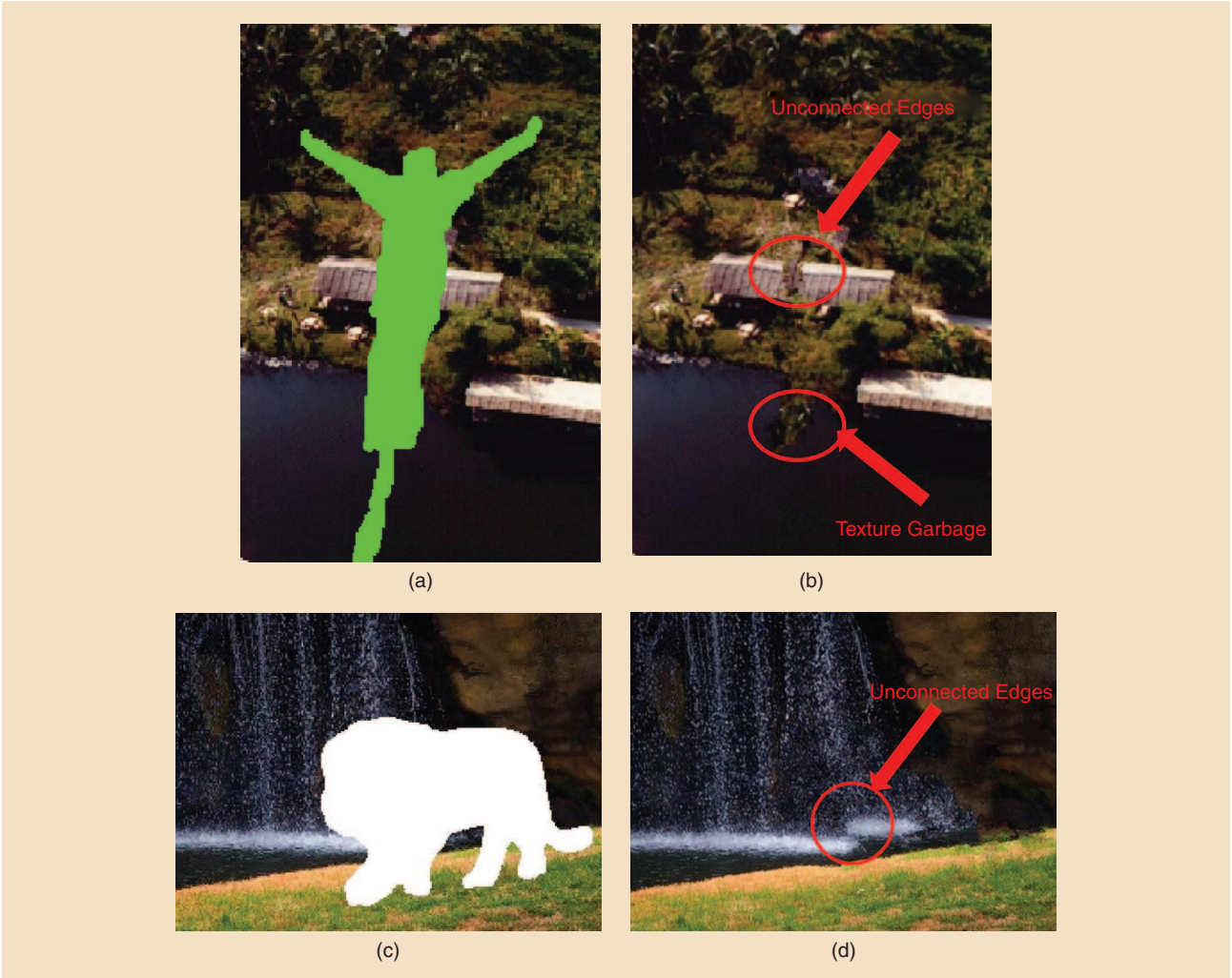
Spatial coherence can also be naturally ensured via a global optimization of MRF energy functions over the entire image. Patch, pixel locations, or offsets are optimized in the MRF by using belief propagation [36] or graph cuts [37]. The shift-map method [37] indeed treats inpainting as a global optimization on the entire image. The method, inspired from the patch transform described in [38], computes a vector field (called shift-map), which maps each pixel in the hole to be filled in to a pixel in the known part of the image. The optimal shift-map is computed as a graph labeling optimization minimizing a global cost function. A node in the graph corresponds to a pixel in the output image labeled by a shift (translational displacement).

The authors in [39] match similar patches in the known part of the image and compute patch offsets. They observed that a majority of patches have similar offsets, and the peaks in the offset distribution correspond to dominant offsets. A stack of images is formed with shifted versions of the input image according to these dominant offsets. The shifted images are then combined by optimizing a global MRF energy function. These methods produce visually pleasing and coherent inpainted images, in particular when the hole to be filled in has homogeneous texture and few structures.

PATCH STITCHING WITH BLENDING AND QUILTING

Filling the unknown part of the input patch may lead to stitching together pieces of texture that are not consistent in terms of color or contrast, or which may contain structures that do not align. Color bleeding or boundary artifacts may therefore be visible in a boundary or transition region. To reduce the boundary effect, the pieces of texture must be stitched together along an optimal seam that will be the best boundary between the two regions.

A quilting method is introduced in [40], which consists of finding an optimal path (called a *seam*) cutting the overlapped regions O, as illustrated in Figure 7. The seam, which is an eight-connected path of pixels in the overlap region, goes from top to bottom for a vertical seam and from left to right for an horizontal seam. The path determines which patch contributes to pixels at different positions in the overlap region. A vertical (respectively, horizontal) seam contains only one pixel per row



[FIG6] **(a)–(d) Typical artifacts of examplar-based methods: unconnected boundaries and texture garbage. The bungee jumper has been inpainted with the method in [31] and the lion image with [22]. [(a) Original bungee jumper image courtesy of M. Bertalmio. (b) and (d) Courtesy of Christine Guillemot and Olivier Le Meur. (c) Courtesy of [39].]**

(respectively, column) in the overlap region. An energy function $e(i, j)$ is defined to evaluate the contrast of the current pixel with respect to its neighbors. A gradient-based energy is commonly used. The energy of a seam s is given by $E(s) = \sum_{(i,j)\in O} e(i, j)$, where (i, j) are the coordinates of a pixel of the seam s inside the overlap region O. The optimal seam s^* having the minimum energy

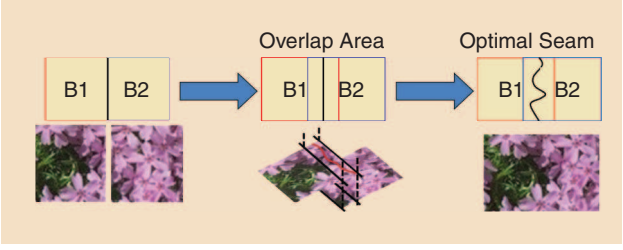
$$s^* = \arg\min_s E(s) \tag{17}$$

is searched by using Dijkstra's shortest path algorithm, dynamic programming [40], or by using graph cuts [23], [41].

Dynamic programming allows solving this problem efficiently by computing the cumulative minimum energy M for all possible connected seams as

$$M(i, j) = e(i, j) + \min(M(i-1, j-1), M(i-1, j), M(i-1, j+1)). \tag{18}$$

For a vertical seam, we look for the minimum value on the last row and backtrack from this minimum to find the optimal vertical path. The quilting approach is limited by the number of possible orientations at each pixel. For instance, for the case of a vertical seam, only three directions (bottom-right, bottom, and bottom-left) can be used to build the seam path. In



[FIG7] **The quilting texture. Two texture chunks are patched together. An optimal seam is computed to cut the overlap region. On the left side of the seam, texture from block *B1* is copied into the final texture. On the right side, pixels belonging to texture *B2* are copied. (Figure adapted from [40] and used with permission from the Massachusetts Institute of Technology.)**

addition, the quilting method is greedy since the seams are defined either in left-to-right or top-to-bottom order.

To cope with these limitations, a graph-cut method can be used to find the optimal seam. A graph cuts consist of first constructing a specialized graph for the energy function to be minimized such that the minimum cut on the graph, computed by max-flow algorithms, minimizes the energy [42], [43]. Graph cuts have been used for a wide variety of vision problems, particularly in [44] to seamlessly combine and stitch different textures together.

Blending (feathering, pyramid blending, alpha blending, or Poisson blending [45]) and image melding [46] can also be used to seamlessly merge the new patch. Blending aims at handling color inconsistency by smoothly interpolating the error in the transition region. Image melding [46] synthesizes the texture in the transition region by transforming one patch (or image) to be stitched into the other.

EMBEDDING NEAREST NEIGHBORS

The first examplar-based methods were estimating the unknown part of the input patch by copying pixels of the single best match among possible candidates. A better estimate of the unknown pixels can be obtained by computing a linear combination of several candidate patches, e.g., of the K-NN, as illustrated in Figure 8. This places the inpainting problem in a neighbor embedding framework. Different approaches can be used to compute the weights of the linear combination of the K-NN. The authors in [47] proposed a nonlocal means approach. The unknown pixels of the patch to be completed are inferred by the nonlocal means of several candidate patches instead of taking the single best match. The weights of the linear combination are computed using a similarity kernel as

$$w_k = \exp\left(-\frac{\| \Psi^S_{p_x} - \Psi^S_{p_j} \|_2^2}{h}\right), \ \mathrm{j} = (i_k, j_k), k = 1 \ldots N, \quad (19)$$

where N is the number of all candidate patches in the search window. The parameter h acts as a filtering parameter. This kernel gives higher weights to patches that are more similar to the known samples of the input patch. This approach is inspired from the nonlocal means (NLM) algorithm used for denoising in [48] and for texture synthesis in [3].

The weights can also be computed using least square approximations of the known pixels of the input patch under various constraints. A constraint that the weights sum to one leads to placing the problem in the locally linear embedding (LLE) framework [49]. The search for the weights of the linear combination of the K best matching patches, centered on pixels p_j located at positions $\mathrm{j} = (i_k, j_k), k = 1 \ldots K$, is expressed as

$$\mathrm{argmin}_{w_k,\, k=1 \ldots K} \left\| \Psi^S_{p_x} - \sum_k^K w_k \Psi^S_{p_j} \right\|_2^2 \quad \text{s.t.} \sum_k^K w_k = 1, \mathrm{j} = (i_k, j_k), k = 1 \ldots K. \quad (20)$$

The nonnegativity constraint places the problem in the nonnegative matrix factorization [50] framework, which leads to formulating the weight computation as the constrained minimization

$$\mathrm{argmin}_{w_k,\, k=1 \ldots N} \left\| \Psi^S_{p_x} - \sum_k^K w_k \Psi^S_{p_j} \right\|_2^2 \quad \text{s.t.} \ w_k \geq 0, \mathrm{j} = (i_k, j_k), k = 1 \ldots K. \quad (21)$$

Once the weights w_k are computed, they are used to linearly combine the pixels of the candidates patches $\Psi^U_{p_j}$, which are located at the same position as the unknown pixels of the input patch (i.e., $\hat{\Psi}^U_{p_x} = \sum_k^K w_k \Psi^U_{p_j}$).

MULTISCALE REFINEMENT

The search for similar patches can be improved by introducing as a priori a rough estimate of the inpainted values using a multiscale approach. The missing regions are iteratively approximated using some guidance from coarse to fine levels [24]. The use of a multiscale approach offers a number of key advantages. The inpainting of a coarse version of the input picture is much easier than performing the inpainting at the full resolution. It is easier to retrieve the main and dominant structures since the local singularities in terms of orientation as well as luminance are less numerous. The results are moreover less sensitive to noise. Drori et al. [24] used, for instance, a rough estimate of the inpainted values to improve the K-NN search method. The drawback of multiscale refinement, as mentioned in [36], is that if an error occurs at a coarse scale, inpainting errors can spread across the finer scale. The authors in [22] perform the inpainting on a coarse resolution of the input picture, and then use a single-image superresolution method to retrieve the high-frequency details of the inpainted areas.

INTRODUCING SPARSITY PRIORS

The inpainting problem can also be solved assuming image sparsity priors. In this case, the image $\mathbf{I}$ is assumed to be a sparse-land signal [51], meaning that the image $\mathbf{I}$ is sparse in a given basis. The basis can be formed by predefined elementary waveforms (also called atoms) which are stored in a dictionary matrix $\mathbf{A}$. The dictionary matrix $\mathbf{A}$ can also be learned using dictionary-learning methods. An image $\mathbf{I}$ is said to be k-sparse in a given basis stored in the dictionary matrix $\mathbf{A}$, if it can be represented by a vector $\mathbf{v}$ having only k nonzero elements (i.e., its l_0 norm is $\|\mathbf{v}\|_0 = k$) verifying $\mathbf{I} = \mathbf{A}\mathbf{v}$. The sparsity of a signal depends on the considered basis, i.e., of the matrix $\mathbf{A}$.

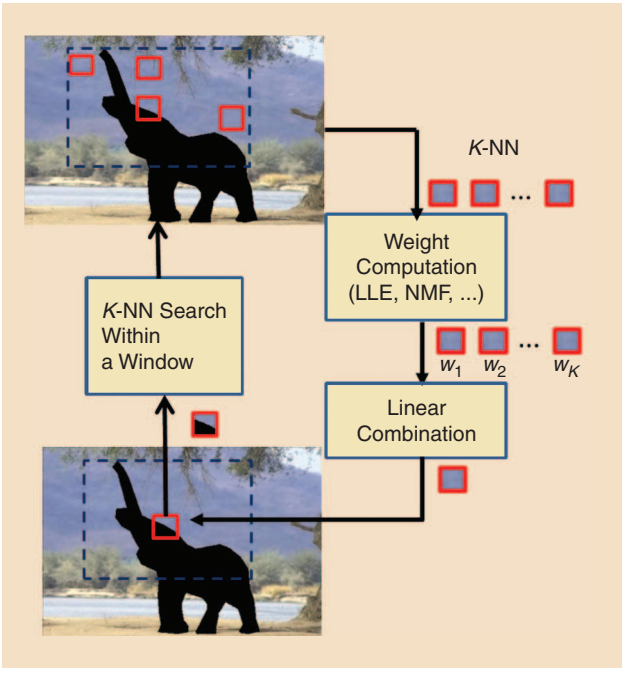
Considering the degraded image $\mathbf{F} = \mathbf{M}\mathbf{I}$, the inpainting problem is therefore formulated as searching for the sparse representation vector $\mathbf{v}$ of the image $\mathbf{F}$, by solving

$$\min \ \|\mathbf{v}\|_0$$

such that

$$\mathbf{F} = \mathbf{M}\mathbf{A}\mathbf{v}. \quad (22)$$

Many solutions exist for searching for the sparse vector $\mathbf{v}$, with the most popular ones belonging to the family of greedy matching pursuit algorithms. A good overview of these matching pursuit algorithms can be found in [51, Ch. 3].



[FIG8] The estimation of unknown pixels with neighbor embedding. The first step consists in searching for the *K*-NN of the input patch. The algorithm then searches for the weights of the linear combination of these *K*-NN patches that best approximates the known pixels of the input patch to be completed. The weights computation can be done using different methods (e.g., NLM, LLE, and NMF).

The above inverse problem is usually solved patch-per-patch rather than directly on the entire image $\mathbf{F}$. For each patch Ψ_{p_x} of the image $\mathbf{F}$ formed by a known part $\Psi^S_{p_x}$ and an unknown part $\Psi^U_{p_x}$, one searches for the sparse vector $\mathbf{v}_{p_x}$, which best approximates the known part of the input patch $\Psi^S_{p_x}$ as $\Psi^S_{p_x} = \mathbf{A}^S \mathbf{v}_{p_x}$ where $\mathbf{A}^S$ is a matrix obtained by masking the rows of the matrix $\mathbf{A}$ corresponding to the positions of the unknown pixels $\Psi^U_{p_x}$ in Ψ_{p_x}, as shown in Figure 9.

The same sparse linear combination of atoms is then used to approximate the unknown pixels $\Psi^U_{p_x}$, as $\hat{\Psi}^U_{p_x} = \mathbf{A}^U \mathbf{v}_{p_x}$, taking this time the masked samples of the atoms (these samples correspond to the positions of the unknown pixels). This general formulation is also used for other image processing problems like denoising and superresolution. The sparse representation area has recently evolved into the more general compressive sampling framework that also naturally applies to image recovery problems [52].

Variants have been introduced exploiting sparsity priors. Assuming that images are composed of locally uniform regions separated by edges, the author in [53] uses adaptive sparse representations. The algorithm performs a nonlinear approximation with adaptively determined sparsity constraints. Hybrid sparse representations enforcing both local and nonlocal sparsity are considered in [54]. The nonlocal sparsity is defined as the sparsity of a three-dimensional (3-D) data array formed by the input patch and its K-NN in the known part of the image, whereas the local sparsity is defined as the sparsity of the 2-D patch, the sparsity constraints being enforced by hard-thresholding in a predefined waveform basis (DCT, fast Fourier transform). Local and nonlocal sparse representations are then combined via Bayesian model averaging [54] to satisfy both constraints of local smoothness and nonlocal similarity, with a constraint of fidelity to the known samples.

Patch-based methods show that texture patches can be relevant dictionary elements. Therefore, instead of using predefined waveforms, a linear combination of candidate patches regularized by a sparseness prior on the weighting coefficients can be used for inferring the unknown pixels [32]. Sparsity is also used in [32] for determining structural patches to be processed first, as explained in the section "Patch Processing Order." The patch sparse representation is moreover constrained by local patch consistency.

HYBRID METHODS SEPARATING STRUCTURE FROM TEXTURE

Diffusion methods work well for small and sparsely distributed gaps. They are also appropriate for piecewise smooth images and for propagating strong structures. But they are unable to restore texture. On the contrary, examplar-based methods work amazingly well in textured regions with homogeneous or regular patterns. Nevertheless, they are not so well suited for preserving edges or structures, or for images with many small distributed holes.

Yet, natural images contain composite structures and textures. The structures constitute primal sketches of an image (e.g., edges, corners) and textures are regions with homogenous patterns or feature statistics. To handle composite textures and structures, it is therefore natural to combine different types of approaches. Two main strategies have been considered. The first strategy consists in first separating the image components (texture and structure), and inpainting them separately with the most suitable method (e.g., diffusion or examplar-based). The two inpainted components are then added together as in [55], [56]. A second strategy consists in combining different approaches in one unique energy function using a variational formulation [21], [36].

STRUCTURE/TEXTURE SEPARATION

Structure can be identified in a supervised way, as in [57], where the user specifies curves corresponding to important missing structures (e.g., object boundaries) in the unknown region. A structure propagation is then performed by copying patches located in the direction of these curves in the known region. The remaining unknown pixels are in a third step estimated using a texture synthesis method as in [19].

Texture and structure can also be separated in an automatic manner, using, for example, a variational method as in [55], where the authors decompose the image as a sum of two functions, one being of BV representing the image structure, and a second one capturing the texture. The structure image is a sketchy approximation of the input image, containing only edges separating smooth regions. These piecewise smooth

images are also referred to as cartoon images in [55]. In [55], the texture layer is inpainted using the texture synthesis method of [3] while the geometric layer is inpainted using the diffusion method of [2], and the two inpainted components are added together to produce the final result.

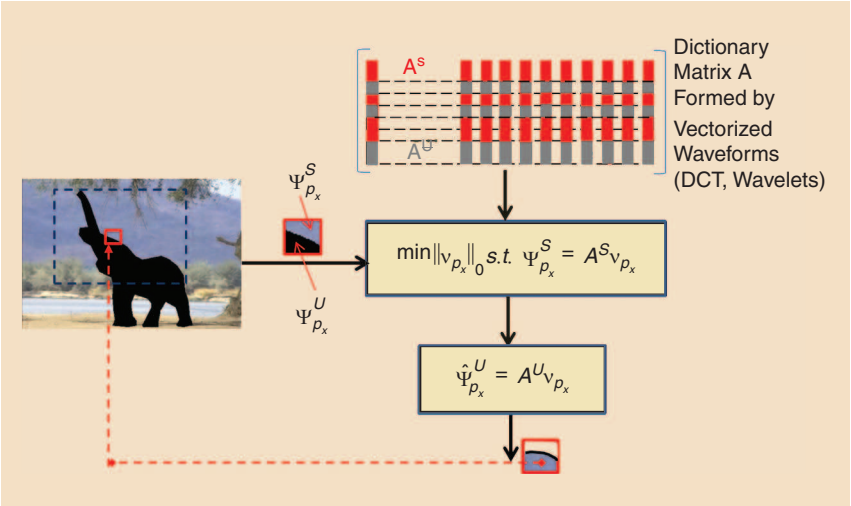
The authors in [56] proposed a method based on sparse representations for decomposing the image into a texture and a geometry component, called layers. Using two dictionaries of different characteristics A_g and A_t, the image is decomposed into structural and textural components as $\mathrm{I} = \mathrm{A}_g \mathrm{v}_g + \mathrm{A}_t \mathrm{v}_t$. The method is called morphological component analysis (MCA). The two dictionaries are mutually incoherent, i.e., each dictionary gives a sparse representation for one component while yielding a nonsparse representation for the other component. Both dictionaries are grouped into a big dictionary which is then used by a basis pursuit algorithm to find the sparse representation of each layer. The authors in [58] propose an algorithm based on this sparsity seeking image separation into two components. Instead of a separate processing of the two components as in [55], the sparse vectors for the two components are obtained by minimizing

$$\min \; ||\mathrm{v}_g||_p + ||\mathrm{v}_t||_p$$

such that

$$\mathrm{F} = \mathrm{M}(\mathrm{A}_g \mathrm{v}_g + \mathrm{A}_t \mathrm{v}_t), \tag{23}$$

where $||.||_p$ denotes the L_p norm, with p often equal to zero or one. To solve this minimization problem, the constraint is introduced as a penalization term. In [58], a TV penalization term is added to regularize the sparse approximation of the image geometry. This approach can fill in a region with composite textures and structures. It however introduces blur when the missing region is large.



[FIG9] Estimation of unknown pixels with sparse priors in transform domain. A dictionary matrix is constructed from waveforms (DCT, wavelets). The grey part of the columns of the dictionary represents the masked rows corresponding to the position of the unknown pixels in the input patch.

COMBINING DIFFERENT METHODS VIA ONE ENERGY TERM

Methods best adapted to different components of the image can also be combined by introducing several energy terms in one unique energy function which can then be globally optimized, without prior separation of structure from texture. In [36], an energy term comprising a texture synthesis term and a term measuring how patches to be stitched together agree in the overlap area is minimized using belief propagation. The authors in [21] combine energy terms related to texture synthesis, coherence, and geometry (by minimizing the TV of the structure of the image) as proposed in [59], into one single energy functional. A correspondence map between pixels to be filled in and pixels in the known part of the image, is then searched to minimize this energy functional which is the sum of three energy measures (self-similarity, coherence, and diffusion). The self-similarity energy term, as in texture synthesis, computes the similarity between the patch centered on the pixel to be filled in and the patch centered on the candidate pixel in the known part of the image. The diffusion term is the energy of the discrete Laplacian of the inpainted part of the image for a given correspondence map. The spatial coherence term measures the similarity between patches corresponding to neighboring pixels.

GLOBAL METHODS

In ill-posed image processing problems such as inpainting, image priors play a very critical role. Statistical and structural priors capture and exploit stationarity and similarity in a local neighborhood or throughout the entire image. In patch-based methods, the missing pixels are patch-wise computed in terms of their neighbors, whereas in diffusion pixel-based methods, they are filled in by propagating neighboring pixels in a way that favors good edge continuation. Rather than searching to capture local relationships or dependencies, one can instead capture the global structure of the input data, using models which reproduce key statistical properties of images or of textures of interest. These models include probabilistic models of coefficients in transformed domains [60], sparsity models (see the section "Introducing Sparsity Priors") or low-rank models.

When using sparse priors, the input texture or image is assumed to have sparse representations in a certain basis. Forcing sparse priors on the recovered image, the missing region is synthesized as a sparse linear combination of elements from an overcomplete dictionary. Similarly, given the spatial coherence and self-similarity which characterize natural images, the high-dimensional input textures or images lie in a subspace of reduced dimension. This low-rank image model has led to a

class of methods based on low-rank matrix and tensor completion, which aim at representing this subspace characterizing the "global" information of the image.

Let $\mathbf{I}$ be the input image to be completed. Let us assume that the image $\mathbf{I}$ is of dimension $P \times Q$ in which each pixel in position $\mathbf{x} = (x, y)$ carries three color components (i.e., $\mathbf{I} \in \mathcal{R}^{P \times Q \times 3}$). Each image color channel of $\mathbf{I}$ forms a 2-D matrix $\mathbf{I}^c$ with known and unknown entries. The matrix completion problem is to search for a low-rank approximation of the matrix $\mathbf{I}^c$, given the known subset of its entries. This problem is formulated as

$$\min_{\hat{\mathbf{I}}^c} \frac{1}{2} |\hat{\mathbf{I}}^c - \mathbf{I}^c|_{\mathbf{S}}^2 \text{ s.t. } \operatorname{rank}(\hat{\mathbf{I}}^c) \leq r, \tag{24}$$

where $\mathbf{S}$ is the set of known pixels in the image (known entries in the image), and the rank is the number of nonvanishing singular values of the matrix. This optimization is not convex. The problem can however be reformulated as a convex programming problem by minimizing the sum of the singular values, i.e., the nuclear (or trace) norm instead of the rank [61]. The problem is in this case of finding the matrix with the minimum nuclear norm agreeing with the observed entries.

Given that a tensor is simply a generalization of a matrix to higher dimensions, the low-rank matrix completion problem has been naturally extended to low-rank tensor completion. The order of a tensor is the number of dimensions also known as ways or modes. The three color component input image $\mathbf{I}$ can therefore be seen as a three-order tensor to be completed. The problem can be formulated in a similar manner as in (24), and solved using the notion of trace norm for tensors as in [62].

Matrix and tensor completion methods work well for inpainting when the missing areas are not too large or when the rank of the original image is quite low. It is hence suitable for restoring images from scratches, or for removing overlayed text, but not so well suited for applications such as disocclusion, object removal or loss concealment.

APPLICATIONS

The problem of inpainting is encountered in various image processing applications: image restoration, editing (e.g., object removal), disocclusion in image-based rendering, interpolation, loss concealment, texture synthesis or image resizing (e.g., enlargement). Inpainting has also been considered in the context of lossy image compression: blocks within the image that can be recovered by inpainting are not transmitted. The goal here is not to assess or benchmark in terms of inpainting results the numerous methods which exist, nor to give an exhausive list of all potential applications. It is instead to illustrate the main applications with some examples of algorithms, showing the limited applicability of some of them for particular use cases. A taxonomy of the methods is given in Table 2. Links to publicly available software code implementing some of the above methods are given in Table 3.

IMAGE RESTORATION

The image restoration problem is concerned with recovering an original image from various forms of degradations. The origin of the degradations to deal with depends on the application: it can be text overlay or scratches in digital photography, in digital cinema, or in pictures taken of ancient paintings [63]. It can also be degradations that result from the capturing process like specular reflections, spots, and cracks in medical images (e.g., in endoscopic images [64]). Another application area is fingerprint restoration in automatic fingerprint identification systems [65].

In the restoration problem, the missing region is generally not too large, hence, local diffusion and patch-based or global methods give satisfactory results. Figure 10 illustrates the problem and gives inpainting results obtained with methods of different types: anisotropic diffusion [11], examplar-based inpainting, the hybrid method in [21], which performs a global minimization of an energy term, and the tensor completion method [62]. The examplar-based method used here is a simple approach computing the patch processing order as in [31], but using similarity weights [47] to combine candidate patches rather than inferring the unknown pixels from the single best match. Given the small gaps to be filled in, most methods (diffusion, examplar, a fortiori combinations of the two, as well as more global techniques) give satisfactory results.

OBJECT REMOVAL

Another natural application of inpainting is image editing in which the user removes objects, hence, uncovering unkown parts of the image foreground. This application is well illustrated by the images of Figure 11 in which one foreground object has been removed, leaving a hole to be filled. Figure 11 shows inpainting results with methods of different categories. Figure 11(b) illustrates the limitations of diffusion methods when the gap to be filled in is large. Diffusion introduces smoothing and blurring artifacts in the synthesized region.

[TABLE 2] A TAXONOMY OF INPAINTING METHODS.

FEATURES	PDE-BASED DIFFUSION	EXAMPLAR-BASED INPAINTING	HYBRID METHODS	GLOBAL
PRIORS	SMOOTHNESS	SELF-SIMILARITY, SPARSITY	SMOOTHNESS + SIMILARITY/SPARSITY	STATISTICAL, LOW RANK
OPTIMIZATION	GREEDY	GREEDY OR GLOBAL	GREEDY OR GLOBAL	GLOBAL
SENSITIVITY TO SETTING	LOW	HIGH	HIGH	HIGH
HOLES	SMALL	MEDIUM TO BIG	MEDIUM TO BIG	SMALL TO MEDIUM
APPLICATIONS	RESTORATION	RESTORATION, EDITING, DISOCCLUSION, CONCEALMENT	RESTORATION, EDITING DISOCCLUSION, CONCEALMENT	RESTORATION

To recover the texture of the hole, examplar-based methods [66], methods using sparse representations [32], or solutions combining structure diffusion and examplar-based texture recovery as, e.g., [21] are more appropriate. However, examplar-based techniques, although conceptually very simple, work strikingly well for these type of applications. The method used in Figure 11(c) uses the same patch processing order as in [31] but performs a linear combination of best matching patches with weights computed using the constrained least squares approximation given in (20). The method in [39], based on statistics of patch offsets, also gives very good results in this case. However, global methods based on low-rank approximation or tensor-based completion [62] are, in general, not suitable for this type of application, except if the hole is small or the input image is low rank.

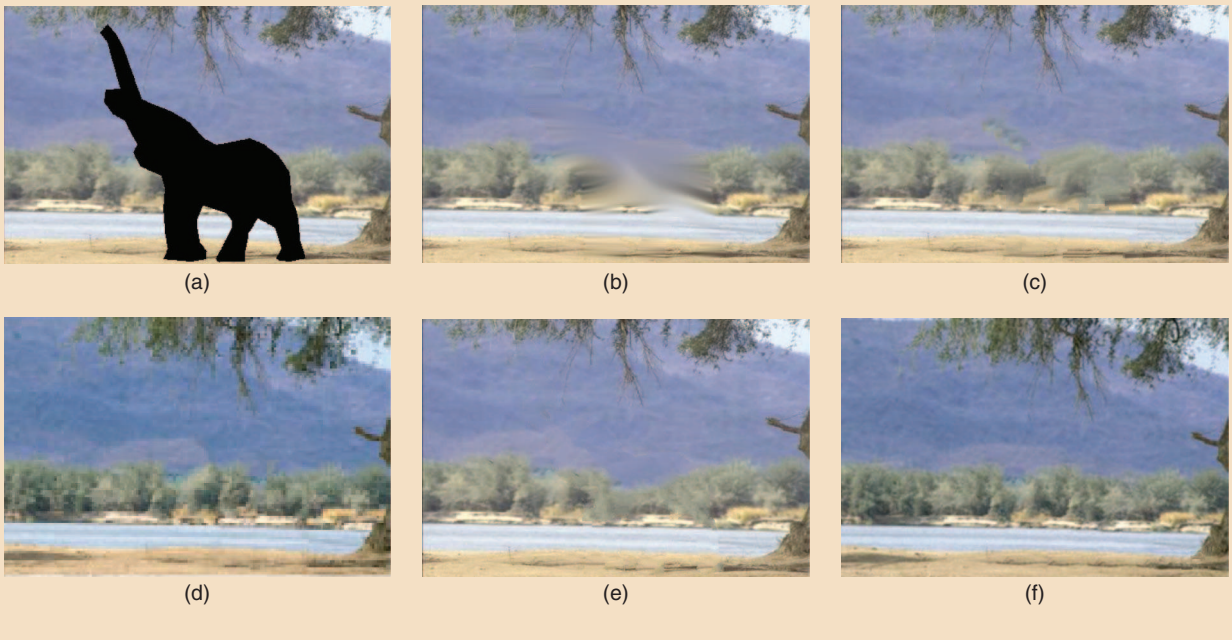
[TABLE 3] LINKS TO PUBLICLY AVAILABLE CODE.

METHODS	URL
PDE-BASED DIFFUSION	FAST MARCHING DIFFUSION [10] (IN OPENCV): http://docs.opencv.org/modules/photo/doc/inpainting.html
EXAMPLAR BASED	THIRD PARTY CODE OF [31]: http://www.csee.wvu.edu/~xinl/source.html http://daviddoria.com/?p=126 [22]:http://people.irisa.fr/Olivier.Le_Meur/publi/2013_TIP/index.html
SPARSE METHODS	[51]: http://www.cs.technion.ac.il/~elad/software/ ADAPTIVE [53]: http://eeweb.poly.edu/~onur/source.html#recover_code LOCAL/NONLOCAL [54]: http://www.csee.wvu.edu/~xinl/demo/inpainting.htmL TEXTURE/STRUCTURE SEPARATION (MCA) [56]: http://jstarck.free.fr/mca.html
GLOBAL/HYBRID METHODS	TENSOR COMPLETION [62]: http://pages.cs.wisc.edu/~ji-liu/

DISOCCLUSION

Inpainting methods are also needed in 3DTV rendering on stereoscopic or autostereocopic displays, as well as in the context of free viewpoint rendering of a 3-D scene. For the user to navigate in the 3-D scene, virtual views that may be distant from original ones must be synthesized. The intermediate views (or virtual views) are usually computed with IBR algorithms that use the original images, scene geometry information (depth maps), and camera parameters. A 3-D point on the reconstructed geometry is projected onto the image plane of a virtual camera. The color of the 3-D point is first computed by inverse projection of the original image into the 3-D space. During this projection process, some parts of the 3-D scene may be visible in the virtual view but not in the original views: they may be

[FIG10] **Image restoration. (a) The original image (courtesy of [62]), (b) mask, (c) inpainting results with anisotropic diffusion, (d) examplar-based method with similarity weights (NLM), (e) with the hybrid method (courtesy of [21]), and (f) with the tensor completion technique (courtesy of [62]).**

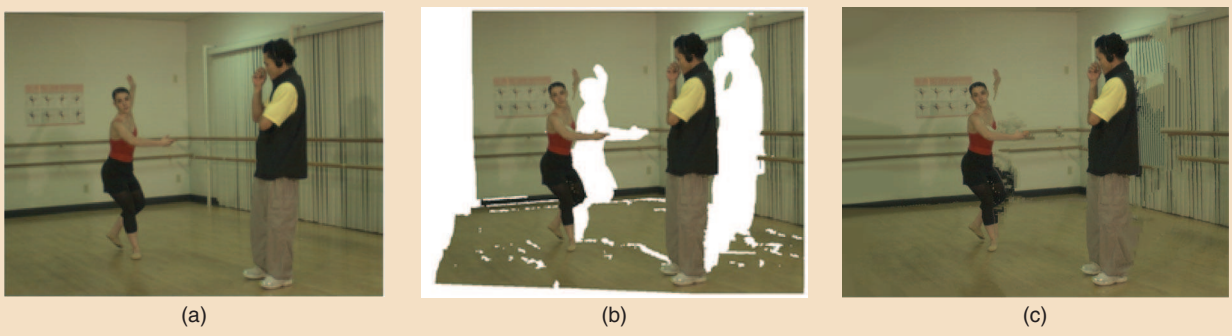


[FIG11] Object removal application (a) mask and inpainting results with methods from different categories, (b) anisotropic diffusion (courtesy of [11]), (c) examplar-based with LLE (courtesy of [66]), (d) patch sparse representation (courtesy of [32]), (e) hybrid with one global energy minimization (courtesy of [21]), and (f) patch offsets (courtesy of [39]). [(a) courtesy of www.magazinehive.com.]

hidden by foreground objects. When synthesizing a virtual view, these parts become disoccluded, resulting in pixels with unknown color. These pixels need to be estimated using inpainting techniques.

Figure 12 illustrates this application by showing in (a) a reference view (view four of the multiview sequence called ballet (see [67]) and in (c) the result of the projection on a virtual viewpoint corresponding to a displacement to the right of the camera. Disoccluded areas due to the projection appear and have to be filled in. When the camera is moving to the right, missing areas (e.g., disocclusion areas) appear on the right side of foreground objects. Therefore, to prevent the propagation of foreground patches into the background, the filling has to be performed from the right to the left side. Otherwise for a leftward shift of the camera, the missing areas should be filled in from the left to the right side. Figure 12(c) shows the inpainted result using the examplar-based method in [66] adapted so that the candidate patches are searched in a window shifted according to the camera displacement (i.e., shifted to the right for a rightward camera shift).

The gaps to be filled in do not result from explicit object removal as in an image editing application. However, the difficulties and characteristics, as far as the inpainting algorithm is concerned, are very similar for both applications. In this application, the inpainting can nevertheless also benefit from taking into account the depth information in addition to the three color components, e.g., for finding best matching patches.



[FIG12] The view synthesis for the sequence ballet. (a) The known reference view projected into virtual viewpoints. (b) Projection of the reference view on a virtual viewpoint with rightward shift of the camera (white areas are disoccluded areas). (c) The inpainted version of the projected image. (Images used courtesy of [67].)

LOSS CONCEALMENT

Image and video transmission over best-effort packet-based networks suffers from packet losses that result in missing areas in the decoded images. The loss concealment is a postprocessing performed after decoding that searches to recover lost parts of an image by exploiting dependencies within the image or between adjacent images in a video sequence. The positions of the missing areas depend on the data packetization scheme. Transmission schemes, such as the flexible macroblock ordering, avoiding placing adjacent blocks of pixels in the same packet, have been specified in the H.264 standard, to ease the concealment process in case of packet losses.

In transmission and decoding systems that are practically used today, the loss concealment is usually performed using a simple copy from previous frames that have been correctly received or using simple spatiotemporal interpolation. However, inpainting techniques are useful tools in this context as well, although complexity issues become quite crucial in this context, where the inpainting must be done in real time. This application is illustrated in Figure 13, where the inpainting is done using the hybrid local/nonlocal sparse representation method. As far as the inpainting methods are concerned, the problem is similar to the object removal application, therefore, in this case as well, patch-based methods using examplars or sparse priors, or hybrid methods combining diffusion and examplar-based techniques give very good results.

[FIG13] An illustration of the loss concealment problem. The lost areas of the image are recovered with the hybrid local/nonlocal sparse representation [54].

CONCLUSIONS

Image inpainting has received a lot of attention in the past few years. Numerous and different types of approaches have been proposed with varying applicability in restoration, object removal, disocclusion, or in texture synthesis. These algorithms have, however, limited direct applicability for video inpainting, which remains an open problem, despite preliminary solutions making assumptions in terms of moving objects or camera motion. Tracking moving objects in a video (the ones to be removed or others impacted by missing data due to losses or occlusions) in an unsupervised manner remains a difficult problem.

The quality assessment of inpainted images is another open and difficult issue, as no quantitative metrics exists. Fidelity metrics cannot be used given that, for most inpainting applications, the ground truth is in general unknown. One has to rely on a subjective assessment to evaluate whether the inpainted images are visually pleasing and physically plausible.

AUTHORS

Christine Guillemot (Christine.Guillemot@inria.fr) is the director of research at the Institut National de Recherche en Informatique et en Automatique and head of a research team dealing with image and video modeling, processing, coding, and communication. She received her Ph.D. degree from Ecole Nationale Superieure des Telecommunications, Paris, and a "Habilitation for Research Direction" from the University of Rennes. From 1985 to 1997, she was with France Telecom, where she was involved in various projects in the area of image and video coding for television, high-definition television, and multimedia. From 1990 to 1991, she was a visiting scientist at Bellcore, New Jersey. She has (co)authored 15 patents, eight book chapters, 50 journal papers, and 140 conference papers. She has been an associate editor of *IEEE Transactions on Image Processing* (2000–2003), *IEEE Transactions on Circuits and Systems for Video Technology* (2004–2006), and *IEEE Transactions on Signal Processing* (2007–2009). She is currently an associate editor of EURASIP's *Journal on Image Communication* and is a member of the editorial board of *IEEE Journal on Selected Topics in Signal Processing* (2013–2015). She is a member of the IEEE Image, Video, and Multidimensional Signal Processing Technical Committees. She is a Fellow of the IEEE.

Olivier Le Meur (olivier.le_meur@irisa.fr) received his Ph.D. degree from the University of Nantes in 2005. From 1999 to 2009, he worked in the media and broadcasting industry. In 2003, he joined the research center of Thomson-Technicolor at Rennes, where he supervised a research project concerning the modeling of the human visual attention. He has been an associate professor of image processing at the University of Rennes 1 since 2009. He is a part of the IRISA/TEMICS team, which includes computational modeling of the visual attention and saliency-based applications (video compression, objective assessment of video quality, and retargeting). His research interests are dealing with the understanding of the human visual attention.

REFERENCES

[1] S. Masnou and J. Morel, "Level-lines based disocclusion," in *Proc. IEEE Int. Conf. Image Processing (ICIP)*, Chicago, IL, Oct. 1998, vol. 3, pp. 259–263.

[2] M. Bertalmio, G. Sapiro, C. Ballester, and V. Caselles, "Image inpainting," in *Proc. ACM SIGGRAPH*, July 2000, pp. 417–424.

[3] A. Efros and T. Leung, "Texture synthesis by non-parametric sampling," in *Proc. Int. Conf. Computer Vision (ICCV)*, Sept. 1999, pp. 1033–1038.

[4] S. Di Zenzo, "A note on the gradient of a multi-image," *Comput. Vision, Graphics, Image Process.*, vol. 33, no. 1, pp. 116–125, Jan. 1986.

[5] P. Perona and J. Malik, "Scale-space and edge detection using anisotropic diffusion," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 12, no. 7, pp. 629–639, 1990.

[6] F. Voci, S. Eiho, N. Sugimoto, and H. Sekiguchi, "Estimating the gradient threshold in the Perona-Malik equation," *IEEE Signal Processing Mag.*, vol. 21, no. 3, pp. 39–46, May 2004.

[7] J. Weickert, "Theoretical foundations of anisotropic diffusion in image processing," *Comput. Suppl.*, vol. 11, pp. 221–236, 1996.

[8] J. Weickert, "Coherence-enhancing diffusion filtering," *Int. J. Comput. Vis.*, vol. 32, no. 2–3, pp. 111–127, 1999.

[9] M. Bertalmio, A. Bertozzi, and G. Sapiro, "Navier-stokes, fluid dynamics, and image and video inpainting," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, Dec. 2001, pp. 355–362.

[10] A. Telea, "An image inpainting technique based on the fast marching method," *J. Graphics Tools*, vol. 9, no. 1, pp. 25–36, 2004.

[11] D. Tschumperlé, "Fast anisotropic smoothing of multi-valued images using curvature-preserving PDE's," *Int. J. Comput. Vis.*, vol. 68, no. 1, pp. 65–82, 2006.

[12] L. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Phys. D*, vol. 60, pp. 259–268, Nov. 1992.

[13] 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, Aug. 2001.

[14] T. Chan and J. Shen, "Local inpainting models and TV inpainting," *SIAM J. Appl. Math.*, vol. 62, no. 3, pp. 1019–1043, 2001.

[15] A. Levin, A. Zomet, and Y. Weiss, "Learning how to inpaint from global image statistics," in *Proc. Int. Conf. Computer Vision (ICCV)*, Oct. 2003, pp. II. 305–313.

[16] T. Chan and J. Shen, "Non texture inpainting by curvature-driven diffusion," *J. Visual Commun. Image Representation*, vol. 12, no. 4, pp. 436–449, 2001.

[17] T. Chan, S. Kang, and J. Shen, "Euler's elastica and curvature-based inpainting," *SIAM J. Appl. Math.*, vol. 63, no. 2, pp. 564–592, 2003.

[18] L. Wei and M. Levoy, "Fast texture synthesis using tree-structured vector quantization," in *Proc. ACM SIGGRAPH*, July 2000, pp. 479–488.

[19] M. Ashikhmin, "Synthesizing natural textures," in *Proc. ACM Symp. Interactive 3D Graphics*, Mar. 2001, pp. 217–226.

[20] L. Liang, C. Liu, Y. Xu, B. Guo, and H. Shum, "Real-time texture synthesis by patch-based sampling," *ACM Trans. Graph.*, vol. 20, no. 3, pp. 127–150, 2001.

[21] A. Bugeau, M. Bertalmio, V. Caselles, and G. Sapiro, "A comprehensive framework for image inpainting," *IEEE Trans. Image Processing*, vol. 19, no. 10, pp. 2634–2645, Oct. 2010.

[22] O. Le Meur and C. Guillemot, "Super-resolution-based inpainting," in *Proc. European Conf. Computer Vision (ECCV)*, Oct. 2012, pp. 554–567.

[23] Y. Zhang, J. Xiao, and M. Shah, "Region completion in a single image," in *EUROGRAPHICS*, Sept. 2004.

[24] I. Drori, D. Cohen-Or, and H. Yeshurun, "Fragment-based image completion," *ACM Trans. Graph.*, vol. 22, no. 2003, pp. 303–312, 2005.

[25] J. Bentley, "Multidimensional binary search trees used for associative searching," *Commun. ACM*, vol. 18, no. 9, pp. 509–517, Sept. 1975.

[26] P. Yanilos, "Data structures and algorithms for nearest neighbor search in general metric spaces," in *Proc. 4th Annu. ACM/SIGACT-SIAM Symp. Discrete Algorithms (SODA)*, 1993, pp. 311–321.

[27] N. Kumar, L. Zhang, and S. K. Nayar, "What is a good nearest neighbors algorithm for finding similar patches in images?" in *Proc. European Conf. Computer Vision (ECCV)*, Oct. 2008, pp. 364–378.

[28] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman, "Patchmatch: A randomized correspondence algorithm for structural image editing," *ACM Trans. Graph.*, vol. 28, no. 3, pp. 24:1–24:11, July 2009.

[29] S. Korman and S. Avidan, "Coherency sensitive hashing," in *Proc. Int. Conf. Computer Vision (ICCV)*, 2011, pp. 1607–1614.

[30] C. Barnes, E. Shechtman, D. B. Goldman, and A. Finkelstein, "The generalized PatchMatch correspondence algorithm," in *Proc. European Conf. Computer Vision (ECCV)*, Sept. 2010, vol. 6313, pp. 29–43.

[31] A. Criminisi, P. Pérez, and K. Toyama, "Region filling and object removal by examplar-based inpainting," *IEEE Trans. Image Processing*, vol. 13, no. 9, pp. 1200–1212, Sept. 2004.

[32] Z. Xu and J. Sun, "Image inpainting by patch propagation using patch sparsity," *IEEE Trans. Image Processing*, vol. 19, no. 5, pp. 1153–1165, May 2010.

[33] O. Le Meur, J. Gautier, and C. Guillemot, "Examplar-based inpainting based on local geometry," in *Proc. IEEE Int. Conf. Image Processing (ICIP)*, Sept. 2011, pp. 3401–3404.

[34] G. Kanizsa, *Organization in Vision*. New York: Praeger, 1979.

[35] Y. Wexler, E. Shechtman, and M. Irani, "Space-time video completion," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, June 2004, vol. 1, pp. I-120–I-127.

[36] N. Komodakis and G. Tziritas, "Image completion using global optimization," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2006, pp. 442–452.

[37] Y. Pritch, E. Kav-Venaki, and S. Peleg, "Shift-map image editing," in *Proc. Int. Conf. Computer Vision (ICCV)*, Kyoto, Sept. 2009, pp. 151–158.

[38] T. Cho, M. Butman, S. Avidan, and W. Freeman, "The patch transform and its applications to image editing," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, June 2008, pp. 1–8.

[39] K. He and J. Sun, "Statistics of patch offsets for image completion," in *Proc. European Conf. Computer Vision (ECCV)*, Oct. 2012, vol. 7773, pp. 16–29.

[40] A. Efros and W. Freeman, "Image quilting for texture synthesis and transfer," in *Proc. Annu. Conf. Computer Graphics and Interactive Techniques, ACM SIGGRAPH*, July 2001, pp. 341–346.

[41] S. Avidan and A. Shamir. (2007). Seam carving for content-aware image resizing. *Proc. ACM SIGGRAPH*. [Online]. Available: http://doi.acm.org/10.1145/1275808.1276390

[42] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *IEEE Trans. Pattern Anal. Mach. Intell. (PAMI)*, vol. 23, no. 11, pp. 1222–1239, Nov. 2001.

[43] V. Kolmogorov and R. Zabih, "What energy functions can be minimized via graph cuts?" *IEEE Trans. Pattern Anal. Mach. Intell. (PAMI)*, vol. 26, no. 2, pp. 147–159, Feb. 2004.

[44] V. Kwatra, A. Schodl, I. Essa, G. Turk, and A. Bobik, "Graphcut textures: image and video synthesis using graph cuts," in *ACM SIGGRAPH*, July 2003, vol. 22, pp. 277–286.

[45] P. Pérez, M. Gangnet, and A. Blake, "Poisson image editing," in *ACM SIGGRAPH*, 2003, pp. 313–318.

[46] S. Darabi, E. Shechtman, C. Barnes, D. B. Goldman, and P. Sen, "Image melding: Combining inconsistent images using patch-based synthesis," *ACM Trans. Graph. (TOG) (Proc. SIGGRAPH 2012)*, vol. 31, no. 4, pp. 82:1–82:10, 2012.

[47] A. Wong and J. Orchard, "A nonlocal-means approach to exemplar-based inpainting," in *Proc. IEEE Int. Conf. Image Processing (ICIP)*, 2006, pp. 2600–2603.

[48] A. Buades, B. Coll, and J. M. Morel, "A review of image denoising algorithms, with a new one," *SIAM J. Multi Scale Modeling Simul.*, vol. 4, no. 2, pp. 490–530, 2005.

[49] S. Roweis and L. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, pp. 2323–2326, Dec. 2000.

[50] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," *Adv. Neural Inform. Process. Syst. (NIPS)*, pp. 556–562, Nov. 2000.

[51] M. Elad, *Sparse and Redundant Representations: From Theory to Applications in Signal and Image Processing*. New York: Springer, 2010.

[52] C. Studer, P. Kuppinger, G. Pope, and H. Bolckei, "Recovery of sparsely corrupted signals," *IEEE Trans. Inform. Theory*, vol. 58, no. 5, pp. 3115–3130, May 2012.

[53] O. Guleryuz, "Nonlinear approximation based image recovery using adaptive sparse reconstructions and iterated denoising—Part II: Adaptive algorithms," *IEEE Trans. Image Processing*, vol. 15, no. 3, pp. 555–571, Mar. 2006.

[54] X. Li, "Image recovery via hybrid sparse representations: A deterministic annealing approach," *IEEE J. Select. Topics Signal Process.*, vol. 5, no. 5, pp. 953–962, Sept. 2011.

[55] 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, Aug. 2003.

[56] J. Stark, M. Elad, and D. Donoho, "Image decomposition via the combination of sparse representations and variational approach," *IEEE Trans. Image Processing*, vol. 14, pp. 1570–1582, Oct. 2005.

[57] J. Sun, L. Yuan, J. Jia, and H. Shum, "Image completion with structure propagation," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 861–868, 2005.

[58] M. Elad, J.-L. Stark, P. Querre, and D. Donoho, "Simultaneous cartoon and texture image inpainting using morphological component analysis (MCA)," *J. Appl. Comput. Harmonic Anal.*, vol. 19, pp. 340–358, Nov. 2005.

[59] J. Aujol, S. Ladjal, and S. Masnou, "Examplar-based inpainting from a variational point of view," *SIAM J. Math. Anal.*, vol. 42, no. 3, pp. 1246–1285, Mar. 2010.

[60] J. Portilla, E. Simoncelli, and Y.-J. Zhang, "A parametric texture model based on joint statistics of complex wavelets," *Int. J. Comput. Vis.*, vol. 40, no. 1, pp. 49–71, 2000.

[61] E. Candes and B. Recht, "Exact matrix completion via convex optimization," *Commun. ACM*, vol. 55, no. 6, pp. 111–119, 2012.

[62] J. Liu, P. Musialski, P. Wonka, and J. Ye, "Tensor completion for estimating missing values in visual data," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 1, pp. 208–220, Jan. 2013.

[63] S.-C. Pei, Y.-C. Zeng, and C.-H. Chang, "Virtual restoration of ancient Chinese paintings using color contrast enhancement and lacuna texture synthesis," *IEEE Trans. Image Processing*, vol. 13, no. 3, pp. 416–429, Mar. 2004.

[64] M. Arnold, A. Ghosh, S. Ameling, and G. Lacey, "Automatic segmentation and inpainting of specular highlights for endoscopic imaging," *EURASIP J. Image Video Process.*, vol. 2010, article 9, ID 814319, pp. 1–12, Dec. 2010.

[65] F. Bornemann and T. Marz, "Fast image inpainting based on coherence transport," *J. Math. Imaging Vis.*, vol. 28, no. 3, pp. 259–278, July 2007.

[66] C. Guillemot, M. Turkan, O. L. Meur, and M. Ebdelli, "Image inpainting using LLE-LDNR and linear subspace mappings," in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing (ICASSP)*, May 2013.

[67] C. Zitnick, S. Kang, M. Uyttendaele, S. Winder, and R. Szeliski, "High-quality video view interpolation using a layered representation," in *ACM SIGGRAPH*, 2004, pp. 600–608.

[SP]