Cartoon-Texture Image Decomposition using Orientation Characteristics in Patch Recurrence[*]

Ruotao Xu[†], Yong Xu[†], Yuhui Quan[†#], and Hui Ji[‡]

Abstract. Cartoon-texture image decomposition is about decomposing an image into the linear sum of two layers: cartoon and texture, where the key challenge is how to resolve the ambiguity between two layers. It is observed that the recurrence of texture patches occurs along multiple orientations, and the recurrence of cartoon patches only occurs along certain orientation. This paper proposes to separate these two layers by exploiting their orientation characteristics of image patch recurrence, *i.e.*, isotropy property of texture patch recurrence versus anisotropy property of cartoon patch recurrence. Together with the sparsity-based regularizations in the image domain, a variational method is then developed in this paper for cartoon-texture decomposition. The experiments show that the proposed method noticeably outperforms many well-established ones on test images.

Key words. Cartoon-texture decomposition, Patch recurrence, Regularization method

AMS subject classifications. 68T05, 68U10, 65D18

1. Introduction.

How to decompose an image into different semantic layers is an important problem with many applications in practice. One such problem is about decomposing an image into the linear combination of a cartoon layer and a texture layer, *i.e.* the so-called *cartoon-texture image decomposition*. Cartoon layer refers to the one with piece-wise smooth parts, and texture layer refers to the one with repetitive patterns. For instance, a cartoon layer contains homogeneous regions, object contours, and significant image edges, while a texture layer contains grasses, fabric textures, barks and many others with small repetitive features.

Cartoon-texture decomposition sees its usage in many image processing tasks. As cartoon layer and texture layer exhibit rather different characteristics in terms of visual appearance, the operations on these two layers are often different in many applications. In image recovery, cartoon regions and texture regions need to be treated differently for better visual quality [4, 20, 33]. For optical flow estimation in motion analysis, texture parts need to be removed from images for improved robustness to shading reflections and shadows [48]. For depth estimation in stereoposis, cartoon parts are extracted from images for better stability of the

[#]Pound indicates the corresponding author.

[*]Submitted to the editors DATE.

Funding: This work is supported by National Natural Science Foundation of China (61872151, 61672241, U1611461), Natural Science Foundation of Guangdong Province (2017A030313376), Science and Technology Program of Guangdong Province (2019A050510010), Science and Technology Program of Guangzhou (201802010055), and Singapore MOE AcRF (R146000229114, MOE2017-T2-2-156).

[†]Ruotao Xu, Yong Xu and Yuhui Quan are with School of Computer Science and Engineering at South China University of Technology, Guangzhou 510006, China. Yong Xu is also with Peng Cheng Laboratory, Shenzhen 510852, China. Yuhui Quan is also with the Guangdong Provincial Key Laboratory of Computational Intelligence and Cyberspace Information, Guangzhou 510006, China. (email: xu.ruotao@mail.scut.edu.cn; yxu@scut.edu.cn; csyhquan@scut.edu.cn).

[‡]Hui Ji is with Department of Mathematics at National University of Singapore, Singapore 119076. (email: matjh@nus.edu.sg).

algorithm [10]. Other applications include image segmentation [11], image compression [27], image editing [26, 29], color transfer [26], pattern recognition [18], biomedical engineering [18], remote sensing [1, 22, 50], and many others.

In the past, there have been extensive works on cartoon-texture decomposition. Following the seminal works [32, 36], most existing works model an image $\boldsymbol{f}$ as the sum of two layers:

$$\boldsymbol{f} = \boldsymbol{u} + \boldsymbol{v}, \tag{1.1}$$

where $\boldsymbol{u}$ denotes the cartoon layer and $\boldsymbol{v}$ denotes the texture layer. The goal is then to estimate both $\boldsymbol{u}$ and $\boldsymbol{v}$ from $\boldsymbol{f}$. Clearly, it is an under-determined linear inverse problem with many solutions. To resolve such an ambiguity, one prominent approach is to impose certain priors on both $\boldsymbol{u}$ and $\boldsymbol{v}$. Then, the problem (1.1) is re-formulated as an optimization problem:

$$\min_{\boldsymbol{u},\boldsymbol{v}} \phi(\boldsymbol{u}) + \psi(\boldsymbol{v}), \quad \text{s.t.} \quad \boldsymbol{u} + \boldsymbol{v} = \boldsymbol{f}, \tag{1.2}$$

where $\phi(\cdot)$ and $\psi(\cdot)$ denote the regularization terms on the two layers $\boldsymbol{u}$ and $\boldsymbol{v}$ respectively, which are derived from the priors imposed on $\boldsymbol{u}, \boldsymbol{v}$.

The decomposition quality using (1.2) largely depends on the choice of the two priors imposed on $\boldsymbol{u}$ and $\boldsymbol{v}$. To have a high-quality cartoon-texture decomposition, the priors need to satisfy two conditions:

1. The imposed prior fits the characteristics of the corresponding layer well.
2. The priors on the two layers are sufficiently discriminative to resolve the ambiguity between them.

Indeed, one main difference among existing methods lies in what kind of prior is imposed on cartoon/texture for decomposition. In the next, we give a brief review on existing cartoon-texture decomposition methods.

1.1. Related Work. In most existing approaches, cartoon layer is modeled as the part that can be well approximated by a piece-wise smooth function, *i.e.* the function whose discontinuities are geometrically regular and well separated in space. The prominent regularization for estimating such piece-wise smooth functions is the ℓ_1-norm-relating functional that exploits the sparsity of its discontinuities. For example, the total variation (TV) based approach (*e.g.* [32, 47, 36, 4, 1, 51, 52, 22, 15, 53, 49, 43, 26]) defines $\phi(\boldsymbol{u}) = \|\nabla \boldsymbol{u}\|_1$, where ∇ denotes the first-order gradient operator or its higher-order generalizations. The wavelet/curvelet transform based approach (*e.g.* [45, 41, 16, 42, 31, 35]) uses high-pass wavelet/curvelet transform to measure the intensity variation, and regularizes the cartoon layer by defining $\phi(\boldsymbol{u}) = \|W\boldsymbol{u}\|_1$, where the operator W denotes the high-pass wavelet/curvelet transform.

These ℓ_1-norm relating regularizations work well when recovering cartoon regions in many image processing tasks, *e.g.* image denoising and deconvolution. However, for cartoon-texture decomposition, there is no clear cut on cartoon and texture in terms of sparsity degree of discontinuities. For instance, a small portion of image edges (discontinuities) with large magnitude in the cartoon layer might be wrongly assigned to the texture layer, as such assignments will make the cartoon layer contain even less discontinuities. In other words, such an erroneous assignment does not break the discontinuity-based sparsity prior of the cartoon layer. In short, although the discontinuity-based sparsity prior holds true for the cartoon layer, but it is not powerful enough to correctly determine the ownership of each individual image edge, *i.e.*, which layer it belongs to.

In comparison to the cartoon layer, the texture layer is more difficult to characterize. In the past, many characterizations have been proposed for modeling texture. Each has its own strength and weakness, and is only applicable to certain types of textures. Early works [32, 47] model texture as the patterns with strong oscillation, characterized in the Besov space $B_{\infty}^{-1,\infty}$. Since then, there is an enduring effort on the analysis of the model and the development of related numerical methods; see *e.g.* [36, 40, 1, 2, 3, 22, 15]. There are also other variations with strong motivations from the works above, including linear filtering models [7, 8] and multi-scale model [21]. Also, similar idea is used for solving other texture-related image processing problems; see *e.g.* [4, 27, 33]. Nevertheless, it is pointed out in [39] that one main weakness of such an approach is that it often fails on the texture parts that contain regular patterns with small magnitude.

Similar to the sparsity-based prior on the gradients of cartoon layers, one alternative approach to model texture layers assumes that texture can be sparsely approximated under some transform; see *e.g.* [41, 16, 31, 35, 11, 53, 37, 24]. The transform for sparsifying texture can be either a pre-defined one such as local Fourier transform [31], or the one learned from texture images [53, 37]. Then, the regularization on texture can be formulated as some sparsity-prompting functional, *e.g.* ℓ_1-norm or ℓ_0-norm, on the transform coefficients of texture. These sparse-approximation-based characterizations work well for texture with regular patterns. However, they do not work well on natural textures generated by some stochastic processes. Furthermore, to resolve the ambiguity between cartoon and texture, the sparsifying transform for texture should have very low coherence with that of cartoon, which leads to a challenging non-convex minimization problem; see [16] for more details.

Instead of using sparsity-based characterization, another approach to model texture is to examine the rank of the matrix of texture patches after certain alignment. Motivated by linear dependence of texture patches after alignment, Schaeffer and Osher [39] proposed a low-rank prior on the matrix formed by these aligned texture patches, and used a nuclear-norm-based regularization for texture extraction. The nuclear norm used is replaced in [17] by the so-called log-determinant function for better performance. As these two methods use all texture patches of the whole image to form the matrix, they are not applicable to the texture layer which contains different types of textures. To address such an issue, Ono and Miyata [34] proposed to consider the matrix formed by the texture patches in a local region. However, it is challenging to set appropriate region size for an image with multiple texture regions of varying size.

Recently, following the same idea of the BM3D method for image denoising [13], Ma *et al.* [30] proposed to decompose cartoon and texture over the group of matched similar patches. In the matrix formed by a group of matched image patches, the matrix is decomposed into the sum of a low-rank one and a structured sparse one, where the low-rank one represents the texture layer and the sparse one represents the cartoon layer. Similarly, texture parts are extracted in Sur *et al.* [44] over the group of matched patches by using some power-spectrum-based statistical measurements. It is noted that the low-rank-based characteristic on texture proposed in [30, 44] is different from that in [39, 34, 17]. The characteristic proposed in [30, 44] is based on the phenomena of non-local patch recurrence in natural images [13, 38], *i.e.*, small

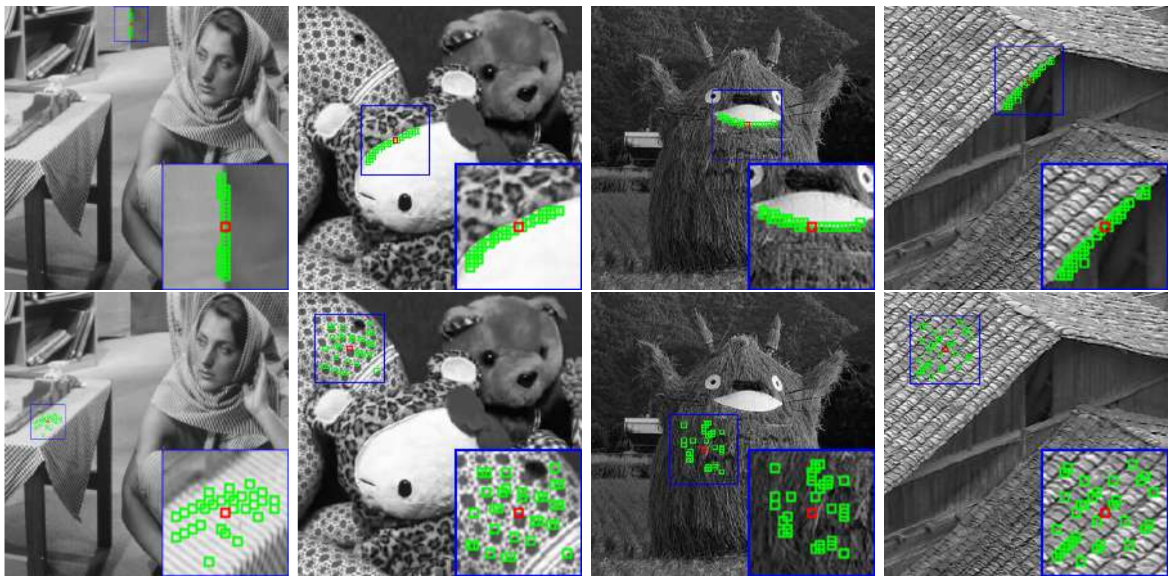


Figure 1. *Top row: Anisotropic patch recurrence in cartoon layer. Bottom row: Isotropic patch recurrence in texture layer.*

image patches are very likely to repeat themselves over the whole image.

1.2. Main Idea.

The prior of patch recurrence is a powerful one that has seen its success in many image recovery tasks. For instance, many image denoising methods with state-of-the-art performance are the non-local methods based on such a non-local patch recurrence prior for nature images, including BM3D [13], non-local means [6], low-rank approximation [25], and many others. It is noted that, in order to exploit such a non-local prior, one needs to run a patch matching to search image patches over the image that are similar to a given candidate patch. Such a process can be very computationally expensive if one searches all possible image patches of the whole image. It is empirically observed that similar patches are much more likely to exist in the neighborhood of the candidate image patch. Indeed, the practical implementation of most non-local methods only utilizes the prior of local patch recurrence by searching similar patches in the neighborhood of a given candidate patch; see *e.g.* [13, 6, 25].

Indeed, the prior of local patch recurrence is a powerful prior for texture. In many image recovery tasks, it is the texture layer where those non-local methods outperform the sparsity-based regularization methods. The main reason is that the sparsity prior of local intensity variation does not hold true for texture regions. However, such a prior cannot be directly used for cartoon-texture decomposition, as it holds true for both cartoon and texture. There are only few attempts on investigating the potential of patch recurrence in cartoon-texture decomposition [30, 44]. In [30, 44], the decomposition of cartoon and texture is done by examining the rank of the matrix associated with matched patches. Such an approach does not work well on the textures with complex patterns.

1.2.1. Orientation-based prior on patch recurrence.

Based on the local patch recurrence of natural images, this paper proposes a new characteristic for distinguishing cartoon layers from texture layers, which examines the orientation property of the spatial distribution of the matched patches in the neighborhood of a candidate image patch. Such a prior is motivated by the following observation:

- *Anisotropic recurrence for cartoon patches*: Consider a candidate cartoon patch, most of its similar patches are likely to exist along a specific direction in its neighborhood.
- *Isotropic recurrence for texture patches*: Consider a candidate texture patch, most of its similar patches are likely to scatter around in its neighborhood without any preference on some specific direction.

See Figure 1 for an illustration of anisotropic patch recurrence (cartoon) vs isotropic patch recurrence (texture) of some sample images. The anisotropic recurrence prior for cartoon patches comes from the fact that a cartoon patch usually only contains some isolated edge segment, which is one part of the contour of an object. Thus, similar patches can be found along the direction of such a line segment, but not other directions. In contrast, the isotropic recurrence of texture patches comes from the fact that a candidate texture patch is inside a statistically-homogeneous texture region, *e.g.*, patterns in clothes, markings of animals, and grains of woods.

The patch-recurrence-based orientation prior can be formulated by considering running a specific ordering scheme on matched patches. Briefly, given a candidate patch, we stack its matched patches column-wisely in a specific order to form a matrix. Such a matrix shows strong oscillations row-wisely for the cartoon parts and is smooth for the texture parts. Then, for the stack of cartoon patches, the anisotropic recurrence prior is reformulated as a periodicity prior. For the stack of texture patches, the isotropic recurrence prior is re-formulated as a smoothness prior. See Figure 2 for an illustration of the stack of matched patches and its associated prior: periodic oscillation (cartoon) vs. smoothness (texture). It is interesting to see that based on the proposed patch stack, we have

- *Cartoon*: Piece-wise smoothness prior column-wise (image domain) vs periodicity prior row-wise (patch recurrence).
- *Texture*: Deterministic/statistical periodicity prior column-wise (image domain) vs smoothness prior row-wise (patch recurrence)

The priors listed above lead to new regularization strategies on the stacks of cartoon patches and texture patches for image decomposition.

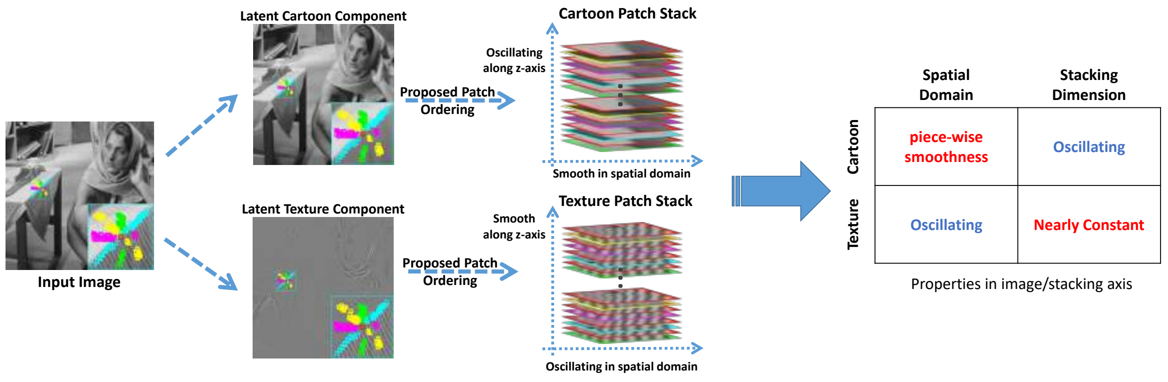


Figure 2. *Illustration of discriminative properties of cartoon/texture patch stacks.*

1.2.2. Outline of the proposed method. Based on the orientation prior discussed in the previous section, we propose a variational method for cartoon-texture decomposition. Two transforms will be used in the proposed method. One is high-pass wavelet transform, denoted by $\boldsymbol{W}$, which will be used for regularizing cartoon as it is known for the optimality on modeling piece-wise smooth functions [41, 16, 9]. The other is high-pass discrete cosine transform (DCT), denoted by $\boldsymbol{D}$, which will be used for regularizing cartoon, as it is very effective on modeling oscillation patterns. Let $\boldsymbol{L}, \boldsymbol{H}$ represent the operators that convolves a patch stack along the stacking axis by the low-pass filter $[1, \dots, 1]$ and the high-pass filter $[1, -1]$ respectively.

Consider a cartoon patch stack U and a texture patch stack V. We propose to regularize them by the following functional terms:

$$\Phi(U) = \underbrace{\alpha_1 \|\boldsymbol{W} \circ U\|_1}_{\text{Image domain}} + \underbrace{\alpha_2 \|\boldsymbol{L} \circ \boldsymbol{W} \circ U\|_1}_{\text{Patch stack}}, \tag{1.3}$$

$$\Psi(V) = \underbrace{\beta_1 \|\boldsymbol{D} \circ V\|_1}_{\text{Image domain}} + \underbrace{\beta_2 \|\boldsymbol{H} \circ \boldsymbol{D} \circ V\|_1}_{\text{Patch stack}}. \tag{1.4}$$

In the regularization (1.3) on cartoon, the first term exploits the piece-wise smoothness prior of cartoon in the image domain. The second term exploits the periodicity prior of a cartoon patch stack along the stacking axis. Such a prior is equivalent to the prior that the overall energy of its low-pass components is close to 0. In the regularization (1.4) on texture, the first term exploits the oscillation prior of texture in the image domain, which is equivalent to the prior that the energy of its DC component is close to 0. The second term exploits the smoothness prior of a texture patch stack along the stacking axis, which is equivalent to the prior that the overall energy of its high-pass components is close to 0.

In both (1.3) and (1.4), owing to its robustness to the outliers in patch matching, the ℓ_1-norm is used as the metric for measuring the energy associated with the priors of patch recurrence. Furthermore, Each entry of cartoon/texture layer will appear in many patches. If one processes each patch independently and then fuse them to form the final result of the two layers, the cartoon/texture layer appears to be blurry as the estimate of each entry is the average of multiple estimations from multiple patches. For reducing such blurring effects, we re-formulate the regularization defined on patch stacks to the one defined directly on image pixels.

Consider a set of image patches that covers all image pixels. Let $\boldsymbol{P}_i$ denote the projection operator that maps an image to the i-th patch, and let $\boldsymbol{S}_i$ denote the projection operator that maps an image to the patch stack containing the patches matched to the i-th patch. Then, we propose the following regularization model for cartoon-texture decomposition:

$$\min_{\boldsymbol{u},\boldsymbol{v}} \phi(\boldsymbol{u}) + \psi(\boldsymbol{v}), \quad \text{subject to} \quad \boldsymbol{u} + \boldsymbol{v} = \boldsymbol{f}, \tag{1.5}$$

where

$$\begin{cases} \phi(\boldsymbol{u}) = \sum_i \lambda_i^c (\alpha_1 \|\boldsymbol{W} \circ \boldsymbol{P}_i \circ \boldsymbol{u}\|_1 + \alpha_2 \|\boldsymbol{L} \circ \boldsymbol{W} \circ \boldsymbol{S}_i \circ \boldsymbol{u}\|_1), \\ \psi(\boldsymbol{v}) = \sum_i \lambda_i^t (\beta_1 \|\boldsymbol{L} \circ \boldsymbol{P}_i \circ \boldsymbol{v}\|_1 + \beta_2 \|\boldsymbol{H} \circ \boldsymbol{D} \circ \boldsymbol{S}_i \circ \boldsymbol{v}\|_1). \end{cases} \tag{1.6}$$

The rest of this paper is organized as follows. Section 2 first presents the construction of the matched patch stack with a specific ordering, and then analyzes its orientation property. Section 3 is devoted to the description of the proposed model and the corresponding numerical solver. The experimental evaluation is given in Section 4. Section 5 concludes the paper.

2. Construction of Patch Stack and Analysis of Its Orientation Property.

Similar to the practical implementations of most non-local image recovery methods [13, 6, 25], given a candidate image patch, we only search the similar patches in its neighborhood. The similarity between two patches is measured by the ℓ_2-norm-based distance. In this section, we propose a new scheme of patch matching such that the resulting patch stack shows different orientation characteristics between cartoon and texture. Throughout this paper, bold upper letters are used for denoting operators, normal upper letters for matrices, bold lower letters for images or image patches, and normal lower letters for vectors. Let 0 and I denote the zero vector and the identity matrix. Let $\mathrm{diag}(x)$ denote the square diagonal matrix with the elements of x on its main diagonal, *i.e.*, for $x \in \mathbb{R}^n$,

$$\mathrm{diag}(x) = \begin{bmatrix} x_1 & 0 & \cdots & 0 \\ 0 & x_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & x_n \end{bmatrix} \in \mathbb{R}^{n\times n}.$$

Let comma and semicolon denote the horizontal and vertical concatenation operators of vectors, and let $\otimes$ denotes the Kronecker product. Recall that, given two matrices $A \in \mathbb{R}^{m\times n}$ and $B \in \mathbb{R}^{p\times q}$, the Kronecker product $A \otimes B$ is defined as

$$A \otimes B = \begin{bmatrix} a_{11}B & \cdots & a_{1n}B \\ \vdots & \ddots & \vdots \\ a_{m1}B & \cdots & a_{mn}B \end{bmatrix} \in \mathbb{R}^{mp\times nq}.$$

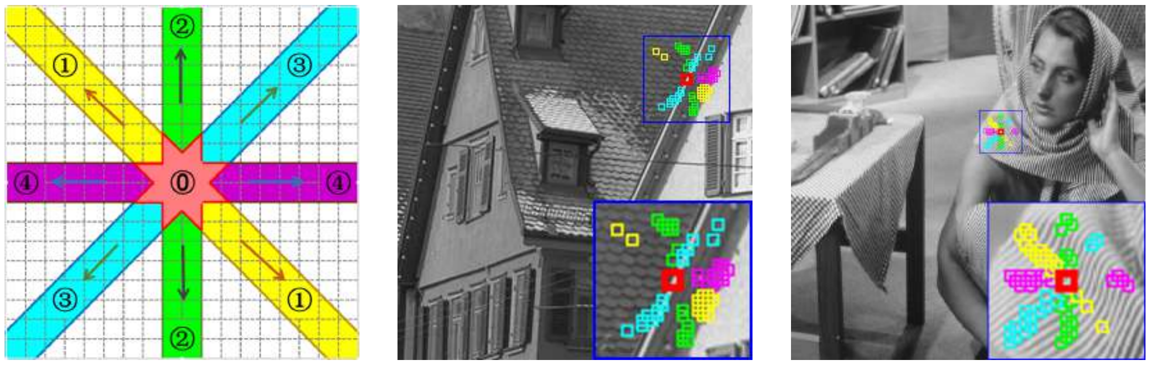
2.1. Construction of Patch Stack with A Specific Ordering.

Given a patch $\boldsymbol{p}$ of size $m \times m$ centering at r_0, we consider finding its similar patches inside its neighborhood, a region centering at r_0 and of size $n \times n$. The neighborhood is then partitioned into R bands centered at r_0 and slanted at R different orientations: $0°, \frac{180°}{R}, \cdots, (R-1)\frac{180°}{R}$. As these bands overlap at a small centering region, such a centering region is excluded from the bands to avoid the patches inside repeating themselves in the search. See Figure 3(a) for an example of $R = 4$, where four bands used for patch matching are colored by yellow, green, blue and purple respectively, and the region colored by red is the excluded overlap region. Denote these bands by $B^{(1)}, B^{(2)}, \ldots, B^{(R)}$ clockwise.

Different from the patch matching done in existing non-local methods which finds the set of most similar patches in the neighborhood of the patch $\boldsymbol{p}_i$, the patch matching in our approach is about finding K patches with the smallest distance to the candidate patch $\boldsymbol{p}_i$ in each band $B_i^{(r)}$, for $r = 1, 2, \ldots, R$. The outcome is then R sets of matched patches denoted by $\{\mathbb{S}_i^{(r)}\}_{r=1}^R$. The number K is fixed for all the bands, which is set to 16 in the implementation. See Figure 3(b) and Figure 3(c) for the illustration of patch matching inside each band. In other words, the matched patches in our approach are not the KR patches most similar to

$\boldsymbol{p}_i$, but the union of the sets each of which contains the K patches with the smallest distance to $\boldsymbol{p}_i$ in a band. Such a process enables us to exploit the orientation characteristic of the distribution of similar patches.

More specifically, consider a cartoon patch containing a dominant line segment, which is very likely one part of a line in the image. Then, its similar patches are likely to distribute along the same orientation of that line segment. As a result, only the K patches matched along one orientation are similar to the candidate. The other patches along different orientations have large distances to the candidate patch, *i.e.*, they are much less similar to the candidate patch. In contrast, for a texture patch containing oscillation patterns, its similar patches are likely to scatter over all orientations. Thus, those K patches along each orientation will have small distances to the candidate patch. Thus, they are all similar to the candidate patch.



(a) Schematic diagram of proposed partition scheme. (b) Matching on cartoon region. (c) Matching on texture region.

Figure 3. *Illustration of directional patch matching using neighborhood partition with four directions ($R = 4$).*

Based on the patch matching scheme described above, we propose the following ordering scheme to assemble the patch stack. Consider the i-th image patch $\boldsymbol{p}_i \in \mathbb{R}^{m\times m}$. Let $\{\boldsymbol{p}_k^{(r)}\}_{k=1}^K$ denote the set of the K matched patches along the r-th band as described above. Then, the patch stack for $\boldsymbol{p}_i$ is formed by concatenating the set of all matched patches $\{\boldsymbol{p}_k^{(r)}\}_{1\le r\le R, 1\le k\le K}$ in the following order:

$$[\underbrace{\boldsymbol{p}_i, \boldsymbol{p}_1^{(1)}, \boldsymbol{p}_i, \boldsymbol{p}_1^{(2)}, \ldots, \boldsymbol{p}_i,\boldsymbol{p}_1^{(R)}}, \underbrace{\boldsymbol{p}_i, \boldsymbol{p}_2^{(1)}, \boldsymbol{p}_i, \boldsymbol{p}_2^{(2)}, \ldots, \boldsymbol{p}_i,\boldsymbol{p}_2^{(R)}}, \ldots, \underbrace{\boldsymbol{p}_i, \boldsymbol{p}_K^{(1)}, \boldsymbol{p}_i, \boldsymbol{p}_K^{(2)}, \ldots, \boldsymbol{p}_i,\boldsymbol{p}_K^{(R)}}].$$

In other words, the stack is formed by alternatively connecting the candidate and the top k-th matched patches from each band. For each patch $\boldsymbol{p}_i \in \mathbb{R}^{m\times m}$, let $p_i \in \mathbb{R}^M$ with $M = m^2$ denotes the column vector formed by sequentially concatenating all columns of $\boldsymbol{p}_i$. Then, we define a matrix form of the patch stack *w.r.t.* the candidate patch of $\boldsymbol{p}_i$:

$$\tilde{S}_i = [p_i, p_1^{(1)}, \ldots, p_i,p_1^{(R)}, p_i, p_2^{(1)}, \ldots, p_i,p_2^{(R)}, \ldots, p_i, p_K^{(1)}, \ldots, p_i,p_K^{(R)}] \in \mathbb{R}^{M\times 2KR}. \tag{2.1}$$

See Figure 4(b) for an illustration. Given an image $\boldsymbol{f}$ and its i-th patch $\boldsymbol{p}_i$, let $\boldsymbol{P}_i$ denote the projection operator that maps the image $\boldsymbol{f}$ to the patch p_i and $\boldsymbol{S}_i$ the operator that maps $\boldsymbol{f}$

to the corresponding patch stack of $\boldsymbol{p}_i$:

$$\begin{aligned} \boldsymbol{P}_i &: \boldsymbol{f} \to p_i \in \mathbb{R}^M, \\ \boldsymbol{S}_i &: \boldsymbol{f} \to \tilde{S}_i \in \mathbb{R}^{M\times 2RK}. \end{aligned} \tag{2.2}$$

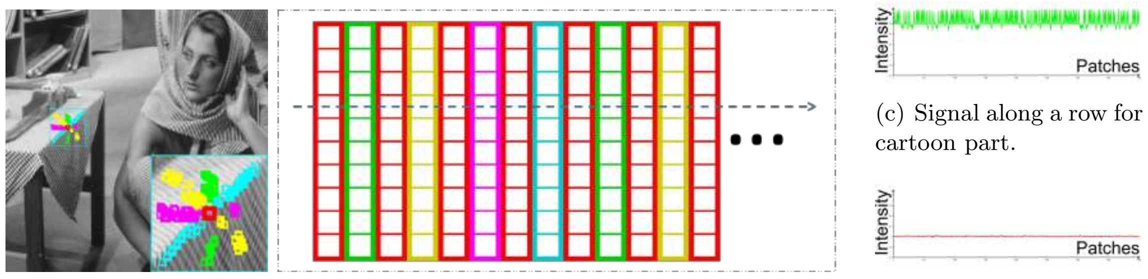
Suppose that $\boldsymbol{u}, \boldsymbol{v}$ are the cartoon layer and the texture layer related by

$$\boldsymbol{f} = \boldsymbol{u} + \boldsymbol{v}.$$

Then we have the sets of matched patch stacks $\{U_i\}_i \subset \mathbb{R}^{M\times 2RK}, \{V_i\}_i \subset \mathbb{R}^{M\times 2RK}$ for the two layers, which are defined by

$$U_i = \boldsymbol{S}_i \circ \boldsymbol{u}, \quad V_i = \boldsymbol{S}_i \circ \boldsymbol{v}. \tag{2.3}$$

See Figure 4 for an illustration of the above patch stack construction scheme.



(a) Directional patch matching. (b) Alternate patch ordering. (c) Signal along a row for cartoon part. (d) Signal along a row for texture part.

Figure 4. *Illustration of construction of patch stack in proposed method.*

2.2. Orientation Property of Matched Patch Stacks.

The matrix U_i from cartoon and the matrix V_i from texture have different characteristics along the stacking axis, *i.e.* along the rows. Consider a matrix V_i that represents a texture patch stack. Then, for $k = 1, 2, \dots, 2RK$, the k-th column of $V_i \in \mathbb{R}^M$ represents a vectorized version of a 2D texture patch $\boldsymbol{v}_k \in \mathbb{R}^{m\times m}$. For each row of V_i, its entries are drawn from the matched patches from R bands with different orientations. By the isotropic patch recurrence of texture patches, these entries are all similar to the corresponding pixels of the candidate patch $\boldsymbol{p}_i$. Therefore, each row can be well approximated by a constant row. The same conclusion holds for the texture patch stack under a 2D DCT.

For the matrix V_i, let DV_i denote the matrix whose k-th column denotes the vectorized form of the output of running a 2D DCT on the patch $\boldsymbol{v}_k$. As each row of DV_i can be well approximated by a constant row, the output of each row of DV_j after convolved by a 1D high-pass filter $[1, -1]$ will be close to $\boldsymbol{0}$. Such an phenomena leads to a row-wise regularization term on V_i that exploits the isotropic orientation property of the texture patch stack:

$$\|\boldsymbol{H} \circ \boldsymbol{D} \circ V_i\|_1 = \|DV_iH^\top\|_1 \quad \text{(row-wise regularization along stacking axis)}$$

where H denotes the matrix form of the circulant matrix with the filter $[1, -1]$.

Remark 2.1. As the number of patches matched to the candidate patch is fixed, it happens that some candidate patches have more similar patches, and some have less similar patches. The outliers in patch matching are then often seen. Thus, we use ℓ_1-norm as the metric for better robustness to outliers.

Furthermore, recall that a texture patch refers to the one that shows strong oscillations, and thus we impose that its mean is close to 0, which leads to a column-wise regularization:

$$\|EV_i\|_1 \quad \text{(Column-wise regularization in image domain)}$$

where $E \in \mathbb{R}^{R\times M}$ presents the constant matrix with constant $= 1$. Overall, we propose the following regularization for a texture patch stack:

$$\Psi(V_j) = \underbrace{\beta_1\|EV_j\|_1}_{\text{column-wise}} + \underbrace{\beta_2\|DV_jH^\top\|_1}_{\text{row-wise}}. \tag{2.4}$$

See Figure 5(a) for an illustration.

Similarly, consider a matrix U_i that represents a cartoon patch stack. Then, for $k = 1, 2, \dots, 2RK$, the k-th column of $U_i \in \mathbb{R}^M$ represents a vectorized version of a 2D cartoon patch $\boldsymbol{u}_k \in \mathbb{R}^{m\times m}$. For each row of U_i, its entries are also drawn from the matched patches from R bands with different orientations. However, different from the texture patch stack, the cartoon patches matched to the candidate cartoon patch from different bands are not necessarily similar to the candidate patch. Indeed, only along one or two orientations, there exist patches that are similar to the candidate patch. In other words, the entries of each row are rather different such that the row shows strong oscillations. Such irregular oscillations are further amplified in the high-pass wavelet transform as the low-frequency components are suppressed.

For the matrix U_i, let WU_i denote the matrix whose k-th column is the vectorized form of the output of running a 2D high-pass wavelet transform on the patch $\boldsymbol{u}_k$. As each row of WU_i shows strong oscillations, the output of each row after convolved by a low-pass filtering, *e.g.* $[1, 1, \dots, 1]$ will be close to $\mathbf{0}$. Such an observation leads to the following row-wise regularization form on U_i:

$$\|\boldsymbol{L} \circ \boldsymbol{W} \circ U_i\|_1 = \|WU_iL^\top\| \quad \text{(row-wise regularization along stacking axis)}$$

where L is the matrix form of circulant convolution with the kernel $[1, 1, \dots, 1] \in \mathbb{R}^{2KR}$. Recall that a cartoon patch is modeled as a piece-wise smooth function. Then, we have the well-established high-pass wavelet transform based sparsity prior on the cartoon parch:

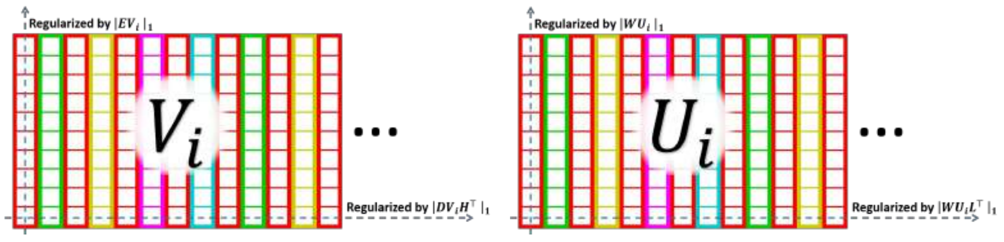
$$\|WU_i\|_1 \quad \text{(column-wise regularization in image domain)}$$

Combining both the high-pass wavelet transform based sparsity prior of cartoon patches in the image domain and the oscillation prior of cartoon patch stack along the stacking axis, we propose the following regularization for a cartoon patch stack:

$$\Phi(U_j) = \alpha_1\|WU_j\|_1 + \alpha_2\|WU_jL^\top\|_1. \tag{2.5}$$

See Figure 5(b) for an illustration. Based on the discussion above, we propose the following variational model for cartoon-texture decomposition over any patch stack $\tilde{S}_j$:

$$\min_{U_j,V_j} \Phi(U_j) + \Psi(V_j), \quad \text{subject to} \quad U_j + V_j = \tilde{S}_j. \tag{2.6}$$


Regularized by $|EV_i|_1$
V_i
Regularized by $|DV_iH^\top|_1$
Regularized by $|WU_i|_1$
U_i
Regularized by $|WU_iL^\top|_1$

(a) Texture patch stack. (b) Cartoon patch stack.

Figure 5. *Illustration of the regularizations on cartoon/texture patch stacks row-wise and column-wise.*

3. Variational Model on Image Pixels and Numerical Scheme.

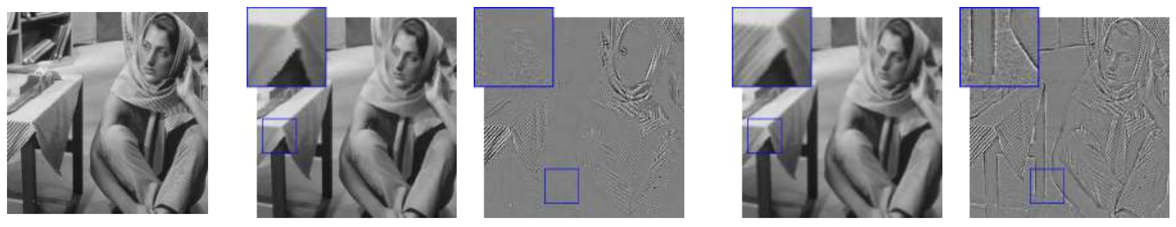
When partitioning an image into the set of small image patches $\{\boldsymbol{p}_i\}_i$ and grouping them into the set of matched stacks $\{\tilde{S}_j\}_j$, each image patch $\boldsymbol{p}_i$ will be contained in multiple stacks. Most non-local image denoisers such as BM3D [13] take a two-stage approach. Each patch stack is first processed independently such that there are multiple estimations of each image patch $\boldsymbol{p}_i$. Then, the final estimate of $\boldsymbol{p}_i$ is defined as a weighted average of multiple estimations. Such a two-stage approach is not suitable for cartoon-texture decomposition, as it will cause noticeable artifacts in the two layers. See Figure 6(c) for an illustration when taking such a two-stage approach. It can be seen that some patterns of the tablecloth remains in the cartoon layer and some cartoon edges are presented in the texture layer.

To avoid the artifacts caused by the two-stage approach discussed above, we map the regularization term (2.6) defined on patch stacks to the one directly defined on image pixels of two layers such that the values of these image pixels are directly estimated. See Figure 6(b) for an illustration when taking such a single-stage approach. It can be seen that there are much less artifacts in the two layers, in comparison to that shown in Figure 6(c).

Moreover, there is another advantage by directly defining the regularization terms on the pixels of the two layers. The regularization terms on the pixels involve much less unknowns than that on patch stacks. As a result, it consumes much less memory, which is very important for a GPU-based implementation. In our GPU-based implementation, the proposed single-stage approach requires about 8 minutes for processing an image of size 512×512 while the two-stage approach takes more than one hour.

3.1. Variational Model.

Recall that for each image patch $\boldsymbol{p}_i$, the operator $\boldsymbol{P}_i$ and $\boldsymbol{S}_i$ defined by (2.2) project the image $\boldsymbol{f}$ to the i-th patch and the associated patch stack. Let f, u, v denote the column vector form of the image $\boldsymbol{f}$ and two components $\boldsymbol{u}, \boldsymbol{v}$ by sequentially concatenating all column vectors of $\boldsymbol{f}, \boldsymbol{u}, \boldsymbol{v}$ respectively. Let P_i denote the matrix form of $\boldsymbol{P}_i$ that projects the vector f to the vector form of the i-th patch p_i, and let S_i denote the matrix



(a) Input (b) Results of proposed solver (c) Results of BM3D-like solver

Figure 6. *Decomposition results of the proposed solver and a BM3D-like solver on "Barbara". Compared with the proposed solver, the BM3D-like solver generate obvious artifacts on both cartoon and texture component.*

form of $\boldsymbol{S}_i$ that projects the vector f to the vector form of the i-th patch stack. Then, we have

$$u_i = P_i u, \quad \bar{U}_i = S_i u,$$

$$v_i = P_i v, \quad \bar{V}_i = S_i v.$$

Based on two regularization terms (2.4) and (2.5) on texture and cartoon patch stacks, we propose the following variational model for cartoon-texture decomposition:

$$\text{(3.1)} \qquad \min_{u,v} \ \phi(u) + \psi(v) \quad \text{s.t.} \quad u + v = f,$$

with

$$\text{(3.2)} \qquad \begin{cases} \phi(u) &= \alpha_1 \sum_i \lambda_i^{\mathrm{c}} \|W P_i u\|_1 + \alpha_2 \sum_i \lambda_i^{\mathrm{c}} \|(L \otimes W) S_i u\|_1, \\ \psi(v) &= \beta_1 \sum_i \lambda_i^{\mathrm{t}} \|E P_i v\|_1 + \beta_2 \sum_i \lambda_i^{\mathrm{t}} \|(H \otimes D) S_i v\|_1, \end{cases}$$

where $\{\lambda_i^{\mathrm{c}}, \lambda_i^{\mathrm{t}}\}_i$ are parameters that balance two regularization terms on cartoon and texture, which vary for different patches. In the next, we will discuss how to set these parameters.

The regularization parameters

$$\lambda^{\mathrm{c}} = [\lambda_1^{\mathrm{c}}, \lambda_2^{\mathrm{c}}, \cdots, \lambda_N^{\mathrm{c}}] \quad \text{and} \quad \lambda^{\mathrm{t}} = [\lambda_1^{\mathrm{t}}, \lambda_2^{\mathrm{t}}, \cdots, \lambda_N^{\mathrm{t}}]$$

play an important role in the model. Intuitively, λ_i^{c} should be large at texture regions and small at contour regions. Oppositely, λ_i^{t} should be small at texture regions and large at contour regions. Based on the orientation property of patch recurrence for cartoon/texture, we propose the following scheme to determine the regularization parameters:

$$\text{(3.3)} \qquad \lambda_i^{\mathrm{c}} = 1 - e^{-\eta_1 \rho_i^{\mathrm{c}}}, \lambda_i^{\mathrm{t}} = 1 - e^{-\eta_2 \rho_i^{\mathrm{t}}},$$

where η_1 and η_2 are two parameters to be manually determined. The range of $\lambda_i^{\mathrm{c}}, \lambda_i^{\mathrm{t}}$ is in $(0, 1)$. In (3.3), ρ_i^{c} is the quantity that measures the degree of isotropy, which is defined by

$$\text{(3.4)} \qquad \rho_i^{\mathrm{c}} = \sum_{r=1}^{R} \sum_{j \in \mathbb{S}_i^{(r)}} ||P_i f - P_j f||_2^2,$$

and ρ_i^{t} is the quantity that measures the degree of anisotropy, which is defined by

$$\rho_i^{\mathrm{t}} = \frac{\sum_{j\in\mathbb{S}_i^{(d)}} ||P_i f - P_j f||_2^2}{\sum_{k=1}^{R}\sum_{j\in\mathbb{S}_i^{(k)}} ||P_i f - P_j f||_2^2} \text{ with } d = \operatorname*{argmin}_{1\le r\le R} \sum_{j\in\mathbb{S}_i^{(r)}} ||P_i f - P_j f||_2^2. \tag{3.5}$$

It can be seen that $\lambda_i^{\mathrm{c}}/\lambda_i^{\mathrm{t}}$ is large/small if the corresponding regions exhibit strong isotropic patch recurrence which implies more/less confidence on texture/cartoon, and vice versa. See Figure 7 for an illustration of λ^{t}.

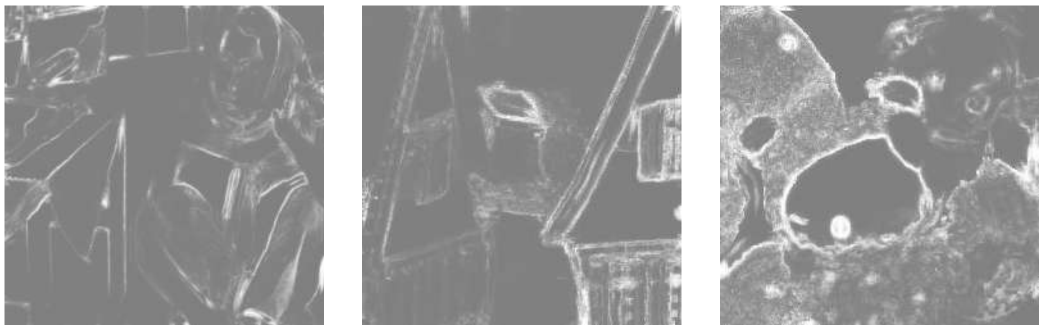

(a) Image "Barbara" (b) Image "House" (c) Image "Friends"

Figure 7. *Illustration of the regularization parameter vector λ^t on three images. Darker color denotes smaller values.*

3.2. Numerical Scheme. The optimization problem (3.1) is an ℓ_1-norm-relating problem, and there exist many well-established efficient numerical solvers for such a type of problems, *e.g.* the ADMM method [5] and the split Bregman iterative method [23]. Define a variable

$$x = \begin{pmatrix} u \\ v \end{pmatrix},$$

and define $A = [I, I]$. Note that the two regularization terms defined in (3.2) are the weighted sums of ℓ_1-norms of the linear measurements on u and on v. Then, we rewrite $\phi(u) + \psi(v)$ as

$$\|\mathrm{diag}(\lambda)\Gamma x\|_1,$$

where $\lambda = [\lambda_1^c, \lambda_1^t, \cdots, \lambda_N^c, \lambda_N^t]$ and Γ is the matrix that maps x to those linear measurements:

$$\Gamma : x = \begin{pmatrix} u \\ v \end{pmatrix} \longrightarrow \left[\begin{pmatrix} \alpha_1 W P_i u & \alpha_2 E P_i v \\ \beta_1 (L\otimes W) S_i u & \beta_2 (H \otimes D) S_i v \end{pmatrix}\right]_i^{\top}. \tag{3.6}$$

Then, the problem (3.1) can be re-expressed as

$$\min_x \|\mathrm{diag}(\lambda)\Gamma x\|_1, \quad \text{s.t.} \quad Ax = f. \tag{3.7}$$

The ADMM method is called for solving (3.7). For the completeness, the detailed description of the method is given below.

By introducing an auxiliary variable y, we rewrite (3.7) as follows:

$$\min_x \|\mathrm{diag}(\lambda)y\|_1, \quad \text{s.t.} \quad y = \Gamma x, \quad Ax = f, \tag{3.8}$$

The problem (3.8) can be solved via the following alternating iteration scheme: for $k = 0, 1, \ldots,$

$$\begin{cases} (x^{(k+1)}, y^{(k+1)}) = \mathrm{argmin}_{x,y} \ \|\mathrm{diag}(\lambda)y\|_1 + \frac{\gamma_1}{2}\|Ax - f + e^{(k)}\|_2^2 + \frac{\gamma_2}{2}\|\Gamma x - y + b^{(k)}\|_2^2, \\ b^{(k+1)} = b^{(k)} + \delta(\Gamma x^{(k+1)} - y^{(k+1)}), \\ e^{(k+1)} = e^{(k)} + \delta(Ax^{(k+1)} - f), \end{cases} \tag{3.9}$$

where $\gamma_1, \gamma_2 \in \mathbb{R}^+$, $\delta \in (0, 1]$. The first sub-problem in (3.9) is also solved by an alternating iterative scheme, which leads to the following iterative scheme:

$$\begin{cases} x^{(k+1)} = \underset{x}{\mathrm{argmin}} \ \gamma_1\|Ax - f + e^{(k)}\|_2^2 + \gamma_2\|\Gamma x - y^{(k)} + b^{(k)}\|_2^2, \\ y^{(k+1)} = \underset{y}{\mathrm{argmin}} \|\mathrm{diag}(\lambda)y\|_1 + \frac{\gamma_2}{2}\|\Gamma x^{(k+1)} - y + b^{(k)}\|_2^2, \\ b^{(k+1)} = b^{(k)} + \delta(\Gamma x^{(k+1)} - y^{(k+1)}), \\ e^{(k+1)} = e^{(k)} + \delta(Ax^{(k+1)} - f). \end{cases} \tag{3.10}$$

In the iterative scheme above, the first sub-problem when updating x has an analytic solution given by

$$x^{(k+1)} = (\gamma_1 A^\top A + \gamma_2 \Gamma^\top \Gamma)^{-1}(\gamma_1 A^\top (f - e^{(k)}) + \gamma_2 \Gamma^\top (y^{(k)} - b^{(k)})), \tag{3.11}$$

which is computed via the conjugate gradient (CG) method in the implementation. The second sub-problem when updating y also has an analytic solution:

$$y^{(k+1)} = \boldsymbol{T}_{\frac{\lambda}{\gamma_2}}(\Gamma x^{(k+1)} + b^{(k)}), \tag{3.12}$$

where $\boldsymbol{T}_\beta(\cdot)$ is the element-wise operator that applies the soft-thresholding operation on each element of the input:

$$(\boldsymbol{T}_\beta(x))_\ell = \mathrm{sgn}(x_\ell)\max(|x_\ell| - \beta_\ell, 0). \tag{3.13}$$

See Algorithm 3.1 for the outline of the proposed cartoon-texture decomposition method.

4. Experimental Evaluation.

The implementation details of the proposed method used in the experiments are listed as follows. The wavelet transform used in the method is the 2D single-level undecimal linear spline wavelet framelet transform [14] whose filter bank is the set of tensor product of the following 3 filters:

$$h_0 = \frac{1}{4}[1, 2, 1]; \ h_1 = \frac{\sqrt{2}}{4}[1, 0, -1]; \ h_2 = \frac{1}{4}[-1, 2, -1].$$

The DCT used in the implementation is of size 5×5. Recall that only wavelet transform and DCT in high-pass channels are used in the proposed method. The maximal number of iteration

```
Algorithm 3.1 Cartoon-Texture Decomposition
Input: Image $\boldsymbol{f}$
Output: Cartoon component $\boldsymbol{u}$, Texture component $\boldsymbol{v}$,
Main procedure:
    1. Partition images into patch set $\{\boldsymbol{p}_i\}$.
    2. Run the routine of patching matching and assembly patch stacks $\{\tilde{S}_i\}_i$ by (2.1).
    3. Construct the matrix $A = [I, I]$.
    4. Construct the matrix $\Gamma$ by (3.6).
    5. Calculate the parameter vector $\lambda$ by (3.3).
    6. Set $x^{(0)} := [f; 0]$,$y^{(0)} := \Gamma x^{(0)}, b^{(0)} = e^{(0)} := 0$;
    7. For $k = 0, \cdots, k_0 - 1$:
        $\begin{cases} x^{(k+1)} := (\gamma_1 A^\top A + \gamma_2 \Gamma^\top \Gamma)^{-1}(A^\top(f - e^{(k)}) + \gamma_2 \Gamma^\top (y^{(k)} - b^{(k)})), \\ y^{(k+1)} := \boldsymbol{T}_{\lambda/\gamma_2}(\Gamma x^{(k+1)} + b^{(k)}), \\ b^{(k+1)} := b^{(k)} + \delta(\Gamma x^{(k+1)} - y^{(k+1)}), \\ e^{(k+1)} := e^{(k)} + \delta(A x^{(k+1)} - f). \end{cases}$
    8. $u := [I, \mathbf{0}]\ x^{(k_0)}$, and $v := [\mathbf{0}, I]\ x^{(k_0)}$.
```

is set to $k_0 = 30$. For patch matching, the size of patch is set to 5×5, the neighborhood size for patch matching is set to 51×51, the number of bands is set to 4, the number of matched patches along each band is $K = 16$, and the similarity between patches is measured by ℓ_2-distance. For the regularization parameters, we set $\eta_1 = 20$ and $\eta_2 = 0.0004$. For the numerical solver, the parameter γ_1 is set to 1, γ_2 is set to 0.05 and the update step δ is set to 1. For the model parameters in (1.2), we have different settings for different configurations of the problems. For cartoon-texture decomposition over a full image, the parameters are set as $\alpha_1 = 0.25, \alpha_2 = 0.20, \beta_1 = 0.12, \beta_2 = 0.03$. For cartoon-texture decomposition over an image with missing pixel values, the parameters are set as $\alpha_1 = 0.40, \alpha_2 = 0.20, \beta_1 = 0.12, \beta_2 = 0.03$. Our method is implemented in Matlab with GPU acceleration. For easier visual inspection, $\tilde{v} = 0.5 + 3v$ is shown as the texture component in all the figures of this section.

A more comprehensive evaluation on the proposed cartoon-texture decomposition method is conducted with different perspectives, which includes the experiments on synthetic images for quantitative evaluation, the experiments on real images for visual evaluation, and the experiments on the images with missing pixel values for robustness evaluation.

The methods for comparison are selected so that they can cover different types of the approaches for cartoon-texture decomposition, including

- Ng *et al.* [33], one of the most recent PDE-based methods, where the TV norm and its dual are used to characterize cartoon and texture respectively. We use the algorithm 2 in [33] for comparison.
- Ono *et al.* [34], which utilizes the low-rank property of texture patches;
- Ma *et al.* [30], which forms the groups of similar image patches and conducts low-rank decomposition on each matched patch group;
- Papyan *et al.* [37], which uses convolutional sparse dictionary learning to discover the underlying patterns of cartoon and texture.
- Gu *et al.* [24], which integrates the analysis operator and synthesis operator in convo-

lutional sparse coding to better model cartoon and texture.

- Sur *et al.* [44], one of the latest non-local methods that regularizes the group of matched patches by some power-spectrum-based statistical measurements.

4.1. Quantitative Evaluation on Synthetic Images. We first conduct the experiments on 120 synthetic images, which are synthesized as follows. For cartoon component, we generated some piece-wise constant images in the following steps: (i) randomly/manually select several seed points; (ii) divide image pixels into several non-overlapping groups, which is done via the spatially-nearest-neighbor clustering using the seed points as centers, with the pth-order Minkowski distance; (iii) assign a unique pixel value to each group of pixels. See Figure 8(a) for some samples of synthesized cartoon layers. Note that the region boundaries are line segments when $p = 2$ and become curves when $p > 2$. In addition, we also include some cartoon images downloaded from the internet, including icons, logos and cartoon characters. See Figure 8(b) for some samples.

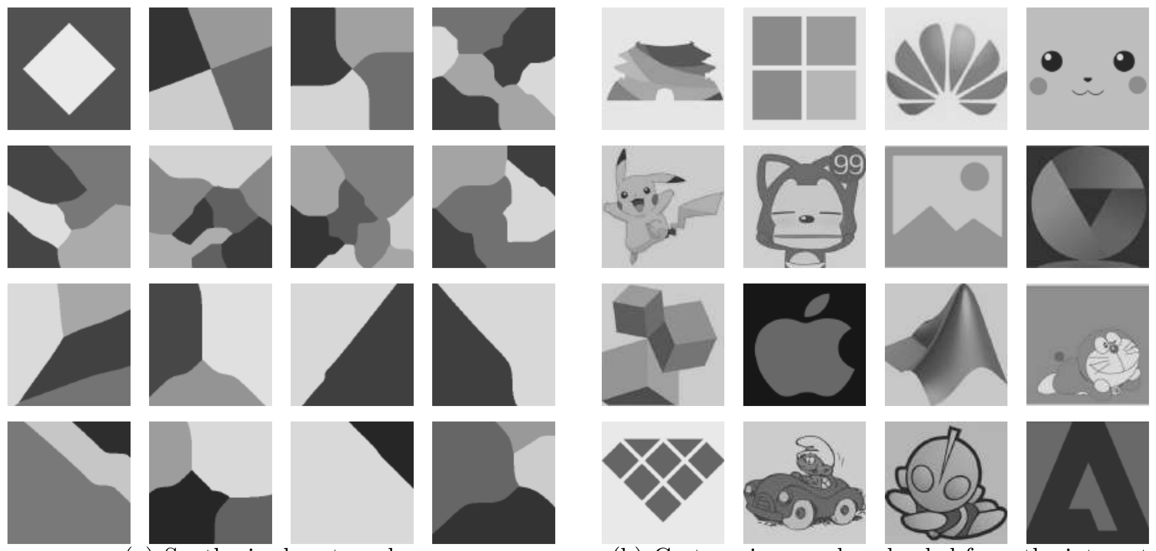

(a) Synthesized cartoon layers (b) Cartoon images downloaded from the internet

Figure 8. *Examples of ground-truth cartoon layers.*

For texture layer, they are collected from several datasets of natural texture images available online, including Brodatz [46], Kylberg [28], KTH-TIPS [19] and DTD [12]. See Figure 9 for some samples.

Two schemes are adopted to synthesize test images for cartoon-texture decomposition using ground-truth cartoon layer and texture layer: (1) weighted average of two layers; (2) using a randomly-generated cartoon region mask, assign different texture layers to different cartoon regions. See Figure 10 for some examples of the synthesized images.

The quantitative evaluation is based on two metrics: peak signal-to-noise ratio (PSNR) and Structural Similarity Index (SSIM). See Table 1 for the list of the comparison of tested methods in terms of average PSNR/SSIM on test dataset. It can be seen that the proposed method is the best performer among all methods in terms of both PSNR and SSIM.

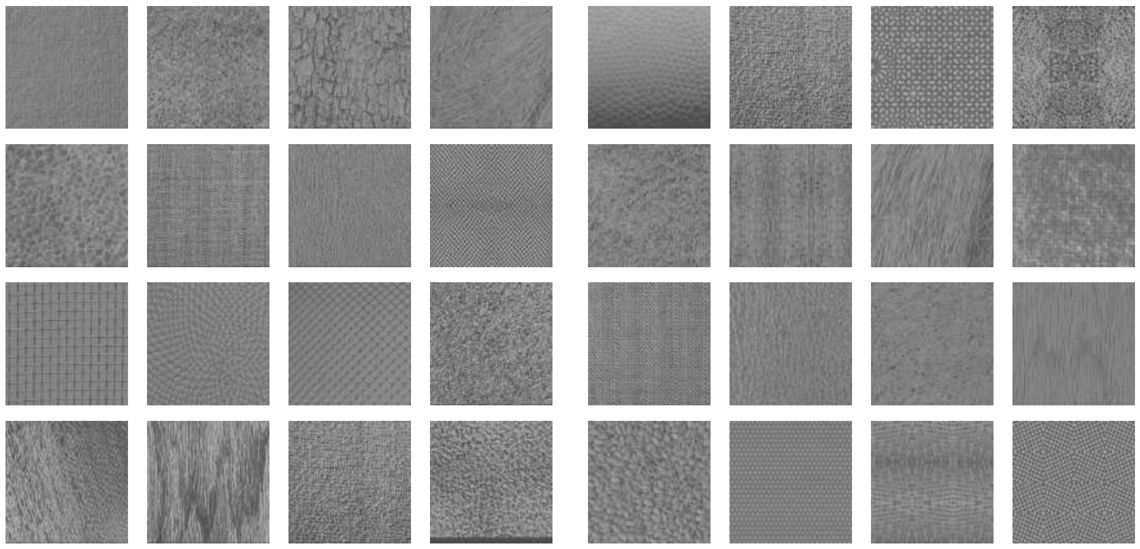


Figure 9. *Examples of ground-truth texture components.*

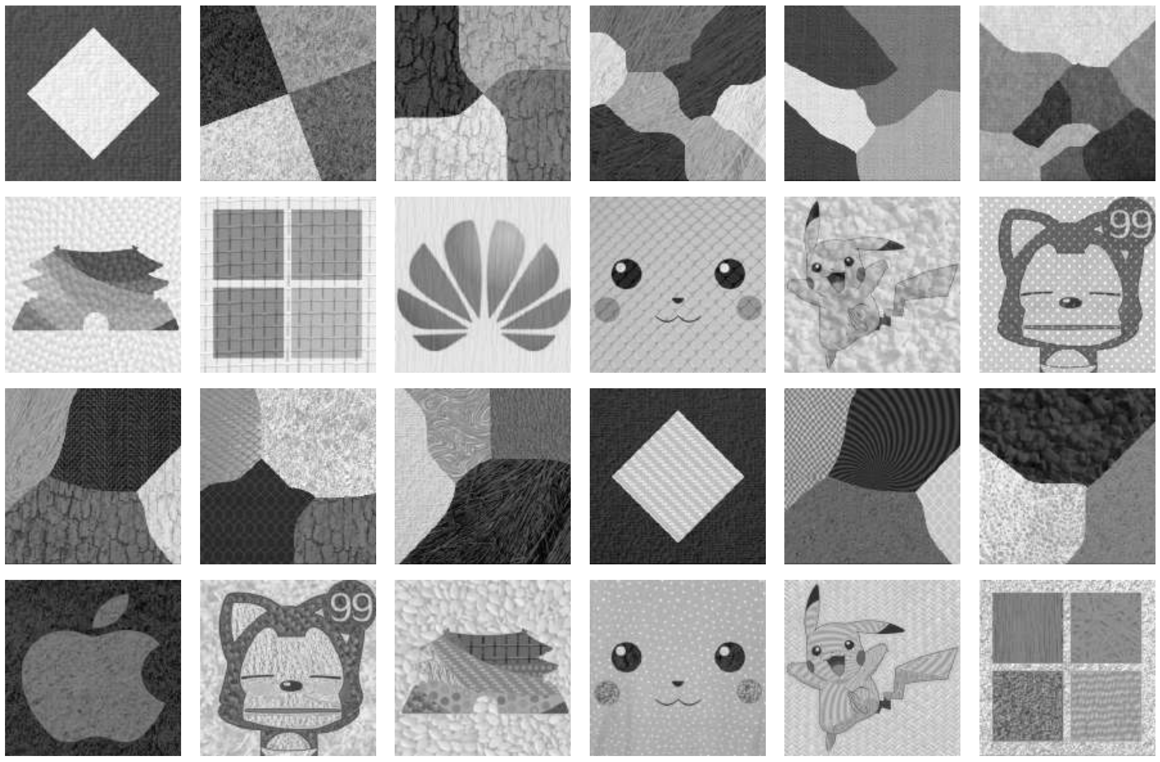


Figure 10. *Examples of synthesized images.*

The advantage of the proposed method over other methods is also shown in terms of visual quality. See Figure 11 for the visualization of some results. For the cartoon layer, it can be

Table 1
Average PSNR (dB) and SSIM values of the decomposition results on synthetic images by different methods.

Criterion	Ng [33]	Ono [34]	Sur [44]	Ma [30]	Gu [24]	Papyan [37]	Ours
PSNR(Cartoon)	31.42	26.98	28.43	26.67	26.94	29.72	**33.37**
PSNR(Texture)	28.70	26.98	28.43	26.70	25.86	29.74	**33.26**
SSIM(Cartoon)	0.948	0.596	0.696	0.570	0.753	0.777	**0.964**
SSIM(Texture)	0.900	0.735	0.839	0.754	0.796	0.894	**0.965**

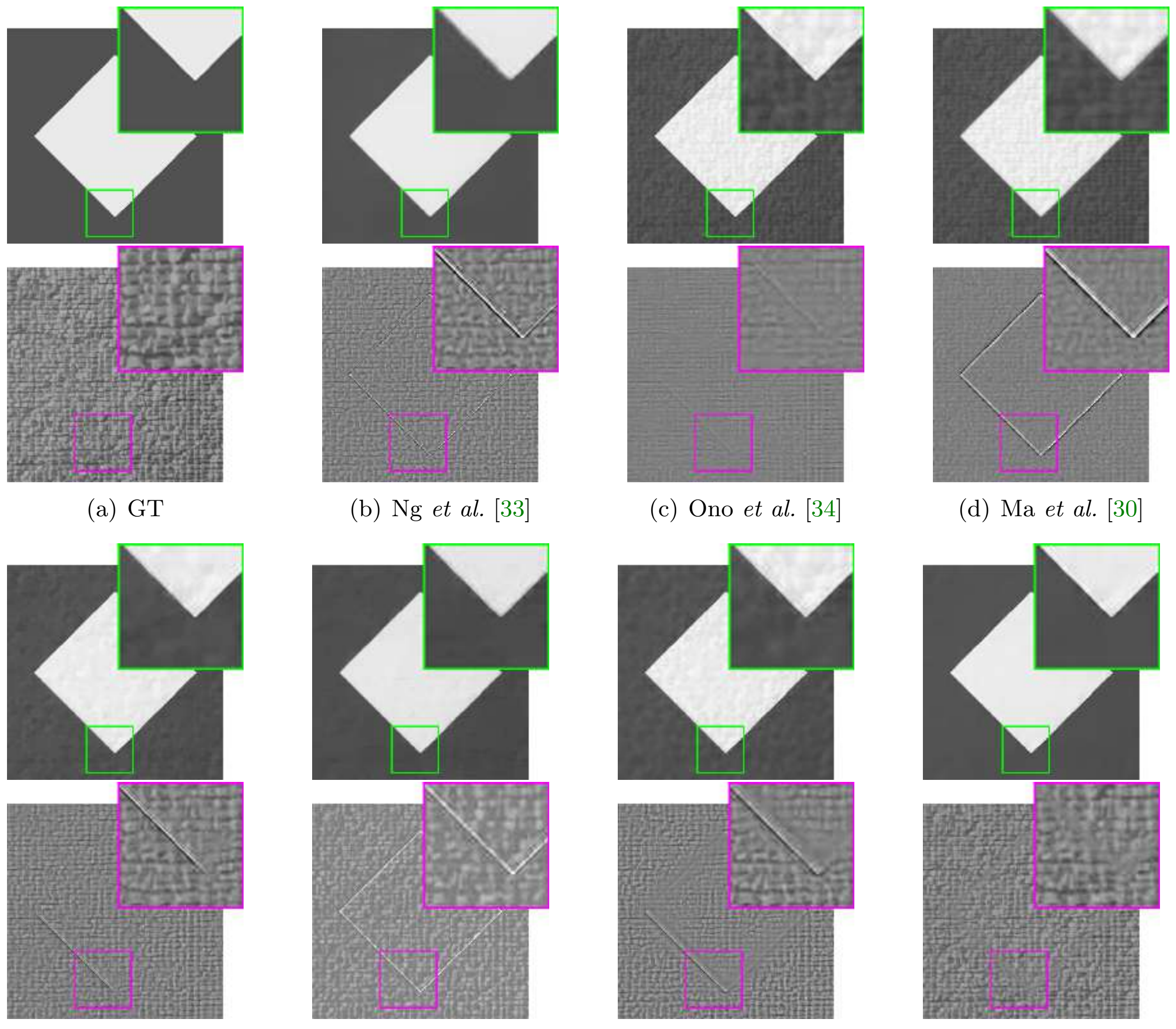

(a) GT (b) Ng *et al.* [33] (c) Ono *et al.* [34] (d) Ma *et al.* [30]
(e) Papyan *et al.* [37] (f) Gu *et al.* [24] (g) Sur *et al.* [44] (h) Ours

Figure 11. *The cartoon layer (top) and the texture layer (bottom) of the decomposition results on 'Diamond' (i.e. the 1st synthetic images). (a) Ground truth. (b)-(g) the results of different methods.*

seen that the existing methods either produce blurred edges such as Ng *et al.* [33], or wrongly contain texture patterns such as Ono *et al.* [34], Ma *et al.* [30], Papyan *et al.* [37] and Sur *et al.* [44]. In contrast, the result from the proposed method keeps sharp edge and does not contain texture patterns. For texture layer, the existing methods either produce incorrect texture patterns such as the Ono *et al.* [34], or wrongly contain line segmentation from the cartoon layer such as Ng *et al.* [33], Ma *et al.* [30], Gu *et al.* [25], Papyan *et al.* [37] and Sur *et al.* [44]. Again, the result of the proposed method produces the most accurate texture layer.

4.2. Experiments on Real Images. The methods are also evaluated on 5 real images that are used in existing literatures (*e.g.* [34, 30, 37]). These 5 images contain different types of textures, including both natural textures with strong randomness and man-made textures with regular patterns. There is no standard evaluation strategy for cartoon-texture decomposition. Thus, the evaluation only can be done by visual inspection. We consider it an accurate cartoon-texture decomposition if (1) object contours only appear in the cartoon layer; (2) image features with highly random or highly oscillating pattern only appear in the texture layer; and (3) no noticeable artifacts appear in both layers. See Figure 12-15 for the visual inspection of the results from different methods.

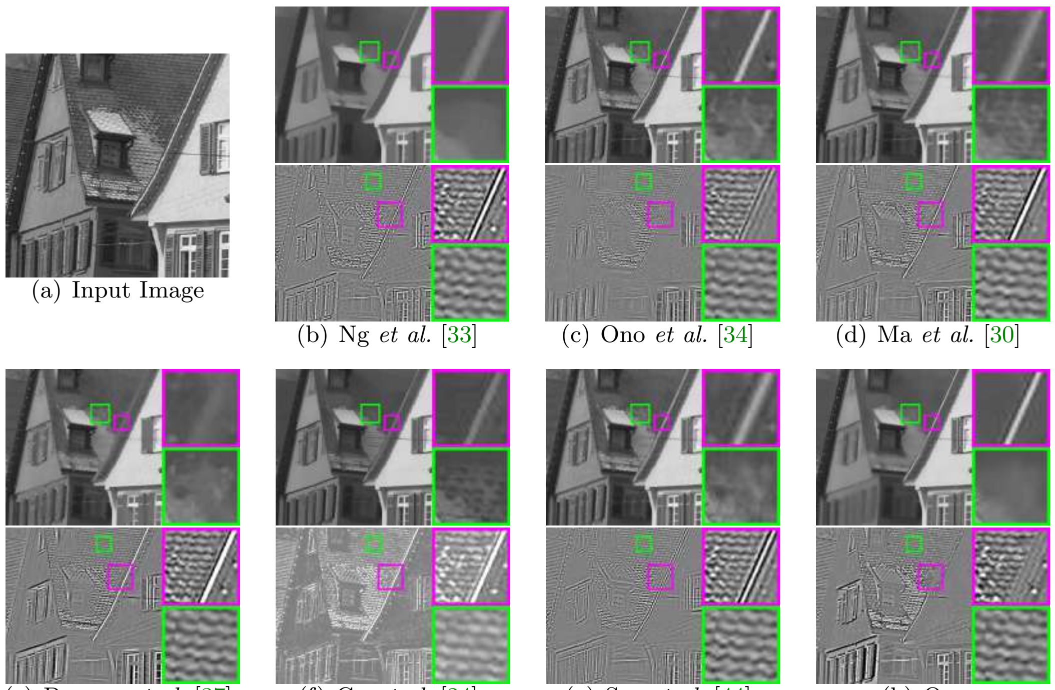

(a) Input Image (b) Ng *et al.* [33] (c) Ono *et al.* [34] (d) Ma *et al.* [30]
(e) Papyan *et al.* [37] (f) Gu *et al.* [24] (g) Sur *et al.* [44] (h) Ours

Figure 12. *The input image (a) and the decomposition results of different methods on test image 'House' (b-h). The cartoon layers are at the top, and the texture layers are at the bottom.*

For image "House", it can be seen that the proposed method did well at keeping contour edges in the cartoon layer. For instance, the eave is completely kept only in the cartoon layer. In contrast, other methods, except Ono *et al.* [34], wrongly assigned a strong edge along the eave to the texture layer, which results in the blurring or even removing of the eave in the

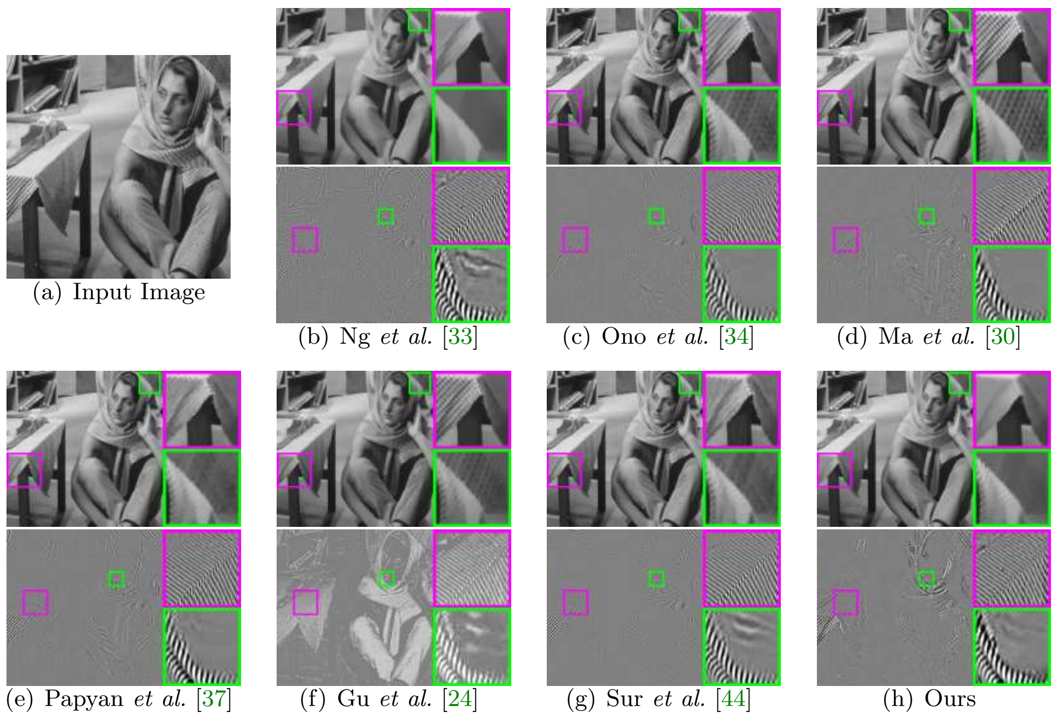


Figure 13. *The input image (a) and the decomposition results of different methods on test image 'Barbara' (b-h). The cartoon layers are at the top, and the texture layers are at the bottom.*

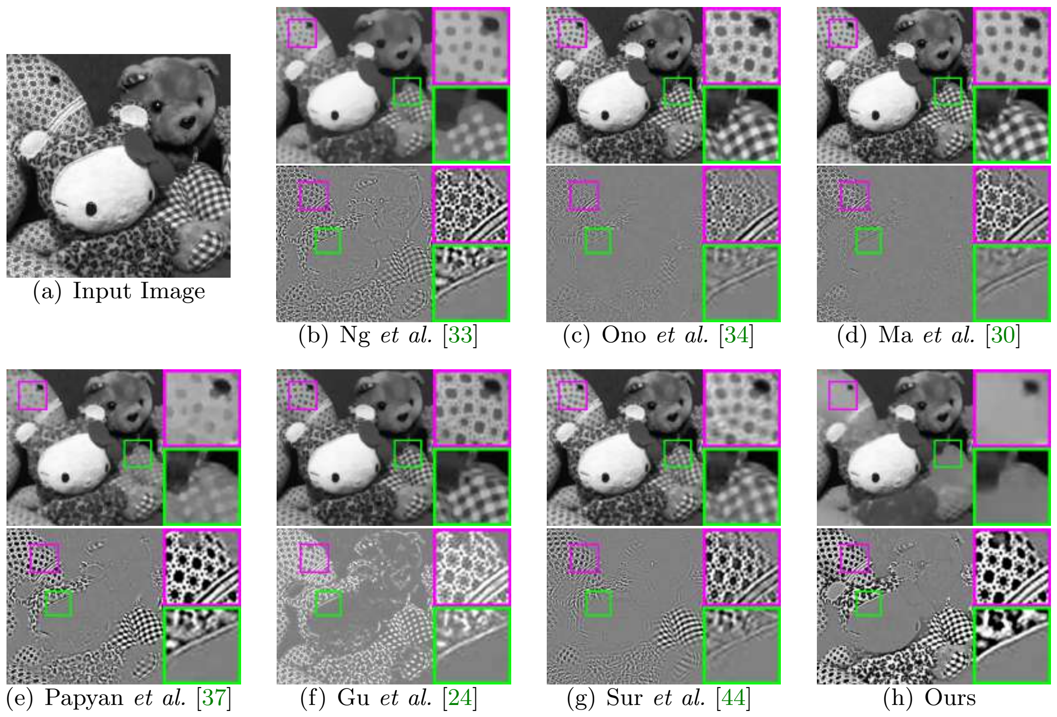


Figure 14. *The input image (a) and the decomposition results of different methods on test image 'Friends' (b-h). The cartoon layers are at the top, and the texture layers are at the bottom.*

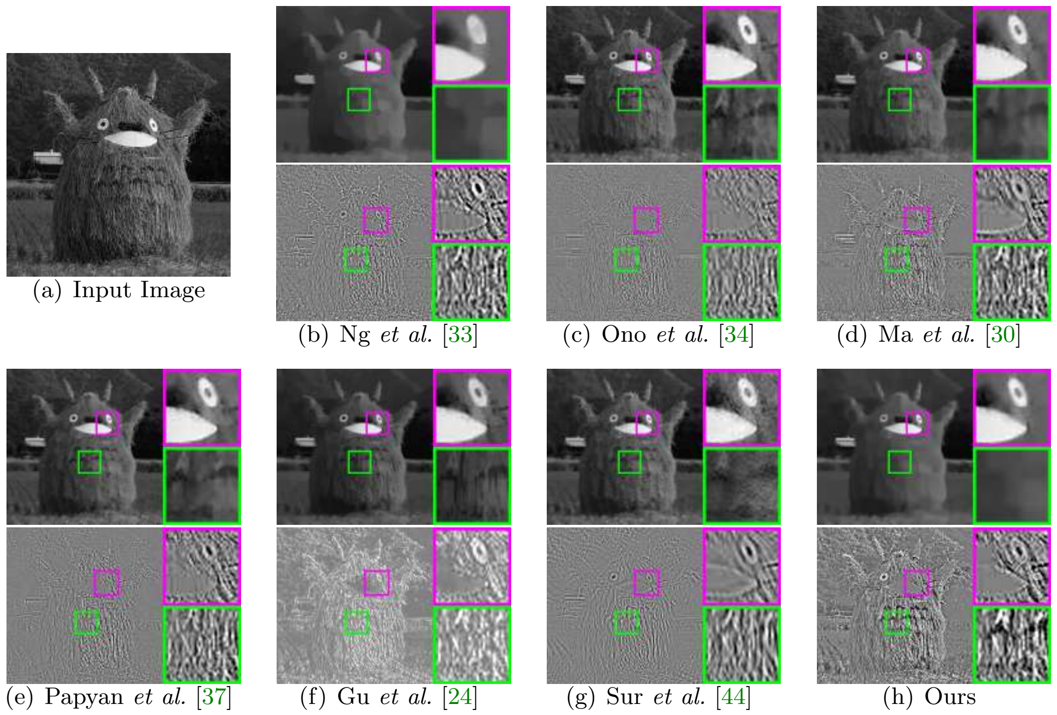


Figure 15. *The input image (a) and the decomposition results of different methods on test image 'Jackstraw' (b-h). The cartoon layers are at the top, and the texture layers are at the bottom.*

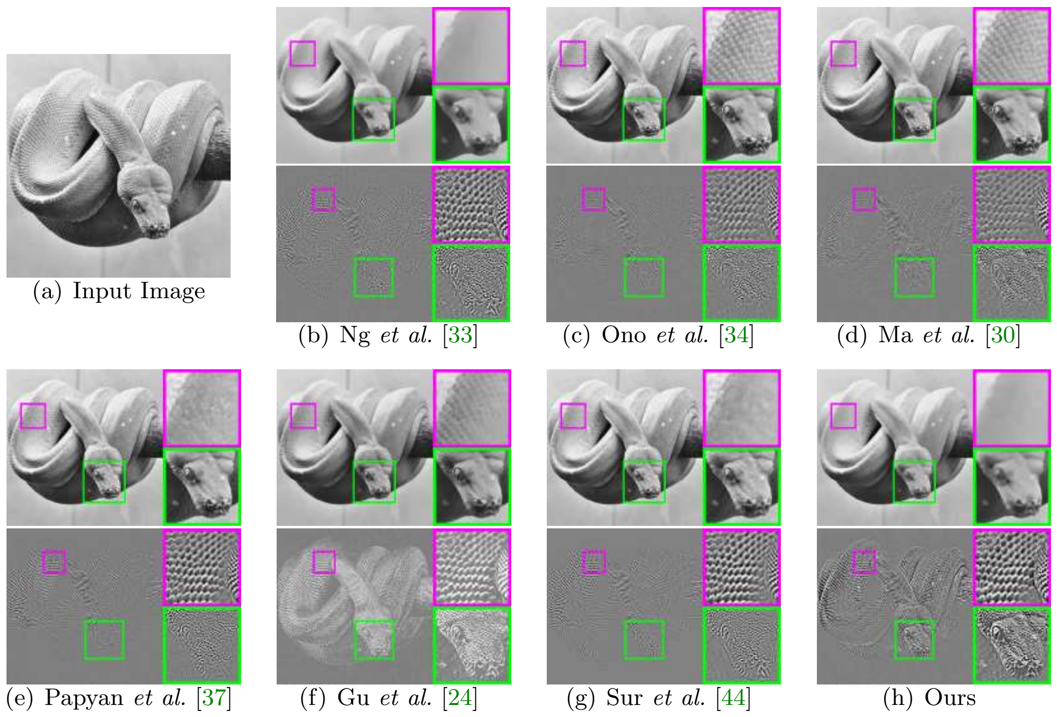


Figure 16. *The input image (a) and the decomposition results of different methods on test image 'Snake' (b-h). The cartoon layers are at the top, and the texture layers are at the bottom.*

cartoon layer. In particular, two patch-recurrence-based methods including Ma *et al.* [30] and Sur *et al.* [44] produced noticeable blurring on the eave. For image "Barbara", the proposed method is capable of keeping image features with periodic patterns only in the texture layer. In addition, only Ng *et al.* [33] and the proposed method completely removed the textures from the cartoon layer. In comparison to Ng *et al.* [33] which obviously blurred the cartoon layer, ours produced sharper edges. For the other three images "Friends", "Jackstraw" and "Snake", it can be seen that except the proposed method, all other methods either wrongly assigned straw textures to the cartoon layer or wrongly assigned object contours to the texture layer. Overall, our method is the most accurate one that separates object contours and textures.

The proposed method can be extended to processing color images by simply processing each color channel separately. See Figure 17 for the visualization of the results of the proposed method on the color version of two images, "Barbara" and "Friends". It can be seen that the results generated by such a simple process does not produce noticeable color artifacts.

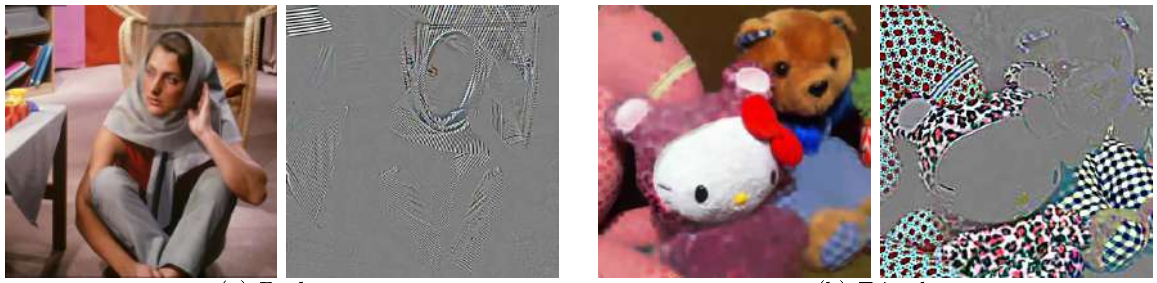

(a) Barbara (b) Friends

Figure 17. *The decomposition results of the proposed method on color images.*

In summary, visual inspection of the results from different methods showed that the proposed one outperformed the existing ones in terms of decomposition accuracy, which indicates the effectiveness of orientation characteristic of patch recurrence in natural images for distinguishing cartoon and texture.

4.3. Decomposition of Images with Missing Pixel Values. In this experiment, the proposed method is used for cartoon-texture decomposition of images with missing pixel values. Such a decomposition can see its application in image inpainting, as it is shown in [34] that running different inpainting schemes on cartoon part and texture part will lead to better results in in-painting. The proposed method can be extended for solving such a problem by replacing the constraint $u + v = f$ in (3.1) by

$$\Xi(u + v) = \Xi f,$$

where Ξ is a diagonal matrix whose diagonal entry is 1 if its corresponding pixel value is available and 0 otherwise. The resulting algorithm only needs a minor modification on Algorithm 3.1, *i.e.* simply replacing $A = [I, I]$ by $A = [\Xi, \Xi]$. After running the cartoon-texture decomposition, we can have an estimation on the image f:

$$\tilde{f} = \Xi f + (1 - \Xi)(u + v).$$

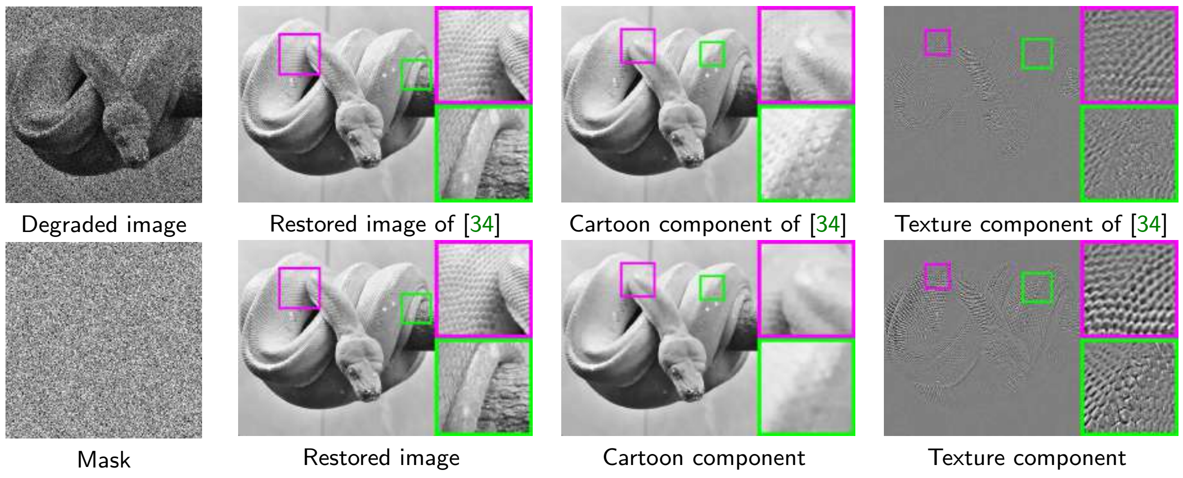
Table 2
PSNR (dB) and SSIM values of inpainted images

(a) 40% pixel values are missing

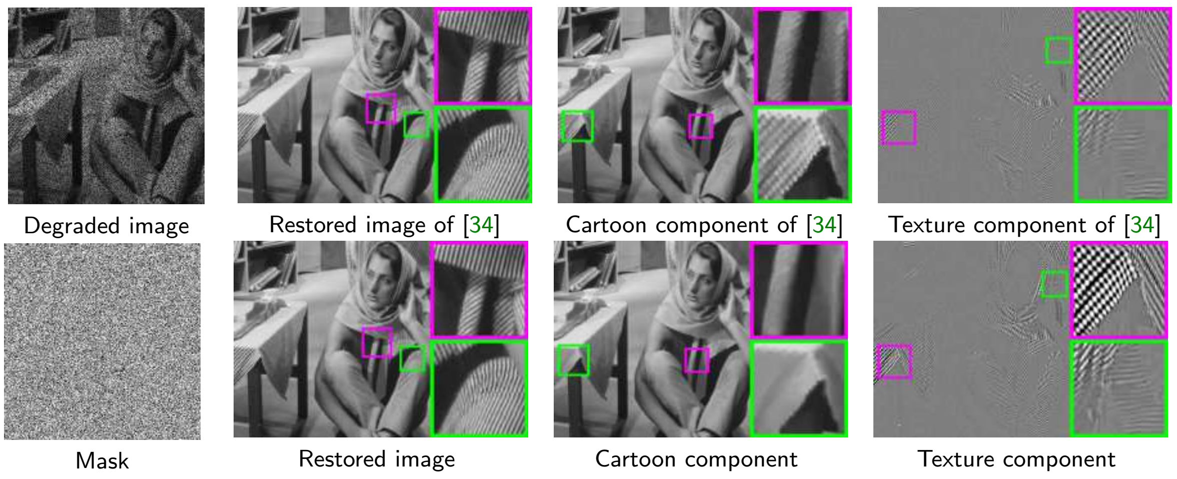
Method	Measure	Barbara	House	Jackstraw	Snake
Ono [34]:	PSNR	32.87	28.99	28.56	28.93
	SSIM	0.959	0.922	0.860	0.905
Ours	PSNR	**34.86**	**30.10**	**28.95**	**29.70**
	SSIM	**0.968**	**0.937**	**0.870**	**0.918**

(b) 50% pixel values are missing

Method	Measure	Barbara	House	Jackstraw	Snake
Ono [34]:	PSNR	30.58	27.39	27.23	27.54
	SSIM	0.934	0.886	0.805	0.870
Ours	PSNR	**32.16**	**28.30**	**27.64**	**28.13**
	SSIM	**0.948**	**0.903**	**0.817**	**0.880**

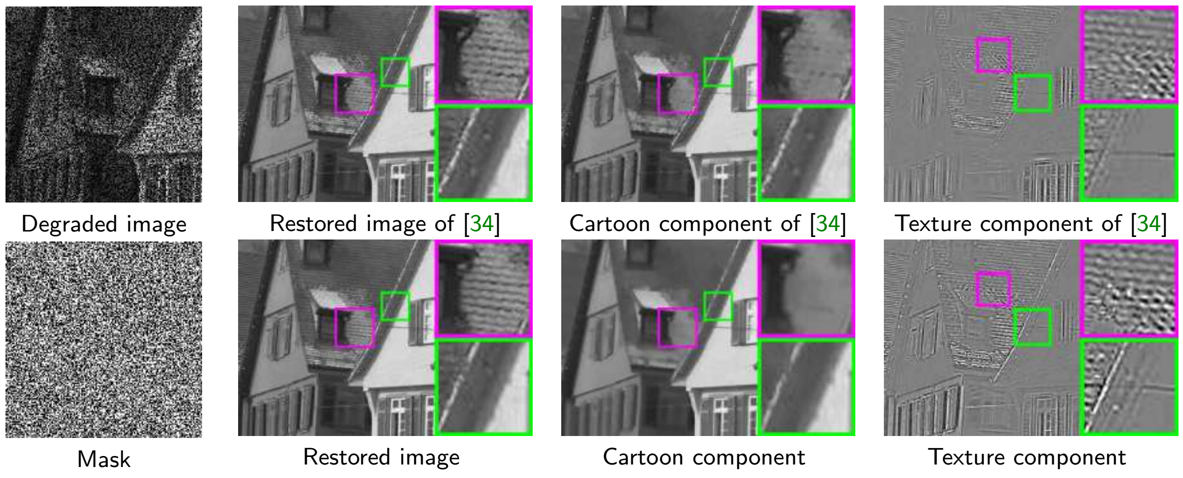


(a) Results on "Snake"

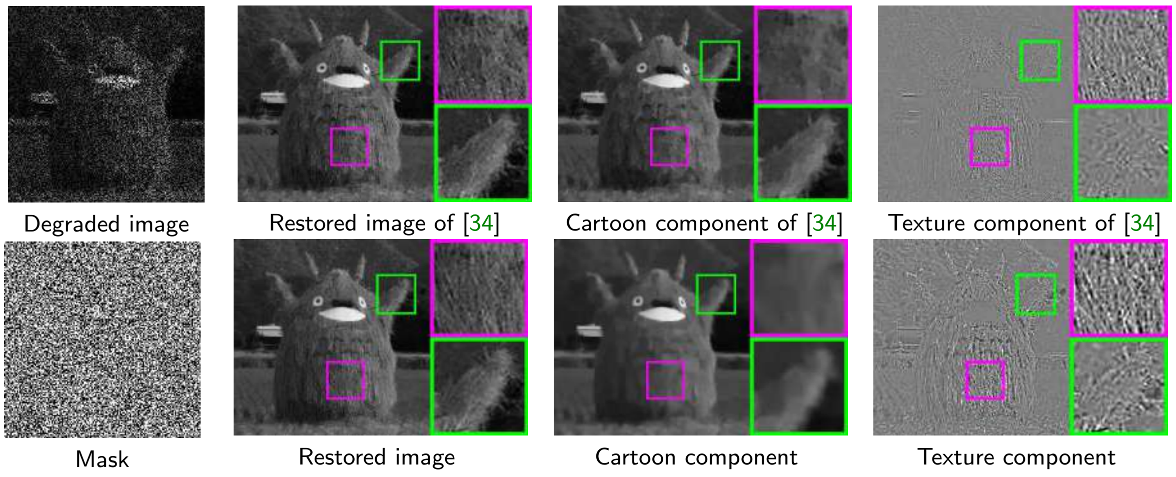


(b) Results on "Barbara"

Figure 18. *Our decomposition results on "Snake" and "Barbara" with* 40% *pixels missing.*

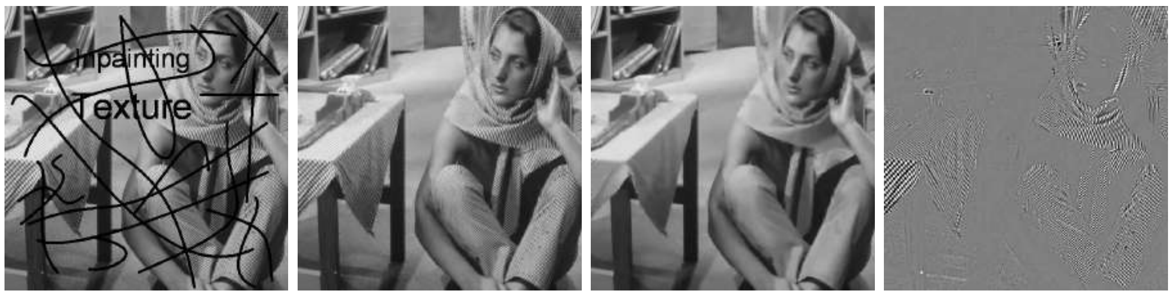


(a) Results on “House”



(b) Results on “Jackstraw”

Figure 19. *Results on “House” and “Jackstraw” with* 50% *pixels missing.*



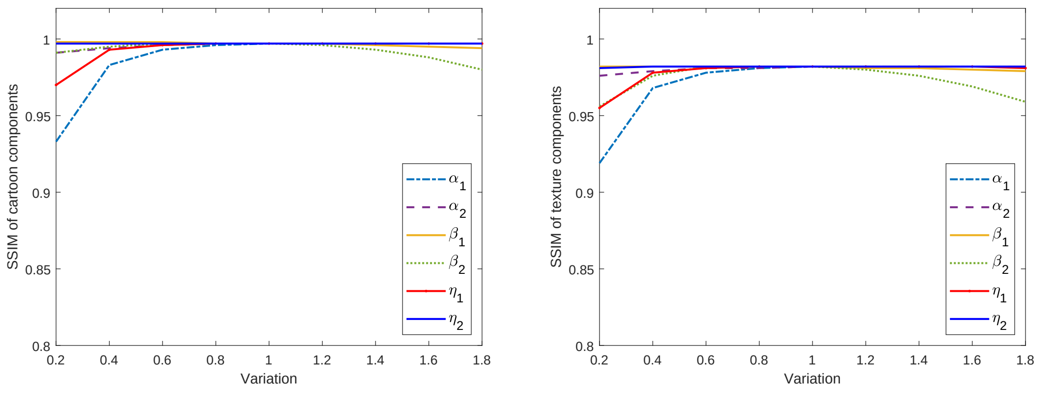
(a) Degraded image (b) Restored image (c) Cartoon component (d) Texture component

Figure 20. *Results on 'Barbara' with pixels in a generated mask missing.*

Such an extended version of the proposed method is tested on four test real images shown in the last subsection with missing rate of 40% and 50%, and is compared to Ono *et al.* [34]. See Table 2 for the comparison of inpainting performance in terms of PSNR/SSIM. It can be seen that the proposed method outperformed Ono *et al.*[34] by a large margin on the test images. See Figure 18 for visual illustration of two results on “Snake” and “Barbara” with missing rate of 40%. It can be seen that Ono *et al.* [34] wrongly assigned nearly all textures on the snake and the tie of Barbara to the cartoon layer, which makes the resulting texture layer rather weak. In comparison, our method produces a much clearer cartoon layer and a texture layer, which leads to better inpainted results. In Figure 19, we show two results on “House” and “Jackstraw” with missing rate of 50%. Similar phenomenon can be observed.

In addition, we also tested the performance of the proposed method in the case that the missing pixels are regular patterns such as text and scratches. The degraded image and the corresponding results are shown in Figure 20. Since such masks may bring undesired patterns with isotropic or anisotropic recurrence, we initialize the input image with a linear interpolation before running the proposed algorithm. From Figure 20, it can be seen that the propose method can restore the image well and generate an accurate decomposition.

4.4. Sensitivity to Parameter Settings. In this experiments, we tested how the performance of the proposed method is sensitive to the setting of the parameters involved in the algorithm. The evaluation is done by modifying the parameters, α_1, α_2, β_1, β_2, η_1, η_2 in the range of 20%-180% of their default values. See Figure 21 for the plot of the SSIM values of the corresponding results on a synthesized image. It can be seen that the decomposition results are stable when the parameters vary in $[0.6, 1.4]$ of their default values. In comparison, the sensitivity of the performance of the proposed method is relatively higher for the parameters α_1, β_2, η_2 than for other parameters.



(a) SSIM of the cartoon component (b) SSIM of the texture component

Figure 21. *SSIM values of the cartoon and texture layers generated using the parameters α_1, α_2, β_1, β_2, η_1 and η_2 varying from 20% to 180% of their default values.*

To visualize the impact caused by different parameter settings, some results obtained using different parameter settings are shown in Figure 22-24. Figure 22 shows the results on “Friends” with varying α_1 and β_1. It can be seen that the patterns on the horse are clear in the cartoon component when $\alpha_1 = 0.05$, and disappears gradually as α_1 increases. When

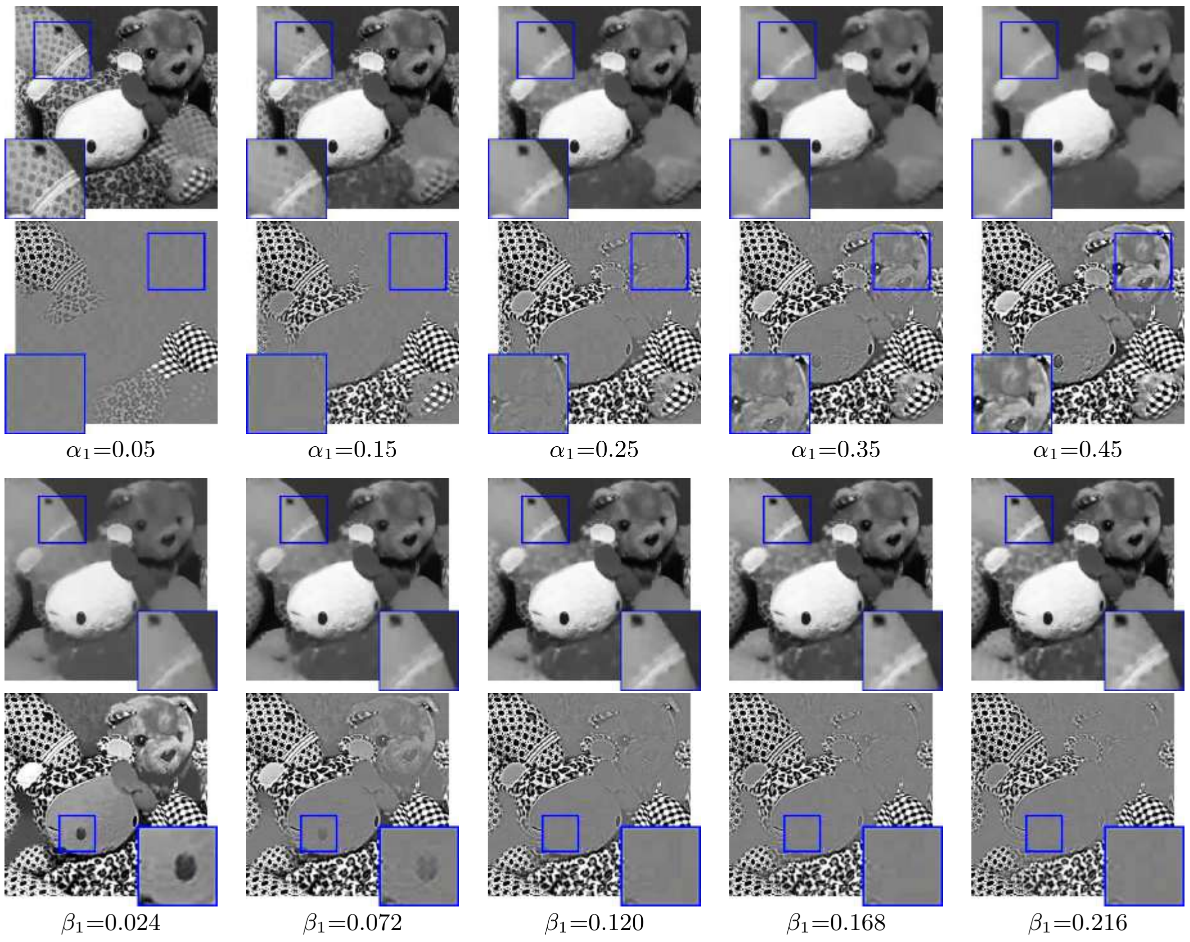


Figure 22. *The decomposition results of the proposed method on 'Friends' with varying α_1 and β_1.*

$\alpha_1 = 0.45$, the patterns are entirely removed, and even the white edge gets a little blurred. As for the texture components, the Teddy becomes clear as α_1 increases. On the contrary, when β_1 grows from 0.024 to 0.216, more and more patterns/details are carried from the texture components to the cartoon components.

The results about α_2 and β_2 are shown in Figure 23. As α_2 increases, more and more details are removed from the cartoon components while some week textures becomes stronger in the texture components. On the other hand, when β_2 grows from 0.006 to 0.054, it can also be observed that some undesired texture in the cartoon components become more and more clear while some structures from the texture components gradually get removed.

As for η_1 and η_2, the proposed method performs more stable when the parameters varies. The results are shown in Figure 24, where the edge of the roof keeps clear in the cartoon components and is almost invisible in the texture components. However, slight differences can still be observed. In Figure 24, the figure on the roof is not clear in the texture component and remains in the cartoon component when $\eta_1 = 4$, but it becomes clear in the texture component and removed from the cartoon component when $\eta_1 = 36$. Similar phenomena can

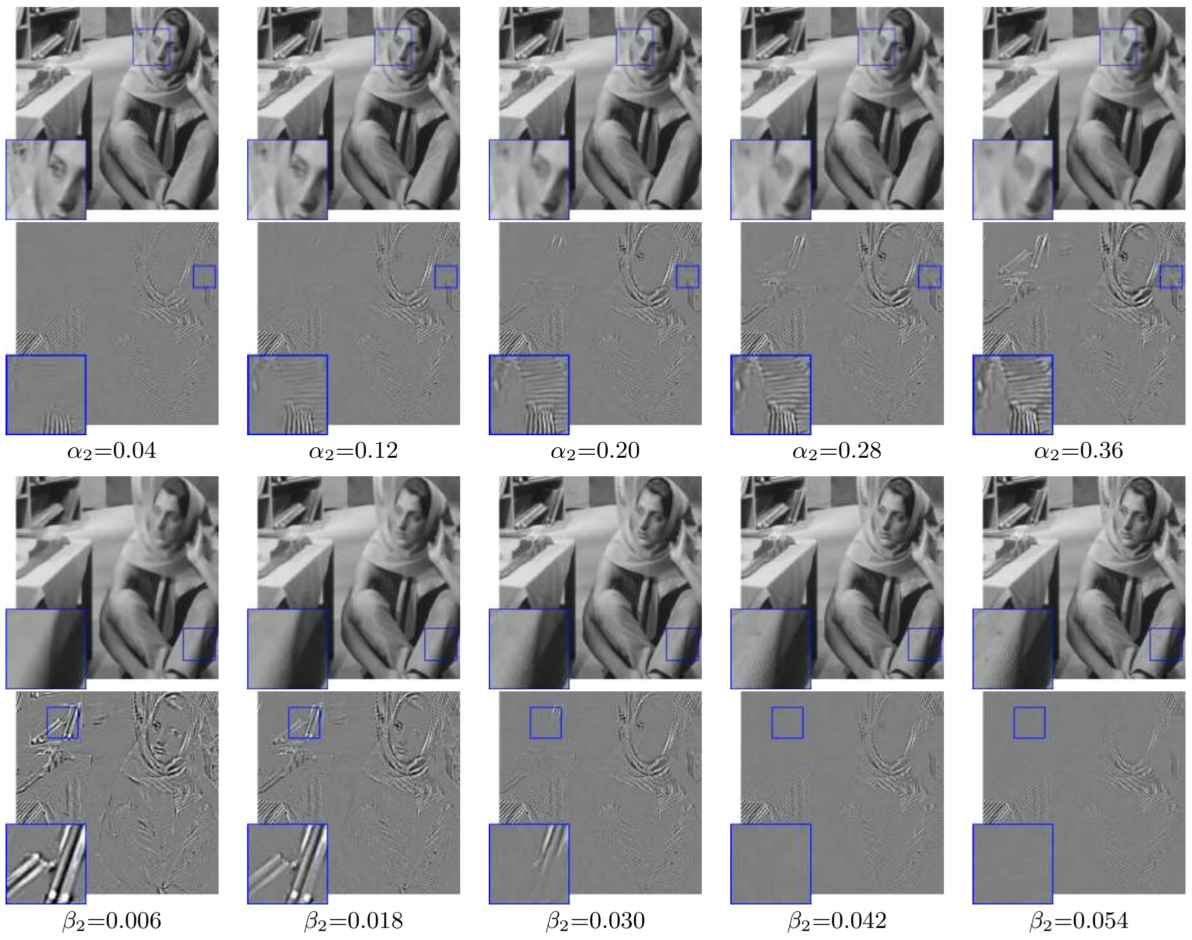


Figure 23. *The decomposition results of the proposed method on 'Barbara' with varying α_2 and β_2.*

also be seen in Figure 24 for varying η_2.

4.5. Computational Cost. To evaluate the efficiency of the proposed method, we compare its running time with other compared methods when processing images of size 256×256 and 512×512 on the same computational environment: Intel i7-6700 CPU and RTX TITAN GPU. The results are listed in Table 3. From Table 3, it can be seen that the PDE-based approach of Ng *et al.* [33] is the fastest and much faster than the patch-matching-based methods including Ma *et al.* [30] and ours. The time cost of our approach is around 1.5 times as that of Ma *et al.* [30] which is still acceptable. Compared to Papyan *et al.* [37], our approach is much faster.

Table 3
The running time (seconds) of the proposed and compared methods on different size of images.

Size	Ng [33]	Ono [34]	Ma [30]	Papyan [37]	Gu [24]	Sur [44]	Ours
256x256	2.0	12.0	84.0	393.9	63.2	5.4	126.7
512x512	9.1	90.2	293.0	1333.7	308.5	17.7	453.7

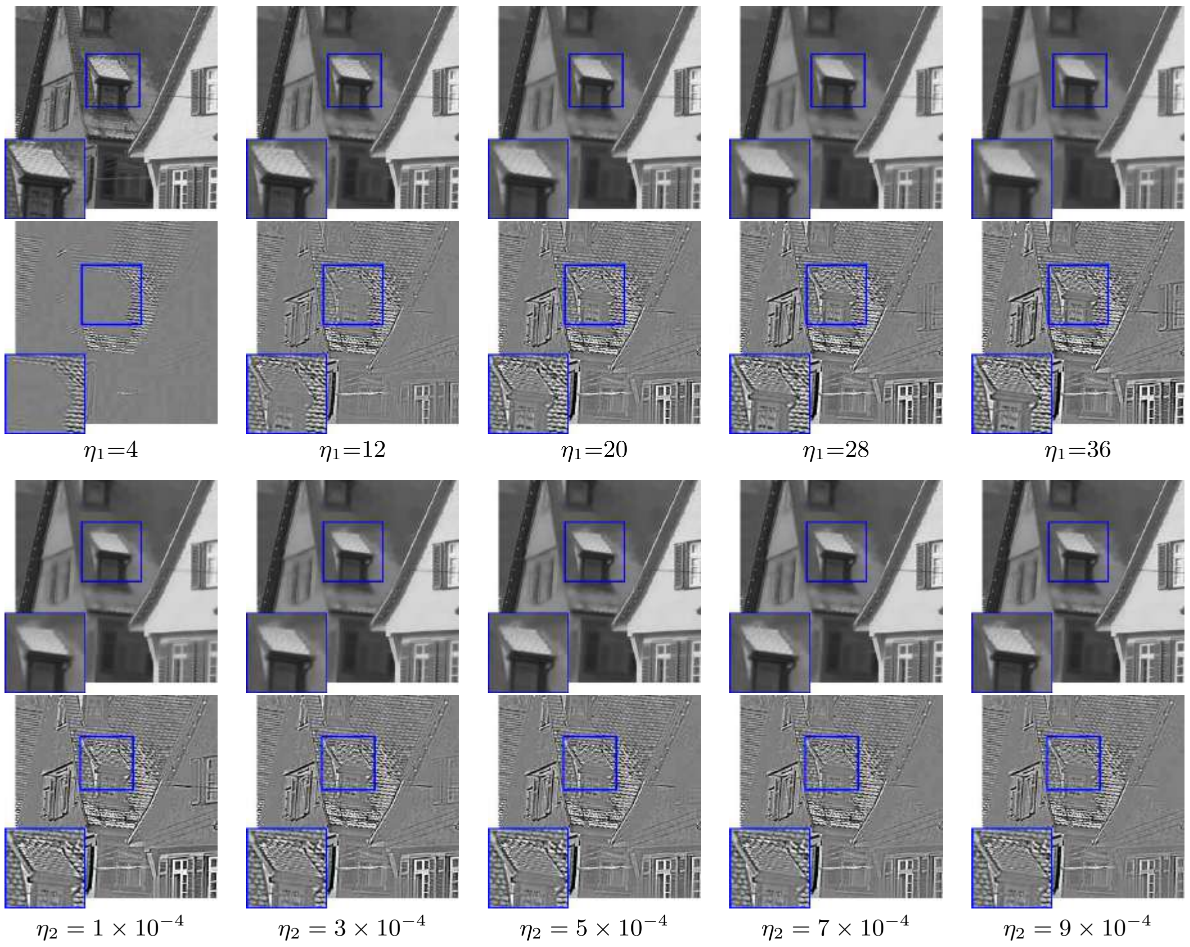

η_1=4
η_1=12
η_1=20
η_1=28
η_1=36
$\eta_2 = 1 \times 10^{-4}$
$\eta_2 = 3 \times 10^{-4}$
$\eta_2 = 5 \times 10^{-4}$
$\eta_2 = 7 \times 10^{-4}$
$\eta_2 = 9 \times 10^{-4}$

Figure 24. *The decomposition results of the proposed method on 'House' with varying η_1 and η_2.*

5. Conclusion.

In this paper, we showed that the patch recurrence for cartoon and texture have different orientation properties: anisotropy (cartoon) vs. isotropy (texture). Based on such a new observation, we proposed a new patch recurrence prior for distinguishing the cartoon layer and texture layer, and developed an approach to exploit such a prior for cartoon-texture decomposition. The experiments showed the advantage of the proposed method over existing state-of-the-art methods, in terms of both quantitative measurement and visual quality. In future, we would like to investigate the application of the proposed prior in other image processing problems.

REFERENCES

[1] J.-F. Aujol, G. Aubert, L. Blanc-Féraud, and A. Chambolle, *Image decomposition into a bounded variation component and an oscillating component*, J. Math. Imaging Vis., 22 (2005), pp. 71–88.
[2] J.-F. Aujol and A. Chambolle, *Dual norms and image decomposition models*, Int. J. Comput. Vis., 63 (2005), pp. 85–104.
[3] J.-F. Aujol, G. Gilboa, T. Chan, and S. Osher, *Structure-texture image decomposition—modeling,*

algorithms, and parameter selection, Int. J. Comput. Vis., 67 (2006), pp. 111–136.

[4] M. Bertalmio, L. Vese, G. Sapiro, and S. Osher, *Simultaneous structure and texture image inpainting*, IEEE Trans. Image Proc., 12 (2003), pp. 882–889.

[5] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, et al., *Distributed optimization and statistical learning via the alternating direction method of multipliers*, Found. Trends Mach. Learn., 3 (2011), pp. 1–122.

[6] A. Buades, B. Coll, and J.-M. Morel, *A non-local algorithm for image denoising*, in Proc. IEEE Conf. Comput. Vis. Pattern Recognition, 2005.

[7] A. Buades, T. M. Le, J.-M. Morel, and L. A. Vese, *Fast cartoon+ texture image filters*, IEEE Trans. Image Proc., 19 (2010), pp. 1978–1986.

[8] A. Buades and J. L. Lisani, *Directional filters for color cartoon+ texture image and video decomposition*, J. Math. Imaging Vis., 55 (2016), pp. 125–135.

[9] J.-F. Cai, B. Dong, and Z. Shen, *Image restoration: a wavelet frame based model for piecewise smooth functions and beyond*, Appl. Comput. Harmonic Anal., 41 (2016), pp. 94–138.

[10] F. Calderero and V. Caselles, *Recovering relative depth from low-level features without explicit t-junction detection and interpretation*, Int. J. Comput. Vis., 104 (2013), pp. 38–68.

[11] K. Cao, E. Liu, and A. K. Jain, *Segmentation and enhancement of latent fingerprints: A coarse to fine ridgestructure dictionary*, IEEE Trans. Pattern Anal. Mach. Intell., 36 (2014), pp. 1847–1859.

[12] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, , and A. Vedaldi, *Describing textures in the wild*, in Proc. IEEE Conf. Comput. Vis. Pattern Recognition, 2014.

[13] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, *Image denoising with block-matching and 3d filtering*, in Image Process: Algorithms Syst., Neural Networks, Mach. Learning, vol. 6064, International Society for Optics and Photonics, 2006, p. 606414.

[14] I. Daubechies, B. Han, A. Ron, and Z. Shen, *Framelets: Mra-based constructions of wavelet frames*, Appl. Comput. Harmonic Anal., 14 (2003), pp. 1–46.

[15] V. Duval, J.-F. Aujol, and L. A. Vese, *Mathematical modeling of textures: Application to color image decomposition with a projected gradient algorithm*, J. Math. Imaging Vis., 37 (2010), pp. 232–248.

[16] M. J. Fadili, J.-L. Starck, J. Bobin, and Y. Moudden, *Image decomposition and separation using sparse representations: an overview*, Proc. The IEEE, 98 (2010), pp. 983–994.

[17] Y.-R. Fan, T.-Z. Huang, T.-H. Ma, and X.-L. Zhao, *Cartoon–texture image decomposition via non-convex low-rank texture regularization*, J. Franklin Inst., 354 (2017), pp. 3170–3187.

[18] I. N. Figueiredo, S. Kumar, C. M. Oliveira, J. D. Ramos, and B. Engquist, *Automated lesion detectors in retinal fundus images*, Comput. Biol. Med., 66 (2015), pp. 47–65.

[19] M. Fritz, E. Hayman, B. Caputo, and J.-O. Eklundh, *The kth-tips database*, 2004.

[20] G. Gilboa, N. Sochen, and Y. Y. Zeevi, *Variational denoising of partly textured images by spatially varying constraints*, IEEE Trans. Image Proc., 15 (2006), pp. 2281–2289.

[21] J. Gilles, *Multiscale texture separation*, Multiscale Model. Simul., 10 (2012), pp. 1409–1427.

[22] J. Gilles and Y. Meyer, *Properties of $bv - g$ structures + textures decomposition models. application to road detection in satellite images*, IEEE Trans. Image Proc., 19 (2010), pp. 2793–2800.

[23] T. Goldstein and S. Osher, *The split bregman method for l1-regularized problems*, SIAM J. Imaging Sci., 2 (2009), pp. 323–343.

[24] S. Gu, D. Meng, W. Zuo, and L. Zhang, *Joint convolutional analysis and synthesis sparse representation for single image layer separation*, in Proc. IEEE Int. Conf. Comput. Vis., IEEE, 2017, pp. 1717–1725.

[25] S. Gu, L. Zhang, W. Zuo, and X. Feng, *Weighted nuclear norm minimization with application to image denoising*, in Proc. IEEE Conf. Comput. Vis. Pattern Recognition, 2014.

[26] Y. Han, C. Xu, G. Baciu, M. Li, and M. R. Islam, *Cartoon and texture decomposition-based color transfer for fabric images*, IEEE Trans. Multimedia, 19 (2017), pp. 80–92.

[27] X. Hu, W. Xia, S. Peng, and W.-L. Hwang, *Multiple component predictive coding framework of still images*, in Proc. IEEE Int. Conf. Multimedia and Expo, IEEE, 2011, pp. 1–6.

[28] G. Kylberg, *The kylberg texture dataset v. 1.0*, tech. report, http://www.cb.uu.se/~gustaf/texture/.

[29] Z. Liang, J. Xu, D. Zhang, Z. Cao, and L. Zhang, *A hybrid l1-l0 layer decomposition model for tone mapping*, in Proc. IEEE Conf. Comput. Vis. Pattern Recognition, 2018, pp. 4758–4766.

[30] T.-H. Ma, T.-Z. Huang, and X.-L. Zhao, *Group-based image decomposition using 3-d cartoon and*

texture priors, Inform. Sci., 328 (2016), pp. 510–527.

[31] P. Maurel, J.-F. Aujol, and G. Peyré, *Locally parallel texture modeling*, SIAM J. Imaging Sci., 4 (2011), pp. 413–447.

[32] Y. Meyer, *Oscillating patterns in image processing and nonlinear evolution equations: the fifteenth Dean Jacqueline B. Lewis memorial lectures*, vol. 22, American Mathematical Soc., 2001.

[33] M. K. Ng, X. Yuan, and W. Zhang, *Coupled variational image decomposition and restoration model for blurred cartoon-plus-texture images with missing pixels*, IEEE Trans. Image Proc., 22 (2013), pp. 2233–2246.

[34] S. Ono, T. Miyata, and I. Yamada, *Cartoon-texture image decomposition using blockwise low-rank texture characterization*, IEEE Trans. Image Proc., 23 (2014), pp. 1128–1142.

[35] S. Ono, T. Miyata, I. Yamada, and K. Yamaoka, *Image recovery by decomposition with component-wise regularization*, IEICE Trans. Fundam. Electron., Commun. Comput. Sci., 95 (2012), pp. 2470–2478.

[36] S. Osher, A. Solé, and L. Vese, *Image decomposition and restoration using total variation minimization and the h^{-1} norm*, Multiscale Model. Simul., 1 (2003), pp. 349–370.

[37] V. Papyan, Y. Romano, J. Sulam, and M. Elad, *Convolutional dictionary learning via local processing*, in Proc. IEEE Int. Conf. Comput. Vis., 2017, pp. 5296–5304.

[38] Y. Quan, H. Ji, and Z. Shen, *Data-driven multi-scale non-local wavelet frame construction and image recovery*, J. Scientific Comput., 63 (2015), pp. 307–329.

[39] H. Schaeffer and S. Osher, *A low patch-rank interpretation of texture*, SIAM J. Imaging Sci., 6 (2013), pp. 226–262.

[40] J.-L. Starck, M. Elad, and D. L. Donoho, *Image decomposition: Separation of texture from piecewise smooth content*, in Wavelets: Appli. Signal Image Process., vol. 5207, International Society for Optics and Photonics, 2003, pp. 571–583.

[41] J.-L. Starck, M. Elad, and D. L. Donoho, *Image decomposition via the combination of sparse representations and a variational approach*, IEEE Trans. Image Proc., 14 (2005), pp. 1570–1582.

[42] J.-L. Starck, Y. Moudden, J. Bobin, M. Elad, and D. Donoho, *Morphological component analysis*, in Wavelets XI, vol. 5914, International Society for Optics and Photonics, 2005, p. 59140Q.

[43] Y. Sun, S. Schaefer, and W. Wang, *Image structure retrieval via l_0 minimization*, IEEE Trans. Vis. Comput. Graphics, 24 (2018), pp. 2129–2139.

[44] F. Sur, *A non-local dual-domain approach to cartoon and texture decomposition*, IEEE Trans. Image Proc., 28 (2019), pp. 1882–1894.

[45] E. Tadmor, S. Nezzar, and L. Vese, *A multiscale image representation using hierarchical (bv, l 2) decompositions*, Multiscale Model. Simul., 2 (2004), pp. 554–579.

[46] K. Valkealahti and E. Oja, *Reduced multidimensional co-occurrence histograms in texture classification*, IEEE Trans. Pattern Anal. Mach. Intell., 20 (1998), pp. 90–94.

[47] L. A. Vese and S. J. Osher, *Modeling textures with total variation minimization and oscillating patterns in image processing*, J. Sci. Comput., 19 (2003), pp. 553–572.

[48] A. Wedel, T. Pock, C. Zach, H. Bischof, and D. Cremers, *An improved algorithm for tv-$l\ell_1$ optical flow*, in Statist. Geom. Approaches Visual Motion Anal., Springer, 2009, pp. 23–45.

[49] L. Xu, Q. Yan, Y. Xia, and J. Jia, *Structure extraction from texture via relative total variation*, ACM Trans. Graphics, 31 (2012), p. 139.

[50] I. Yanovsky and A. B. Davis, *Separation of a cirrus layer and broken cumulus clouds in multispectral images*, IEEE Trans. Geosci. Remote Sens., 53 (2015), pp. 2275–2285.

[51] W. Yin, D. Goldfarb, and S. Osher, *Image cartoon-texture decomposition and feature selection using the total variation regularized l1 functional*, in Variational, Geometric, and Level Set Methods in Computer Vision, Springer, 2005, pp. 73–84.

[52] W. Yin, D. Goldfarb, and S. Osher, *Total variation based image cartoon-texture decomposition*, tech. report, COLUMBIA UNIV NEW YORK DEPT OF INDUSTRIAL ENGINEERING AND OPERATIONS RESEARCH, 2005.

[53] H. Zhang and V. M. Patel, *Convolutional sparse coding-based image decomposition.*, in Proc. British Mach. Vis. Conf., 2016.