TSINGHUA SCIENCE AND TECHNOLOGY
ISSN 1007-0214 15/21 pp1138–1151
DOI: 10.26599/TST.2023.9010068
Volume 29, Number 4, August 2024

A Parameter Adaptive Method for Image Smoothing

Suwei Wang, Xiang Ma, and Xuemei Li*

Abstract: Edge is the key information in the process of image smoothing. Some edges, especially the weak edges, are difficult to maintain, which result in the local area being over-smoothed. For the protection of weak edges, we propose an image smoothing algorithm based on global sparse structure and parameter adaptation. The algorithm decomposes the image into high frequency and low frequency part based on global sparse structure. The low frequency part contains less texture information which is relatively easy to smoothen. The high frequency part is more sensitive to edge information so it is more suitable for the selection of smoothing parameters. To reduce the computational complexity and improve the effect, we propose a bicubic polynomial fitting method to fit all the sample values into a surface. Finally, we use Alternating Direction Method of Multipliers (ADMM) to unify the whole algorithm and obtain the smoothed results by iterative optimization. Compared with traditional methods and deep learning methods, as well as the application tasks of edge extraction, image abstraction, pseudo-boundary removal, and image enhancement, it shows that our algorithm can preserve the local weak edge of the image more effectively, and the visual effect of smoothed results is better.

Key words: image smoothing; parameter adaptation; bicubic interpolation; polynomial fitting

1 Introduction

Image smoothing is a crucial and extensively applied technique in the field of image processing. It aims to reduce noise and texture in images while preserving the salient edges during the smoothing process. Image smoothing finds widespread applications in tasks such as image segmentation, edge extraction, image enhancement, image decomposition, and artifact removal. Due to the inherent characteristics of natural images, which consist of texture and structure components, texture often exhibits intricate, irregular, and anisotropic properties in contrast to the image structure. Consequently, the challenging task of image smoothing arises in achieving an optimal balance between preserving the structural integrity and effectively eliminating the texture. This delicate trade-off is a fundamental problem encountered in the field of image smoothing. Presently, image smoothing algorithms can be broadly classified into three main categories: local information based filters, global optimization based frameworks, and deep learning based approaches. Each category offers distinctive methodologies to address the intricate challenge of texture removal while maintaining the coherence of underlying structures. The choice of a suitable image smoothing algorithm depends on specific application requirements and the desired compromise between texture suppression and the fidelity of structural details.

Bilateral Filtering (BLF)[1] is a commonly used image filtering technique that takes into account the spatial distance between pixels and the grayscale

• Suwei Wang is with School of Computer Science and Technology, Shandong University, Jinan 250101, China. E-mail: wangsuwei1@163.com.
• Xiang Ma and Xuemei Li are with School of Software, Shandong University, Jinan 250101, China. E-mail: xiangma@mail.sdu.edu.cn; xmli@sdu.edu.cn.
* To whom correspondence should be addressed.
Manuscript received: 2023-05-29 ; revised: 2023-06-26; accepted: 2023-07-02

© The author(s) 2024. The articles published in this open access journal are distributed under the terms of the Creative Commons Attribution 4.0 International License (http://creativecommons.org/licenses/by/4.0/).

differences between pixels. It achieves smoothing by computing the weighted average of pixels, where the weights are determined by the spatial distance and similarity of pixel values. Since its introduction, the algorithm has undergone various improvements[2–6], mostly involving modifications to its Gaussian kernel. Among these improvements, Bilateral Texture Filtering (BTF)[7] utilizes local texture information to adjust the weights, resulting in better preservation of image edges during the smoothing process. However, it tends to produce less regular results in flat regions, leading to suboptimal visual effects. Tree filtering[8, 9] is capable of selectively smoothing different details in an image based on its hierarchical structure, effectively mitigating the presence of ringing artifacts. However, erroneous pixel classification can lead to the generation of spurious boundaries. Mean filtering achieves smoothing by replacing the value of a central pixel with the average value of its neighboring pixels, but it lacks the ability to adequately preserve fine image details. Other locally-informed filtering techniques encompass anisotropic filters[10, 11], extremum smoothing algorithms[8, 12], among others. Most filters are relatively simple but rely on local information during the image smoothing process, which can potentially result in the occurrence of ringing artifacts.

Weighted Least Squares (WLS)[13] is a globally optimized and relatively robust image smoothing algorithm. It achieves better smoothing performance by considering both the spatial relationships between pixels and the similarity of their values. It is especially suitable for progressive coarsening and edge-preserving multi-scale detail extraction of images. Compared with local filtering methods, it is more flexible than optimization frameworks such as WLS with edge preservation as a regular term. Inspired by the Gaussian difference feature extraction algorithm Difference of Gaussian (DoG)[14, 15], Relativity of Gaussian (RoG)[16, 17] is an algorithm based on local regularization, by describing the difference between Gaussian filters of different sizes relationship to smooth image. Although it can effectively remove image details and textures, it has problems such as difficulty in parameter adjustment and loss of local information. Total Variation (TV)[18] is a very well-known algorithm for smoothing images and retaining edge information. The regularization optimization of the smoothing result is performed based on the L_1 norm of the image's first-order gradients. Image smoothing is achieved by minimizing the total variation of the image. It has good results in image denoising, image inpainting, and other tasks. However, this algorithm cannot perform reasonable smoothing on images with rich texture information. In other words, it is difficult for TV to distinguish which texture information needs to be removed and which boundary information needs to be protected. The improvement of the Relative Total Variation (RTV)[19] algorithm is mainly reflected in the constraints on the results. Through the introduction of the relative norm, it can better balance the smoothness of the image and the ability to preserve details, which is better than TV to a certain extent. The L_0[20] algorithm is an improvement on TV, a smoothing method that approximates prominent image structures by controlling the number of non-zero gradients and sparse structures. Its visual quality is significantly better than TV, but it cannot reasonably smooth images with rich texture information. L_0p[21, 22] uses Alternating Direction Method of Multipliers (ADMM) to improve L_0, making the setting of smoothing parameters more flexible. However, the result of L_0p has obvious false boundaries, which will affect the subsequent image processing. Algorithm[23] solves the L_0p problem by limiting the gradient of smoothed images, which can improve the visual quality of the results, but only works for a small number of images. Liu et al.[24] proposed a generalized framework to achieve different image smoothing properties by introducing a truncated Huber penalty function. It also produces smooth behavior that is rarely achieved by previous methods, leading to excellent performance.

With the development of deep learning, data-driven image smoothing algorithms have been gradually proposed. However, since there is no ground truth for image smoothing, conventional supervised learning and semi-supervised learning methods cannot be well utilized. The algorithm[25–27] attempts to use a unified Convolutional Neural Network (CNN) framework to emulate the previous approach[19] for image smoothing. However, it does not break away from the limitations of the original algorithm. The algorithm Deep Variational Priors (DVP)[28, 29] optimizes the image smoothing process by training parameters, and the effect is improved to a certain extent, but the generalization of DVP is poor. Li et al.[30] and Fan et al.[31] proposed a scale-aware image smoothing method

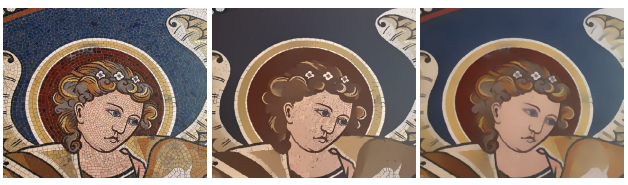
based on deep learning, which isbuilt upon an attention-guided downscaling-upscaling mechanism. The downscaling mechanism is used to predict large-scale salient structures, while the upscaling mechanism is employed to identify and infer unimportant small-scale details from the salient structures. Deep Flexible Structure Preserving method (DFSP)[32, 33] follows the divide-and-conquer principle to decouple the original problem into two specific functions, namely control guidance prediction and image smoothing based on prediction guidance. A deep network is developed from the perspectives of control flexibility, smooth structure preservation, and efficiency.

Combining local filters and a global optimization framework is one way to try to alleviate the problem, but there are still some limitations. The flexibility of the optimization framework method makes it suitable for various image smoothing tasks, but the selection of global regularizers is difficult to preserve local information at the same time. In addition, deep learning based methods still have limitations in terms of generalization. Although the combination of local filters and global optimization framework alleviates these problems to some extent, it still has the problem of not better preserving weak edges of images.

Through long-term research, we have summarized three core problems faced by the current image smoothing field and proposed corresponding improvement schemes:

(1) The smoothing effect of rich texture image is expected to be enhanced. In general, as the texture in the image increases, the smoothing effect of the algorithm will become poor, as shown in Fig. 1a. This is because the more textures there are, the harder it is for the algorithm to distinguish between edges and textures. Therefore, we set the global sparse regularization item to remove part of the texture in the image in advance, which improves the effect of the smoothing algorithm.

(2) The inconsistency of the smoothing results in different parts needs to improve. In general, the illumination, contrast, and other elements of the image are different in different regions of the image, so different smoothing parameter values should be used in different regions. We adopt a block-wise strategy to select parameters to determine the size of parameters in different regions in an adaptive manner. At the same time, in order to solve the problem of pseudo-boundary, we propose to construct the parameter fitting surface, so as to allow the parameters of the whole region to change continuously.



(a) An image with abundant texture information (b) Smoothed result by L_0 gradient minimization algorithm[20] (c) Smoothed result by our algorithm

Fig. 1 Examples of image smoothing.

(3) An evaluating method of the smoothing effect is necessary. Due to the lack of ground truth in image smoothing, numerical indicators cannot be used to quantitatively analyze the smoothing results. It is too subjective to compare different algorithms only based on vision, so a quantitative index is urgently needed as an evaluation standard. The difficulty of image smoothing is to distinguish textures and edges, and the human eye has a good ability to distinguish them. Therefore, we can evaluate the smoothing effect by comparing the edges extracted from the smoothed result with those manually extracted from the original image. In addition, image smoothing will change the distribution of image gradient values, and the smaller the gradient value, the smoother the image. Therefore, we can compare the gradient values of different algorithm results to evaluate the smoothing effect. Based on the above, we propose three indicators, namely edge integrity rate, texture removal rate, and gradient value distribution, to quantitatively evaluate smoothing results from the perspective of edges and gradients.

In summary, we combine the idea of local filters with a global optimization framework and propose an image smoothing algorithm based on global sparse structure and bicubic interpolation fitting. First, a global sparse decomposition is used to pre-remove part of the texture to improve smoothing performance. Then, the parameter fitting surface is obtained by block selection and bicubic interpolation to ensure the locality and continuity of the parameters. Finally, the L_1 smoothing norm is used to limit the gradient change of the image to achieve image smoothing. The flowchart is shown in Fig. 2.

The main structure of this paper is as follows. Section 2 mainly introduces the model framework of this algorithm, including global sparse structure, and selection of base parameters and parameter matrix.

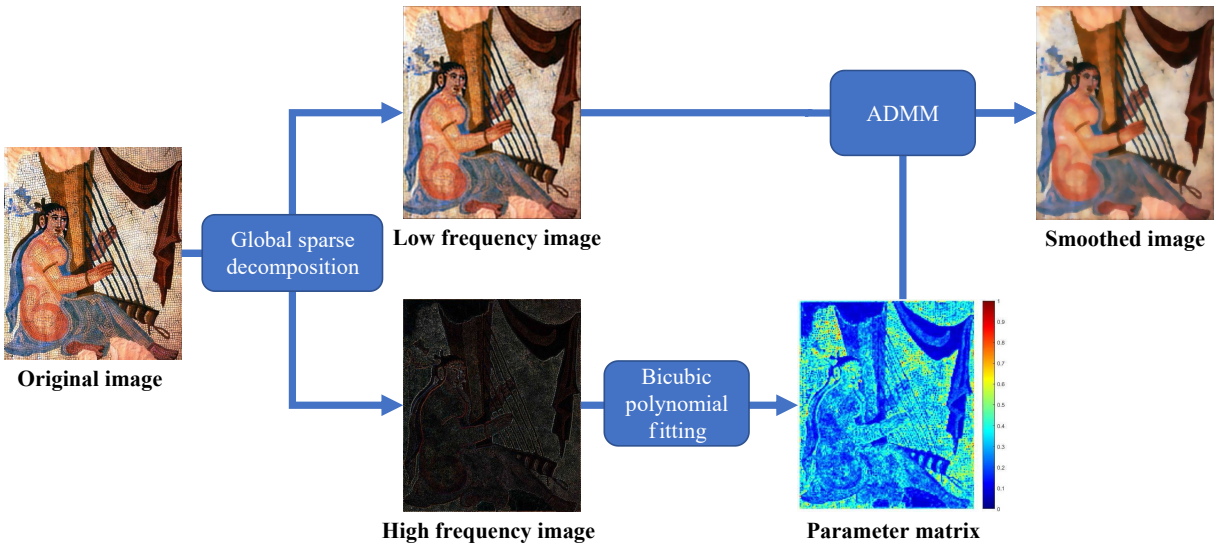


Fig. 2 Framework of our method.

Section 3 describes the solution of this algorithm based on ADMM in detail. Section 4 compares the visual effect of our algorithm with other algorithms and the effect of selecting different smoothing parameters. Section 5 introduces the application of proposed algorithm in edge extraction, image abstraction, pseudo-boundary removal, and image enhancement. Finally, Section 6 summarizes the paper.

2 Problem Formulation

Image smoothing, which we can express as a smooth image affected by complex useless texture, can be expressed as

$$y = x + n \tag{1}$$

where n is redundant texture information and y is the image we get directly. The final goal of image smoothing is to remove texture information n from the known image y as much as possible and obtain the smoothed image x.

In order to get more accurate results, we can improve the quality of smoothing results by giving a prior term. Many smoothing methods use variational method to set prior terms. TV model is a representative optimization algorithm. The smoothed result graph of the TV algorithm is obtained by following energy functions:

$$\hat{x}_{\mathrm{TV}} = \arg\min_x \frac{1}{2}\|y - x\|_2^2 + \lambda R_{\mathrm{TV}}(x) \tag{2}$$

where $\|y - x\|_2^2$ is the fidelity term, which is used to describe the matching degree between the existing image and the target image. The smaller this value, the closer the result is to the target image. $R(x)$ is the regular term, the form of which depends on the specific given image prior information, such as $R_{\mathrm{TV}}(x) = \|\nabla x\|_2^2$. λ is a parameter of the regular term to control the effect of $R(x)$ on the energy function. It is worth noting that there is a need to satisfy $\lambda > 0$, otherwise $R(x)$ will not give the right prior guidance for the energy function. For example, when $\lambda = -1$, the latter term $\lambda R_{\mathrm{TV}}(x)$ in Eq. (2) will be $-\|\nabla x\|_2^2$, which makes us expect a larger gradient of x when solving the minimum value, but this is actually contrary to our goal of removing texture information in x.

Many algorithms[18, 34] are based on variational models for image smoothing, but they inevitably have the following problems. Firstly, for parameter, the algorithms[20, 35] set it as a constant, which means that all regions of the image are given the same restrictions. In this way, local edges will be smoothed because the fluctuation is less than the fluctuation of texture information that needs to be removed. But in fact, these local edges are what we want to keep in the result. In addition, the original image y itself contains too much unnecessary texture information which will directly affect the smoothing process and the setting of smoothing parameters, so if y contains less texture information, we can get better result.

In this paper, we solve the above problems by combining global sparse regularization decomposition, L_1 norm prior, and local continuity selection of smoothing parameters to achieve global smoothing of images based on the local information. The image decomposition algorithm based on global sparse structure will preliminarily screen out the redundant

texture information in y, so as to reduce its impact on smoothing process. Regular prior L_1 norm is used to obtain smoothed result by making gradient sparse. In fact, during the experiment, we tried to use other norms (L_0 norm, L_2 norm, etc.), but the effect was not as good as that of L_1 norm. The reason is that L_1 norm can prevent image color distortion and get sparse gradient at the same time. Moreover, we use the continuous local adaptive parameter matrix instead of the constant parameter to avoid over-smoothing of the local edge. The framework of our algorithm is shown in Fig. 2.

2.1 Global sparse structure

In the fields of image super-resolution and image reconstruction, in order to avoid blurring the boundary or restoring the lost details of the image, it is usually decomposed into low frequency components and high frequency components in advance[36, 37], and the results are supplemented based on the structure information. Inspired by this, we can use a similar method to decompose the image into low frequency part and high frequency texture part and use the low frequency image instead of the original one for smoothing, so as to reduce the influence of the complex texture of the image itself. The energy function of image decomposition is described as

$$y = f_L \otimes y_L + y_H \tag{3}$$

where y_L and y_H are the low and high frequency feature graph of input image y, respectively. f_L is a low-pass filter of size 3×3, and $\otimes$ is the convolution operator. This operation ensures that the smooth component contains low frequencies, thereby modeling high-level information in the image. We can model image sparse structure decomposition by setting two prior terms:

$$\begin{aligned} &R_{\text{str}}(y) = \|y_H\|_1 + \frac{\kappa}{2}\sum_{d=1}^{4}\|\nabla_d \otimes y_H\|_2^2, \\ &\text{s.t.,} \quad y = f_L \otimes y_L + y_H \end{aligned} \tag{4}$$

where κ is used to control the smoothness level of low frequency part. The larger value of κ, the more image information y_H contains. ∇_d means calculating the gradient in d direction, and $d = \{1 = \text{horizontal}, 2 = \text{vertical}, 3 = 45°, 4 = 135°\}$.

It is well known that L_p norm can promote sparsity when $p \leqslant 1$. Here we use $\|y_H\|_1$ to force the high frequency component to be the sparse structure component under L_1 norm (L_1 norm is used to ensure the convexity of the structure), making y_H contain only high frequency texture information. This will cause y only lose the part of high frequency in the process of decomposition and keep all the low frequency information. Then we use y_L instead of y in Eq. (2) to weaken the effect of texture on smoothing process.

There is the comparison between image gradient and absolute value of y_H in $\kappa = 1$. The structure and texture information are labeled in different colors according to the values. It is obvious that y_H is sparser than gradient. Figure 3 further analyses this characteristic. The numerical distributions of gradient and y_H with different κ values are given, where κ is 0.1, 1, and 10. We can find that κ will affect the sparsity of residuals. The smaller the value of κ, the higher the density near zero. More importantly, the distribution characteristics of y_H are more consistent with the Laplace distribution than gradient, the Laplace distribution is concave and the peak value is non-zero. And as we all know, L_1 norm regularization can be regarded as negative logarithmic prior in accordance with Laplacian distribution. On logarithmic scale, this distribution is shown as a downward inclined line. Therefore, applying L_1 norm to the gradient of the image as a regular term, such as in TV algorithm[18], will lead to the loss of non-texture details.

2.2 Local selection of base parameters and parameter matrix

We use image decomposition result y_L to replace the original image for smoothing to reduce unnecessary texture. However, the original purpose of our works is

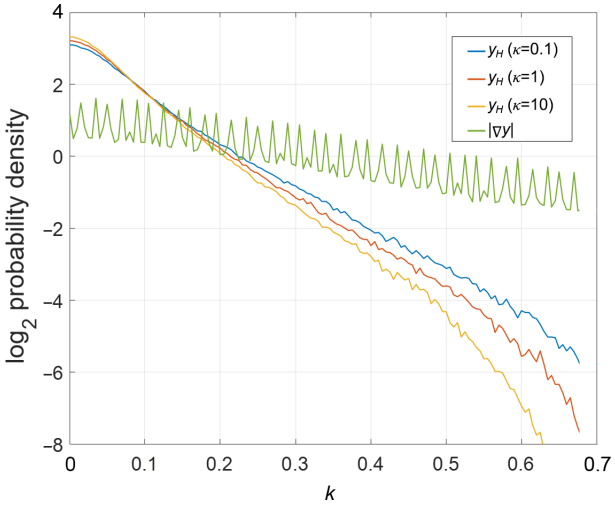


Fig. 3 Image gradient distribution and y_H distribution with different κ values.

to prevent local edges from over-smoothing, so we propose an adaptive selection method for smoothing parameters to improve our L_1 norm sparse operation on images. For the total-variational model, the texture information we need to remove can be expressed as

$$x - \hat{x}_{\mathrm{TV}} = -\lambda \mathrm{curv}(\hat{x}_{\mathrm{TV}}) \tag{5}$$

where $\mathrm{curv}(\hat{x}_{\mathrm{TV}})$ represents the curvature of all levels. TV algorithm gives a globally consistent constant parameter λ which can lead to over-smoothing. Our proposed parameter λ_{y_H} can smooth the information of different fluctuation levels for different local regions, thus protecting local edges.

2.2.1 Base parameters selection

In the following section, we will describe how we construct λ_{y_H}. Based on the above symbols, our model with new smoothing parameters can be described as

$$\begin{aligned} \hat{x}_{\mathrm{our}} &= \arg\min_x \frac{1}{2}\|y_L - x\|_2^2 + \lambda_{y_H} R_1(x) + \alpha R_{\mathrm{str}}(y), \\ R_1(x) &= \sum_d^4 \|\nabla_d x\|_1 \end{aligned} \tag{6}$$

where $R_{\mathrm{str}}(y)$ is Eq. (4), and y in Eq. (2) has been replaced by y_L. R_1 refers to the restriction of gradient sparsity of x in four different directions, and d is consistent with the value in $R_{\mathrm{str}}(y)$. In order to construct λ_{y_H}, we firstly segment the image to obtain the fluctuation degree of different image patches and adjust the local smoothing parameters λ_p according to the degrees. The solution formula of λ_p is as follows:

$$P_{i,j} = \frac{2}{1 + \dfrac{\mathrm{std}(y_H)}{\varepsilon + \mathrm{std}(\Omega_{i,j})}} \tag{7}$$

$$\lambda_p = P_{i,j}\lambda_G \tag{8}$$

where λ_G is the user-controlled global smoothing parameter. For ease of description, we use $P_{i,j}$ to denote the local variation of λ_G at patch p. $\Omega_{i,j}$ refers to patch, and std(·) refers to the standard deviation of the matrix. $\varepsilon = 0.001$ is a small value to prevent denominator from being zero. As shown in Fig. 4, when the patch Ω fluctuates more than y_H, $P_{i,j}$ will gradually approach 2. Conversely, when the fluctuation of Ω is less than y_H, $P_{i,j}$ will approach zero quickly. And when the fluctuation of Ω approaches y_H, it tends to be 1. Through Eqs. (7) and (8), the parameters of the image are selected preliminarily.

As described in Eq. (7), we choose y_H as the parameter selection object, not y or y_L. This is because

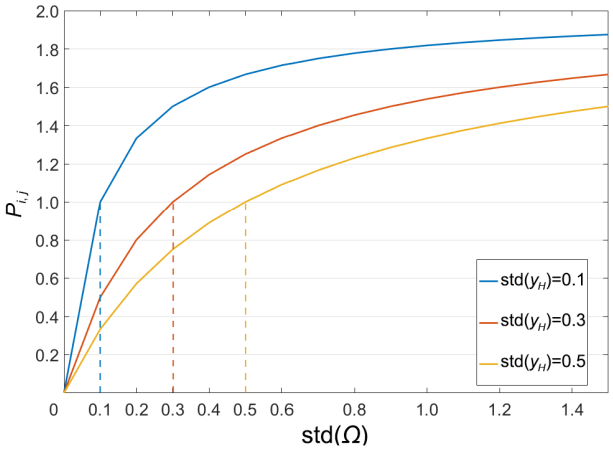


Fig. 4 $P_{i,j}$ **with different** std(y_H) **values.**

the parameters selected from high frequency part can only be set according to the information with large fluctuation of image, while the original image or low frequency part contains more low frequency information, which increases the parameter value of the flat part of the image, and the weak local boundary of the image is over-smoothed.

However, no matter what patch selection method we adopt, the local parameters selected by Eqs. (7) and (8) will lead to the obvious pseudo-boundary in the junction area of patches in the final result. If the constant parameter λ can be regarded as a flat, the result of patch selection is a step-like surface which leads approximately identical points at the junction of image patches to different values.

2.2.2 Parameter matrix construction

There are two ways to solve this problem. The first one is to set the step of patch to 1, which is to calculate Eq. (7) for each pixel, so the cost is extremely time-consuming. So, we propose another solution. Firstly, we still calculate the smoothing parameters of each patch by the old method and set step as the half of the patch size. Then we can get a parameter matrix with equal ratio reduction from the original graph. The parameter matrix can be regarded as a low-resolution image, and what we need is to enlarge the matrix to the same size as the original image by interpolation. Experiments show that this method can not only ensure the continuity of parameters but also reduce the operation time. We use bicubic interpolation to fit the image parameters. To enable the effective of bicubic interpolation, we propose a polynomial fitting method.

We define the target parameter surface as $F(x,y)$. P consists of $n \times n$ pixels $P_{i,j}$, $i,j = 1,2,\ldots,n$. $P_{i,j}$ is a

sample of $F(x,y)$. Without loss of generality, let the coordinates of $P_{i,j}$ in the oxy coordinate system be (i,j). The basic idea is to construct a bicubic polynomial fitting surface $P_{i,j}(u,v)$ on the unit region $[0,1]\times[0,1]$, where $P_{i,j}$, $P_{i+1,j}$, $P_{i+1,j+1}$, and $P_{i,j+1}$ are located. Compute upsampled points on $P_{i,j}(u,v)$. $P_{i,j}(u,v)$ is constructed as follows. As shown in Fig. 5, corresponding to each pixel $P_{i,j}$, $i,j=2,3,4,\ldots,n-1$, construct a quadratic polynomial surface patch $F_{i,j}(u,v)$, which is a fitting of nine pixels centered on $P_{i,j}$. The surface patch $P_{i,j}(u,v)$ on the unit region $[0,1]\times[0,1]$ corresponding to $P_{i,j}$, $P_{i+1,j}$, $P_{i+1,j+1}$, and $P_{i,j+1}$ is produced by the weighted average of $F_{i,j}(u,v)$, $F_{i+1,j}(u,v)$, $F_{i+1,j+1}(u,v)$, and $F_{i,j-1}(u,v)$.

Let us first discuss construction $F_{i,j}(u,v)$, $i,j=2,3,4,\ldots,n-1$. For the nine adjacent points $P_{i,j}$ centered on $P_{i+k,j+l}$, $k,l=-1,0,1$, as shown in Fig. 5. Since the four pixels of $P_{i+1,j}$, $P_{i-1,j}$, $P_{i,j+1}$, and $P_{i,j-1}$ are the nearest neighbors of $P_{i,j}$, they should play a larger role in constructing $F_{i,j}(u,v)$. Therefore, $F_{i,j}(u,v)$ interpolates the five pixels of $P_{i,j}$, $P_{i+1,j}$, $P_{i-1,j}$, $P_{i,j+1}$, and $P_{i,j-1}$ and approximates the four pixels of $P_{i-1,j-1}$, $P_{i-1,j+1}$, $P_{i+1,j+1}$, and $P_{i+1,j-1}$. Therefore, $F_{i,j}(u,v)$ can be written as

$$F_{i,j}(u,v)=a_{i,j}u^2+b_{i,j}uv+c_{i,j}v^2+d_{i,j}u+e_{i,j}v+P_{i,j} \tag{9}$$

where $u=x-i$ and $v=y-j$.

$$a_{i,j}=(P_{i+1,j}+P_{i-1,j})/2-P_{i,j} \tag{10}$$

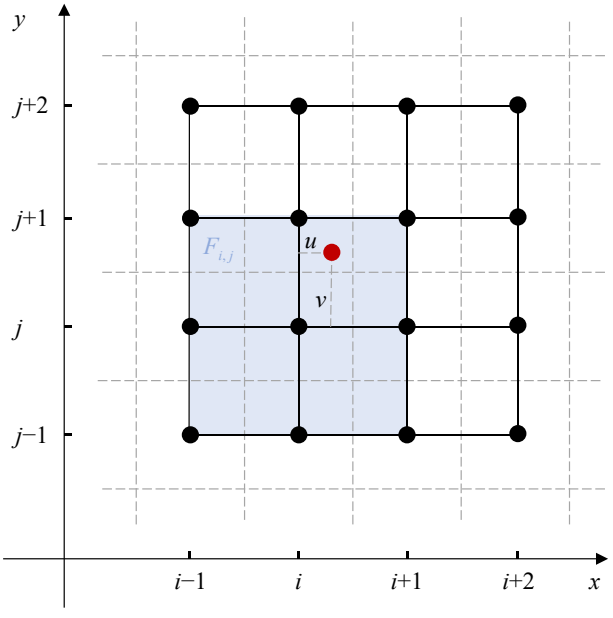


Fig. 5 Nine pixels in the $P_{i,j}$ neighborhood.

$$b_{i,j}=(P_{i+1,j+1}+P_{i-1,j-1}-P_{i-1,j+1}-P_{i+1,j-1})/4 \tag{11}$$

$$c_{i,j}=(P_{i,j+1}+P_{i,j-1})/2-P_{i,j} \tag{12}$$

$$d_{i,j}=(P_{i+1,j}-P_{i-1,j})/2 \tag{13}$$

$$e_{i,j}=(P_{i,j+1}-P_{i,j-1})/2 \tag{14}$$

After constructing $F_{i,j}(u,v)$, $i,j=2,3,4,\ldots,n-1$, the surface $P_{i,j}(u,v)$ on the small region $[0,1]\times[0,1]$ corresponding to $P_{i,j}$, $P_{i+1,j}$, $P_{i+1,j+1}$, and $P_{i,j+1}$ can be defined by the following formula:

$$\begin{aligned}P_{i,j}(u,v)=&F_{i,j}(u,v)w_1(u,v)+\\&F_{i+1,j}(u-1,v)w_2(u,v)+\\&F_{i+1,j+1}(u-1,v-1)w_3(u,v)+\\&F_{i,j+1}(u,v-1)w_4(u,v)\end{aligned} \tag{15}$$

$$w_1(u,v)=(1-u)(1-v) \tag{16}$$

$$w_2(u,v)=u(1-v) \tag{17}$$

$$w_3(u,v)=uv \tag{18}$$

$$w_4(u,v)=(1-u)v \tag{19}$$

Since $F_{i,j}(u,v)$, $F_{i+1,j}(u,v)$, $F_{i+1,j+1}(u,v)$, and $F_{i,j-1}(u,v)$ are quadratic polynomial surfaces, the surface $P_{i,j}(u,v)$ is a bicubic polynomial function.

By Eq. (15), we can get the parameter of each pixel, the adaptive smoothing model can be formulated as

$$\begin{aligned}&x-\hat{x}_{\text{our}}=-\lambda_{y_H}\text{curv}(\hat{x}_{\text{our}}),\\&\text{s.t.},\quad \lambda_{y_H}=\lambda_G F(x,y)\end{aligned} \tag{20}$$

where $F(x,y)$ is the obtained parameter matrix via Eq. (15).

3 Algorithm

In order to improve the efficiency of the algorithm, based on the above ideas, we combine the sparse decomposition prior, L_1 norm prior, and the adaptive parameter selection to get our total-variational smoothing model[38–40]:

$$\begin{aligned}&\arg\min_{x,y_L,y_H}\frac{1}{2}\|y_{\text{pre}}-x\|_2^2+\lambda_{y_H}\sum_d\|\nabla_d x\|_1+\\&\alpha\|y_H\|_1+\frac{\kappa}{2}\sum_d\|\nabla_d\otimes y_L\|_2^2,\\&\text{s.t.},\quad y=f_L\otimes y_L+y_H,\ y_{\text{pre}}=y_L\end{aligned} \tag{21}$$

where α and κ are used to weigh data fidelity and sparsity of y_H. λ_{y_H} is used to control the sparsity of the result gradient. We note that Formula (21) is non-

differentiable and non-linear, and it is difficult to solve it directly. So, we use the ADMM to solve the problem. Based on this strategy, two Lagrange constraints are added to Eq. (21) as

$$\begin{aligned}&\arg\min_{x,y_L,y_H,T}\frac{1}{2}\|y_{\text{pre}}-x\|_2^2+\lambda_{y_H}\|T\|_1+\alpha\|y_H\|_1+\\&\frac{\kappa}{2}\|\nabla\otimes y_L\|_2^2+\frac{\gamma_1}{2}\|y-(f_L\otimes y_L+y_H)-\mu_1\|_2^2+\\&\frac{\gamma_2}{2}\|T-\nabla x-\mu_2\|_2^2\end{aligned}\tag{22}$$

In order to write conveniently, we omit d and replace the operators in four directions with ∇. γ_1 and γ_2 are the parameters of the two Lagrange constraints, which affect the convergence of ADMM method. In practical applications[41, 42], γ_1 and γ_2 are initialized with a small positive value and added in each iteration to ensure convergence. μ_1 and μ_2 are the Lagrange multipliers. T is an auxiliary parameter. All the parameters in Formula (22) are convex, so we can update each parameter iteratively until convergence. It should be noted here that $y_{\text{pre}}^{(1)}$ has two different ways to given initial values: $y_{\text{pre}}^{(1)}=y$ or $y_{\text{pre}}^{(1)}=y_L^{(1)}$. The value will affect not only the order of solving the subproblem but also the effect of result. In view of the above two points, after many experiments, we choose $y_{\text{pre}}^{(1)}=y_L^{(1)}$ to do the operation. Then we can alternately solve all the parameters and update μ_1 and μ_2.

Subproblem 1: Computing y_L

After fixing the remaining variables in Formula (22), we can get

$$\arg\min_{y_L}\frac{\kappa}{2}\|\nabla\otimes y_L\|_2^2+\frac{\gamma_1}{2}\|y-(f_L\otimes y_L+y_H)-\mu_1\|_2^2\tag{23}$$

For Formula (23), the global minimum can be obtained directly by Gradient descent, and the operation process can be accelerated by two-dimensional Fast Fourier Transform (FFT), as shown below:

$$y_L=\mathcal{F}^{-1}\left(\frac{\gamma\cdot\overline{\mathcal{F}(f_L)}\mathcal{F}(y-y_H-\mu_1)}{\kappa\cdot\overline{\mathcal{F}(\nabla)}\mathcal{F}(\nabla)+\gamma_1\cdot\overline{\mathcal{F}(f_L)}\mathcal{F}(f_L)}\right)\tag{24}$$

where $\mathcal{F}(\cdot)$ and $\mathcal{F}^{-1}(\cdot)$ represent fast Fourier transform and Inverse Fast Fourier Transform (IFFT), respectively. $\overline{\mathcal{F}(\cdot)}$ is complex conjugation operator.

Subproblem 2: Computing y_H

Let $\Lambda=y-f_L\otimes y_L+\mu_1$, we can compute y_H by the following function:

$$\arg\min_{y_H}\alpha\|y_H\|_1+\frac{\gamma_1}{2}\|y_H-\Lambda\|_2^2\tag{25}$$

This problem can be solved independently for each pixel i via a simple soft-thresholding:

$$[y_H]_i=\text{sign}([y_H]_i)\cdot\max(0,[\Lambda]_i-\frac{\alpha}{\gamma_1})\tag{26}$$

where $[\cdot]_i$ represents the pixel i of the image.

Subproblem 3: Computing T

Similarly, the variables other than T in Formula (22) are fixed. The solution of T can be expressed as follows:

$$\arg\min_T\lambda_{y_H}\|T\|_1+\frac{\gamma_2}{2}\|T-\nabla x-\mu_2\|_2^2\tag{27}$$

We need get λ_{y_H} by Eqs. (7)–(20) and Formula (25) firstly. Then, by using the same way as Eq. (26), it can be obtained that

$$[T]_i=\text{sign}([T]_i)\cdot\max(0,[\nabla x+\mu_2]_i-[\frac{\lambda_{y_H}}{\gamma_2}]_i)\tag{28}$$

Subproblem 4: Computing x

When variables T, y_L, and y_H are solved, we make $y_{\text{pre}}=y_L$. So, the optimization of x can be described as

$$\arg\min_x\frac{1}{2}\|y_L-x\|_2^2+\frac{\gamma_2}{2}\|T-\nabla x-\mu_2\|_2^2\tag{29}$$

The above function also meets the requirements of gradient descent and can be solved as solving Formula (23):

$$x=\mathcal{F}^{-1}\left(\frac{\mathcal{F}(y_L)+\gamma_2\cdot\overline{\mathcal{F}(\nabla)}\mathcal{F}(T-\mu_2)}{1+\gamma_2\cdot\overline{F(\nabla)}\mathcal{F}(\nabla)}\right)\tag{30}$$

Update μ_1 and μ_2

At the end of each iteration, the Lagrange multipliers are updated according to the ADMM:

$$\begin{aligned}\mu_1&=\mu_1+(f_L\otimes y_L+y_H-y),\\\mu_2&=\mu_2+(\nabla x-T)\end{aligned}\tag{31}$$

The steps for parameter estimation are summarized in Algorithm 1.

In the whole smoothing process, the most time-consuming part is the solution of smoothing parameter λ_{y_H}. The time of this step depends on the number of pixels N, the number of image patches K, and the size k of it. The time complexity of this step is $O(NKk^2)$. However, during the experiment, it is found that the disparity between the two adjacent iterations of λ_{y_H} is gradually decreasing. So here we take some strategies to reduce the number of calculations. After 10 iterations, it is changed to calculate every 5 generations, which can not only ensure the correctness of λ_{y_H} but also effectively reduce the operation time. In terms of convergence, all variables in Formula (22) are

Algorithm 1 Image smoothing based on global structure sparsity and parameter adaptation

Require:

Original image: y

ADMM parameters: μ_1, μ_2, γ_1, and γ_2

Ensure:

Smoothed image: x

Initialization: $x = y$, $\mu_1 = 0$, and $\mu_2 = 0$

while not converged **do**

  Solve Subproblem y_L by computing Eq. (24);

  Solve Subproblem y_H by computing Eq. (26);

  Obtain adaptive parameter λ_{y_H} from y_H;

  Solve Subproblem T by computing Eq. (28);

  Updata image x by solving Eq. (30);

  Updata μ_1 and μ_2 using Eq. (31);

end while

convex. And we use ADMM to solve the alternating optimization problem, which can ensure that the variables converge when the values of γ_1 and γ_2 are large enough, which has been proved in Refs. [43–46]. In practice, we initialize these parameters with smaller positive values and then increase them by a given factor (e.g., 5%) at each iteration, until the smoothing effect is good enough.

4 Analysis

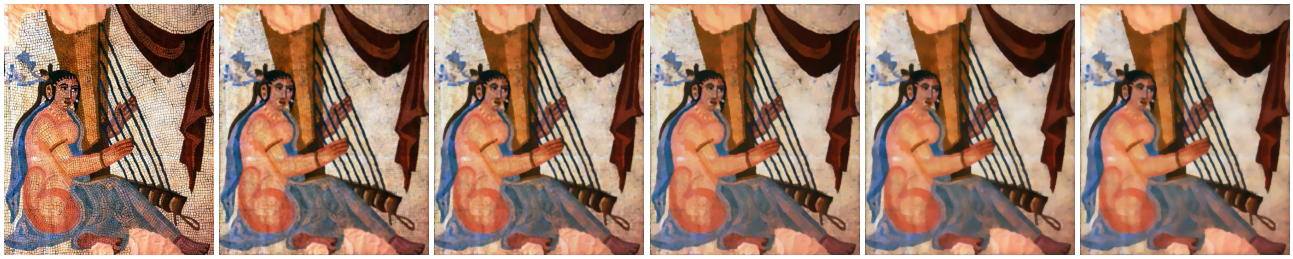
4.1 Parameter settings and dateset

For γ_1, γ_2, μ_1, and μ_2 used in ADMM algorithm, the initial setting and iteration increment of algorithm[41] are directly used. The experimental results show that this ADMM auxiliary parameter setting is also applicable to our algorithm. We set the size of f_L to 6×6 according to the convergence speed and operation efficiency. The parameters α and κ of image decomposition regularization directly affect the smoothness of y_L and the amount of information contained in y_H. The selection of these two parameters is explained in detail in Ref. [41]. Here we set $\alpha = 1$ and $\kappa = 30$. For parameter λ_G, users can give it according to their needs, and different λ_G will get different smoothing images, as shown in Fig. 6. Our dataset is combined by the images of MA[47], Deep Variational Prior (DVP)[28], and Berkeley Segmentation Dataset 500 (BSD500).

4.2 Comparison of visual effects with various methods

In order to prove the effectiveness of our algorithm, we choose the filter-based algorithm BTF[7], the variational methods L_0[20], and MA[47] to compare with our algorithm. The contrast images are all from the Refs. [28, 47]. All the algorithms are based on the code provided by the author and the parameters are manually adjusted.

The main purpose of our algorithm is to preserve local edge information while achieving image smoothing, so it can be directly compared from the visual effect. Figure 7a is the original image which has rich texture information. The purpose of smoothing is to remove texture effectively. Figure 7b is the result of BTF which can preserve the edge information of the image, such as the contour of the task face and the boundary of the wings. But its smoothing process is easily affected by texture, which makes the edge very uneven. At the same time, it can be observed that the flat area has obvious fluctuation, such as the blue background part, and the color fluctuation can still be seen from the graph, which shows that the smooth effect of the algorithm is not good for the flat area. Figure 7c is the result of L_0. It can be seen that the effect of the algorithm is very poor. For such images with abundant texture information, L_0 algorithm cannot distinguish texture from edge very well. For example, the orange stripes in the upper half of the image and the wing of the character are smoothed, while the texture of the faces of the character still exists. MA (Fig. 7d)



(a) Original image (b) λ_G = 0.001 (c) λ_G = 0.005 (d) λ_G = 0.01 (e) λ_G = 0.02 (f) λ_G = 0.025

Fig. 6 Smoothing results of different λ_G.

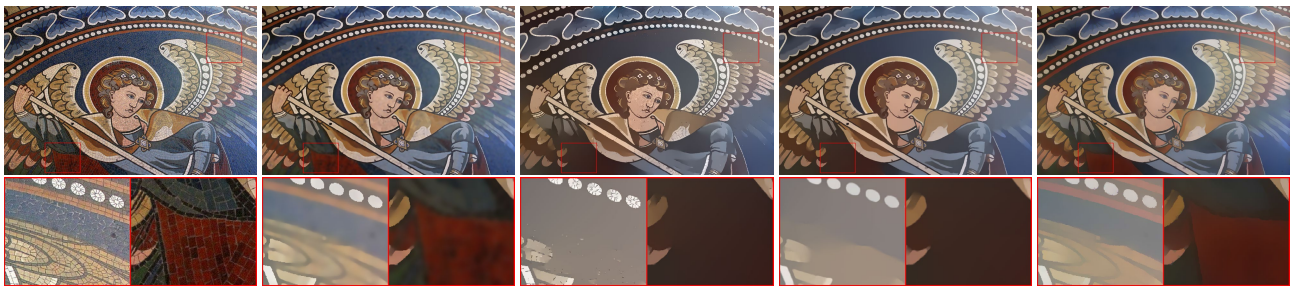


Fig. 7 Comparisons of smoothed results.

can guarantee the smoothness of the image. It can be seen from the blue background and the face of the person that it has a better effect on removing texture information. However, MA can lead to the loss of some local edge information. For example, the lower left cape and the upper right wing edge of the image have been smoothed off. In contrast, our algorithm shown in Fig. 7e can remove the texture of the flat area of the image reasonably and ensure that the local edges are not corrupted. As shown in Fig. 7e, the texture information of the figure's face and the pattern above the image have been smoothed, while the edges between the large color blocks, such as the figure's cape and wings, are well preserved.

4.3 Comparison with deep learning method

In this part, we will compare with the deep learning method. Here we choose representative algorithms DVP[28] and Fan et al.[31] for comparison. In Fig. 8, by observing Figs. 8a, 8b, and 8c, it is obvious that DVP[28] and Fan et al.[31] can remove the texture very well, and the greatest advantage of it is that it can obtain a smoother image boundary. But it is not good at preserving the details, for example, the teeth of the fish have been smoothed. Although our method is not as smooth as DVP, it is better for image details preservation, which is in line with our purpose of protecting local imageedges. Meanwhile, in order to further prove the protection effect of our method for image edges, we compare the texture removed by the two algorithms by calculating the absolute value of the difference between the original image and the results of the two algorithms, as shown in Fig. 9.

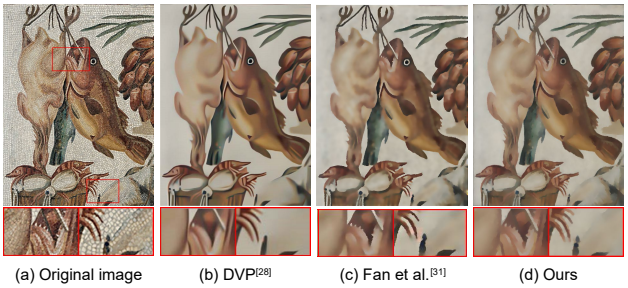


Fig. 8 Visual effect comparison with deep learning algorithm.

It can be seen clearly that the edge information contained in the texture image (Fig. 9b) removed by our algorithm is very little, most of which are the texture of the original image itself, while the texture image by DVP contains more obvious image edge information, for example, the back edge of the orange fish, which should have been retained, can be clearly observed in Fig. 9a.

There are no invariably adopted evaluation methods for image smoothing. But following the assumption that the smoothed part and the texture part are not correlated[28], we can measure the correlation between texture and smoothed result:

$$\mathrm{corr}(y-x,x)=\frac{\mathrm{cov}(y-x,x)}{\sqrt[2]{\mathrm{std}(y-x)^2\mathrm{std}(x)^2}} \tag{32}$$

where y is the original image, x is its smoothed result, and $\mathrm{cov}(\cdot)$ and $\mathrm{std}(\cdot)$ denote the covariance and the

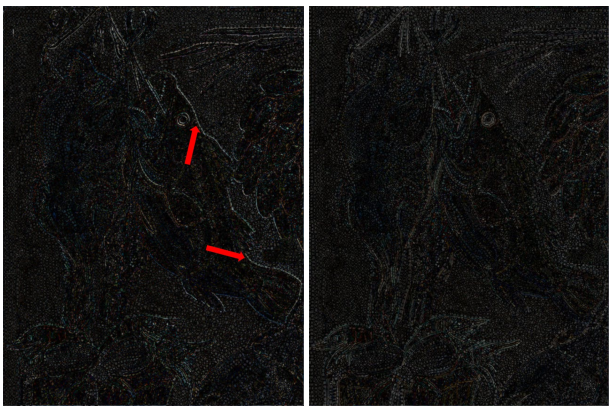


Fig. 9 Residual comparison between DVP[28] and ours.

standard deviation, respectively. We use BSD500 database to compute corr(·) of various methods and report the average value of it in Table 1.

In summary, although our algorithm has some shortcomings in image edge smoothness compared with DVP, the local edge preservation effect of our algorithm is obviously stronger than it.

5 Application

5.1 Edge extraction

In the process of edge extraction, the extraction results should be as consistent as possible with the boundaries of the objects in the visually observed figure. However, due to the influence of complex structure and unavoidable noise, it is usually difficult to obtain continuous, accurate, and good visual effect edges. And image smoothing can preserve the main edges and smooth the redundant texture, which can obviously improve the edge extraction effect. We applied the well-known Canny edge detector to original images and filtered images generated by our method and compared the resulting edge maps. As can be seen from Fig. 10, it is clear that our algorithm can preserve the edges very well, such as the overall contour of white horse and the boundary between green plants and beige background. Reasonable texture removal makes it more convenient for us to select the edge of the image. As shown in Figs. 10c and 10d, texture extraction directly from the original image will be affected by texture information, so it is difficult to extract the appropriate edge. After smoothing, the edges we extracted can perfectly represent the contours of plants, white horses, and birds in the picture.

5.2 Image abstraction

Our algorithm can obtain the non-real image with detail suppression, and the edge of the original image can be extracted by using DoG[48] operator, which can obtain the abstract image by enhancing boundary features and superimposing smoothing result, as shown in Fig. 11.

Table 1 Correlation value between smoothed and texture part.

Method	corr($y-x$, x)
BTF[7]	0.3568
L_0[20]	0.1872
MA[47]	0.2189
DVP[28]	0.1058
Fan et al.[31]	0.1139
Ours	**0.1056**

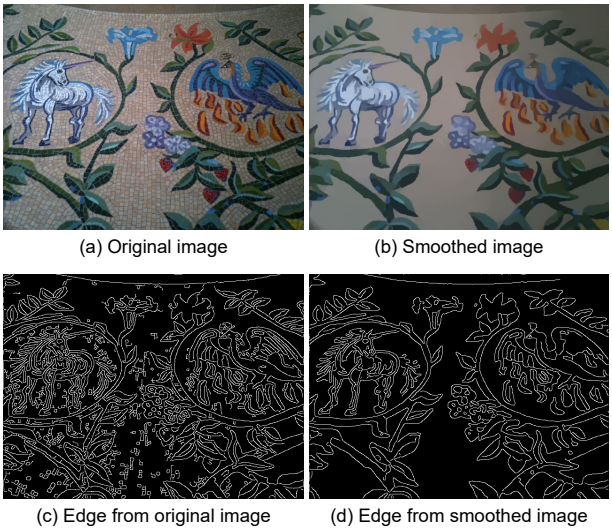


Fig. 10 Edge extraction.

5.3 Pseudo-boundary removal

Image processing operations such as compression or enlargement will lead to distortion of cartoons, clips, and other images and produce pseudo-boundary at the edge, which cannot be directly removed by conventional denoising algorithm. By preserving the main edge structure of the image, our algorithm can remove the pseudo-boundary of such images better. As shown in Fig. 12, there will be obvious pseudo-boundary in Figs. 12a−12c, such as the color distortion of animal facial contour in Fig. 12b. After smoothing by our algorithm, Figs. 12d−12f can get better results.

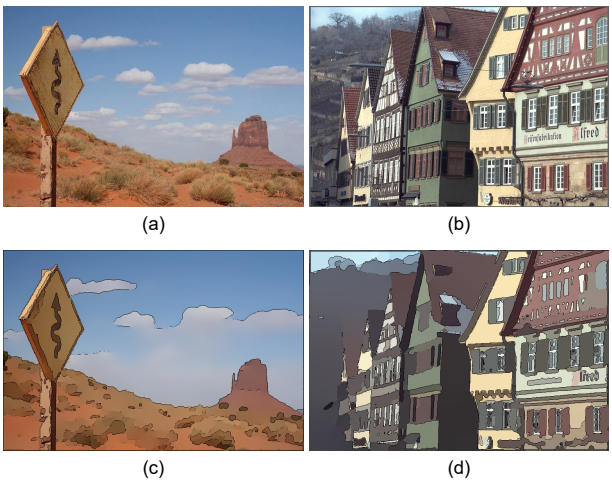


Fig. 11 Image abstraction: (a) and (b) original images; (c) and (d) abstraction images.

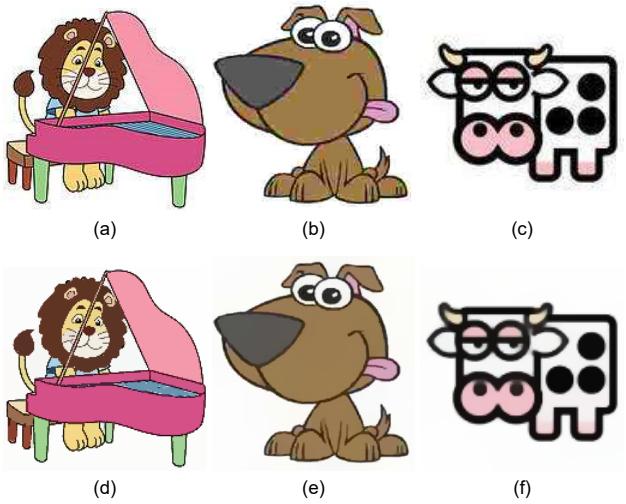


Fig. 12 Pseudo-boundary removal: (a)–(c) orginal images; (d)–(f) result images.

5.4 Image enhancement

In the process of acquiring natural images, due to the influence of hardware, environment, and shooting location, the brightness and contrast of the image will be affected, and the detail information is not obvious enough. In this case, we can use our method to enhance the original image. A smoothing layer can be obtained from the image by our algorithm, and the difference between the original image and the smoothing layer can be regarded as the detail layer. Then we enhance gradients in the detail layer, e.g., using a DoG operator. Finally, the enhanced detail layer is composed with the smooth layer. As shown in Fig. 13, the operation perfects the details and enhances the expressiveness of the image. Compared with Fig. 13a, the color of flowers in Fig. 13b is more brilliant, and the color of stems and leaves in Fig. 13a is gray while that in Fig. 13b is green. Similarly, the color of plants in Fig. 13d is brighter, and the texture of butterfly back is clearer than Fig. 13c. Compared with Fig. 13e, the detail of grassland and mountain is more obvious in Fig. 13f. And the algae in Fig. 13h has a stronger contrast and higher light intensity than Fig. 13g.

6 Conclusion

In this paper, we propose an image smoothing algorithm based on global sparsity structure and parameter matrix. We decompose the image into low frequency part and high frequency part by global sparse prior and select the smoothing parameters from the high frequency part. In order to ensure the continuity and non-negativity of the parameters, the smoothing parameters are fitted into the same size parameter matrix as the original image by a bicubic polynomial fitting method, and the matrix is used as the guide graph to smooth the low frequency image. The algorithm uses ADMM to solve the problem step by step and optimizes the final smooth image iteratively. Compared with other algorithms, our algorithm can better protect the details, avoid local edge over-smoothing caused by solving global optimization, and has better visual effect.

References

[1] C. Tomasi and R. Manduchi, Bilateral filtering for gray and color images, in *Proc. 6th Int. Conf. Computer Vision* (*IEEE Cat. No. 98CH36271*), Bombay, India, 1998, pp. 839–846.

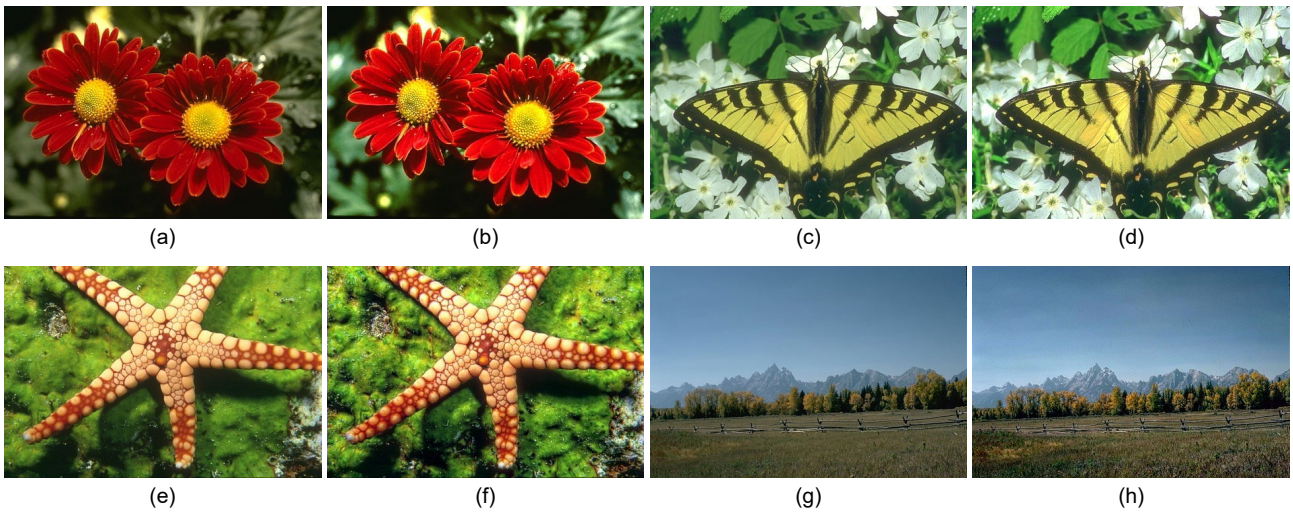


Fig. 13 Image enhancement:(a), (c), (e), and (g) original images; (b), (d), (f), and (h) enhanced images.

[2] J. Chen, S. Paris, and F. Durand, Real-time edge-aware image processing with the bilateral grid, *ACM Trans. Graph.*, vol. 26, no. 3, p. 103, 2007.

[3] S. Wu, S. Shen, X. Xu, Y. Chen, X. Zhou, D. Liu, X. Xue, and L. Qi, Popularity-aware and diverse web APIs recommendation based on correlation graph, *IEEE Trans. Comput. Soc. Syst.*, vol. 10, no. 2, pp. 771–782, 2023.

[4] C. Yang, X. Xu, X. Zhou, and L. Qi, Deep Q network–driven task offloading for efficient multimedia data analysis in edge computing–assisted IoV, *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 18, no. 2s, pp. 1–24, 2022.

[5] Y. Jia, B. Liu, W. Dou, X. Xu, X. Zhou, L. Qi, and Z. Yan, CroApp: A CNN-based resource optimization approach in edge computing environment, *IEEE Trans. Ind. Inform.*, vol. 18, no. 9, pp. 6300–6307, 2022.

[6] Y. Xu, X. Gao, C. Zhang, J. Tan, and X. Li, High quality superpixel generation through regional decomposition, *IEEE Trans. Circuits Syst. Video Technol.*, vol. 33, no. 4, pp. 1802–1815, 2023.

[7] H. Cho, H. Lee, H. Kang, and S. Lee, Bilateral texture filtering, *ACM Trans. Graph.*, vol. 33, no. 4, pp. 1–8, 2014.

[8] L. Bao, Y. Song, Q. Yang, H. Yuan, and G. Wang, Tree filtering: Efficient structure-preserving smoothing with a minimum spanning tree, *IEEE Trans. Image Process.*, vol. 23, no. 2, pp. 555–569, 2014.

[9] Z. Li, X. Xu, T. Hang, H. Xiang, Y. Cui, L. Qi, and X. Zhou, A knowledge-driven anomaly detection framework for social production system, *IEEE Trans. Comput. Soc. Syst*., doi: 10.1109/2022.3217790.

[10] 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.

[11] X. Xu, J. Gu, H. Yan, W. Liu, L. Qi, and X. Zhou, Reputation-aware supplier assessment for blockchain-enabled supply chain in industry 4.0, *IEEE Trans. Ind. Inform.*, vol. 19, no. 4, pp. 5485–5494, 2023.

[12] L. Qi, W. Lin, X. Zhang, W. Dou, X. Xu, and J. Chen, A correlation graph based approach for personalized and compatible web APIs recommendation in mobile APP development, *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 6, pp. 5444–5457, 2023.

[13] Z. Farbman, R. Fattal, D. Lischinski, and R. Szeliski, Edge-preserving decompositions for multi-scale tone and detail manipulation, *ACM Trans. Graph.*, vol. 27, no. 3, pp. 1–10, 2008.

[14] K. Mikolajczyk and C. Schmid, An affine invariant interest point detector, in *Proc. 7th European Conf. Computer Vision*, Copenhagen, Denmark, 2002, pp. 128–142.

[15] H. Dai, J. Yu, M. Li, W. Wang, A. X. Liu, J. Ma, L. Qi, and G. Chen, Bloom filter with noisy coding framework for multi-set membership testing, *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 7, pp. 6710–6724, 2023.

[16] B. Cai, X. Xing, and X. Xu, Edge/structure preserving smoothing via relativity-of-Gaussian, in *Proc. 2017 IEEE Int. Conf. Image Processing* (*ICIP*), Beijing, China, 2018, pp. 250–254.

[17] X. Xu, Z. Fang, L. Qi, X. Zhang, Q. He, and X. Zhou, TripRes: Traffic flow prediction driven resource reservation for multimedia IoV with edge computing, *ACM Trans. Multimedia Computing, Communications, and Applications*, vol. 17, no. 2, pp. 1–21, 2021.

[18] L. I. Rudin, S. Osher, and E. Fatemi, Nonlinear total variation based noise removal algorithms, *Phys. D Nonlinear Phenom.*, vol. 60, nos.1–4, pp. 259–268, 1992.

[19] L. Xu, Q. Yan, Y. Xia, and J. Jia, Structure extraction from texture via relative total variation, *ACM Trans. Graph.*, vol. 31, no. 6, pp. 1–10, 2012.

[20] L. Xu, C. Lu, Y. Xu, and J. Jia, Image smoothing via L_0 gradient minimization, *ACM Trans. Graph.*, vol. 30, no. 6, pp. 1–12, 2011.

[21] S. Ono, L_0 gradient projection, *IEEE Trans. Image Process*., vol. 26, no. 4, pp. 1554–1564, 2017.

[22] Y. Wang, L. Qi, R. Dou, S. Shen, L. Hou, Y. Liu, Y. Yang, and L. Kong, An accuracy-enhanced group recommendation approach based on DEMATEL, *Pattern Recognit. Lett.*, vol. 167, pp. 171–180, 2023.

[23] Y. Akai, T. Shibata, R. Matsuoka, and M. Okuda, L0 smoothing based on gradient constraints, in *Proc. 2018 25th IEEE Int. Conf. Image Processing* (*ICIP*), Athens, Greece, 2018, pp. 3943–3947.

[24] W. Liu, P. Zhang, Y. Lei, X. Huang, J. Yang, and M. Ng, A generalized framework for edge-preserving and structure-preserving image smoothing, *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 10, pp. 6631–6648, 2022.

[25] Q. Chen, J. Xu, and V. Koltun, Fast image processing with fully-convolutional networks, in *Proc. 2017 IEEE Int. Conf. Computer Vision* (*ICCV*), Venice, Italy, 2017, pp. 2516–2525.

[26] L. Xu, J. Ren, Q. Yan, R. Liao, and J. Jia, Deep edge-aware filters, in *Proc. 32nd Int. Conf. Machine Learning*, Lille, France, 2015, pp. 1669–1678.

[27] Y. Liu, D. Li, S. Wan, F. Wang, W. Dou, X. Xu, S. Li, R. Ma, and L. Qi, A long short-term memory-based model for greenhouse climate prediction, *Int. J. Intell. Syst.*, vol. 37, no. 1, pp. 135–151, 2022.

[28] Y. Kim, B. Ham, M. N. Do, and K. Sohn, Structure-texture image decomposition using deep variational priors, *IEEE Trans. Image Process.*, vol. 28, no. 6, pp. 2692–2704, 2019.

[29] L. Qi, Y. Liu, Y. Zhang, X. Xu, M. Bilal, and H. Song, Privacy-aware point-of-interest category recommendation in Internet of Things, *IEEE Internet Things J.*, vol. 9, no. 21, pp. 21398–21408, 2022.

[30] J. Li, K. Qin, R. Xu, and H. Ji, Deep scale-aware image smoothing, in *Proc. ICASSP 2022 - 2022 IEEE Int. Conf. Acoustics, Speech and Signal Processing* (*ICASSP*), Singapore, 2022, pp. 2105–2109.

[31] Q. Fan, J. Yang, D. Wipf, B. Chen, and X. Tong, Image smoothing via unsupervised learning, *ACM Trans. Graph.*, vol. 37, no. 6, pp. 1–14, 2018.

[32] M. Li, Y. Fu, X. Li, and X. Guo, Deep flexible structure preserving image smoothing, in *Proc. 30th ACM Int. Conf. Multimedia*, Lisboa, Portugal, 2022, pp. 1875–1883.

[33] Y. Liu, H. Wu, K. Rezaee, M. R. Khosravi, O. I. Khalaf, A. A. Khan, D. Ramesh, and L. Qi, Interaction-enhanced

and time-aware graph convolutional network for successive point-of-interest recommendation in traveling enterprises, *IEEE Trans. Ind. Inform.*, vol. 19, no. 1, pp. 635–643, 2023.

[34] Y. Yang, H. Hui, L. Zeng, Y. Zhao, Y. Zhan, and T. Yan, Edge-preserving image filtering based on soft clustering, *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 7, pp. 4150–4162, 2022.

[35] X. Ma, X. Li, Y. Zhou, and C. Zhang, Image smoothing based on global sparsity decomposition and a variable parameter, *Comput. Vis. Medium.*, vol. 7, no. 4, pp. 483–497, 2021.

[36] X. Zhang, Y. Sun, H. Liu, Z. Hou, F. Zhao, and C. Zhang, Improved clustering algorithms for image segmentation based on non-local information and back projection, *Inf. Sci.*, vol. 550, pp. 129–144, 2021.

[37] X. Yu, H. Liu, Y. Lin, Y. Wu, and C. Zhang, Auto-weighted sample-level fusion with anchors for incomplete multi-view clustering, *Pattern Recognition*, vol. 130, p. 108772, 2022.

[38] Y. Wang, J. Yang, W. Yin, and Y. Zhang, A new alternating minimization algorithm for total variation image reconstruction, *SIAM J. Imaging Sci.*, vol. 1, no. 3, pp. 248–272, 2008.

[39] T. Yao, X. Kong, H. Fu, and Q. Tian, Discrete semantic alignment hashing for cross-media retrieval, *IEEE Trans. Cybern.*, vol. 50, no. 12, pp. 4896–4907, 2020.

[40] Q. He, S. Tan, F. Chen, X. Xu, L. Qi, X. Hei, H. Jin, and Y. Yang, EDIndex: Enabling fast data queries in edge storage systems, in *Proc. 46th Int. ACM SIGIR Conf. Research and Development in Information Retrieval*, Taipei, China, 2023, pp. 675–685.

[41] M. Zhang and C. Desrosiers, High-quality image restoration using low-rank patch regularization and global structure sparsity, *IEEE Trans. Image Process.*, vol. 28, no. 2, pp. 868–879, 2019.

[42] J. Pan, D. Sun, J. Zhang, J. Tang, J. Yang, Y. W. Tai, and M. H. Yang, Dual convolutional neural networks for low-level vision, *Int. J. Comput. Vis.*, vol. 130, no. 6, pp. 1440–1458, 2022.

[43] J. Eckstein and D. P. Bertsekas, On the Douglas—Rachford splitting method and the proximal point algorithm for maximal monotone operators, *Math. Program.*, vol. 55, nos. 1–3, pp. 293–318, 1992.

[44] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, Distributed optimization and statistical learning via the alternating direction method of multipliers, *Found. Trends® Mach. Learn.*, vol. 3, no. 1, pp. 1–122, 2011.

[45] R. Zhang and J. T. Kwok, Asynchronous distributed ADMM for consensus optimization, in *Proc. 31st Int. Conf. Mach. Learn.*, Beijing, China, 2014, pp. 3689–3697.

[46] D. L. Sun and C. Févotte, Alternating direction method of multipliers for non-negative matrix factorization with the beta-divergence, in *Proc. 2014 IEEE Int. Conf. Acoustics, Speech and Signal Processing* (*ICASSP*), Florence, Italy, 2014, pp. 6201–6205.

[47] G. H. Ma, M. L. Zhang, X. M. Li, and C. M. Zhang, Image smoothing based on image decomposition and sparse high frequency gradient, *J. Comput. Sci. Technol.*, vol. 33, no. 3, pp. 502–510, 2018.

[48] D. G. Lowe, Distinctive image features from scale-invariant keypoints, *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2004.

Suwei Wang received the BS degree in software engineering from Shandong University, Jinan, China in 2015. She is currently pursuing the PhD degree at School of Computer Science and Technology, Shandong University, Jinan, China. Her research interests include saliency detection and image segmentation.

Xuemei Li received the master and PhD degrees from Shandong University, Jinan, China in 2004 and 2010, respectively. From 2013 to 2014, she was a visiting scholar of University of Houston, USA. She is currently a professor at School of Software, Shandong University, China, and a member of Intelligent Computing and Information Visualization (IC&IV) lab. She is engaged in research on intelligent graphics and image processing, Computer-Aided Geometric Design (CAGD), time series analysis, and information visualization.

Xiang Ma received the master degree from Shandong University, China in 2020. He is currently pursuing the PhD degree at School of Software, Shandong University, Jinan, China. His research interests include image processing, time series analysis, and deep learning.