International Journal of Innovative
Computing, Information and Control
Volume **12**, Number **1**, February **2016**

ICIC International ©2016 ISSN 1349-4198
pp. 113–124

A NEW INPAINTING METHOD FOR OBJECT REMOVAL BASED ON PATCH LOCAL FEATURE AND SPARSE REPRESENTATION

Lei Zhang[1,2], Baosheng Kang[1,*], Benting Liu[1] and Zhenhua Bao[1]

[1]School of Information Science and Technology
Northwest University
No. 1, Xuefu Road, Chang'an District, Xi'an 710127, P. R. China
*Corresponding author: bskang@163.com; inpainting@126.com; {benting; baozhenhua07}@163.com

[2]Public Computer Teaching Department
Yuncheng University
No. 1155, Fudan West Street, Yuncheng 044000, P. R. China

Received July 2015; revised November 2015

Abstract. *The traditional inpainting methods for object removal from image need to traverse the entire source region and search for the exemplar patches, so it may take a long time to recover the image. Furthermore, these methods simply use the Sum of Squared Differences (SSD) to measure the degree of similarity, which may result in that the target patch is replaced by the inappropriate exemplar patch, and thus there is a risk of introducing undesired objects in recovered image. In view of this situation, we propose a new inpainting method based on patch local feature and sparse representation. First of all, we classify the patches according to their local feature. For smooth patch, we calculate its sparse coefficient over a redundant dictionary, and then recover it using the dictionary and the sparse coefficient. For texture patch, we recover it using the Criminisi method so as to protect the image details. In this way, the sparse representation of the patch can be skillfully used to remove objects from an image. A number of examples on real images show that, the proposed method not only costs less running time, but also avoids introducing undesired objects in recovered images.*

Keywords: Image inpainting, Object removal, Local feature, Sparse representation

1. **Introduction.** Image inpainting technique remains a longstanding challenge in image processing and computer vision [1,2], which aims to use undamaged image information to restore the lost or damaged parts of image according to certain criteria, so that the restored image can be as close as possible to the original visual effect [3]. It has been widely used in real life [4-7]. For example, in the film and television production process, we can use it to remove the unwanted objects in the scene, without the need to re-shoot, which can save production costs to a large extent. In terms of heritage conservation, we can scan the damaged cultural relics, use the technique to remove the damaged regions, and recover the original cultural relics according to restored results, which can reduce the risk of secondary damage to cultural relics.

The most fundamental inpainting methods are based on the Partial Differential Equation (PDE), in which the missing regions are filled by diffusing the image information from the known regions into the missing regions [8,9]. Bertalmio et al. [10] presented the PDE-based method and used it in image inpainting. Chan and Shen [11] presented Total Variance (TV) model for inpainting problem and Curvature-Driven Diffusions (CDD) [12] model used to fix connectivity problem in TV model. Although PDE-based methods have achieved convincingly excellent results in recovering the small, non-textured region,

they tend to induce over-smooth effect or stair-case effect in the larger, textured missing region.

The second category of methods are based on the exemplar, which stem from texture synthesis [13]. They aim to restore the large or textured missing regions in a visually plausible manner [14], and they have received considerable attention in the past few years [15,16]. Up to now, the famous exemplar-based inpainting method is proposed by Criminisi et al. [17], it uses a predefined priority function to decide the filling order and selects the target patch with the highest priority, then searches for a patch which is the most similar to the target patch in the source region according to the similarity criterion, and at last the value of each unknown pixel in the target patch is copied from its corresponding position inside the found patch. Compared with PDE-based approach, exemplar-based approach can obtain more visually reasonable results even in the large missing region.

Recently, the sparse representation of signals has attracted more and more attention of researchers [18,19]. Using an over-complete dictionary that contains prototype signal-atoms, signals are described by sparse linear combinations of these atoms. Based on sparse representation theory, the K-SVD algorithm [20] was used to design over-complete dictionaries that could lead to the best representation for each signal, and it was applied to filling the missing pixels in images. Elad et al. [21] introduced an image inpainting method that is capable of filling in holes in overlapping texture and cartoon image layers. It is a direct extension of the image decomposition method called Morphological Component Analysis (MCA) [22,23]. These methods can obtain convincingly excellent results in filling missing pixels and restoring scratches. However, some details of images obtained from recovering the large missing region or removing large objects from images are not satisfactory.

Removing object from the image means that we need to recover the large missing region, and the most commonly used method is proposed by Criminisi et al. [17], but it still suffers from some limitations. For each target patch, it must traverse the entire source region of image to search for the most similar exemplar patch, which will cost much time and reduce the efficiency of algorithm. Besides, it simply uses the Sum of Squared Differences (SSD) of the already existing pixels in the two patches to measure the degree of similarity, which may result in that the target patch is replaced by the inappropriate exemplar patch, and this error may be accumulated as the inpainting progresses, and finally some undesired objects may be introduced in recovered images.

However, what is exciting is that we find the sparse representation can solve these problems. Based on the above analysis, we propose a new inpainting method for object removal. We divide all the image patches into two categories, including smooth patch and texture patch, and identify which category the target patch belongs to. If it is a smooth patch, we calculate its sparse coefficient over a redundant dictionary, and then recover the target patch using the dictionary and the sparse coefficient. If it is a texture patch, we recover it using the famous Criminisi method in order to protect the details of image.

Compared with the traditional methods, the proposed method can skillfully combine the sparse-representation-based method and the exemplar-based method. We can make use of the respective advantages of these methods. The experimental results show that the proposed method costs less running time and avoids introducing undesired objects in recovered images.

The rest of this paper is organized as follows. In Section 2, we introduce the basic theory of sparse representation, and describe how to use it to recover the damaged image patch in our method. In Section 3, the details of the proposed algorithm are introduced.

The experiments and comparisons are performed in Section 4. Finally, we conclude this work in Section 5.

2. Sparse Representations.

2.1. Sparse representations of signals. At present, researchers are increasingly interested in sparse representations of signals and gradually apply it to image processing. Using an over-complete dictionary $D \in \mathbb{R}^{n\times K}$ that contains K prototype signal-atoms for columns $\{d_j\}_{j=1}^{K}$, a signal $x \in \mathbb{R}^n$ can be represented as a sparse linear combination of these atoms [20]. The representation of x may either be exact:

$$x = D\alpha \tag{1}$$

Or approximate:

$$x \approx D\alpha \ \text{ s.t. } \ \|x - D\alpha\|_2 \leq \varepsilon \tag{2}$$

where, the vector $\alpha \in \mathbb{R}^K$ contains the representation coefficients of the signal x.

For each image patch, which contains $\sqrt{n} \times \sqrt{n}$ pixels, ordered lexicographically as column vector $x \in \mathbb{R}^n$, we assume that there is a corresponding redundant dictionary $D \in \mathbb{R}^{n\times K}$ $(K > n)$ which can represent the x very sparsely [24]:

$$\hat{\alpha} = \arg\min_{\alpha} \|\alpha\|_0 \ \text{ s.t. } \ x \approx D\alpha \tag{3}$$

where, the solution of Formula (3) is indeed very sparse, i.e., $\|\hat{\alpha}\|_0 \ll n$, the notation $\|\hat{\alpha}\|_0$ is l_0-norm, counting the nonzero entries in $\hat{\alpha}$. It means that the image patch can be represented as a linear combination of few atoms from the redundant dictionary D.

Notice that the constraint can be turned into a penalty term, so the constrained optimization is replaced with unconstrained optimization, and the above optimization task can be changed to:

$$\hat{\alpha} = \arg\min_{\alpha} \|D\alpha - x\|_2^2 + \lambda\|\alpha\|_0 \tag{4}$$

where, the term $\|\alpha\|_0$ encourages the sparsity of the coefficient vector, and the parameter λ controls the tradeoff between the reconstruction error and the sparsity.

While this problem is, in general, very hard to solve, the matching and the basis pursuit algorithms can be used quite effectively to get an approximated solution [25]. In this paper, we use the orthonormal matching pursuit (OMP) [26] because of its simplicity and efficiency.

2.2. Choice of the dictionary. As described above, in order to compute the sparse coefficients of an image, we first have to choose a suitable over-complete dictionary, depending on the problem to be solved. At present, the K-SVD algorithm is often used to obtain an over-complete dictionary. Given a set of training signals, it seeks the dictionary that leads to the best possible representations for each member in this set with strict sparsity constraints [20]. It is flexible and can work with any pursuit method.

In our work, we adopt the DCT (Discrete Cosine Transform) dictionary which is also a commonly used dictionary. There are two reasons: the first and the most important reason is that it can obtain satisfactory results because we only use the redundant dictionary to restore the smooth image patch in our work. The second is that it does not need to take a lot of time to train a large number of samples and it can be easily obtained.

2.3. **Recovery of damaged patch.** Here we describe in detail how to recover the damaged patch in the proposed method. Suppose that there is a damaged image patch which contains $\sqrt{n} \times \sqrt{n}$ pixels. For ease of computation, we turn it into a column vector $x \in \mathbb{R}^n$ in the lexicographic order. From the inpainting point of view, removing objects from image means that some pixels in x are corrupted and we need to recover these pixels, so the first question is how do we know which pixels are damaged in patch. Fortunately, this problem can easily be solved, because the corresponding mask image can indicate the indexes of damaged pixels. Thus, we use a set R to record the indexes of corrupted pixels. The new vector x' which only contains undamaged pixels can be obtained by removing the corrupted pixels from x. In the same way, the new dictionary D' can be obtained by removing the corresponding rows from dictionary D. The sparse coefficient $\hat{\alpha}'$ can be estimated as follows:

$$\hat{\alpha}' = \arg\min_{\alpha} \|D'\alpha - x'\|_2^2 + \lambda\|\alpha\|_0 \tag{5}$$

Then, we use the OMP algorithm to obtain the sparse coefficient $\hat{\alpha}'$. Finally, we recover the target patch according to the following formula:

$$\hat{x_i} = \begin{cases} x_i & i \notin R \\ D\hat{\alpha}' & i \in R \end{cases} \tag{6}$$

where, i is the index of each pixel in x. In this way, the current target patch can be recovered efficiently.

3. **Inpainting Algorithm.** For easy understanding, we adopt the same notations used in [17]. As shown in Figure 1, Ω is the target region (i.e., the missing region) which will be removed and filled, Φ is the source region (i.e., the known region), it may be defined as the entire image I minus the target region Ω ($\Phi = I - \Omega$), and $\partial\Omega$ denotes the boundary of the target region Ω. Suppose that the patch Ψ_{p} centered at the point p ($\mathrm{p} \in \partial\Omega$) is to be filled. Given the patch Ψ_{p}, n_{p} is the unit vector orthogonal to the boundary $\partial\Omega$ and $\nabla I_{\mathrm{p}}^{\perp}$ is the isophote at point p.

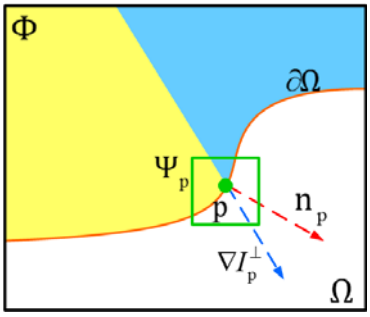


FIGURE 1. Notation diagram

3.1. **Classification of image patches.** In our work, in order to recover different image patches using different methods, we divide them into two categories: smooth patch and texture patch. The local variance can reflect the structural characteristics of the image patch, and the calculation is simple, so we use the local variance to classify image patches, and we judge which category a patch belongs to. The local variance of the image patch is defined as follows [27]:

$$v = \frac{1}{n}\sum_{i=0}^{n-1}(x_i - \bar{x_i})^2 \tag{7}$$

where, n is the number of pixels in the patch, $\bar{x}_i$ is the mean of all pixels in the patch and it is defined as follows:

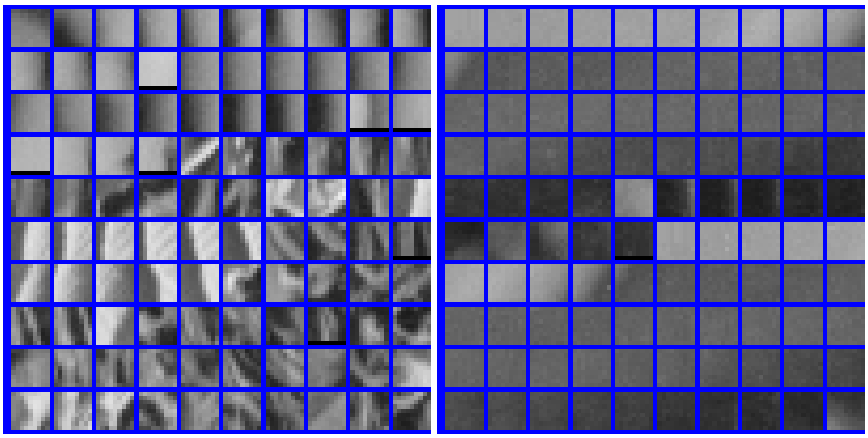
$$\bar{x}_i = \frac{1}{n}\sum_{i=0}^{n-1} x_i \tag{8}$$

Thus we can set a threshold β, and if $v < \beta$, we think that it is a smooth patch, else it is a texture patch. Figure 2 shows parts of the classification results of an image.

However, there is a question: how to determine the threshold? Next, we introduce the determination of the threshold in our work. We use a simple yet very effective approach to determine the threshold β. First of all, calculate the variances of all image patches. Then they are sorted in ascending order and saved in a matrix $\mathbf{M} \in \mathbb{R}^{n\times 1}$. Finally, the threshold β is set as follows:

$$\beta = \mathbf{M}[n \times \omega] \tag{9}$$

where, n is the number of image patches, ω is set to different values according to the distribution of the variances. In Figure 3 and Figure 4, we show two natural images and their respective distribution of variances. If the majority of the variances is small, as shown in Figure 4(a), it means the image contains a large amount of smooth patches, as shown in Figure 3(a). Hence, ω is set between 0.6 and 0.8. If the distribution is relatively uniform, as shown in Figure 4(b), it means the texture region and smooth region in the

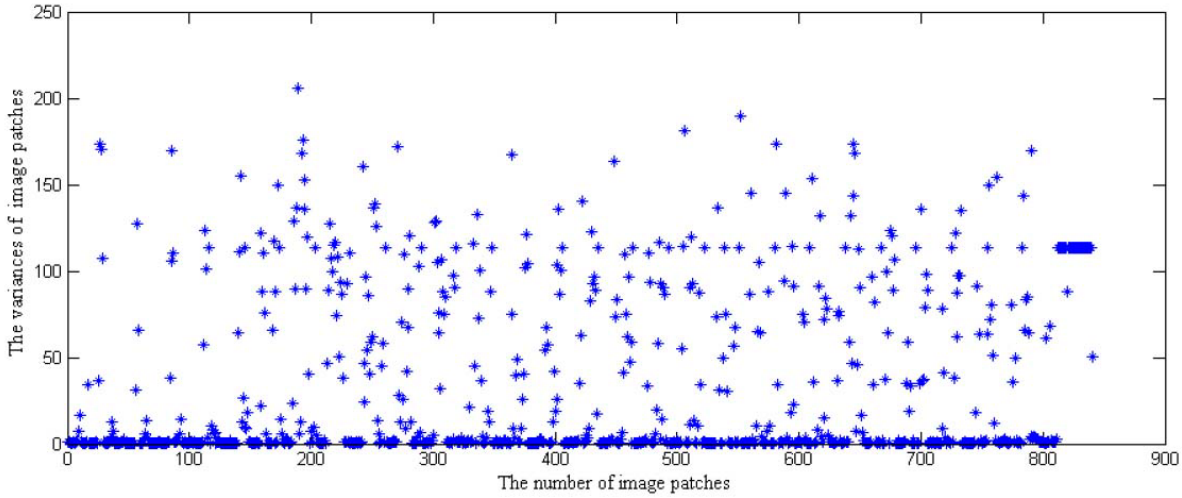


(a) Texture patches (b) Smooth patches

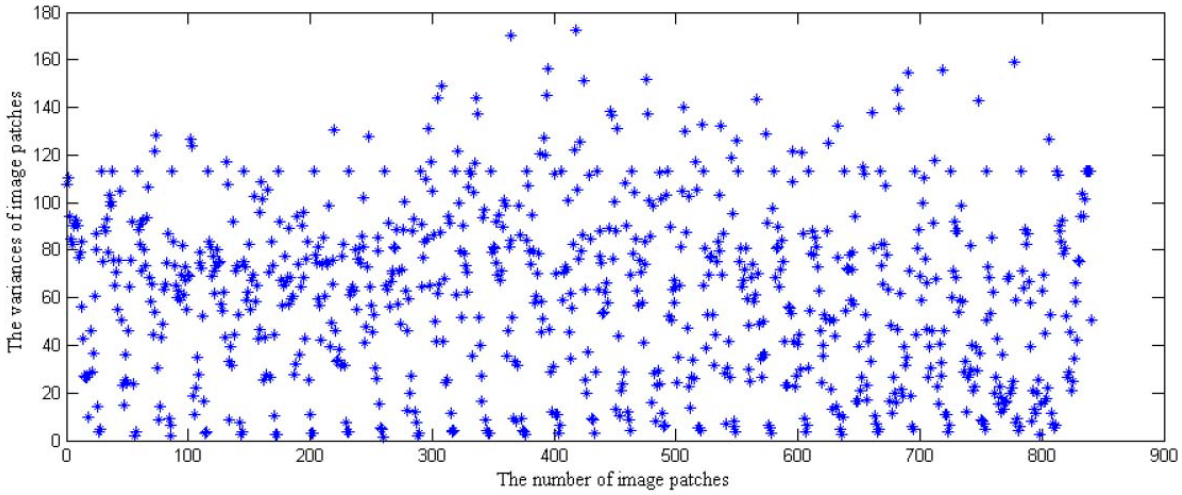
FIGURE 2. Classification results of image patches

(a) (b)

FIGURE 3. Two natural images



(a) The majority of the variances being small



(b) The distribution being relatively uniform

FIGURE 4. The different distribution of the variances

image are almost the same, as shown in Figure 3(b). In order to better protect the details, ω is set between 0.2 and 0.4. Later experiments prove the effectiveness of this approach.

3.2. **Patch priority.** We adopt the method proposed in [17] to compute the patch priority and determine the filling order, and it is biased toward those patches which are on the continuation of strong edges and which are surrounded by high-confidence pixels, so it can reserve the edge information efficiently.

For each patch Ψ_{p} centered at the point p ($\mathrm{p} \in \partial\Omega$) having a patch priority, it is defined as follows:

$$P(\mathrm{p}) = C(\mathrm{p}) \times D(\mathrm{p}) \tag{10}$$

where, $C(\mathrm{p})$ is the confidence term and $D(\mathrm{p})$ is the data term. The confidence term $C(\mathrm{p})$ indicates how many existing pixels there are in the patch Ψ_{p}. It is defined as follows:

$$C(\mathrm{p}) = \frac{\sum_{\mathrm{q} \in \Psi_{\mathrm{p}} \cap \bar{\Omega}} C(\mathrm{q})}{|\Psi_{\mathrm{p}}|} \tag{11}$$

where, $|\Psi_{\mathrm{p}}|$ is the area of patch Ψ_{p}. During the initialization, $C(\mathrm{p})$ is set as follows:

$$C(\mathrm{p}) = \begin{cases} 0 & \forall \mathrm{p} \in \Omega \\ 1 & \forall \mathrm{p} \in \Phi \end{cases} \tag{12}$$

The data term $D(\mathrm{p})$ indicates how strong the isophote hitting the boundary is. It is especially important because it encourages the linear structure to be synthesized first. It is defined as follows:

$$D(\mathrm{p}) = \frac{\left|\nabla I_{\mathrm{p}}^{\perp} \bullet n_{\mathrm{p}}\right|}{\alpha} \tag{13}$$

where, α is a normalization factor. Once all priorities have been computed, we find the target patch $\Psi_{\hat{\mathrm{p}}}$ with the highest priority.

3.3. **Recovery of target patch.** We identify which category the target patch belongs to, and if it is a smooth patch, we calculate its sparse coefficients according to Formula (5), and then recover the target patch using the dictionary and the sparse coefficients according to Formula (6).

If it is a texture patch, we adopt the algorithm proposed in [17] to recover the patch. We search in the source region for the exemplar patch which is the most similar to $\Psi_{\hat{\mathrm{p}}}$ according to the following formula:

$$\Psi_{\hat{\mathrm{q}}} = \arg\min_{\Psi_{\mathrm{q}} \in \Phi} d\left(\Psi_{\hat{\mathrm{p}}}, \Psi_{\mathrm{q}}\right) \tag{14}$$

where, $d\left(\Psi_{\mathrm{A}}, \Psi_{\mathrm{B}}\right)$ is defined as the SSD of the already existing pixels in the two patches:

$$d\left(\Psi_{\mathrm{A}}, \Psi_{\mathrm{B}}\right) = \sum_{i,j \in \mathrm{A} \cap \bar{\Omega}} \left|\Psi_{\mathrm{A}}(i,j) - \Psi_{\mathrm{B}}(i,j)\right|^2 \tag{15}$$

After finding the most similar patch $\Psi_{\hat{\mathrm{q}}}$, the target patch $\Psi_{\hat{\mathrm{p}}}$ can be replaced as follows:

$$\Psi_{\hat{\mathrm{p}}}(i,j) = \Psi_{\hat{\mathrm{q}}}(m,n) \tag{16}$$

where, $(i,j) \in \Psi_{\hat{\mathrm{p}}} \cap \Omega$, (m,n) is the corresponding position to (i,j) inside $\Psi_{\hat{\mathrm{q}}}$.

3.4. **Algorithm description.** Here, the proposed algorithm steps are described as follows:

Step 1. Identify the target regions. The target regions are extracted manually, and use the mask image to indicate the target regions.

Step 2. Compute the local variances of all the image patches, and determine the threshold β according to Formula (9).

Step 3. Find the image patches which locate on the boundary of the target regions, compute the patch priorities according to Formula (10), and select the target patch $\Psi_{\hat{\mathrm{p}}}$ with the highest priority.

Step 4. Calculate the local variance v of the target patch $\Psi_{\hat{\mathrm{p}}}$. According to the value of v and β, use different methods to recover the target patch $\Psi_{\hat{\mathrm{p}}}$.

Step 4.1. If $v \leq \beta$, indicating that the target patch is a smooth patch, we recover it according to Formula (6), i.e., we use the sparse representation to recover the current patch.

Step 4.2. If $v > \beta$, indicating that the target patch is a texture patch, we recover it according to Formula (16), i.e., we use the Criminisi method to recover the current patch.

Step 5. After recovering the target patch, update the value of confidence term according to the following formula:

$$C(\mathrm{q}) = C(\hat{\mathrm{p}}) \quad \forall \mathrm{q} \in \Psi_{\hat{\mathrm{p}}} \cap \Omega \tag{17}$$

The algorithm iterates the above steps until all pixels in the target region have been filled.

4. **Experimental Results.** In this section, we demonstrate the experimental results achieved by applying our method to some images. In these experiments, the size of each image is 256×256, the size of image patch is 9×9, the size of redundant dictionary is 81×256, and all the experiments are run on the computer with the configuration of 2.1GHz processor and 12GB RAM.

4.1. **Application of the proposed method.** In order to verify the practicality of the proposed method, we apply it on a variety of natural images. We choose some objects in these images, and then mark them as green, and finally, recover these regions using the proposed method. Some experimental results are shown in Figure 5. In each row, the first is the original image, the second is the green target region, and the third is the recovered image.

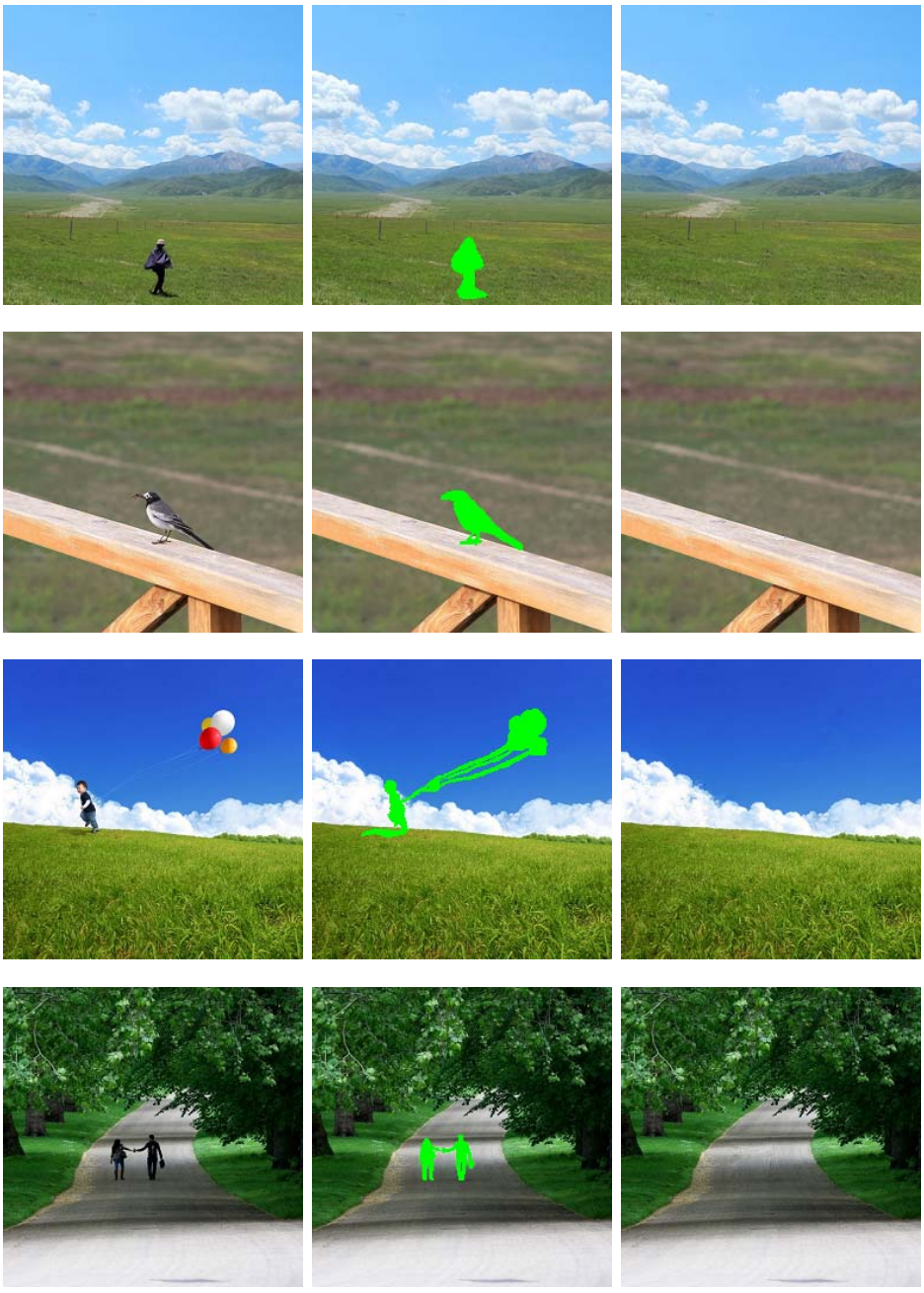


FIGURE 5. The original images and the recovered images

As can be seen from Figure 5, the proposed method can remove these objects in a visually reasonable way and obtain convincingly excellent results.

4.2. **Comparison of inpainting effect.** Compared with the traditional method for object removal, our method has two main advantages. The first is that it can avoid introducing undesired objects and inducing stair-case effect in the recovered image. To better illustrate, we select some images, use our method and the classical Criminisi method [17] to recover them respectively, and the results are shown in Figure 6. In each row, the first is the original image, the second is the green target region, the third is the recovered image using the Criminisi method, and the last one is the result of our method.

From Figure 6 we can see that, the results of our method are more natural, but some unwanted objects have been introduced in the recovered images of Criminisi method. The reason is that we use the sparse representation of image patch to recover the target patch,

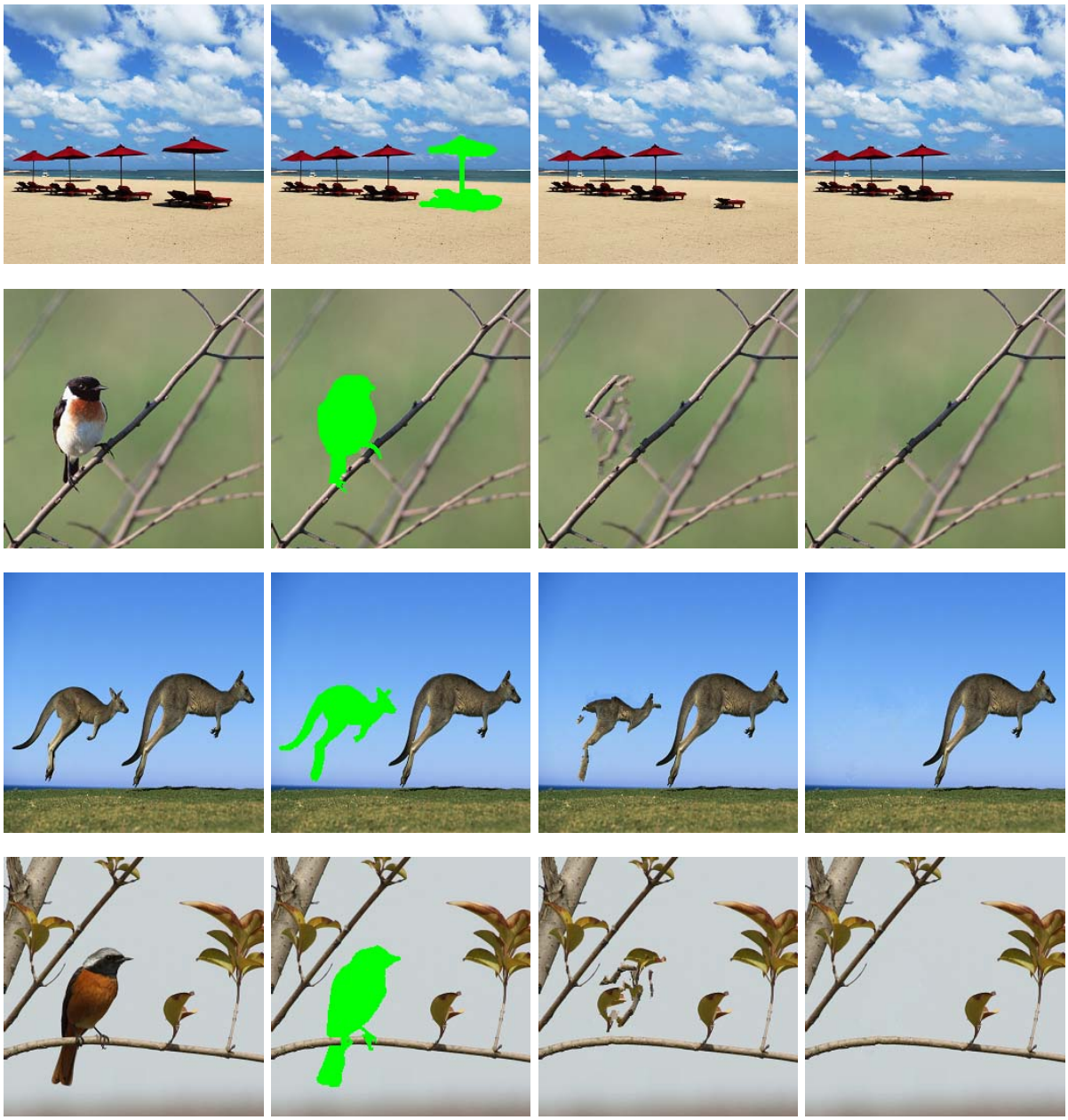


FIGURE 6. The results of Criminisi method and our method

rather than blindly use exemplar patch found in the source region to replace the target patch, so our method can avoid using unreasonable patch to replace the target patch. However, Criminisi method simply uses the SSD to measure the degree of similarity, which may result in that the target patch is replaced by the inappropriate exemplar patch, and this error may be accumulated as the inpainting progresses, and finally some undesired objects may be introduced in recovered images.

4.3. **Comparison of running time.** Another advantage of our method is that it is a fast inpainting method for object removal. Compared with the classical method, it costs less time when processing the same image. For ease of comparison, we record the respective running time they cost when using the Criminisi method and our method to recover all of the above images. The results are demonstrated in Table 1, where all of the above images are numbered sequentially, the object size is the number of pixels in the object to be removed, and the time is measured in seconds.

TABLE 1. The running time of Criminisi method and our method (uints: seconds)

Image	Object size	Time of Criminisi method	Time of proposed method	Reduction rate
Image 1	910	10.21	8.48	16.94%
Image 2	1158	12.46	9.86	20.87%
Image 3	2582	29.93	23.66	20.95%
Image 4	751	9.73	8.70	10.59%
Image 5	2278	24.95	17.96	28.02%
Image 6	4165	38.29	23.14	39.57%
Image 7	3528	33.40	22.46	32.75%
Image 8	4950	45.21	34.95	22.69%
Average		25.52	18.65	26.92%

As can be seen from Table 1, compared with the Criminisi method, the average running time of our method is reduced by 6.87 seconds, and the average reduction rate is 26.92%. The reason is that we directly use sparse representation to recover the smooth patch, however, the Criminisi method traverses the entire source region to search for the most similar exemplar patch, which will cost much time. During all the images, the reduction rates of image 6 and image 7 are relatively large, this is because in these images, the vast majority of image patches are relatively smooth, and they are recovered based on sparse representation, instead of traversing the entire image to search for the exemplar patch, which can reduce a lot of running time. The reduction rate of the image 4 is relatively small, the reason is that in these images, most of the image patches contain rich texture, and thus the running time of two methods is almost the same.

5. **Conclusions.** In this paper, we propose a new image inpainting method for object removal based on patch local feature and sparse representation. We classify image patches according to their local variances, and use different methods to recover these patches. In this way, we cleverly use the sparse representation of image to solve the inpainting problem of removing objects from an image. The experimental results demonstrate the effectiveness of our method. It not only costs less running time, but also avoids introducing undesired objects in recovered images. Next, we will study how to classify the texture patches in more detail according to the local structure characteristics of the image patch so as to improve the inpainting effect.

Acknowledgment. The research is supported by the National Natural Science Foundation of China (No. 61272286), and Provincial Natural Science Foundation research project of Shaanxi (No. 2014JM8346). All of the authors would like to thank the anonymous referees for their valuable comments and suggestions.

REFERENCES

[1] K. Liu, J. Tan and B. Su, Exemplar-based image inpainting using structure Tesnor, *International Conference on Advanced Computer Science and Electronics Information (ICACSEI 2013)*, pp.619-623, 2013.

[2] C. Guillemot and O. Le Meur, Image inpainting: Overview and recent advances, *Signal Processing Magazine*, vol.31, no.1, pp.127-144, 2014.

[3] J. Li, M. Li and H. Fan, Image inpainting algorithm based on low-rank approximation and texture direction, *Mathematical Problems in Engineering*, pp.1-11, 2014.

[4] Y. Bahat, Y. Y. Schechner and M. Elad, Self-content-based audio inpainting, *Signal Processing*, vol.111, pp.61-72, 2015.

[5] P. D. Wagh and D. R. Patil, Text detection and removal from image using inpainting with smoothing, *International Conference on Pervasive Computing (ICPC)*, pp.1-4, 2015.

[6] X. Zhang, F. Ding, Z. Tang et al., Salt and pepper noise removal with image inpainting, *AEU-International Journal of Electronics and Communications*, vol.69, no.1, pp.307-313, 2015.

[7] R. Sharma and A. Agarwal, An innovative approach to show the hidden surface by using image inpainting technique, *Proc. of the 3rd International Conference on Frontiers of Intelligent Computing: Theory and Applications (FICTA)*, pp.395-403, 2015.

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

[9] P. Buyssens, M. Daisy, D. Tschumperle et al., Exemplar-based inpainting: Technical review and new heuristics for better geometric reconstructions, *IEEE Trans. Image Processing*, vol.24, no.6, pp.1809-1824, 2015.

[10] M. Bertalmio, G. Sapiro, V. Caselles et al., Image inpainting, *Proc. of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, pp.417-424, 2000.

[11] T. F. Chan and J. Shen, Mathematical models for local nontexture inpaintings, *SIAM Journal on Applied Mathematics*, vol.62, no.3, pp.1019-1043, 2002.

[12] T. F. Chan and J. Shen, Nontexture inpainting by curvature-driven diffusions, *Journal of Visual Communication and Image Representation*, vol.12, no.4, pp.436-449, 2001.

[13] A. A. Efros and T. K. Leung, Texture synthesis by non-parametric sampling, *Proc. of the 7th IEEE International Conference on Computer Vision*, vol.2, pp.1033-1038, 1999.

[14] Y. Liu and V. Caselles, Exemplar-based image inpainting using multiscale graph cuts, *IEEE Trans. Image Processing*, vol.22, no.5, pp.1699-1711, 2013.

[15] M. Ghayoumi and C. C. Lu, Improving exemplar based inpainting method with a fuzzy approach, *International Conference on Audio, Language and Image Processing*, pp.671-675, 2014.

[16] Z. Liang, G. Yang, X. Ding et al., An efficient forgery detection algorithm for object removal by exemplar-based image inpainting, *Journal of Visual Communication and Image Representation*, vol.30, pp.75-85, 2015.

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

[18] T. Guha, E. Nezhadarya and R. K. Ward, Sparse representation-based image quality assessment, *Signal Processing: Image Communication*, vol.29, no.10, pp.1138-1148, 2014.

[19] X. Y. Zhang and Q. H. He, Time-frequency audio feature extraction based on tensor representation of sparse coding, *Electronics Letters*, vol.51, no.2, pp.131-132, 2015.

[20] M. Aharon, M. Elad and A. Bruckstein, K-SVD: An algorithm for designing over-complete dictionaries for sparse representation, *IEEE Trans. Signal Processing*, vol.54, no.11, pp.4311-4322, 2006.

[21] M. Elad, J. L. Starck, P. Querre et al., Simultaneous cartoon and texture image inpainting using morphological component analysis (MCA), *Applied and Computational Harmonic Analysis*, vol.19, no.3, pp.340-358, 2005.

[22] J. L. Starck, Y. Moudden, J. Bobin et al., Morphological component analysis, *International Society for Optics and Photonics*, 2005.

[23] J. Bobin, J. L. Starck, J. M. Fadili et al., Morphological component analysis: An adaptive thresholding strategy, *IEEE Trans. Image Processing*, vol.16, no.11, pp.2675-2681, 2007.

[24] M. Elad and M. Aharon, Image denoising via learned dictionaries and sparse representation, *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol.1, pp.895-900, 2006.
[25] M. Elad and M. Aharon, Image denoising via sparse and redundant representations over learned dictionaries, *IEEE Trans. Image Processing*, vol.15, no.12, pp.3736-3745, 2006.
[26] Y. C. Pati, R. Rezaiifar and P. S. Krishnaprasad, Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition, *Conference Record of the 27th Asilomar Conference on Signals, Systems and Computers*, pp.40-44, 1993.
[27] Q. S. Lian and W. Zhang, Image super-resolution algorithms based on sparse representation of classified image patches, *Acta Electronica Sinica*, vol.40, no.5, pp.920-925, 2012.