Fast Directional Image Completion

Chih-Wei Fang and Jenn-Jier James Lien

Robotics Laboratory, Department of Computer Science and Information Engineering,
National Cheng Kung University, Tainan, Taiwan, R.O.C.
{nat, jjlien}@csie.ncku.edu.tw
http://robotics.csie.ncku.edu.tw

Abstract. We developed a fast image completion system using the multi-resolution approach to accelerate the convergence of the system. The down-sampling approach is for the texture-eigenspace training process based on the multi-level background region information. The up-sampling approach is for the image completion process to synthesize the replaced foreground region. To avoid the discontinuous texture structure, we developed the directional and non-directional image completions to reconstruct the global geometric structure and maintain local detailed features of the replaced foreground region in the lower- and higher-resolution levels, respectively. In addition, the Hessian matrix decision value (HMDV) is generated to decide the priority order and direction of the synthesized patch in the replaced region. To avoid the rim effect of the synthesized result, the border of each patch defined as O-shaped pattern is selected for matching comparison instead of using entire patch. Finally, additional texture refinement process is guaranteed to have high-resolution result.

Keywords: Texture Analysis, Texture Synthesis, Image Completion, Hessian Matrix, Eigenspace.

1 Introduction

Photographs sometimes include unwanted objects. After removing the unwanted foreground objects, holes will leave at that photograph. Although many existing image completion techniques can fill those holes, there still exists the discontinuity problem of the texture structure between the new fill-in foreground regions and the original background regions.

One major factor caused this kind of texture discontinuity is the priority of the synthesis (or fill-in) order for each hole. Since texture structure is the important clue to judge the completion performance in general appearance of a photo and edge is the most important component to construct the completion of the texture structure. Once the texture structure is damaged, the discontinuities of the edges become obvious. In order to synthesize (or reconstruct) complete texture structure, authors in [4], [5], [16], [19] proposed the texture synthesis approach starting along the damaged edges to fill the hole or the work in [3] divided image into structure components and texture components, and synthesized separately. Bertalmio et al. [2], [3] expand structure from boundary, so the structure continuity can be maintained particularly for the long and thinner removal region. The work in [13] segmented the image into several regions and synthesized each region individually. Sun et al. [16] manually drew the curve first, and then compared the matching patches along the curve. Moreover, the work in [5] is based on the inner product of the vertical direction of pixel gradient and normal vector of the border of the removal region, and then the result is multiplied by the proportion of the information that patch keeps in to find the priority. The purpose is to find the starting point of filling in the matching patch from searching exhaustively. But this approach is easy to be influenced by high-frequency components, such as noises or complicated texture components, where these high-frequency components easily cause higher synthesis order than the structure components. Therefore, it misleads the error of priority and destroys the integrality of structure.

The other factor is the size of the synthesis patch whether it is pixel-based patch [1], [17], [18] or patch-based patch [8], [9], [11], [12], [14], [15] (or exemplar-based), which has been used mostly. The pixel-based approach is slower and the synthesis result trends to fuzzy result. The patch-based approach is faster but remains obvious discontinuous flaws between neighboring patches. Efros and Freeman [8] used dynamic programming (DP) to find the minimum error for cutting the overlaps between two discontinuous neighboring patches. Drori et al. [6] mixed different scaling patches according to different frequencies of texture complexity. The synthesis result is good but the computational time is very slow.

To improve the computational time of the image completion process, Freeman et al. [11] and Liang et al. [14] modified the similarity measure between patches by comparing only the border pixels of the patches instead of comparing those of entire patches. In addition, comparing the pixels of the entire patch will result in not easy to connect between the patches It is because the number of pixels inside the patch are more than those at the border of the patch.

According to above existing problems, we develop a novel image completion system based on the patch-based multi-resolution approach. This approach can not only accelerate computational time and is capable of handling the large removed region, but also maintain both the texture structure and detailed features. We develop the directional and non-directional image completions to maintain the texture structure and detailed features of the replaced foreground region, respectively. Moreover, to solve the priority problem of the synthesis order and avoid affecting by noises and complicated texture components, Hessian matrix is employed to have the stable and correct priority of synthesis order.

Section 2 describes we develop the texture analysis module for analyzing and training input image. At Section 3, we apply Hessian matrix to decide the synthesis order and image completion methods: either using the directional image completion to propagate the structure or using the non-directional image completion to synthesize detailed features. Subsequently, texture refinement is used to recover the lost feature at the training process. Section 4 presents the current experimental results by evaluating time complexity of the training process and image completion, and analyzing our developed approaches. Finally, Section 5 discusses the conclusion.

2 Training Process Based on Background Region

This training process is similar to our previous work in [10]. Here we just briefly describe as following: An input image is given and is annotated as I_0, as show in

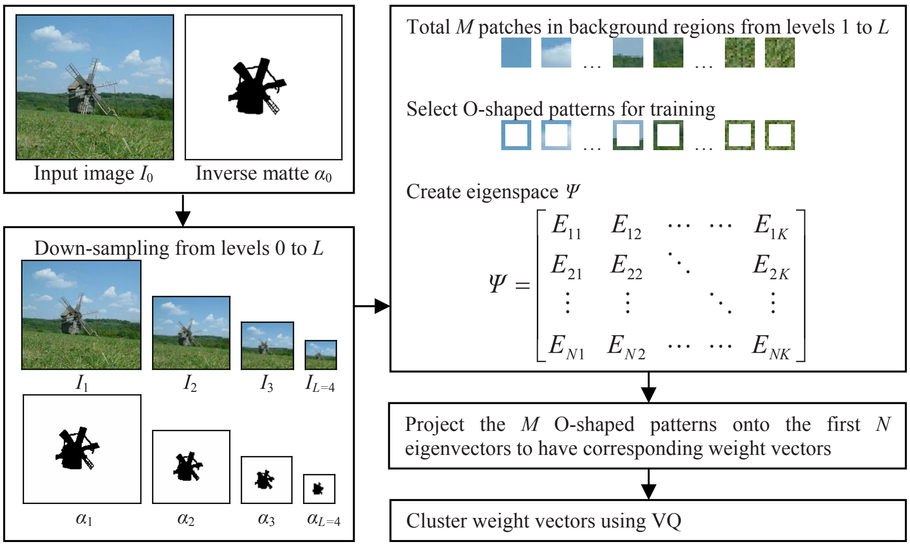

Input image I_0
Inverse matte α_0
Down-sampling from levels 0 to L
I_1
I_2
I_3
$I_{L=4}$
α_1
α_2
α_3
$\alpha_{L=4}$
Total M patches in background regions from levels 1 to L
Select O-shaped patterns for training
Create eigenspace Ψ
$\Psi = \begin{bmatrix} E_{11} & E_{12} & \cdots & \cdots & E_{1K} \\ E_{21} & E_{22} & \ddots & & E_{2K} \\ \vdots & \vdots & & \ddots & \vdots \\ E_{N1} & E_{N2} & \cdots & \cdots & E_{NK} \end{bmatrix}$
Project the M O-shaped patterns onto the first N eigenvectors to have corresponding weight vectors
Cluster weight vectors using VQ

Fig. 1. Flowchart of the texture training (or analysis) process

Fig.1. The mask image, which is manually labeled the replaced foreground region, is called the inverse matte, α_0, as show in Fig. 1. This matte is a binary image having the same size as input image. The pixels in the white regions, which are the known regions, are set to 1, while the pixels in the black regions, which are going to be removed and then synthesized based on the background information, are set to 0. In addition, the replaced regions can comprise many sub-regions, but they must contain the removable objects; they can exceed the boundaries of the removable objects, and they can be any shapes. However, too many or too large replaced regions will cause the quality of the synthesis result to become worse. Moreover, the known background regions serve as the source of the replaced regions.

For the training process, initially, we have input image I_0 and corresponding inverse matte α_0, and then we down-sample $\downarrow$ the original image I_0 L times to obtain different lower resolution images I_i and corresponding inverse matte α_i at level i, where i=1~L. The background regions corresponding to known regions having value 1 at the inverse matte are extracted to several patches to be the training data. In order to reduce the computational time during the similarity measure at the synthesis process, principal component analysis (PCA) is employed to reduce the dimensions of the training data, and vector quantization (VQ) is adopted for clustering the projection weight vectors in the eigenspace so as to reduce the comparison time. Fig. 1 illustrates the system flowchart of the training process extending from the input image to get the weight vector of each patch.

2.1 Multi-resolution Preprocessing from Levels 0 to *L*

We apply the multi-resolution approach [20] to our system for three purposes. The first purpose is to avoid the computational condition of being unable to converge and the worse initial result of synthesized image. The second purpose is to have more training patches with various scales [7] and texture features. At different levels, the patches have various properties. The patches at the lower-resolution images (ex. I_{L-1}~I_L) contain stronger structural information. The patches at the higher-resolution images (ex. I_0~I_1) have more detailed feature information. The third purpose is to reduce the computational time.

The multi-resolution approach down-sample ↓ the input image I_0 and corresponding inverse matte α_0 L times by factor 2 to have input image I_i and inverse matte α_i for ith level, where i=1~L. The background known region is denoted as B_i as:

$$B_i = I_i \alpha_i \quad , \quad i = 0 \sim L \tag{1}$$

and preserved for the training data. The foreground unknown region F_i is denoted as:

$$F_i = I_i \overline{\alpha_i} \quad , \quad i = 0 \sim L \tag{2}$$

and is going to be synthesized patch by patch by utilizing the information of the background B_i. In addition, the smallest image size at level L depends on the correlation between the patch size and the foreground region at level L. That is, the patch size needs to be big enough to cover most foreground region. More details will be discussed later.

2.2 Create Eigenspace Ψ Based on O-shaped Patterns from Levels 1 to L

During the training process, one *Wp*x*Hp*-pixel (width x height = 15x15-pixel) search window shifting pixel by pixel from the top-left corner to the right-bottom corner in the background regions for images from I_1 to I_L to extract training patch data. There are M total *Wp*x*Hp*-pixel patches. The reason we do not include the image I_0 at level 0 is because of following reasons based on our empirical experiences: (1) It is unnecessary to use entire patches of image I_0 at level 0, i.e., 100000~300000 patches per 320x240-pixel, because it will include many unnecessary patches and require a large amount of template matching operations. In addition, the training time of the vector quantization (VQ) will increase exponentially following the increasing number of clusters. It also increases the probability of mismatching the incorrect patch. (2) Image I_0 at level 0 contains more noises, which affect the result of the PCA process, and will increase the mismatch probability. When we practically don't include the image I_0 during the training process, the computational time reduces from 15 seconds to 2 seconds for 320x240-pixel image size. But this approach causes the synthesized result losing the detailed feature information and decreases the high-resolution quality of image completion result. So an additional process (Section 3.2) for the texture refinement at the 0th level is required in order to make the image completion result have the same high resolution as the input image.

Furthermore, including whole patch elements to the training data for further matching comparison may result in a discontinuous structure of the patch and will certainly increase the training time. In addition, during the synthesis process, the similarity

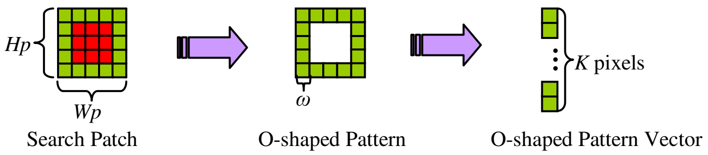


Fig. 2. Acquire the four borders with thickness ω (ω=2) pixels for each search patch. Wp (Wp=15) is the width of the patch, and Hp (Hp=15) is the height of the patch. There are K pixels in each O-shaped pattern, where $K = 2\omega(Wp+Hp) - 4\omega^2$.

$$\begin{bmatrix} P_{11} & P_{12} & \cdots & \cdots & P_{1M} \\ P_{21} & P_{22} & \cdots & \cdots & P_{2M} \\ \vdots & \vdots & \ddots & & \vdots \\ \vdots & \vdots & & \ddots & \vdots \\ P_{K1} & _{K2} & \cdots & \cdots & P_{KM} \end{bmatrix} \overset{\text{PCA}}{\Rightarrow} \begin{bmatrix} E_{11} & E_{12} & \cdots & \cdots & E_{1K} \\ E_{21} & E_{22} & \cdots & \cdots & E_{2K} \\ \vdots & \vdots & \ddots & & \vdots \\ \vdots & \vdots & & \ddots & \vdots \\ E_{K1} & E_{K2} & \cdots & \cdots & E_{KK} \end{bmatrix} \overset{98\%}{\Rightarrow} \Psi = \begin{bmatrix} E_{11} & E_{12} & \cdots & \cdots & E_{1K} \\ E_{21} & E_{22} & \ddots & & E_{2K} \\ \vdots & \vdots & & \ddots & \vdots \\ E_{N1} & E_{N2} & \cdots & \cdots & E_{NK} \end{bmatrix}$$

Fig. 3. Total M O-shaped patterns can be obtained from levels L-1 to 0. Each pattern vector has K elements (see Fig. 2). During training process, PCA is used to transform the original $K{\times}M$ matrix of all training pattern vectors to a $N{\times}K$ eigenvector matrix in the eigenspace, where the first N eigenvectors corresponding to 98% energy of total eigenvalues and $N < K << M$.

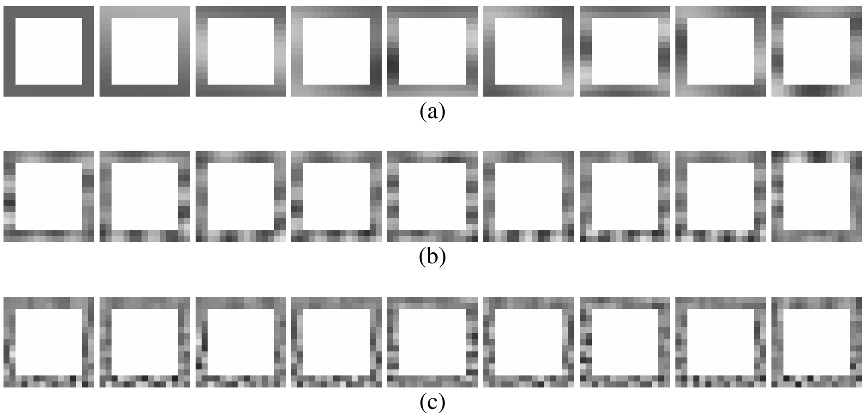


Fig. 4. (a) The first several eigenvectors of the O-shaped patterns control the global geometrical structure of the photographs. (b) The middle eigenvectors of the O-shaped patterns control the local detailed features. (c) The last few eigenvectors of the O-shaped patterns control some noises.

measure by considering the whole contents of the patch will generally produce unsatisfactory results, and it may cause the rim effect to become distinct. Thus, this work adopts the O-shaped pattern instead of whole patch for the training data, as shown in Fig. 2.

PCA is applied to the entire O-shaped training patterns to obtain the eigenspace Ψ, as shown in Fig. 3. Two important properties of PCA are employed to have the best performance for the image completion result: (1) PCA process can reduce the dimensions of the data representation from K dimensions to N dimensions, where $N < K$, without losing the significant characteristics of the original data, as shown in Fig. 3. (2) Recombine the features of the O-shaped pattern. because the PCA process and sorting the eigenvalues with corresponding eigenvectors, we found that the first several eigenvectors, as shown in Fig. 4, control the global geometrical structure of the photographs, while the middle eigenvectors control the local detailed features, and some noises are controlled by the last few eigenvectors. This study uses only the first N eigenvectors, whose corresponding eigenvalues occupy 98% of total eigenvalues, for comparison purpose to identify the suitable matching patch.

2.3 Create Corresponding Weight Vectors by Projecting O-shaped Patterns onto Eigenspace Ψ

Each K-dimensional O-shaped pattern vector P ($P=[P_1 \ldots P_K]^T$) is projected onto the N-dimensional eigenspace, which consists of N eigenvectors ($E_{1i} \ldots E_{Ni}$, where i=1~K) as:

$$\begin{bmatrix} E_{11} & E_{12} & \cdots & \cdots & E_{1K} \\ E_{21} & E_{22} & \ddots & & E_{2K} \\ \vdots & \vdots & & \ddots & \vdots \\ E_{N1} & E_{N2} & \cdots & \cdots & E_{NK} \end{bmatrix} \begin{bmatrix} P_1 \\ P_2 \\ \vdots \\ \vdots \\ P_K \end{bmatrix} = \begin{bmatrix} W_1 \\ W_2 \\ \vdots \\ W_N \end{bmatrix} \tag{3}$$

So the corresponding N-dimensional weight vector W ($W=[W_1 \ldots W_N]^T$) for each pattern vector can be obtained to represent the original patch. Therefore, there are a total of M N-dimensional weight vectors in the training database.

2.4 Cluster Weight Vectors Using Vector Quantization

For the similarity measure during the synthesis process, it is extreme time consuming to compare the projection weight vector of each unknown pattern with those of all training patterns. To reduce the comparison time, the vector quantization (VQ) is applied to cluster entire training patterns into several clusters, ex. c (c=32) clusters. Thus, the comparison time is reduced from $O(M)$ to $O(\sqrt{M})$. The tree-structured vector quantization (TSVQ) [17] has the potential to reduce computation costs further. However, there is a risk in applying this technique in the present cases. For example, any misclassification in the parent node will affect the final classification result. Therefore, in our work, initially the closest cluster is directly identified according to its mean vector, and this cluster is then searched exhaustively to locate the best matching vector.

3 Directional and Non-directional Image Completion

For the synthesis process of the image completion, it starts from image I_L at level L to image I_0 at level 0, as shown in Fig. 5. Since the patches at the lower-resolution images contain more texture structure information, and the patches at the higher-resolution images contain more detailed feature information, so different resolution images will apply different synthesis processes. In addition, the pixel gray values at the lower level i can serve as the initial pixel gray values at the higher level i-1 for the corresponding foreground region.

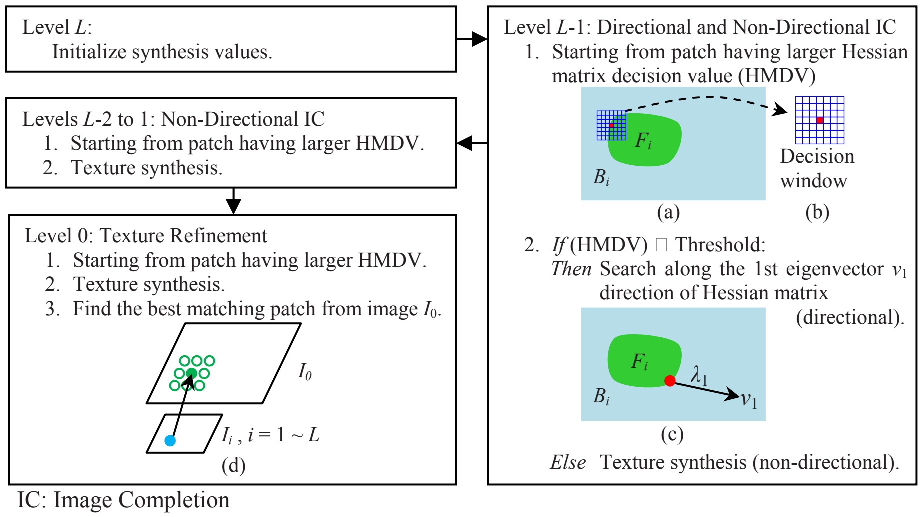

Level L:
Initialize synthesis values.
Levels L-2 to 1: Non-Directional IC
1. Starting from patch having larger HMDV.
2. Texture synthesis.
Level 0: Texture Refinement
1. Starting from patch having larger HMDV.
2. Texture synthesis.
3. Find the best matching patch from image I0.
I0
Ii , i = 1 ~ L
(d)
Level L-1: Directional and Non-Directional IC
1. Starting from patch having larger Hessian matrix decision value (HMDV)
Fi
Bi
Decision window
(a)
(b)
2. If (HMDV) □ Threshold:
Then Search along the 1st eigenvector v1 direction of Hessian matrix (directional).
λ1
v1
(c)
Else Texture synthesis (non-directional).
IC: Image Completion

Fig. 5. Flowchart of the image completion from level L to level 0. (a) F_i is the removal foreground region and B_i is the background region at the level i. Each 2x2 Hessian matrix G is constructed based on a 7x7-pixel decision window. (b) The decision window goes along the boundary of the foreground region to decide the priority of the synthesis order. (c). Search along the direction of the 1st eigenvector v_1. (d) The matching patch is found at the blue point, but it doesn't contain high-resolution content. Therefore, the blue point is mapping to the green point at level 0, and then we search the best matching patch near the green point.

3.1 Level *L*: Initialize Synthesis Values

For image I_L at level L, most pixels of the foreground region are located inside the O-shape pattern. Then the search window (or patch) will scan from the top-left corner to the bottom-right corner at the foreground region. Usually the search window has only one for image I_L. Considering the missing data problem for the PCA process, the O-shape pattern of each search window won't project on the eigenspace to produce the corresponding weight vector. Instead, the O-shaped pattern of each search window is directly compared with the pixels' gray values of each O-shaped pattern over entire training patches. The similarity measure uses the Euclidean distance. Then the patch of the best matching pattern in the training patches is directly pasted onto the corresponding location of the foreground region. Here, the computational time for the comparison process at level *L* is very fast because the number of search windows usually has only one. So the projection process of the PCA does not apply here.

After synthesizing the foreground regions, the composite image C_i at level *i* is defined as:

$$C_i = I_i \alpha_i + I'_i \overline{\alpha}_i \quad , \quad i = 0 \sim L \tag{4}$$

where I_i is the original image and I_i' is the synthesized image. Subsequently, the up-sampling ↑ process is applied to the composite image C_i from level *i* to level *i*-1. Then following process is applied to update the image $I_{i\text{-}1}$ at level *i*-1.

$$I_{i-1} = I_{i-1}\alpha_{i-1} + (C_i \uparrow)\overline{\alpha}_{i-1} \quad , \quad i = 0 \sim L-1 \tag{5}$$

That is, each pixel gray value of the foreground region at image $I_{i\text{-}1}$ can be assigned an initial value, which is obtained from the composite image C_i at level *i*.

3.2 Levels *L*-1 to 0: Hessian Matrix Decision for Directional and Non-Directional Image Completion

Synthesizing the foreground regions is necessary to maintain the continuity of the texture structure extending from the background regions. Thus, the priority of synthesis order becomes very important. We apply the gradient-directional property of the Hessian matrix to play the role of the priority decision maker. We designed that each 2x2 Hessian matrix *G* is constructed by a 7x7-pixel decision window, *W*, as (also see Fig. 5(b)):

$$G = \begin{bmatrix} \sum_{(x,y)\in W} \frac{\partial^2 I(x,y)}{\partial x^2} & \sum_{(x,y)\in W} \frac{\partial^2 I(x,y)}{\partial x \partial y} \\ \sum_{(x,y)\in W} \frac{\partial^2 I(x,y)}{\partial x \partial y} & \sum_{(x,y)\in W} \frac{\partial^2 I(x,y)}{\partial y^2} \end{bmatrix} \tag{6}$$

where *I*(*x*,*y*) is the gray value at location (*x*,*y*) belonging to the decision window *W*. The centroid of the decision window goes along the boundary of the foreground region and both eigenvalues of the Hessian matrix are used to decide the priority of the synthesis order, as shown in Fig. 5(a). That is, assuming the eigenvalues $\lambda_1 \geqq \lambda_2$, the Hessian matrix decision value (HMDV), *V*, is defined as:

$$V = \frac{\lambda_1 + \varepsilon}{\lambda_2 + \varepsilon} \tag{7}$$

where ε is a very small value (ε=0.001) to avoid the denominator term becoming zero. The HMDV will exist three conditions: (1) Higher HMDV value *V* (*V* >> 1.0) means that the decision window is directional and exists stronger edge. (2) If the HMDV

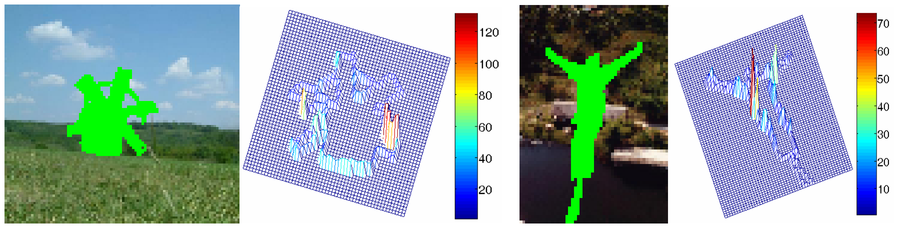


Fig. 6. Above two groups of images: Left image is the level L-1 image. Right image is the distribution of HMDV. The red peak is the starting point to fill in the matching patch.

value V is close to or equal to 1, there exist two conditions: (2.1) If both λ_1 and λ_2 have higher values, then the decision window contains more detailed features or high-frequency noises. (2.2) If both λ_1 and λ_2 have lower values, the patch of decision window is smooth. Therefore, when the HMDV value V of the decision window is bigger than or equal to the predefined threshold value, the search patch of the corresponding centroid has the higher priority of synthesis order and higher HMDV value has higher priority. Then the directional image completion is applied to this search patch as shown in Fig. 6. Conversely, if the HMDV value V of the decision window is smaller than the predefined threshold, it will be synthesized as non-directional image completion after directional image completion.

Level *L*-1: Directional Image Completion for Texture Structure.
To synthesize the foreground region at level L-1, initially the centroid of the decision window goes along the boundary of the foreground region and the HMDV value V of each decision window will be recorded. If the HMDV value V of the decision window is bigger than or equal to the threshold, it means that the corresponding search patch exists stronger edge components. After sorting the HMDV values, which are bigger than or equal to the threshold, the synthesis process will start from the search patch, whose corresponding decision window has the maximum HMDV value. Then the search patch will scan along the direction of the eigenvector v_1 corresponding to eigenvalue λ_1 or opposite direction depending on the location of background region, as shown in Fig. 5(c). The direction of eigenvector v_1 is the tangent direction of edge. Subsequently, the patch of the best matching pattern in the background region will be directly pasted onto the location of the search patch. Again, the centroid of the decision window will go along the inner boundary of the replaced foreground patch to calculate the HMDV values. Thus, the same threshold, sorting (if necessary), matching (or comparison), pasting and HMDV calculating processes compute iteratively until none of any HMDV values are bigger than or equal to the threshold or the patches of entire foreground region are updated.

Since this kind of comparison process exists only few training patches for each search patch, so the computational time of the similarity measure won't take too much. Thus, the similarity measure is based on the gray values instead of the projection weights. In addition, above procedure for structure synthesis can be defined as the directional image completion. For remaining decision windows having the smaller HMDV values than the threshold, the texture synthesis procedure [10], [14] will be defined as the non-directional image completion and will be described at next section.

Levels L-1 to 1: Non-Directional Image Completion for Detailed Features

For image I_{L-1} at level L-1, after the directional image completion process, the remaining search patch located at the foreground boundary will be synthesized following the order from higher HMDV in the foreground region. Initially, the O-shaped pattern of each search patch is projected onto the eigenspace Ψ to obtain the corresponding weight vector. Based on the similarity measure of Euclidean distance between the weight vector of this search pattern and those of the cluster centers, this search pattern will be classified to the nearest cluster. Then this search pattern will compare with all patterns within the same cluster to find the best matching pattern. The patch corresponding to the best matching pattern will be directly pasted onto the location of the search patch. This texture synthesis process will iterative process until all remaining search patches are updated.

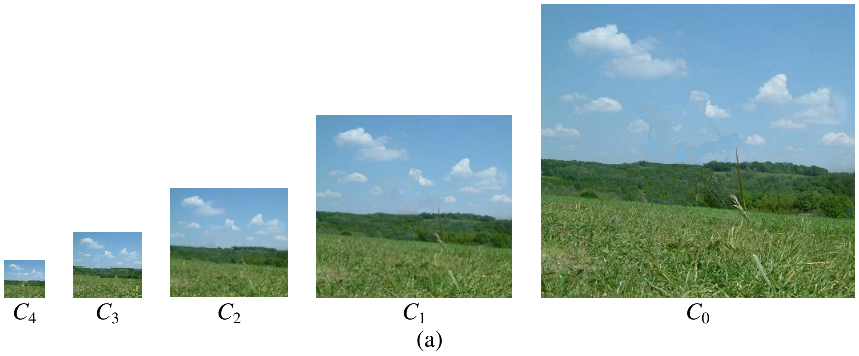
Since the directional image completion is able to construct the texture structure for image I_{L-1} at level L-1, so we can concentrate on the enhancement of detailed features for remaining texture synthesis process from level L-2 to level 1. The similar procedure to image I_{L-1} at level L-1 is applied except that the search patch does not need to scan along the eigenvector direction. The priority of synthesis order still relies on the HMDV value for each decision window. And the similarity measure is the same as the procedure of the non-directional image completion in the eigenspace. Thus, the images from level L-2 to 1 must use HMDV to determine the order of image completion and avoid destroying the already existing edges at level L-1.

Level 0: Texture Refinement

Because of above-mentioned considerations in the training and synthesis processes, the patches at the highest-resolution level 0 are not included in the training database, thus the most detailed texture information will lose for the final synthesized result. Therefore, at the level 0, when the matching patch is found from the training database, we do not paste the matching patch directly. Instead, we search more detailed features of the patch neighboring the position, which is the position of the matching patch corresponding to the highest level, on the highest level, as shown in Fig. 5(d). This texture refinement process makes the removal region (foreground region) and the reserved region (background region) have the consistent resolution.

4 Experimental Results

In acquiring the experimental statistics presented below, each process was performed ten times and then the average time was calculated. The experiments were performed on a personal computer having an Intel Core 2 Duo E6300 (1.86 GHz) processor. The computational times from the training process to the image completion in various kinds of images are as follows: Fig. 7 and Fig. 8. The information of Fig. 7 and Fig. 8 in Table 1, and then it shows the processing time of other existing method. In addition, the result of exemplar-based image inpainting in [5] is unable to converge at the case of larger removal region. When the removal region is not narrow and long, the



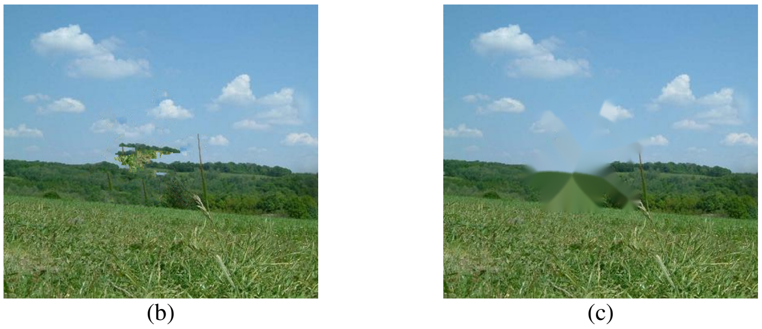


Fig. 7. The image size is 392 x 364 pixels. The ratio of removal region is 7.2%. (a) Show our reconstructed image C_i from the lowest level L to the original level 0, i=0~4. C_i does up-sampling ↑, and then serves as initial value of $I_{i\text{-}1}$ for searching the matching patch to fill the removed (or replaced) region of $C_{i\text{-}1}$. (b) The result of exemplar-based image inpainting by [5]. (c) The result of image inpainting by [2].

Table 1. Show the information of image size and the ratio of removal region. Then compare the processing time of other existing method. Units of time in seconds (s).

		Image Size (pixel)	Ratio of removal region	Our metho d	Exemplar-based image inpainting by [5]	Image Inpainting by [2]
Windmill	Fig. 7	392 × 364	7.2%	11	104	2
Slope	Fig. 8(a)	213 × 284	8.7%	4	46	1
Diving	Fig. 8(b)	206 × 308	12.6%	2	38	1
Mother	Fig. 8(c)	538 × 403	25.5%	57	724	6
Wall	Fig. 8(d)	400 × 400	28.4%	35	420	4
Mountain	Fig. 8(e)	392 × 294	5.3%	7	60	1

result of image inpainting in [2], which expands structure from boundary, tends to the blurred result without obvious structure and edge.

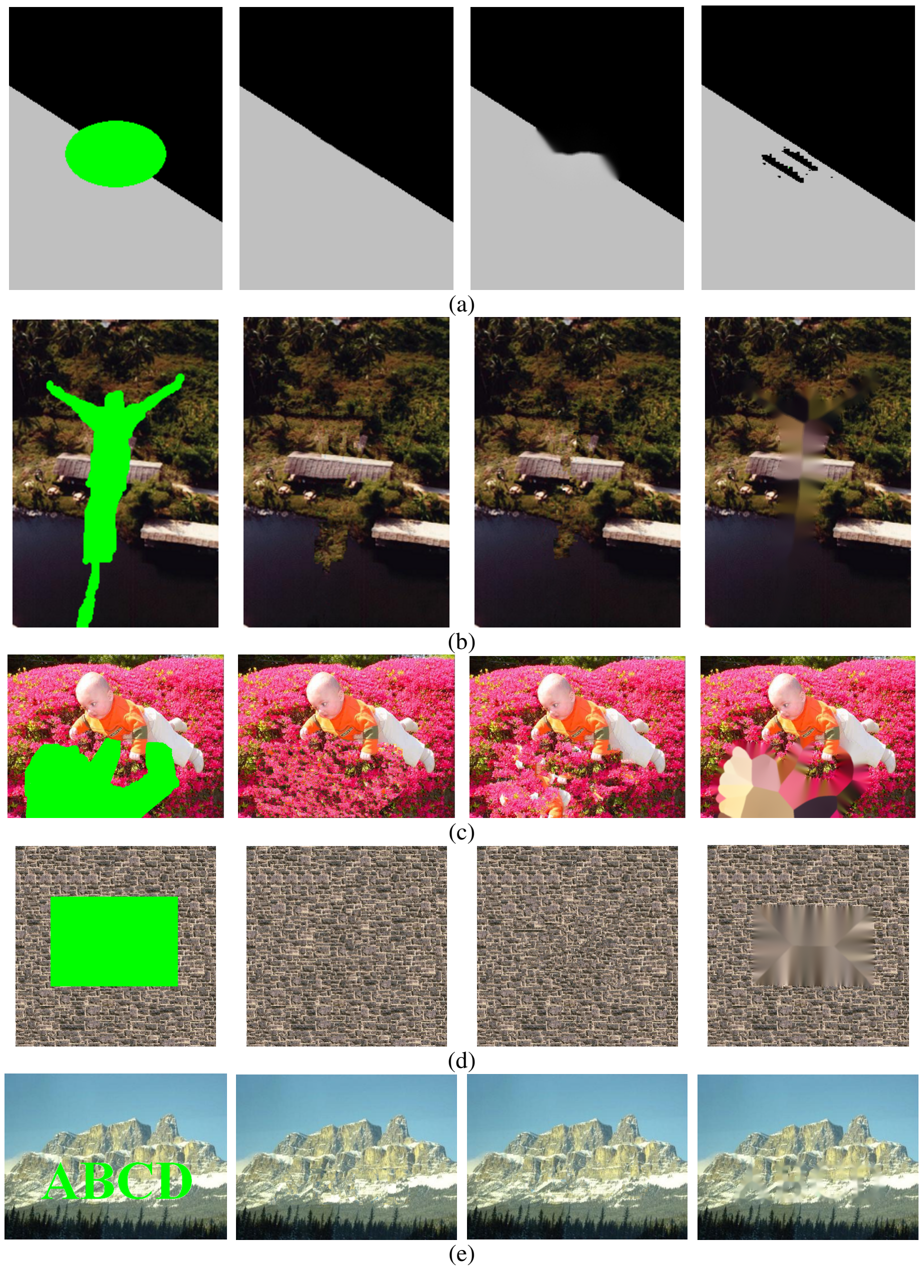


Fig. 8. The images of first column are input images. The image of second column shows the results of our method. The images of third column are the results of exemplar-based image inpainting by [5]. The images of fourth column are the results of image inpainting by [2].

5 Conclusions

The multi-resolution approach is applied to image completion. The down-sampling approach is used for the analysis process, such as compiling the training data, and the up-sampling approach is used for the synthesis process, such as initial values of level i-1. Therefore, this approach enables the system to handle the large removed region and converge quickly. We only take the border of patch (O-shaped pattern) for training. In addition, the patches at the highest-resolution level are not included in the training data in order to speed up the computational time of the training process, and it reduces noise impact on PCA and improves the result of matching patch. Above training process reduces further the time of comparison and searching patch.

Subsequently, Hessian matrix is used for the decision of the synthesis order, and it is more stable than the existing methods of using the differentiation [5] in the patch with more noise or detailed feature in patch. During the synthesis process, the developed HMDV is applied for the decision of synthesis order and direction in order to propagate the structure continuity from the background region. For the directional image complete, for each higher HMDV decision window, we search the matching patch along the direction of the eigenvector of Hessian matrix. This directional process can decrease the time of exhaustive search and make a better structure continuity between background scene and foreground replaced region. Finally, we use texture refinement for resuming the lost detailed features of the training process.

References

1. Ashikhmin, M.: Synthesizing Natural Textures. ACM Symposium Interactive 3D Graphics, 217–226 (2001)
2. Bertalmio, M., Sapiro, G., Ballester, C., Caselles, V.: Image Inpainting. ACM SIGGRAPH, 417–424 (2000)
3. Bertalmio, M., Vese, L., Sapiro, G., Osher, S.: Simultaneous Structure and Texture Image Inpainting. IEEE Trans. on Image Processing 12(8), 882–889 (2003)
4. Chan, T., Shen, J.: Non-Texture Inpainting by Curvature-Driven Diffusions (CDD). Jour. of Visual Communication and Image Representation 12(4), 436–449 (2001)
5. Criminisi, A., Perez, P., Toyama, K.: Region Filling and Object Removal by Exemplar-Based Image Inpainting. IEEE Trans. on Image Processing 13(9), 1200–1212 (2004)
6. Drori, I., Cohen-Or, D., Yeshurun, H.: Fragment-Based Image Completion. ACM SIGGRAPH, 303–312 (2003)
7. De Bonet, J.S.: Multiresolution Sampling Procedure for Analysis and Synthesis of Texture Images. ACM SIGGRAPH, 361–368 (1997)
8. Efros, A.A., Freeman, W.T.: Image Quilting for Texture Synthesis and Transfer. ACM SIGGRAPH, 341–346 (2001)
9. Efros, A.A., Leung, T.K.: Texture Synthesis by Non-parametric Sampling. International Conf. on Computer Vision, 1033–1038 (1999)
10. Fang, C.-W., Lien, J.-J.: Fast Image Replacement Using Multi-resolution Approach. Asian Conference on Computer Vision, 509–520 (2006)
11. Freeman, W.T., Jones, T.R., Pasztor, E.C.: Example-Based Super-Resolution. IEEE Computer Graphics and Applications 22(2), 56–65 (2002)

12. Igehy, H., Pereira, L.: Image Replacement through Texture Synthesis. IEEE International Conf. on Image Processing 3, 186–189 (1997)
13. Jia, J., Tang, C.K.: Image Repairing: Robust Image Synthesis by Adaptive ND Tensor Voting. IEEE Conf. on Computer Vision and Pattern Recognition 1, 643–650 (2003)
14. Liang, L., Liu, C., Xu, Y., Guo, B., Shum, H.-Y.: Real-Time Texture Synthesis using Patch-Based Sampling. ACM Trans. on Graphics 20(3), 127–150 (2001)
15. Liu, Y., Lin, W.-C., Hays, J.: Near-Regular Texture Analysis and Manipulation. ACM SIGGRAPH, 368–376 (2004)
16. Sun, J., Yuan, L., Jia, J., Shum, H.-Y.: Image Completion with Structure Propagation. ACM SIGGRAPH, 861–868 (2005)
17. Wei, L.-Y., Levoy, M.: Fast Texture Synthesis using Tree-structured Vector Quantization. ACM SIGGRAPH, 479–488 (2000)
18. Wexler, Y., Shechtman, E., Irani, M.: Space-Time Video Completion. IEEE Conf. on Computer Vision and Pattern Recognition 1, 120–127 (2004)
19. Wu, Q., Yu, Y.: Feature Matching and Deformation for Texture Synthesis. ACM SIGGRAPH, 362–365 (2004)
20. Yamauchi, H., Haber, J., Seidel, H.-P.: Image Restoration using Multiresolution Texture Synthesis and Image Inpainting. Computer Graphics International, 120–125 (2003)