Image in-painting based FMM algorithm by edge prediction using gradient matrix

Qian Fan[a,b,*] , Lifeng Zhang[a] ,Xuelong Hu[b]

[a] Department of Electrical Engineering and Electronics, Kyushu Institute of Technology, Fukuoka, Japan
[b] School of Information Engineering, Yangzhou University, Yangzhou, China
*cnzshakka@gmail.com

Abstract

In this paper, we propose an improved image in-painting method based on Fast Matching Method (FMM) algorithm. The traditional approach speeds less time but it cannot contribute an optimal edge result. To overcome this disadvantage and improve the edge effect. First we use gradient matrix to select less but more significant pixels to join into the gray value calculation. Secondly we use an edge prediction method to predict the edge in the in-painting region and reset the in-painting sequence. Furthermore, this procedure also had an advantage in in-painting the image which had a large destroyed region. Therefore, our improved method contributes an obvious edge for in-painting procedure than the traditional method.

Keywords: Image in-painting, FMM algorithm, Gradient matrix, Edge prediction

1.Introduction

For many reasons, digital photos and film videos may be destroyed with several regions losing their pixels. These destructions may destroy the integrity of the data. Then digital image in-painting technology came out to solve this problem. By using the remaining information and one algorithm or one process, we can repair these pixels and it gives a good visual effect.

The most fundamental in-painting approach is the diffusion based approaches [(1-5)], in which the missing region is filled by diffusing the image information from the known region into the missing region at the pixel level. These algorithms are well founded on the theories of partial differential equation (PDE) and variational method. Bertalmio et al. [(1)] filled in holes by continuously propagating the isophote (i.e., lines of equal gray values) into the missing region. They further introduced the Navier-Strokes equation in fluid dynamics into the task of in-painting [(2)]. In [(3)], Chan and Shen recover the missing information by using a variational framework based on total variation (TV) model. Then a curvature-driven diffusion equation was proposed to realize the connectivity principle which does not hold well in the TV model [(6)]. A joint interpolation of isophote directions and gray-levels was also designed to incorporate the principle of continuity in a variational framework [(7)]. Recently, image statistics which learns from the natural images are applied to the task of image in-painting [(8-10)]. The diffusion-based in-painting algorithms have achieved convincingly excellent results for filling the non-textured or relatively smaller missing region. However, they tend to introduce the smooth effect in the textured region and larger missing region.

The second category approaches are the examplar-based in-painting algorithm. These approaches propagate the image information from the known region into the missing region at the patch level. This idea stems from the texture synthesis technique proposed in [(11)], in which the texture is synthesized by sampling the most suitable patch from the known region. However, natural images are composed of structures and textures, the structures always constitute one image's primal sketches (e.g., the edges, corners, etc.) and the textures are always represented the image regions with homogenous patterns or feature statistics (including the flat patterns). Due to Pure texture synthesis technique cannot clearly handle the missing region with composite textures and structures, Bertalmio et al. [(12)] proposed to firstly divided the image into structure and texture layers and then in-paint them by different method, using diffusion-based method to in-paint the structure layer and using texture synthesis technique to in-paint texture layer. It overcomes the smooth effect disadvantage brought from the diffusion-based in-painting

DOI: 10.12792/icisip2014.048

algorithm; however, recovering the larger missing structures also is very hard to achieve. Criminisi et al. [13] designed an examplar-based in-painting algorithm by propagating the known patches (i.e., examplars) into the missing patches gradually. To handle the missing region with composite textures and structures, patch priority is defined to encourage the filling-in of patches on the structure. Wu [14] introduced the concept of cross-isophotes to the examplar-based in-painting algorithm, in which he designed a cross-isophotes patch priority term based on the analysis of anisotropic diffusion. Wong [15] improved the examplar-based in-painting algorithm by proposing a nonlocal means approach. The image patch is in-painted by not the most suitable patch but a set of candidate patches in the known region. Compared with diffusion-based in-painting algorithms, the examplar-based in-painting algorithms are more suitable for in-painting the large missing region.

FMM algorithm focuses on the diffusion-based in-painting approach. This algorithm firstly proposed by Telea, it has a high speed in processing and has a similar result with other method [16]. Yang adds the Anisotropic diffusion theory to it and have a good result in large region in-painting (17). In[18] ,the author adds a pixel selection processing to select significant pixels into the gray calculation.

This paper is organized as follows. In section 2, it tells the development of FMM algorithm in the recent years, In section 3, it tells the improvement of the new method. the paper[18] can make the edge become clearer, but sometimes it looks not natural, so here the proposed method adds a edge prediction processing to first predict the edge in the destroyed region and then inpainting the image. The experiment and result is shown in section 4 and finally we conclude this work in section 5.

2 .The current development of the FMM algorithm

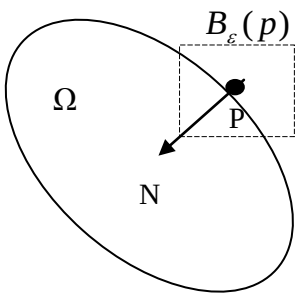


Fig .1.image in-paint theory

In the theory of the FMM algorithm, we regard the Ω as the region need to be in-painted, regard as the edge of the in-paint region, divide the pixels into 3 sets: Band (the pixels on the edge of region), Known(the grey value we've already known), Inside(grey value unknown pixels). By using a time function $T(x, y)$ to determine the sequence of the in-paint, then in-paint the region from the edge to the inside.

In-paint process: the p is the pixel needed to be in-painted, we select a window $B(p)$, use the pixel q in the window to calculate the gray value of pixel p

$$I_q(\mathrm{p}) = \mathrm{I}(\mathrm{q}) + \nabla \mathrm{I}(\mathrm{q}) * (\mathrm{p} - \mathrm{q}) \tag{1}$$

To compute the grey value of p, $I(q)$ means the gradient of q, for different q, we use a weight function w(p, q)to weight the influence between p and q, and finally weighted average it, so the equation changed to be.

$$I(\mathrm{p}) = \frac{\sum_{q \in B_\varepsilon(\mathrm{p})} w(\mathrm{p},\mathrm{q})(\mathrm{I}(\mathrm{q}) + \nabla \mathrm{I}(\mathrm{q})(\mathrm{p} - \mathrm{q}))}{\sum_{q \in B_\varepsilon(\mathrm{p})} w(\mathrm{p},\mathrm{q})} \tag{2}$$

for the weight function w(p, q),it has three indexes

$$w(p,q) = dir(p,q) * dst(p,q) * lev(p,q) \tag{3}$$

where, the three index are

$$\begin{aligned} dir(\mathrm{p},\mathrm{q}) &= \frac{p-q}{\|p-q\|} \bullet N(\mathrm{p}) \\ \mathrm{dst}(\mathrm{p},\mathrm{q}) &= \frac{d_0}{\|p-q\|^2} \\ lev(\mathrm{p},\mathrm{q}) &= \frac{T_0}{1 + |T(\mathrm{p}) - \mathrm{T}(\mathrm{q})|} \end{aligned} \tag{4}$$

These three indexes represent the 3 points of the known pixels have influence to the in-painting. *dir* index shows the direction influence for the in-paint, it means if the pixel *q*'s direction is close to the pixel *p*'s direction, the more influence it will have. *dst* index shows the distance influence for the in-paint, it means if the pixel *q* is near the pixel *p*, the more influence it will have. *lev* index shows the time influence for the in-paint, if the pixel *q* is close to the known information, the more influence it will have.

From the sets of equation, we can see that at last the three index have the same importance in it, but actually in the in-painting, the pixels which have the same texture direction with the destroyed pixel will do more contribute to the in-painting. So, if we compute under this method, the pixels which are near the destroyed pixel but with different texture direction will regard have much useful information, in fact, this kind pixels always are noise pixels. So it will make the edge unclear.

The most important advantage of this method is that it can spend very little time and achieves a good result. But when in-paint one pixel, this method use all the neighbor information, so it may add any noise into the in-paint pixel. And finally we can see that if the width of the in-paint area is larger than 20 pixels, the result will be unclear, especially in the isolux line and edge tangent place.

3 .Use gradient matrix to improve the FMM algorithm

As the disadvantage of the old FMM method, we know that the FMM algorithm is weak in maintaining the edge of the in–painting image. So we consider a new method to remain the fast advantage and give a good edge remaining result at the same time. We use the gradient information of the remaining pixels to make the in-painting process on the texture direction and contribute a better result.

3.1 Extract gradient matrix from the image

First we should compute the gradient amplitude and direction of the all the Known pixels. In order to reduce the calculation time, the extraction of the gradient matrix is just focused on the in-painting area and a neighborhood. For Known pixels:

$$G_{\mathrm{a}}(x,y)=\sqrt{(I(x+1,y)-I(x-1,y))^2+(I(x,y+1)-I(x,y+1))^2}$$
$$G_{\mathrm{d}}(x,y)=\tan^{-1}((I(x+1,y)-I(x-1,y))/(I(x,y+1)-I(x,y+1))) \quad (5)$$

For Band pixels, they are indispensable in the computing, but one hand of band pixels may be the inside pixels, so the grey value we don't know and we cannot get the Band pixels' gradient, then we need to change the equation, use the band pixel itself replace the inside pixel.

Finally we get two matrices, one matrix $G_{\mathrm{a}}(x,y)$ represents the gradient amplitude of every pixel, and the other matrix $G_{\mathrm{d}}(x,y)$ represents the gradient direction of every pixel.

With the gradient matrix and an edge detection algorithm (here we use morphologic edge detection algorithm, because it can give a clear and thin result) then we can get several breaking fields between the known region and the in-paint region.

3.2 Edge prediction

After we get the breaking field of the detected edge, then we should use the gradient information of the breaking field to predict the edge of the in-painting region.

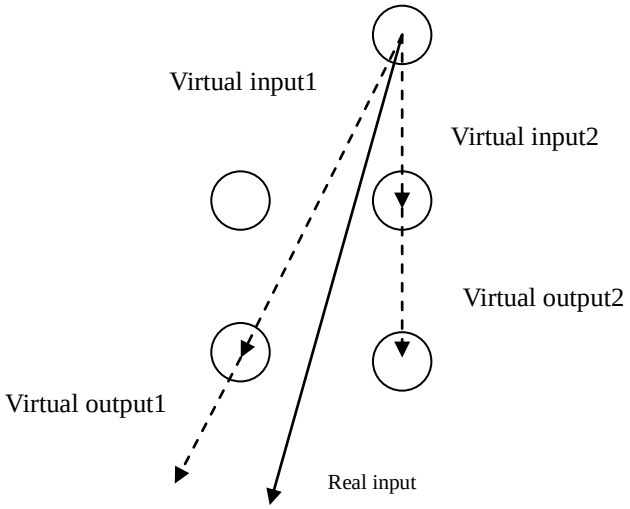


Fig 2.The theory of edge prediction

Figure 2 introduces the math model of the edge prediction. Every pixel's gradient contains amplitude and direction, here we call it the gradient vector. Every gradient vector can equals to effect on the neighborhood in 8 directions. We regard these gradient vectors as the input vector to the next layer pixels and the output vector from the last layer pixels.

For these vectors get by gradient calculation, we call them the real vector. Their direction value is continuous, so it's hard for us to see the influence with the neighbor pixels obvious. But after discretization, every real vector can be divided into several virtual vectors, then the influence made by the real vector can be regard as the influence made by different virtual vectors to the corresponding pixels, as the figure 2 shows us. By the transform processing from real vectors to virtual vectors we can clearly see the influence made by one pixel to the neighbor pixels, as the equation (6) shows us:

$$Riv=\sum_{i=1}^{N}Viv(i)$$
$$Rov=\sum_{i=1}^{N}Vov(i) \quad (6)$$

where, Riv、Rov mean the real input vector and real output vector, $Viv(i)$、$Vov(i)$ mean the virtual input vector and virtual output vector, N means the number of the virtual vector divided from the real vector. So from equation (6) we can regard the real vector as the sum of the virtual vectors. In fact, in the calculation these vectors need to transform to scalars, and the equation(6) is renewed to equation (7):

$$
\begin{aligned}
Riv_{(am)}\cos(Riv_{(dir)}) &= \sum_{i=1}^{N} Viv_{(am)}\cos(Viv(i)_{(dir)}) \\
Riv_{(am)}\sin(Riv_{(dir)}) &= \sum_{i=1}^{N} Viv_{(am)}\sin(Viv(i)_{(dir)}) \\
Rov_{(am)}\cos(Rov_{(dir)}) &= \sum_{i=1}^{N} Vov_{(am)}\cos(Vov(i)_{(dir)}) \\
Rov_{(am)}\sin(Rov_{(dir)}) &= \sum_{i=1}^{N} Vov_{(am)}\sin(Vov(i)_{(d\ i)r})
\end{aligned}
\tag{7}
$$

It means that the real vectors and the virtual vectors should satisfy the two group functions. For a smooth edge, we regard the curvature of it changes uniform. So by detecting the vectors on the same direction, we can then contribute the change rate r of the edge.

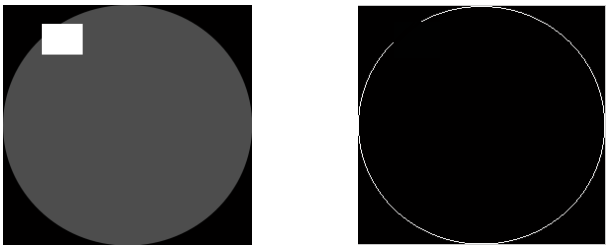
$$
r = \frac{\sum_{i=1}^{N}(Riv_{i+1} - Riv_i)}{N} \tag{8}
$$

Here the N is the number of the vectors on the same directions. With the change rate r and the gradient amplitude of virtual input vector Viv_{am} then we can contribute the gradient amplitude of virtual output vector Vov_{am}:

$$
Vov_{am} = Viv_{am} * (1 + r) \tag{9}
$$

On the surface, these processing just on the gradient amplitude, but in fact due to the gradient amplitude on different virtual directions are changed. So the gradient amplitude and direction of real vector are both already changed, and the pixels' gradient in the in-painting region is then predicted.

The whole processing actually completed to using the pixel's gradient information to predict the gradient of the next pixel on the same direction. Loop this processing until to the detected edge pixel or the known pixel, then the prediction of this breaking region is end. Then predicting the other regions one after another, at last we completed to predict all the destroyed regions.



(a) (b)

Fig 3.The example shown the theory of edge prediction

(a) is original image, (b) is the edge detection result

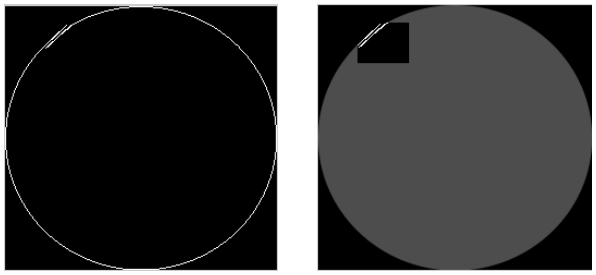
For example Fig 3(a) is a in-painting image, we contribute an edge like Fig 3(b) and the gradient information of the up breaking field is shown in Blank 1,the values out of the bracket are the gradient amplitudes and the values in the bracket are the gradient directions

Table 1. The comparison between the real and the predicted value

				0.11 (60)	0.21 (101)	0.22 (95)	0.08 (49)
		0.08 (50)	0.20 (95)	0.22 (100)	0.11 (60)		
0.06 (30)	0.19 (86)	0.23 (105)	0.14 (70)				
0.06 (25)	**0.19 (85)**	**0.22 (100)**	**0.16 (60)**				

In the Table 1 the first row and second row show the real gradient value of the breaking field on the top in the Fig 3(b), the values out of the bracket are the gradient directions(from 0 to 0.5，0.11means the direction is 0.11π) and the values in the bracket are the gradient amplitudes, the third row is also the real gradient value and the fourth row is the gradient value predicted by the first row and the second row gradient information.

From the result we can see this method is greatly contributing the gradient information in the in-painting area, and then we use this method to predict the whole edge in the in-painting area, just like the Fig 4 shows to us.



(a) (b)

Fig 4.the whole edge prediction result

(a) is the edge detection image ,(b) is the original image

At last we successfully predict the edge of the in-painting part, just like cut the in-painting region into many parts, then go to the next in-painting process to in-paint the each part with the improved FMM method.

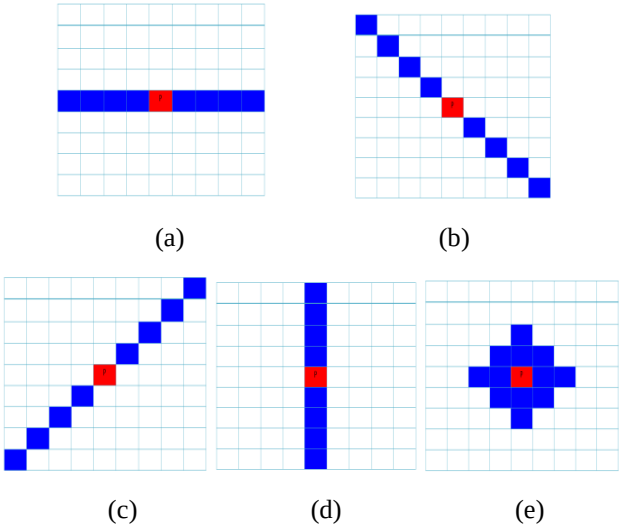
3.3 Select the texture direction by the gradient matrix

Now we've get the two gradient matrix, the value are continuous, but we hope the gradient direction to be a discrete value to carry on the next work .So we should define several gradient direction to replace the gradient direction value. According to the number of the defined direction, it will influence the in-paint result; here we just

for demonstrating the feasibility of our method, so we use a easier condition to work, set the direction number to 4.so the gradient direction will be divided into 4 situations.

Then add another situation represent the situation that the gradient value of all the pixels in the window $B_\varepsilon(p)$ are very small .we set a threshold, if the sum of the gradient value is less than the threshold, regard it gradient direction as this situation.

Count the gradient value on every direction of all the known pixels in the window, select the direction which has the biggest value as the in-paint pixel's gradient direction. Then according to the gradient direction, only select the pixels on its texture direction (from gradient direction we can get texture direction because they are vertical) to compute the grey value.



(a) (b) (c) (d) (e)

Fig.5 the gradient direction situation (a)-(d) are the 4 gradient direction, (e) is the adding situation. Red pixels are the in-painting pixels, blue pixels are the selected pixels in each situation.

3.4 Compute the weight

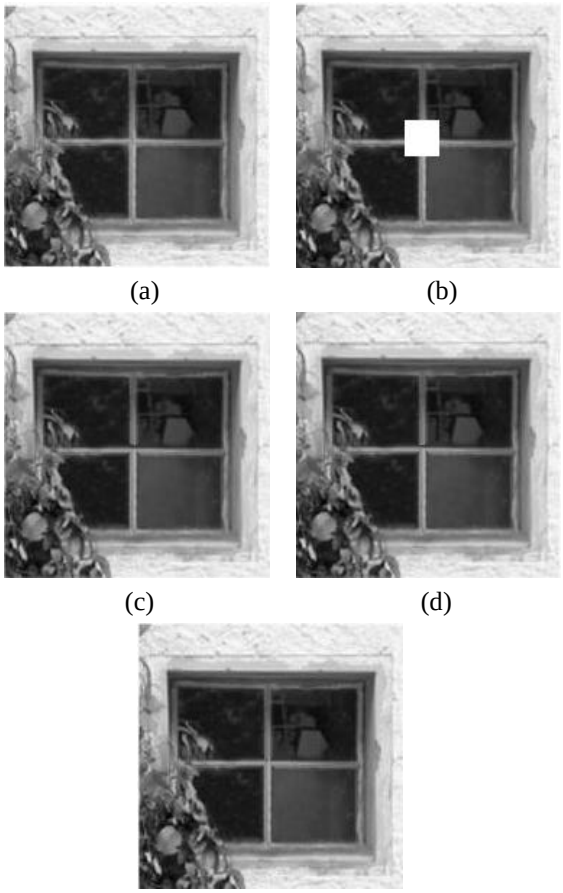
At last, using the FMM algorithm to compute grey value of the pixels, due to the new method, here we just remain the *dst* index and the lev index to compute, so the w (p, q) is changed to be

$$w(p,q) = dst(p,q) * lev(p,q) \qquad (3\text{-}4)$$

In this equation, the dst index and lev index are the same as the old method.

4 .Experiment result

In this section, we test the proposed improved FMM algorithm on a variety of natural images. And then compare the result with the old method. In the following examples, they show the advantages of our method and the feasibility in theory. The window $B_\varepsilon(p)$ is set as 9 by 9 (according to a lot experiment result, too small window will not have enough pixels, too large will bring more noise), the direction number is set to 4, and the threshold is set to 25.

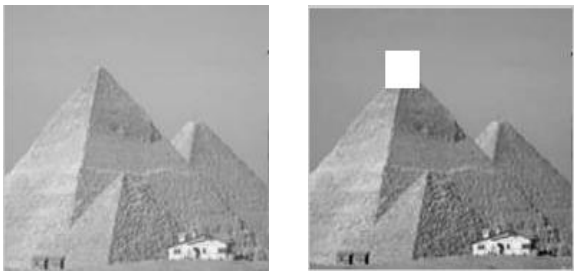


(a) (b) (c) (d) (e)

Fig 6.Small region in-painting result

(a)is original image,(b)is destroyed image,(c)is traditional method in-painting result, (d)is the result made by the method in[18],(e) is the proposed method result.

Fig.6 shows the first example, the new one is a little better, especially breaking region of the window is well connected.



(a) (b)

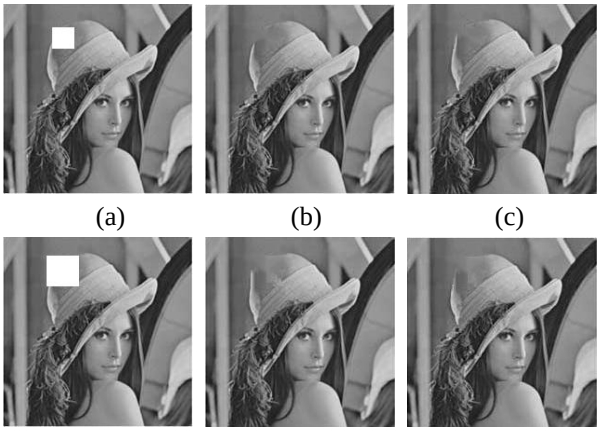
(c) (d)

(e)

Fig 7.Large region in-painting result

(a)is original image,(b)is destroyed image,(c)is traditional method in-painting result, (d)is the result made by the method in[18],(e) is theproposed method result.

Fig 7 shows another example, from the result we can see that two main edges are improved. The edge on the left is still fuzzy. By the edge prediction result we find the reason is that the left edge is not detected by the edge detection method based on Morphology. The pixels on the both sides of the edge are not changed obvious, so this region is still fuzzy, as the figure 7(d) shows us.



(a) (b) (c)

(d) (e) (f)

Fig.8 The result for in-painting one image of different destroyed region sizes

For one image, when the size of the destroyed region become larger, the traditional in-painting method may become weaker in maintaining a correct edge, but by my improved in-painting method, it can give a better edge effect, Table 2 uses PSNR to show the advantage in in-painting large destroyed region.

Table 2. In-painting result of Fig 8

Image Lena	Tradition method result (PSNR)	My method result (PSNR)
Small destroyed region	48.9702	48.2079
Large destroyed region	39.9890	42.6244

5. Conclusion

In this work, two new ideas are used to the FMM in-painting algorithm. It overcomes the disadvantage of traditional FMM in-painting method, which cannot contribute an optimal edge result. The first one is using gradient matrix to select pixels which has more influence with inpainting region. The other one is by changing the inpainting sequence to improve the edge effect. To predict the edge in the in-painting region and cut it into several sub regions, then inpaint the sub region one after another. The result images seem more reasonable and more similar to the natural scenes. At last, we can see the proposed approach not only remains the advantage of fast processing speed, but also contributes a better estimation result of edges compared with the traditional method.

REFERENCES

(1)M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester, “Imagein-painting,” in Proc. SIGGRAPH, pp. 417–424. 2000.

(2)M.Bertalmio, A. L. Bertozzi, and G. Sapiro, “Navier–Strokes, fluid dynamics, and image and video in-painting,” in Proc. IEEE Computer Society Conf. Computer Vision and Pattern Recognition, pp.417–424. 2001.

(3)T. Chan and J. Shen, “Localin-painting models and tvin-painting,” SIAM J. Appl. Math., vol. 62, no. 3, pp. 1019–1043, 2001.

(4)H Lu, L Zhang, and S Serikawa, “Maximum local energy: An effective approach for multisensor image fusion in beyond wavelet transform domain,” in Computers & Mathematics with Applications, vol.64, no.5, pp.996-1003, 2012.

(5)S.serikawa and H Lu, “Underwater image dehazing using joint trilateral filter” in Computers &Electrical Engineering vol.40,no.1,pp.41-50,2014

(6)T. Chan, and J. Shen, “Non-texturein-painting by curvature-driven diffusions” J. Vis. Commun. Image Represent, vol. 4, no. 12, pp. 436–449, 2001.

(7)C. Bertalmio, M. Bertalmio, V. Caselles, G. Sapiro, and

J. Verdera, “Filling-in by joint interpolation of vector fields and gray levels,” IEEE Transactions on Image Processing, vol. 10, no. 8,pp. 1200–1211, 2001.

(8)A. Levin, A. Zomet, and Y. Weiss, “Learning how to in-paint from global image statistics,” in Proceedings of International Conference on Computer Vision, vol.1,pp. 305–313,2003

(9)S. Roth and M. J. Black, “Fields of experts: A framework for learning image priors,” in Proc. IEEE Computer Society Conf. Computer Vision and Pattern Recognition,vol.2, pp. 860–867, 2005.

(10)S. Roth and M. J. Black, “Steerable random fields,” in Proc. IEEE Computer Society Conf. Computer Vision and Pattern Recognition, pp. 1–8, 2007

(11)A. Efros and T. Leung, “Texture synthesis by non-parametric sampling,” in Proceedings of International Conference on Computer Vision, pp. 1033–1038, 1999.

(12)M. Bertalmio, L. Vese, G. Sapiro, and S. Osher, “Simultaneous structure and texture image in-painting,” IEEE Trans. Image Process., vol. 12, pp. 882–889, 2003.

(13)A. Criminisi, P. Perez, and K. Toyama. “Region filting and object removal by examplar-based image inpainting ,” in IEEE Trans. Image Process.,vol. 33,pp. 1200-1212,2004.

(14)J.Wu and Q. Ruan, “Object removal by cross isophotes examplar-based image in-painting,” in Proc. Int. Conf. Pattern Recognition, pp. 810–813. 2006.

(15) A. Wong and J. Orchard. “A nonlocal-means approach to examplar based in-painting.” published in the IEEE Int. Conf. Image Processing, pp. 2600-2603,2008.

(16) A. Telea “An image in-painting technique based on the fast marching method,” in Journal of Graphics Tools, vol 9,no. 1,pp.23-34, 2004.

(17)Y. Yang, X. Juan, “An improved image in-painting algorithm based on fast marching method,” in Journal of Xi'an University of Technology, vol.25, no.2, pp.129-134,2009.

(18)Qian Fan, Xuelong Hu, Lifeng Zhang, "Image inpainting based FMM algorithm by a direction selection method", Proc. of ICIAE2014, pp.12-15, 2014.