IMAGE INPAINTING VIA SPARSE REPRESENTATION

Bin Shen[1], Wei Hu[2], Yimin Zhang[2] , Yu-Jin Zhang[1]

[1]Tsinghua National Laboratory for Information Science and Technology
[1]Department of Electronic Engineering, Tsinghua University, Beijing 100084, China
chenbin03@mails.tsinghua.edu.cn, zhang-yj@mail.tsinghua.edu.cn
[2]Intel China Research Center, Beijing, P.R. China
{wei.hu, yimin.zhang}@intel.com

ABSTRACT

This paper proposes a novel patch-wise image inpainting algorithm using the image signal sparse representation over a redundant dictionary, which merits in both capabilities to deal with large holes and to preserve image details while taking less risk. Different from all existing works, we consider the problem of image inpainting from the view point of sequential incomplete signal recovery under the assumption that the every image patch admits a sparse representation over a redundant dictionary. To ensure the visually plausibility and consistency constraints between the filled hole and the surroundings, we propose to construct a redundant signal dictionary by directly sampling from the intact source region of current image. Then we sequentially compute the sparse representation for each incomplete patch at the boundary of the hole and recover it until the whole hole is filled. Experimental results show that this approach can efficiently fill in the hole with visually plausible information, and take less risk to introduce unwanted objects or artifacts.

Index Terms**—** Image inpainting, texture synthesis, sparse representation, Lasso, L1 norm minimization

1. INTRODUCTION

Image inpainting aims to automatically fill in the holes of image that are to be removed for artistically or confidential considerations. Generally speaking, there are two main categories of image inpainting approaches in the literature: PDE based approaches [2,18] and exemplar based approaches. The formers aim to extend the lines or edges in known area into the user specified areas, which pay sufficient attention to structure propagation, but do not suitably deal with large regions due to the blur effects in their case. The later approaches [8,9] adopt texture synthesis method to synthesize the pixels in the user specified region. For a particular target patch, they search the most suitable source patch from remaining regions to replace the target patch. Later, [6] considers the structure propagation as well as texture synthesis by computing patch priorities for determining the filling order. Their improvements [3, 13] have also been proposed by some researchers.

Some others also pay attention to both texture and structure propagations. Sketch model [11] has been used to restore the missing structure, and then patch based synthesis is deployed to fill in the regions [5]. Similar works in [10, 16] use PDE based method and tensor voting, respectively, to restore the structure, and then deploy texture synthesis to fill the specified regions. Works in [17] allow user to manually specify important missing structure information, and then deploy dynamic programming or belief propagation to complete the structure before synthesizing the pixels in target regions.

Generally speaking, exemplar based approaches work well to synthesize the texture in target areas, thus are more suitable to deal with large regions, e.g. to remove background persons in a photograph, and to keep image details in the filled region. However, it always selects the most suitable patch for the current place, thus is a greedy method, which results in a risk of introducing unwanted object or artifact to the area to inpaint, which can be seen in the bottom right images in figure2 (a) and (b). As we will point out, this inpainting method can be viewed as a special case of our proposed algorithm here, and the general cases of our methods are less greedy. Our main contribution is to borrow the signal sparse representation technique to address the inpainting problem, and bridging the gap between sparse representation and texture synthesis. Signal sparse representation means that the signal admits a sparse representation over a redundant dictionary, which we will review in the following section. In this paper, we view this problem as the recovery of incomplete image signals, with each signal corresponding to a patch. We fill the hole patch-wisely based on the sparse representation of each patch.

The rest part of this paper is organized as follows. In the second section, we simply review the used sparse representation technique: Lasso. In the third section, we describe our inpainting algorithm. The experimental results are listed in section four. Finally, we discuss the relation between the proposed algorithm and some related issues, and then we conclude this paper.

2. SPARSE REPRESENTATION

Recently, signal sparse representation draws lots of researchers' attention. Donoho [7] proved that the L1 norm is a good approximation of L0 norm. Thus, many techniques are supported.

Tibshirani [18] proposed a regression method: Lasso. He added a L1 norm penalty to the loss function of ordinary least square regression, which results in the sparsity of the coefficient. We briefly review it now.

Given the dictionary $\mathbf{x} = [\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^N]$ and input signal $\mathbf{y} = [y_1, y_2, \ldots, y_p]^T$. For convenience, we assume the data $\mathbf{x}, \mathbf{y}$ are normalized. The Lasso algorithm is to estimate the coefficient $\boldsymbol{\beta}$ of a signal over the given dictionary by

$$\hat{\boldsymbol{\beta}} = \arg\min\{||\mathbf{y} - \mathbf{x}\boldsymbol{\beta}||_2^2 + \lambda\, ||\boldsymbol{\beta}||_1\} \quad (1)$$

The term $||\boldsymbol{\beta}||_1$ encourages the sparsity of the fitted coefficient vector, and the parameter λ controls the tradeoff between the reconstruction error and the sparsity. This formulation is based on the model that $\mathbf{y} = \mathbf{x}\boldsymbol{\beta}$ and $\boldsymbol{\beta}$ is sparse, i.e. only a few components of $\boldsymbol{\beta}$ are nonzero.

It is more interesting, when some components of signal are corrupted, which means the model is modified into

$$\mathbf{y} = \mathbf{x}\boldsymbol{\beta} + \mathbf{e} \quad (2)$$

$\mathbf{e}$ is the error and the e_i is nonzero if and only if the y_i is corrupted. It is a problem to find out which components of the signal $\mathbf{y}$ are corrupted. However, in our application, this is avoided because the user specifies holes, i.e. the target region to inpaint, which means that we can know which pixels are corrupted or not. So here we only consider the case that the indexes of the corrupted components of $\mathbf{y}$ are known. We denote the corrupted signal component index set by $I, I = \{i|e_i \neq 0\}$. Let $\mathbf{y}_{\backslash I}$ denotes the vector obtained by removing the components whose indexes are in I from $\mathbf{y}$, i.e. $\mathbf{y}_{\backslash I}$ is made up by all the non-corrupted components of $\mathbf{y}$. $\mathbf{x}_{\backslash I}$ is the corresponding dictionary matrix, which is obtained by removing all the columns whose indexes are in I from $\mathbf{x}$. Now the sparse coefficient $\boldsymbol{\beta}$ can be estimated as follows.

$$\hat{\boldsymbol{\beta}} = \arg\min\{|\mathbf{y}_{\backslash I} - \mathbf{x}_{\backslash I}\boldsymbol{\beta}|_2^2 + \lambda\, ||\boldsymbol{\beta}||_1\} \quad (3)$$

Then, we recover the corrupted signal via the $\hat{\boldsymbol{\beta}}$ in (4).

$$\hat{y}_i = \begin{cases} y_i, & if\ i \notin I \\ (\mathbf{x}\hat{\boldsymbol{\beta}})_i, & if\ i \in I \end{cases} \quad (4)$$

So, combining (3) and (4), we can recover the corrupted signal.

3. INPAINTING ALGORITHM

Now we describe our algorithm for image inpainting via sparse representation.

3.1. Filling order

Given an input image, the user selects the target region, which is to be removed and filled. Then usually the left is treated as the target region. Of course, it can be also specified by the user. We denote the target region Ω, the source region Φ and the boundary of target region $\delta\Omega$.
We grow the image from the boundary of the hole towards the inside.

Here we follow [6] to determine the filling order because it efficiently reserves the structure information. At each iteration, we compute the priority $P(p)$ for every pixel p on the boundary $\delta\Omega$, we select the pixel with maximum priority as p_m. The patch centers at p_m is to be addressed in the current iteration. Since the patch centers on the boundary, so some pixels of the patch are in the target region, thus the patch can be view as a incomplete signal with the existing components corresponding to the pixels in source region and lost components corresponding to the pixels in target region. To recover this selected signal (patch), we use sparse representation, which will be detailed in 3.2. After recovery of the current patch, the boundary of target region is updated. Then we compute the priority for the next iteration. For details of the priority computing, please refer to [6].

3.2. Signal recovery

Now, we address the selected p_m. Let a k-dimensional vector Ψ_{p_m} denotes the patch centered at p_m. Obviously, $k = n \times n$, where n is the width or height of the patch.
Since p_m is at the boundary, some components of Ψ_{p_m} belongs to the target region and the others belongs to the source region. Fortunately, based on the user's selection, we easily know the division of Ψ_{p_m}. Recall what we have discussed in section 2. Now we consider the Ψ_{p_m} as the $\mathbf{y}$, which is to be recovered. Intuitively, the index set of components of $\mathbf{y}$ belong to target region can be considered as I.

So now, we compute the sparse representation of the current patch and recover it by (3) and (4). Detailedly,

$$\hat{\boldsymbol{\beta}} = \arg\min\{\left|\Psi_{p_m \backslash I} - \mathbf{x}_{\backslash I}\boldsymbol{\beta}\right|_2^2 + \lambda\, ||\boldsymbol{\beta}||_1\} \quad (5)$$

$$\hat{\Psi}^i_{p_m} = \begin{cases} \Psi^i{}_{p_m}, & if\ i \notin I \\ (\mathbf{x}\hat{\boldsymbol{\beta}})_i, & if\ i \in I \end{cases} \quad (6)$$

where $\mathbf{x}$ is the dictionary for sparse representation and how to construct the dictionary will be discussed in 3.3.
Now we get the recovered $\hat{\Psi}_{p_m}{}^i$, according to which we reset the value of the pixels in the patch centered p_m to visualize our result.

3.3. Dictionary construction

To compute the sparse representation of signal, we first have to construct a dictionary, based on which we solve the Lasso regression. Many techniques can be adopted to fix the dictionary, such as Matching Pursuit [15] Basis Pursuit [4], or K-SVD [1].

According to our observation, the filled target region should be visually consistent with source region so that the entire image looks plausible. This means that not only the texture should be consistent, but also the noise should also be the same level. So we directly samples or even use all the patches in the source region to construct the dictionary without any preprocessing. For example, if we get M patches from the source region, the fixed dictionary should have M columns. Each column corresponds to a patch. This technique of using the original image as dictionary has been used in face recognition, background modeling, etc [12,19] to achieve encouraging results.

3.3. Overall algorithm

As discussed above, the overall algorithm we proposed is given in Table 1 for convenience.

Table 1: Inpainting Algorithm Proposed

```
Input: an image, source region $\Phi$, target region $\Omega$
1. Dictionary Construction: Directly sampling or use all
   patches from $\Phi$ to construct the dictionary
2. Do until all pixels in $\Omega$ are filled.
      -a. Compute the priority for all pixels at the
          boundary $\delta\Omega$ , and select the patch with
          maximum priority to address. This patch is
          viewed the incomplete signal.
      -b. Use (5), (6) to recover the incomplete
          signal(patch)
      -c. Set the pixels in the target region of selected
          patch to values according to recovered signal,
          and update the boundary $\delta\Omega$
Output: the inpainted image
```

4. EXPERIMENTAL RESULTS

To verify our algorithm, we conduct experiments on a lot of natural images, which we will describe concisely.

Figure 1 shows some results by our algorithm. In each row, the left image is the original image, the middle shows the target area selected by user, and the right one is the final inpainting result. We can easily see that our algorithm can deal with large holes, which we will discuss in 5.1. In the first two rows, we can also see that the algorithm recovers the structure very well. Figure 2 shows two results of our algorithm compared with the results of classic algorithm by [6]. In Figure2(a) and (b), the top-left images are the original ones, the top-right ones show the target area selected by users, the bottom-left are the inpainting results by our proposed algorithm, while the bottom-right ones show the results by [6]. We find that the results by [6] introduce unwanted objects while our algorithm does not take such risk, and gains better result. Actually, it is because our algorithm is less greedy than [6]. In each step, [6] selects the local best patch to inpaint the target patch, while our algorithm selects a small set of patches and inpaints the target patch based on a linear combination of them. So our algorithm is less greedy for not selecting a local optimal patch, more efficient to reduce the SSD error to ensure visually plausibility for using a linear combination, and will not suffer the blurring of image for L1 norm constraint encourages the sparsity of the linear combination, i.e. the L1 norm reduces the number of patches selected. Based on all the merits and supporting experimental results, it can be shown that general sparse representation based algorithm proposed works better on image inpainting better than texture synthesis based algorithm[6], which is the state of art.

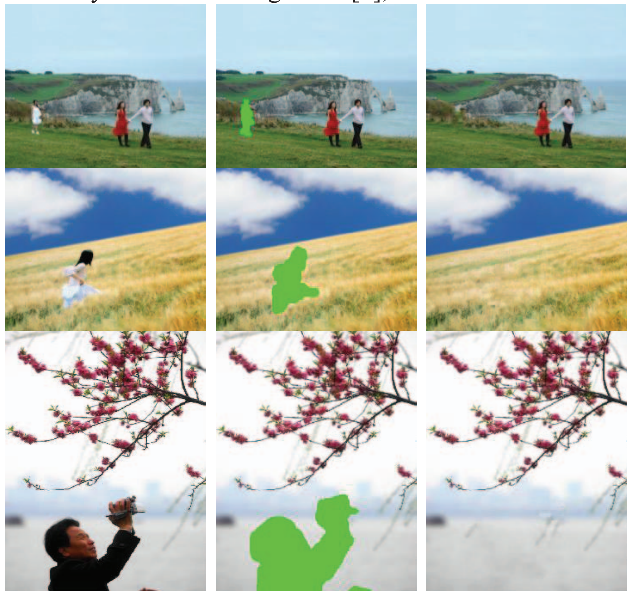


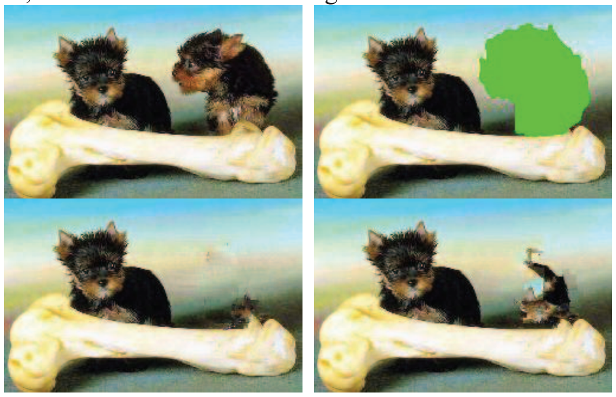
Figure 1: Inpainting results of propose algorithm

5. DISCUSSION AND CONCLUSION

Here we discuss the relation between our algorithm and several related ones, and then we simply conclude this paper.

5.1. Relation with some related works

A famous texture synthesis based inpainting algorithm proposed by [6] selects the most suitable patch from the source region at each step, thus it can be viewed as a special case of our algorithm if we constrain that only one patch is selected in each step. Actually, it can be implemented to tune the parameter λ each step. So, texture synthesis is actually a sparse representation method with the constraint that L0 norm (the number of nonzero entity in vector) equals to 1. Thus, the gap between texture synthesis and sparse representation is bridged. However, in general, our algorithm selects several columns of the dictionary, i.e. several different patches if we construct the dictionary by directly sampling patches, and use a linear combination of them to fit the target patch. Thus, our algorithm has higher capability to fit diverse patches for its form of linear combination, and is less greedy for not choosing the optimal one, which has been shown in figure 2.


(a)

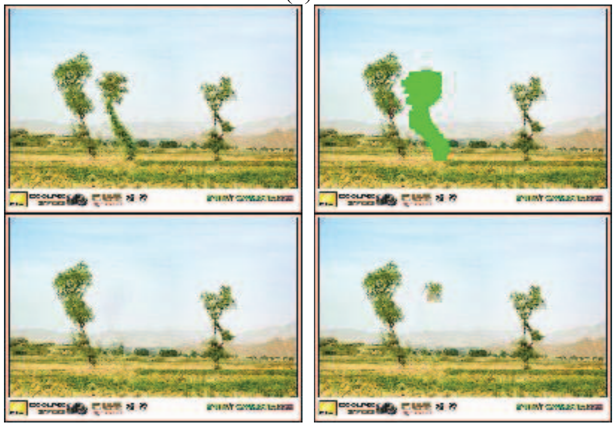

(b)

Figure 2: Inpainting results of proposed algorithm and [6]

A similar algorithm [14] with ours also use sparse representation to do image inpainting. However they are mainly aimed on image restoration, not image inpainting specially. Thus their algorithm suffers from several drawbacks. First of all, as they said in the paper, they can only address holes smaller than 10×10 pixels. Secondly, they have to use additional information rather the given image to construct the dictionary, while our algorithm only relies on the given image. Last but not least, they view the inpainting as denoising: they treated the pixels to inpaint as image noise. However, we claim that inpainting is not denoising: if the surroundings are polluted by heavy noise, the filled region should also be polluted by noise so as to ensure the visually plausibility. [14] uses k-SVD [1] to construct a dictionary, which is efficient for denoising but insufficient for inpainting because it fails to capture the the noise property in images. In our algorithms, we directly sample patches from the source region to construct the dictionary, which render our algorithm preserves the noise property very well.

5.2. Conclusion

In this paper, we propose an algorithm for image inpainting based on the sparse representation. Our algorithm can not only address large holes, but also maintain both texture consistency and noise consistency. Another merit of our algorithm is that it is less greedy than traditional texture synthesis based inpainting algorithm. We also bridge the gap between texture synthesis and sparse representation as discussed in 5.1. The experimental results show our algorithm is efficient to remove unwanted objects from digital graphs and is less greedy than state of art algorithm.

6. REFERENCES

[1] M. Aharon, M. Elad, and A.M. Bruckstein, “K-SVD: Design of Dictionaries for Sparse Representation”, Proceedings of SPARSE'05, Rennes, France, November 2005, pp.9-12.
[2] M. Bertalmio, G. Saporo, V. Caselles and C. Ballester, “Image Inpainting,” In SIGGRAPH 2000
[3]Q. Chen, Y. Zhang, and Y. Liu, “Image Inpainting with Improved Exemplar-Based Approach”, MCAM 2007, LNCS 477, pp.242-251, Springer-Verlag Berlin Heidelberg 2007
[4]S.S. Chen, D.L. Donoho, and M.A. Saunders. “Atomic Decomposition by Basis Pursuit.” SIAM Review, 43(1), 2001, pp. 129-159.
[5]Y. Chen, Q. Luan, H. L, and O. Au, “Sketch-guided Texture-based Image Inpainting,” In ICIP 2006
[6] A. Criminisi, P. P´erez, and K. Toyama, “Object Removal by Exemplar-Based Inpainting,” In CVPR 2003
[7]D. Donoho, “For most large underdetermined systems of equations, the minimal l1-norm near-solution approximates the sparsest near-solution”, Communications on Pure and Applied Mathematics, 59 (2006), pp. 907-934.
[8]A. A. Efros and W. T. Freeman, “Image Quilting for Texture Synthesis and Transfer,” In SIGGRAPH 2001
[9] A. A. Efros and T. K. Leung, “Texture Synthesis by Non-parametric Sampling,” In ICCV 1999
[10]H. Grossauer, “A Combined PDE and Texture Synthesis Approach to Inpainting,” In ECCV 2004
[11]C. Guo, S-C. Zhu and Y. N. Wu, “Towards a Mathematical Theory of Primal Sketch and Sketchability”, In ICCV 2003
[12]J. Huang, X. Huang, and D. Metaxas, “Simultaneous Image Transformation and Sparse Representation Recovery”. In CVPR 2008
[13]J. Jia and C-K. Tang, “Image Repairing: Robust Image Synthesis by Adaptive ND Tensor Voting”, In CVPR 2003
[14] J. Mairal, M. Elad, and G. Sapiro, “Sparse Representation for Color Image Restoration,” IEEE Transaction on Image Processing, Vol. 17, No. 1, January 2008 pp.53-69
[15]S. Mallat and Z. Zhang. “Matching pursuits with time-frequency dictionaries.” IEEE Transaction on Signal Processing. 41(12), 1993, pp. 3397-3415.
[16]S. Masnou, J.-M. Morel, “Level Lines based Disocclusion,” In ICIP 1998
[17]J. Sun, L. Yuan, J. Jia, and H-Y. Shum, “Image Completion with Structure Propagation,” In SIGGRAPH 2005
[18] R. Tibshirani, “Regression Shrinkage and Selection via the Lasso”, Journal of the Royal Statistical Society. Series B(Methodological), Volume 58, Issue 1(1997), pp. 267-288
[19] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, “Robust Face Recognition via Sparse Representation.” to appear IEEE Transaction on Pattern Analysis and Machine Intelligence.