DATA-DRIVEN IMAGE COMPLETION BY IMAGE PATCH SUBSPACES

Hossein Mobahi, Shankar R. Rao, Yi Ma

Coordinated Science Laboratory
University of Illinois at Urbana Champaign Urbana, IL 61801

ABSTRACT

We develop a new method for image completion on images with large missing regions. We assume that similar patches form low dimensional clusters in the image space where each cluster can be approximated by a (degenerate) Gaussian. We use sparse representation for subspace detection and then compute the most probable completion. Our results show almost no blurring or blocking effects. In addition, both the texture and structure of the missing regions look realistic to the human eye.

Index Terms— Inpainting, Image Subspaces, Sparse Representation, Degenerate Gaussians

1. INTRODUCTION

Image completion (also called image inpainting) is the task of filling in or replacing an image region with new data such that the modified image looks natural and real to human eyes. This problem has a number of applications such as removing unwanted items off a picture, fixing cracks or corrupted portions of old photographs, and filling holes caused by occlusion in 3D image reconstruction.

Early approaches to image completion focused on texture synthesis [1]. This trend alone could not be scaled to natural images due to presence of geometrical structures that are different from texture information. One way of completing structure is evolving a missing region from its boundary inwards using partial differential equations (PDE) [2]. Other lines of work seek to transform images to a domain where their representations are *sparse*, having few nonzero entries [3]. The image is filled in such a way that the representation of the image as a whole is as sparse as possible. In practice, these methods uses generic pre-defined transformations such as curvelets and cosines.

All of the above methods are strongly model-driven and have little adaption to the data. The reported results are either on very small missing regions or, when applied to larger regions, exhibit blur and blocking effects. Exemplar-based methods take a completely different approach. Rather than building mathematical models from scratch, these methods synthesize the missing parts of an image from a collection of pieces of real images, known as a *patch dictionary*. These methods bypasses the inevitable modeling error and gross simplifications with model-based approaches, and have been successfully applied to larger missing regions.

The majority of exemplar methods replace the missing region with patches exactly as they arise in the training data [4, 5]. Since these methods cannot produce novel patches, the patch dictionary must contain at least one suitable candidate for (a portion of) the missing region. This is a very strong assumption even with access to millions of image patches. Successful examples in these works are images where repeating a single patch can approximate a large region of the image [4] or when the missing region cuts exactly a whole object [5]. While the latter situation is reasonable for semi-supervised applications, such as image editing, in more general image restoration tasks, missing regions can often cross object boundaries. This is much more difficult, because both the object and its boundary must be completed in a perceptually meaningful way.

In this work, we relax the aforementioned assumption by allowing missing regions to be filled using novel patches synthesized from a limited dictionary of real patches. In particular, the novel patches are linear combinations of the real image patches in the dictionary. Thus, even if no patch in the dictionary matches a given missing region, it is still possible to fill in the region with an appropriate linear interpolation of patches. To avoid the problem of over fitting, we seek novel patches that have sparse representations w.r.t. to the dictionary. Finally, we assume that a set of patches having similar texture and structure have a distribution that can be well approximated by a degenerate Gaussian. This assumption provides some probabilistic justification for our approach, and works well with the notion of sparsity.[1]

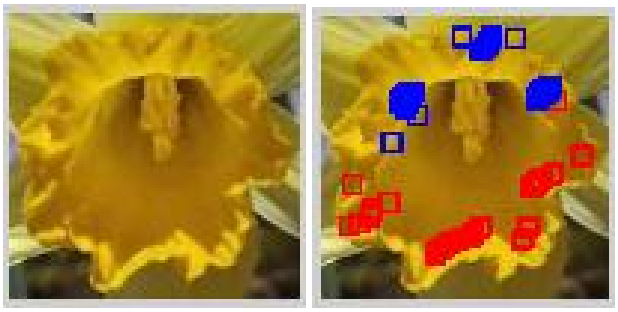
2. IMAGE COMPLETION BY PATCH SUBSPACES

2.1. Problem Formulation

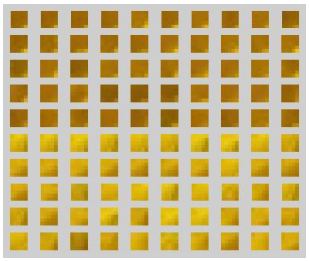
To provide some motivation for our assumptions, we first consider the example given in Figure 1 (a). We select a random

This work is partially supported by grants NSF CRS-EHS-0509151, NSF CCF-TF-0514955, and ONR YIP N00014-05-1-0633.

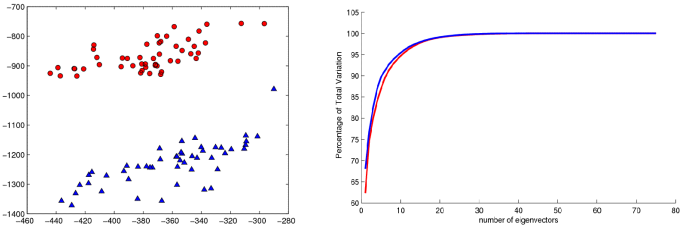
[1] Our strategy is similar in spirit to [6], as both approaches use sparse representation as a tool for filling in missing regions. However, as we will show in the next section, our degenerate Gaussian assumption results in more accurate and robust completion, allowing our approach to be applied to larger missing regions in an image.



(a) Two groups of similar patches



(b) Top 5 rows are red and the rest are blue patches



(c) Patches projected onto 2D by PCA

Fig. 1. Distribution of two categories of patches

5×5 patch in the image, and then find the 49 other patches in the image that are most similar to the given patch. Here, the measure of similarity is the normalized inner product between patches. In this fashion, we obtain two groups of 50 similar patches, as seen in Figure 1 (b). Though the ambient dimensionality of each patch is 75 (25 pixels per RGB channel), for each set of patches, 95% of the total variation is contained in the top 10 eigenvectors (see Figure 1 (c)). This indicates that each set lies approximately on a 10-dimensional subspace. The first two principal components of each patch are shown in Figure 1 (c).

Based on these observations, we assume that the set of patches with similar appearance form a cluster in the image space. We further assume that the intrinsic dimensionality of such clusters is much lower than ambient dimension. The latter assumption stems from high correlation among pixel values in natural images. We approximate each of such clusters with a multivariate normal distribution whose covariance has a very low rank. Note that the rank of each Gaussian cluster (or equivalently the dimensionality of the subspace where the samples live in) might be different, depending on the complexity of the patches in the cluster. For example, the set of patches from a plain white wall should lie on a lower-dimensional subspace than patches from carpet.

Based on the above assumptions, we formally state the problem of filling a single patch with missing pixels. We represent each patch as a vector by stacking up pixel values of each patch.

Problem 1 (Data-Driven Completion of a Gaussian Vector) *Suppose we have a set of training vectors $\{\boldsymbol{z}_i\}_{i=1}^n$, where each $\boldsymbol{z}_i \in \mathbb{R}^d$ is an i.i.d. sample from one of K degenerate normal distributions $\{\mathcal{N}_k(\boldsymbol{\mu}_k, \Sigma_k)\}_{k=1}^K$. We do not know from which of the K distributions each $\boldsymbol{z}_i$ is drawn, and K is also unknown. We associate to each $\mathcal{N}_k$ a subspace $\mathcal{S}_k$ spanned by the eigenvectors of Σ_k that correspond to its nonzero eigenvalues. Given a test observation with m missing entries*

$$\tilde{\boldsymbol{z}} = [\tilde{\boldsymbol{x}}^T \ \tilde{\boldsymbol{y}}^T]^T, \tag{1}$$

where $\tilde{\boldsymbol{x}} \in \mathbb{R}^{d-m}$ denotes the visible components and $\tilde{\boldsymbol{y}} \in \mathbb{R}^m$ denotes the missing components, estimate $\tilde{\boldsymbol{y}}$ from $\tilde{\boldsymbol{x}}$ and $\{\boldsymbol{z}_i\}_{i=1}^n$.

2.2. Our Approach

Our first task is to identify the subspace $\mathcal{S}^* \in \{\mathcal{S}_1, ..., \mathcal{S}_K\}$ for which $\tilde{\boldsymbol{z}} \in \mathcal{S}^*$. Though $\tilde{\boldsymbol{z}}$ can be represented by any d vectors that span the ambient space, it can only be represented *sparsely* by vectors from $\mathcal{S}^*$. However, we can only observe $\tilde{\boldsymbol{x}}$, a portion of the vector $\tilde{\boldsymbol{z}}$. If the number of missing entries m is small, then we can recover the sparse representation for $\tilde{\boldsymbol{z}}$ *without* observing $\tilde{\boldsymbol{y}}$. Later, we will show that we can design our algorithm so that m is *guaranteed* to be small.

The observed portion $\tilde{\boldsymbol{x}}$ is related to $\tilde{\boldsymbol{z}}$ by $\tilde{\boldsymbol{x}} = \Pi\tilde{\boldsymbol{z}}$, where $\Pi \in \mathbb{R}^{(d-m)\times d}$ is the linear projection that removes the entries in $\tilde{\boldsymbol{y}}$ from $\tilde{\boldsymbol{z}}$. As long as the dimension d_k of each subspace $\mathcal{S}_k$ is strictly less than $d - m$, an arbitrary $(d - m)$-dimensional projection preserves the structural relationships between the subspaces with probability one. In particular, if $\tilde{\boldsymbol{z}} \in \mathcal{S}^*$ can be linearly represented as

$$\tilde{z} = \sum_{j=1}^{d^*} a_j^* \boldsymbol{z}_j^*, \quad \text{for } \{\boldsymbol{z}_j^* \in \mathcal{S}^*\}_{j=1}^{d^*}, \tag{2}$$

then, with probability one, $\tilde{\boldsymbol{x}}$ has a representation $\tilde{\boldsymbol{x}} = \sum_{j=1}^{d^*} a_j^* \boldsymbol{x}_j^*$, where $\boldsymbol{x}_j^* = \Pi\boldsymbol{z}_j^*, j = 1...d^*$, and $\{a_j^*\}_{j=1}^{d^*}$ are the *same* coefficients used in (2). Thus, when m is small, we can determine $\mathcal{S}^*$ by constructing $\tilde{\boldsymbol{x}}$ from a *sparse* linear combination of $\boldsymbol{x}_1, ..., \boldsymbol{x}_n$.

In principle, the sparse representation of $\tilde{\boldsymbol{x}}$ can be obtained by solving the following problem.

$$\min_{\boldsymbol{a}} \|\boldsymbol{a}\|_0 \qquad subj. \ \tilde{\boldsymbol{x}} = \sum_i^n \boldsymbol{a}_i \boldsymbol{x}_i \tag{3}$$

where $\boldsymbol{a} = [a_1, a_2, ..., a_n]^T$ and $\|\boldsymbol{a}\|_0$ is equal to the number of non-zero coefficients in $\boldsymbol{a}$. Solving (3) is in general *NP-Hard*, and is thus computationally intractable, but it can be relaxed by replacing $\|\boldsymbol{a}\|_0$ with $\|\boldsymbol{a}\|_1$. The latter is a convex problem and can be solved efficiently via linear programming. It is known that if the problem indeed has a sparse enough solution, then this relaxation will recover the solution of the original problem [7]. In practice, using normalized $\boldsymbol{x}_i$'s provides better scaling for the coefficients $\boldsymbol{a}_i$, so we use $\boldsymbol{x}_i/\|\boldsymbol{x}_i\|_2$ instead.

Another issue with this formulation is the hard equality constraint $\boldsymbol{x} = \sum_i^n a_i \boldsymbol{x}_i/\|\boldsymbol{x}_i\|_2$. Real data, like the patches in Figure 1, are often noisy, and thus do not perfectly satisfy the equality constraint. Therefore, we allow some error in the constraint, but penalize it in the objective function. We use ℓ^1-norm penalty of the error so that the optimization remains a linear program.[2]

$$(\boldsymbol{a}^*, \boldsymbol{e}^*) = \underset{\boldsymbol{a},\boldsymbol{e}}{\operatorname{argmin}} \|\boldsymbol{a}\|_1 + \|\boldsymbol{e}\|_1 \quad (4)$$

$$subj. \qquad \tilde{\boldsymbol{x}} + \boldsymbol{e} = \sum_i^n a_i \boldsymbol{x}_i/\|\boldsymbol{x}_i\|_2$$

Denoting the support set by $\text{supp}(\tilde{\boldsymbol{z}}) = \cup_{a_i^* \neq 0}\{\boldsymbol{z}_i\}$, $\mathcal{S}^*$ will be span(supp($\tilde{\boldsymbol{z}}$)). We could estimate (μ^*, Σ^*) from supp($\tilde{\boldsymbol{z}}$). However, supp($\tilde{\boldsymbol{z}}$) only contains d^* vectors, and though those vectors are all drawn from the same Gaussian $\mathcal{N}^*$, they are *not* independent; these vectors were chosen precisely because they produce the sparsest representation. In order to obtain a reliable estimate of (μ^*, Σ^*), we need to find more vectors that lie on or near $\mathcal{S}^*$.

We denote the set of all training points whose $\boldsymbol{x}$ part falls into this subspace by $\mathcal{Z}^*$. Again we should consider noise and let points be considered on the subspace upto a tolerance distance ε. [3]

$$\mathcal{Z}^* = \{\boldsymbol{z}_i : d(\boldsymbol{z}_i, \mathcal{S}^*) < \varepsilon\} \quad (5)$$

Given $\mathcal{Z}^*$, we obtain sample estimates $(\hat{\boldsymbol{\mu}}, \hat{\Sigma})$ for the mean and covariance of $\mathcal{N}^*$. The most probable estimate for $\tilde{\boldsymbol{y}}$, denoted by $\boldsymbol{y}^*$ can be obtained as follows.

$$\begin{aligned}
\boldsymbol{y}^* &= \underset{\tilde{\boldsymbol{y}}}{\text{argmax}}\ \Pr(\tilde{\boldsymbol{y}} \,|\, \tilde{\boldsymbol{x}}\,;\, \hat{\boldsymbol{\mu}}, \hat{\Sigma}) \\
&= \underset{\tilde{\boldsymbol{y}}}{\text{argmax}}\ \Pr(\tilde{\boldsymbol{x}}\,,\, \tilde{\boldsymbol{y}}\,;\, \hat{\mu}, \hat{\Sigma}) / \Pr(\tilde{\boldsymbol{x}}\,;\, \hat{\boldsymbol{\mu}}, \hat{\Sigma}) \\
&= \underset{\tilde{\boldsymbol{y}}}{\text{argmax}}\ \Pr(\tilde{\boldsymbol{x}}\,,\, \tilde{\boldsymbol{y}}\,;\, \hat{\boldsymbol{\mu}}, \hat{\Sigma}) \\
&= \underset{\tilde{\boldsymbol{y}}}{\text{argmin}}\ (\left[\begin{smallmatrix}\tilde{\boldsymbol{x}}\\ \tilde{\boldsymbol{y}}\end{smallmatrix}\right] - \hat{\boldsymbol{\mu}})^T (\hat{\Sigma} + \lambda \mathbf{I})^{-1} (\left[\begin{smallmatrix}\tilde{\boldsymbol{x}}\\ \tilde{\boldsymbol{y}}\end{smallmatrix}\right] - \hat{\boldsymbol{\mu}}) \quad (6)
\end{aligned}$$

We regularize $\hat{\Sigma}$ by adding the scaled identity to facilitate inversion (λ is a small positive number). Because $\hat{\Sigma} + \lambda \mathbf{I}$ is positive definite, the problem of computing $\boldsymbol{y}^*$ becomes a convex quadratic program.

To complete a large missing region, we iteratively apply the single patch filling procedure. At every step, we find an incomplete patch window in the image that contains the least number of missing pixels. This ensures that the coefficients found by solving (4) indeed recover a basis for $\mathcal{S}^*$. We then solve (6) to obtain the estimate $\boldsymbol{y}^*$ for that patch. We only allow one pixel to be filled at a time (the one that is closest to the center of the patch), even if we have recovered more than one pixel by $\boldsymbol{y}^*$. This way the boundary is filled layer by layer and the filling gradually moves inwards. The complete algorithm is summarized in Algorithm 1.

Algorithm 1 Data-Driven Completion by Patch Subspaces

1: Input: Image $\mathcal{I}$, training patches $\boldsymbol{z}_1, ..., \boldsymbol{z}_n$, error tolerance ε, and regularization parameter λ
2: **while** # missing pixels in $\mathcal{I} > 0$ **do**
3: Choose an incomplete patch $\tilde{\boldsymbol{z}}$ from $\mathcal{I}$ with the least number of missing pixels
4: Solve $\min_{\boldsymbol{a},\boldsymbol{e}} \|\boldsymbol{a}\|_1 + \|\boldsymbol{e}\|_1$ $subj.$ $\tilde{\boldsymbol{x}} + \boldsymbol{e} = \sum_i^n a_i \boldsymbol{x}_i/\|\boldsymbol{x}_i\|_2$
5: Compute $\mathcal{S}^* = \text{span}(\text{supp}(\tilde{\boldsymbol{z}}))$, where $\text{supp}(\tilde{\boldsymbol{z}}) = \cup_{a_i^* \neq 0}\{\boldsymbol{z}_i\}$
6: Estimate $(\hat{\boldsymbol{\mu}}, \hat{\Sigma})$ from $\mathcal{Z}^* = \{\boldsymbol{z}_i : d(\boldsymbol{z}_i, \mathcal{S}^*) < \varepsilon\}$
7: Find $\boldsymbol{y}^* = \underset{\tilde{\boldsymbol{y}}}{\text{argmin}}\ (\left[\begin{smallmatrix}\tilde{\boldsymbol{x}}\\ \tilde{\boldsymbol{y}}\end{smallmatrix}\right] - \hat{\boldsymbol{\mu}})^T (\hat{\Sigma} + \lambda \mathbf{I})^{-1} (\left[\begin{smallmatrix}\tilde{\boldsymbol{x}}\\ \tilde{\boldsymbol{y}}\end{smallmatrix}\right] - \hat{\boldsymbol{\mu}})$
8: Pick the pixel in $\boldsymbol{y}^*$ that is closest to the center of the patch and use it for filling in the missing pixel values.
9: **end while**
10: Output: Completed Image $\mathcal{I}$

3. RESULTS

We evaluated our method on the flower dataset provided by [8]. This set contains 17 different flower categories with 80 images per category. We only used the first category for our experiment (some examples are given in Figure 2). We cropped the images so that the flower part dominates the image (this could also done automatically, e.g. using color histogram matching). This way, we can accommodate a larger set of relevant patches in a smaller dictionary. We also reduced the resolution to a half of the original for the computational efficiency.

We encoded color information by simply using RGB values of each pixel. The patches are 5×5 windows, so each patch becomes a 75 dimensional vector. When testing on a partially erased flower image, the non-erased regions of that image plus the whole region of the other 79 images were used for extracting training patches.

We sampled 100 patches from each image to construct the dictionary. We experimented with two ways of sampling, random and selective. In the selective case, when filling each

[2] Given $\boldsymbol{a}^*$, one can, in principle, recover an estimate $\tilde{\boldsymbol{y}}_{\ell^1} = \sum_{i=1}^n a_i^* \boldsymbol{y}_i$. However this ℓ^1-based estimate is sensitive to outliers in the following sense. Suppose there is an outlier patch $\boldsymbol{z}_o$ in the training data whose $\boldsymbol{x}_o$ component agrees with $\tilde{\boldsymbol{x}}$. Since $\boldsymbol{z}_o$ is not drawn from $\mathcal{N}^*$, it can potentially corrupt the estimate $\tilde{\boldsymbol{y}}_{\ell^1}$ so that it bears little resemblance to the true $\tilde{\boldsymbol{y}}$.

[3] $d(\boldsymbol{z}_i, \mathcal{S}^*)$ is the Euclidean distance between $\boldsymbol{z}_i$ and the subspace $\mathcal{S}^*$.

Fig. 2. Some Examples from a Flower Dataset

patch, a dictionary tailored to that patch is created on the fly. This was achieved by searching all 5×5 patches within each image and choosing top 100 patches that best match the patch to be filled. The similarity was measured by normalized dot product. In random sampling, 100 patches were extracted from each image at random locations. Therefore, with either sampling, the dictionary consists of 8000 vectors.

Surprisingly, the filling results from random and selective sampling were not much different. This indicates that an effective dictionary for image completion can be constructed in an extremely cheap way.

Our results on some of the images are shown in Figure 3 (The original images on the left are provided only as a reference). Observe that while the missing regions are relatively large, the completed images look very realistic; there is no blurring or blocking effect. In addition, both the texture and structure are filled in a perceptually pleasant way.

4. CONCLUSION

We proposed a new idea for image completion with large missing region. Our method relies on the assumption that similar patches form low dimensional clusters in the image space where each cluster can be approximated by a degenerate Gaussian distribution. We supported this claim by an illustrative example. We then proposed subspace identification followed by parameter estimation to obtain the most likely assignment to the missing pixel values.

The results are impressive. No blurring or blocking effects are observed, which is beyond the performance of model-driven methods [3, 2]. Both texture and structure of the missing regions were filled in a perceptually pleasant way. Note that the appearance variation among patches of the flower images does not allow repeated use of a patch for filling. In addition, the missing regions are not whole object so that the whole region could be replaced with another object. Exemplar based methods cannot cope with these problems [4, 5].

The algorithm is very simple to implement and involves solving a linear program per missing pixel. We observed that

Fig. 3. Left : Original , Middle : Input , Right : Output

random patch sampling can perform as good as the selective procedure, as long as the object category dominates the area of the training images. This allows handling a large dictionary without too much cost as in [6].

5. REFERENCES

[1] A. A. Efros and T. K. Leung, "Texture synthesis by non-parametric sampling," *ICCV*, 1999.

[2] M. Bertalmo, L. Vese, G. Sapiro, and S. Osher, "Simultaneous Structure and Texture Image Inpainting," *IEEE Transactions on Image Processing*, vol. 12, no. 8, pp. 882–889, August 2003.

[3] M. Elad, J-L. Starck, P. Querre, and D. L. Donoho, "Simultaneous Cartoon and Texture Image Inpainting Using Morphlogical Component Analysis (MCA)," *Journal on Applied and Comp. Harmonic Analysis*, vol. 19, pp. 340–358, November 2005.

[4] N. Komodakis, "Image Completion Using Global Optimization," *CVPR*, pp. 442–452, 2006.

[5] J. Hays and A. A. Efros, "Scene Completion Using Millions of Photographs," *SIGGRAPH*, vol. 26, no. 3, August 2007.

[6] J. Mairal, M. Elad, and G. Sapiro, "Sparse Representation for Color Image Restoration," *IEEE Transactions on Image Processing*, vol. 17, no. 1, pp. 53–69, January 2008.

[7] A. M. Bruckstein, D. L. Donoho, and M. Elad, "From sparse solutions of systems of equations to sparse modeling of signals and images," to appear in *SIAM Review*, 2009.

[8] M. E. Nilsback and A. Zisserman, "A visual vocabulary for flower classification," *CVPR*, 2006.