Exploiting Self-similarities for Single Frame Super-Resolution

Chih-Yuan Yang, Jia-Bin Huang, and Ming-Hsuan Yang

Electrical Engineering and Computer Science
University of California at Merced
Merced, CA 95343, USA

Abstract. We propose a super-resolution method that exploits self-similarities and group structural information of image patches using only one single input frame. The super-resolution problem is posed as learning the mapping between pairs of low-resolution and high-resolution image patches. Instead of relying on an extrinsic set of training images as often required in example-based super-resolution algorithms, we employ a method that generates image pairs directly from the image pyramid of one single frame. The generated patch pairs are clustered for training a dictionary by enforcing group sparsity constraints underlying the image patches. Super-resolution images are then constructed using the learned dictionary. Experimental results show the proposed method is able to achieve the state-of-the-art performance.

1 Introduction

Super-resolution algorithms aim to construct a high-resolution image from one or multiple low-resolution input frames [1]. They address an important problem with numerous applications. However, this problem is ill-posed because the ground truth is never known, and numerous algorithms are proposed with different assumptions of prior knowledge so that extra information can be exploited for generating high-resolution images from low-resolution ones. Existing super-resolution algorithms can be broadly categorized into three classes: reconstruction-based, interpolation-based, and example-based approaches.

Interpolation-based super-resolution methods assume that images are spatially smooth and can be adequately approximated by polynomials such as bilinear, bicubic or level-set functions [1,2,3]. This assumption is usually inaccurate for natural images and thus over-smoothed edges as well as visual artifacts often exist in the reconstructed high-resolution images. These edge statistics can be learned from a generic dataset or tailored for a particular type of scenes. With the learned prior edge statistics, sharp-edged images can be reconstructed well at the expense of losing some fine textural details.

For reconstruction-based algorithms, super-resolution is cast as an inverse problem of recovering the original high-resolution image by fusing multiple low-resolution images, based on certain assumed prior knowledge of an observation model that maps the high-resolution image to the low resolution images [4, 5].

R. Kimmel, R. Klette, and A. Sugimoto (Eds.): ACCV 2010, Part III, LNCS 6494, pp. 497–510, 2011.

Each low-resolution image imposes a set of linear constraints on the unknown high-resolution pixel values. When a sufficient number of low-resolution images are available, the inverse problem becomes over-determined and can be solved to recover the high-resolution image. However, it has been shown that the reconstruction-based approaches are numerically limited to a scaling factor of two [5].

For example-based methods, the mapping between low-resolution and high-resolution image patches is learned from a representative set of image pairs, and then the learned mapping is applied to super resolution. The underlying assumption is that the missing high-resolution details can be learned and inferred from the low-resolution image and a representative training set. Numerous methods have been proposed for learning the mapping between low-resolution and high-resolution image pairs [3, 6, 7, 8, 9, 10, 11] with demonstrated promising results.

The success of example-based super-resolution methods hinge on two major factors: collecting a large and representative database of low-resolution and high-resolution image pairs, and learning their mapping. Example-based super-resolution methods often entail the need of a large dataset to encompass as much image variation as possible [3, 6, 7, 8, 9, 10, 11] with ensuing computational load in the learning process. Moreover, the mapping learned from a general database may not be able to recover the true missing high-frequency details from the low-resolution image if the input frame contains textures that do not appear in the database. For example, the mapping function learned from low-resolution/high-resolution image pairs containing man-made objects (e.g., buildings or cars) is expected to perform poorly on natural scenes. Furthermore, the rich image structural information contained in an image is not exploited. In light of this, Glasner et al. [12] propose a method that exploits patch redundancy among in-scale and cross-scale images in an image pyramid to enforce constraints for reconstructing the unknown high-resolution image.

In [10], Yang et al. present a super-resolution algorithm by employing sparse dictionary learning on high-resolution and low-resolution images. In this algorithm, the low-resolution images are considered as a downsampled version of high-resolution ones with the same sparse codes. Using a representative set of image patches, a dictionary (or bases) is learned for sparse coding using both high-resolution and low-resolution images. Their approach performs well under the assumption that image patches of the input image are similar to the ones in the training data, e.g., similar types of images. Existing dictionary learning algorithms often operate on individual data samples without taking their self-similarity into account in searching for the sparsest solutions [13]. Observing this, Mairal et al. [14] recently propose an algorithm exploiting the intuition that similar patches in an image should admit similar sparse representation over the dictionary. By enforcing group sparsity, their experimental results on image denoising and demosaicing demonstrate improvements over existing methods.

We propose a super-resolution method that exploits self-similarities and group structural constraints of image patches using only one single input frame. In contrast to [10], our algorithm exploits patch self-similarity within the image and introduces the group sparsity for better regularization in the reconstruction

process. Compared with [14], we exploit not only the patch similarity within scale but also across scales. In addition, we are the first to show structural sparsity can be successfully applied to the image super-resolution (which is not a trivial extension). Different from [12], we enforce constraints in constructing high-resolution image patches within an image pyramid, and exploit group sparsity and generate better super resolution images. Experimental results show the proposed method is able to achieve the state-of-the-art performance for image super resolution using one single frame.

2 Proposed Algorithm

We present the proposed algorithm in this section. Our approach exploits both patch similarity across scale and group structural constraint underlying the natural images. In contrast to existing super-resolution algorithms that resort to a large data of disparate images, we show that the training patches generated directly from the input image itself facilitate finding more similar patches.

Our algorithm consists of two main steps in which we exploit self-similarities among image patches. We first generate high-resolution/low-resolution patch pairs from one single frame by exploiting self-similarities. To generate high-resolution/low-resolution patch pairs from one single frame, we create an image pyramid and build the patch pairs between corresponding high-resolution/low-resolution images. As shown in [12], the use of an image pyramid provides an effective method to generate a sufficient number of high-resolution patches from low-resolution ones.

After creating high-resolution/low-resolution patch pairs, we enforce the group sparsity constraints among similar patch pairs. The group sparsity constraints have been shown to be effective for image denoising and demosaicing [14]. In contrast to [14], we exploit not only the patch similarity within image scale but also across image scale. In addition, we show that structural sparsity can be successfully applied to the image super-resolution. We present the details of our algorithm in the following sections.

2.1 Exploiting Self-similarities to Generate Example Pairs

In the first step, we generate a set of high-resolution/low-resolution patch pairs from one single input image. These generated patch pairs are used to construct the output high-resolution image in the second step. Conventionally, the source of image pairs for example-based algorithms can be extracted from an extrinsic large dataset that encompasses a wide range of scenes or a category-specific one (e.g., [6, 10]). Alternatively, such image pairs can be extracted intrinsically from one single frame (e.g., [12]). The advantage of using extrinsic dataset is the availability of plentiful patch pairs, which may facilitate finding matches between high-resolution and low-resolution image patches. However, the drawback is the ensuing problem with large image variation inherent among image

pairs from diverse sources. Consequently these algorithms may find similar low-resolution patches from the dataset, but the paired high-resolution patches are not necessarily suitable for constructing high quality super-resolution images.

To avoid this problem, we generate patch pairs naturally bearing strong similarities directly from the input low-resolution image itself. Motivated by the observations of [12], we build image patch pairs from an image pyramid to provide highly similar patch pairs.

Assume the relationship between high-resolution image I_h, and low-resolution image I_l is

$$I_l = (I_h * B) \downarrow_s, \tag{1}$$

where $*$ is a convolution operator, B is an isotropic Gaussian kernel, and $\downarrow_s$ is a subsampling operator with scaling factor s. From an input image I_0 shown in Fig. 1, we first generate low-resolution images I_k $(k = -1, \ldots, -n)$. By well controlling the scaling factors and the variance parameters of the Gaussian kernels, it is possible to create high-resolution patches by exploiting self-similarity among the input image and generated low-resolution images. Fig. 1 illustrates the concept, and Proposition 1 states the relationship between scaling factors and the corresponding Gaussian variance parameters.

Proposition 1. *For any two downsampled images* $I_p = (I_0 * B_p) \downarrow_{s_p}$ *and* $I_q = (I_0 * B_q) \downarrow_{s_q}$ *of the image pyramid, the variances of their Gaussian kernels are related by* $\sigma_p^2 = \sigma_q^2 \cdot \log(s_p)/\log(s_q)$.

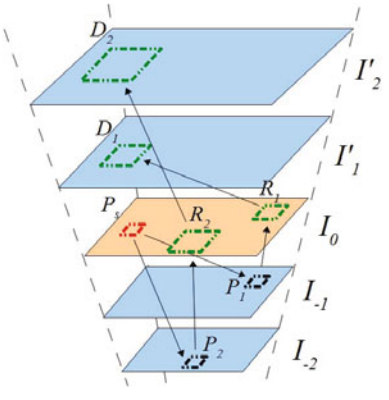


Fig. 1. Exploiting cross-scale patch redundancy in an image pyramid: I_0 is the input image. I_{-1} and I_{-2} are downsampled layers from I_0. The pixels of I'_1 and I'_2 are copied and enlarged from image patches of I_0. For a source patch P_s in I_0, several similar patches (P_1 and P_2) can be found in lower-resolution images (I_{-1} or I_{-2}). For each found patch (P_1 or P_2), a corresponding region (R_1 or R_2) in I_0 are determined. Similarly, a corresponding region (D_1 or D_2) are determined by two factors: (1) the region of source patch P_s, (2) the layer index of the found patch (-1 of I_{-1} or -2 of I_{-2}). Finally, the intensity value of R_1 are copied to D_1 with enlarged area, so as R_2 to D_2.

The proof of this proposition is presented in Appendix 1. We assume the input image I_0 is a downsampled result from an unknown high-resolution image I_k ($k \geq 1$), so that we can exploit patch similarity across scales to fill regions in I_k. We set $s_k = s^{k/n}$ ($k = -1, \ldots, -n$) where s is the expected scaling factor for final output image and n is the number of low-resolution images. This exponential setting is critical because our goal is to create high-resolution/low-resolution patch pairs for second part. Only with this setting described in Proposition 1, the Gaussian kernel variances between I_k to I_{k-n} are the same as I_n to I_0.

For a source patch P_s in the input image I_0, we use the approximate nearest neighbor algorithm [15] to find most similar patches in low-resolution images. Assume two patches are found, i.e., P_1 and P_2 in Fig. 1, their corresponding regions (R_1 and R_2) in I_0 have larger size than P_1 and P_2. Similarly any image patch P_s of I_0 can be assumed to be generated by high-resolution images with Equation 1, and the corresponding regions in the high-resolution images are D_1 and D_2. The relationship between P_k to R_k should be similar as P_s to D_k, and thus we set D_k to have the same intensity as R_k. However, P_s is not completely the same as P_k and R_k is not completely the same as D_k. We compute their weights based on their similarity with $\exp(-\|P_s - P_k\|^2/\sigma^2)$ to average the overlapped high-resolution patches, where σ controls the degree of similarity.

Denote the high-resolution images are I'_1 and I'_2 in Fig. 1, they contain many copied patches but may have some uncovered regions (i.e., some source patches in I_0 may not find similar patches in the image pyramid). We fill the uncovered area with the back projection algorithm [4] for improving image resolution. Because the blur kernels are known in our formulation, we generate high-resolution images by compensating low-resolution images

$$I_h = I''_h - (I''_l - I'_l) \uparrow_s, \tag{2}$$

where I''_h is an initial high-resolution image, I''_l is the image generated by I''_h in Equation 1, and I'_l is the images where D_k is copied to. The upsampling operator $\uparrow_s$ we use here is bicubic interpolation. If I'_l has uncovered areas, we ignore these regions and set their pixel values to to zero. We generate the initial I''_n with bicubic interpolation of I_0, and compensate I'_n to I_0. We summarize the first step to generate high-resolution/low-resolution image pairs in Algorithm 1.

2.2 Exploiting Group Self-similarities to Construct High-Resolution Images

The method presented in Section 2.1 can generate a high-resolution image H, but the resulting image may contain significant amount of noise. In this section we propose a method to further refine it by exploiting the group sparsity constraints among image patches. As the high-resolution image H and low-resolution image L are known, and the width of the Gaussian kernel σ is also known, we can generate several high-resolution images from H by the downsampling process described in Equation 1.

From the first step, we have $n + 1$ pairs of images between I_k and I_{k-n} ($k = 0, \ldots, n$). We form image pairs that every low-resolution patch in I_{k-n} has

Algorithm 1. Construct high-resolution images from single input frame

Data: Input image L, Zooming factor z, Gaussian kernel variance σ_6^2, Number of similar patches m, Similarity weight parameter σ_w, Back-projection loop number l_b

Result: High-resolution images I_1 to I_n (n is decided by z)

Set $I_0 = L$ with resolution (h_0, w_0);
for $k = 1, \ldots, 6$ **do**
  Set scaling factor $s_{-k} = (1/1.25)^k$;
  Compute convoluted image C_{-k} by convolving I_0 with a Gaussian kernel whose variance $\sigma_{-k}^2 = \sigma_6^2 * \log(k)/\log(6)$;
  Set $h_{-k} = h_0 * s_{-k}$, and $w_{-k} = w_0 * s_{-k}$ (possibly non-integer);
  Compute image I_{-k} by subsampling C_{-k} to the resolution (h_{-k},w_{-k});
for $k = 0, \ldots, 5$ **do**
  for each 5×5 patch P_s in I_{-k} **do**
    Compute the corresponding region R_s in $C_{-(k+1)}$ (boundary coordinates of R_s are usually non-integer);
    Compute Q_s by subsampling R_s into a 4×4 patch;
    Save patch pair (Q_s, P_s) into patch pair database B;
Compute number of upsampling image n=roundup($\log(z)/\log(1.25)$);
for $k = 1, \ldots, n$ **do**
  Compute image I_k's resolution as $(h_0 \times (1.25)^k, w_0 \times (1.25)^k)$;
  for each 5×5 region in I_k **do**
    Compute the corresponding region R_q in I_{k-1} (boundary coordinates of R_q are usually non-integer);
    Compute query patch Q_q by subsampling R_q into a 4×4 patch;
    Query Q_q in database B to find similar patches $Q_1 \sim Q_m$ with paired 5×5 patches $P_1 \sim P_m$ and difference value $d_t = \|Q_q - Q_t\|_2$;
    for $t = 1, \ldots, m$ **do**
      Compute patch weight $w_m = \exp(-d_t/\sigma_w)$;
      Record each patch P and weight w;
  Compute average image A by weighted average overlapped patches $\{P\}$ and weights $\{w\}$;
  Set scaling factor $s_k = 1.25^k$;
  Compute Gaussian kernel whose variance $\sigma_k^2 = \sigma_6^2 * \log(k)/\log(6)$;
  Set the initial value of back-projected image Y as A;
  for $t = 1, \ldots, l_b$ **do**
    Compute back-projected image Y respect to I_0 with Gaussian projection kernel (variance = σ_k^2), downscale and upscale factor s_k, back-projection kernel the same as projection kernel;
  Set $I_k = Y$;
  Add patch pairs (Q,P) to B from image pairs I_{k-1} and I_k as above;

Algorithm 2. Refine image through group sparse coding

Data: Image Pyramid $\{I_k\}$ $k = -6, \ldots,$ n , Zooming factor z, Gaussian kernel variance σ_6^2, Low-resolution patch size m, Cluster number c, Group sparsity threshold δ, Dictionary size d, Dictionary update loop number K

Result: Refined high-resolution image H

for *k=0, . . . , 6* **do**
- Denote low-resolution image $L_k = I_{-k}$;
- Compute expected scaling factor $s = 1.25^{-k} * z$ and index $t =$roundup$(\log(s)/\log(1.25))$;
- Denote upsampled image $I_s = I_t$;
- Set $\sigma^2 = \sigma_6^2 * 6 * \log(1.25)/\log(s)$;
- Compute I_c by convolving I_s with a Gaussian kernel whose variance is σ^2;
- Set expected resolution $(h_h, w_h) = (s * h_0, s * w_0)$ where (h_0, w_0) is I_0's resolution;
- Compute H_k by subsampling I_c to resolution (h_h, w_h) ;
- **for** each $m \times m$ patch P_i^l on L_k **do**
  - Set patch $P_i^h =$ the corresponding $mz \times mz$ patch of P_i^l on H_k;
  - Compute high-resolution feature vector $f_i^{h,r} = P_i^h - \text{mean}(P_i^h)$;
  - Compute low-resolution feature vector $f_i^{l,r}$ with gradient vectors P_i^l ;
  - Normalize feature vector $f_i^{h,r}$ to $f_i^{h,n}$ and record the norm value v_i^h;
  - Normalize feature vector $f_i^{l,r}$ to $f_i^{l,n}$;
  - Concatenate vectors $f_i^{h,n}$ and $f_i^{l,n}$ to single vector f_i^c;
  - Normalize vector f_i^c to vector y_i, and save f_i^c's norm value v_i^c;
- Cluster all $\{f_i^{l,r}\}$ by K-means clustering to get c clustering sets $\{U_j\}$, $j = 1 \ldots c$, from vector set. Each U_j contains several indexes of similar $f^{l,r}$;
- Denote Y as all vectors $\{y_i\}$ and set initial dictionary $D^0 =$ first d non-repeated y_i vectors;
- **for** *k=1 , . . . , K* **do**
  - For every cluster U_j, find the coefficient set A_j by Equation 3;
  - Denote A^k as all coefficient sets $\{A_j\}$ $j = 1, \ldots, c$ and compute residual $r^k = \|Y - D^{k-1}A^k\|_F$;
  - **for** each $m \times m$ patch P_i^l on l_0 **do**
    - Reconstruct $y_i^r = D \cdot a_i$, where a_i is y_i's coefficients in A_j;
    - De-normalized $y_i^d = y_i^r \cdot v_i^c$;
    - Reconstruct normalized high-resolution feature vector $f_i^{h,r} = \text{de-concatenate}_{high}(y_i^d)$;
    - Reconstruct de-normalized feature vector $f_i^{h,d} = f_i^{h,r} \cdot v_i^h$;
    - Reconstruct high-resolution intensity patch $P_i^{h,r} = f_i^{h,d} + \text{mean}(P_i^h)$ where P_i^h is P_i^l's corresponding $mz \times mz$ patch on H_k;
  - Compute $H^k =$ average of overlapped $P_i^{h,r}$;
  - Update dictionary D^k from D^{k-1} by Equation 4;
- Set $H = H^k$, where $k = \arg\min\{r^k\}$;

a corresponding high-resolution patch in I_k whose scaling factor is s. We use all the patch pairs to learn a dictionary with their group sparsity in order to capture the relationship among all the high-resolution or low-resolution patches, respectively.

In order to train this dictionary, we first extract features from low-resolution patches and high-resolution patches similar to [10]. The features we extract from low-resolution patch are two first-order image gradients and two second-order image gradients along horizontal and vertical axes, i.e. $[1, 0, -1]$, $[1, 0, -1]^\top$, $[-1, 0, 2, 0, -1]$, $[-1, 0, 2, 0, -1]^\top$. For each high-resolution patch, each feature vector is formed by raster scan of pixel values after subtracting the mean value of that patch.

For each high-resolution/low-resolution patch pair, we compose one concatenated feature vector. As the dimensions of low-resolution patch feature and high-resolution patch feature are different, we normalize both feature vectors independently in order to balance their contributions, before concatenating them into one single vector. All of the concatenated feature vectors are normalized to unit-norm vectors for dictionary learning with group sparsity constraints. Due to the feature design, it is possible that both of the high-resolution feature vector and low-resolution feature vector are zero. In such cases, these feature vectors are discarded.

To exploit the group similarity among patch pairs, we group pairs with similar feature vectors into clusters by K-means clustering. The feature we choose is the image gradient generated by low-resolution patches regardless of high-resolution patches because the low-resolution patches are more reliable than high-resolution patches.

With a given dictionary D, we solve the group sparse coefficients for each cluster U_i as

$$\min_{A_i} \|A_i\|_{1,2} \quad \text{s.t. } \|Y_i - DA_i\|_F \leq \sqrt{n_i}\delta, \tag{3}$$

where $\|A\|_{1,2} = \sum_{k=1}^{n} \|R^k\|_2$ and R^k is A's k-th row. In the equation above, Y_i is the column-wise feature vector in cluster U_i, n_i is the column number of Y_i, $\|\cdot\|_F$ is the Frobenius norm, and δ is a threshold controlling how similar the reconstructed feature vectors should be constructed from the original feature vectors. We use the SPGL1 package [16] to solve the above optimization problem.

As the group sparse coefficients are solved within separated cluster and the dictionary is given before solving the above equation, we need to update the dictionary for overall optimization. We denote A as the union of all coefficients A_i, and Y as the union of all feature vectors Y_i. The dictionary D is updated by the K-SVD algorithm [13],

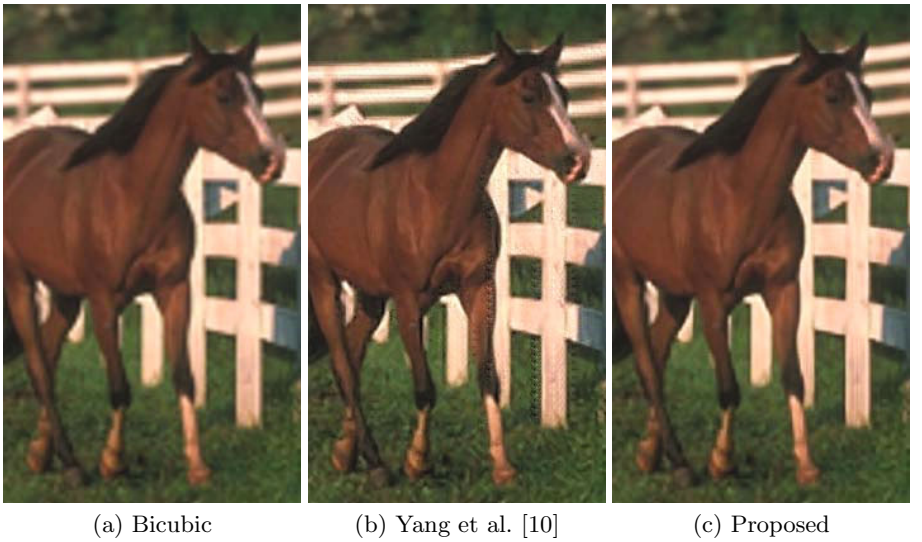
$$D = \arg\min_{D} \|Y - DA\|_F \quad \text{s.t. } \|D_j\|_2 = 1 \ \ \forall\, j, \tag{4}$$

where D_j is the j-th column of D. We iteratively solve group sparse coefficients in Equation 3 and Equation 4 until both A and D converge. The product of dictionary D and coefficient A contains the resulting feature vectors by patch similarity not only within each cluster but also among all clusters. We use these

feature vectors to generate the output high-resolution image. We summarize the process of this step in Algorithm 2.

3 Experimental Results

In this section, we describe the experimental setups and present the results using the proposed method and other algorithms. For all the experiments, we set the number of support low-resolution image $n = 6$, the number of nearest neighbor $m = 9$, variance of Gaussian blur kernel $\sigma^2 = 0.8$, scaling factor $s = 3$, and group sparse coding threshold $\delta = 0.05$. For a color input image, we convert it to YCbCr space and apply our algorithm only on luma component Y, and simply bicubic interpolate chroma components CbCr since human eyes are much more sensitive to luma rather than chroma. To compare with the state-of-the-art example-based algorithms, we use the original code provided by [10], and implement the algorithm of [12][1]. More results and MATLAB code can be found on `http://eng.ucmerced.edu/people/cyang35`



(a) Bicubic (b) Yang et al. [10] (c) Proposed

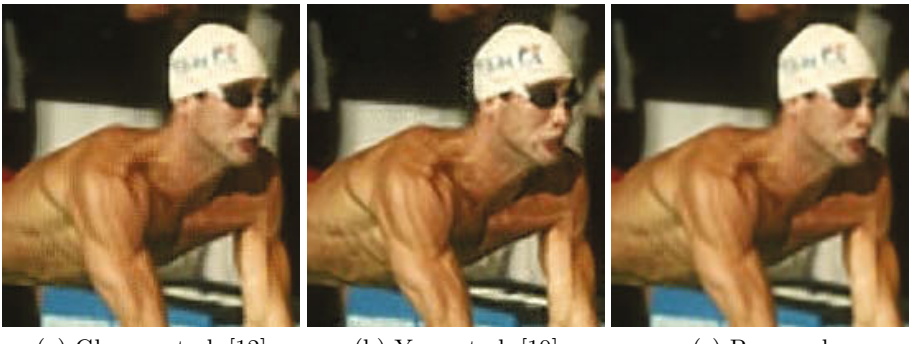
Fig. 2. Horse (results best viewed on a high-resolution display) Our result shows sharper edge than bicubic interpolation and less artifacts than [10] along fence and front legs.

We use images in the Berkeley segmentation dataset [17] for experiments. As shown in Fig. 2-7, the proposed algorithm generates shaper images with less

[1] This is based on our best efforts to implement the algorithm by Glasner et al. [12] with their help and suggestions as the authors do not release their code. The results may not be exactly the same as their reported results due to parameter settings.

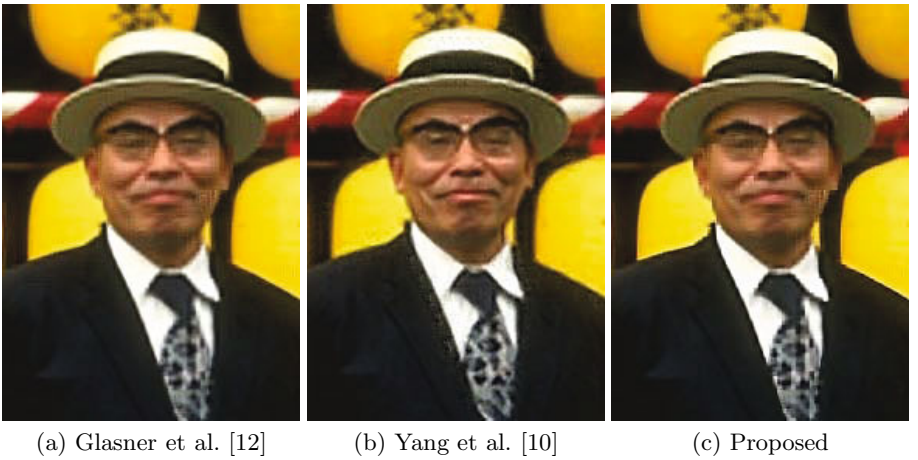
(a) Glasner et al. [12] (b) Yang et al. [10] (c) Proposed

Fig. 3. Deer (results best viewed on a high-resolution display). Compared with result generated [12], our super-resolution image has fewer artifacts (e.g., the antler region is smoother).Compared with result generated by [10], our super-resolution image has fewer artifacts (e.g., the antler region).



(a) Glasner et al. [12] (b) Yang et al. [10] (c) Proposed

Fig. 4. Swimmer (results best viewed on a high-resolution display). Compared with result generated by [12], our result has fewer artifacts (e.g., muscle and rib regions). Compared with result generated by [10], our result has fewer artifacts (e.g., around the head region).



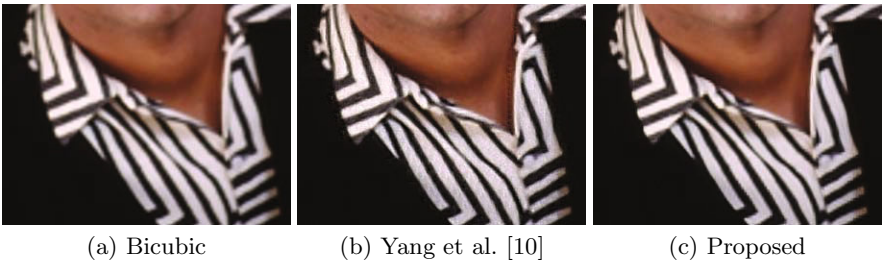
(a) Glasner et al. [12] (b) Yang et al. [10] (c) Proposed

Fig. 5. Gentleman (results best viewed on a high-resolution display). Compared with result generated by [12], our result has less artifacts (e.g., on the forehead). Compared with result generated by [10], our result has less artifacts (e.g., on the collar region).

(a) Original (b) Proposed

(c) Yang et al. [10] (d) Glasner et al. [12]

Fig. 6. Boy (results best viewed on a high-resolution display). Compared with result generated by [12], our super-resolution image has fewer artifacts (e.g., several blotches in the facial and collar regions). Compared with result generated by [10], our super-resolution image has fewer artifacts (e.g., several large blotches in the lip and contour regions).



(a) Bicubic (b) Yang et al. [10] (c) Proposed

Fig. 7. Young Man (results best viewed on a high-resolution display). Our result shows sharper edge than bicubic interpolation and less artifacts than [10] along the collar and the stripes.

artifacts than the ones obtained by the example-based super-resolution algorithm [10]. Due to space limitation, we cannot present the full resolution images in this manuscript and these images are best viewed on high-resolution displays (additional results with high resolution images can be found in the supplementary material). For example, the super-resolution images generated by [10] have more artifacts along vertical strips or regions with intensity discontinuity, e.g., the horse legs in Fig. 2, the swimmer's cap in Fig. 4, the gentleman's collar in Fig 5, and the stripes in Fig. 7. In addition, the proposed algorithm outperforms the conventional super-resolution algorithm using bicubic interpolation. The results can be explained by the assumption of example-based super-resolution algorithm which entails the need to find matches between low-resolution and high-resolution image pairs from a large training set. However, this assumption does not always hold when the training set contains disparate images which are not directly relevant to the test image (i.e., the trade-off between generality and specialty). In contrast, our algorithm does not have this problem because the training set is constructed directly from the input frame rather than a fixed dictionary.

Compared with the results generated by [12], the super-resolution images by our method also have fewer artifacts, e.g., along antlers of the deer in Fig. 3 and facial regions around eyes and mouth in Fig. 6. The success of [12] depends on whether there are plentiful similar patches in the image pyramid generated by the input frame. For images with numerous repetitive patterns (e.g., sunflower fields or butterfly wings), this algorithm tends to work well. This algorithm is not expected to perform well for an image containing a unique object, e.g., a human standing in a natural scene as shown in Fig. 6. As this unique object occupies a relatively small region, this algorithm is not able to find a sufficient number of similar patches in the natural image using the low-resolution patches from the unique object (e.g., faces), and consequently produce improper high-resolution patches (i.e., generate super-resolution image patches of foreign objects). The resulting effects are especially noticeable as these unique objects are usually the focus of attention in these images. Our proposed algorithm does not have such artifacts because we exploit both of group similarity and patch similarity rather than mere patch similarity in [12]. Although the patches on human faces are few, they can be included in similar groups to maintain the similarity in the dictionary learning. Consequently, they produce much fewer artifacts in the super-resolution images.

4 Concluding Remarks

In this paper we propose an example-based super-resolution algorithm by exploiting self-similarities using one single input image. We exploit self-similarities on two fronts: both in generating image pairs and learning dictionary with group sparsity. Experimental results show our algorithm is able to achieve the state-of-the-art super-resolution images. Our future work will focus on algorithms that take the geometrical relationships among image patches into account for efficient and effective dictionary learning.

Acknowledgments. We would like to thank Daniel Glasner and Oded Shahar for numerous discussions regarding implementation details of their super-resolution algorithm.

References

1. Park, S.C., Park, M.K., Kang, M.G.: Super-resolution image reconstruction: A technical overview. IEEE Signal Processing Magazine, 21–36 (2003)
2. Morse, B., Schwartzwald, D.: Image magnification using level set reconstruction. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 333–341 (2001)
3. Fattal, R.: Image upsampling via imposed edge statistics. In: SIGGRAPH 2007: ACM SIGGRAPH 2007 papers. ACM, New York (2007)
4. Irani, M., Peleg, S.: Improving resolution by image registration. Computer Vision, Graphics and Image Processing 53, 231–239 (1991)
5. Lin, Z., Shum, H.Y.: Fundamental limits of reconstruction-based superresolution algorithms under local translation. IEEE Transactions on Pattern Analysis and Machine Intelligence 26, 83–97 (2004)
6. Freeman, W.T., Jones, T.R., Pasztor, E.C.: Example-based super-resolution. IEEE Computer Graphics and Applications, 56–65 (2002)
7. Sun, J., Zheng, N.N., Tao, H., Shum, H.Y.: Image hallucination with primal sketch priors. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 729–736 (2003)
8. Chang, H., Yeung, D.Y., Xiong, Y.: Super-resolution through neighbor embedding. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 275–282 (2004)
9. Sun, J., Sun, J., Xu, Z., Shum, H.Y.: Image super-resolution using gradient profile prior. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (2008)
10. Yang, J., Wright, J., Huang, T., Ma, Y.: Image super-resolution via sparse representation of raw image patches. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (2008)
11. Xiong, X., Sun, X., Wu, F.: Image hallucination with feature enhancement. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (2009)
12. Glasner, D., Bagon, S., Irani, M.: Super-resolution from a single image. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 349–356 (2009)
13. Aharon, M., Elad, M., Bruckstein, A.: K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. IEEE Transactions on Signal Processing 54, 4311–4322 (2006)
14. Mairal, J., Bach, F., Ponce, J., Sapiro, G., Zisserman, A.: Non-local sparse models for image restoration. In: Proceedings of IEEE International Conference on Computer Vision, pp. 2272–2279 (2009)
15. Arya, S., Mount, D.M.: Approximate nearest neighbor queries in fixed dimensions. In: SODA 1993: Proceedings of the Fourth Annual ACM-SIAM Symposium on Discrete algorithms, pp. 271–280 (1993)
16. Berg, E.v., Friedlander, M.P.: SPGL1: A solver for large-scale sparse reconstruction (2007), `http://www.cs.ubc.ca/labs/scl/spgl1`

17. Martin, D., Fowlkes, C., Tal, D., Malik, J.: A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In: Proceedings of IEEE International Conference on Computer Vision, pp. 416–423 (2001)

Appendix

Proof of Proposition 1: Assume $s_2 = s_1^n$, where n is a natural number and the subsample operator $\downarrow$ does not decrease image quality, then $I_{in} * B_2$ is equivalent to $((((I_{in} * B_1) \downarrow_{s_1}) * B_1) \downarrow_{s_1} \cdots * B_1) \downarrow_{s_1}$.

Also assuming the subsample operator can be ignored, it implies $I_{in} * B_2 = ((I_{in} * B_1) * B_1) \cdots * B_1$ n times. Using the associative law of a convolution operator in the discrete domain, i.e., $(f * g) * h = f * (g * h)$, it follows $I_{in} * B_2 = I_{in} * (B_1 * \cdots * B_1)$, and $B_2 = B_1 * \cdots * B_1$ n times.

Because we use Gaussian blur kernel and the convolution of two Gaussian kernels is still a Gaussian kernel whose variance is the sum of the two variances, i.e., $\sigma_2^2 = n \cdot \sigma_1^2$ as $B_2 = B_1 * \cdots * B_1$. With these equation together, $\sigma_2^2 = n \cdot \sigma_1^2$ and $s_2 = s_1^n$, it follows that $\sigma_1^2 = \sigma_2^2 \cdot \log(s_1)/\log(s_2)$. □