Sparse Dictionary Learning for Edit Propagation of High-resolution Images

Xiaowu Chen[1], Dongqing Zou[1], Jianwei Li[1]*, Xiaochun Cao[2], Qinping Zhao[1], Hao Zhang[3]
[1]State Key Laboratory of Virtual Reality Technology and Systems
School of Computer Science and Engineering, Beihang University, Beijing, China
[2] Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China [3] Simon Fraser University, Canada

Abstract

We introduce a method of sparse dictionary learning for edit propagation of high-resolution images or video. Previous approaches for edit propagation typically employ a global optimization over the whole set of image pixels, incurring a prohibitively high memory and time consumption for high-resolution images. Rather than propagating an edit pixel by pixel, we follow the principle of sparse representation to obtain a compact set of representative samples (or features) and perform edit propagation on the samples instead. The sparse set of samples provides an intrinsic basis for an input image, and the coding coefficients capture the linear relationship between all pixels and the samples. The representative set of samples is then optimized by a novel scheme which maximizes the KL-divergence between each sample pair to remove redundant samples. We show several applications of sparsity-based edit propagation including video recoloring, theme editing, and seamless cloning, operating on both color and texture features. We demonstrate that with a sample-to-pixel ratio in the order of 0.01%*, signifying a significant reduction on memory consumption, our method still maintains a high-degree of visual fidelity.*

Figure 1. Sparsity-based edit propagation allows interactive recoloring (top), color theme editing (middle), and seamless cloning (bottom) with high visual fidelity. The input images (left column) contain 39M, 30M, and 70M pixels, respectively. Our edit propagation is based on 70, 210, and 45 sparse samples (sample-to-pixel ratio of 0.01%) respectively for these three experiments.

1. Introduction

One of the more frequently applied operations in interactive image/video editing involves efficiently propagating sparse user edits to an entire data domain. Edit propagation has been a well-studied problem in computer vision and it enables a variety of applications including interactive color or tonal editing, seamless cloning, and image matting. The best known edit propagation approach is affinity-based [1, 22] and so far many variants have been developed [9, 27]. Typically, the solution involves a global optimization to propagate the edits to remaining pixels.

With the increasing availability of high-resolution images and video, the demand for scalable image processing techniques also increases. Edit propagation based on global optimization inevitably incurs a high cost both in memory consumption and processing speed. For example, a naive implementation of the method of An *et al*. [1] to process an image containing 60M pixels requires about 23GB memory, which exceeds the capability of many low-end commodity computers. A recent remedy is to rely on stochastic approximation algorithms [17, 27], but at the expense of lower visual fidelity with editing artifacts.

Our key observation is that to achieve efficient and scalable edit propagation on high-resolution images or video, the operations may be performed on a *reduced* data representation. With this in mind, we resort to *sparse representations* for images. Research from signal processing has suggested that important classes of signals such as images and videos are naturally sparse with respect to fixed representa-

*corresponding author(email: lijw@vrlab.buaa.edu.cn)

tive samples, or concatenations of such samples. That is, if a proper sample dictionary is found for an image or video, it suffices to edit the sparse samples instead of all pixels, thus achieving efficiency and scalability.

Using a sparse representation, instead of propagating an edit to a whole input image or video according to relationships among all pixels, we compute a set of sparse, representative samples (or features) over the image or video, apply the edit to these samples only, and then propagate the edit to all pixels. Our approach follows the principles of sparse representations [2] and signal decorrelation using the Kullback-Leibler or KL divergence (i.e. relative entropy) to obtain an optimal sample set and the corresponding coding coefficients for the sparse representation.

By maximizing the KL divergence between each pair of samples in the dictionary, the cross correlation between samples is minimized so that the set of samples is both sparse and representative. These samples intrinsically capture a natural *basis* for the input data: each pixel in the input image is represented as a linear sum over the sampled pixels with the coding coefficients supplying the weights. Through the coding coefficients, an edit applied to the sparse samples can be mapped to an edit with respect to all pixels. However, the number of samples is far less than the number of pixels in a high-resolution image or video.

By working with the representative samples and their corresponding coding coefficients for edit propagation, we leverage a representation of the input data with higher fidelity compared to Fourier or wavelet transforms [16]. By performing the edit propagation on representative samples while keeping the coding coefficients fixed, the linear relationship between the input data and representative samples is preserved. In turn, this ensures that the input data is faithfully represented by the representative samples.

Our main contributions are: 1) an efficient edit propagation method for high-resolution images or video via sparse dictionary learning; 2) an automatic scheme based on K-L divergence to obtain a compact and discriminative sample set for edit propagation; 3) application of the learned sparse dictionary to several frequently encountered image/video processing tasks including interactive recoloring, color theme editing, and seamless cloning. We demonstrate through numerous experiments that our method significantly reduces memory cost while still maintaining high-fidelity editing results and avoiding the artifacts encountered by previous methods [17, 27].

2. Related work

Edit propagation is a proven and well-adopted technique for image/video editing. The work of Levin *et al*. [15] provided the first framework for propagating user edits. Later, this work was extended by [1, 17, 27] to ensure that the edits can be propagated for fragmented regions according to properly defined affinities between all pixels. Farbman *et al*. [9] employed the diffusion distance to measure affinities between pixels. However, these methods do not adequately constrain pixels in blended regions, often causing halo artifacts. Chen *et al*. [6] addressed this problem by preserving the manifold structure underlying the image pixels.

Also related are a large body of works on edge-aware or edge-centric image processing [5, 11] and image upsampling techniques [12]. Generally speaking, preserving affinity relations among all pixels is expensive, and prohibitive for high-resolution data. It is difficult for existing methods to propagate edits with a limited time or memory budget while ensuring visual fidelity of the editing results.

In the past few years, various adaptations of sparse representations have been applied to the problem domain of image analysis and understanding. Yang *et al*. [29] and Allen *et al*. [28] used sparse representations for image super-resolution by assuming that the high-resolution image patches have a sparse representation with respect to that of low-resolution images. Julier *et al*. [21] developed a supervised framework to learn multiscale sparse representations of color images and video for image and video denoising and inpainting. Yan *et al*. [25] proposed a semi-coupled dictionary learning model for cross-style image synthesis.

Sparse representations have also been applied to face recognition [26], image background modeling [3], and image classification [20]. More works along those lines can be found in a recent comprehensive survey [8]. We adopt sparse representations for edit propagation for the first time and demonstrate its effectiveness in several key applications when applied to high-resolution images and video.

3. Sparsity-based edit propagation

Given a high-resolution image, we first obtain an overcomplete dictionary D_{init} via online dictionary learning [19]. Then we utilize the KL divergence (i.e. relative entropy) [14] as an optimization objective to turn the initial dictionary D_{init} into a sparser and more representative sample set. An image edit is applied to the samples and propagated from the samples to all pixels in the image.

3.1. Dictionary initialization

Let x_i represent a pixel i in the input image X, where x_i can be a pixel color or any other editable feature, and N_{init} represent the number of samples in the overcomplete dictionary $D_{init} = [d_1, d_2, ..., d_{N_{init}}]$. D_{init} and the corresponding coefficients α for input X are obtained by minimizing

$$\sum_i \|x_i - D_{init}\alpha_i\|_2^2 + \lambda \sum_i \|\alpha_i\|_0 \,, \qquad (1)$$

where the first term gives the reconstruction error and the l_0 norm $||\alpha_i||_0$ counts the number of nonzero entries in the

coefficient α_i. Minimizing Equation (1) can be transformed into an iterative sparse coding problem [18, 19], which can be solved efficiently.

3.2. Dictionary optimization

The initial dictionary D_{init} is overcomplete with most samples redundant. A compact and discriminative dictionary should capture the main information entropy of D_{init} with the chosen samples decorrelated; this essentially encourages signals having similar features to possess similar sparse representations. We use the KL-divergence to measure the difference between two samples so as to obtain a compact and discriminative sample set D^*.

In information theory, the KL-divergence is a non-symmetric measure of the difference between two probability distributions P and Q,

$$KL(P\|Q) = \sum_i P(i) \ln \left(\frac{P(i)}{Q(i)} \right). \tag{2}$$

Although the KL-divergence is often intuited as a distance metric, it is not a true metric, e.g., due to its asymmetry. We utilize a symmetrized version of KL-divergence to estimate the difference between two samples,

$$KL_s(P\|Q) = KL(P\|Q) + KL(Q\|P). \tag{3}$$

We optimize the initial dictionary D_{init} according to the symmetric KL-divergence (3). Obviously, two similar signals in input X would use similar samples in D_{init} to make the sparse decompositions. Thus the similarity between two signals can be measured by comparing the corresponding sparse coefficients. In the same way, we can estimate the similarity of two samples in D_{init} by comparing the number of signals using them, and their contributions in the sparse decomposition [24]. Specifically, each sample $d_i \in D_{init}$ maps all the input signals to its corresponding row of coefficients $\alpha_{d_i} = [\alpha_{i1} ... \alpha_{iN}]$. For each pair of samples d_i and d_j, we define the dissimilarity between them as $R(d_i, d_j) = KL_s(\alpha_{d_i} \| \alpha_{d_j})$. To avoid taking logarithms of zero, we smooth the sparse coefficients by adding a small constant term $\varepsilon = 10^{-16}$ and normalizing the distribution.

Our dictionary optimization problem is therefore to find a sample set D^* to ensure that the KL-divergence between each sample pair in D^* is maximized:

$$arg \max_{D^*} \sum_{d_i, d_j \in D^*} R(d_i, d_j), D^* \subseteq D_{init}. \tag{4}$$

Since obtaining the global optimum is difficult, we employ a greedy heuristic which resembles farthest point sampling. Given D_{init}, we start with $D^* = \{d_0\}$, where d_0 is selected from D_{init} by $\arg\max_{d_0} \sum_{d_i \in D_{init}} R(d_0, d_i)$, implying that it has the maximum dissimilarity with other samples. Then we iteratively select the next best sample d^* from $D_{init} \setminus D^*$ which is the most dissimilar with D^*, *i.e.*, $\arg\max_{d^*} \sum_{d_i \in D^*} R(d^*, d_i)$, adding d^* to D^* and removing the samples from D_{init} whose dissimilarities with d^* are less than a threshold τ, until $D_{init} = \emptyset$. We use the threshold τ to control the compactness of D^*.

We finally update the coefficients α according to the optimized sample set D^* to obtain the corresponding coefficients α^* for D^* with dictionary learning [19].

3.3. Edit propagation

We use $Y = [y_1, y_2, ..., y_N]$ to represent the corresponding edit propagation result from input X, where N denotes the number of pixels in X. We denote the resulting samples after edit propagation with respect to D^* by $\tilde{D} = [\tilde{d_1}, \tilde{d_2}, ..., \tilde{d_n}]$. Since the input image can be well represented as a linear combination of the samples D^* and coefficients α^*: $x_i \approx D^* \alpha_i^*$, it suffices to edit the sparse set of representative samples D^* in place of all pixels, leading to significant cost savings.

Suppose that G represents the user edits (e.g., user scribbles) for a subset $\mathcal{S}$ of pixels. We propagate the edits via sparse representation by minimizing the energy,

$$\arg\min_{(\tilde{d_i}, y_i)} E = \gamma_1 E_1 + \gamma_2 E_2 + \gamma_3 E_3, \tag{5}$$

where

$$E_1 = \sum_{i \in \mathcal{S}, g_i \in G} \left(\tilde{d_i} - g_i \right)^2; \; E_2 = \sum_i (\tilde{d_i} - \sum_{\tilde{d}_{ij} \in N_i} w_{ij} \tilde{d}_{ij})^2$$

$$E_3 = \sum_{i, \alpha_i^* \in \alpha^*} \left\| y_i - \tilde{D} \alpha_i^* \right\|_2^2,$$

and N_i is a set containing the neighbors of d_i. The energy term E_1 ensures that the final representative samples are close to the user specified value g_i. E_2 seeks to maintain the relative relationship between the samples during edit propagation. Chen *et al.* [6] proposed a manifold preserving edit propagation method which maintains the relative relationship between pixels by using locally linear embedding during edit propagation. Here, we follow the same principle to compute the local linear relationship for all representative samples. Specifically, we obtain the relationship weights w_{ij} by minimizing

$$\sum_{i=1, d_i^* \in D^*}^{n} \left\| d_i^* - \sum_{j=1}^{K} w_{ij} d_{ij}^* \right\|^2, \tag{6}$$

subject to the constraint $\sum_{j=1}^{K} w_{ij} = 1$, where $\{d_{ij}^* | j = 1, 2, ...K\}$ are the K nearest neighbors (kNN) of d_i^*.

Finally, the last energy term E_3 is a fidelity term, which constrains the final result to be faithfully represented by the sparse representative samples after edit propagation.

Equation (5) is a quadratic function in $\tilde{D}$ and Y, which can be minimized by solving the two equations

$$(\Lambda_1+{\alpha^*}^T\Lambda_3\alpha^*+\Lambda_2(I-W)^T(I-W))\tilde{D}-\Lambda_3\alpha^*Y=\Lambda_1 G \quad (7)$$

and

$$\left\|Y-\tilde{D}\alpha^*\right\|_2=0, \quad (8)$$

where I is the identity matrix, Λ_1, Λ_2, and Λ_3 are diagonal matrixes and G is a vector with

$$\Lambda_1(i,i)=\begin{cases}\gamma_1 & i\in\mathcal{S}\\ 0 & otherwise\end{cases} \qquad G_i=\begin{cases}g_i & i\in\mathcal{S}\\ 0 & otherwise\end{cases}$$

and $\Lambda_2(i,i)=\gamma_2$ and $\Lambda_3(i,i)=\gamma_3$. We set $\gamma_1=1000$, $\gamma_2=5$, and $\gamma_3=1$ in all our experiments. Equations (7) and (8) can be solved efficiently, as shown in Algorithm (1), which gives a summary of our edit propagation scheme.

Algorithm 1 Sparsity-based edit propagation.

Require: $\mathbf{X}=[x_1,...,x_N]$ (input image/video)
Ensure: $\mathbf{Y}=[y_1,...,y_N]\approx\tilde{D}\alpha^*$ (resulting image/video)

1: Calculate the initial samples D_{init} from $\mathbf{X}$ using an online dictionary learning method [19].
2: Calculate the source representative samples D^* from D_{init} using the KL-divergence method.
3: Set g_i in Equation (5) as the specified edits by the user according to specific applications; see Section 4.
4: Propagate edits across D^* and obtain result samples $\tilde{D}$.
5: **for** each pixel x_i of input data $\mathbf{X}$ **do**
6: Calculate coefficient α_i^* of x_i by Equation (1).
7: Calculate the result $y_i=\tilde{D}\alpha_i^*$.
8: **end for**
9: **return** $\mathbf{Y}$ (resulting image/video).

Cost analysis. We use the online dictionary learning method in [19] (line 1) for initial dictionary construction; this scheme can manage a large dataset with low memory and computational costs. In line 2, we calculate the KL-divergence of each pair of samples in D_{init}. Thus to save D_{init}, we need a space complexity of $O(N_{init}d)$, where d is the dimension of each sample (vector). In line 3, the user edit g_i is a small part of the source representative samples and does not consume additional space. The memory consumption of the edit propagation in line 4 is subject to the size n of D^* and the size K of kNN. In the for loop from lines 5 to 8, we solve the coefficient α^* one by one, thus the memory space used here is negligible. From the analysis above, we can see that the space complexity of the algorithm is $max\{O(N_{init}d),O(Knd)\}$.

Parameters. The only two parameters to set during sample construction are N_{init}, the number of samples in D_{init} and the threshold τ; the method is otherwise automatic. For all experiments shown in the paper, the same parameter values $N_{init}=500$ and $\tau=20$ are used. Therefore, $n\leq N_{init}=500$ is determined by the sample optimization progress and varies depending on the dataset processed. The other parameters are also fixed: $K=10$ and $d=3$ for RGB color features and $d=8$ for texture features.

4. Applications

We now describe how to apply sparsity-based edit propagation to various classical images and videos editing tasks.

Video object recoloring. When a user draws scribbles with desired colors at some pixels in a frame, our method only propagates the edit to the representative samples we compute; see Algorithm (1). The whole video will change with the representative samples synchronously according to the coding coefficients, where the user-edited pixels are replaced by the representative samples. Specifically, in Equation (5), we set $\mathcal{S}$ as the nearest colors in the dictionary set D^* to that of pixels covered by the user strokes, using the Euclidean distance in RGB space, and g_i as the strokes' color with black enforcing color preservation.

Video color scheme editing. For this task, the step for learning representative samples is the same as that for video object recoloring. Instead of specifying scribbles, the user utilizes a color theme to edit an input video or image. We classify the representative samples into several clusters according to the probabilistic mapping provided in [4], and calculate the mean color of each cluster to generate the color theme T of the representative samples. Then the user can edit T to a desired color theme R to alter the color scheme of the whole video or image. Specifically, in Equation (5), we set $\mathcal{S}$ as the color theme T. For each color $i\in\mathcal{S}$, g_i denotes the corresponding color value in R.

Image cloning. Our edit propagation method can also be employed to paste a source image or video patch into another scene seamlessly. We treat image cloning as a membrane interpolation problem [10]. The pixel colors along the boundary of the source image constitute the source dictionary in our method. The target dictionary consists of the color differences on the boundary between the source and target images. We compute the coding coefficients which can reconstruct the source image patch by the source dictionary. Then we use the coding coefficients and the target dictionary to generate the membrane. By adding the membrane to the target image, we achieve the final cloning result. Specifically, in Equation (5), we set $\mathcal{S}$ as all the colors on the blending boundary and g_i as the color differences on the boundary between the source and target images.

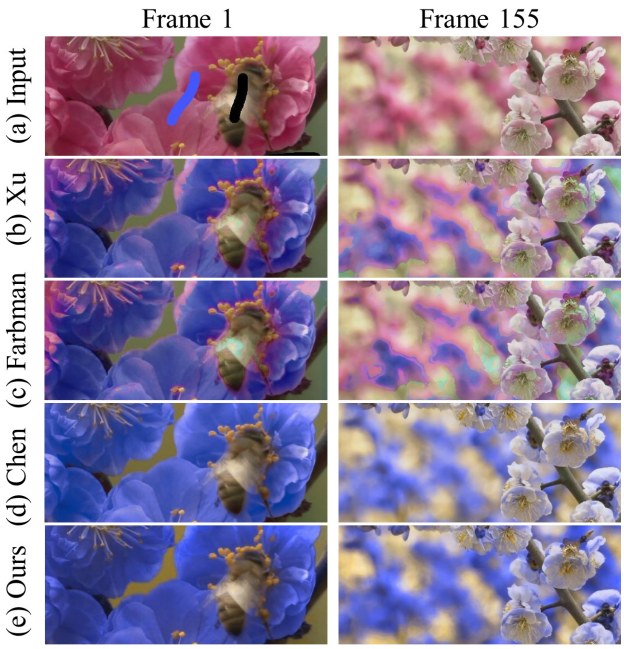

Figure 2. Video object recoloring: comparison to (b) [Xu *et al*. 2009], (c) [Farbman *et al*. 2010] and (d) [Chen *et al*. 2012] on two example frames. Input video (a) "*Peony*" contains 30M pixels. [Xu *et al*. 2009] and [Farbman *et al*. 2010] generate halo artifacts at flower boundaries. Our results (e) are more natural as well as more efficient than [Chen *et al*. 2012].

5. Experimental results

We demonstrate our method on applications mentioned above and compare to state-of-the-art edit propagation methods such as [27, 9, 6] to show the space efficiency and visual fidelity achieved by our sparsity-based approach. The authors of these three papers kindly share their code with us. We also demonstrate the efficiency of our method by operating on higher dimensional features, such as textures, within our framework for video editing. Our experiments were performed on a PC with Intel Core i7 Quad processor (3.40GHz, 8 cores) and 4 GB memory.

Due to space limitations, we are only able to show selected results in the paper for a demonstration. We would like to point out that these results are representative of the efficiency and advantages that our method would offer, in comparison to previous state-of-the-art approaches. More results can be found in the supplementary material.

Object recoloring in video. Figure 2 shows a comparison, where results from competing methods were obtained under suggested parameter settings in the original papers [27, 9, 6]. The user applied strokes to one frame of the video to change the red flowers to blue. One can see that the foreground flowers are not edited appropriately in the results of Xu *et al*. [27] and Farbman *et al*. [9]: halo artifacts appear around flower boundaries. In comparison, our method and the method of Chen *et al*. [6] produced more natural results, while our method incurs a much smaller memory usage; see Table 1. Specifically, for the input video *"Peony"* containing 30M pixels, our method takes 6.5MB RAM, while Xu *et al*. [27], Farbman *et al*. [9] and Chen *et al*. [6] required 350MB, 825MB, and 65MB RAM, respectively.

Performance comparison for video object recoloring.			
	Data	*Peony*	*Canoeing*
	Resolution	30M	96M
Xu *et al*.	Memory	350MB	1110MB
	Timing	43s	132s
Farbman *et al*.	Memory	825MB	2400MB
	Timing	49s	141s
Chen *et al*.	Memory	65MB	220MB
	Timing	480s	825s
Ours	Samples num.	222	210
	Memory	6.5MB	5.8MB
	Timing	125s	425s

Table 1. Memory and timing comparison between our method and those of [Xu *et al*. 2009], [Farbman *et al*. 2010] and [Chen *et al*. 2012] for object recoloring in video.

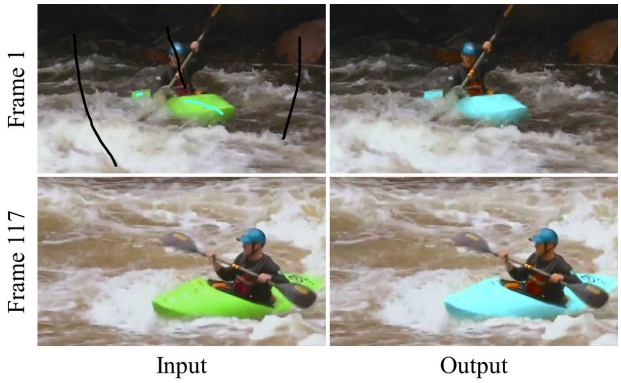

Figure 3. More visual results for video object recoloring. Shown on the left are some example frames of "*Canoeing*". The right shows corresponding recoloring results. The video contains 96M pixels. Our method only takes up 5.8MB memory space.

Figure 3 shows a result for high-resolution video sequences. The input video *"Canoeing"* contains 96M pixels. We use only 210 samples in our sparsity-based edit propagation, resulting in only 5.8MB RAM usage. The running time for this example is 425 seconds.

Video color theme editing. Figure 4 shows a comparison with the method of Chen *et al*. [6] on color theme editing. The input video *"Big Bunny"* has 468 frames whose resolution is 1920×1080; it contains 970M pixels in total. We compute 240 sparse representative samples automatically for edit propagation. Visually, we can obtain similar results with the method of [6]. However, our method takes 8MB

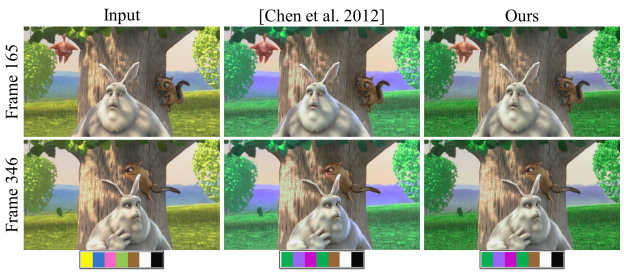

Figure 4. Comparison with [Chen *et al*. 2012] on color theme editing. The first column contains sample frames from the original video "*BigBunny*". The next two columns are results from [Chen *et al*. 2012] and our method, respectively.

Figure 5. Results on video color theme editing. Shown on the left are some example frames of the video "*Daisy*". Right are the corresponding recolored results.

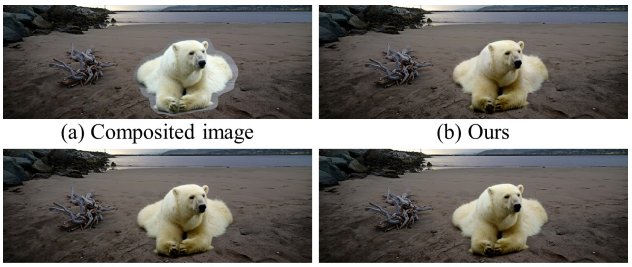

Figure 6. Comparison with previous cloning methods. (a) is the composited image and (b) is our result; (c) and (d) were obtained by [Pérez *et al*. 2003] and [Farbman *et al*. 2009], respectively. The three results are visually indistinguishable.

memory space for the task while Chen *et al*. [6] consumes about 1,000MB RAM; see Table 2.

Figure 5 shows another result from video color theme editing. The target video *"Daisy"* contains 129 frames and 30M pixels. Our method takes up 6MB RAM and 212 seconds to propagate the edits to the whole video.

Image cloning. Figure 6 compares one of our image cloning results to Pérez *et al*. [23] and Farbman *et al*. [10]. While it is difficult to provide a quantitative evaluation

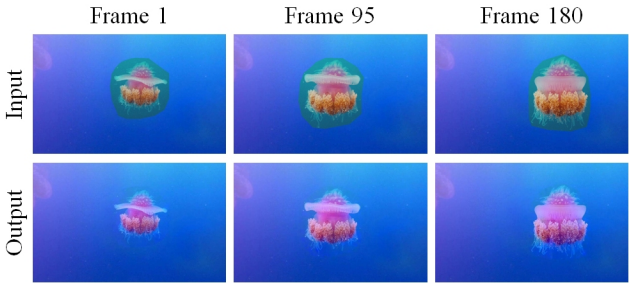

Figure 7. Visual results for video cloning. The first row shows the composited result without blending after scaling the input objects while the second row is the blending result with our method.

Comparison for video color theme editing.			
	Data	*Daisy*	*Big Bunny*
	Resolution	30M	970M
Chen *et al.*	Memory	80MB	1000MB
	Timing	650s	3200s
Ours	Samples num.	212	240
	Memory	6MB	8MB
	Timing	212s	630s

Table 2. Memory and timing comparison between our method and [Chen *et al*. 2012] for video color theme editing.

Figure 8. Edit propagation results with and without texture features. (a) input image; (b) edit result with only color features; (c) result with additional texture features. Our method automatically selects 165 samples from the input to arrive at the result (c).

for seamless cloning, we find all three results to be visually pleasing and almost indistinguishable. However, our method only uses 107 samples, translating to a 3MB memory footprint, for the shown example, which is a lot more memory efficient than the other methods. Figure 7 shows our results for video cloning. A video clip of a jellyfish was blended into another video as shown in the first row. The second row shows the blended results. Our method achieves temporally consistent video blending by using only 45 samples and the task was executed in180 seconds.

Texture features. Our method is applicable to higher dimensional features such as textures, as shown in Figure 8. We calculate the texture features for the input by using the method in [7]. It is easy to detect that the input image (a) contains two different textures with only one color. Editing based on color features only would lead to the result in (b) in which all regions are recolored to red, while these regions exhibit two distinctive texture patterns. Figure 8(c) shows the importance of taking into account texture features. Our

Figure 9. Comparison between using and without using KL-divergence in sample construction. The three columns from left to right: objects recoloring, color theme editing, and image cloning. Artifacts appear when KL-divergence optimization was not applied, while the counterpart produces more natural results.

method automatically selected 165 samples, requiring only 11MB memory, for this shown example.

Test on KL-divergence. Figure 9 shows a comparison between results obtained using vs. without using KL-divergence in representative sample construction, for all three applications. Our method automatically selects 222, 210, and 160 samples for these three applications, respectively. It is evident that with the same number of samples, the KL-divergence based scheme achieves better results.

We tested our method on 1,053 images to evaluate the reconstruction error, which is computed using Sum of Absolute Differences. With KL divergence optimization, the error is 0.0073, while for the method of Mairal *et al.* [19] without sample optimization, the error is at 0.0111.

We also compared KL divergence against Maximum Mutual Information (MMI) [13], a well-known information entropy theory, in sample selection for reconstruction and edit propagation. As shown in Figure 10, given the same number of initial samples, our method generated a natural editing result while artifacts emerged in the result produced with MMI. This demonstrates that the use of KL-divergence tends to capture more representative samples than other alternatives of information entropy, such as MMI.

Varying size of sample set. We evaluate our method in Figure 11 with different number of representative samples. Desirably, the number of representative samples should be just large enough to best represent the input data. For this example, our method automatically selected 70 samples and the result shown in (c) is natural. If we randomly eliminate 20 samples from the representative set, the propagation result exhibits artifacts along the boundary of the blossom, as shown in (b). On the other hand, the propagation result does not improve if we add more samples from the initial dictionary, as shown in (d).

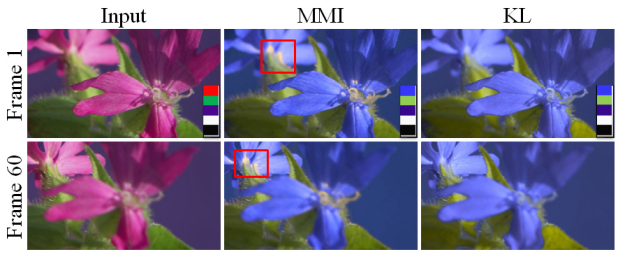

Figure 10. Comparison between using KL divergence and using maximum mutual information (MMI) for images editing. Note the artifacts with the use of MMI.

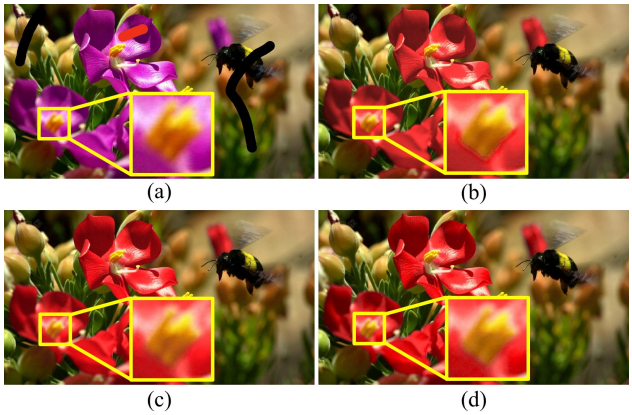

Figure 11. Varying the number of representative samples. (a) input image; (b) 50 samples — too few with small artifacts near the blossom (see inset); (c) 70 samples using KL-divergence — adequate. (d) 80 samples — more samples do not improve results.

Figure 12. Several editing results on consumer video or photos.

Finally, we show a few editing results obtained on consumer video or photos to further demonstrate our method on complex image data, as shown in Figure 12.

6. Conclusion and discussion

We develop a novel sparsity-based edit propagation method for high-resolution images or video. Instead of propagating an edit to the whole image or video according to relationships among all pixels, by using sparse dictionary learning, we derive a set of representative samples (or features), apply the edit to these samples only, and then propagate. Our method significantly improves the memory

efficiency while maintaining a high-degree of visual fidelity in the editing results. Several frequently encountered applications enabled by our proposed method, including interactive image/video recoloring, color theme editing, and image cloning, demonstrate the effectiveness of our approach.

One limitation of our current method is that it is not designed to alter only one object: if two far away objects in an image share the same features, editing one object would change the other. As Figure 11 shows, two flowers are recolored into red even if only one is scribbled by the user. Although some video cutout methods can be applied as a remedy, it is labor-intensive and time-consuming for users to perform additional operations. For future work, we would like to incorporate spatial information for edit propagation and investigate further means of optimization, e.g., by employing GPU processing, to improve efficiency.

Acknowledgement. We thank the reviewers for their valuable feedback. This work is supported in part by grants from NSFC (61325011), 863 program (2013AA013801), SRFDP (20131102130002), BUAA (YWF-13-A01-027), and NSERC (611370).

References

[1] X. An and F. Pellacini. Appprop: all-pairs appearance-space edit propagation. *ACM Trans. Graph.*, 27(3):40:1–40:9, Aug. 2008. 1, 2

[2] A. M. Bruckstein, D. L. Donoho, and M. Elad. From sparse solutions of systems of equations to sparse modeling of signals and images. *SIAM Rev.*, 51(1):34–81, Feb. 2009. 2

[3] V. Cevher, A. Sankaranarayanan, M. F. Duarte, D. Reddy, R. G. Baraniuk, and R. Chellappa. Compressive sensing for background subtraction. In *Proc. ECCV*, pages 155–168, Berlin, Heidelberg, 2008. 2

[4] Y. Chang, S. Saito, and M. Nakajima. A framework for transfer colors based on the basic color categories. In *Proc. of Computer Graphics International*, pages 176–183, 2003. 4

[5] J. Chen, S. Paris, and F. Durand. Real-time edge-aware image processing with the bilateral grid. *ACM Trans. Graph.*, 26(3), 2007. 2

[6] X. Chen, D. Zou, Q. Zhao, and P. Tan. Manifold preserving edit propagation. *ACM Trans. Graph.*, 31(6):132:1–132:7, Nov. 2012. 2, 3, 5, 6

[7] J. G. Daugman. Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters. *Optical Society of America A*, 2(7):1160–1169, 1985. 6

[8] M. Elad, M. A. T. Figueiredo, and Y. Ma. On the Role of Sparse and Redundant Representations in Image Processing. *Proc. the IEEE*, 98(6):972 –982, 2010. 2

[9] Z. Farbman, R. Fattal, and D. Lischinski. Diffusion maps for edge-aware image editing. *ACM Trans. Graph.*, 29(6), 2010. 1, 2, 5

[10] Z. Farbman, G. Hoffer, Y. Lipman, D. Cohen-Or, and D. Lischinski. Coordinates for instant image cloning. *ACM Trans. Graph.*, 28(3), 2009. 4, 6

[11] R. Fattal, R. Carroll, and M. Agrawala. Edge-based image coarsening. *ACM Trans. Graph.*, 29(1), 2009. 2

[12] J. Kopf, M. F. Cohen, D. Lischinski, and M. Uyttendaele. Joint bilateral upsampling. *ACM Trans. Graph.*, 26(3), July 2007. 2

[13] A. Krause, A. Singh, and C. Guestrin. Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies. *J. Mach. Learn. Res.*, 9:235–284, June 2008. 7

[14] S. Kullback and R. A. Leibler. On information and sufficiency. *Ann. Math. Statist.*, 22(1):79–86, 1951. 2

[15] A. Levin, D. Lischinski, and Y. Weiss. Colorization using optimization. *ACM Trans. Graph.*, 23(3), 2004. 2

[16] M. S. Lewicki and T. J. Sejnowski. Learning overcomplete representations. *Neural Comput.*, 12(2):337–365, 2000. 2

[17] Y. Li, T. Ju, and S.-M. Hu. Instant propagation of sparse edits on images and videos. *Comput. Graph. Forum*, 29(7):2049–2054, 2010. 1, 2

[18] J. Mairal, F. Bach, and J. Ponce. Task-driven dictionary learning. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(4):791–804, 2012. 3

[19] J. Mairal, F. Bach, J. Ponce, and G. Sapiro. Online learning for matrix factorization and sparse coding. *J. Mach. Learn. Res.*, 11:19–60, Mar. 2010. 2, 3, 4, 7

[20] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman. Discriminative learned dictionaries for local image analysis. In *CVPR*, pages 1–8, 2008. 2

[21] J. Mairal, G. Sapiro, and M. Elad. Learning multiscale sparse representations for image and video restoration. *Multiscale Modeling & Simulation*, 7(1):214–241, 2008. 2

[22] F. Pellacini and J. Lawrence. Appwand: editing measured materials using appearance-driven optimization. *ACM Trans. Graph.*, 26(3), 2007. 1

[23] P. Pérez, M. Gangnet, and A. Blake. Poisson image editing. *ACM Trans. Graph.*, 22(3):313–318, 2003. 6

[24] I. Ramirez, P. Sprechmann, and G. Sapiro. Classification and clustering via dictionary learning with structured incoherence and shared features. In *CVPR*, pages 3501–3508, 2010. 3

[25] S. Wang, L. Zhang, L. Y., and Q. Pan. Semi-coupled dictionary learning with applications in image super-resolution and photo-sketch synthesis. In *Proc. CVPR*, 2012. 2

[26] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma. Robust face recognition via sparse representation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 31(2):210–227, 2009. 2

[27] K. Xu, Y. Li, T. Ju, S.-M. Hu, and T.-Q. Liu. Efficient affinity-based edit propagation using k-d tree. *ACM Trans. Graph.*, 28(5), 2009. 1, 2, 5

[28] A. Y. Yang, J. Wright, Y. Ma, and S. S. Sastry. Unsupervised segmentation of natural images via lossy data compression. *Comput. Vis. Image Und.*, 110(2):212–225, 2008. 2

[29] J. Yang, J. Wright, T. S. Huang, and Y. Ma. Image super-resolution via sparse representation. *Trans. Img. Proc.*, 19(11):2861–2873, Nov. 2010. 2