Epitome Transform Coding: Towards Joint Compression of a Set of Images

ABSTRACT

In this paper, we present epitome transform coding, an approach to joint compression of a set of images. The first step of the epitome transform coding is to construct a compact epitome image representation for an image collection, where each image block extracted from each image in the collection has a corresponding prototype block in the epitome. Then the indices of the prototype in the epitome image as well as the residue (a.k.a, the difference between the image block and its corresponding prototype) are compressed, respectively. As demonstrated by our extensive experimental evaluation, the proposed epitome transform coding can effectively exploit the repetitive patterns and hence redundancies across the images in the collection for better compression. To date, such cross image redundancy has not been effectively used to improve compression. To the best of our knowledge, the epitome transform coding represents the first of its kind for joint compression of a collection of images. When compared with JPEG, which compresses each image independently, the epitome transform coding shows clearly improvement in terms of rate-distortion when compressing a collection of images.

Categories and Subject Descriptors

I.4.9 [**Image processing and Computer Vision**]: Applications; I.4.5 [**Image processing and Computer Vision**]: Reconstruction; E.4 [**Coding And Information Theory**]: Data Compaction and Compression

General Terms

Algorithm, Experimentation, Performance

Keywords

Image Compression, Image Epitome, Reconstruction, Residual

1. INTRODUCTION

In this era of digital Big Data, over 1.8 billion photos are uploaded every day in 2014 [12], via social networking platforms such as Instagram and Facebook, which urgently motivates efficient image compression for bandwidth saving and economic storage for transmitting. A large portion of the photos shared on the Internet, contain highly similar textured regions, including the lawn, the sky, and the facade of skyscrapers, etc. For example, the photos captured by tourists in famous landmarks from different perspectives, contain the similar objects and people. Thus it becomes favorable to efficiently exploit the similarity and repetition within and across them, for obtaining better compression. Based on this motivation, we present an epitome transforming coding towards joint compression of a collection of images, by taking advantage of the spatially textural re-occurrences and redundancies among them.

Our proposed image compression method, is developed based on epitome transform coding. It starts with the construction of a single compact and generative epitome of all the images in the collection [6, 9], where each fixed-size rectangular image patch, extracted from each image in the collection, is transformed to a corresponding prototype block in the epitome. Then the indices of the prototype in the epitome, as well as the residual (i.e., the difference between the image block and its corresponding prototype), are further compressed via lossy and/or lossless compression schemes. Specifically, in the first step of our approach, the Expectation Maximization (EM) algorithm is applied to learn a condensed epitome, which contains high-order statistics of the texture and shape properties of the images in the collection. During learning, a patch extracted from the training images, is mapped into (i.e., appears) somewhere in the epitome with a local maximal probability, and the probability and the associated transform map are iteratively updated, respectively. After finishing epitome learning, those images are reconstructed directly from the epitome and transform maps. To achieve good compression ratio for our method, the residues is encoded via appropriate compression techniques. To sum up, our approach based on epitome transform coding, first learns the epitome and the associated transform mapping, then implements reconstruction, and encodes and decodes epitome, transform maps, and residuals, in order to gain a good performance evaluated by rate-distortion.

Traditional compression techniques still focus on a single image compression. For example, JPEG [19] and JPEG2000 [18] have demonstrated great performance when compressing a single image. Since they only focus on the local self-similarity within one image, they cannot provide superiority when compressing a set of similar image, compared with a single image. Unlike JPEG and JPEG2000, which compress each image independently, our method exploits the similarity across a set of images to gain better joint compression ratio. For video compression techniques, like MPEG, exploit block-based motion vectors between successive frames to make a prediction from frame-to-frame. However,

MM '15 26-30 October 2015, Brisbane, Australia

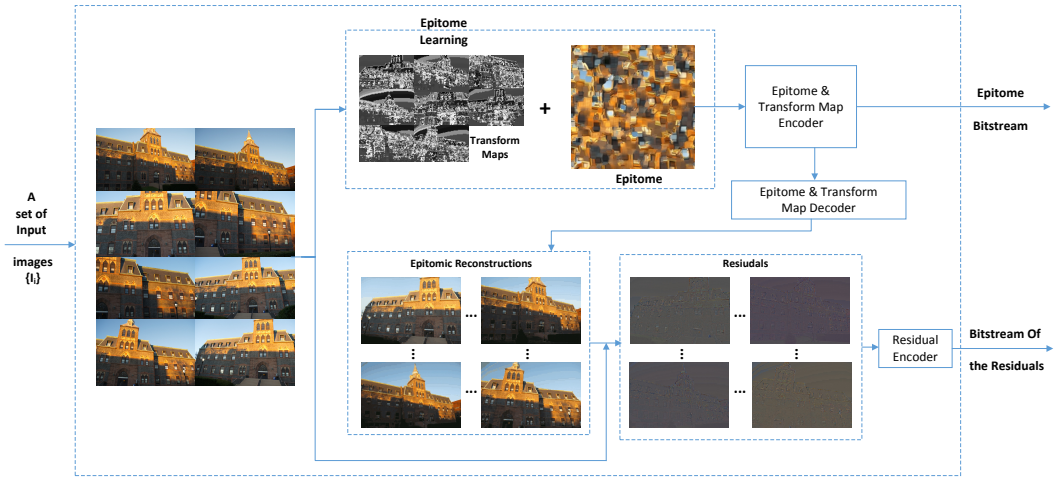


Figure 1: Learning the epitome E of a collection of images $\{I_i\}$, and doing the reconstruction via the epitome and the associated transform map $\{\Phi_i\}$. Then the entropy-encoded quantized bitstream of epitome, transform maps, and residuals, can be transmitted for the following rendering after decoding.

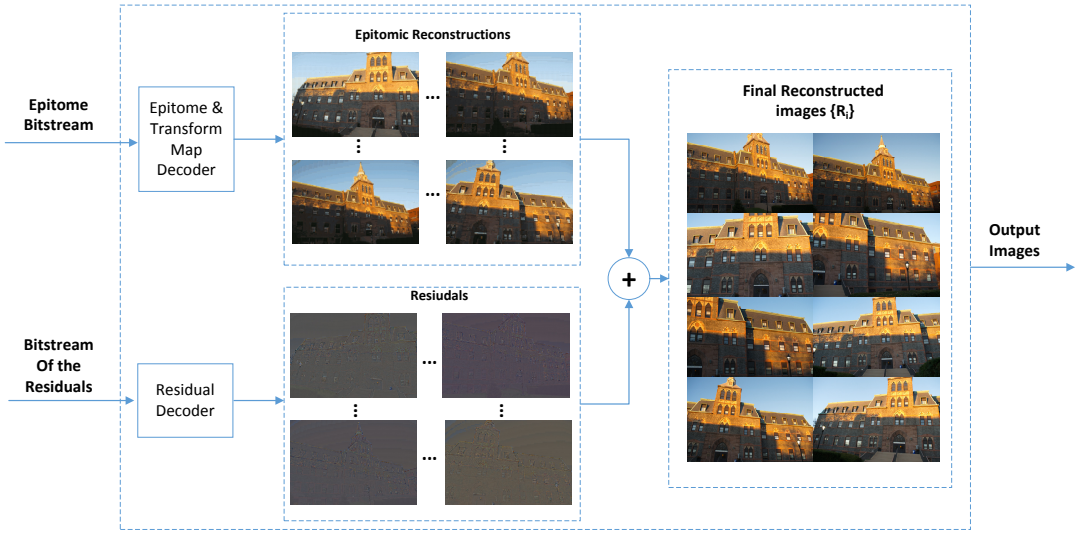


Figure 2: The transmitted bitstream of the epitome, transform maps, as well as the residuals, will be decoded for rendering final reconstructions.

via learning epitome through a collection of training image patches, our epitome-based method is able to combine all the repetition in texture and shape at once, thus works well on the prediction.

Multi-image compression by exploiting the redundance is being popular among researchers in this field. One typical kind of approach involves fractal compression, where iterated function systems are employed to reconstruct an image as the attractor of a group of recursive transformations in geometry and color [5,16,20]. An image cluster compression using a multi-image fractal partitioned iterated function systems, is proposed to compress a group of images by building a special eigenimage library for extracting principal component based on similarities between images [14]. In [13], Karhunen-Loeve compression algorithm was extended to multiple images. In order to utilize textural redundancy among a group of images, the algorithm proposed in [15] extracts textured regions from an image, and merges those textures with similar texture data from other images. Another type of approach based on sparse coding has been introduced in [1], to extract epitome-like signatures from images using sparse coding and dictionary learning. In [2], structured dictionaries are learned from an epitome, or a set of epitomes, to provide sparse image decompositions with shift-invariance properties.

Another category of multi-image compression involves epitome-based techniques. The epitome, as well as appropriate inference algorithms, has been extensively applied in the field of image segmentation, denoising, recognition, texture synthesis, video super-resolution and video interpolation [4, 8]. Recently some epitome-based image compression algorithms have been proposed, and provide state-of-the-art solutions. Wang et al. [20] presented a method of factoring the repeated content within and among images, which obtains better performance the JPEG2000 at high compression rates. Without considering the residuals between the input and the epitomic reconstruction, the quality of the finally rendered imagesuffers from lower Peak Signal-to-Noise Ration (PSNR). Instead, our ap-

proach incorporates the residues, therefore achieves a good reconstruction quality (PSNR $\approx$ 37) when the compression rate is large than 0.8 bits per pixel (bbp).

The overall framework is illustrated in Figure 1 and Figure 2. Specifically, it begins with generating a condensed epitome, which contains many of the shape and texture characteristics within and among those similar images, and the associated transform maps, so that each image can be efficiently reconstructed via them. Then based on the epitomic reconstruction, the residuals are calculated, encoded/decoded, and transmitted, together with the epitome and the transform mapping, to the codec for rendering the output images.

The contributions of this work are summarized as follows:

- Our proposed epitome transform coding represents the first of its kind for joint compression of a collection of images, which share highly similar textured regions.
- A dataset is constructed, currently including 5 categories of photos, containing both similar textures and distinctive ones.
- A detailed and systematic parametric configuration is tested to guarantee a better compression performance.
- An end-to-end comparatione between the proposed approach and JPEG, evaluated via rate-distortion, are provided to show the superiority of our method, when compressing a collection of similar images.

The remainder of this paper is organized as follows. The overall work flow and the essential technique of our epitome transform encoding approach, towards joint compression of a set of images, are demonstrated in section 2. After that, in section 3 different variants of our approach are testified in a series of experiments, and finally the results are shown and compared with that of JPEG in section 4, in terms of rate-distortion.

2. APPROACH

Our approach utilizes the similarity among a set of similar images via epitome transform coding, to achieve high joint compression ratio. This section describes the overall framework and core techniques used in the approach. For a collection of images $\mathbf{I} = \{\mathbf{I_i}\}$. The first step is to construct a compact epitome and do image reconstruction, followed by encoding the epitome, transform maps, and the residuals, i.e., the difference between the original input images and their corresponding epitomic reconstructions. Figure 1 and Figure 2 illustrate the overall work flow of our approach as follows:

- To construct a condensed epitome $\mathbf{E}$ of a set of images$\{\mathbf{I_i}\}$, which contains many of the shape and texture characteristics within and among those similar input images;
- For each image $\mathbf{I_i}$, to learn transform map ϕ_i, consisting of the $(row, column)$ coordinates of each transformed image patch in the epitome $\mathbf{E}$, so that each image is efficiently reconstructed via $\{\phi_i\}$ and $\mathbf{E}$;
- To generate epitome bitstream by encoding the epitome $\mathbf{E}$ and transform maps $\{\phi_i\}$;
- Based on input images and the prediction reconstructed from the decoded epitome bitstream, to generate, decode, and transmit the residual bitstream $\{\mathbf{R_i}\}$;
- Decoding the bitstream of epitome $\mathbf{E}$, $\{\phi_i\}$, and residuals $\{\mathbf{R_i}\}$, for rendering the final output images;
- Appropriate quantization and compression methods are employed, in order to gain better performance in terms of reconstruction quality and compression ration, evaluated by PSNR and bbp, respectively.

2.1 Epitome Learning and Reconstruction

Following the formulas of the epitome model in [6,9], a brief review is provided. For a collection of N images, denoted as $\{\mathbf{I_n}\}_{n=1}^{N}$, a set of overlapping training patches $\{\mathbf{X_i}\}_{i=1}^{P}$ are extracted. Each image patch $\mathbf{X_i}$ is interpreted as a function $x(\mathbf{k})$ on the image domain $\mathbf{K}$ (i.e., the set of all the pixels in the image). $\mathbf{k}$, as a 2-D vector representing the (r, c) coordinate of the pixel in $\mathbf{X_i}$, equals,

$$\mathbf{k} = (r, c)^T \tag{1}$$

where, r and c are the row and column indices of the pixel in image patch $\mathbf{X_i}$. Thus a 3-channel color image patch $\mathbf{X_i}$, of the size $\mathbf{s} \times \mathbf{s}$, is expressed as

$$\mathbf{X_i} = \mathbf{x}(\mathbf{k}) = (x_r(\mathbf{k}), x_g(\mathbf{k}), x_b(\mathbf{k}))^T \tag{2}$$

where $\mathbf{k} \in \mathbf{K}$. The $H_e \times W_e$ epitome $\mathbf{E}$ is actually a mixture of $H_e W_e$ Gaussian distribution, notated as

$$\{\mathbf{Norm}(\mu_\mathbf{j}, \mathbf{\Sigma_j})\}_{j=1}^{H_e W_e} \tag{3}$$

where $\mu_\mathbf{j}$ is the mean, and $\mathbf{\Sigma_j}$ is the covariance matrix. The hidden transform mapping Φ_i (only simple translational mapping is considered here), will translate each image patch $\mathbf{X_i}$ into some position $\mathbf{e} = \Phi_i(\mathbf{k})$ in epitome $\mathbf{E}$, with the maximal posterior probability $P(\Phi_i|\mathbf{X_i}, \mathbf{E})$. Given Φ_i and $\mathbf{E}$, the original images can be efficiently reconstructed. All the parameters are learned iteratively by EM algorithm [3], and in our experiments we run EM for 10 iterations.

2.1.1 Initial Epitomic Configuration

Due to each image patch $\mathbf{X_i}$, an $\mathbf{s} \times \mathbf{s}$ matrix composed of $3-D$ vector $\mathbf{x}(\mathbf{k}) = (x_r(\mathbf{k}), x_g(\mathbf{k}), x_b(\mathbf{k}))^T$, thus we consider the gaussian distribution of 3-D random variables, and for simplicity $\mathbf{x}(\mathbf{k})$ is denoted as $\mathbf{x}$,

$$P(\mathbf{x}) = \frac{1}{\sqrt{(2\pi)^3 |\mathbf{\Sigma}|}} e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T \mathbf{\Sigma}^{-1} (\mathbf{x}-\boldsymbol{\mu})} \tag{4}$$

where, $\boldsymbol{\mu} = (\mu_r, \mu_g, \mu_b)^T$, and $\mathbf{\Sigma} =$

$$\begin{pmatrix} \sigma_r^2 & 0 & 0 \\ 0 & \sigma_g^2 & 0 \\ 0 & 0 & \sigma_b^2 \end{pmatrix}$$

This Gaussian distribution is interpreted as the multiplication of three independent Gaussian ones of 1-D variable, i.e.,

$$P(\mathbf{x}) \propto P(x_r) P(x_g) P(x_b) \tag{5}$$

and for each channel $x_t \in \{x_r, x_g, x_b\}$,

$$P(x_t) \propto \frac{1}{\sqrt{2\pi} |\sigma_t|} e^{-\frac{1}{2}(x_t - \mu_t)^2 \sigma_t^{-1}} \tag{6}$$

where, $t \in \{r, g, b\}$.

For $H_e \times W_e$ epitome $\mathbf{E}$, we separately configure the parameters of each color channel as follows:

- The variances are initialized as $\mathbf{1}$s', i.e., $\sigma_r = \sigma_g = \sigma_b = 1.0$;

Table 1: EM algorithm for learning the epitome E, and the transform map Φ

EM algorithm for epitome learning and image reconstruction.

Input: a number of patches $\{X_i\}_{i=1}^{P}$ extracted out of the image set $\{I_n\}_{n=1}^{N}$

Output: a condensed $H_e \times W_e$ epitome E, composed of mixture of H_eW_e Gaussian components, and the associated transform map $\Phi = \{\Phi_i\}_{i=1}^{P}$.

01: **Initialization:** set the variances as 1s', and the means was randomized with Gaussian noise, whose mean and variance is determined by the mean and the standard deviation of all pixel values in the same channel of the images.
02: **for** 10 times EM iteration
03: **for** n = 1 : N, i.e., each image I_n
04: **for** i = 1 : P, i.e., each image patch X_i
05: calculate posterior for each channel
06: find Φ_i based on maximal posterior
07: **end for**
08: to sum up posterior through 3 channels
09: to normalize posterior
10: to accumulate posterior information
11: **end for**
12: update epitome E for next EM iteration
13: **end EM**
14: Image reconstruction based on E and Φ

- The means μ_r, μ_g, μ_b are initialized according to another Gaussian distribution, whose mean and variance equal the mean and variance of all values in the same channel.

For one category image, named as ed-5 shown in Table 2, the means μ_r, μ_g, μ_b are initialized with three Gaussian distribution, $\mu = 0.397, 0.34, 0.157$, and $\sigma = 0.246, 0.248, 0.137$, respectively.

2.1.2 *EM Algorithm in Epitome Learning*

EM algorithm iteratively maximize the log-probability [6, 7, 10],

$$L(E) = \sum_{i=1}^{P} \log\Big(\sum_{\Phi_j \in \Phi} \rho(\Phi_j) p(X_i|\Phi_j, E)\Big) \tag{7}$$

that image patches $\{\mathbf{X_i}\}_{i=1}^{P}$ were generated from the epitome $\mathbf{E}$, according to the posterior distribution over the transform mapping Φ_j,

$$p(\Phi_j|X_i, E) = \frac{\rho(\Phi_j) p(X_i|\Phi_j, E)}{\sum_{\Phi_j \in \Phi} \rho(\Phi_j) p(X_i|\Phi_j, E)} \tag{8}$$

which will be calculated in the current E step of EM, and be used for the next EM iteration. The EM flowchart is illustrated in Table 1.

2.2 Epitome Encoding

During the epitome learning, all the parameters are represented by a 64-bit floating number between 0 and 1 for calculation. But in order to reduce the overhead of our approach, after the computing, the epitome is encoded and compressed with different techniques, such as JPEG and PNG. In our experiments, we have tested that the epitome, multiplied by 255, can be encoded using 8-bit integers and compressed by JPEG, has small file size and the almost same reconstruction quality evaluated via PSNR, compared with that encoded by 64-bit floating numbers saved in the YML file, as shown in Table 2.

Table 2: comparison between 8-bit 64-bit epitomes

epitome size	64-bit double		8-bit integer	
	YML file size/KB	PSNR /dB	JPEG file size/KB	PSNR /dB
64*64	330	28.6783	4	28.5643
128*128	1320	29.9399	14	29.771
256*256	5000	31.3099	58	31.0858

2.3 Transform Map Encoding

After the epitome construction in section 2.1, the image patches are mapped into the epitome. Due to the highly repeating texture and shape within and between them, a large number of patches will be transformed to nearby or even the same position in epitome. Therefore the transform maps themselves are spatially redundant, similar to the input images, shown in Figure 3. In addition, the transform maps contain the row and column coordinates of the transformed patches in the epitome model. They can also be uniformly quantized to some extend, without obvious loss of the epitomic reconstruction quality, shown in Figure 4.

To sum up, the transform map is highly redundant both spatially and numerically. So it is quantized to some extend and entropy encoded, even at the cost of some reconstruction error. How to effectively encode those column and row indices is essential to the final compression ration. Encoding those indices with less bits, will not only result in efficient storage, but also affect the reconstruction quality. Therefore, there exists a tradeoff between the transform map compression and the epitomic image reconstruction quality. A large transform map, generating high quality of reconstruction, will have large file size, and thus reduce the compression ratio, and vice versa.

Figure 3: The transform maps, consisting of (a) column indices, and (b) row indices, are spatially redundant and similar to the input image (a).

2.4 Residual Processing and Encoding

As another key part of our proposed method, how to encode and compress the residual between the original images and the epitomic reconstructions, plays an important role of achieving large compression ration, and at the same time, retaining the high reconstruction quality. Those procedures are involved as below.

2.4.1 *Thresholding*

Since the epitome reconstruction is a good prediction of the input images, a majority of pixels in the sparse residuals have small intensities. After analyzing their histograms, an appropriate threshold is able to be set up, to ignore the rarely occurring intensities below

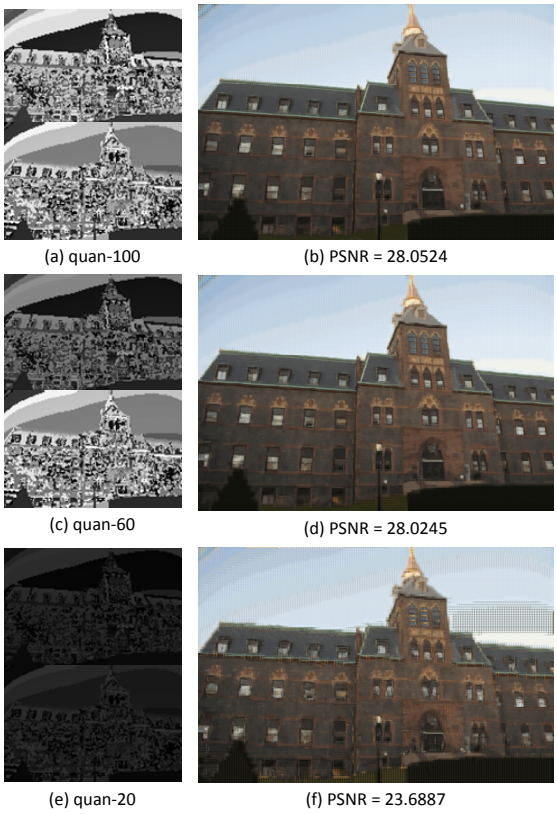


Figure 4: Do different level uniform quantization to the transform maps (higher value meaning less loss), and their corresponding epitome reconstructions evaluated by the PSNR.

the threshold, and thus to encode the residual values with few bits.

2.4.2 *Quantization and Compression*

After appropriate threshold processing, those dense intensities in residuals are then transmitted to quantizers for further compression. Quantizing the residual in different levels will generate a series of reconstruction qualities (i.e., PSNR) and compression ratio (i.e., bbp). The quantized residuals will be further entropy-compressed, the same as what the JPEG or JPEG2000 do. In our experiments, we apply JPEG2000 to do the lossless compression to the quantized residuals after thresholding.

3. EXPERIMENTS

3.1 Datasets

Experiments are performed on our own dataset consisting of natural images in several categories, where all the images are kept in the format of BMP. It contains **5** categories, i.e., *Edwin, Howe Center, Empire State Building, Flower, and Lacrosse*, showin in Figure 5 and illustrated in Table 3. And the dataset has considered the following:

- **Natural images** All the images in our dataset are natural images. Each category contains similar objects, captured from different perspectives.
- **Similarity** Since our method exploits the similarity and repetition within and across the images, the images of each category contain similar shape and textures.

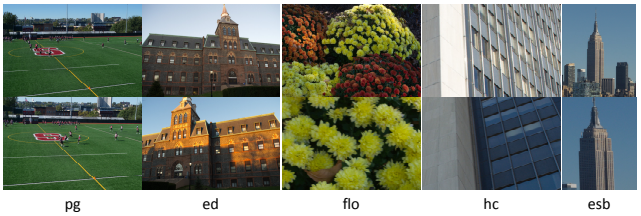


Figure 5: Our dataset consists of 5 categories, and each category is further classified into several sub classes due to the variety of image number.

Table 3: 4 categories of datasets used in our experiments.

Category	Abbreviation	Sub-category
Empire State Building	esb	esb-23
Edwin	ed	ed-5, ed-8, ed-50, ed-155, ed-255
Howe Center	hc	hc-6, hc-80, hc-150
Flower	flo	flo-10, flo-20, flo-50
Lacrosse	la	la-10

- **Variety** To testify that our approach is capable of compressing well a variety of images, the dataset includes variant and irregular shapes and textures both in high-frequency and low-frequency.p

In addition, each category above is further classified into different subclasses due to the number of images they contain. So we name each sub-category using its abbreviate name followed by the image numbers, like ed-5. The reason why we set up so many categories of images, is that we want to demonstrate that our method can achieve reliable compression result within a large range of image categories.

3.2 Experimental setup

Different experiments have been performed. Appropriate set-up of parameters for epitome learning, image reconstruction, and transform map and residual encoding, is very essential to the final compression result, which will be evaluated by the rate-distortion curves. With a series of parametric configuration, a number of rate-distortion curves are generated, from which the optimal ones will be picked out.

3.2.1 *Image patch extraction*

Given a set of images, the criteria of image patch extraction can determine how many patches will be generated for learning epitome and transform maps. Take one image in ed-5 dataset for example, shown in Figure 6, its dimension is 1024×768, and the epitome is 256×256. During the epitome learning, 8×8 patches are collected, with fixed extraction spacing, like, 4-pixel of spacing, so that a large number of overlapping patches are extracted for training an informative epitome. The file size of transform maps is determined by the number of image patches involved in the inference. Thus for the reconstruction or inference, a range of sizes of patch spacing, e.g., 4-, 6- and 8- pixel, is used to generate different number of image patches and then transform maps.

As shown in Figure 7, three transform maps with the size of 255×191, 171×128, and 128×96, respectively, have been produced due to three extraction steps of 4-, 6-, and 8-pixel.

Figure 8 compares the reconstruction from the 3 extraction steps.

Figure 6: A 1024×768 input image in ed-5, and the 256×256 epitome

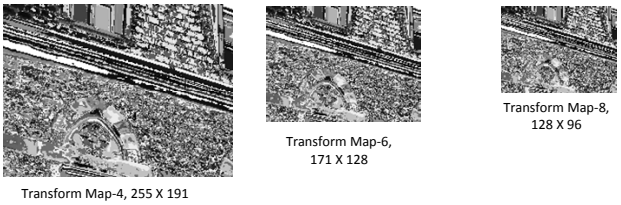


Figure 7: For image reconstruction, 3 kinds of extraction steps, 4-, 6- and 8- pixel, are used to generate different transform maps, in order to find a better compression ratio.

Figure 8: 3 kinds of extraction steps, 4-, 6- and 8- pixel, are used to generate different epitome reconstruction.

Wherein, 4-pixel step, corresponding to the most dense patches extraction, create the best reconstruction quality; In the reconstruction of 8-pixel step, the blur boundary between the non-overlapping image patches, is obvious. For 6-pixel step, it provides a good balance between improving reconstruction quality and reducing the transform maps' size. Since the file size of transform maps and residuals, and efficient encoding and/or compressing them, are the two important factors affecting the compression ratio of our approach. The rate-distortion results are evaluated for comparing those three patch extraction methods, and choosing an optimal solution out of them. In terms of PSNR and BBP, as shown in Figure 9, the 6-pixel step achieves a better performance, compared with the other two cases.

3.2.2 Parametric Configuration

As we have known before, the transform maps and residuals are key factor for our approach's pursuing higher PSNR and lower bit rates for image compression. Thus reducing their file size itself, and the appropriately designed compression technique can dramatically improve the performance of our proposed method. In addition, some parameters not only determine the compression result, but also the computational complexity, since an expensive computation of convolution, fast fourier transform are involved. The following will set up those specific parameters.

- **Patch Size s** We set up different strategies for s, since

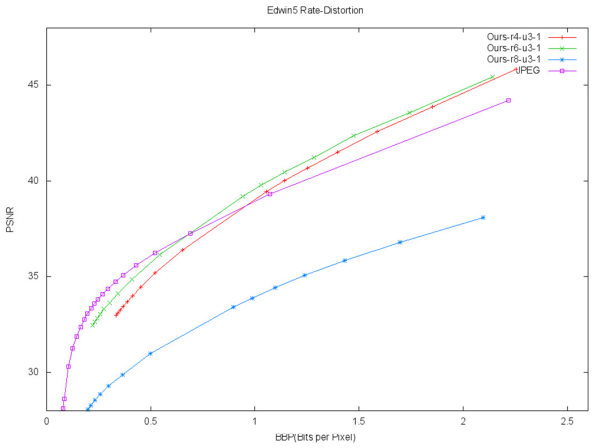


Figure 9: The rate-distortion of our epitome-based impage compression method on dataset ed-5, for three kind of extraction steps, 4-, 6- and 8- pixel.

Figure 10: 3 kinds of patch size are used to generate different reconstruction quality.

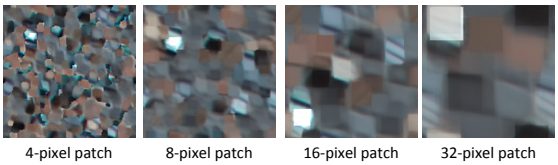


Figure 11: Image patch sizes affect the generating of epitome. Smaller images will help training the details in the input images.

it affects the epitome learning and reconstruction differently. During the learning stage, a sequent of size 4, 8, 16 of patches are extracted for training the epitome. Smaller patch can capture more details, but larger patch is useful to preserve the completeness of the texture and shape occurring in the input images. If only using the fixed s, as illustrated in Figure 10, the reconstruction quality deteriorates greatly when the patch size varies from 4 to 32. And in Figure 11, we can see the image patch sizes affect the generating of epitome. Smaller image patches is helpful for training the details in the input images. Still the image patch can not be small enough, since they will destroy the flat information existing in large image blocks, and generate more transform maps. It will increase the difficulty of encoding, and reduce the final compression ratio of our method. Thus during the reconstruction process, we find 8×8 patches work well in terms of reconstruction quality and compression ratio.

- **Spacing of Patch Extraction Δs** This parameter together with s, determines how and how many training patch-

es will be extracted out of the input images, and further the file size of associated transform maps. In our experiments, this parameter ranges from $\frac{s}{2}$ to s. Since we set patch size s being 8 for image reconstruction, then accordingly Δs equals 4, 6, and 8. As we have discussed previously, 4-, 6- and 8-pixel steps will generate transform maps in different scales, like the three mapping with the size of 255×191, 171×128, and 128×96 shown in Figure 7. They can be transmitted separately to render multi-scale images. Or the specific parts of the 255×191 transform map will be transmitted separately, generating the reconstruction from coarsely to finely.

- **Epitome Size** H_e In our experiments the epitome is square, i.e., its height and width are identical. Thus for simplicity, we only use its height when referring to its dimension. The height of epitome affects computational complexity, reconstruction quality, and compression ratio. Since for each convolutional calculation via FFT between the $s \times s$ patch and the $H_e \times H_e$ epitome E, the complexity for FFT is $\mathcal{O}(H_e^2 \log_2 H_e^2)$. Thus when the epitome size grows twice, the complexity only for FFT will be increased by over three times. In addition, large epitome, albeit beneficial for good reconstruction, takes up much physical storage, and consequently do not clearly improve the final compression ratio. We consider a series of heights varying from 128 to 512 in our experiments. As Figure 12 shown, in terms of rate-distortion, the performance for $H_e = 512$ is better than the others. But considering the computational complexity, $H_e = 256$ provides a good balance between the compression performance and the time consuming for epitome transform coding.

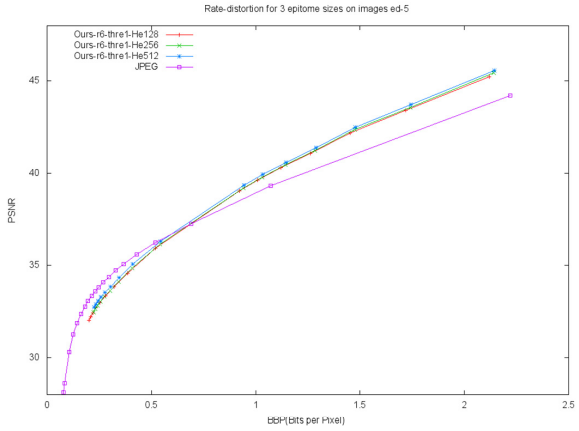


Figure 12: Rate-distortion curves calculated for 3 epitome sizes 128, 256, and 512, on images ed-5.

- **Transformation Map Quantization** How to effectively encode/decode the indices is another factor, essential to the final compression ration. Given the epitome model with dimension of 256×256, the two translation coefficients of $(row, column)$ can be represented by $2 \times 8 = 16$ bits integral numbers. In addition, the transform map Φ is itself highly redundant both spatially and numerically. So it is quantized to some extend and entropy encoded, even at the cost of some reconstruction error. On one hand, the transform map Φ compresses well due to its spacial coherence and numerical redundance. On the other hand, the map is quantized in different quantization level, for lossy compression. In our experiments, we set the quantization level as 100/100 (i.e., no quantization loss), 60/100 (i.e., 40% loss), and 20/100(i.e., 80% loss), to seek an optimal performance evaluated by the rate-distortion curves. As shown in Figure 13, the 60/100 quantizer is able to achieve the best performance, although nearly the same as that of the 60/100 one.

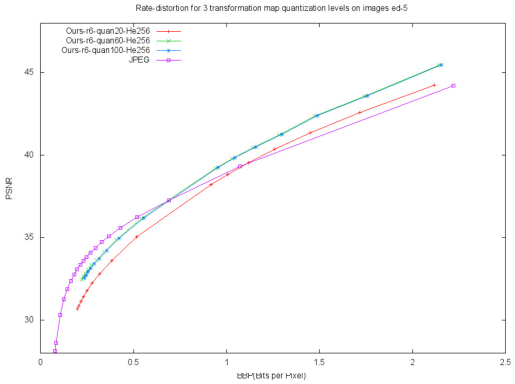
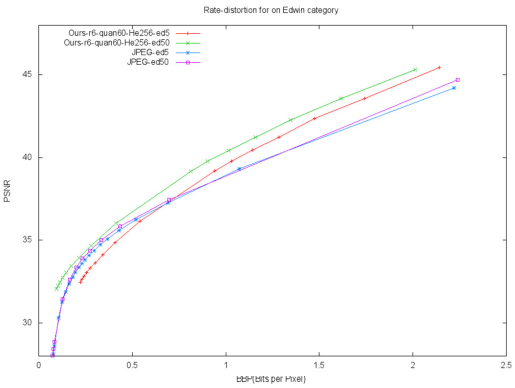


Figure 13: Compression performance evaluated via rate-distortion curves, for 3 different quantization levels to transform maps, on images ed-5.

- **Image Number** Our approach exploits the similarity and repetition, within and across the images in the collection, to pursue a joint compression of them. Not only the parameters discussed above, the number of the images in a collection, determines the number of training image patches, the epitome size, the epitomic reconstruction qualities, and thus the compression performance of our epitome transform coding approach. As shown in Figure 14, as the number of images increase, the performance of our approach improves, due to the fact that the epitome transform coding will "absorb" more repetition and similarity among the original input images, and hence more accurately reconstruct them, after learning more training image patches. For 5 images of ed-5, our method exceeds JPEG for over 0.8 bits per pixel; but it will have better performance than JPEG for ed-50 containing 50 images, no matter for low bit rates or high bit rates. Although the expensive computation, the improvement compared with JPEG, guarantees our approach has practical usage.



p

Figure 14: Rate-distortion curves vary due to the number of images in Edwin category, including ed-5 and ed-50.

3.2.3 *Residual Compression*

After histogram-based thresholding, the residuals are uniformly quantized with different quantization levels, and then be further compressed by lossless compression techniques, like lossless JPEG2000 or lossless PNG. For the input image and its epitomic reconstruction in Figure 6, most pixels in the residual, which has been normalized into the range of [0, 255] for visualization, occur in a dominant intensity range, as shown in Figure 16-(a). And the most dominant scaled intensities for R-G-B channels occur at 95, 79, and 82, respectively. The effective ranges are [67, 119], [54, 102], [57, 105], respectively. For appropriate threshold T, the trivial pixels below T are neglected. In Figure 15, the rate-distortion with 3 thresholds $T = 0.001, 0.005, 0.01$, are calculated, demonstrating that $T = 0.01$ is too large to reduce the performance of our approach in terms of PSNR.

After quantization with different quantization levels, of those filtered ranges, the values in residuals get more concentrated, as shown in Figure 16-(b).

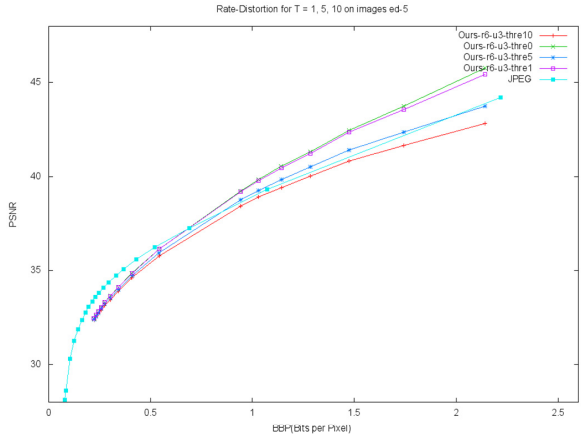


Figure 15: The residual scaled into the range of [0 255], and its histogram in this range.

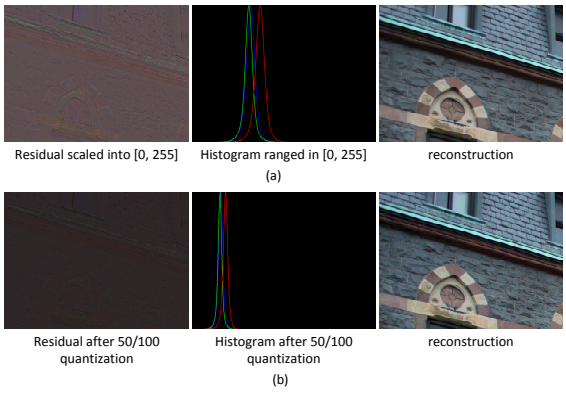


Figure 16: The residual scaled into the range of [0 255], and its histogram in this range.

4. RESULTS

To testify the performance of our proposed epitome-based images compression approach, we have done a series of experiments on our dataset. As demonstrated by our experimental evaluation, the proposed epitome transform coding can effectively take advantage of the repetitive patterns and hence redundancies across the images in each image category, to obtain joint compression. When compared with JPEG, which separately compresses each image in the collection, thus does not utilize the similarity and repetition among the images, the epitome transform coding shows clearly improvement, evaluated via rate-distortion curves.

Based on the parametric configuration illustrated in section 3.2.2, a number of experiments are finished, demonstrating that our approach achieves good practical application.

- **Edwin Category**
  In this category, the rate-distortion curves are analyzed on ed-5 and ed-50, with 3 patch extraction spacing, i.e., $\Delta s = 4, 6, 8$, as shown in Figure 9 and Figure 17.

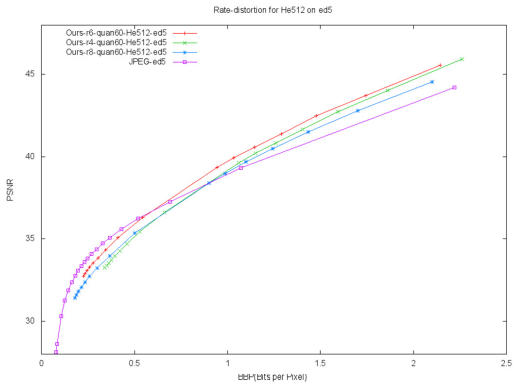


Figure 17: rate-distortion on ed-5, for epitome size of 512×512.

- **Howe Center Category**
  In this category, the rate-distortion curves are analyzed on hc-6, with 3 patch extraction spacing, i.e., $\Delta s = 4, 6, 8$, as shown in Figure 18.

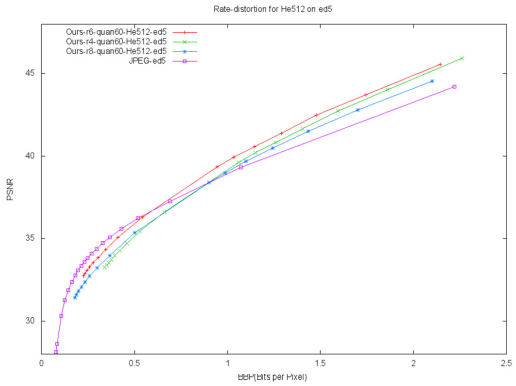


Figure 18: rate-distortion on hc-6, for epitome size of 256×256.

- **Empire State Building Category**
  In this category, the rate-distortion curves are analyzed on esb-23, with 1 patch extraction spacing, i.e., $\Delta s = 6$, as shown in Figure 19.

- **Flower category**
  In this category, the rate-distortion curves are analyzed on flo-10, flo-20, flo-50, and flo-100, with 3 patch extraction spacing, i.e., $\Delta s = 4, 6, 8$, as shown in Figure 20.

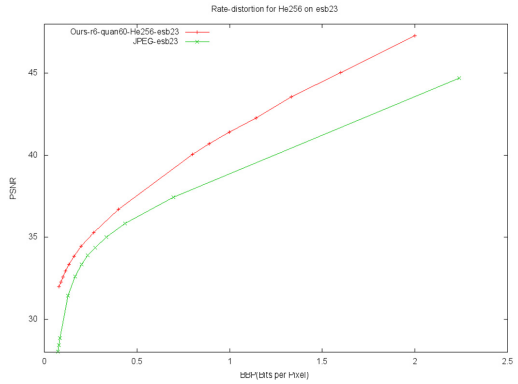


Figure 19: rate-distortion on esb-23, for epitome size of 256×256.

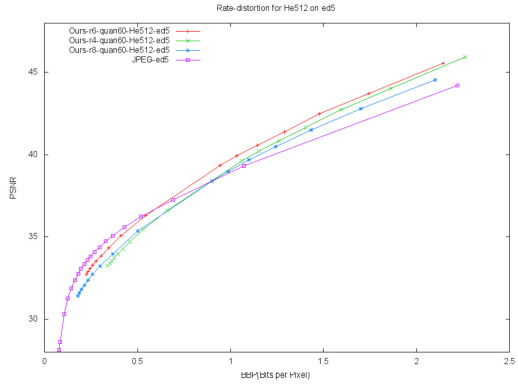


Figure 20: rate-distortion on the Flower category, for epitome size of 256×256.

- **Lacrose Category**
  In this category, the rate-distortion curves are analyzed on la-10, with 3 patch extraction spacing, i.e., $\Delta s = 4, 6, 8$, as shown in Figure 21.

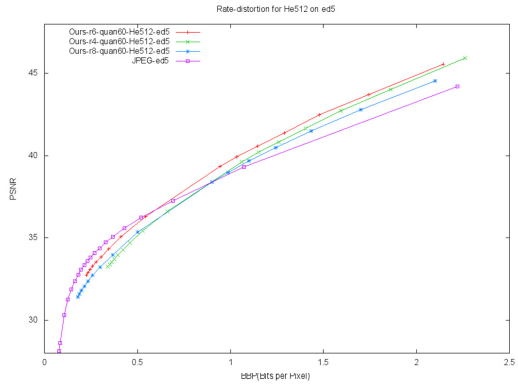


Figure 21: rate-distortion on the Lacrose category, for epitome size of 256×256.

5. SUMMARY AND FUTURE WORK

By exploiting the similarity and repetition of shape and texture among a set of images, we have presented an effective multi-image compression approach, which works well at high compression rates for all the images in our dataset, and even at low bit rate for some image categories in the dataset. Appropriate encoding and decoding of the epitome and transform maps, makes our approach outperform JPEG when the compression bit rate is larger than 0.8-1.0 bits per pixel. When the number of images increase to some extend, the compression performance will be also improved.

Our future work include:

- Improve the epitomic reconstruction quality, by introducing a sophisticated transform in terms of intensity change and geometric deformation [11, 17].
- More efficiently encoding the transform mapping.
- Accelerate the epitome learning.
- Extend our approach to video compression.

6. REFERENCES

[1] Michal Aharon and Michael Elad. Sparse and redundant modeling of image content using an image-signature-dictionary. *SIAM Journal on Imaging Sciences*, 1(3):228–247, 2008.

[2] L. Benoit, J. Mairal, F. Bach, and J. Ponce. Sparse image representation with epitomes. In *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '11, pages 2913–2920, Washington, DC, USA, 2011. IEEE Computer Society.

[3] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[4] Vincent Cheung, Brendan J. Frey, and Nebojsa Jojic. Video epitomes. In *in Proc. IEEE Conf. Comput. Vis. Pattern Recog*, pages 42–49, 2005.

[5] Yuval Fisher. Fractal image compression. *Fractals*, 2(03):347–361, 1994.

[6] Brendan J. Frey and Nebojsa Jojic. Learning the ąőepitomeąŕ of an image. Technical report, 2002.

[7] Brendan J Frey and Nebojsa Jojic. Advances in algorithms for inference and learning in complex probability models. *IEEE Trans. PAMI*, 2003.

[8] Nebojsa Jojic and Brendan J Frey. Learning flexible sprites in video layers. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–199. IEEE, 2001.

[9] Nebojsa Jojic, Brendan J. Frey, and Anitha Kannan. Epitomic analysis of appearance and shape. In *Proceedings of the Ninth IEEE International Conference on Computer Vision - Volume 2*, ICCV '03, pages 34–, Washington, DC, USA, 2003. IEEE Computer Society.

[10] Michael I Jordan, Zoubin Ghahramani, Tommi S Jaakkola, and Lawrence K Saul. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233, 1999.

[11] Anitha Kannan, John Winn, and Carsten Rother. Clustering appearance and shape by learning jigsaws. In *Advances in Neural Information Processing Systems*, pages 657–664, 2006.

[12] Naina Khedekar. *We now upload and share over 1.8 billion photos each day: Meeker Internet report*, 2014 (accessed on march 30, 2015).

[13] Matthias Kramm. Compression of image clusters using karhunen loeve transformations. In *Electronic Imaging 2007*,

pages 64920G–64920G. International Society for Optics and Photonics, 2007.

[14] Matthias Kramm. Image cluster compression using partitioned iterated function systems and efficient inter-image similarity features. In *Signal-Image Technologies and Internet-Based System, 2007. SITIS'07. Third International IEEE Conference on*, pages 989–996. IEEE, 2007.

[15] Matthias Kramm. Image group compression using texture databases. In *Electronic Imaging 2008*, pages 680613–680613. International Society for Optics and Photonics, 2008.

[16] Dietmar Saupe, Raouf Hamzaoui, and Hannes Hartenstein. *Fractal image compression: an introductory overview*. Univ., Inst. für Informatik, 1997.

[17] Denis Simakov, Yaron Caspi, Eli Shechtman, and Michal Irani. Summarizing visual data using bidirectional similarity. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

[18] David S. Taubman and Michael W. Marcellin. *JPEG 2000: Image Compression Fundamentals, Standards and Practice*. Kluwer Academic Publishers, Norwell, MA, USA, 2001.

[19] Gregory K. Wallace. The jpeg still picture compression standard. *Commun. ACM*, 34(4):30–44, April 1991.

[20] Huamin Wang, Yonatan Wexler, Eyal Ofek, and Hugues Hoppe. Factoring repeated content within and among images. *ACM Trans. Graph.*, 27(3):14:1–14:10, August 2008.