

Combining Active and Semi-Supervised Learning for Video Compression

Chiyuan Zhang¹ and Ming Ji²

¹ (Department of Electrical Engineering & Computer Science,
Massachusetts Institute of Technology, USA)

² (Department of Computer Science, University of Illinois at Urbana-Champaign, USA)

Abstract Video compression algorithms manipulate video signals to dramatically reduce the storage and bandwidth required while maximizing perceived video quality. Typical video compression methods include discrete cosine transform, vector quantization, fractal compression, and discrete wavelet transform. Recently, a machine learning based approach has been proposed which converts the color images (frames) to gray scale images (frames) and the color information for only a few representative pixels is kept. A learning model is then trained to predict the color values for the gray scale pixels across frames. Selecting the most representative pixels is essentially an active learning problem, while colorization is a semi-supervised learning problem. In this paper, we propose to combine active and semi-supervised learning for video compression. The basic idea is to minimize the size of the covariance matrix of the regularized least squares estimates, in which the regression model assumes that each pixel can be reconstructed by the other pixels with similar spatial location and intensity value. The experimental results demonstrate the effectiveness of the proposed approach for video compression.

Key words: video compression; active learning; semi-supervised learning

Zhang CY, Ji M. Combining active and semi-supervised learning for video compression.
Int J Software Informatics, Vol.7, No.3 (2013): 453–468. <http://www.ijsi.org/1673-7288/7/i163.htm>

1 Introduction

With advances in the computer technologies and the fast growth of the World Wide Web, there has been an explosion in the amount and complexity of digital videos. Video compression is a crucial technique for reducing the bandwidth required to transmit videos. Video data contains spatial and temporal redundancy. Similarities can thus be encoded by merely registering differences within a frame (spatial), and between frames (temporal). Typical compression methods include discrete cosine transform, vector quantization, fractal compression, and discrete wavelet transform.

Recently, machine learning based approach has been proposed for video compression^[5]. Instead of performing a frequency transformation, Cheng et al. proposed to convert the color video to a gray scale video. A few representative pixels are selected whose color information is stored. The gray scale video and the selected color pixels are used to learn a statistical model to predict the color values

for the rest of the pixels. Their empirical result has shown that good compression ratio can be achieved while the image quality is reasonably good according to Peak Signal to Noise Ratio (PSNR) score.

From a machine learning perspective, there are two fundamental problems. First, how to select the most representative pixels, which is essentially an active learning problem. The selected pixels, together with the gray scale video, are stored as the encoding process. Second, how to combine color and gray scale information of the pixels to learn a model, which is essentially a semi-supervised learning problem. The learned model is used to recover the color video as the decoding process. Here, color pixels are considered as labeled points, and gray scale pixels are considered as unlabeled points. Cheng et al. developed a straightforward active learning algorithm which iteratively selects the pixels on which the prediction errors are the highest. They applied Laplacian Regularized Least Squares (LapRLS, Ref. [3]) to solve the semi-supervised learning problem. The major disadvantage of Cheng's approach is that there is no theoretical guarantee that the prediction error can actually be reduced by using the selected pixels as labeled points.

In this paper, we propose to combine active and semi-supervised learning for video compression. The central idea of our approach is that pixel selection and colorization should be optimized simultaneously. Specifically, for pixel colorization, we apply a regularized least squares to learn a color prediction model, which assumes that the color of each pixel can be reconstructed by those of the other pixels with similar spatial location and intensity value. For pixel selection, we use the same loss function and select those pixels such that the covariance of the coefficients is minimized if the selected pixels are used as labeled points for training. We also discuss how to combine active and semi-supervised learning for video compression. The learned model is used to predict color information for not only the current frame but also the next several frames. When the PSNR score drops below some certain threshold, a new model will be trained. Figure 1 gives an illustrative example about how our approach works.

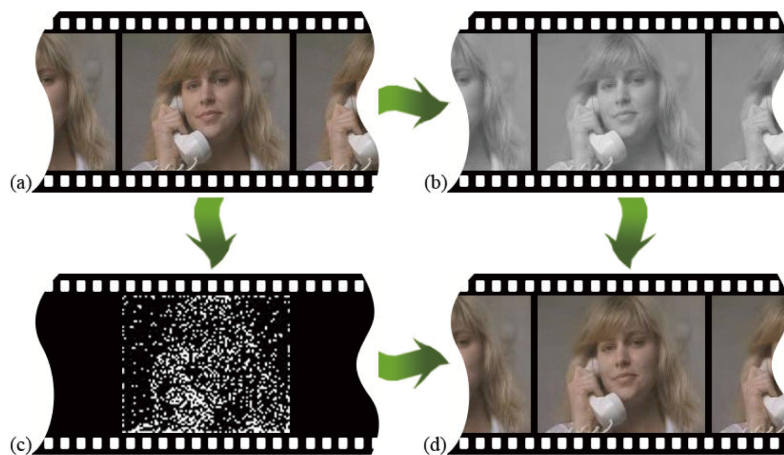


Figure 1. An illustrative example of learning based approach for video compression. (a) Original color frames; (b) Grayscale frames; (c) Selected pixels with color information; (d) Recovered color frames.

The rest of the paper is organized as follows: In Section 2, we provide a brief

review of the related works. The semi-supervised learning algorithm used for pixel colorization is introduced in Section 3. Section 4 discusses how to select the most informative pixels by using active learning method. We then present a unified active and semi-supervised learning framework for video compression in Section 5. The experimental results are presented in Section 6. Finally, we conclude in Section 7.

2 Related Work

In this section, we provide a brief description of Cheng's approach to video compression^[5] which is the most related work to our approach. From machine learning perspective, video compression can be considered as a semi-supervised learning problem^[5]. Given a set of color pixels (labeled examples) and a set of grayscale pixels (unlabeled examples) of a frame, we want to learn a function which will predict color (labels) on the grayscale pixels of the current and next several frames.

In the following, we describe the semi-supervised algorithm LapRLS used for colorization. The use of LapRLS for video compression is based on the assumption that if two pixels have similar intensity values and are spatially close to each other then it is very likely that they have similar color values. We use $\mathbf{z} \in \mathbb{R}^d$ to denote the labeled point, and $\mathbf{x} \in \mathbb{R}^d$ to denote any point (either labeled or unlabeled).

Consider a linear regression model $y = \mathbf{a}^T \mathbf{x} + \epsilon$, where y is the *observation*, \mathbf{x} is the *independent variable*, \mathbf{a} is the *weight vector* and ϵ is an unknown error with zero mean. Different observations have errors that are independent, but with equal variances σ^2 . We define $f(\mathbf{x}) = \mathbf{a}^T \mathbf{x}$ to be the learner's output given input \mathbf{x} and the weight vector \mathbf{a} .

The LapRLS algorithm makes use of both labeled and unlabeled points to learn a regression model f . Suppose there are a total of m points out of which k points are labeled. Let W be a similarity matrix. Thus, the LapRLS algorithm solves the following optimization problem:

$$\mathcal{J}_{LapRLS}(f) = \sum_{i=1}^k V(\mathbf{x}_i, y_i, f) + \lambda_1 \mathcal{R}(f) + \lambda_2 \|f\| \quad (1)$$

where V is a loss function, which is typically chosen as the square loss

$$V(x, y, f) = (f(x) - y)^2,$$

and \mathcal{R} is an appropriate regularization term which should reflect the intrinsic geometrical structure in the data. Belkin et al.^[3] suggests the following Laplacian regularization term

$$\mathcal{R}_{LapRLS}(f) = \sum_{i,j=1}^m (f(\mathbf{x}_i) - f(\mathbf{x}_j))^2 W_{ij},$$

which ensures the learned function f as smooth as possible along the geodesics of the data manifold. There are many choices of the similarity matrix W . Please refer to Ref. [4] for different definitions of W . In this work, we used binary KNN graph, which means if \mathbf{x}_i is within the k nearest neighbors of \mathbf{x}_j , or vice versa, then $W_{ij} = 1$; otherwise, $W_{ij} = 0$.

Let D be a diagonal matrix, $D_{ii} = \sum_j W_{ij}$, and $L = D - W$. Let $Z = (\mathbf{z}_1, \dots, \mathbf{z}_k)$, $X = (\mathbf{x}_1, \dots, \mathbf{x}_m)$, and $\mathbf{y} = (y_1, \dots, y_k)^T$. It is easy to show that

$$\mathcal{R}_{LapRLS}(f) = \mathbf{a}^T X L X^T \mathbf{a}$$

Thus, the solution to the minimization problem (1) is given as follows:

$$\hat{\mathbf{a}} = (Z Z^T + \lambda_1 X L X^T + \lambda_2 I)^{-1} Z \mathbf{y} \quad (2)$$

where I is an $d \times d$ identity matrix. LapRLS can also be performed in Reproducing Kernel Hilbert Space (RKHS, Ref. [18]) which gives rise to nonlinear solution. For more details about the algorithm, please see Ref. [3].

Our approach shares many common properties with previous works^[7,8]. The major differences are as follows:

1. This paper is focused on video compression, which is much more complicated than image compression considered in [8].
2. The approaches described in Ref. [7], Ref. [8] make use of graph Laplacian for regularization, while is essentially based on Laplacian Eigenmap^[2]. This work assumes that each pixel can be *reconstructed* by pixels with similar spatial location and intensity value. This kind of regularization is essentially based on Locally Linear Embedding^[14].
3. The approaches described in Ref. [7], Ref. [8] apply D-optimality as a criterion for data selection. D-optimality minimizes the content of the ellipsoidal confidence regions for the parameters of the linear model, this is, the volume of the ellipsoid. In some situations, this ellipsoid may be very long and thin, leading to small volume but unduly large variances. For this reason, this work applies A-optimality which minimizes the average variance.

3 Semi-Supervised Learning for Colorization

In this section, we introduce a novel semi-supervised learning algorithm for colorization. The basic assumption is that the color value of each pixel can be reconstructed by the color values of the pixels with similar spatial location and intensity value. This algorithm presented here is fundamentally motivated from many previous works^[3,6,14]. The major difference of the proposed algorithm from LapRLS is that we use different regularization term. The regularization term used here is motivated from Locally Linear Embedding^[14].

3.1 The algorithm

We characterize each pixel in an image by its spatial location and intensity. Thus, each pixel can be represented by a vector $\mathbf{x} \in \mathbb{R}^d$. Similar to LapRLS, we construct a nearest neighbor graph G on the pixels $\mathbf{x}_1, \dots, \mathbf{x}_m$, to capture their relationships. We minimize the following graph regularized loss function:

$$\mathcal{J}(f) = \sum_{i=1}^k (y_i - f(\mathbf{x}_i))^2 + \lambda_1 \mathcal{R}(f) + \lambda_2 \|f\|^2. \quad (3)$$

where f is defined in some Hilbert space, and $\|\cdot\|$ the corresponding norm in the Hilbert space.

For naturally generated images, it is often the case that the spatial location and intensity value of a pixel may be estimated from those of its neighboring pixels. Thus, we can characterize the relationship between pixels by linear coefficients that reconstruct each pixel vector from its neighbors. Reconstructing errors are measured by the cost function^[14]:

$$\phi(W) = \sum_{i=1}^m \left\| \mathbf{x}_i - \sum_{j=1}^m W_{ij} \mathbf{x}_j \right\|^2,$$

which adds up the squared distances between all the pixels and their reconstructions. Note that, W_{ij} vanishes if there is no link between \mathbf{x}_i and \mathbf{x}_j . This cost function is originally used to learn geometrical structure of the data space in the Locally Linear Embedding algorithm^[11,14]. Please see Ref. [14] for details.

Now, consider the problem of estimating a label (color value) for each pixel so that the label of each pixel can be represented as a linear combination of the labels of its neighbors with coefficients W_{ij} . A reasonable regularizer for choosing a “good” classifier is as follows:

$$\mathcal{R}_{LLE}(f) = \sum_{i=1}^m \left(f(\mathbf{x}_i) - \sum_{j=1}^m W_{ij} f(\mathbf{x}_j) \right)^2. \quad (4)$$

Note that, in (4), W_{ij} are fixed weights. For linear functions $f(\mathbf{x}) = \mathbf{a}^T \mathbf{x}$, (4) becomes

$$\mathcal{R}_{LLE}(\mathbf{a}) = \sum_{i=1}^m \left(\mathbf{a}^T \mathbf{x}_i - \sum_{j=1}^m W_{ij} \mathbf{a}^T \mathbf{x}_j \right)^2. \quad (5)$$

Following some simple algebraic steps, we have

$$\begin{aligned} \mathcal{R}_{LLE}(\mathbf{a}) &= \mathbf{a}^T \left(\sum_{i=1}^m (\mathbf{x}_i - \sum_{j=1}^m W_{ij} \mathbf{x}_j) (\mathbf{x}_i - \sum_{j=1}^m W_{ij} \mathbf{x}_j)^T \right) \mathbf{a} \\ &= \mathbf{a}^T (X - XW^T) (X - XW^T)^T \mathbf{a} \\ &= \mathbf{a}^T X (I - W)^T (I - W) X^T \mathbf{a} \\ &\doteq \mathbf{a}^T X M X^T \mathbf{a}, \end{aligned}$$

where $M = (I - W)^T (I - W)$. It is easy to see that the matrix M is symmetric and positive semi-definite. Let $y_i = f(\mathbf{x}_i)$ and $\mathbf{y} = (y_1, y_2, \dots, y_k)^T$. The least square error on the labeled images can be computed as follows:

$$\begin{aligned} \sum_{i=1}^k (y_i - \mathbf{a}^T \mathbf{x}_i)^2 &= (\mathbf{y} - Z^T \mathbf{a})^T (\mathbf{y} - Z^T \mathbf{a}) \\ &= \mathbf{y}^T \mathbf{y} - 2\mathbf{a}^T Z \mathbf{y} + \mathbf{a}^T Z Z^T \mathbf{a}. \end{aligned}$$

Noticing that $\mathbf{y}^T \mathbf{y}$ is a constant, the loss function (3) can be finally reduced to:

$$\mathcal{J}(\mathbf{a}) = -2\mathbf{a}^T Z \mathbf{y} + \mathbf{a}^T Z Z^T \mathbf{a} + \lambda_1 \mathbf{a}^T X M X^T \mathbf{a} + \lambda_2 \mathbf{a}^T \mathbf{a}. \quad (6)$$

Requiring the derivative of \mathcal{J} with respect to \mathbf{a} vanish, we have

$$\begin{aligned} \frac{\partial \mathcal{J}}{\partial \mathbf{a}} &= 0 \\ \Rightarrow -2Z\mathbf{y} + 2ZZ^T\mathbf{a} + 2\lambda_1 XMX^T\mathbf{a} + 2\lambda_2\mathbf{a} &= 0 \\ \Rightarrow \mathbf{a} &= \left(ZZ^T + \lambda_1 XMX^T + \lambda_2 I \right)^{-1} Z\mathbf{y} \end{aligned} \quad (7)$$

It is easy to see that the matrix $ZZ^T + \lambda_1 XMX^T$ is positive semi-definite. Therefore, the matrix $ZZ^T + \lambda_1 XMX^T + \lambda_2 I$ is invertible. Once the regression function f is obtained, the label of any pixel \mathbf{x} can be estimated by $f(\mathbf{x}) = \mathbf{a}^T \mathbf{x}$.

3.2 Nonlinear generalization

In this subsection, we discuss how to generalize the regularized regression algorithm to nonlinear problem by using kernel tricks^[18].

We consider the problem in a Reproducing Kernel Hilbert Space (RKHS) \mathcal{H} associated with a kernel function $\mathcal{K} : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$. The loss function in \mathcal{H} can be written as follows:

$$\mathcal{J}(f) = \sum_{i=1}^k (y_i - f(\mathbf{x}_i))^2 + \lambda_1 \mathcal{R}_{LLE}(f) + \lambda_2 \|f\|_{\mathcal{H}}, f \in \mathcal{H}. \quad (8)$$

By representer theorem^[3,18], the optimal f has the following form:

$$f(\mathbf{x}) = \sum_{i=1}^m \alpha_i \mathcal{K}(\mathbf{x}, \mathbf{x}_i) \quad (9)$$

Let K_{XX} be a $m \times m$ kernel matrix such that $K_{XX,ij} = \mathcal{K}(\mathbf{x}_i, \mathbf{x}_j)$, and let K_{ZX} be a $k \times m$ matrix such that $K_{ZX,ij} = \mathcal{K}(\mathbf{z}_i, \mathbf{x}_j)$. We define:

$$\mathbf{f}_Z = (f(\mathbf{z}_1), \dots, f(\mathbf{z}_k))^T.$$

Clearly, $\mathbf{f}_Z = K_{ZX}\boldsymbol{\alpha}$. Therefore,

$$\begin{aligned} \sum_{i=1}^k (y_i - f(\mathbf{z}_i))^2 &= (\mathbf{y} - \mathbf{f}_Z)^T (\mathbf{y} - \mathbf{f}_Z) \\ &= (\mathbf{y} - K_{ZX}\boldsymbol{\alpha})^T (\mathbf{y} - K_{ZX}\boldsymbol{\alpha}) \\ &= \mathbf{y}^T \mathbf{y} - 2\mathbf{y}^T K_{ZX}\boldsymbol{\alpha} + \boldsymbol{\alpha}^T K_{ZX}^T K_{ZX}\boldsymbol{\alpha} \end{aligned}$$

Similarly, we define:

$$\mathbf{f}_X = (f(\mathbf{x}_1), \dots, f(\mathbf{x}_m))^T = K_{XX}\boldsymbol{\alpha}.$$

Thus,

$$\begin{aligned} \mathcal{R}_{LLE}(f) &= \sum_{i=1}^m \left(f(\mathbf{x}_i) - \sum_{j=1}^m W_{ij} f(\mathbf{x}_j) \right)^2 \\ &= (K_{XX}\boldsymbol{\alpha} - WK_{XX}\boldsymbol{\alpha})^T (K_{XX}\boldsymbol{\alpha} - WK_{XX}\boldsymbol{\alpha}) \end{aligned}$$

$$\begin{aligned}
&= \boldsymbol{\alpha}^T K_{XX} (I - W)^T (I - W) K_{XX} \boldsymbol{\alpha} \\
&= \boldsymbol{\alpha}^T K_{XX} M K_{XX} \boldsymbol{\alpha}
\end{aligned}$$

And,

$$\begin{aligned}
\|f\|_{\mathcal{H}}^2 &= \langle f, f \rangle \\
&= \left\langle \sum_{i=1}^m \alpha_i \mathcal{K}(\cdot, \mathbf{x}_i), \sum_{j=1}^m \alpha_j \mathcal{K}(\cdot, \mathbf{x}_j) \right\rangle \\
&= \sum_{i,j} \alpha_i \alpha_j \langle \mathcal{K}(\cdot, \mathbf{x}_i), \mathcal{K}(\cdot, \mathbf{x}_j) \rangle \\
&= \sum_{i,j} \alpha_i \alpha_j \mathcal{K}(\mathbf{x}_i, \mathbf{x}_j) \\
&= \boldsymbol{\alpha}^T K_{XX} \boldsymbol{\alpha}
\end{aligned}$$

So, the loss function can be reduced to

$$\begin{aligned}
\mathcal{J}(f) &= \mathbf{y}^T \mathbf{y} - 2\mathbf{y}^T K_{ZX} \boldsymbol{\alpha} + \boldsymbol{\alpha}^T K_{ZX}^T K_{ZX} \boldsymbol{\alpha} \\
&\quad + \lambda_1 \boldsymbol{\alpha}^T K_{XX} M K_{XX} \boldsymbol{\alpha} + \lambda_2 \boldsymbol{\alpha}^T K_{XX} \boldsymbol{\alpha}
\end{aligned}$$

Requiring that the gradient of \mathcal{L} vanish gives the following solution:

$$\boldsymbol{\alpha} = (K_{ZX}^T K_{ZX} + \lambda_1 K_{XX} M K_{XX} + \lambda_2 K_{XX})^{-1} K_{ZX}^T \mathbf{y} \quad (10)$$

4 Active Learning for Pixel Selection

A key step in learning based video compression is to select the most representative pixels. In this section, we introduce an active learning approach for this purpose. The active learning approach shares the same loss function as the semi-supervised learning algorithm described in the last section. Specifically, we expect that the prediction error can be minimized if the selected pixels are used as training data.

4.1 The algorithm

Since $y_i = \mathbf{a}^T \mathbf{z}_i + \epsilon$, we have $\mathbf{y} = Z^T \mathbf{a} + \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} = (\epsilon, \dots, \epsilon)^T$. Clearly $E(\boldsymbol{\epsilon}) = \mathbf{0}$, where $\mathbf{0} = (0, \dots, 0)$. We define

$$H = ZZ^T + \lambda_1 X M X^T + \lambda_2 I \quad (11)$$

$$\Lambda = \lambda_1 X M X^T + \lambda_2 I. \quad (12)$$

Previous studies have shown that the bias of the estimator has the following expression

$$E(\hat{\mathbf{a}} - \mathbf{a}) = -H^{-1} \Lambda \mathbf{a}, \quad (13)$$

and the covariance matrix of $\hat{\mathbf{a}}$ can be well approximated by $\sigma^2 H^{-1}$, please see Ref. [8] for details. In order to make the estimator $\hat{\mathbf{a}}$ as stable as possible, the size of covariance matrix $Cov(\hat{\mathbf{a}})$ has to be as small as possible. Different measures of the size of the covariance matrix lead to different optimality criteria. Previous work has suggested to use D-optimality^[7,8], in which the *determinant* of the covariance matrix

is minimized. D-optimality was motivated by reference to the ellipsoidal confidence regions for the parameters of the linear model. A D-optimum design minimizes the content of this confidence region and so minimizes the volume of the ellipsoid^[1]. The major disadvantage of D-optimality is that the ellipsoid may be very long and thin. In this case, the volume of the ellipsoid may be sufficiently small, whereas the variances of some parameter estimates, or their linear combinations, can be unduly large, and consequently the parameters will be imprecisely estimated. For this reason, we adopt the A-optimality^[1] which minimizes the average variance.

Let $\mathcal{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_m\}$ denote the set of all the data points and $\mathcal{Z}_k = \{\mathbf{z}_1, \dots, \mathbf{z}_k\}$ denote the set of the first k selected points. Clearly, $\mathcal{Z}_k \subseteq \mathcal{X}$ and $0 \leq k \leq m$. Thus, the most informative pixels can be selected by solving the following optimization problem:

$$\min_{Z=(\mathbf{z}_1, \dots, \mathbf{z}_k)} \text{Tr}(H^{-1}), \quad \mathbf{z}_i \in \mathcal{X}, i = 1, \dots, k \quad (14)$$

where Tr denotes the trace operator. In the following, we describe a sequential construction of graph regularized A-optimal experimental designs. We define

$$H_k = Z_k Z_k^T + \lambda_1 X M X^T + \lambda_2 I, k \geq 1, \quad (15)$$

and

$$H_0 = \lambda_1 X M X^T + \lambda_2 I. \quad (16)$$

Suppose $k(\geq 0)$ points have been selected. In other words, Z_k and H_k are given. Thus, the $(k+1)$ -th point is given by

$$\mathbf{z}_{k+1} = \arg \min_{\mathbf{z} \in \mathcal{X} - \mathcal{Z}_k} \text{Tr} \left((H_k + \mathbf{z} \mathbf{z}^T)^{-1} \right) \quad (17)$$

By using the Sherman-Morrison formula^[16], we have

$$(H_k + \mathbf{z} \mathbf{z}^T)^{-1} = H_k^{-1} - \frac{H_k^{-1} \mathbf{z} \mathbf{z}^T H_k^{-1}}{1 + \mathbf{z}^T H_k^{-1} \mathbf{z}} \quad (18)$$

Since $\text{Tr}(H_k^{-1})$ is a constant while selecting the $(k+1)$ -th point, Eq. (17) can be reduced to finding

$$\begin{aligned} \mathbf{z}_{k+1} &= \arg \max_{\mathbf{z} \in \mathcal{X} - \mathcal{Z}_k} \frac{\text{Tr}(H_k^{-1} \mathbf{z} \mathbf{z}^T H_k^{-1})}{1 + \mathbf{z}^T H_k^{-1} \mathbf{z}} \\ &= \arg \max_{\mathbf{z} \in \mathcal{X} - \mathcal{Z}_k} \frac{\text{Tr}(\mathbf{z}^T H_k^{-1} H_k^{-1} \mathbf{z})}{1 + \mathbf{z}^T H_k^{-1} \mathbf{z}} \\ &= \arg \max_{\mathbf{z} \in \mathcal{X} - \mathcal{Z}_k} \frac{\|H_k^{-1} \mathbf{z}\|^2}{1 + \mathbf{z}^T H_k^{-1} \mathbf{z}} \end{aligned} \quad (19)$$

Once \mathbf{z}_{k+1} is selected, the inverse of H_{k+1} can be updated based on the inverse of H_k , according to Eq. (18)

$$H_{k+1}^{-1} = (H_k + \mathbf{z}_{k+1} \mathbf{z}_{k+1}^T)^{-1} = H_k^{-1} - \frac{(H_k^{-1} \mathbf{z}_{k+1})(H_k^{-1} \mathbf{z}_{k+1})^T}{1 + \mathbf{z}_{k+1}^T H_k^{-1} \mathbf{z}_{k+1}} \quad (20)$$

The above derivation has shown that we only need to compute the inverse of H_0 and there is no other computation of matrix inverse required. Therefore, the selection of data points can be very efficient.

4.2 Nonlinear generalization

In this section, we describe how to generalize our algorithm to nonlinear problem by using kernel tricks^[18].

We consider the problem in a feature space \mathcal{H} induced by some nonlinear mapping: $\phi : \mathbb{R}^d \rightarrow \mathcal{H}$. For a properly chosen ϕ , an inner product $\langle \cdot, \cdot \rangle$ can be defined on \mathcal{H} which makes for a so-called Reproducing Kernel Hilbert Space (RKHS). More specifically, $\langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle = \mathcal{K}(\mathbf{x}_i, \mathbf{x}_j)$ holds where $\mathcal{K}(\cdot, \cdot)$ is a positive semi-definite kernel function.

Similarly, we define the corresponding data matrix and the matrix of selected points as follows

$$\begin{aligned} \phi_X &= \left(\phi(\mathbf{x}_1), \dots, \phi(\mathbf{x}_m) \right), \\ \phi_Z &= \left(\phi(\mathbf{z}_1), \dots, \phi(\mathbf{z}_k) \right). \end{aligned}$$

We define

$$\phi_H = \phi_Z \phi_Z^T + \lambda_1 \phi_X M \phi_X^T + \lambda_2 I \tag{21}$$

However, ϕ is usually unknown. It is unclear how to compute the trace of ϕ_H^{-1} . Instead, we consider the regression model in RKHS

$$y = \boldsymbol{\nu}^T \phi(\mathbf{x}) + \epsilon, \quad \boldsymbol{\nu} \in \mathcal{H}. \tag{22}$$

Thus, the objective function (3) in RKHS can be written as follows:

$$\mathcal{J}(\boldsymbol{\nu}) = \sum_{i=1}^k (\boldsymbol{\nu}^T \phi(\mathbf{z}_i) - y_i)^2 + \lambda_1 \sum_{i=1}^m (\boldsymbol{\nu}^T \phi(\mathbf{x}_i) - \sum_{j=1}^m W_{ij} \boldsymbol{\nu}^T \phi(\mathbf{x}_j))^2 + \lambda_2 \|\boldsymbol{\nu}\|_{\mathcal{H}}^2 \tag{23}$$

We have the following proposition.

Proposition 4.1. Let $\mathcal{H}_X = \{ \sum_{i=1}^m \alpha_i \phi(\mathbf{x}_i) | \alpha_i \in \mathbb{R} \}$ be a subspace of \mathcal{H} , the minimum solution to the problem (23) is in \mathcal{H}_X .

Please refer to Ref. [8] for the proof. From Proposition 4.1, we see that $\boldsymbol{\nu}$ can be represented as a linear combination of $\phi(\mathbf{x}_i)$, $i = 1, \dots, m$:

$$\boldsymbol{\nu} = \sum_i \alpha_i \phi(\mathbf{x}_i) = \phi_X \boldsymbol{\alpha}, \tag{24}$$

where $\boldsymbol{\alpha} = (\alpha_1, \dots, \alpha_m)^T$. Since ϕ_X is a constant matrix in RKHS, Eq. (24) tells us that the only parameter we need to estimate is $\boldsymbol{\alpha}$. Similar to the linear algorithm described in Section 4.1, here we select the points such that the size of $Cov(\boldsymbol{\alpha})$ is minimized. Previous studies on regularized experimental design^[8] have shown that $Cov(\boldsymbol{\alpha})$ can be well approximated by $\sigma^2 (K_{ZX}^T K_{ZX} + \lambda_1 K_{XX} M K_{XX} + \lambda_2 K_{XX})^{-1}$, where K_{XX} and K_{ZX} are defined in Section 3.2. The nonlinear problem is thus defined as follows:

$$\max_{Z=(\mathbf{z}_1, \dots, \mathbf{z}_k)} \text{Tr} \left((K_{ZX}^T K_{ZX} + \lambda_1 K_{XX} M K_{XX} + \lambda_2 K_{XX})^{-1} \right).$$

Let \mathbf{u}_i be the i -th column vector of K_{XX} and \mathcal{U} be the set of $\{\mathbf{u}_1, \dots, \mathbf{u}_m\}$. \mathbf{u}_i corresponds to \mathbf{x}_i , and $\mathbf{u}_i = (\mathcal{K}(\mathbf{x}_i, \mathbf{x}_1), \dots, \mathcal{K}(\mathbf{x}_i, \mathbf{x}_m))^T$. We define:

$$A_k = K_{Z_k X}^T K_{Z_k X} + \lambda_1 K_{XX} M K_{XX} + \lambda_2 K_{XX}, k \geq 0, \quad (25)$$

and

$$A_0 = \lambda_1 K_{XX} M K_{XX} + \lambda_2 K_{XX}.$$

Suppose k data points have been selected, i.e. $\mathcal{V}_k = \{\mathbf{v}_1, \dots, \mathbf{v}_k\} \subset \mathcal{U}$. The $(k+1)$ -th point is selected by solving the following optimization problem:

$$\mathbf{v}_{k+1} = \arg \max_{\mathbf{v} \in \mathcal{U} - \mathcal{V}_k} \text{Tr} \left((\mathbf{v}\mathbf{v}^T + A_k)^{-1} \right), \quad (26)$$

which is essentially similar to the problem (17).

Following some similar algebraic steps, we have

$$\mathbf{v}_{k+1} = \arg \max_{\mathbf{v} \in \mathcal{U} - \mathcal{V}_k} \frac{\|A_k^{-1} \mathbf{v}\|^2}{1 + \mathbf{v}^T A_k^{-1} \mathbf{v}}. \quad (27)$$

Once \mathbf{v}_{k+1} is obtained, the inverse of A_k can be easily updated as follows:

$$A_{k+1}^{-1} = A_k^{-1} - \frac{(A_k^{-1} \mathbf{v}_{k+1})(A_k^{-1} \mathbf{v}_{k+1})^T}{1 + \mathbf{v}_{k+1}^T A_k^{-1} \mathbf{v}_{k+1}} \quad (28)$$

As can be seen, the computation of the nonlinear algorithm is essentially the same as that of the linear algorithm, except that the data vectors $\mathbf{x}_i (i = 1, \dots, m)$ are replaced by $\mathbf{u}_i (i = 1, \dots, m)$.

5 Combining Active and Semi-Supervised Learning for Video Compression

In summary, in the encoding stage, active learning algorithm is applied to choose suitable labeled points. The selected color pixels and the grayscale image are then stored (and optionally transferred). At the decoding stage, semi-supervised learning algorithm is run to train a color predictor based on the stored labeled and unlabeled data. The trained predictor is then used to predict the colors. The decoding results are obtained by combining the predicted colors with the stored grayscale image.

Video compression shares many common properties with image compression. Each frame of a video is actually an image. However, video is much more complicated than image in that there is temporal correlation between consecutive frames. In this section we describe how to combine active and semi-supervised learning for video compression. There are many ways to measure the image quality^[9,12]. In this work, we employ Peak Signal to Noise Ratio (PSNR) score to measure the quality of video compression.

Following previous work on video colorization and compression^[5,10], we work in YUV space where Y is the intensity (luminance) channel, U and V are the chrominance channels that encode the color. We predict U, V values independently. In Ref. [5], the features used to represent a pixel include the spatial location in the image grid and the local texture information. However, our empirical investigation

has shown that texture information is not useful for predicting color information. Therefore, in our work we only use spatial location and grayscale value to characterize each pixel.

Given a video with ℓ frames and each frame is an image with n pixels. For the first frame, we apply the semi-supervised learning algorithm described in Section 3 to learn a colorization model which predicts the color values of the grayscale pixels for the first and the next several frames. If the PSNR score drops below some certain threshold, a new colorization model will be trained. The semi-supervised learning algorithm involves inverting a dense $m \times m$ matrix, which is computationally intractable. To reduce the computational burden, Cheng et al.^[5] propose to use *super-pixel* representation^[13] obtained by segmenting the image into regions using Normalized Cut (NCut, Ref. [17]), and pick pixels randomly from these regions. However, NCut itself is a time consuming algorithm. In our work we avoid using any sophisticated segmentation algorithms, but simply segment the image into rectangular regions. The number of regions is set to 2000 in our experiments according to the image size. We randomly pick one pixel from each region, so there are totally 2000 pixels over which we then construct a 4-nearest neighbor graph.

Once the graph is constructed, we can apply the active learning algorithm to select the most representative pixels from the 2000 pixels. The quality of the compressed video usually varies with the number of selected color pixels (labels). The more selected color pixels, the higher image quality. On the other hand, selecting more color pixels reduces the compression ratio. Since the graph is constructed on the 2000 pixels, rather than all the pixels in the image, the maximum number of the selected color pixels (labels) is 2000. For decoding, the semi-supervised learning algorithm is applied to learn a function to predict the color on the rest of the grayscale pixels.

6 Experimental Evaluation

In this section, we investigate the performance of our proposed approach for video compression. Illustrative examples and quantitative evaluations are provided.

Since the frames of a video generally do not change too fast, the color prediction model learned from a single frame can also be applied to successive frames without much visible distortions. In our experiments, we set a threshold Δ_{PSNR} . Let $PSNR^*$ denote the colorization quality of the frame on which the model is learned. We keep using the learned colorization model for the successive frames until the PSNR drops below $PSNR^* - \Delta_{PSNR}$ at which time a new colorization model needs to be learned. Since the colorization model is trained on a single frame, we call it Single Frame Model (SFM). Finally we may have several colorization models denoted by SFM_1, \dots, SFM_k . Correspondingly the frames on which SFM_i 's are trained are called Key Frames, denoted by KF_1, \dots, KF_k .

In order to make use of temporal locality property of a video sequence, we further extend our method as follows. An extra feature of time t is introduced and combined with the spatial and grayscale features based on which the active and semi-supervised learning is performed. Moreover, instead of learning the colorization model by using only one frame, here we learn the colorization model by using multiple key frames

obtained from SFM. Specifically, we divide these key frames into several time intervals such that for each interval the difference of the key frames would not be too large. The key frames in each interval are used to learn a colorization model. We call this method Multiple Frame Model (MFM). Likewise, we may have several colorization models for a video, denoted by $MF M_1, \dots, MF M_i$. Note that, each $MF M_i$ is only applied to frames within its corresponding time interval.

6.1 Evaluation on color pixel selection

We first compare our proposed active learning algorithm with Cheng's approach for selecting the most informative color pixels. We use the same video of a call center employee as that in Ref. [5]. It contains 301 frames, each of size 240×130 .

Figure 2(a) and 2(d) show the first color frame and its corresponding grayscale image. Figure 2(b) and 2(e) depict the color pixels chosen by Cheng's active learning approach and our active learning approach. For each approach, 500 color pixels are chosen. Comparing to Cheng's approach, our algorithm chooses relatively more pixels along the boundaries where color changes occur. From the pixels chosen by our algorithm, we can see a rough contour of the face. Figure 2(c) and 2(f) show the colorized (or uncompressed) frames obtained by Cheng's approach and our approach, respectively. As can be seen, our approach produces visually more pleasing results. Cheng's approach achieves a PSNR of 35.38 for the first frame, and our approach achieves a PSNR of 38.99. Our approach achieves significantly higher quality as measured by PSNR.

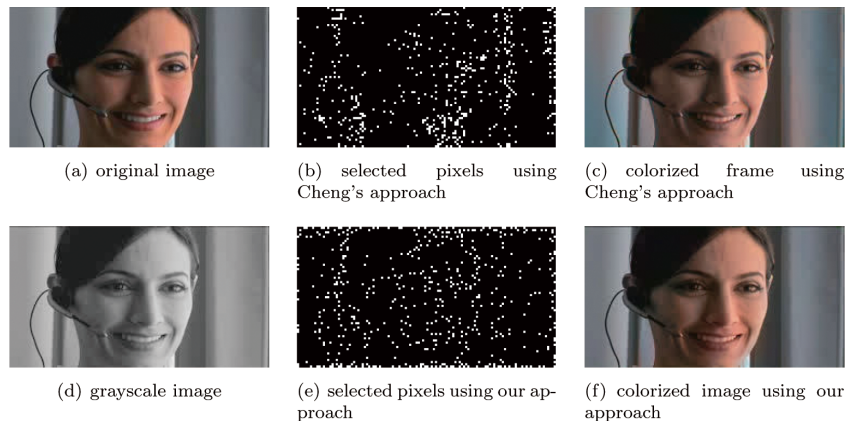


Figure 2. Pixel selection and colorization on the first frame.

6.2 Evaluation on video compression quality

The aim of this experiment is to compare our approach to Cheng's approach and MPEG4 in terms of video compression quality. By fixing the total number of selected color pixels, we set the compression ratio to a fixed value. Then the overall PSNR is compared for those two methods.

The number of models trained during video compression is dependent on Δ_{PSNR} and the quality of the colorization result. In this experiment, Cheng's approach needs to train 9 models. Our SFM and MFM need to train 9 and 3 models, respectively.

It is generally impossible to make the number of selected color pixels for different approaches exactly the same. For Cheng's approach, each time when a new model is trained, 500 color pixels are selected. Thus, 4500 color pixels are selected as labels in total. For our SFM approach, we also select 500 color pixels at each time when a new model is trained, thus also 4500 color pixels are selected in total. For our MFM approach, we select certain number of color pixels such that the resulting PSNR would not be below that of SFM. This way, 3900 color pixels are selected in total.

We use *MPlayer's Movie Encoder* to encode the original color video into an MPEG-4 format video of 756,586 bytes. Then we compress the resultant video with the same MPEG-4 codec to a grayscale video which occupies 250,288 bytes. The grayscale video will be used while decoding. In order to recover the color video, we need to store the spatial location and color value for each selected pixel, besides the grayscale video. Thus, 4 bytes are needed per selected pixel (2 for color information and 2 for location). This adds a modest 18,000 bytes of extra storage for both Cheng's approach and SFM, while MFM needs only 15,600 bytes.

In Fig. 3(a), we show the first 48 frames of the original video. The yellow marker denotes the frame on which a new colorization model is trained for our approach, while the red marker denotes the frame on which a new colorization model is trained

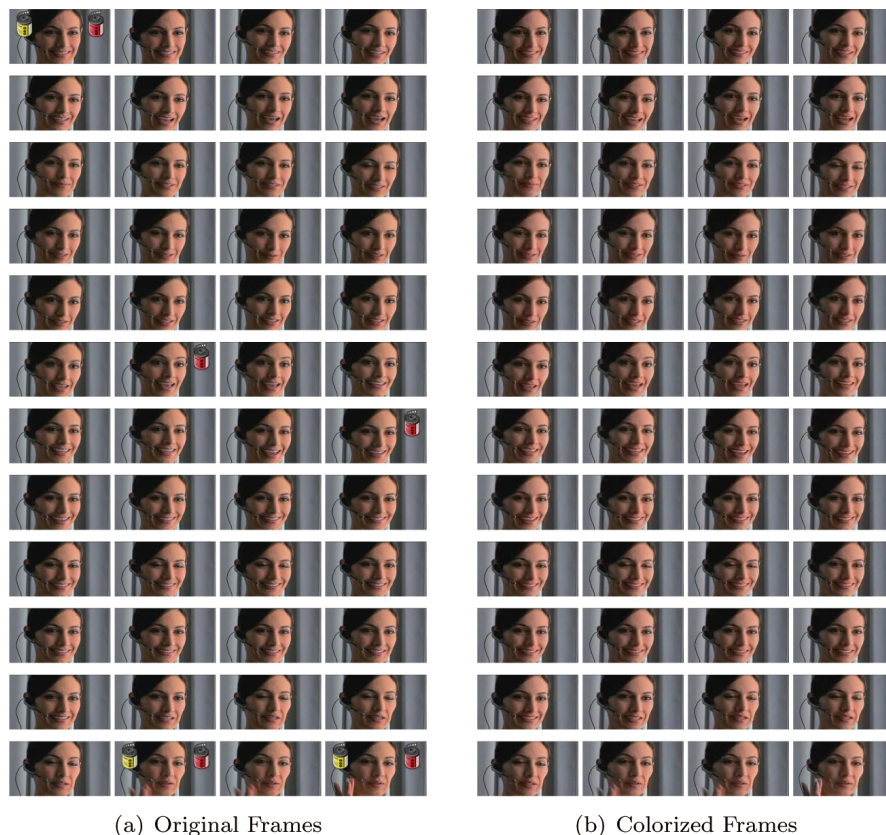


Figure 3. The first 48 frames of video telemarket. We put a yellow film icon where our method trained a new model and a red film icon for Cheng's approach.

for Cheng's approach. As can be seen, for the second red marker (row 6 and column 2), there is no big difference between the current frame and its preceding one. However, Cheng's approach select this frame to train a new colorization, which may not be necessary. On the contrary, for all the yellow markers, there is big difference between the current and preceding frames. Our approach only trains a new colorization model on these frames. Figure 3(b) shows the recovered color frames by using our approach.

In order to compare the video compression qualities of different approaches, we depict the PSNR values at each frame, as shown in Fig. 4(a). The x -axis is the frame number of the video, and the y -axis is the PSNR score of the uncompressed frame. The solid circles are drawn on the frames at which the colorization models are trained. The size of the circle is directly proportional to the number of selected color pixels.

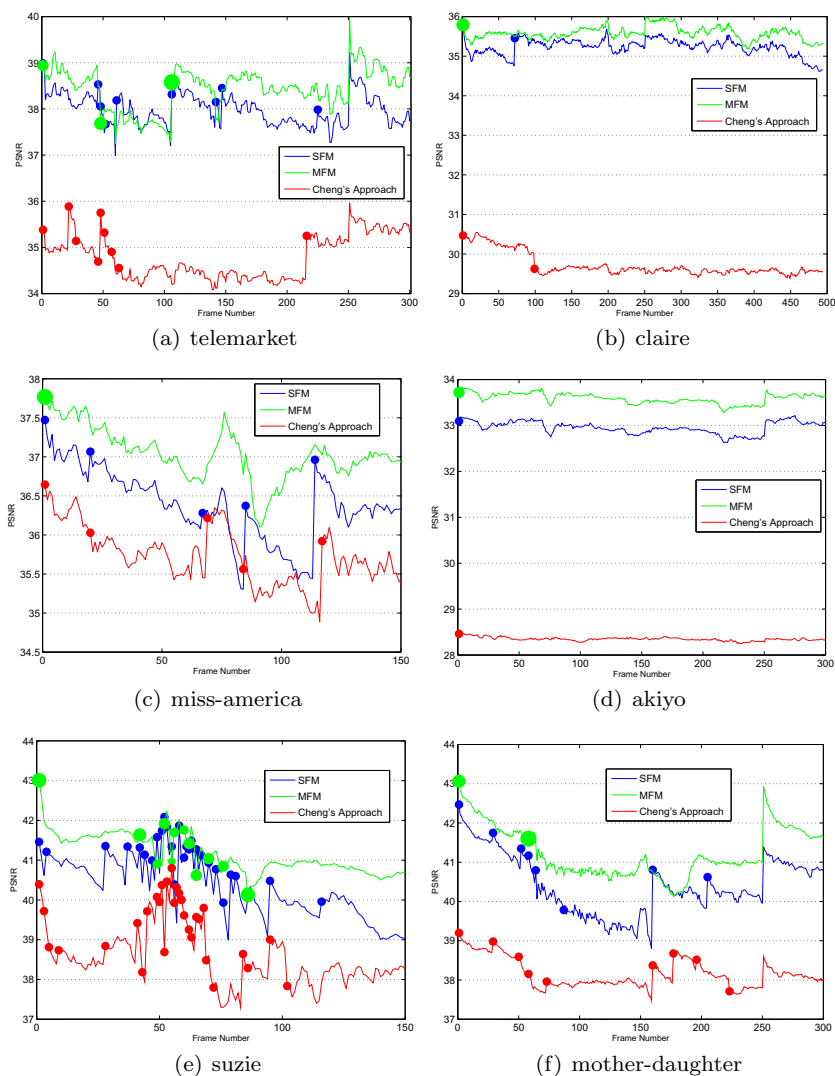


Figure 4. The PSNR plots. For all these examples, our approach obtains significantly higher PSNR than Cheng's approach.

As can be seen, with approximately the same total number of color pixels selected (and hence the similar compression ratio), our SFM and MFM approaches achieve much higher quality as measured by PSNR. Specifically, the *overall* PSNR^[15] values for Cheng's approach, SFM, and MFM are 34.8, 38.0, and 38.4, respectively.

Table 1 shows several other examples for video compression¹. Their PSNR plots are also depicted in Fig. 4. We also compare with the standard MPEG-4 compression result here. The original video is compressed by MPEG-4 codec with a similar compression ratio and the resultant PSNR is calculated. As can be seen, with similar compression ratio, our algorithms achieve higher PSNR than Cheng's method and sometimes beat the standard MPEG-4 encoder.

Table 1 Video compression comparison. cr denotes compression ratio

video	orig. size	gray. size	Cheng's approach		SFM		MFM		MPEG-4	
			cr	PSNR	cr	PSNR	cr	PSNR	cr	PSNR
claire	463452	44556	10.48%	29.7	10.48%	35.2	10.65%	35.6	10.48%	35.0
miss-america	229148	17082	11.82%	35.7	11.82%	36.4	10.60%	37.0	11.09%	36.0
akiyo	226780	26588	12.61%	28.3	12.61%	33.0	13.31%	33.6	12.77%	33.9
suzie	466278	98926	34.08%	38.5	33.23%	40.3	30.61%	41.1	39.29%	40.7
mother-daughter	466836	117302	28.98%	38.2	28.55%	40.4	27.70%	41.2	27.54%	39.9

In terms of computational efficiency. Since our methods use active learning to choose labeled points, more computational power is needed. However, since we avoid using costly Ncut preprocessing to obtain a super-pixel representation, the overall computation time of our methods is actually shorter than Cheng's method. But all the machine learning based algorithms are still much slower than the industrial standard MEGP-4.

7 Conclusions

We have presented a unified active and semi-supervised learning framework for video compression. The proposed algorithm is motivated from recent progresses on graph based semi-supervised learning^[3]. Using techniques from optimal experimental design, we select those points such that the size of the covariance matrix of the estimated coefficients is minimized. For video compression, the active learning algorithm is applied to select the most informative pixels, while the semi-supervised learning algorithm is applied for colorization. The experimental results have demonstrated that our proposed approach significantly outperforms the state-of-the-art approaches.

We hope the presented algorithm and the framework of analysis can provide with a new perspective for image and video analysis. The primary focus of this paper is on compression. However, similar idea may also be applied to other problems such as salient region detection. Moreover, an image can be considered as a matrix, or second order tensor, while a video can be considered as a third order tensor. It would be interesting to investigate the use of tensor based techniques^[19,20,21,22] for video compression.

¹ The videos used in this test are available at <http://trace.eas.asu.edu/yuv/index.html>.

References

- [1] Atkinson AC, Donev AN. *Optimum Experimental Designs*. Oxford University Press, 2007.
- [2] Belkin M, Niyogi P. Laplacian eigenmaps and spectral techniques for embedding and clustering. *Advances in Neural Information Processing Systems*. MIT Press, Cambridge, MA. 2001, 14: 585–591.
- [3] Belkin M, Niyogi P, Sindhvani V. Manifold regularization: A geometric framework for learning from examples. *Journal of Machine Learning Research*, 2006, 7: 2399–2434.
- [4] Cai D, He XF, Han JW. Document clustering using locality preserving indexing. *IEEE Trans. on Knowledge and Data Engineering*, December 2005, 17(12): 1624–1637.
- [5] Cheng Li, Vishwanathan SVN. Learning to compress images and videos. *Proc. of International Conference on Machine Learning*. 2007.
- [6] He XF, Cai D, Yan SC, Zhang HJ. Neighborhood preserving embedding. *Proc. of International Conference on Computer Vision (ICCV'05)*. Beijing, China. 2005.
- [7] He XF. Laplacian regularized d-optimal design for active learning and its application to image retrieval. *IEEE Trans. on Image Processing*. 2009. [to appear]
- [8] He XF, Ji M, Bao HJ. A unified active and semi-supervised learning framework for image compression. *IEEE International Conference on Computer Vision and Pattern Recognition*. Miami, FL. 2009.
- [9] Lu W, Gao XB, Li XL, Tao DC. An image quality assessment metric based contourlet. *International Conference on Image Processing*. 2008. 1172–1175.
- [10] Levin A, Lischinski D, Weiss Y. Colorization using optimization. *ACM SIGGRAPH*. Los Angeles, CA. 2004.
- [11] Li X, Lin S, Yan S, Xu D. Discriminant locally linear embedding with high-order tensor data. *IEEE Trans. on Systems, Man, and Cybernetics, Part B*, 2008, 38(2): 342–352.
- [12] Li X, Tao D, Gao X, Lu W. A natural image quality evaluation metric. *Signal Processing*, 2009, 89(4): 548–555.
- [13] Ren X, Malik J. Learning a classification model for segmentation. *Proc. of the 9th IEEE International Conference on Computer Vision*. France. 2003.
- [14] Roweis S, Saul L. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 2000, 290(5500): 2323–2326.
- [15] Shen K, Delp EJ. Wavelet based rate scalable video compression. *IEEE Trans. on Circuits and Systems for Video Technology*, 1999, 9(1).
- [16] Sherman J, Morrison WJ. Adjustment of an inverse matrix corresponding to a change in one element of a given matrix. *Annals of Mathematical Statistics*, 1950, 21: 124–127.
- [17] Shi J, Malik J. Normalized cuts and image segmentation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2000, 22(8): 888–905.
- [18] Schölkopf B, Smola AJ. *Learning with Kernels*. MIT Press, 2002.
- [19] Tao D, Li X, Wu X, Maybank SJ. General tensor discriminant analysis and gabor features for gait recognition. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2007, 29(10): 1700–1715.
- [20] Tao D, Li X, Wu X, Maybank SJ. Tensor rank one discriminant analysis - a convergent method for discriminative multilinear subspace selection. *Neurocomputing*, 2008, 71(10–12): 1866–1882.
- [21] Tao D, Song M, Li X, Shen J, Sun J, Wu X, Faloutsos C, Maybank SJ. Bayesian tensor approach for 3-d face modeling. *IEEE Trans. on Circuits and Systems for Video Technology*, 2008, 18(10): 1397–1410.
- [22] Vasilescu MAO, Terzopoulos D. Multilinear subspace analysis for image ensembles. *IEEE Conference on Computer Vision and Pattern Recognition*. 2003.