Proceedings of the 2009 IEEE International Conference on Systems, Man, and Cybernetics
San Antonio, TX, USA - October 2009

On Video Textures Generation: A Comparison Between Different Dimensionality Reduction Techniques

Wentao Fan and Nizar Bouguila
Institute for Information Systems Engineering
University of Concordia
Montreal, Canada
wenta_fa@ciise.concordia.ca, bouguila@ciise.concordia.ca

***Abstract*—Video texture is a new type of medium which can provide a new video with a continuously varying stream of images from a recorded video. It is created by reordering the input video frames in a way which can be played without any visual discontinuity. Recently, a new method of generating video textures has been proposed. It first apply principal components analysis (PCA) to extract signatures or patterns from the original video sequence, and then implement an autoregressive process (AR) model to synthesize new video textures. In this paper, we extend this video texture generation method by comparing PCA with other dimensionality reduction techniques such as probabilistic principal components analysis, kernel principal components analysis, independent component analysis, local linear embedding and Isomap. According to our experiments, these approaches prevail the original approach by providing us video textures with better quality.**

***Index Terms*—Video texture, computer vision, dimensionality reduction, autoregressive process**

I. Introduction

Video texture which has been introduced by Schödl *et al*. [1], is a new type of medium that can generate a continuous, infinitely changing stream of images from a recorded video. The term "video texture" is used because it is very similar to image textures. This technique can be considered as video-based rendering (VBR) because it has similar features with image-based rendering (IBR) technique [2], that is, both of them are able to reuse the already existing resources to synthesize new objects. For video textures, a recorded video is used to make a new video stream without any visual discontinuity by changing the order of the original frames. This would be useful in movie and game industries, since it may create new objects by reusing existing resources so that time and human resources may be saved. More applications may be found in [3] [4] [5] [6]. However, same as the original video textures technique, all of these works can only generate new video by just switching the order of frames and the result would suffer from 'dead-ends'.

Recently in computer vision, there are increased number of researches on time series analysis to model the dynamical characteristics of complex systems. Autoregressive (AR) process [7] is a tool used for understanding and predicting future values in a time series. In [8], Fitzgibbon have introduced a new method for creating video textures by applying principal components analysis (PCA) and AR process. All frames in the generated video are new and consist with the motions in the original video, and 'dead ends' would never appear. In [9], Campbell *et al*. have extended this approach to work with strongly non-linear sequences by applying a spline and a combined appearance model.

The rest of this paper is organized as follows: We briefly introduce different dimensionality reduction techniques in section 2. Then, we compare the experimental results of applying different dimensionality reduction techniques to generate video textures in section 3. Finally, in section 4 we conclude and describe some topics for future work.

II. Dimensionality Reduction Approaches

Dimensionality reduction is an important research topic in the area of data analysis. The goal of dimensionality reduction techniques is to discover a low-dimensional subspace that best represents a given set of data points. In this paper, we extend the work of Fitzgibbon [8] by comparing PCA with other representative dimensionality reduction techniques to extract signatures from video frames and then synthesize new video textures. The techniques we have applied are: probabilistic principal components analysis, kernel principal components analysis, Isomap, local linear embedding and independent component analysis. In this section, we will give a brief introduction for each technique individually.

A. Probabilistic Principal Component Analysis

In [10], Bishop has proposed a probabilistic model for PCA by showing that PCA can be represented as the maximum likelihood solution of a probabilistic latent variable model. This novel form of PCA is known as probabilistic principal component analysis (PPCA).

PPCA can be formulated by first choosing an explicit M-dimensional latent variable $\mathbf{z}$ corresponding to the principal component subspace and then sampling the D-dimensional observed variable $\mathbf{x}$ conditioned on this latent variable. We define a Gaussian prior distribution $p(\mathbf{z})$ for the latent variable, and a conditional Gaussian distribution $p(\mathbf{x}|\mathbf{z})$ for the observed

978-1-4244-2794-9/09/$25.00 ©2009 IEEE

variable $\mathbf{x}$ conditioned on the latent variable $\mathbf{z}$ is given by

$$p(\mathbf{x}|\mathbf{z}) = \mathcal{N}(\mathbf{x}|\mathbf{W}\mathbf{z} + \mu, \sigma^2\mathbf{I}) \ . \tag{1}$$

Where the latent variable distribution over $\mathbf{z}$ is $p(\mathbf{z}) = \mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{I})$, the $\mathbf{W}$ is a $D \times M$ linear transformation matrix, and the parameters μ and σ^2 govern the mean and variance of $\mathbf{x}$, respectively. By integrating the latent variable $\mathbf{z}$, the marginal distribution of $\mathbf{x}$ is given by

$$p(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\mu, \mathbf{W}\mathbf{W}^T + \sigma^2\mathbf{I}) \ . \tag{2}$$

The unknown parameters can be evaluated by using maximum likelihood as shown in [10]. We may obtain

$$\mathbf{W}_{ML} = \mathbf{U}(\mathbf{L} - \sigma^2\mathbf{I})^{1/2}\mathbf{R} \ . \tag{3}$$

$$\sigma^2_{ML} = \frac{1}{D-M}\sum_{i=M+1}^{D} \lambda_i \ . \tag{4}$$

where $\mathbf{U}$ is a $D \times M$ matrix whose columns are the eigenvectors of the covariance matrix of $\{\mathbf{x}_n\}$, the $M \times M$ diagonal matrix $\mathbf{L}$ contains the corresponding eigenvalues λ_n, $\mathbf{R}$ is an arbitrary rotational matrix. Furthermore, it is natural to extend PPCA to mixtures [11].

B. Kernel Principal Component Analysis

Kernel principal component analysis (Kernel PCA) [12] has been introduced as a nonlinear generalization of the conventional principal component analysis. First, a given data set $\{\mathbf{x}_n\}$ with N data points is mapped onto a high-dimensional feature space through a nonlinear transformation function $\phi(\mathbf{x})$. Then, we can perform standard PCA in this feature space, which represents a nonlinear PCA model in the original data space. The feature space is constructed using a kernel function $k(\mathbf{x}_n, \mathbf{x}_m) = \phi(\mathbf{x}_n)^T\phi(\mathbf{x}_m)$. Rather than computing the covariance matrix, Kernel PCA computes the principal eigenvectors of the kernel matrix $\mathbf{K}$. As described in [12], a projection f of a data point $\mathbf{x}$ onto a principal component in the feature space can be computed as

$$f = \sum_{n=1}^{N} a_{in}k(\mathbf{x}, \mathbf{x}_n) \ . \tag{5}$$

where a_{in} is an element of the coefficients $\mathbf{a}_i$ and is calculated as $\mathbf{a}_i = \frac{1}{\sqrt{\lambda_i}}\mathbf{v}_i$, $\mathbf{v}_i$ are the eigenvectors of the kernel matrix and λ_i are the corresponding eigenvalues. The performance of Kernel PCA highly depends on the choice of the kernel function. In our work, a 'gaussian kernel' with the form $k(\mathbf{x}, \mathbf{x}') = \exp(- \parallel \mathbf{x} - \mathbf{x}' \parallel^2 /0.1)$ is implemented.

C. Isomap

Isomap [13] is global nonlinear dimensionality reduction technique which reduces the dimensionality by finding a low dimensional manifold hidden in observational space. It extends multidimensional scaling (MDS) technique. Euclidean distance is used to calculate the dissimilarity between pairs of data points in MDS. In Isomap, instead of using Euclidean distance, Geodesic distance on the manifold is used as the measurement of dissimilarity. Isomap algorithm contains three major steps. The first step is to determine the neighborhood relations for all data points by constructing a weighted graph, it uses edge weights to indicate distances between all data points in original space. The following step is to discover the geodesic distances between all pairs of data points on the manifold by computing the shortest path distances in the weighted graph. The last step is to apply classical MDS to the dissimilarity matrix formed by the shortest path distances, constructing an embedding of the data points in a low-dimensional space. However, Isomap is not able to uncover the underlying spatio-temporal structure of the data set. This drawback can be solved by the modified version of Isomap: spatio-temporal Isomap (ST-Isomap) [14]. It augments the general Isomap framework to consider the temporal relationships in local neighborhoods of data points. Similar as the standard Isomap, ST-Isomap preserves the intrinsic geometry of the data, and it retains the structure of temporal coherence as well.

D. Locally Linear Embedding

Locally linear embedding (LLE) [15] is a local nonlinear dimensionality reduction technique. It is able to compute a low-dimensional embedding of the high dimensional data while preserving the local neighborhood structure of the original space. For a high-dimensional manifold, it can be decomposed into many small patches. If there exists sufficient data points, we may assume that each data point and its neighbors lie on or close to a locally linear patch of the manifold. Thus, each data point can be represented by a linear combination of its neighbors in a form of: $\mathbf{x}_i \approx \sum_j w_{ij}\mathbf{x}_j$. Here, w_{ij} is the coefficient of the local geometry of each patch. The detail of how to calculate the weight matrix $\mathbf{W}$ is described in [15]. The next step is to obtain a linear mapping of each neighborhood structure from high-dimensional coordinates to a lower-dimensional space. A mapping matrix $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_N]$ that preserves the local neighborhood information can be obtained by minimizing the embedding cost function:

$$\theta(\mathbf{Y}) = \sum_i \parallel \mathbf{y}_i - \sum_j w_{ij}\mathbf{y}_j \parallel^2 \tag{6}$$

Here, the vector $\mathbf{y}_i$ is the global internal coordinates of the data point $\mathbf{x}_i$ on the manifold.

E. Independent Component Analysis

Independent component analysis (ICA) [16] is a statistical technique for separating a multidimensional random vector into linear additive subcomponents which are maximally statistically independent from each other. One of the most essential features of ICA model is nongaussianity, this feature is the key to reveal the independence of each components.

The goal of ICA is to maximize the statistical independence between the components of the basis vectors. Unlike other dimension reduction techniques, ICA searches for components which are both statistically independent and non-gaussian. We can find the independent components by maximizing nongaussianity, it is an important principle in ICA estimation.

Fig. 1. (a) The original input video.

III. Experimental results

In our experiments, the process for generating new video textures contains three steps: the first step is to implement the dimensionality reduction techniques to extract the signatures from the frames of a input video. The second step is to apply an AR process to predict new frame signatures based on the signatures we obtained in the previous step. Here, we have a time series made of a sequence of frame signatures $\{\mathbf{x}_n\}$ where $n = 1, ..., N$, and N is the total number of frames. A zero-mean AR process of order p for a series of frame signatures in a d-dimensional space may be modelled by:

$$\mathbf{x}_n = \sum_{k=1}^{p} A_k \mathbf{x}_{n-k} + \mathbf{w} \tag{7}$$

Where the $d \times d$ matrices $A \equiv (A_1, ..., A_p)$ are the coefficient matrices of the AR process model. And $\mathbf{w}$ is a d-dimensional random vector drawn from Gaussian white noise distribution with zero-mean. In our case, the order p is set to 2, which means, the next frame signature will be generated based on the previous 2 signatures. The last step is to project these new frame signatures back to the image space and compose them together as a video texture. All of our experiments are done by using *Matlab* on a Windows platform.

A. Extract Frame Signatures and Synthesize New Video Textures

In order to test the effect of choosing different dimension reduction techniques for creating video textures, we implemented several input videos.[1] For instance, one of the input videos we used is a video clip of a person moving a pen (as shown in Fig. 1). It is 15 seconds long and contains 450 frames, and each frame has a size of 160 $\times$ 128 (20480) pixels . For each frame, the new image vector would be a vector with 20480 dimensions, and the total image matrix for 450 frames is a 450×20480 matrix, with images in the rows, and dimensions in the columns.

After acquiring the image matrix, we apply different dimensionality reduction techniques to capture the signatures of frames. In our experiments, the dimensionality of 30 to 40 is good enough for representing each individual frame. In the implementation of LLE and Isomap approaches, we use K-nearest method with the number of neighbors $K = 12$. In the case of applying ICA, we implemented the FastICA algorithm. A 'gaussian kernel' with the form $k(\mathbf{x}, \mathbf{x}') = \exp(- \parallel \mathbf{x} - \mathbf{x}' \parallel^2 /0.1)$ is implemented for the Kernel PCA. At the end of this step, each frame is represented by its signature which only has 30 dimensions. [2]

Subsequently, AR process is applied to synthesize new signatures. The number of new signatures is decided manually, for this input video, we set it to 60. At the end, 60 new frames are synthesized and then define our video texture.

[1] Some of our input test movies are obtained from the "Video Textures" web site: http://www.cc.gatech.edu/cpl/projects/videotexture

TABLE I
Performances of Different dimension reduction techniques for generating video textures

Technique	Complexity	Run-time
PCA	$O(D^3)$	30.56
PPCA	$O(nD^2)$	38.17
Kernel PCA	$O(n^3)$	60.41
Isomap	$O(n^3)$	56.05
LLE	$O(Dn^2)$	45.49
ICA	$O(Dn^2)$	49.20

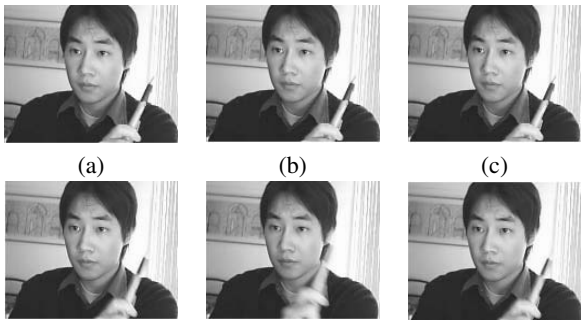

Fig. 2. The first frame that is synthesized by using: (a) PCA , (b) PPCA, (c) kernel PCA, (D) Isomap, (e) LLE and (f) ICA.

B. Comparison of the results

We have Implemented six different approaches (PCA, PPCA, Kernel PCA, Isomap, LLE and ICA) separately, all of them can produce good quality video textures. Each frame in the result is new and consists with the motions in the original video sequence. Fig. 2 illustrates the first frame of the result generated by PCA, PPCA, kernel PCA, Isomap, LLE and ICA respectively. Table I demonstrates the computational complexity and the run-time for generating the new video texture. In Table I, D is the dimensionality of the input data in the observed space, n represents the number of input data points and the unit for measuring run-time is in seconds. The Fig. 3 shows the signatures extracted by applying different dimension reduction techniques.

Although the result seems very appealing, there still exist one problem, which is the occurrence of noise (i.e frames contain some 'ghost' in it as shown in Fig. 4). For all the results, after some periods, the noise will start to become visible and make the video blur. The reason is that the AR process model predicts the new frame signature based on previous frames, the

[2] We have also applied mixtures of probabilistic principal component analysis [11] to generate the signatures of video frames, but the only good result we obtained is when the mixture size is equal to one which is same as the standard PPCA.

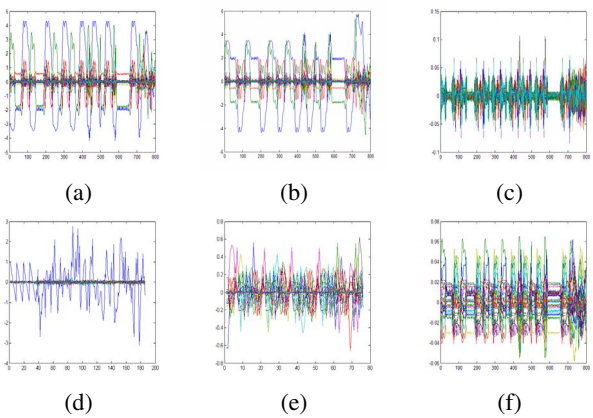

Fig. 3. The signatures of frames extracted by (a) PCA , (b) PPCA, (c) Kernel PCA, (D) Isomap, (e) LLE and (f) ICA. The last 100 signatures are newly created by AR process. Here, the x-axis means the number of frames and the y-axis represents the value of coefficient of each input frame. Each color stands for one dimensionality of the frame signature (up to 30-dimensions).

Fig. 4. The synthesized video frame which contains visible noise.

noise is cumulated as the AR process iteratively generates more new signatures. And this feature reflects the difference among PCA, PPCA, Kernel PCA, Isomap, LLE and ICA approaches for generating video textures. Table II demonstrates the noise start to appear at which frame in the synthesized video texture.

From Table II, we may notice that for PCA, the noise starts to blur the frame at 16th frame, which is the earliest among all the six techniques. For PPCA and ICA, the noise appeared later than others (at 50th and 45th frames, respectively). That means, a more robust decomposition approach such as PPCA and ICA would generate more typical signature and therefore can improve the performance of the result. Among all the six approaches, PPCA provides the best result. This is reasonable, for PPCA, it is a latent variable model which conditioned on the observations data set, and it naturally reduces the dimensionality of the data from a probabilistic perspective. The order p of the AR process is another factor that may affect the performance of synthesizing video textures. According to

TABLE II
THE NOISE START TO BLUR THE RESULT AT WHICH FRAME NUMBER

Dimensionality Reduction Techniques	Frame Number
PCA	16th
PPCA	50th
Kernel PCA	18th
Isomap	24th
LLE	20th
ICA	45th

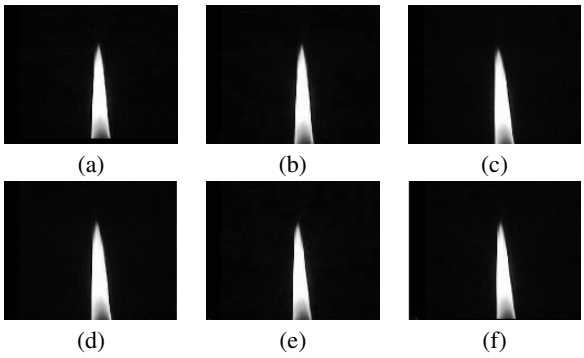

Fig. 5. The first frame that is synthesized by using: (a) PCA , (b) PPCA, (c) Kernel PCA, (D) Isomap, (e) LLE and (f) ICA

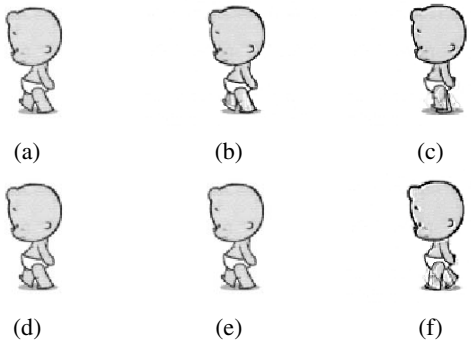

Fig. 6. The first frame that is synthesized by using: (a) PCA , (b) PPCA, (c) Kernel PCA, (D) Isomap, (e) LLE and (f) ICA for an animation of cartoon.

our experiments, as the value of p increases, the noise may blur the results earlier and faster. The reason is that, order p represents the amount of previous signatures that will be used for predicting a new signature, higher value of p will make the noise accumulated much faster. In our case, the best results are obtained when p is set to 2.

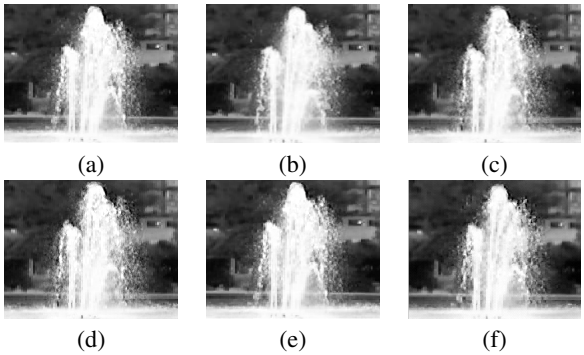

Fig. 7. The first frame that is generated by using: (a) PCA , (b) PPCA, (c) Kernel PCA, (D)Isomap, (e) LLE and (f) ICA for a movie of fountain.

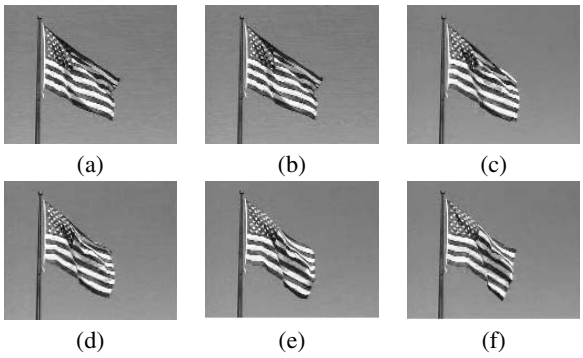

Fig. 8. The first frame that is generated by using: (a) PCA , (b) PPCA, (c) Kernel PCA, (D)Isomap, (e) LLE and (f) ICA for a movie of flag.

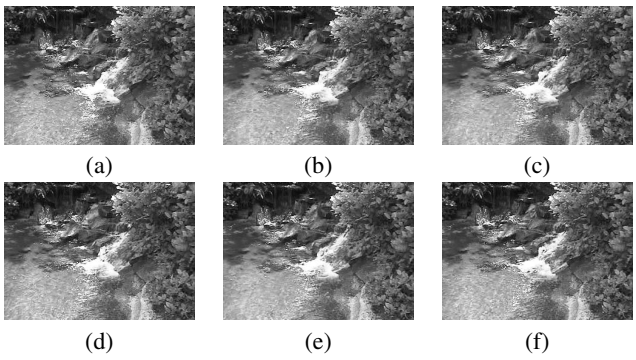

Fig. 9. The first frame that is generated by using: (a) PCA , (b) PPCA, (c) Kernel PCA, (D)Isomap, (e) LLE and (f) ICA for a movie of waterfall.

C. More experimental results

We have also experimented other movies during our study. Here, the goal is to test the video texture generation approach on some structurally complex scenarios such as fountain. And we also investigated to use it on some cartoon movies, since compared to the real shooted movie, cartoon movies contain more sparse space and less color intensity. Fig. 5 shows the synthesized result from a movie which contains a movement of a burning candle for each dimensionality reduction technique. In Fig. 6, we illustrate the generated video textures for a cartoon animation. Fig. 7 demonstrates the results of the video textures for a movie of fountain. Fig. 8 and Fig. 9 show the video texture results of a movie of waving flag and a waterfall, respectively. Tables III demonstrates at which frame number the noise start to become visible for the flame, cartoon, fountain, flag and waterfall movies, respectively. Here, from Table III we may notice that, for cartoon movie, IsoMap and LLE provide better solution. This is because a cartoon movie are more spare compared to the real movies, and dimensionality reduction techniques based on preserving the neighborhood relationship such as Isomap and LLE are more suitable than others [3].

TABLE III
THE NOISE START TO BLUR THE RESULT AT WHICH FRAME NUMBER FOR DIFFERENT INPUT MOVIES AND TECHNIQUES)

Movie	PCA	PPCA	K-PCA	Isomap	LLE	ICA
Flame (Fig. 5)	17th	47th	21th	20th	18th	38th
Cartoon (Fig. 6)	20th	34th	24th	38th	35th	30th
Fountain (Fig. 7)	13th	39th	15th	22th	24th	35th
Flag (Fig. 8)	17th	40th	19th	25th	22th	34th
Waterfall (Fig. 9)	14th	36th	15th	24th	23th	33th

IV. Conclusion and Future work

In this paper, we have extended the work of applying PCA and AR process to generate video textures by replacing PCA with five other dimension reduction techniques (PPCA, Kernel PCA, Isomap, LLE and ICA). Based on our experiments, by using these dimensionality reduction techniques, the quality of video textures are improved compared with applying PCA to extract frame signatures. The synthesized video textures may contain similar motions as the input video and will never be repeated exactly. All the frames in the results are synthesized and have never appeared before. By comparing the experimental results among these dimensionality reduction techniques, PPCA provides more typical signatures which can be used to synthesize video texture than others. But unfortunately, noise may still appear and makes the video frame blur after some period of time. The results can be improved by reducing the noise during the process of synthesizing new signatures. One of the possible solutions for improving the performance is to apply a more sophisticated model, for predicting the new data, other than autoregressive process model.

Acknowledgment

The completion of this research was made possible thanks to the Natural Sciences and Engineering Research Council of Canada (NSERC) and a NATEQ Nouveaux Chercheurs Grant. Finally, the authors thank SMC reviewers for their constructive comments.

References

[1] A. Schödl, R. Szeliski, D. Salesin and I. Essa, "Video textures," In: Proceedings of the 27th annual conference on Computer graphics and interactive techniques, pp. 489–498, ACM Press/Addison-Wesley Publishing Co. New York (2000)

[2] M. M. Oliveira, "Image-based modeling and rendering techniques: A survey," RITA, vol. 9, pp. 37–66 (2002)

[3] A. Schödl and I. Essa, "Machine learning for video-based rendering," Advances in Neural Information Processing Systems. vol. 13, pp. 1002–1008 (2001)

[4] A. Schödl and I. Essa, "Controlled animation of video sprites," ACM SIGGRAPH Symposium on Computer Animation. pp. 121–128 (2002)

[5] A. Agarwala, C. Zheng, C. Pal, M. Agrawala, M. Cohen, B. Curless, D. Salesin and R. Szeliski, "Panoramic video textures," ACM Transactions on Graphics. vol. 24, pp. 821–827 (2005)

[6] Y. Li, T. Wang and H. Shum, "Motion texture: a two-Level statistical model for character motion synthesis," In Proceedings of ACM SIGGRAPH 2002, pp. 465–472. ACM, New York (2002)

[7] S. M. Pandit and S. M. Wu, "Time series and system analysis with applications," John Wiley & Sons Inc. (1983)

[3] ST-IsoMap is also tested for cartoon movies, it can generate a better results than all other results. A work based on using ST-IsoMap to generate cartoon textures can be found in [17].

[8] A. W. Fitzgibbon, "Stochastic rigidity: image registration for nowhere-static scenes," In: Proceedings of International Conference on Computer Vision (ICCV) 2001, vol. 1, pp. 662–669 (2001)
[9] N. Campbell, C. Dalton, D. Gibson and B. Thomas, "Practical generation of video textures using the auto-regressive process," Image and Vision Computing. vol. 22, pp. 819–827 (2004)
[10] C. M. Bishop, "Probabilistic principal component analysis," Journal of the Royal Statistical Society, Series B. vol. 61, pp. 611–622 (1999)
[11] C. M. Bishop, "Mixtures of probabilistic principal component analyzers," Neural Computation. vol. 11, pp. 443–482, The MIT Press (2000)
[12] B. Schölkopf, A. Smola and K. Müller, "Nonlinear component analysis as a kernel eigenvalue problem," Neural Computation. vol. 10, pp. 1299–1319, MIT Press, Cambridge (1998)
[13] J. B. Tenenbaum, V. de Silva and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," Science. vol. 290, pp. 2319–2323 (2000)
[14] O. C. Jenkins and M. J. Mataric, "A spatio-temporal extension to Isomap nonlinear dimension reduction," In: Proceedings of the twenty-first international conference on Machine learning, ACM International Conference Proceeding Series, pp. 441–448. ACM, New York (2004)
[15] S. Roweis and L. Saul, "Nonlinear dimensionality reduction by locally linear embedding," Science. vol. 290, pp. 2319–2323 (2000)
[16] A. Hyvarinen and E. Oja, "Independent component analysis: algorithms and applications," Neural Networks. vol. 13, pp. 411–430 (2000)
[17] C. Juan and B. Bodenheimer, "Cartoon textures," In: Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation pp. 181–184. Eurographics Association, Switzerland (2004)