Global Coordination Based on Matrix Neural Gas for Dynamic Texture Synthesis*

Banchar Arnonkijpanich[1] and Barbara Hammer[2]

[1] Department of Mathematics, Faculty of Science,
Khon Kaen University, Thailand, 40002
and
Centre of Excellence in Mathematics, The Commission on Higher Education,
Si Ayutthaya Rd., Bangkok 10400, Thailand
[2] University of Bielefeld, CITEC, Bielefeld, Germany

Abstract. Matrix neural gas has been proposed as a mathematically well-founded extension of neural gas networks to represent data in terms of prototypes and local principal components in a smooth way. The additional information provided by local principal directions can directly be combined with charting techniques such that a nonlinear embedding of a data manifold into low dimensions results for which an explicit function as well as an approximate inverse exists. In this paper, we show that these ingredients can be used to embed dynamic textures in low dimensional spaces such that, together with a traversing technique in the low dimensional representation, efficient dynamic texture synthesis can be obtained.

1 Introduction

Neural gas (NG) and topology representing networks as proposed by Martinetz constitute particularly robust methods to represent a given data set and its topology in terms of a lattice of neurons [11,12]. In contrast to the popular self-organizing map [7], no prior lattice structure is specified such that direct visualization of data is not possible. On the contrary, the correct, probably irregular topology of the underlying data manifold can be inferred which accounts for the particular robustness of the approach.

Neural gas is often used for data preprocessing, e.g. data compression or clustering. Recently, extensions of NG have been proposed which also adapt local matrices during training such as to minimize the quantization error [13,1]. This corresponds to local coordinate systems which represent smooth local principal directions of data. It has been demonstrated in [2], that these additional parameters offer sufficient information to extract explicit local coordinate systems from the data which can be further processed to obtain a global nonlinear projection of the underlying manifold e.g. using manifold charting [3]. This way, an explicit mapping together with its approximate inverse is obtained which can map high dimensional data into low dimensional space. In [2], the possibility to use this

* This research is partially supported by Centre of Excellence in Mathematics, the Commission on Higher Education, Thailand.

F. Schwenker and N. El Gayar (Eds.): ANNPR 2010, LNAI 5998, pp. 84–95, 2010.

mapping for low dimensional data visualization and representation has been explored in comparison to popular alternative visualization schemes as referenced e.g. in [10]. Unlike popular alternative visualization methods such as locally linear embedding, maximum variance unfolding, or stochastic neighbor embedding, manifold charting in combination with matrix neural gas does not only embed the given data points, but it provides an explicit low dimensional embedding of the data manifold and an approximate inverse of the map. Hence additional information is available which can be explicitly used in further applications.

In this contribution, we will demonstrate the suitability of manifold embedding by matrix neural gas for an interesting problem from computer graphics, the efficient synthesis of dynamic texture based on given examples. Dynamic texture synthesis is the process of producing an animation of dynamic textures which preserve the behavior of the system similar to its original appearance. There exist two fundamentally different approaches for dynamic texture synthesis: physics-based methods generate dynamic texture based on mathematical models of the natural phenomena, see e.g. [18,4]. Physics-based models provide a flexible synthesis, since the dynamic texture can be controlled through a few parameters in a mathematical equation system. The drawback of this approach is that each model is appropriate only for a particular texture and cannot be transferred to other domains. As an alternative, image-based models overcome this limitation. They use a global model for different textures and synthesize dynamic textures from a model based on the appearance of the whole texture in a series of images. Different principled approaches can be distinguished such as simple extensions of static texture synthesis to 3D [20], spatiotemporal models based on the pixel level [16], or dynamical models on the image level, such as proposed in [15,6]. The latter approach is particularly promising since it can capture common non-trivial dynamical development such as rotation. This way, dynamic texture synthesis becomes a problem of system identification based on a sequence of image data such that dynamic texture can be directly generated along the trajectory of the system based on given initial conditions.

Typically, image sequences possess a very high dimensionality such that system identification is not possible in the raw image space. Therefore, the approaches presented in [15,6] first project the images onto low dimensional space with a standard principal component analysis (PCA), performing system identification e.g. using classical linear auto-regressive models in the low dimensional projection space, afterwards. Since PCA gives rise to an approximate inverse by means of the pseudo-inverse of the transformation matrix, dynamic textures can be generated this way. The resulting model is rather flexible, but it has the drawback that a global linear embedding is used such that images are not appropriately sampled and represented in particular at points in time with rapid movements (e.g. flapping flag). Therefore, it has been proposed e.g. in [8,9] to use recent nonlinear manifold learning techniques as proposed in machine learning instead of a global linear embedding.

In this contribution, we demonstrate that recent matrix learning schemes for neural gas together with a global coordination technique, manifold charting,

give rise to a nonlinear manifold embedding which can successfully be used in this context. This way, neural low-dimensional nonlinear manifold embedding can serve as an essential step in the highly non-trivial application in computer graphics to automated dynamic texture synthesis. Now, we first introduce matrix neural gas which allows us to extract local linear manifold projections from a given data set. These can be combined to a global nonlinear embedding with approximate inverse using manifold charting. We describe the inclusion of this method into the general pipeline for dynamic texture synthesis, afterwards, and we demonstrate its applicability in a variety of examples.

2 Nonlinear Manifold Embedding Based on Matrix Neural Gas

Matrix neural gas

Assume data $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m\} \subset \mathbb{R}^N$ are sampled from a manifold $X \subset \mathbb{R}^N$. The aim of neural gas is to represent the given data in terms of prototypes $\{\boldsymbol{w}_1, \ldots, \boldsymbol{w}_k\} \subset \mathbb{R}^N$ faithfully such that the prototype $\boldsymbol{w}_i$ adequately resembles its receptive field $R_i := \{\boldsymbol{x} \mid i = \text{argmin}_j\{d(\boldsymbol{x}, \boldsymbol{w}_j)\}\}$ where usually the Euclidean distance is used

$$d(\boldsymbol{x}, \boldsymbol{w}_i) = (\boldsymbol{x} - \boldsymbol{w}_i)^t(\boldsymbol{x} - \boldsymbol{w}_i)\,. \tag{1}$$

NG has been derived in [11] as a stochastic gradient descent of the following cost function

$$E_{\text{NG}}(\boldsymbol{w}) \sim \frac{1}{2}\sum_{i=1}^{k}\int h_\sigma(k_i(\boldsymbol{x}))\cdot d(\boldsymbol{x}, \boldsymbol{w}_i)\, P(d\boldsymbol{x}) \tag{2}$$

where P refers to the probability distribution of the data points $\boldsymbol{x}$. $k_i(\boldsymbol{x}) \in \{0, \ldots, k-1\}$ constitutes a permutation of prototypes arranged according to the distance, i.e.

$$k_i(\boldsymbol{x}) := |\{\boldsymbol{w}_j \mid d(\boldsymbol{x}, \boldsymbol{w}_j) < d(\boldsymbol{x}, \boldsymbol{w}_i)\}|$$

If distances coincide, ties are broken deterministically. $h_\sigma(t) = \exp(-t/\sigma)$ is an exponential curve with neighborhood range $\sigma > 0$. For vanishing neighborhood $\sigma \to 0$, the standard quantization error is obtained.

As an alternative to online optimization, a batch approach can be taken if data are given in advance. For a discrete finite data set, the cost function (2) can be optimized in a batch scheme repeatedly updating prototypes and rank assignments until convergence [5]. Usually, during training, the neighborhood range σ is annealed to 0 such that the quantization error is recovered in final steps. In intermediate steps, a neighborhood structure of the prototypes is determined by the ranks according to the given training data. This choice accounts for a high robustness of the algorithm with respect to local minima of the quantization error, further, a smooth update of neighbored points is guaranteed this way [12].

Classical NG relies on the Euclidean metric which induces isotropic cluster shapes. More general ellipsoidal shapes can be achieved by the generalized metric form

$$d_{\Lambda_i}(\boldsymbol{x}, \boldsymbol{w}_i) = (\boldsymbol{x} - \boldsymbol{w}_i)^t \Lambda_i (\boldsymbol{x} - \boldsymbol{w}_i) \tag{3}$$

instead of the squared Euclidean metric (1) where $\Lambda_i \in \mathbb{R}^{N \times N}$ is a symmetric positive definite matrix with $\det \Lambda_i = 1$. These constraints are necessary to guarantee that the resulting formula defines a metric which does not degenerate to a trivial form ($\Lambda_i = 0$ constituting an obvious trivial optimum of the cost functions). A general matrix Λ_i can account for correlations and appropriate nonuniform scaling of the data dimensions. The parameters Λ_i in (3) can be optimized during training together with the prototype parameters and assignments. The corresponding cost function (2) which uses (3) instead of (1) can be optimized in batch mode, yielding matrix NG (MNG):

init $\boldsymbol{w}_i$ randomly
init Λ_i as identity I
repeat until convergence
set $k_{ij} := k_i(\boldsymbol{x}_j)$
set $\boldsymbol{w}_i := \sum_j h_\sigma(k_{ij})\boldsymbol{x}^j / \sum_j h_\sigma(k_{ij})$
set $\Lambda_i := S_i^{-1}(\det S_i)^{1/n}$ where
$S_i := \sum_j h_\sigma(k_{ij})(\boldsymbol{x}^j - \boldsymbol{w}^i)(\boldsymbol{x}^j - \boldsymbol{w}^i)^t$

It has been shown in [1] that this update scheme converges to a local optimum of the NG cost function under mild conditions.

Local linear projections

The matrix S_i corresponds to the correlation of the data centered at prototype $\boldsymbol{w}_i$ and weighted according to its distance from the prototype. For vanishing neighborhood $\sigma \to 0$, the standard correlation matrix of the receptive field is obtained. The resulting Mahalanobis distance corresponds to a scaling of the principal axes of the data space by the inverse eigenvalues in the eigendirections. Thus, ellipsoidal cluster shapes arise which are aligned according to local principal components of the data. Since neighborhood cooperation is applied to both, prototype adaptation and matrix learning during batch training, a regularization of matrix learning is given and neighbored matrices have a similar form.

Local matrices as learned by MNG provide local linear transformations of the data in the following way: Assume the eigenvalue decomposition

$$\Lambda_i = \Omega_i^t \cdot D_i \cdot \Omega_i$$

is given with a diagonal matrix D_i of eigenvalues and eigenvectors collected in Ω_i. Assume data should be mapped to dimensionality n where, usually, $n < N$. Then we can reduce D_i to only the d smallest eigenvalues (which are the main eigenvalues of S_i, i.e. they belong to the main principal components of the receptive field) getting the $n \times N$ matrix D_i^{red}. The formula

$$A_i : \mathbb{R}^m \to \mathbb{R}^d, \boldsymbol{x} \mapsto D_i^{\mathrm{red}} \cdot \Omega_i^t \cdot (\boldsymbol{x} - \boldsymbol{w}_i) \tag{4}$$

gives the local linear projection of the data points to the main principal components of the receptive field induced by the ith chart of the data manifold. If n is

chosen at most 3, every map A_i provides a linear visualization of the manifold which is faithful within the receptive field of prototype $\boldsymbol{w}^i$ because it corresponds to the main eigenvalues of the local chart, as proposed in the contribution [2].

Note that, depending on the dimensionality N of the original data points, full matrix learning in MNG is rather time consuming, requiring matrix inversion of order $\mathcal{O}(N^3)$ in every step. Since only the minor n eigenvalues of the matrix Ω_i are relevant for the local projection, we can priorly reduce the matrix such that the scaling of only the minor d principal components is individually adapted while the scaling remains identical (and nonzero to avoid degeneration) for all other directions. This can be achieved efficiently with any algorithm which extracts the major n principal components of the generalized data correlation matrix, e.g. a generalized Sanger rule as proposed in [13], reducing the matrix determination to $\mathcal{O}(N)$. It has been demonstrated in [13] that the reduction to the largest n principal components can be explicitely included into the metric computation step, such that an overall reduction of the complexity to $\mathcal{O}(N)$ results, assuming independence of the number of iterations for eigenvector learning and neural gas of N.

Global coordination by manifold charting

The projections (4) provide valid local linear transformations of the data in the neighborhood of the respective prototype. Different methods which allow to combine these local projections to a global map have been proposed e.g. [17,3]. We will rely on manifold charting as introduced in [3] which glues the linear pieces together such that a good agreement can be observed at the overlaps.

Assume that linear projections $A_1, \ldots, A_k$ are given by formula (4) which define k local projections of the data points $\boldsymbol{z}_{1i} = A_1(\boldsymbol{x}_i), \ldots, \boldsymbol{z}_{ki} = A_k(\boldsymbol{x}_i)$ of the data points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m$. Assume that, in addition, a responsibility value $p_{ij} = p_i(\boldsymbol{x}_j)$ is specified for every data point $\boldsymbol{x}_j$ and chart A_i which defines the responsibility of this chart for the data point, whereby $\sum_i p_{ij} = 1$ for every j. Here, we can use Gaussian bells centered at the prototypes to arrive at these responsibilities. More precisely, set $N_i = |R_i|$ as the number of points in the ith receptive field. Let S_i be the correlation matrix of the ith receptive field as computed in MNG and $\tilde{S}_i = S_i/N_i$ the associated matrix. Then we set the responsibility of the ith receptive field for point $\boldsymbol{x}_j$ as

$$\tilde{p}_{ij} = \tilde{p}_i(\boldsymbol{x}_j) = \frac{N_i}{p} \cdot \frac{1}{(2\pi)^{m/2}\sqrt{|\tilde{S}_i|}} \cdot \exp(-0.5 \cdot (\boldsymbol{x}_j - \boldsymbol{w}_i)^t \cdot \tilde{S}_i^{-1} \cdot (\boldsymbol{x}_j - \boldsymbol{w}_i)) \quad (5)$$

where the prior N_i/p refers to the relative number of points in chart i. The responsibilities p_{ij} are obtained thereof by normalization $p_{ij} = \tilde{p}_{ij} / \sum_i \tilde{p}_{ij}$.

The goal is to combine the local charts A_i by means of local affine mappings $B_i : \mathbb{R}^n \to \mathbb{R}^n$ to a global mapping such that the compositions lead to matching points if more than one chart is responsible for a data point. The mappings B_i are determined in such a way that the following costs are minimized

$$E_{\text{charting}} = \frac{1}{2} \cdot \sum_{i,j,l} p_{ji} p_{li} \|B_j(\boldsymbol{z}_{ji}) - B_l(\boldsymbol{z}_{li})\|^2 \quad (6)$$

which (as can be seen by a simple algebraic transformation) also give the difference of the globally mapped points and the local affine transformations of the points. As shown in [3] a unique analytic solution of this problem can be found. The final points are then obtained by the formula $\boldsymbol{x}_j \mapsto \sum_i p_{ij} B_i(\boldsymbol{z}_{ij})$.

Note that, depending on the distribution of the prototypes, the assignments will be sparse since p_{ij} will be almost zero for receptive fields i which lead to a high rank w.r.t $\boldsymbol{x}_j$. To speed up the computation, it is possible to cut off these small assignments and work with sparse matrices. Based on these choices of p_{ij} provided by (5) and $\boldsymbol{z}_{ij}$ provided by the affine transformations (4), MNG can be combined with manifold charting to give a global nonlinear visualization of data. Obviously, this visualization can be described by an explicit mapping of $\mathbb{R}^N \to \mathbb{R}^n$ by means of the formula

$$\boldsymbol{x} \mapsto \sum_i p_i(\boldsymbol{x}) \cdot B_i(A_i(\boldsymbol{x})) \tag{7}$$

where $p_i(\boldsymbol{x})$ is computed according to (5), the local linear mappings A_i are given by (4), and the affine transformations B_i to glue the charts together are determined solving equation (6).

Inverse map

To arrive at an approximate inverse map, we take a simple point of view which allows us to compute the inverse algebraically. Note that every local linear projection A_i possesses an approximate inverse A_i^{-1} induced by the pseudo-inverse of $D_i^{\text{red}} \cdot \Omega_i$. Since A_i maps to lower dimensions, this is, of course, no exact inverse but its best approximation in a least squares sense. Further, obviously, the affine transformations B_i can be inverted exactly. Thus, for every $\boldsymbol{z} \in \mathbb{R}^n$, an approximate inverse of the image of (7) can be determined in the following way: for a given $\boldsymbol{x}$, we determine the inverse under chart i: $A_i^{-1} \circ B_i^{-1}(\boldsymbol{z})$. From these possibly preimages, we take the one with maximum responsibility according to (5).

3 Dynamic Texture Synthesis by System Traversal

In [8,9], dynamic texture synthesis is modelled as a system identification problem. First, images are nonlinearly mapped to a low-dimensional space. In low dimensions, a method to track temporal developments based on initial conditions is defined. Since every point in the embedding space can be inversely mapped to a point in the original high dimensional space, a sequence of images corresponding to a dynamic texture results. This way, a compressed representation of dynamic textures can be obtained since it is sufficient to store the parameters of the global nonlinear map and its inverse and only the low dimensional projections of the given image sequence which, for $n \ll N$, requires much less space than the original texture sequence. Further, interpolation of texture images as well as generation of texture based on new starting points becomes possible since a model is available to track the dynamics in the low dimensional projection space.

The contribution [9] relies on a mixture of probabilistic principal component analysis (MPPCA) together with global coordination to obtain a global nonlinear embedding of the data manifold [19,17]. Here we propose to substitute MPPCA by matrix NG since, as we will demonstrate in experiments, a greater robustness and smoothness of the method can be achieved. Thus, we combine global coordination based on matrix NG as described in the previous section with the tracking dynamics in the low dimensional projection space as introduced in [9]. For convenience, we shortly describe the traversal mechanism as proposed in [9].

Assume a dynamic texture is given which, making the temporal dependency explicit, is denoted as $\boldsymbol{x}(t)$, $\boldsymbol{x}(t+1)$, $\ldots \in \mathbb{R}^N$. The corresponding low-dimensional projections are denoted as $\boldsymbol{z}(t)$, $\boldsymbol{z}(t+1)$, $\ldots \in \mathbb{R}^n$. Motion prediction starts from a sequence of at least two points $\boldsymbol{g}(t-1)$, $\boldsymbol{g}(t)$ in $\mathbb{R}^n$ which are probably obtained as projections of images. Now the temporal successors of $\boldsymbol{g}(t)$ are obtained in six steps based on the low dimensional vectors $\boldsymbol{z}(i)$ as follows (σ_1, α, σ_2 are positive parameters):

1. Sampling neighbors: K nearest neighbors $\mathcal{N}$ of $\boldsymbol{g}(t)$ are sampled from the data $\boldsymbol{z}(i)$ and exponentially weighted according to the distance from $\boldsymbol{g}(t)$ with weight $W_i^1 := \exp(-\|\boldsymbol{g}(t) - \boldsymbol{z}(i)\|^2/\sigma_1^2)$.
2. Temporal smoothness: The similarity of the difference vectors $d\boldsymbol{z}(i) := \boldsymbol{z}(i) - \boldsymbol{z}(i-1)$ of the neighbors $\boldsymbol{z}(i)$ in $\mathcal{N}$ and the considered trajectory $d\boldsymbol{g}(t) := \boldsymbol{g}(t) - \boldsymbol{g}(t-1)$ is computed based on the cosine distance and exponentially weighted, yielding weight $W_i^2 := \exp(\alpha(d\boldsymbol{z}(i)^t d(\boldsymbol{g}(t))/(\|d\boldsymbol{z}(i)\| \cdot \|d\boldsymbol{x}(t)\|) - 1))$.
3. Noise perturbation: for every neighbor $\boldsymbol{z}(i)$ in $\mathcal{N}$, noisy successors of $\boldsymbol{g}(t)$ are sampled using the direction of the trajectory at $\boldsymbol{z}(i)$ and a Gaussian noise vector $\boldsymbol{\nu}_j$ with components $\sim N(0, \sigma_2)$ leading to possible positions $\boldsymbol{g}^{ij}(t+1) = \boldsymbol{g}(t) + (\boldsymbol{z}(i+1) - \boldsymbol{z}(i)) + \boldsymbol{\nu}_j$.
4. Drift prevention: Each candidate is weighted according to its distance from the trajectory leading to the weight $p(\boldsymbol{g}^{ij}(t+1)) = W_i^1 W_i^2 \sum_l \varphi((\boldsymbol{g}^{ij}(t+1) - \boldsymbol{z}_k)/h)$ where φ is a window function with window width h.
5. Normalization: These weights are normalized such that $\sum_{ij} p(\boldsymbol{g}^{ij}(t+1)) = 1$.
6. Prediction: The successor is chosen from these points according to the probability $p(\boldsymbol{g}^{ij}(t+1))$.

This way, the overall direction of the trajectory gives rise to the respective successor of a given starting position. Slight noise accounts for typical effects when dealing with natural phenomena. An additional neighborhood integration makes sure that the created trajectory does not diverge from the dynamics as determined by the given data set.

4 Experiments

An important example of dynamic texture is given by image sequences of natural phenomena as available in the DynTex database [14]. Each pixel gradually

Table 1. Number of local models used to map the respective dynamic texture into low dimensional space and number of frames included in the dynamic textures

Dataset	no. local models	no. frames
wave	3	200
escalator	4	251
smoke	5	251
fall	2	200
straw	4	251

changes its intensity or color level depending on the kind of image sequence. Examples of natural phenomena used in this work are referred to as wave, escalator, smoke, straw, and fall. All images have an original size of 288 by 352 pixels with RGB color codes. Because of the high dimensionality, we resized the images to 50% with respect to the original size resulting in 144 times 176 pixels. Then, each image corresponds to a 76,032 dimensional vector formed by the RGB values of the pixels. Before applying matrix NG, data were projected to 100 dimensions using simple principal component analysis.

We compare the result of matrix NG and mixtures of probabilistic component analysis as described in [19] together with global manifold charting and trajectory traversal as described above. The dimensionality n has been chosen as 40. The number k of local components is chosen such that, on average, about 50 frames are represented by one local model. The lengths of the considered dynamic textures and the number of local models is shown in Tab. 1.

We evaluate the method by the mean absolute distance of the generated images and the original images averaged over time, as shown in Tab. 2. Further, we exemplarily show a visual comparison of the images as obtained by MNG and MPPCA in comparison to the original image in Figs. 2 - 5. The synthesis results indicate that manifold embedding based on MNG is able to generate high-quality video, while charting based on MPPCA generates lower visual quality of video over time. The synthesized image sequences by MNG are smooth with respect to temporal evolution due to the included neighborhood cooperation, and sharp features are better preserved in the single images. MPPCA contains blurring in single images and a larger trend towards discontinuities when generating image sequences. This manifests in a larger error of the generated images. The development of the absolute error over time per pixel is depicted in Fig. 1. Obviously, for MPPCA, the error is not uniformly distributed but it accumulates at points in time such that errors are clearly observable for MPPCA. In comparison, the error of MNG is very smooth such that no abrupt changes in the visual appearance can be observed.

Both methods, dynamic tecture generation based on MPPCA or MNG, rely on prototypes which represent parts of the data space. Commonly, these prototypes are computed as averages, such that both methods have the drawback that they somehow smooth details in the images. Since high contrast features are of particular relevance for the human observer, deviations from the original images

Table 2. Average absolute reconstruction errors averaged over the number of frames on the given image sequences and standard deviations

Dataset	MPPCA	MNG
wave	33.7319 ± 2.7425	31.3815 ± 0.5540
escalator	30.8114 ± 2.0043	24.5836 ± 0.1290
smoke	14.2246 ± 0.5515	12.5402 ± 0.2278
fall	48.4483 ± 0.8815	47.3001 ± 0.0537
straw	37.3818 ± 0.5610	36.7924 ± 0.1960

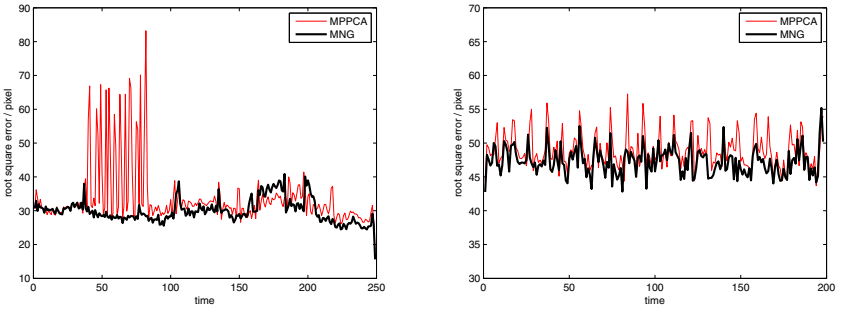

Fig. 1. Absolute error per pixel between the generated images and the true image sequence over time for MPPCA and MNG for the wave texture (left) and fall texture (right). Obviously, the error obtained by MPPCA is large for some time points in which a low quality of the reconstructed texture can visually be observed.

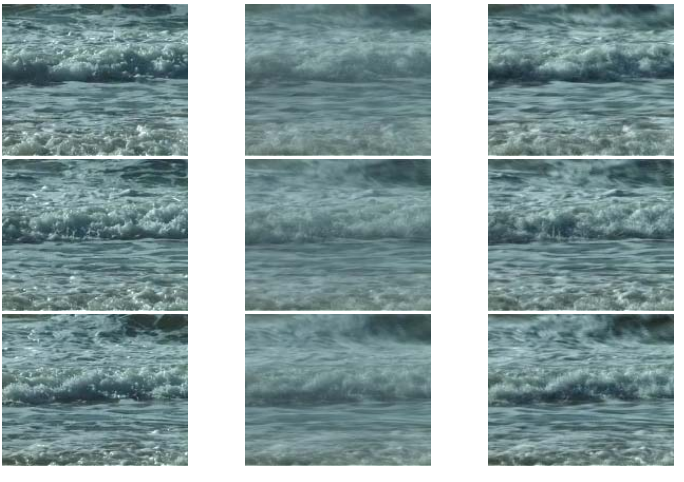

Fig. 2. This figure shows reconstructed image sequence of waves. The first column represents the original reference frames. The second and third columns demonstrate the frames reconstructed by manifold charting based on MPPCA and matrix NG, respectively.

Fig. 3. This figure shows reconstructed image sequence of smoke. The first column represents original reference frames. The second and third columns demonstrate the frames reconstructed by manifold charting based on MPPCA and matrix NG, respectively.

Fig. 4. This figure shows reconstructed image sequence of fall. The first column represents original reference frames. The second and third columns demonstrate the frames reconstructed by manifold charting based on MPPCA and matrix NG, respectively.

can clearly be observed by humans, albeit the error is small, as can be seen for the examples 'fall' and 'straw' which include high contrast or lots of details, respectively. This drawback could be prevented by substituting the averaged prototypes by features with more contrast or details, respectively, obtained e.g. by appropriate postprocessing.

Fig. 5. This figure shows reconstructed image sequence of straw. The first column represents original reference frames. The second and third columns demonstrate the frames reconstructed by manifold charting based on MPPCA and matrix NG, respectively.

5 Discussion

We have introduced a method which allows the global nonlinear embedding of complex manifolds into low dimensions based on matrix neural gas, leading to an explicit embedding function as well as its approximate inverse. Thereby, an essential part is given by matrix neural gas as presented in this paper, which extracts prototypes and local principle directions from the data in a robust and topology preserving way such that a set of smooth local linear maps is obtained. These can be combined using charting techniques such that a global smooth embedding arises. In comparison to alternatives such as mixtures of probabilistic principal components, topology preservation has the beneficial effect that manifold charting deals with smooth maps which can be glued together more easily, and the reconstruction error by means of the approximate inverse shows good agreement to the original manifold also at the borders of the local linear pieces. We have demonstrated that this technique can successfully be used in a complex task from computer graphics, namely the synthesis of dynamic textures based on given image sequences.

References

1. Arnonkijpanich, B., Hammer, B., Hasenfuss, A.: Local matrix adaptation in topographic neural maps, Technical Report IfI-08-07, Department of Computer Science, Clausthal University of Technolgy (September 2008)
2. Arnonkijpanich, B., Hasenfuss, A., Hammer, B.: Local matrix learning in clustering and applications for manifold visualization. In: Neural Networks (to appear, 2010)

3. Brand, M.: Charting a manifold. In: Advances in Neural Information Processing Systems, vol. 15, pp. 961–968. MIT Press, Cambridge (2003)
4. Chuang, Y.Y., Goldman, D.B., Zheng, K.C., Curless, B., Salesin, D.H., Szeliski, R.: Animating pictures with stochastic motion textures. ACM Transactions on Graphics 24(3), 853–860 (2005)
5. Cottrell, M., Hammer, B., Hasenfuss, A., Villmann, T.: Batch and median neural gas. Neural Networks 19, 762–771 (2006)
6. Doretto, G., Chiuso, A., Wu, Y., Soatto, S.: Dynamic Textures. Int. Journal of Computer Vision 51(2), 91–109 (2003)
7. Kohonen, T.: Self-Organizing Maps. Springer, Heidelberg (1995)
8. Liu, C.B., Lin, R.S., Ahuja, N.: Modeling Dynamical Textures Using Subspace Mixtures. In: IEEE International Conference on Multimedia and Expo 2005, pp. 1378–1381 (2005)
9. Liu, C.B., Lin, R.S., Ahuja, N., Yang, M.H.: Dynamic Textures Synthesis as Nonlinear Manifold Learning and Traversing. In: British Machine Vision Conference 2006, vol. 2, pp. 859–868 (2006)
10. van der Maaten, L.J.P., Postma, E.O., van den Herik, H.J.: Matlab Toolbox for Dimensionality Reduction. In: MICC, Maastricht University (2007)
11. Martinetz, T., Berkovich, S.G., Schulten, K.J.: 'Neural-gas' network for vector quantization and its application to time-series prediction. IEEE Transactions on Neural Networks 4, 558–569 (1993)
12. Martinetz, T., Schulten, K.: Topology representing networks. Neural Networks 7, 507–522 (1994)
13. Möller, R., Hoffmann, H.: An Extension of Neural Gas to Local PCA. Neurocomputing 62, 305–326 (2004)
14. R. Peteri, M. Huiskes, and S. Fazekas, The DynTex database Homepage, `http://old-www.cwi.nl/projects/dyntex/database.html`, Centre for Mathematics and Computer Science, Amsterdam
15. Soatto, S., Doretto, G., Wu, Y.N.: Dynamic Textures. In: IEEE International Conference on Computer Vision, vol. 2, pp. 439–446 (2001)
16. Szummer, M., Picard, R.W.: Temporal texture modeling. In: IEEE International Conference on Image Processing, vol. 3, pp. 823–826 (1969)
17. The, Y.W., Roweis, S.: Automatic alignment of local representations. Advances in Neural Information Processing Systems 15, 841–848 (2003)
18. Treuille, A., McNamara, A., Popovic, Z., Stam, J.: Keyframe control of smoke simulations. ACM Transactions on Graphics 22(3), 716–723 (2003)
19. Tipping, M.E., Bishop, C.M.: Mixtures of probabilistic principal component analyzers. Neural Computation 11, 443–482 (1999)
20. Wei, L.-Y., Levoy, M.: Fast texture synthesis using tree-structured vector quantization. In: Proceedings of ACM SIGGRAPH 2000, pp. 479–488 (2000)