Prediction and Classification of Motion Trajectories Using Spatio-Temporal NMF

Julian Eggert, Sven Hellbach, Alexander Kolarow, Edgar Körner, Horst-Michael Groß

2009

Preprint:

This is an accepted article published in {KI} 2009: Advances in Artificial Intelligence, 32nd Annual German Conference on {AI}, Paderborn, Germany, September 15-18, 2009. Proceedings. The final authenticated version is available online at: https://doi.org/[DOI not available]

Prediction and Classification of Motion Trajectories using Spatio-Temporal NMF

Sven Hellbach[1], Alexander Kolarow[1], Julian P. Eggert[2], Edgar Körner[2], and Horst-Michael Gross[1]

[1] Ilmenau University of Technology, Neuroinformatics and Cognitive Robotics Labs, POB 10 05 65, 98684 Ilmenau, Germany
sven.hellbach@tu-ilmenau.de
[2] Honda Research Institute Europe GmbH, Carl-Legien-Strasse 30, 63073 Offenbach/Main, Germany
julian.eggert@honda-ri.de

Abstract. This paper's intention is to present a new approach for decomposing motion trajectories. The proposed algorithm is based on non-negative matrix factorization, which is applied to a grid like representation of the trajectories. From a set of training samples a number of basis primitives is generated. These basis primitives are applied to reconstruct an observed trajectory. The reconstruction information can be used afterwards for classification. An extension of the reconstruction approach furthermore enables to predict the observed movement into the future. The proposed algorithm goes beyond the standard methods for tracking, since it does not use an explicit motion model but is able to adapt to the observed situation. In experiments we used real movement data to evaluate several aspects of the proposed approach.

Key words: Non-negative Matrix Factorization, Prediction, Movement Data, Robot, Motion Trajectories

1 Introduction

The understanding and interpretation of movement trajectories is a crucial component in dynamic visual scenes with multiple moving items. Nevertheless, this problem has been approached very sparsely by the research community. Most approaches for describing motion patterns, like [1], rely on a kinematic model for the observed human motion. This causes the drawback that the approaches are difficult to adapt to other objects. Here, we aim at a generic, model-independent framework for decomposition, classification and prediction.

Consider the simple task for a robot of grasping an object which is handed over by the human interaction partner. First of all, the grasping task has to be recognized by the robot. We assume that this information can be gained from the motion information. Furthermore, to avoid a purely reactive behavior, which might lead to 'mechanical' movements of the robots, it is necessary to predict the further movement of the human's hand.

In [2] an interesting concept for a decomposition task is presented. Like playing a piano a basis alphabet – the different notes – are superimposed to reconstruct the observation (the piece of music). The much less dimensional description of when each basis primitive is used, can be exploited for further processing. While the so-called piano model relies on a set of given basis primitives, our approach is able to learn these primitives from the training data.

Beside the standard source separation approaches, like PCA and ICA, another promising algorithm exists. It is called non-negative matrix factorization (NMF) [3]. The system of basis vectors which is generated by the NMF is not orthogonal. This is very useful for motion trajectories, since one basis primitive is allowed to share a common part of its trajectory with other primitives and to specialize later.

The next section introduces the standard non-negative matrix factorization approach and two extensions that can be found in the literature. In section 3 the new approach for decomposing motion trajectories is presented. The experiments with their conditions and results are presented in section 4, while the paper concludes in section 5.

2 Non-negative Matrix Factorization

Like other approaches, e. g. PCA and ICA, non-negative matrix factorization (NMF) [3] is meant to solve the source separation problem. Hence, a set of training data is decomposed into basis primitives: $\mathbf{V} \approx \mathbf{W} \cdot \mathbf{H}$ Each training data sample is represented as a column vector $\mathbf{V}_i$ within the matrix $\mathbf{V}$. Each column of the matrix $\mathbf{W}$ stands for one of the basis primitives. In matrix $\mathbf{H}$ the element H_i^j determines how the basis primitive $\mathbf{W}_j$ is activated to reconstruct training sample $\mathbf{V}_i$. Since NMF is an iterative method, the training data $\mathbf{V}$ can only be approximated by the product of $\mathbf{W}$ and $\mathbf{H}$. This product will be referred to as reconstruction $\mathbf{R} = \mathbf{W} \cdot \mathbf{H}$ later.

Unlike PCA or ICA, NMF aims to a decomposition, which only consists of non-negative elements. This means that the basis primitives can only be accumulated. No primitive exists which is able to erase a 'wrong' superposition of other primitives. This leads to a more specific set of basis primitives, which is an advantage for certain applications, e. g. face recognition [4].

For generating the decomposition, optimization-based methods are used. Hence, an energy function E has to be defined:

$$E(\mathbf{W}, \mathbf{H}) = \frac{1}{2} \left\| \mathbf{V} - \mathbf{W} \cdot \mathbf{H} \right\|^2 \tag{1}$$

By minimizing the energy equation, it is now possible to achieve a reconstruction using the matrices $\mathbf{W}$ and $\mathbf{H}$. This reconstruction is aimed to be as close as possible to the training data $\mathbf{V}$. No further constraints are given in the standard formulation of the NMF. As it can be seen in equation 1, the energy function depends on the two unknown matrices $\mathbf{W}$ and $\mathbf{H}$.

Since both matrices usually have a large number of elements, the optimization problem seems to be an extensive task. Fortunately, each training sample can be

regarded independent from the others. Furthermore, both matrices are adapted in an alternating fashion. This helps to reduce the number of dimensions for the optimization process and allows a training regime with fewer examples. The algorithm is formulated in the following description in vectorwise notation:

1. Calculate the reconstruction

$$\mathbf{R}_i = \sum_j H_i^j \mathbf{W}_j \tag{2}$$

2. Update the activities

$$H_i^j \leftarrow H_i^j \odot \left(\mathbf{V}_i^T \mathbf{W}_j \oslash \mathbf{R}_i^T \mathbf{W}_j\right) \tag{3}$$

3. Calculate the reconstruction with the new activities

$$\mathbf{R}_i = \sum_j H_i^j \mathbf{W}_j \tag{4}$$

4. Update the basis vectors

$$\mathbf{W}_j \leftarrow \mathbf{W}_j \odot \left(\sum_i H_i^j \mathbf{V}_i \oslash \sum_i H_i^j \mathbf{R}_i\right) \tag{5}$$

Where the operations $\odot$ and $\oslash$ denotes a component-wise multiplication and division. Steps 1 to 4 are iterated until a defined convergence criterion is reached. For the criterion the energy function can be used in a usual fashion.

2.1 Sparse Coding

As it could be seen in equation 1 the energy function is formulated in a very simple way. This results in a decomposition, which is quite arbitrary with no further characteristics. This can lead, for example, to redundant information. Especially, if the number of basis primitives is chosen higher than needed to decompose the given training data. To compensate this drawback, it is useful to introduce a constraint which demands a sparse activation matrix, like it was introduced in [5]. This avoids the fact, that several basis primitives are activated at the same time, and hence are being superimposed.

$$E(\mathbf{W}, \mathbf{H}) = \frac{1}{2} \left\|\mathbf{V} - \mathbf{W} \cdot \mathbf{H}\right\|^2 + \lambda \sum_{i,j} H_i^j \tag{6}$$

The influence of the sparsity constraint can be controlled using the parameter λ. In this paper, we only discuss a special case for the sparsity term. A more detailed discussion can be found in [5]. The algorithmic description is similar to the one of the standard NMF. The only thing that has to be considered is that the basis primitives need to be normalized.

2.2 Transformation Invariance

Beside the sparsity constraint another extension to NMF has been published in [6]. The concept of transformation invariance allows to move, rotate, and scale

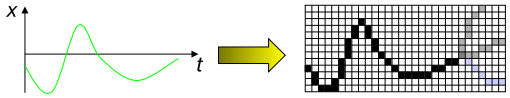

Fig. 1. Motion Trajectories are transferred into a grid representation. A grid cell is set to 1 if it is in the path of the trajectory and set to zero otherwise. During the prediction phase multiple hypotheses can be gained by superimposing several basis primitives. This is indicated with the gray trajectories on the right side of the grid.

the basis primitives for reconstructing the input. In this way, we do not have to handle each possible transformation using separate basis vectors. This is achieved by adding a transformation $\mathbf{T}$ to the decomposition formulation: $\mathbf{V} \approx \mathbf{T} \cdot \mathbf{W} \cdot \mathbf{H}$

However the activation matrix $\mathbf{H}$ has to be adapted in a way that each possible transformation carries its own activation. Hence, the matrix $\mathbf{H}$ becomes an activation tensor $\mathbf{H^m}$, while $\mathbf{m}$ is a vector describing the transformation parameters (rotation, scaling and translation).

$$\mathbf{V_i} \approx \sum_j \sum_{\mathbf{m}} H_i^{j,\mathbf{m}} \cdot \mathbf{T^m} \cdot \mathbf{W_j} \tag{7}$$

For each allowed transformation the corresponding activity has to be trained individually.

3 Decomposing Motion Trajectories

For being able to decompose and to predict the trajectories of the surrounding dynamic objects, it is necessary to identify them and to follow their movements. For simplification, a tracker is assumed, which is able to provide such trajectories in real-time. A possible tracker to be used is presented in [7]. The given trajectory of the motion is now interpreted as a time series $\mathcal{T}$ with values $\boldsymbol{s}_i = (x_i, y_i, z_i)$ for time steps $i = 0, 1, \ldots, n-1$: $\mathcal{T} = (\mathbf{s}_0, \mathbf{s}_1, \ldots, \mathbf{s}_{n-1})$.

It is now possible to present the vector $\mathcal{T}$ directly to the NMF approach. But this could result in an unwanted behavior. Imagine two basis primitives, one representing a left turn and another representing a right turn. A superposition of those basis primitives would result in a straight movement.

The goal is to have a set of basis primitives, which can be concatenated one after the other. Furthermore, it is necessary for a prediction task to be able to formulate multiple hypotheses. For achieving these goals, the x-t-trajectory is transferred into a grid representation, as it is shown in figure 1. Then, each grid cell (x_i, t_j) represents a certain state (spatial coordinate) x_i at a certain time t_j. Since most of the state-of-the-art navigation techniques rely on grid maps [8], the prediction can be integrated easily. This 2D-grid is now presented as image-like input to the NMF algorithm using the sparsity constraint as well as transformation invariance (See section 2.1 and 2.2 respectively). Using the grid representation of the trajectory also supports the non-negative character of the basis components and their activities.

It has to be mentioned, that the transformation to the grid representation is done for each of the dimensions individually. Hence, the spatio-temporal NMF

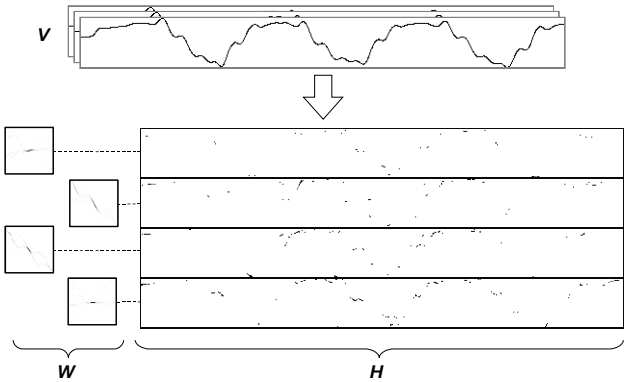

Fig. 2. Training with Spatio-Temporal NMF. Given is a set of training samples in matrix **V**. The described algorithm computes the weights **W** and the corresponding activities **H**. Only the weights are used as basis primitives for further processing.

has to be processed on each of these grids. Regarding each of the dimensions separately is often used to reduce the complexity of the analysis of trajectories (compare [9]). Theoretically, the algorithm could also handle multi-dimensional grid representation.

While applying an algorithm for basis decomposition to motion trajectories it seems to be clear that the motion primitives can undergo certain transformations to be combined to the whole trajectory. For example, the same basis primitive standing for a straight move can be concatenated with an other one standing for a left turn. Hence, the turning left primitive has to be moved to the end of the straight line, and transformation invariance is needed while decomposing motion data. For our purposes, we concentrate on translation. This makes it possible to simplify the calculations and to achieve real time performance.

It is known for each basis primitive,how they have to be combined to reconstruct the input data. Hence, one of the algorithm's outputs is a description of the sequence of the basis primitives which can be used as input for a classifier.

The sparse coding constraint helps to avoid trivial solutions. Since the input can be compared with a binary image, one possible solution would be a basis component with only a single grid cell filled. The trajectory is then simply copied into the activities.

3.1 Learning Phase

The goal of the learning phase is to gain a set of basis primitives which allow to decompose an observed and yet unknown trajectory (see Fig. 2). As it is discussed in section 3, the training samples are transferred into a grid representation. These grid representations are taken as input for the NMF approach and are therefore represented in matrix **V**. On this matrix **V** the standard NMF approach, extended by the sparsity constraint and by translation invariance, is applied. The algorithm is summarized in Fig. 4.

Beside the computed basis primitives, the NMF algorithm also provides the information of how each of the training samples can be decomposed by the basis

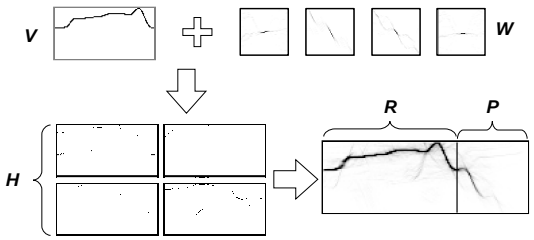

Fig. 3. The basis primitives **W**, which were computed during the training, are used to reconstruct (matrix **R**) the observed trajectory **V**. This results in a set of sparse activities – one for each basis primitive – which describe on which position in space and time a certain primitive is used. Beside the reconstruction of the observed trajectory (shown in Fig. 3), it is furthermore possible to predict a number of time steps into the future. Hence, the matrix **R** is extended by the prediction horizon **P**.

primitives. To be able classify the trajectory later, this information is used for training the classifier. Since trajectories and hence the calculated activities are temporal sequences, a recurrent neural network seems to be an adequate classifier. The network is trained by presenting a discrete time step of the activities (i. e. a single column in the matrix)

3.2 Application Phase

As it is indicated in Fig. 3, from the learning phase a set of motion primitives is extracted. During the application phase, we assume that the motion of a dynamic object (e. g. a person) is tracked continuously. For getting the input for the NMF algorithm, a sliding window approach is taken. A certain frame in time is transferred into the already discussed grid like representation. For this grid the activation of the basis primitives is determined. The algorithm is identical to the one sketched in Fig. 4 beside that steps 4 and 5 can be skipped.

The resulting activities are now used as input for the classifier, which was adapted to the activities in the learning phase. In this way, a classification result (e. g. the interest for interaction or the performed action) can be assigned to the regarded time window.

The standard approach to NMF implies that each new observation at the next time step demands a new random initialization for the optimization problem. Since an increasing column number in the grid representation stands for an increase in time, the trajectory is shifted to the left while moving further in time. For identical initialization, the same shift is then reflected in the activities after the next convergence. To reduce the number of iterations until convergence, the shifted activities from the previous time step are used as initialization for the current one.

To fulfill the main goal discussed in this paper – the prediction of the observed trajectory into the future – the proposed algorithm had to be extended. Since the algorithm contains the transformation invariance constraint, the computed basis primitives can be translated to an arbitrary position on the grid. This means that they can also be moved in a way that they exceed the borders of the

1. Normalize the basis vectors according to

$$\overline{\mathbf{W}}_j = \mathbf{W}_j \cdot \|\mathbf{W}_j\|^{-1} \tag{8}$$

2. Calculate the reconstruction

$$\mathbf{R}_i = \sum_j \sum_{\mathbf{m}} H_i^{j,\mathbf{m}} \mathbf{T}^{\mathbf{m}} \overline{\mathbf{W}}_j \tag{9}$$

3. Update the activities

$$H_i^{j,\mathbf{m}} \leftarrow H_i^{j,\mathbf{m}} \odot \left(\mathbf{V}_i^T \mathbf{T}^{\mathbf{m}} \overline{\mathbf{W}}_j \oslash \mathbf{R}_i^T \mathbf{T}^{\mathbf{m}} \overline{\mathbf{W}}_j \right) \tag{10}$$

4. Calculate the reconstruction with the new activities

$$\mathbf{R}_i = \sum_j \sum_{\mathbf{m}} H_i^{j,\mathbf{m}} \mathbf{T}^{\mathbf{m}} \overline{\mathbf{W}}_j \tag{11}$$

5. Update the basis vectors

$$\mathbf{W}_j \leftarrow \mathbf{W}_j \odot \frac{\sum_i \sum_{\mathbf{m}} H_i^{j,\mathbf{m}} \mathbf{V}_i^T \mathbf{T}^{\mathbf{m}} + \overline{\mathbf{W}}_j \overline{\mathbf{W}}_j^T \sum_i \sum_{\mathbf{m}} H_i^{j,\mathbf{m}} \mathbf{R}_i^T \mathbf{T}^{\mathbf{m}}}{\sum_i \sum_{\mathbf{m}} H_i^{j,\mathbf{m}} \mathbf{R}_i^T \mathbf{T}^{\mathbf{m}} + \overline{\mathbf{W}}_j \overline{\mathbf{W}}_j^T \sum_i \sum_{\mathbf{m}} H_i^{j,\mathbf{m}} \mathbf{V}_i^T \mathbf{T}^{\mathbf{m}}} \tag{12}$$

Fig. 4. Algorithmic description of the Spatio-temporal NMF.

grid. Up to now, the size of reconstruction was chosen to be the same size as the input grid. Hence, using the standard approach means that the overlapping information has to be clipped. To be able to solve the prediction task, we simply extend the reconstruction grid to the right – or into the future (see Fig. 3). So, the previously clipped information is available for prediction.

As discussed, a classifier is trained during the learning phase using the activities of the training samples. In the application phase the activities for the current observation is calculated. The last activity column is used as input for the classifier. So a classification result is gained for each time step on-line.

4 Evaluation

Taking a closer look at the example scenario from introductory section 1 reveals that a robust identification and tracking of the single body parts is needed. To be comparable and to avoid errors from the tracking system influencing the test results, movement data from the Perception Action Cognition Lab at the University of Glasgow [10] is used. The data contains trajectories from 30 persons is recorded performing different actions in different moods. The movement data has a resolution of 60 time steps per second, so that an average prediction of about 50 steps means a prediction of 0.83 seconds into the future. Since most trackers work with a lower resolution, a prediction further into the future is still possible.

In the next subsections, two aspects of the proposed algorithm are investigated in detail. First, it is shown that activity shifting brings a great benefit towards real time performance. Afterwards the focus is set to the quality of the prediction part.

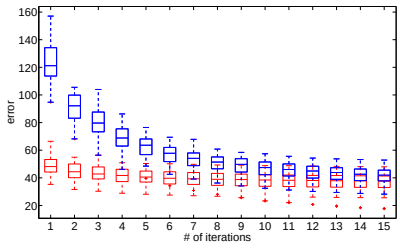

Fig. 5. Box whiskers plot showing the convergence characteristics of the energy function (see eqn. 1) for 15 iteration steps. For the upper (blue) plot the activities are initialized randomly after each shift of the input data. For the lower (red) curve the activities from the previous computations are shifted and used as initialization.

For the experiments, the size of the basis primitives was chosen to be 50×50 grid cells The input grid size during the learning phase was set to 500×50 for each of the trajectories and to 100×50 during application phase.

4.1 Activity shifting

In section 3.2 it has been mentioned that the information from the previous time step can be used as initialization for the current one. Figure 5 shows the energy function, which is defined in equation 1, for both possibilities of initialization. It is plotted only for a low number of iteration steps (up to 15), since already there the effect can be observed. For the upper (blue) plot a random initialization of the activities was used. For the lower (red) curve the activities from the previous computations are shifted and used as initialization. It can clearly be seen that the convergence is faster by a number of about 10 steps in average.

4.2 Prediction

For evaluating the quality of the prediction, the prediction is compared with the grid representation of the actual trajectory $\mathbf{G}$. For each occupied grid cell the value of the columnwise normalized prediction is added. The sum is divided by the length of the trajectory:

$$S_{GT} = |T|^{-1} \sum_{t \in T} \mathbf{P}_t^T \cdot \mathbf{G}_t \cdot \left(\sum_i \mathbf{G}_t^i \right)^{-1} \tag{13}$$

The normalization of the prediction is done separately for each time slice (column in the grid).

The basis primitives can at most be shifted by their width out of the reconstruction grid $\mathbf{R}$. So the maximal size of the prediction horizon equals the width of the basis primitives. Practically this maximum can not be reached, because the basis primitives need a reliable basis in the part where the input is known. Nevertheless, we have chosen to use the theoretical maximum as basis for the evaluation.

The results are depicted in Fig 6. The first plot (Fig. 6(a)) shows the expected decrease of the average prediction quality over the prediction horizon. Nevertheless, the decrease is smooth and no sudden collapses can be observed. For Fig.

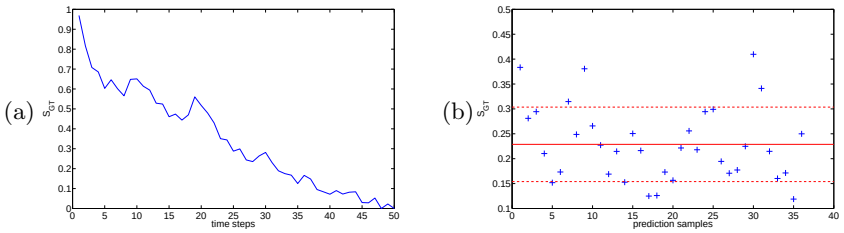

Fig. 6. (a) The mean correlation S_{GT} (see eqn. 13) between the ground truth trajectory and the prediction is plotted for each time step of the prediction horizon. A fit value of 1.0 stands for a perfect prediction over the whole prediction horizon. As it is expected the accuracy of the prediction decreases for a longer prediction period. (b) The plot shows the prediction accuracy for predictions along a sample trajectory. The 36 predictions were performed at each tenth time step of the chosen trajectory. A fit value of 1.0 stands for a perfect prediction over the whole prediction horizon. The constant and dotted lines (red) indicate mean and variance respectively.

6(b) an example trajectory has been selected for the reasons of clearness. The plot is intended to show how the algorithm behaves in practical applications. The predictions were performed at each tenth time step of the chosen trajectory. A fit value of 1.0 stands for a perfect prediction over the maximum prediction horizon, with only a single hypothesis for the prediction. The value decreases significantly with multiple hypotheses being present.

4.3 Classification Task

The goal of the classifier is to classify the current action performed by the proband (throw, walk, knock and lift). The input size is reduced by only using only a single limb (the right wrist). Using half of the data set the basic primitives were trained (twelve for each spatial dimension). As classifier we use a Layer-Recurrent Neural Network in MATLAB, which is a multi-layer version of the well known Elman network. The network was initialised with an input-size of 1800 neurons, 20 recurrent layers, sigmoid transfer function, and an output-size of 4 neurons (binary coded output, one neuron for each class). The classifier was trained using standard back-propagation with MSE. In order to evaluate the classifier during training, the error on a test dataset is computed. Additionally the performance was measured on a validation dataset after training. During the first experiments the classifier scored 40% accuracy on the validation dataset. The poor performance of the classifier may result from the still very high input dimensionality. The classifier is not able to profit from the sparse activation of the basic primitives due to the grid representation. Additionally the grid representation of the sparse input vector leads to many zeros in the input vector which makes a gradient based learning method difficult. In further experiments we will exploit the sparse input representation for the classifier using the position of the maximum activation. The compacter encoding of the input should result into better performance of the classifier.

5 Conclusion and Outlook

This paper presented a new approach for decomposing motion trajectories using non-negative matrix factorization. To solve this problem, sparsity constraint and transformation invariance have been combined. The trajectories were then decomposed using a grid-based representation. It could be demonstrated that the concept of activity shifting clearly decreases the number of iterations needed until convergence. Furthermore it was shown that the proposed algorithm is able to predict the motion into the future. The prediction occurs by a superposition of possible trajectory alternatives, yielding a quasi-probabilistic description. At this point, the information about the sparse activation of the basis primitives was used only for reconstruction purposes, even though it contains significant information about the global motion. The classification task is still insufficent and should be further improved as discussed. Furthermore, different kinds of networks need to be investigated. Nevertheless it could be shown exemplarily, how the suggested motion trajectory representation can be used to solve a classification task.

References

1. Hoffman, H., Schaal, S.: A computational model of human trajectory planning based on convergent flow fields. In: Meeting of the Soc. of Neuroscience. (2007)
2. Cemgil, A., Kappen, B., Barber, D.: A generative model for music transcription. IEEE Transactions on Speech and Audio Processing **14** (2006) 679–694
3. Lee, D.D., Seung, H.S.: Algorithms for non-negative matrix factorization. Advances in Neural Information Processing **13** (2001) 556–562
4. Rajapakse, M., Wyse, L.: NMF vs ICA for face recognition. In: ISPA 2003. Volume 2. (2003) 605–610
5. Eggert, J., Körner, E.: Sparse Coding and NMF. In: IJCNN. (2004) 2529 – 2533
6. Eggert, J., Wersing, H., Körner, E.: Transformation-invariant representation and NMF. In: IJCNN. (2004) 2535 – 2539
7. Otero, N., Knoop, S., Nehaniv, C., Syrdal, D., Dautenhahn, K., Dillmann, R.: Distribution and Recognition of Gestures in Human-Robot Interaction. ROMAN 2006. (Sept. 2006) 103–110
8. Elfes, A.: Using Occupancy Grids for Mobile Robot Perception and Navigation. Computer **12**(6) (June 1989) 46–57
9. Naftel, A., Khalid, S.: Classifying spatiotemporal object trajectories using unsupervised learning in the coefficient feature space. MM Syst. **12**(3) (2006) 227–238
10. http://paco.psy.gla.ac.uk/data_ptd.php