in: Artificial Neural Networks - ICANN 2008, Prague, LNCS 5163, Part I, pp. 710-719, Springer Verlag 2008

Echo State Networks for Online Prediction of Movement Data – Comparing Investigations

Sven Hellbach[1], Sören Strauss[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 adapt Echo State Networks to problems being faced in the field of Human-Robot Interactions. The idea is to predict movement data of persons moving in the local surroundings by understanding it as time series. The prediction is done using a black box model, which means that no further information is used than the past of the trajectory itself. This means the suggested approaches are able to adapt to different situations. For experiments, real movement data as well as synthetical trajectories (sine and Lorenz-attractor) are used. Echo State Networks are compared to other state-of-the-art time series analysis algorithms, such as Local Modeling, Cluster Weighted Modeling, Echo State Networks, and Autoregressive Models. Since mobile robots highly depend on real-time application.

Key words: Echo State Networks, Time Series Analysis, Prediction, Movement Data, Robot, Motion Trajectories

1 Introduction

For autonomous robots, like SCITOS [1], it is important to predict their own movement as well as the motion of people and other robots in their local environment, for example to avoid collisions or to evolve a proactive behavior in Human-Robot-Interaction (HRI). Hence, further actions can be planned more efficiently.

Most publications in this field focus on optimal navigation strategies [2, 3]. This paper, however, suggests to spend more effort into prediction of the motion of the dynamic objects instead. Often, only linear approximations or linear combinations are used to solve this problem.

The approach presented here is the interpretation of movement trajectories as time series and their prediction into the future. For performing this prediction an assortment of time series analysis algorithms was implemented and tested.

Echo State Networks have proven to be able to predict chaotic time series with a comparatively high accuracy [4]. Because of the fact that unexpected

behavior can occur, movement data can also be understood as chaotic time series. Echo State Networks should be able to predict such data well. Their usage for the prediction of movement data has – to our best knowledge – not been investigated yet.

For an application of Echo State Networks, the movement trajectory of a person in the vicinity of the mobile robot, typically gained from a person tracker like [5] needs to be presented as a time series. For this reason, the given trajectory of the motion is now interpreted as $\mathcal{T}$ with values $\mathbf{s}_i$ for time steps $i = 0, \ldots, n-1$:

$$\mathcal{T} = (\mathbf{s}_0, \mathbf{s}_1, \ldots, \mathbf{s}_{n-1}) \tag{1}$$

Each $\mathbf{s}_i$ can be assumed as the tracked object's position, e. g. in a three dimensional Cartesian state space $\mathbf{s}_i = (x_i, y_i, z_i)^T$. Basically, the prediction for each future point on the trajectory is done iteratively for up to 500 time steps (about 8.3 seconds of motion using a sampling rate of 60 Hz).

The prediction in general takes place with the so-called black box model which means that no further background information is used than the past trajectory itself. The aspired prediction shall follow the trajectory's characteristics, which can be found in their past. Furthermore, no explicit model is given, to be able to freely adapt to new types of trajectories, i. e. new situations.

The following section at first briefly explains the principles of Echo State Networks and introduces different versions. In section 3, experiments and comparisons of the results on the movement data are presented, while the last section concludes this paper.

2 Echo State Networks

It is commonly known, that Neural networks are well suited for function approximation tasks. For the specific task of predicting time series, Echo State Networks (ESNs) are often used recently[4].

2.1 Principle

ESNs have some specific features which differ from "standard" neural networks: The hidden layer consists of neurons which are randomly connected (see Fig. 1). If the connectivity is low, this layer provides independent output trajectories. For this reason, the hidden layer is also called *reservoir*. Furthermore, there are neurons which are connected to circles in the reservoir, so that past states "echo" in the reservoir. That is the reason why only the current time series value $\mathbf{s}_n$ is needed as input.

The weights of the reservoir determine the matrix $\mathbf{W}^r$. In [4], it is mentioned that the spectral radius *spec* of this matrix[3] is an important parameter and must not have values above 1.0 to guarantee stable networks. The randomly initialized reservoir matrix $\mathbf{W}^r$ can easily be adapted to a matrix with a desired spectral

[3] The spectral radius of a matrix equals to the largest Eigenvalue.

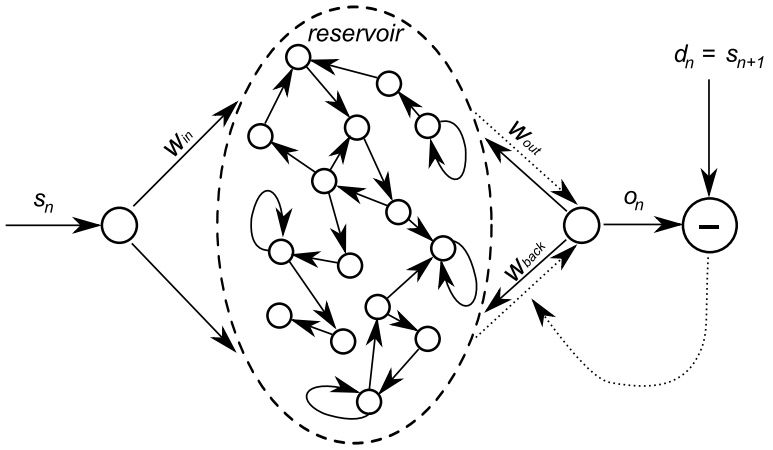


Fig. 1. The design of Echo State Networks has some characteristic features. In addition to the randomly connected reservoir r_n, the training algorithm is pretty simple for neural networks: Only the output weights $\mathbf{w}_{out}$ are adapted.

radius. However, [6] argued that networks with a spectral radius close or slightly above 1.0 may lead to better results. Both possibilities are evaluated for their suitability for motion prediction.

Furthermore, the sparseness of the reservoir matrix plays an important role. A sparse reservoir matrix means that most of the weights in the reservoir are set to zero. This can be interpreted as the reservoir being decomposed into subsets, which are responsible for basic signals being overlaid by the output layer. As suggested in [4] and [6] both sources, about 80% of the weights are set to zero.

Another characteristic of ESNs is that only the output weights $\mathbf{w}_out$ are adapted and learned. All other weights (input, reservoir, feedback) are chosen randomly and stay static.

2.2 Training and Application

For training, the network is initialized randomly, and the training time series is used as network input step by step. The internal states $\mathbf{r}_n$ are calculated by using the following recursive equation:

$$\mathbf{r}_n = f(\mathbf{W}^r \cdot \mathbf{r}_{n-1} + \mathbf{w}_{in} \cdot \mathbf{s}_n + \mathbf{w}_{back} \cdot \mathbf{o}_{n-1}) \tag{2}$$

$\mathbf{r}_n$ describes the internal state at time step n. $\mathbf{W}^r$ stand for the reservoir matrix, while $\mathbf{w}_{in}$ and $\mathbf{w}_{back}$ are the weights at the respective edges (See Fig. 1), while f is the transfer function of the reservoir neurons which can be the Fermi-function or the hyperbolic tangent.

From a predefined starting point, the internal states $\mathbf{r}_n$ can be combined to a matrix $\mathbf{R}$. The starting point should be around the time step 100 or later to overcome possible bad initial values in the network. The adaption step for the

output weights $\mathbf{w}_{out}$ is a linear regression using this matrix and the vector of the related output values $\mathbf{o}$:

$$\mathbf{w}_{out} = (\mathbf{R}^T\mathbf{R})^{-1}\mathbf{R}^T\mathbf{o}. \tag{3}$$

After weight adaption, the network can be applied for prediction. Thereto, the network is fed again with the whole trajectory data as input, this time step by step. If the prediction is taking place (i. e. reaching the last known point in time) the output is fed back to the input. So, the last network output is used as the next input to be able to generate more than one prediction step. In our experiments, up to 800 prediction steps are generated.

2.3 Enhancements

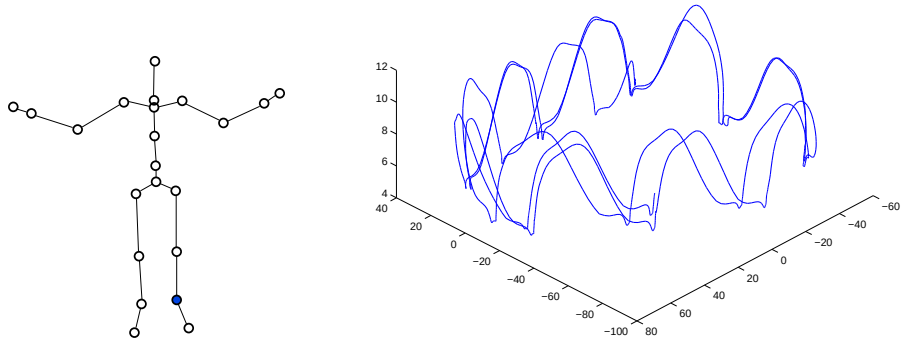
In [6] some additional Echo State Network features are introduced, like an online adapting rule and a plasticity rule to adapt the Fermi transfer function parameters in the reservoir (*intrinsic plasticity*). Furthermore, additional weights such as a direct input-output ($\mathbf{w}_{dir}$) weight and a loop at the output neuron ($\mathbf{w}_{rec}$) are suggested. Apart from the online rule, all other of those enhancements were evaluated and tested.

Intrinsic plasticity is performed online. It helps to adjust the reservoir transfer functions for better adapting to the current prediction task. It takes place before starting the learning of the output weights and shouldn't last longer than 200 time steps, otherwise predictions could get instable. Unfortunately, intrinsic plasticity has the effect that the eigenvalues and thus the spectral radius of the reservoir matrix increases.

Since in Echo State Network a huge number of parameters can be adjusted, a more automated process would be reasonable, especially, for those network weights, which are not changed during the regular training process, i. e. the $\mathbf{w}_{in}$, $\mathbf{w}_{back}$, and W^r. This paper suggests to use multiple instances of the network, as a kind of simple stochastical search in the parameter space. All instances are trained using the same input data, after initializing the fixed weights diferrently (in a random manner). During the training process, the output of each network is compared with the corresponding values of the training trajectory. The network showing the best prediction results for the yet unknown training data is then selected for further application.

3 Motion Prediction

The algorithms presented in this paper are intended to be used for motion prediction to enable a mobile robot evolving a proactive behavior in HRI. To be comparable and reproduceable, however, movement data taken from the University of Glasgow is used [7]. This benchmark data is available as 3D coordinate representation for each limb of a human performing a certain action, e. g. walking (see Fig. 2). Using this data is even more challenging, because several basic



(a) (b)

Fig. 2. Example of movement data from the University of Glasgow. Shown are the body points from which data is available (a) and an exemplary trajectory of the movement of the left ankle while walking in circles (b).

motions are combined (i. e. intrinsic movement, e. g. of the foot combined with the walking direction). The data set consists of 25 trajectories containing 1,500 up to 2,500 sampled points.

3.1 Test Conditions

Trajectories Besides the movement data coming from the University of Glasgow, periodical and "standard" chaotic time series are used. All time series are three-dimensional.

The movement data has a resolution of 60 time steps per second, so that an average prediction of about 500 steps means a prediction of 8.3 seconds into the future. Present movement prediction techniques last considerably shorter.

The periodical trajectories consists of up to three superimposed sine waves, with each dimension being independent from the others.

As chaotic time series the Lorenz-Attractor is used. It is a simple system of differential equations where the single dimensions are not independent. This time series is a typically chaotic one, so small changes in a state leads to dramatic differences after a short time period.

Quality Measures For comparing the prediction results, some kind of quality measures are necessary. The used quality measures are based on the normalized mean square error $NMSE$. Hence, the standard mean square error is normalized using the variance σ^2 of the time series.

$$NMSE = \frac{1}{N \cdot \sigma^2} \sum_{i=1}^{N} (\mathbf{s}_i^{pred} - \mathbf{s}_i^{orig})^2 = \frac{MSE}{\sigma^2} \tag{4}$$

Since the trajectories are three-dimensional and dimensions with greater difference are supposed to be more important, the highest variance of all dimensions is used as normalization.

Two different kinds of the defined measure are used. The first one, the short term error STE, is responsible for evaluating a short period of the prediction. It uses the first $N = 75$ predicted output values (which means 1.25 sec) with a weighting of $\frac{1}{f}$ of the f-th prediction step. Since some of the algorithms show the tendency to drift away, the performance is furthermore evaluated using the long term error LTE, which uses all prediction steps with a weighting of $\frac{1}{\sqrt{f}}$.

3.2 Reference Algorithms

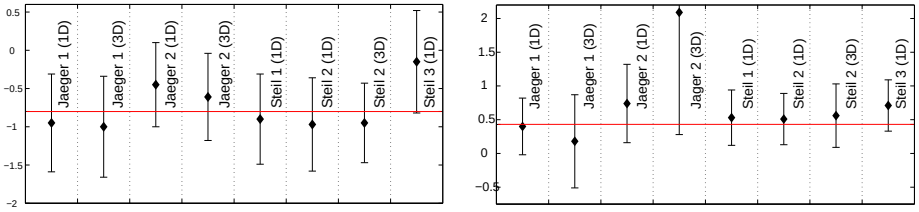
Time Series Analysis Algorithms Echo State Networks are compared to other state-of-the-art time series analysis algorithms, to be able to assess not only their absolute performance. They have been reimplemented in MatLab, following the methods described in the respective papers. Again, the black box model is used to have a similar starting point for all approaches.

1. **Autoregressive Models** assume a linear relation in the observed time series which means that any time series value can be determined by using a linear combination of p previous values. Different approaches can be used to determine these linear coefficients. The Wiener Filter, the Durbin-Levinson algorithm and the Yule-Walker equations were used to calculate the coefficients. For further details see [8] and [9].
2. **Local Modeling**: This algorithm tries to find similar states of the observed trajectory. Therefore, the usually low dimensional time series is transformed in a higher dimensional space, the so-called embedding space. Details from this algorithm can be found in [10] and [11].
3. **Cluster Weighted Modeling**: This approach is similar to the Local Modeling but from a probabilistic point of view. Hence, the embedding space is clustered with Gaussians. For more details see [10] and [11], too.

Trivial Comparison Algorithms The first algorithm, defining the baseline, is a simple repetition of the last observed time series value and is called repetetive algorithm in the following. Also a linear algorithm is used as reference. This algorithm simply does a linear approximation based on the last two points of the time series. The result of the better one is used as reference. Both algorithms have to be outperformed clearly to get useful predictions.

3.3 Test Results

The following tests are to demonstrate the advantages and disadvantages of the Echo State variants and the other time series analysis algorithms presented within this paper. For the application of the algorithms, a lot of parameters had to be specified. The paramter values presented in the following are chosen after extensive tests, which cannot be not discussed here, because of space limitations.



(a) (b)

Fig. 3. The graphs show the STE (a) and LTE (b) plotted for each of the Echo State Network version tested on 1D and 3D movement data. The ordinate uses a logarithmic scale. Hence, lower values mean a better prediction. The error bars represent the standard deviation from the mean. The different tests are labeled with "Jaeger" and "Steil" using the networks presented in [4] and [6] respectively. For Jaeger networks, two versions are tested. On the one hand, parameters, like number of neurons, spectral radius, and sparseness of the reservoir, where set to fixed values. On the other hand, those parameters are obtained randomly. For Steil networks, the number of neurons is increased (25, 100, 250). Additionally, version 3 of Steil network uses input $\mathbf{s_{n-1}}$ and $\mathbf{s_{n-2}}$ as input (not only $\mathbf{s_{n-1}}$ as for all other tests)

Comparison of the Echo State Networks versions Since, the literature provides slightly different variants of Echo State Networks, two different ones are evaluated here. On the one hand, networks with a structure from [4], called in the following Jaeger networks and on the other hand, networks with a structure from [6] (Steil networks). For both networks, the spectral radius is set differently. While Jaeger [4] uses $spec = 0.8$, with Steil networks it is set to $spec = 1.0$.

Both networks are evaluated on real motion data (see Fig. 3). As already mentioned, the motion data is available as a trajectory in 3D Cartesian space. These 3D points are used directly as input for the network (labeled "3D" in Fig. 3), or they are split into three 1D time series, predicted independently with three networks (labeled "1D" in Fig. 3).

It is recommended in [6] to initialize all Steil network weights to 0.05. Since only weights to the output layer are adapted during training process, all other weights stay at 0.05. Actually, this value could not be confirmed with the test on movement data. It could be shown for both network versions, that the the feedback weights $\mathbf{w}_{back}$ must be scaled very low (about 10^{-20}) to guarantee stable networks. Furthermore, the input weights $\mathbf{w}_{in}$ are set to values of about 10^{-5}. For all other weights the influence of the chosen values is not that significantly.

Steil networks have additional weights to the output layer ($\mathbf{w}_{dir}, \mathbf{w}_{rec}$). These weights can be included in the learning process as it is suggested in [6]. Unfortunately, this leads to instable networks, so that these weights were not learned for predicting the movement data. These weights should be scaled low about 10^{-20} as well, because they have a similar function like the feedback weights $\mathbf{w}_{back}$.

Jaeger [4] suggests to use 50 up to 2000 neurons for the reservoir. In the prediction of movement data, the size of the reservoir is set to lie between 25 and 250 neurons. However, a higher number of neurons doesn't lead to significant better results.

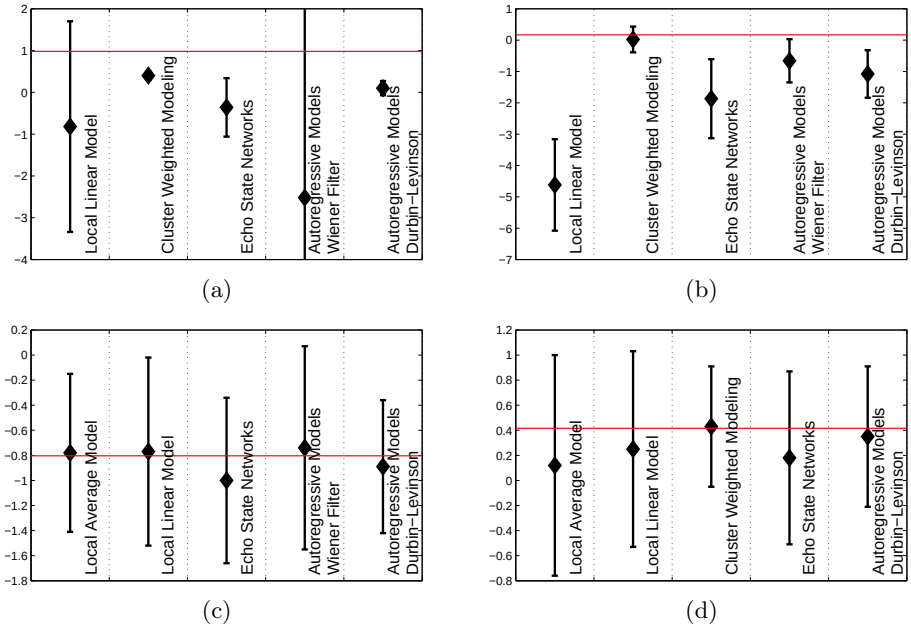


Fig. 4. The graphs shows the STE (b), (c) and LTE (a), (d) plotted for each of the algorithms tested with sine (a), Lorenz-attractor data (b), and movement data (c), (d). The ordinate uses a logarithmic scale. Hence, lower values mean a better prediction. The error bars represent the standard deviation from the mean. For the STE all results lie relatively close together while the reference algorithm can only be beaten clearly by the Echo State Networks. Longer predictions show more differences in the results of the algorithms. Also the mean errors are higher than STE, as expected in longer predictions. The reference is beaten more clearly in general. Local Average Models and Echo State Networks show the best results.

Steil [6] adviises to apply Intrinsic Plasticity (adaption of transfer function parameters) for the first 200 time steps to improve classification results of the network. Those benchmark results were gained by applying the online learning rule. Since only offline learning rule is used here, the results could not be confirmed. In both types of networks, Intrinsic Plasticity seems to have only minor effects when the offline learning rule is applied.

Additionally, Steil networks were extended in a TDNN-like fashion, using more than only the last point of the trajectory as input (labeled "Steil 3" in Fig. 3). It can be observed that this leads to better predictions in the very first steps (about 5) but may destabilize the prediction in the following steps. In general, it leads to worse results for the chosen quality measures as they include 75 prediction steps.

Comparison to the reference algorithms In the prediction of sine trajectories, the Autoregressive Models show the best results in the mean for STE and

LTE (see Fig. 4(a)). Note that these algorithms can build up to high values, so that the standard deviation in this case is very high.

With worse mean errors the standard deviation is also lower. The Local Models and Echo State Networks lead also to quite good prediction results, while the used reference (data repetition of linear approximation) is beaten clearly by all prediction algorithms.

For predicting the chaotic Lorenz-Attractor (see Fig. 4(b)) the Local Linear Model leads to the best results. Echo State Networks perform also well – especially with higher number of neurons. Here, the reference algorithms are outperformed clearly, as well. The standard deviation of the prediction quality is relatively high.

For the prediction of real movement data, the Echo State Networks lead to the best results for the STE as it is shown in Fig. 4(c), while for long term prediction Local Models have slightly better results (Fig. 4(d)). The AR models perform barely better than the reference. Here the Durbin-Levinson algorithm achieves the best prediction quality. Cluster Weighted Models show the worst performance and their mean errors stay even behind the simple reference algorithms.

It seems that the usage of the additional weights in the Steil networks ($\mathbf{w}_{dir}, \mathbf{w}_{rec}$) destabilizes the prediction especially over a long term, because of the fact that these networks have considerably worse prediction results.

In general, the difference between each of the algorithms and to the reference is much smaller than for the predictions of the sine or Lorenz-Attractor trajectories. Nevertheless, the best algorithms still beat the simple references clearly (as expected) and are able to predict movement trajectories, consisting of several hundred hypothetical steps, very well.

It can be concluded that the prediction of movement data seems to be a harder problem than predicting standard chaotic trajectories, such as those generated by the Lorenz-Attractor. This is caused by unique unexpected and unpredictable behavior, which can be observed in the movement data. Therefore, the choice of the number of neurons in the ESN reservoir, for example, has only a minor effect. In tests, the difference in the prediction results of movement data between 25 and 250 neurons were insignificant. It can be presumed that the structure of the movement data does not allow a higher accuracy in the prediction unlike other chaotic time series [4].

4 Conclusion

The intention of this paper was to connect the well-known fields of time series prediction and movement data handling in a new way. It was possible to show, that some of the algorithm are preforming well, while predicting movement data. Generally, it can be said that movement data behaves different than periodical sine and chaotic Lorenz-Attractor time series.

The tested algorithms show good results on movement data. However, some improved versions of the algorithms, which show good results for sine and Lorenz-attractor time series, show no benefit for movement data.

Echo State Networks and Local Models turned out to be suitable algorithms for movement prediction.

Autoregressive Models and again ESNs are able to predict fast enough for an online application without any further adaption. From the current point of view Echo State Networks are the "winning" approach which are able to solve the problem. Hence, further analysis should put the focus on this approach and on the additional improvements, which have not yet been tested.

The other algorithms can be upgraded as well. Local Models could be a good alternative to ESNs, if they could be accelerated without deterioration of quality. Besides, the enhanced versions of the AR Models such as ARMA or ARIMA models could be tested. Furthermore, the usage of an irregular embedding is imaginable.

As a next step, an adequate proactive navigation and interaction strategy exploiting the prediction results needs to be investigated. One drawback for predicting motion data is the fact that human beings may perform unexpected motion. Since the discussed algorithms rely on the past characteristics, it is possible to use them for detection of such unexpected behavior (as some kind of "saliency"-cue) for improving robot human interaction.

References

1. Schröter, C., Böhme, H.J., Gross, H.M.: Memory-efficient gridmaps in rao-blackwellized particle filters for slam using sonar range sensors. In: Proc. European Conference on Mobile Robots. (2007)
2. Pett, S., Fraichard, T.: Safe navigation of a car-like robot within a dynamic environment. In: Proc. of European Conf. on Mobile Robots. (2005) 116 – 121
3. Owen, E., Montano, L.: Motion planning in dynamic environments using the velocity space. In: Proc. of RJS/IEEE IROS. (2005) 997 – 1002
4. Jaeger, H., Haas, H.: Harnessing nonlinearity: predicting chaotic systems and saving energy in wireless telecommunication. Science **April** (2004) 78 – 80
5. Scheidig, A., Müller, S., Martin, C., Gross, H.M.: Generating person's movement trajectories on a mobile robot. In: Proc. of International Symposium on Robots and Human Interactive Communications. (2006) 747 – 752
6. Steil, J.J.: Online reservoir adaptation by intrinsic plasticity for backpropagation-decorrelation and echo state learning. Neural Networks **20** (2007) 353 – 364
7. http://paco.psy.gla.ac.uk/data_ptd.php
8. Wiener:, N.: Extrapolation, Interpolation, and Smoothing of Stationary Time Series. Wiley (1949)
9. Shumway, R.H., Stoffer, D.S.: Time Series Analysis and Its Applications. Springer Texts in Statistics (2000)
10. Abarbanel, H., Parlitz, U.: Nonlinear Analysis of Time Series Data. In: Handbook of Time Series Analysis. WILEY-VCH (2006) 1 – 37
11. Engster, D., Parlitz, U.: Local and Cluster Weighted Modeling for Time Serie Prediction. In: Handbook of Time Series Analysis. WILEY-VCH (2006) 38 – 65
12. Gross, H.M., Richarz, J., Müller, S., Scheidig, A., Martin, C.: Probabilistic multi-modal people tracker and monocular pointing pose estimator for visual instruction of mobile robot assistants. In: Proc. World Congress on Comp. Intelligence. (2006)