Action-Driven Perception for a Humanoid

Jens Kleesiek[1,3], Stephanie Badde[2], Stefan Wermter[3], and Andreas K. Engel[1]

[1] Department of Neurophysiology and Pathophysiology,
University Medical Center Hamburg-Eppendorf, Hamburg, Germany,
{j.kleesiek, ak.engel}@uke.uni-hamburg.de
[2] Department of Biological Psychology and Neuropsychology,
University of Hamburg, Hamburg, Germany,
Stephanie.Badde@uni-hamburg.de
[3] Department of Informatics, Knowledge Technology,
University of Hamburg, Hamburg, Germany,
wermter@informatik.uni-hamburg.de

Abstract. We present active object categorization experiments with a real humanoid robot. For this purpose, the training algorithm of a recurrent neural network with parametric bias has been extended with adaptive learning rates. This modification leads to an increase in training speed. Using this new training algorithm we conducted three experiments aiming at object categorization. While holding different objects in its hand, the robot executes a motor sequence that induces multi-modal sensory changes. During learning, these high-dimensional perceptions are 'engraved' in the network. Simultaneously, low-dimensional PB values emerge unsupervised. The geometrical relation of these PB vectors can then be exploited to infer relations between the original high dimensional time series characterizing different objects. Even sensations belonging to unknown objects can be discriminated from known (learned) ones and kept apart from each other reliably. Additionally, we show that the network tolerates noisy sensory signals very well.

Keywords: Active Perception, RNNPB, Humanoid Robot

1 Introduction

Motor actions determine the sensory information that agents receive from their environment. Combining sensory and motor processes dynamically facilitates many tasks, one of those being object classification.

The intention of this experiment is to provide a neuroscientifically and philosophically inspired model for *what do objects feel like?* For this purpose, we stress the active nature of perception within and across modalities. According to the sensorimotor contingencies theory [1], actions are fundamental for perception and help to distinguish the qualities of sensory experiences in different sensory channels (e.g. 'seeing' or 'touching'). O'Regan and Noë actually suggest that "seeing is a way of acting" [1]. Exactly this is mimicked in our computational study.

It has been shown that if the fruit fly drosophila cannot recognize a pattern it starts to move [2]. It is also known that flies use motion to visually determine the depth of perceived obstacles [3]. Similarly, pigeons bob their heads up and down to recover depth information [4]. Not only living beings, but robots too are embodied [5], and they have the ability to act and to perceive. In the presented experiments the robot actually needs to act to perceive the objects it holds in its hand. The action-driven sensations are guided by the physical properties of its body, the world and the interplay of both.

A humanoid robot moves toy bricks up and down and rotates them back and forth, while holding them in its hand. The induced multi-modal sensory impressions are used to train a modified version of a recurrent neural network with parametric bias (RNNPB), originally developed by Tani and Ito [6]. The robot is able to self-organize the contextual information and in turn, to use this learned sensorimotor knowledge for object classification. Due to the overwhelming generalization capabilities of the recurrent architecture, the robot is even able to correctly classify unknown objects. Furthermore, we show that the proposed model is very robust against noise.

2 Theory

Despite its intriguing properties, the recurrent neural network with parametric bias has hardly been used by anybody other than the original authors. Mostly, the architecture is utilized to model the mirror neuron system [7, 8]. Here we apply the variant proposed by Cuijpers *et al.* [8] using an Elman-type structure [9] at its core. Furthermore, we modify the training algorithm to include adaptive learning rates for training of the weights, as well as the PB values. This results in an architecture that is more stable and converges faster.

2.1 Storage

The recurrent neural network with parametric bias (an overview of the architecture unfolded in time can be seen in Fig. 1) can be used for the storage, retrieval and recognition of sequences. For this purpose, the parametric bias (PB) vector is learned simultaneously and *unsupervised* during normal training of the network. The prediction error with respect to the desired output is determined and backpropagated through time using the BPTT algorithm [9]. However, the error is not only used to correct all the synaptic weights present in the Elman-type network. Additionally, the error with respect to the PB nodes δ^{PB} is accumulated over time and used for updating the PB values after an entire forward-backward pass of a single time series, denoted as epoch e. In contrast to the synaptic weights that are shared by all training patterns, a unique PB vector is assigned to each individual training sequence. The update equations for the i-th unit of the parametric bias pb for a time series of length T is given as:

$$\rho_i(e+1) = \rho_i(e) + \gamma_i \sum_{t=1}^{T} \delta_{i,t}^{\mathrm{PB}} \ , \tag{1}$$

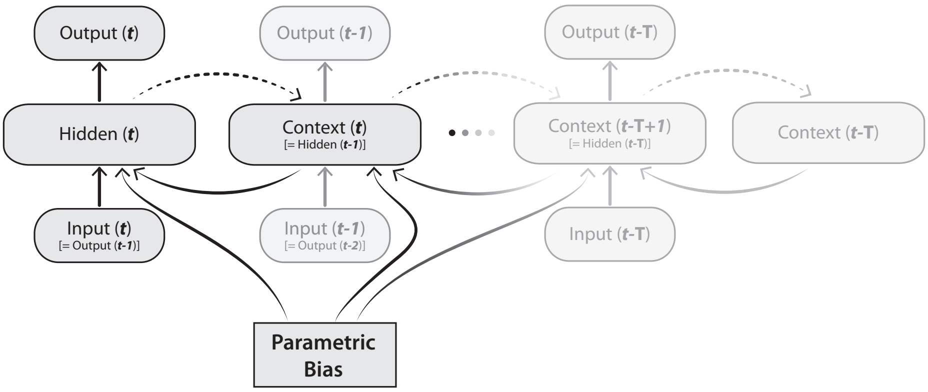

Fig. 1. Network architecture. The Elman-type Recurrent Neural Network with Parametric Bias (RNNPB) unfolded in time. Dashed arrows indicate a verbatim copy of the activations (weight connections set equal to 1.0). All other adjacent layers are fully connected. t is the current time step, T denotes the length of the time series.

$$pb_i(e) = \text{sigmoid}(\rho_i(e)) \ , \tag{2}$$

where γ is the update rate for the PB values, which in contrast to the original version is not constant during training and not identical for every PB unit. Instead, it is scaled proportionally to the absolute mean value of prediction errors being backpropagated to the i-th node over time T:

$$\gamma_i \propto \frac{1}{T} \left\| \sum_{t=1}^{T} \delta_{i,t}^{\text{PB}} \right\| \ . \tag{3}$$

The other adjustable weights of the network are updated via an adaptive mechanism, inspired by the resilient propagation algorithm proposed by Riedmiller and Braun [10]. However, there are decisive differences. First, the learning rate of each neuron is adjusted after every epoch. Second, not the sign of the partial derivative of the corresponding weight is used for changing its value, but instead the partial derivative itself is taken.

To determine if the partial derivative of weight w_{ij} changes its sign we can compute:

$$\epsilon_{ij} = \frac{\partial E_{ij}}{\partial w_{ij}}(t-1) \cdot \frac{\partial E_{ij}}{\partial w_{ij}}(t) \tag{4}$$

If $\epsilon_{ij} < 0$, the last update was too big and the local minimum has been missed. Therefore, the learning rate η_{ij} has to be decreased by a factor $\xi^- < 1$. On the other hand, a positive derivative indicates that the learning rate can be increased by a factor $\xi^+ > 1$ to speed up convergence. This update of the learning rate can be formalized as:

$$\eta_{ij}(t) = \begin{cases} \mathbf{max}(\eta_{ij}(t-1) \cdot \xi^{-}, \eta_{min}) & \text{if } \epsilon_{ij} < 0, \\ \mathbf{min}(\eta_{ij}(t-1) \cdot \xi^{+}, \eta_{max}) & \text{if } \epsilon_{ij} > 0, \\ \eta_{ij}(t-1) & \text{else.} \end{cases} \tag{5}$$

The succeeding weight update $\mathit{\Delta} w_{ij}$ then obeys the following rule:

$$\mathit{\Delta} w_{ij}(t) = \begin{cases} -\mathit{\Delta} w_{ij}(t-1) & \text{if } \epsilon_{ij} < 0, \\ \eta_{ij}(t) \cdot \frac{\partial E_{ij}}{\partial w_{ij}}(t) & \text{else.} \end{cases} \tag{6}$$

In addition to reverting the previous weight change in the case of $\epsilon_{ij} < 0$ the partial derivative is also set to zero ($\frac{\partial E_{ij}}{\partial w_{ij}}(t) = 0$). This prevents changing of the sign of the derivative once again in the succeeding step and thus a potential double punishment.

We use a nonlinear activation function with parameters recommended by LeCun *et al.* [11] for all neurons in the network, as well as for the PB units (Eq. 2):

$$\text{sigmoid}(x) = 1.7159 \cdot \tanh\left(\frac{2}{3} \cdot x\right) . \tag{7}$$

2.2 Retrieval

The PB vector is usually low dimensional and resembles bifurcation parameters of a nonlinear dynamical system, i.e. it characterizes fixed-point dynamics of the RNN. During training the PB values are self-organized, thereby encoding each time series and arranging it in PB space according to the properties of the training pattern. This means that the values of similar sequences are clustered together, whereas more distinguishable ones are located further apart. Once learned, the PB values can be used for the generation of the time series previously stored. For this purpose, the network is operated in closed-loop mode. The PB values are 'clamped' to a previously learned value and the forward pass of the network is executed from an initial input $I(0)$. In the next time steps, the output at time t serves as an input at time $t + 1$. This leads to a reconstruction of the training sequence with a very high accuracy (limited by the convergence threshold used during learning).

2.3 Recognition

A previously stored (time) sequence can also be recognized via its corresponding PB value. Therefore, the observed sequence is fed into the network without updating any connection weights. Only the PB values are accumulated according to Eq. 1 and 2 using a constant learning rate γ this time. Once a stable PB vector is reached, it can be compared to the one obtained during training.

2.4 Generalized Recognition and Generation

The network has substantial generalization potential. Not only previously stored sequences can be reconstructed and recognized. But, (time) sequences apart from the stored patterns can be generated. Since only the PB values but not the synaptic weights are updated in recognition mode, a stable PB value can also be assigned to an unknown sequence.

For instance, training the network with two sine waves of different frequencies allows cyclic functions with intermediate frequencies to be generated simply by operating the network in generation mode and varying the PB values within the interval of the PB values obtained during training. Furthermore, the PB values obtained during recognition of a previously unseen sine function with an intermediate frequency (w.r.t. the training sequences) will lie within the range of the PB values acquired during learning. Hence, the network is able to capture a reciprocal relationship between a time series and its associated PB value.

These generalized recognition and generation capabilities of the modified RNNPB are demonstrated in a more complex example. For this purpose, consider the 2-D sinusoidal sequences described by the following equation:

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = 0.5 \begin{pmatrix} cos\ \theta & -sin\ \theta \\ sin\ \theta & cos\ \theta \end{pmatrix} \begin{pmatrix} sin\ \frac{\pi \cdot t}{6} \\ cos\ \frac{\pi \cdot t}{12} \end{pmatrix} \tag{8}$$

Plotting x_1 *vs.* x_2 results in a figure-eight shape that is rotated according to the angular value specified by θ. Two 2-D time series of length $t = 25$ were generated using $\theta = 90$ and $\theta = 180$, respectively. In contrast to the robot experiments presented below, the network only has a single PB unit.

After training, the network is able to recognize those sequences based on their trained PB values ($\mathrm{PB}_{\theta:90} = -0.01107$ and $\mathrm{PB}_{\theta:180} = -0.65604$), which differ only by a small amount ($\epsilon_{\theta:90} = 0.0005$ and $\epsilon_{\theta:180} = 0.002$) from the ones obtained during storage. The PB values of the two trained sequences are plotted in Fig. 2 using white circular markers. Next to the trained sequences, the network is also fed with novel, previously untrained, sequences. These are generated using Eq. 8 with varying θ values. The network also generates stable PB values for those unknown sequences (black dots in Fig. 2). It can be seen that the PB values are ordered according to the angular value of the underlying time series. This reciprocal relationship can be used to infer the angular value of a sequence generated with an unknown θ value. Thus, varying the PB value results in a rotation of the figure-eight shape.

2.5 Evaluation of the Adaptive Learning Rate

To evaluate the adaptive learning rate proposed in Sec. 2.1, artificial 1-D test data of length $T = 11$ in the interval $[-\pi; \pi[$ is generated using the following equations:

$$x = \sin(t)\,, \tag{9}$$

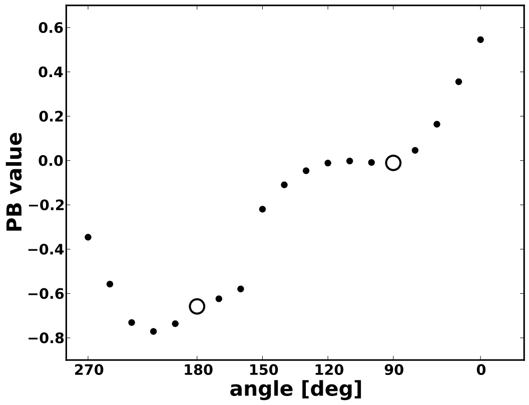

Fig. 2. Generalized recognition of trained and untrained sequences. The PB values of the two trained 2-D time series using Eq. 8 with $\theta = 90$ and $\theta = 180$, respectively, are marked using white circles. In contrast, the PB values obtained by feeding the network with untrained sequences generated with varying θ values are drawn as black dots. These values are arranged in a structured way, emphasizing the self-organization of the PB space.

$$x = \frac{\sin(3t) \cdot \sin(t)}{2t^2} - 0.5 \,. \tag{10}$$

Eq. 9 is referred to as `sin` and Eq. 10 as `sinc`. Except for the following differences, the RNNPB network parameters were identical to the parameters of the robot experiments (see below). The architecture contained only one input and one output node, as well as only one PB unit. The convergence criterion was set to 10^{-4}.

2.6 Network Parameters for Robot Experiments

Based on systematic empirical trials, the following parameters have been determined for our experiments. The network contained two input and two output nodes, 24 hidden and 24 context neurons as well as 2 PB units. The convergence criterion for back propagation through time (BPTT) was set to 10^{-6} in the first, and 10^{-5} in the second experiment. For recognition of a sequence, the update rate γ of the PB values was set to 0.1. The values for all other individual adaptive learning rates (Eq. 5) during training of the synaptic weights were allowed to be in the range of $\eta_{min} = 10^{-12}$ and $\eta_{max} = 50$; depending on the gradient they were either increased by $\xi^{+} = 1.01$ or decreased by a factor $\xi^{-} = 0.9$.

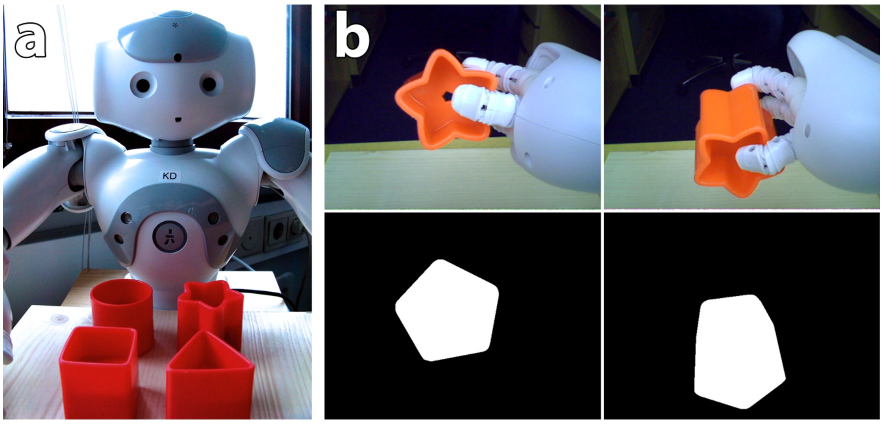

Fig. 3. Scenario. a) Toy bricks in front of the humanoid robot Nao. The toy bricks exist in four different shapes, have an identical color and are either light-weight (15 g) or heavy (50 g). This results in a total of eight categories that have to be distinguished by the robot. b) Rotation movement with the star-shaped object captured by the robot camera. In the upper row the raw camera image is shown, whereas the bottom row depicts the preprocessed image that is used to compute the visual features.

3 Scenario

The humanoid robot Nao[1] is programmed to conduct the experiments (Fig. 3 a). The task for the robot is to identify which object (toy brick) it holds in its hand. In total there are eight object categories that have to be distinguished by the robot: the toy bricks have four different shapes (circular-, star-, rectangular- and triangular-shaped), of which each exists in two different weight versions (light and heavy). Hence, for achieving a successful classification multi-modal sensory impressions are required. Additionally, *active* perception is necessary to induce sensory changes essential for discrimination of –depending on the perspective– similar looking shapes (e.g. star- and circular-shaped objects). For this purpose, the robot performs a predefined motor sequence and simultaneously acquires visual and proprioceptive sensor values.

3.1 Data Acquisition

The recorded time series comprises 14 sensor values for each modality. In each single trial the robot turns its wrist with the object between its fingers by 45.8 ° back and forth twice, followed by lifting the object up and down three times (thereby altering the pitch of the shoulder joint by 11.5 °) and, finally, turning it again twice.

After an action has been completed, the raw image of the lower camera of the Nao robot is captured, whereas the electric current of the shoulder pitch servo

[1] `http://www.aldebaran-robotics.com`

motor is recorded constantly (sampling frequency 10 Hz) over the entire movement interval. For each object category 10 single trial time series are recorded in the described way and processed in real-time. This yields 80 bi-modal time series in total.

3.2 Data Processing

For the proprioceptive measurements only the mean values are computed for the time intervals in between movements. The visual processing, on the other hand, involves several steps (Fig. 3 b), which are accomplished using OpenCV [12]. First, the raw color image is converted to a binary image using a color threshold. Next, the convex hull is computed and, based on that, the contour belonging to the toy brick is extracted [13]. For the identified contour the first Hu moment is calculated [14]. Finally, the visual measurements are scaled to be within the interval $[-0.5, 0.5]$.

3.3 Training and Test Data

For testing, the data of single trials are used, i.e. 10 2-D time series per object category (one dimension for each modality). However, for training, a prototype is determined for each object category and modality (Fig. 4). To obtain this subclass representative, the mean value of pooled single trials, with regard to identical object properties, is computed. This means that, for instance, all circular-shaped objects are combined ($n = 20$) and used to compute the visual prototype for circular-shaped objects. To find the proprioceptive prototype for e.g. all heavy objects, all individual measurements with this property ($n = 40$) are aggregated and used to calculate the mean value at each time step. The subclass prototypes are then combined to form a 2-D multi-modal time series that serves as an input for the recurrent neural network during training.

4 Results

4.1 Evaluation of the Adaptive Learning Rate

To evaluate the improvements caused by introduction of the adaptive learning rate as described in Sec. 2.1, an RNNPB was trained 1000 times with two 1-D sequences (Eq. 9 and 10). The results are statistically evaluated using a t-test. To compensate for the sample size bias, the optimal sample size was determined based on the mean value and the standard deviation of the data. This optimal sample size was used to draw 10,000 random subsets of the data, which were subsequently evaluated to obtain an average p-value for the t-test. The results are summarized in Tab. 1. The modifications lead, on average, to a 22-fold speedup of the training times ($t(5) = -17.13$, $p = 0.000$) for this particular example. Also the number of recognition steps has improved significantly ($t(20) = -3.55$, $p = 0.002$). However, no significant changes of the retrieval accuracy measured with the mean squared error (MSE) can be found.

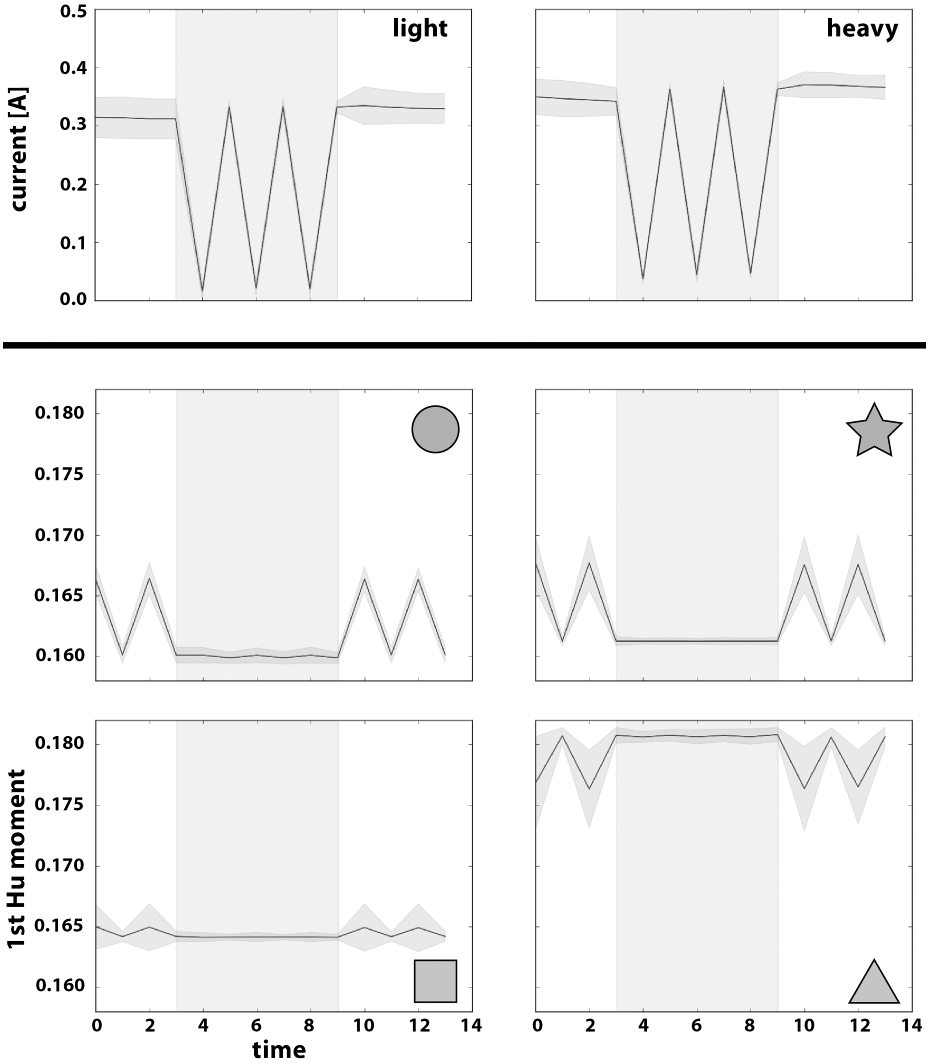

Fig. 4. Training data. The mean values of the two weight conditions (light and heavy, top) and the four visual conditions (matching symbols, bottom) are shown. These mean time series are used as prototypes for training the RNNPB. Vertical gray shaded areas represents the up and down movement, whereas back and forth movements are unshaded. The area surrounding the signals delineates two standard deviations from the mean.

Table 1. Statistical evaluation of the adaptive learning rate. Mean values and standard deviations are shown, significant changes (t-test, $p < 0.005$) are marked bold.

	Modified RNNPB	Classical RNNPB	Factor
Total steps	**5,520 (±1,713)**	122,709 (±20,027)	22.2
Total time	**34 *s* (±10)**	751 *s* (±124)	22
MSE *sin*	4.3×10^{-4} $(\pm 1.2 \times 10^{-3})$	5.5×10^{-4} $(\pm 3 \times 10^{-4})$	–
MSE *sinc*	4.7×10^{-4} $(\pm 8.7 \times 10^{-4})$	1.9×10^{-4} $(\pm 1.9 \times 10^{-4})$	–
Recognition steps	**192 (±85)**	284 (±101)	1.48

Plotting the average MSE against the number of steps needed until the convergence criterion is reached, further highlights the drastic improvement in speed (Fig. 5). The error, shown separately for both sequences, decreases for both algorithms in a similar manner. However, the adaptive version looks 'compressed' in comparison to the classical algorithm. In addition, the fluctuations are reduced, indicating a more stable behavior of the modified RNNPB.

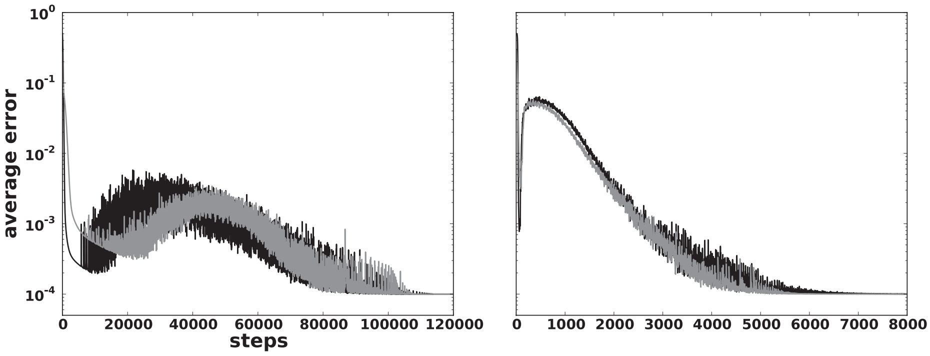

Fig. 5. Error plot comparing classical (left) to modified (right) RNNPB. The mean squared error (MSE) of the `sin` sequence is shown in black, whereas the average MSE of the `sinc` sequence is drawn in gray.

4.2 Classification Using All Object Categories for Training

In the first experiment the modified recurrent neural network with parametric bias was trained with the bi-modal prototype time series of all eight object categories (Sec. 3.3). During training, the PB values for the respective categories emerged in an unsupervised way. This means, the two-dimensional PB space self-organizes based on the inherent properties of the sensory data that was presented to the network. Hence, objects with similar dynamic sensory properties are clustered together. This can be seen in Fig. 6. For instance, the learned PB vectors representing star- and circular-shaped objects, either light-weight (white symbol) or heavy (black symbol), are located in close proximity, whereas the PB values coding for the triangular-shaped objects are positioned more distantly. This is due to the deviating visual sensory impression they generate (Fig. 4). The experiment has been repeated several times with different random initializations of the network weights. However, the obtained PB values of the different classes always demonstrate a comparable geometric relation with respect to each other.

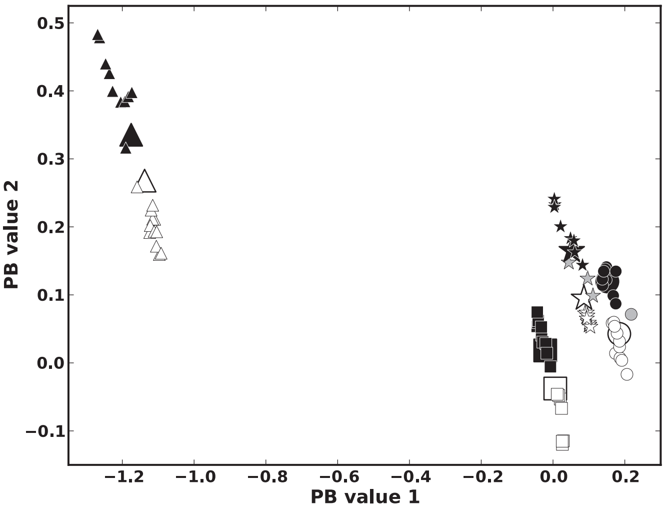

Fig. 6. Classification using all object categories for training. PB values are depicted using white (light-weight) and black (heavy) symbols matching the corresponding shape. Large symbols represent the class prototypes used for training. Smaller symbols depict PB values obtained during testing with bi-modal single trial data. If the objects have not been correctly classified, they are shown gray.

To demonstrate the retrieval properties (Sec. 2.2) of the fully trained architecture, the PB values acquired during training were 'clamped' to the network. Operating the network in closed-loop mode showed that the input sequences used for training can be retrieved with a very high accuracy. As an example this is shown in Fig. 7 (left) for the heavy star-shaped object.

The steps needed until stable PB values are reached, which in turn can be used for recognition, are illustrated in Fig. 8. The bi-modal sensory sequences for all light-weight and heavy objects were fed consecutively into the network. On

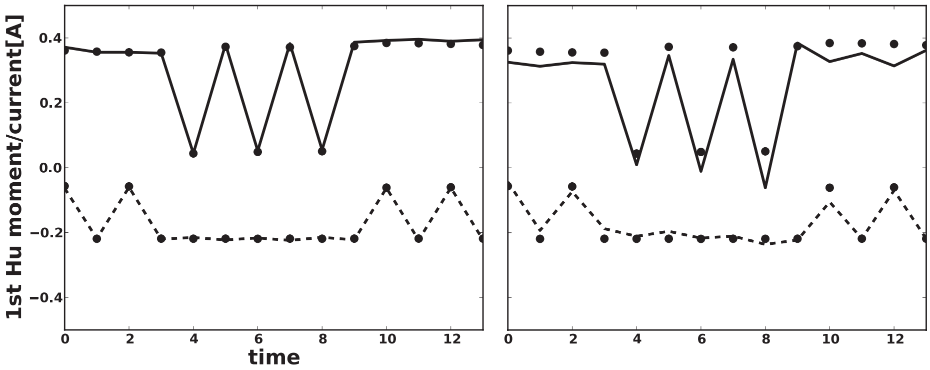

Fig. 7. Retrieval and generation capabilities. Solid dots represent the proprioceptive and visual sampling points of the heavy star-shaped prototype time series (Fig. 4). Dashed (visual) lines and solid (proprioceptive) lines are the time series generated by the network operated in closed-loop with 'clamped' PB values as the only input. The PB values have been acquired unsupervised either during full training (left) or partial training (right). During partial training (right) the network has only been trained with the prototype sequences for the light-weight circle and the heavy triangle. Still, the network is able to generate a fairly accurate sensory prediction for the (untrained) heavy star-shaped object.

average it took less than 100 steps (about 200 ms on a contemporary desktop computer) until the PB values converged. The convergence criterion was set to 20 consecutive iterations where the cumulative change of both PB values was $< 10^{-5}$. To assure that the PB values reached a stable state, this number has been successfully increased to 100,000 consecutive steps in preliminary experiments (not shown). Note, that the network and PB values was not reinitialized when the next sensory sequence was presented to the network. Thus, the robot can continuously interact with the toy bricks and is able to immediately recognize an object based on its sensorimotor sequence.

For testing, the network was operated in generalized recognition mode (Sec. 2.4). Single trial bi-modal sensory sequences were presented to the network that in turn provided an 'identifying' PB value. The class membership, i.e. which object the robot holds in its hand and how heavy this object is, was then determined based on the minimal Euclidean distance to the PB values of the class prototypes (large symbols). In Fig. 6 the PB values of all 80 single trial test patterns are depicted.

Only 4 out of 80 objects are misclassified (shown in gray), yielding an error rate of 5 %. Interestingly, only star- and circular-shaped objects are confused by the network, which indeed generate very similar sensory impressions (cf. Fig. 4). To assess the meaning of the error rate and estimate how challenging the posed problem is, we evaluated the data with two other commonly used techniques in machine learning. First, we trained a multi-layer perceptron (28 input, 14 hidden and one output unit) with the prototype sequences. Testing with the single

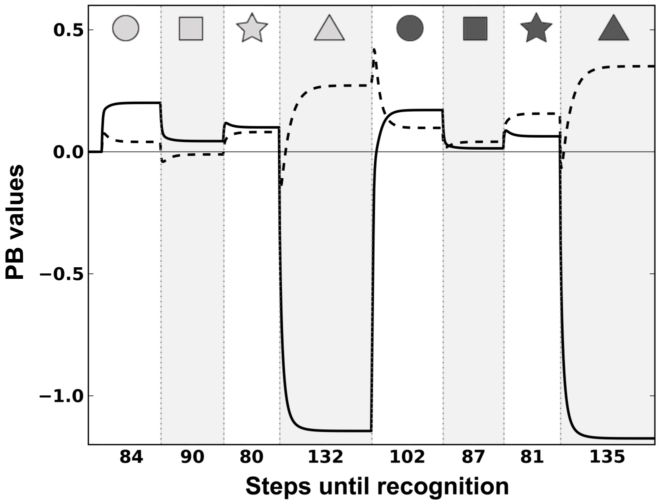

Fig. 8. Steps until stable PB values are reached. Bi-modal sensory sequences for all light-weight and heavy objects (represented by matching symbols in light and dark gray, respectively) are consecutively fed into the network. The time courses of PB value 1 (solid line) and PB value 2 (dashed line) during the recognition process are plotted.

trial data resulted in an error rate of 46.8 %, reflecting weaker generalization capabilities of the non-recurrent architecture. Next, we trained and evaluated our data with a support vector classifier (SVC) using default parameters [15]. In contrast, this method is able to classify the data perfectly.

4.3 Classification Using Only the Light Circular-Shaped and the Heavy Triangular-Shaped Object for Training

In the second experiment, only the bi-modal prototypes for the light circular- and heavy triangular-shaped objects were used to train the RNNPB. Although, the absolute PB values obtained during training differ from the ones being determined in the previous experiment, their relative Euclidean distance in PB space is nearly the same (1.39 *vs.* 1.35), stressing the data-driven self-organization of the parametric bias space.

For testing, initially only the bi-modal sensory time series matching the two training conditions were fed into the network, thereby determining their PB values. Using the Euclidean distance subsequently to obtain the class membership resulted in a flawless identification of the two categories.

Further evaluation of the single trial test data was performed in two stages. In a primary step the remaining test data was presented to the network and the respective PB values were computed (generalized recognition, Sec. 2.4). Despite not having been trained with prototypes for the remaining six object categories, the network is able to cluster PB values stemming from similar sensory situations, i.e. identical object categories. In a succeeding step we computed the centroid for each class (mean PB value) and classified again based on the Euclidean distance. This time only two single trial time series were misclassified by the network (error rate 2.5 %). The results are shown in Fig. 9.

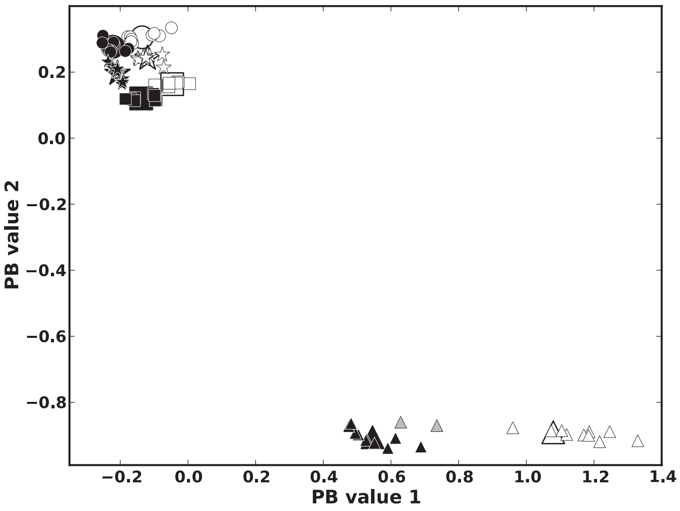

Fig. 9. Classification using only the light circular-shaped and the heavy triangular-shaped object for training. PB values of the class prototypes and the a posteriori computed cluster centers of the untrained object categories are depicted using larger symbols that match the objects shape. Smaller symbols are used for PB values of sensory data of single trials. If the objects have not been classified correctly they are shown in gray, otherwise white is used for light-weight and black for heavy-weight objects.

The generalization potential (Sec. 2.4) of the architecture is presented in Fig. 7 (right) for the heavy star-shaped object. For this purpose, the mean PB values (centroid of the respective class) were clamped to the network, which was operated in closed-loop mode. The network had only been trained with the light circular- and the heavy triangular-shaped object. Still, it was possible to generate sensory predictions for unseen objects, e.g. the heavy star-shaped toy brick, that match the real sensory impressions fairly well.

4.4 Noise Tolerance

Based on the network weights that had been obtained in experiment 2 (training the RNNPB only with the bi-modal prototypes for the light circular- and heavy triangular-shaped objects), we evaluated the noise tolerance of the recurrent neural architecture. For this purpose, uniformly distributed noise of increasing levels was added to the visual prototype time series (Fig. 10).

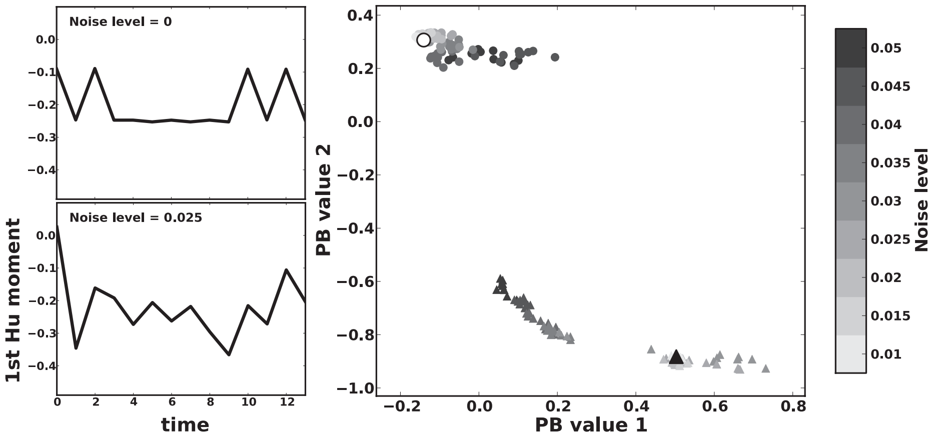

Fig. 10. Noise tolerance. On the right uniformly distributed noise of increasing levels (grayscale coded) is added to the visual prototype time series for the light-weight circle and the heavy triangle. The PB values are determined and marked with a matching symbol. The white circle and large black triangle show the PB values obtained during training without noise. On the left the impact of the noise level is shown exemplarily for the visual prototype time series of the circular shape.

Even high levels of noise allow for a reliable linear discrimination of the two classes. Furthermore, the PB values of increasing noise levels show commonalities and are clustered together, again providing evidence for a data-driven self-organization of the PB space. Thus, determining the Euclidean distance of the PB values obtained from the noisy signals to the class representatives enables not only the class membership to be determined, it also allows the noise level to be estimated with respect to the prototypical sensory impression. It also can be show that the network tolerates noise added to the time series of both modalities very well [16].

5 Discussion

By introducing modifications to the learning algorithm of the RNNPB we were able to achieve a significant 22-fold increase in speed (Tab. 1) for the storage of two 1-D signals. It was also confirmed that the storage and retrieval of those time series was stable and that learning converged in a well-behaved manner (Fig. 5). Admittedly, the storage of other sequences with e.g. a different dimensionality, length or dynamic, may well result in a different performance outcome.

After confirming flawless operation of the training algorithm we conducted three experiments aiming at object categorization. While holding different objects (Sec. 3) in its hand, the robot executes a motor sequence that induces multi-modal sensory changes. During learning these high-dimensional perceptions are 'engraved' in the network. Simultaneously, low-dimensional PB values emerge unsupervised, coding for a sensorimotor sequence characterizing the interplay of the robot with an object. We show that 2-D time series of length $T = 14$ can be reliably represented by a 2-D PB vector and that this vector allows learned sensory sequences to be recalled with a high accuracy (Fig. 7 left). Furthermore, the geometrical relation of PB vectors of different objects can be used to infer relations between the original high dimensional time series, e. g. the sensation of a star-shaped object 'feels' more like a circular-shaped object than a triangular-shaped one. Due to the experimental noise of single trials, identical objects cause varying sensory impressions. Still, the RNNPB can be used to recognize those (Fig. 6). Additionally, sensations belonging to unknown objects can be discriminated from known (learned) ones. Moreover, sensations arising from different unknown objects can be kept apart from each other reliably (Fig. 9).

Comparing the classification results of the fully trained RNNPB with the SVC reveals a superior performance of the support vector classifier. Nevertheless, it has to be kept in mind that the maximum margin classifier cannot be used to generate or retrieve time series. Interestingly, the error rate is lower if the recurrent network is only trained with two object categories (Sec. 4.3). A potential explanation, besides random fluctuations, could be that during training a common set of weights has to be found for all object categories. This process presumably interferes, due to the challenging input data, with the self-organization of the PB space.

A drawback of the presented model is that it currently operates on a fixed motor sequence. It would be desirable if the robot performed *motor babbling* [17] leading not only to a self-organization of the sensory space, but to a self-organization of the sensorimotor space. A simple solution to this problem would be to train the network additionally with the motor sequence most appropriate for an object, i.e. reflecting its affordance [18]. This would lead to an even better classification result because the motor sequences themselves would help to distinguish the objects from each other and, thus, the emerging PB values would be arranged further apart in PB space (conversely, this means currently it does not make sense to train the network with the identical motor sequences in addition). However, that does not address the fact that the robot should identify the object affordances, the movements characterizing an object, by itself.

In related research, Ogata *et al.* also extract multi-modal dynamic features of objects, while a humanoid robot interacts with them [19]. However, there are distinct differences. Despite using fewer objects in total, the problem posed in our experiments is considerably harder. Our toy bricks have approximately the same circumference and identical color. Furthermore, they exist in two weight classes with an identical in-class weight that can only be discriminated via multi-modal sensory information. We provide classification results, compare the results to other methods (MLP and SVC) and evaluate the noise tolerance of the architecture. In addition, only prototype time series are used for training (in contrast to using all single-trial time series), resulting in a reduced training time. Further, it is demonstrated that, if the network has already acquired sensorimotor knowledge of certain objects, it is able to generalize and provide fairly accurate sensory predictions for unseen ones (Fig. 7 right).

There are several potential applications of the presented model. As shown in Fig. 10, the network tolerates noise very well. This fact can be exploited for sensor de-noising. Despite receiving a noisy sensory signal, the robot still will be able to determine the PB values of the class representative based on the Euclidean distance. In turn, these values can be used to operate the RNNPB in retrieval mode (Sec. 2.2), generating the noise-free sensory signal previously stored, which then can be processed further. In fact, Körding and Wolpert suggested that the central nervous system combines, in nearly optimal fashion, visual, proprioceptive and other sensory information to overcome sensory and motor noise [20]. Next to their Bayesian framework an RNNPB might also be a possible way to model this 'de-noising' happening in the brain.

In conclusion, we present a promising framework for object classification based on action-driven perception implemented on a humanoid robot. The underlying design principles are rooted in neuroscientific and philosophical hypotheses.

Acknowledgments. This work was supported by the Sino-German Research Training Group CINACS, DFG GRK 1247/1 and 1247/2, and by the EU projects KSERA under 2010-248085 and eSMCs under ICT-270212. We thank R. Cuijpers and C. Weber for inspiring and very helpful discussions, S. Heinrich, D. Jessen and N. Navarro for assistance with the robot.

References

1. O'Regan, J.K., Noë, A.: A sensorimotor account of vision and visual consciousness. Behav Brain Sci **24**(5) (Oct 2001) 939–73; discussion 973–1031
2. Dill, M., Wolf, R., Heisenberg, M.: Visual pattern recognition in drosophila involves retinotopic matching. Nature **365**(6448) (Oct 1993) 751–3
3. Franceschini, N.: Combined optical, neuroanatomical, electrophysiological and behavioral studies on signal processing in the fly compound eye. In Taddei-Ferretti, C., ed.: Biocybernetics of vision: integrative mechanisms and cognitive processes: proceedings of the International School of Biocybernetics, Casamicciola, Napoli, Italy, 16-22 October 1994. Volume 2. World Scientific, Singapore (1997)
4. Steinman, S.B., Steinman, B.A., Garzia, R.P.: Foundations of binocular vision: a clinical perspective. McGraw-Hill, New York (2000)
5. Pfeifer, R., Lungarella, M., Iida, F.: Self-organization, embodiment, and biologically inspired robotics. Science **318**(5853) (Nov 2007) 1088–93
6. Tani, J., Ito, M.: Self-organization of behavioral primitives as multiple attractor dynamics: A robot experiment. Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on **33**(4) (2003) 481 – 488
7. Tani, J., Ito, M., Sugita, Y.: Self-organization of distributedly represented multiple behavior schemata in a mirror system: reviews of robot experiments using rnnpb. Neural Netw **17**(8-9) (2004) 1273–89

8. Cuijpers, R.H., Stuijt, F., Sprinkhuizen-Kuyper, I.G.: Generalisation of action sequences in RNNPB networks with mirror properties. In: Proceedings of the 17th European symposium on Artifical Neural Networks (ESANN). (2009) 251–256
9. Kolen, J.F., Kremer, S.C.: A field guide to dynamical recurrent networks. IEEE Press, New York (2001)
10. Riedmiller, M., Braun, H.: A direct adaptive method for faster backpropagation learning: the RPROP algorithm. In: Neural Networks, 1993., IEEE International Conference on. (1993) 586 –591 vol.1
11. LeCun, Y., Bottou, L., Orr, G., Müller, K.: Efficient backprop. Lecture Notes in Computer Science **1524** (1998) 5–50
12. Bradski, G.: The OpenCV Library. Dr. Dobb's Journal of Software Tools (2000)
13. Suzuki, S., Be, K.: Topological structural analysis of digitized binary images by border following. Computer Vision, Graphics, and Image Processing **30**(1) (1985) 32–46
14. Hu, M.K.: Visual pattern recognition by moment invariants. Information Theory, IRE Transactions on **8**(2) (1962) 179 –187
15. Chang, C.C., Lin, C.J.: LIBSVM: A library for support vector machines. ACM Transactions on Intelligent Systems and Technology **2** (2011) 27:1–27:27
16. Kleesiek, J., Badde, S., Wermter, S., Engel, A.K.: What do Objects Feel Like? Active Perception for a Humanoid Robot. In: Proceedings of the 4th International Conference on Agents and Artificial Intelligence (ICAART). Volume 1. (2012) 64–73
17. Meltzoff, A.N., Meltzoff, A.N., Moore, M.K.: Explaining facial imitation: a theoretical model. Early Development and Parenting **6** (1997) 179–192
18. Gibson, J.J.: The theory of affordances. In Shaw, R., Bransford, J., eds.: Perceiving, acting, and knowing: Toward an ecological psychology. Hillsdale, NJ: Erlbaum (1977) 67–82
19. Ogata, T., Ohba, H., Tani, J., Komatani, K., Okuno, H.G.: Extracting multi-modal dynamics of objects using RNNPB. Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, Edmonton (2005) 160–165
20. Körding, K.P., Wolpert, D.M.: Bayesian integration in sensorimotor learning. Nature **427**(6971) (Jan 2004) 244–7