An Incremental Self-Organizing Architecture for Sensorimotor Learning and Prediction

Luiza Mici, German I. Parisi, and Stefan Wermter

Abstract—**During visuomotor tasks, robots must compensate for temporal delays inherent in their sensorimotor processing systems. Delay compensation becomes crucial in a dynamic environment where the visual input is constantly changing, e.g., during the interaction with a human demonstrator. For this purpose, the robot must be equipped with a prediction mechanism for using the acquired perceptual experience to estimate possible future motor commands. In this paper, we present a novel neural network architecture that learns prototypical visuomotor representations and provides reliable predictions on the basis of the visual input. These predictions are used to compensate for the delayed motor behavior in an online manner. We investigate the performance of our method with a set of experiments comprising a humanoid robot that has to learn and generate visually perceived arm motion trajectories. We evaluate the accuracy in terms of mean prediction error and analyze the response of the network to novel movement demonstrations. Additionally, we report experiments with incomplete data sequences, showing the robustness of the proposed architecture in the case of a noisy visual input.**

Index Terms—**Hierarchical learning, motion prediction, self-organized networks.**

I. Introduction

REAL-TIME interaction with the environment requires robots to adapt their motor behavior according to the perceived events. However, each sensorimotor cycle of the robot is affected by an inherent latency introduced by the processing time of sensors, transmission time of signals, and mechanical constraints [1]–[3]. Due to this latency, robots exhibit a discontinuous motor behavior which may compromise the accuracy and execution time of the assigned task.

For social robots, delayed motor behavior makes human–robot interaction (HRI) asynchronous and less natural. Synchronization of movements during HRI may increase rapport and endow humanoid robots with the ability to collaborate with humans during daily tasks [4]. A possible solution to the sensorimotor latency is the application of predictive mechanisms which accumulate information from the robot's perceptual and motor experience and learn an internal model which estimates possible future motor states [5], [6]. The learning of these models in an unsupervised manner and their adaptation throughout the acquisition of new sensorimotor information remains an open challenge.

Latencies between perception and possible motor behavior occur in human beings [7] as well. Such discrepancies are caused by neural transmission delays and are constantly compensated by predictive mechanisms in our sensorimotor system that account for both motor prediction and anticipation of the target movement. Miall *et al.* [8] proposed that the human cerebellum is capable of estimating the effects of a motor command through an internal action simulation and a prediction model. Furthermore, there are additional mechanisms for visual motion extrapolation which account for the anticipation of the future position and movement of the target [9]. Not only do we predict sensorimotor events in our everyday tasks but we also constantly adjust our delay compensation mechanisms to the sensory feedback [10] and to the specific task [11].

Recently, there has been a considerable growth of learning-based prediction techniques which mainly operate in a "learn then predict" approach, i.e., typical motion patterns are extracted and learned from training data sequences and then learned patterns are used for prediction [2], [12]–[14]. The main issue with this type of approach is that the adaptation of the learned models is interrupted by the prediction stage. However, it is desirable for a robot operating in natural environments to be able to learn incrementally, i.e., over a lifetime of observations, and to refine the accumulated knowledge over time. Therefore, the development of learning-based predictive methods accounting for both incremental learning and predictive behavior still need to be further investigated.

In this paper, we propose a novel architecture that learns sensorimotor patterns and predicts the future motor states in an online manner. We evaluate the architecture in the context of an imitation task in an HRI scenario. In this scenario, body motion patterns demonstrated by a human demonstrator are mapped to trajectories of robot joint angles and then learned for being immediately imitated by a humanoid robot. We approach the demonstration of the movements through motion capture with a depth sensor, which provides us with reliable estimations and tracking of 3-D human body pose. Thus, the 3-D joint positions of the skeleton model constitute the input to the architecture. The learning module captures

Manuscript received July 6, 2017; revised March 7, 2018; accepted April 9, 2018. Date of publication May 3, 2018; date of current version December 7, 2018. This work was supported in part by the EU- and City of Hamburg-Funded Program Pro-Exzellenzia 4.0, in part by the German Research Foundation DFG under Project CML (TRR 169), and in part by the Hamburg Landesforschungsförderungsprojekt. *(Corresponding author: Luiza Mici.)*

The authors are with the Department of Informatics, Knowledge Technology, University of Hamburg, 22527 Hamburg, Germany (e-mail: mici@informatik.uni-hamburg.de; parisi@informatik.uni-hamburg.de; wermter@informatik.uni-hamburg.de).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCDS.2018.2832844

spatiotemporal dependencies through a hierarchy of growing when required (GWR) [15] networks, which has been successfully applied to the classification of human activities [16], [17]. The learning algorithm processes incoming robot joint angles and progressively learns prototypes of motion segments. Finally, an extended GWR algorithm, implemented at the last layer of our architecture, approximates a prediction function and utilizes the learned motion segments for predicting forthcoming motor commands.

We evaluate our system on a dataset of three subjects performing ten arm movement patterns. We study the prediction accuracy of our architecture while being continuously trained. Experimental results show that the proposed architecture can adapt quickly to unseen patterns and can provide accurate predictions albeit continuously incorporating new knowledge. Moreover, we show that the system can maintain its high performance even when training takes place with missing sensory information.

II. Related Work

A. Motion Prediction

Motion analysis and prediction are an integral part of robotic platforms that counterbalance the imminent sensorimotor latency. Well-known methods for the tracking and prediction are the Kalman filter models, as well as their extended versions which assume nonlinearity of the system, and the hidden Markov models (HMMs). Kalman filter-based prediction techniques require a precise kinematic or dynamic model that describes how the state of an object evolves while being subject to a set of given control commands. HMMs describe the temporal evolution of a process through a finite set of states and transition probabilities. Predictive approaches based on dynamic properties of the objects are not able to provide correct long-term predictions of human motion [18] due to the fact that human motion also depends on other higher-level factors than kinematic constraints, such as plans or intentions.

There are some alternatives to approaches based on probabilistic frameworks in the literature and neural networks are probably the most popular ones. Neural networks are known to be able to learn universal function approximations and thereby predict nonlinear data even though dynamic properties of a system or state transition probabilities are not known [3], [19]. For instance, multilayer perceptrons and radial basis function networks as well as recurrent neural networks have found successful applications as predictive approaches [2], [12], [13], [20]. A subclass of neural network models, namely the self-organizing map (SOM) [21], is able to perform local function approximation by partitioning the input space and learning the dynamics of the underlying process in a localized region. The advantage of the SOM-based methods is their ability to achieve long-term predictions at much less expensive computational time [22].

Johnson and Hogg [23] first proposed the use of multilayer self-organizing networks for the motion prediction of a tracked object. Their model consisted of a low-level SOM layer learning to represent the object states and the higher SOM layer learning motion trajectories through the leaky integration of neuron activations over time. Similar approaches were later proposed by Sumpter and Bulpitt [24] and Hue *et al.* [25], who modeled time explicitly by adding lateral connections between neurons in the state layer, obtaining performances comparable to that of the probabilistic models.

Several other approaches use SOMs extended with temporal associative memory techniques [20], e.g., associating to each neuron a linear autoregressive model [26], [27]. A drawback which is common to these approaches is their assumption of knowing *a priori* the number of movement patterns to be learned. This issue can be mitigated by adopting growing extensions of the SOM such as the GWR algorithm [15]. The GWR algorithm has the advantage of a nonfixed, but varying topology and requires no specification of the number of neurons in advance. Moreover, the prediction capability of the self-organizing approaches in the case of multidimensional data sequences has not been thoroughly analyzed in the literature. In this paper, we present experimental results in the context of a challenging robotic task, whereby real-world sensorimotor sequences have to be learned and predicted.

B. Incremental Learning of Motion Patterns

In the context of learning motion sequences, an architecture capable of incremental learning should identify unknown patterns and adapt its internal structure in consequence. This topic has been the focus of a number of studies on programming by demonstration [28]. Kulić *et al.* [29] used HMMs for segmenting and representing motion patterns together with a clustering algorithm that learns in an incremental fashion based on intramodel distances. In a more recent approach, the authors organized motion patterns as leaves of a directed graph where edges represented temporal transitions [30]. However, the approach was built upon automatic segmentation which required observing the complete demonstrated task, thereby becoming task-dependent. A number of other works have also adapted HMMs to the problem of incremental learning of human motion [31]–[34]. The main drawback of these methods is their requirement for knowing *a priori* the number of motions to be learned or the number of Markov models comprising the learning architecture.

Ogata *et al.* [35] proposed a model that considers the case of long-term incremental learning. In their work, a recurrent neural network was used to learn a navigation task in cooperation with a human partner. The authors introduced a new training method for the recursive neural network in order to avoid the problem of memory corruption during new training data acquisition. Calinon and Billard [36] showed that the Gaussian mixture regression (GMR) technique can be successfully applied for encoding demonstrated motion patterns incrementally through a Gaussian mixture model (GMM) tuned with an expectation-maximization algorithm. The main limitation of this method is the need to specify in advance the number and complexity of tasks in order to find an optimal number of Gaussian components. Therefore, Khansari-Zadeh and Billard [37] suggested a learning procedure capable of modeling demonstrated motion sequences

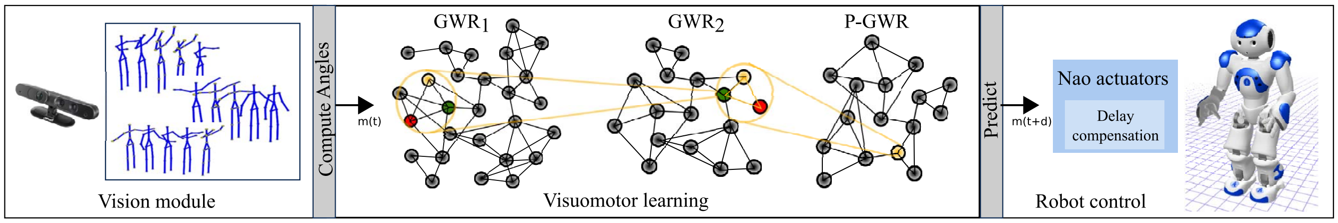


Fig. 1. Overview of the proposed system for the sensorimotor delay compensation during an imitation scenario. The vision module acquires motion from a depth sensor and estimates the 3-D position of joints. Shoulder and elbow angle values are extracted and fed to the visuomotor learning algorithm. The robot then receives predicted motor commands processed by the delay compensation module.

through an adaptive GMM. Cederborg *et al.* [38] suggested to perform a local partitioning of the input space through kd-trees and training several local GMR models.

However, for high-dimensional data, the partitioning of the input space in a real-time system requires additional computational time. Regarding this issue, it is convenient to adopt self-organizing network-based methods that perform in parallel partitioning of the input space through the creation of prototypic representations as well as the fitting of necessary local models. The application of a growing self-organizing network, such as the GWR, allows for the learning of prototypical motion patterns in an incremental fashion [39].

III. Methodology

A. Overview

The proposed learning architecture consists of a hierarchy of GWR networks [15] for processing input data sequences and learning inherent spatiotemporal dependencies (Fig. 1). The first layer of the hierarchy learns a set of spatial prototype vectors. The temporal dependence of the input data is captured as temporally ordered concatenations of consecutively matched prototypes which become more complex and of higher dimensionality when moving toward the last layer. When body motion sequences are provided, the response of the neurons in the architecture roughly resembles the neural selectivity toward temporally ordered body pose snapshots in the human brain [40]. This simple, but effective data sequence representation is also convenient in a prediction application due to implicitly mapping past values to the future ones. The concatenation vector is composed of two parts: the first part carries information about the input data at previous time steps and the second part concerns the desired output of this mapping.

The evaluation of the predictive capabilities of the proposed architecture for compensating robot sensorimotor delay will be conducted in an imitation scenario where a simulated Nao robot imitates a human demonstrator while compensating for the sensorimotor delay in an online manner.

B. Learning With the GWR Algorithm

The GWR network is composed of a set of neurons and edges that link the neurons forming topological relationships. The network starts with a set of two neurons randomly initialized and, during the learning iterations, both neurons and edges can be created, updated, or removed. At each learning iteration, t, the first and the second best-matching units (BMUs) are computed as the neurons with the smallest Euclidean distance with respect to the input sample $\mathbf{x}(t)$. The activity of the network, $a(t)$, is computed as a function of the Euclidean distance between the weight vector of the first BMU, $\mathbf{w}_b$, and the input data sample $\mathbf{x}(t)$

$$a = \exp(-||\mathbf{x}(t) - \mathbf{w}_b||). \quad (1)$$

Whenever the activity of the network is smaller than a given threshold a_T, a new neuron is added with a weight vector

$$\mathbf{w}_r = 0.5 \cdot (\mathbf{x}(t) + \mathbf{w}_b). \quad (2)$$

The activation threshold parameter, a_T, modulates the amount of generalization, i.e., the largest discrepancy between an incoming stimulus and its BMU. Edges are created between the first and the second BMUs. An edge aging mechanism takes care of removing rarely activated edges, i.e., edges exceeding the age threshold and neurons without edges consequently. In this way, neurons representing data samples that have been seen in the far past are eliminated leading to an efficient use of available resources from the lifelong learning perspective. Moreover, a firing rate mechanism that measures how often each neuron has been activated by the input leads to a sufficient training before new neurons are created. The firing rate is initially set to one and then decreases every time a neuron and its neighbors are activated in the following way:

$$\Delta h_i = \rho_i \cdot \kappa \cdot (1 - h_i) - \rho_i \quad (3)$$

where ρ_i and κ are the constants controlling the behavior of the decreasing function curve. Typically, the ρ constant is set higher for the BMU (ρ_b) than for its topological neighbors (ρ_n). Given an input data sample $\mathbf{x}(t)$, if no new neurons are added, the weights of the first BMU and its neighbors are updated as follows:

$$\Delta \mathbf{w}_i = \epsilon_i \cdot h_i \cdot (\mathbf{x}(t) - \mathbf{w}_i) \quad (4)$$

where ϵ_i and h_i are the constant learning rate and the firing counter variable, respectively. The learning of the GWR algorithm stops when a given criterion is met, e.g., a maximum network size or a maximum number of learning epochs.

C. Temporal Sequence Representations

GWR networks do not encode temporal relationships of the input. This limitation has been addressed by different extensions, such as hierarchies of GWRs augmented with a *window in time* memory or recurrent connections [16], [17], [41], [42].

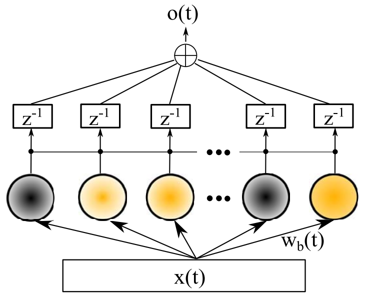


Fig. 2. Schematic of the output computed by each layer of our learning architecture (not all neurons and connections are shown). Given an input data sample $\mathbf{x}(t)$, the weight of the BMU is concatenated with the weights of the previously activated neurons (depicted in fading yellow) in order to compute the output $\mathbf{o}(t)$. The length of the concatenation vector is a constant τ ($\tau = 3$ in this example). The z^{-1} blocks denote the time delay.

Since our goal is to both encode data sequences and to generate them, we adopt the first approach in which relevant information regarding data samples in a given time window is always explicitly available. Also, in contrast to the self-organizing networks equipped with Gamma filters [42], the time-window technique does not introduce additional computational complexity and does not affect the GWR learning dynamics. Moreover, the use of a hierarchy of GWR networks allows for the encoding of multiple time-varying sequences through prototype neurons which can be reused for representing different sequences.

The GWR learning mechanism described in Section III-B is employed for training the first two layers of the proposed architecture, namely the GWR$_1$ and GWR$_2$ networks. The output of both networks is computed as the concatenation of the weights of consecutively activated neurons within a predefined temporal window τ (see Fig. 2)

$$\mathbf{o}(t) = \mathbf{w}_{b(t)} \oplus \mathbf{w}_{b(t-1)} \oplus \cdots \oplus \mathbf{w}_{b(t-\tau+1)} \tag{5}$$

where $\oplus$ denotes the concatenation operator. Moving up in the hierarchy, the output $\mathbf{o}(t)$ will represent the input for the GWR network of the higher layer. In this way, the GWR$_1$ network learns a dictionary of prototypes of the spatial body configurations domain, while the GWR$_2$ and *P-GWR* networks encode body motion patterns accumulated over a short and a relatively longer time period, respectively.

Following this hierarchical learning scheme, we adapt the GWR neuron elimination strategy in a layer-wise manner to address the problem of forgetting rarely encountered, but still relevant, information. For instance, at the level of the GWR$_1$ network which represents spatial body configurations, it is more probable that rarely seen input data samples are due to sensory noise. Therefore, we can set a lower edge age threshold here, leading to a higher rate of neuron elimination. For the GWR$_2$ and *P-GWR* networks, on the other hand, rarely seen data samples are most likely due to subsequences encountered in the far past. We can set a relatively higher edge age threshold so that neurons are removed more rarely.

D. Predictive GWR Algorithm

The problem of one-step-ahead prediction can be formalized as a function approximation problem. Given a multidimensional time series denoted by $\{\mathbf{y}(t)\}$, the function approximation is of the form

$$\hat{\mathbf{y}}(t+1) = \hat{f}(\mathbf{y}(t), \mathbf{y}(t-1), \ldots, \mathbf{y}(t-(p-1))|\Theta) \tag{6}$$

where the input of the function, or *regressor*, has an order of regression $p \in \mathbb{Z}^+$, with Θ denoting the vector of adjustable parameters of the model and $\mathbf{y}(t+1)$ is the predicted value. We adapt the GWR learning algorithm in order to implement this input–output mapping and apply this learning algorithm to the last layer of our architecture, i.e., to the *P-GWR* network.

The input samples fed to the *P-GWR* network are concatenations of the temporally ordered BMUs from the preceding layer (5). We divide the input into two parts: 1) the regressor, $\mathbf{x}^{\text{in}}(t)$ and 2) the desired output, i.e., the value to predict $\mathbf{x}^{\text{out}}(t)$

$$\begin{aligned}\mathbf{x}^{\text{in}}(t) &= \mathbf{x}(t) \oplus \mathbf{x}(t-1) \oplus \cdots \oplus \mathbf{x}(t-p+1)\\ \mathbf{x}^{\text{out}}(t) &= \mathbf{x}(t+1)\end{aligned} \tag{7}$$

with p denoting the number of the past values. Each neuron of the *P-GWR* network will then have two weight vectors which we will call the input $\mathbf{w}^{\text{in}}$ and the output $\mathbf{w}^{\text{out}}$ weight vectors. During training, the input weight vector will learn to represent the input data regressor and the output weight vector will represent the corresponding predicted value. This learning scheme has been successfully applied to the vector-quantized temporal associative memory (VQTAM) model [20], shown to perform well on tasks such as time series prediction and predictive control [43].

The learning procedure for the Predictive GWR algorithm resembles the original GWR with a set of adaptations for temporal processing. During training, the first and the second BMUs at time step t, b, and s, are computed considering only the regressor part of the input

$$\begin{aligned}b &= \arg\min_{n\in A} ||\mathbf{x}^{\text{in}}(t) - \mathbf{w}_n^{\text{in}}||\\ s &= \arg\min_{n\in A/\{b\}} ||\mathbf{x}^{\text{in}}(t) - \mathbf{w}_n^{\text{in}}||\end{aligned} \tag{8}$$

where $\mathbf{w}_n^{\text{in}}$ is the input weight vector of the neuron n and A is the set of all neurons. However, for the weight updates both $\mathbf{x}^{\text{in}}(t)$ and $\mathbf{x}^{\text{out}}(t)$ are considered

$$\begin{aligned}\Delta\mathbf{w}_i^{\text{in}} &= \epsilon_i \cdot c_i \cdot \left(\mathbf{x}^{\text{in}}(t) - \mathbf{w}_i^{\text{in}}\right)\\ \Delta\mathbf{w}_i^{\text{out}} &= \epsilon_i \cdot c_i \cdot \left(\mathbf{x}^{\text{out}}(t) - \mathbf{w}_i^{\text{out}}\right)\end{aligned} \tag{9}$$

with the learning rates $0 < \epsilon_i < 1$ being higher for the BMUs (ϵ_b) than for the topological neighbors (ϵ_n), as in the GWR algorithm. This learning mechanism guarantees that the regressor space is vector quantized while the prediction error (P.E) is decreased at each learning iteration.

The predictive GWR algorithm operates differently from supervised prediction approaches. In the latter, the P.E signal guides the learning, whereas in the predictive GWR the P.E is implicitly computed and minimized without affecting the learning dynamics. Moreover, unlike the SOM-based VQTAM model, the number of input–output mapping neurons, or *local models*, is not predefined nor fixed, but instead adapts to the input data.

E. Predicting Sequences

Given an input regressor at time step t, $\mathbf{x}^{\text{in}}(t)$, the one-step-ahead estimate is defined as the output weight vector of the *P-GWR* BMU

$$\hat{\mathbf{y}}(t+1) = \mathbf{w}_b^{\text{out}} \tag{10}$$

where b is the index of the BMU (8). In the case that the desired prediction horizon is greater than 1, the multistep-ahead prediction can be obtained by feeding back the predicted values into the regressor and computing (8) recursively until the whole desired prediction vector is obtained. An alternative to the recursive prediction is the vector prediction which is obtained by increasing the dimension of the $\mathbf{x}^{\text{out}}$ vector with as many time steps as the desired prediction horizon h. Thus, the input regressor and the desired output would have the following form:

$$\begin{aligned}\mathbf{x}^{\text{in}}(t) &= \mathbf{x}(t) \oplus \mathbf{x}(t-1) \oplus \cdots \oplus \mathbf{x}(t-p+1)\\ \mathbf{x}^{\text{out}}(t) &= \mathbf{x}(t+1) \oplus \mathbf{x}(t+2) \oplus \cdots \oplus \mathbf{x}(t+h)\end{aligned} \tag{11}$$

where p denotes the index of the past values. The same dimensionality should be defined for the weight vectors $\mathbf{w}^{\text{in}}$ and $\mathbf{w}^{\text{out}}$ of the *P-GWR* neurons as well. This solution requires the training of the architecture with this setting of the weights.

IV. Imitation Scenario

A. Overview

The scenario consists of a Nao robot incrementally learning a set of visually demonstrated body motion patterns and simultaneously imitating them while compensating for the sensorimotor delay. We showcase the predictive capabilities of the proposed architecture in the context of an imitation scenario motivated by the fact that it can potentially imply behavior synchronization in the HRI. For humans, the synchronization of behavior is a fundamental principle of motor coordination and is known to increase rapport in daily social interaction [4]. Psychological studies have shown that, during a conversation, humans tend to coordinate body posture and gaze direction [44]. This phenomenon is believed to be connected to the mirror neuron system [45], suggesting a common neural mechanism for both motor control and action understanding. Interpersonal coordination is an integral part of human interaction. Thus, we assume that applied to HRI scenarios it may promote the social acceptance of robots.

A schematic description of the proposed system is given in Fig. 1. The users' body motion is the input of the model and the motor commands for the robot are obtained by mapping the users' arm skeletal configuration to the robot's arm joint angles. This direct motion transfer allows for a simple, yet compact representation of the visuomotor states, that does not require the application of computationally expensive inverse kinematics algorithms. Demonstrated motion trajectories are learned incrementally by training our hierarchical GWR-based algorithm. This allows for extracting prototypic motion patterns which can be used for the generation of robot movements as well as the prediction of future target trajectories in parallel. In this robot task, the prediction of future visuomotor states is necessary to compensate for the sensory delay introduced by the vision sensor, the signal transmission delay as well as the robot's motor latency during motion generation. The simulated Nao robot is used as the robotic platform for the experimental evaluation.

B. System Description

A general overview of the proposed architecture is depicted in Fig. 1. The system consists of three main modules.

1) The *vision module*, which includes the depth sensor and the tracking of the 3-D skeleton through OpenNI/NITE framework.[1]
2) The *visuomotor learning* module, which receives angle values and provides future motor commands.
3) The *robot control* module, which processes motor commands and relays them to the microcontrollers of the robot, which in our case is a locally simulated Nao.

Our contribution is the visuomotor learning module which performs incremental adaptation and early prediction of human motion patterns. Although the current setup uses a simulated environment, we will consider a further extension of the experiments toward the real robot. Therefore, we simulate the same amount of motor response latency as it has been quantified in the real Nao robot, i.e., between 30 and 40 ms [2]. This latency could be even higher due to reduced motor performance, friction, or weary hardware. Visual sensor latency for an RGB and depth resolution of 640×480, together with the computation time required from the skeleton estimation middleware, can peak up to 500 ms [46]. Taking into consideration also possible transmission delays due to connectivity issues, we assume a maximum of 600 ms of overall sensorimotor latency for our experiments described in Section V.

C. Data Acquisition and Representation

The motion sequences were collected with an Asus Xtion Pro camera at 30 frames/s. This type of sensor is capable of providing synchronized color information and depth maps at a reduced power consumption and weight, making it a more suitable choice than a Microsoft Kinect for being placed on our small humanoid robot. Moreover, it offers an efficient and markerless body tracking method [47] which makes the interface less invasive. The distance of each participant from the visual sensor was maintained between the sensor's operational range, i.e., 0.8–3.5 m. To attenuate noise, we computed the median value for each body joint every three frames resulting in ten joint position vectors per second [16].

We encode the demonstrator's postures as vectors of joint angles. The latter allows a straightforward reconstruction of the regressed motion without applying inverse kinematics, which may be difficult due to redundancy and leads to less natural movements. Nao's arm kinematic configuration differs from the human arm in terms of degrees of freedom (DoF). For instance, the shoulder and the elbow joints have only two DoFs while human arms have three. For this reason, we compute only shoulder *pitch* and *yaw* and elbow *yaw* and

[1]OpenNI/NITE: http://www.openni.org/software.

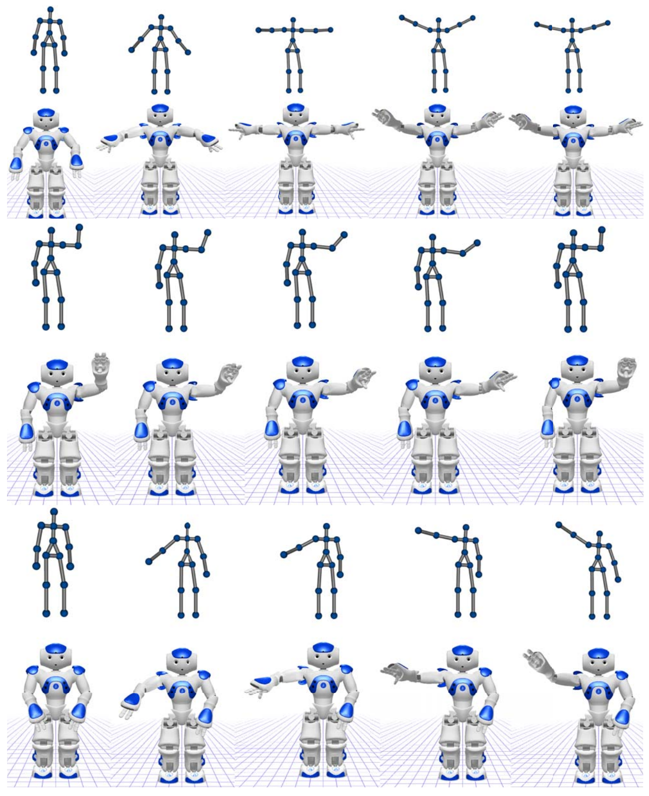


Fig. 3. Examples of arm movement patterns. The visual input data, represented as 3-D skeleton sequences, are mapped to the robots' joint angles.

roll from the skeletal representation by applying trigonometric functions and map them to the Nao's joints by appropriate rotation of the coordinate frames. Wrist orientations are not considered since they are not provided by the OpenNI/NITE framework. Considering the two arms, a frame contains a total of 8 angle values of body motion, which are given as input to the visuomotor learning module.

V. Experimental Results

We conducted experiments with a set of movement patterns that were demonstrated either with one or with both arms simultaneously: raise arm(s) laterally, raise arm(s) in front, wave arm(s), and rotate arms in front of the body both clockwise and counter-clockwise. Examples of the movement patterns are illustrated in Fig. 3. In total, ten different motion patterns were obtained, each repeated ten times by three participants (one female and two male) which were given no explicit indication of the purpose of the study nor instructions on how to perform the arm movements. In total, we obtained 30 demonstrations for each of the patterns. We first describe the incremental training procedure, then we assess and analyze in detail the prediction accuracy of the proposed learning method. We focus on the learning capabilities of the method while simulating a possible recurring malfunctioning of the visual system leading to loss of entire data chunks. We conclude with a model for choosing the optimal predicted value for a system with a variable delay.

TABLE I
Training Parameters for Each GWR Network in Our Architecture for the Incremental Learning of Sensorimotor Patterns

Parameter	Value
Activation Threshold	$a_T = 0.98$
Firing Threshold	$f_T = 0.1$
Learning rates	$\epsilon_b = 0.1, \epsilon_n = 0.01$
Firing counter behavior	$\rho_b = 0.3, \rho_n = 0.1, \kappa = 1.05$
Maximum edge age	{100, 200, 300}
Training epochs	50

A. Hierarchical Training

The training of our architecture is carried out in an online manner. This requires that the GWR networks are trained sequentially one data sample at a time. The networks are initialized with two neurons with random weight vectors. The GWR$_1$ network is trained in order to perform spatial vector quantization. Then the current sequence is gradually encoded as a trajectory of activated neurons as described in (5) and given in input to the GWR$_2$ network of the second layer. The same procedure is then repeated for the second layer until the training of the full architecture is performed. The learning of the 30 demonstrations of one motion pattern from all three subjects constitutes one *training epoch*.

The learning parameters used throughout our experiments are listed in Table I. The parameters have been empirically fine-tuned by considering the learning factors of the GWR algorithm. The firing threshold f_T and the parameters ρ_b, ρ_n, and κ define the decreasing function curve of the firing counter (3) and were set in order to have at least seven trainings of a BMU before inserting a new neuron. It has been shown that increasing the number of trainings per neuron does not affect the performance of a GWR network significantly [15]. In the GWR algorithm, the learning rates are generally chosen to yield faster training for the BMUs than for their topological neighbors. However, given that the neurons' decreasing firing counter modulates the weights update (4), an optimal choice of the learning rates has little impact on the architecture's behavior in the long term. The training epochs were chosen by analyzing the converging behavior of the composing GWR networks in terms of neural growth.

The activation threshold parameter a_T, which modulates the number of neurons, has the largest impact on the architecture's behavior. The closer to 1 this value is, the greater is the number of neurons created and the better is the data reconstruction during the prediction phase. Therefore, we kept a_T relatively high for all GWR networks. We provide an analysis of the impact of this parameter on the prediction performance of our architecture in Section V-B3. Finally, the maximum edge age parameter, which modulates the removal of rarely used neurons, was set increasingly high with each layer. As assumed in Section III-C, the neurons activated less frequently in the lower layer may be representing noisy input data samples, whereas in higher layers the neurons capture spatiotemporal dependencies which may vary significantly from sequence to sequence.

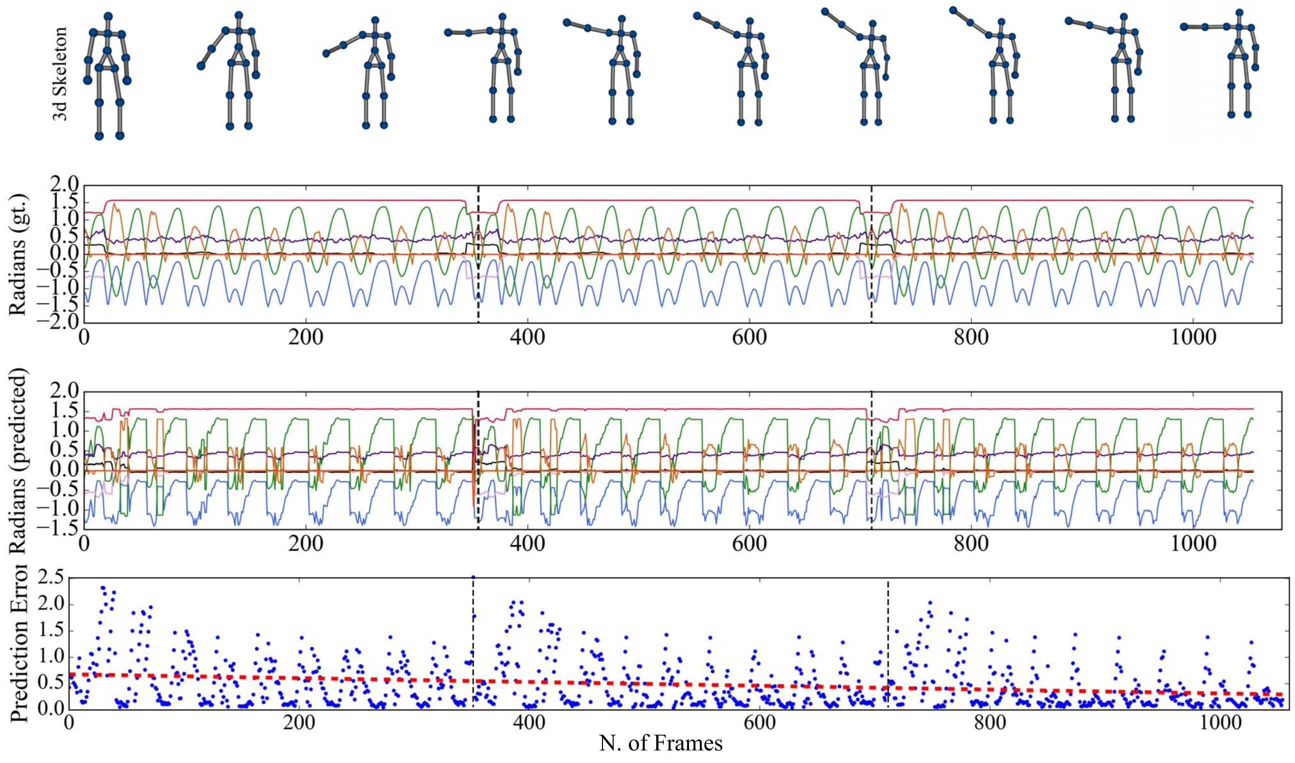


Fig. 4. Behavior of the proposed architecture during training on an *unseen* sequence demonstrated by one subject (three iterations over one sequence are shown). From top to bottom illustrated are: the skeleton model of the visual sequence, the ground truth data of robot joint angles, the values predicted from the network, and the Euclidean distance between predicted values and the ground truth over time (red dashed line indicating the statistical trend).

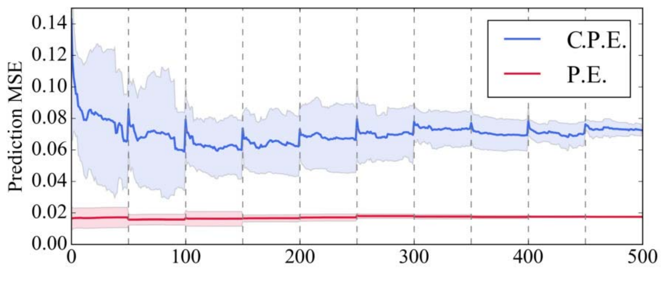
B. Predictive Behavior

We now assess the predictive capabilities of the proposed method while the training is occurring online. Considering that the data sample rate is 10 frames/s (see Section IV-C), we set a prediction horizon of six frames in order to compensate for the estimated delay of 600 ms.

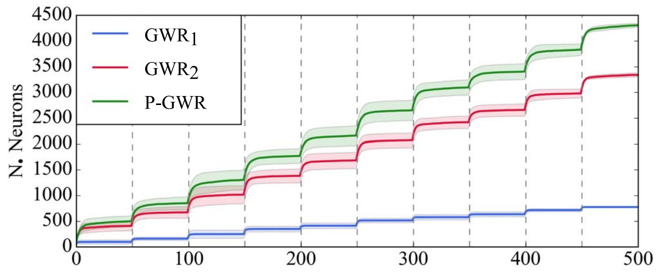
1) How Fast Does the Architecture Adapt to New Sequence?: An example of the online response of the architecture is shown in Fig. 4. We observed that, except in cases of highly noisy trajectories, the network adapted to an unseen input already after a few video frames, e.g., $\approx$ 100 frames which correspond to 10 s of the video sequence, and refined its internal representation after three iterations over the motion sequence demonstrated by one subject, i.e., after 30 demonstrations. This can be seen by the statistical trend of the P.E.

2) Behavior Analysis and Prediction Performance During Incremental Learning: We presented the movement sequences one at a time and let the architecture train for 50 epochs on each new sequence. The training phase was a total of 500 epochs for the whole dataset. Then, we reran the same experiment by varying the presentation order of the sequences and report the results averaged across all trials. In this way, the behavior analysis does not depend on the order of the data given during training. We analyzed the cumulative P.E (C.P.E) of the model by computing the mean squared error (MSE) over all movement sequences learned up to each training epoch. For comparison, we also computed the MSE between the values predicted by the model and the sensory input after being processed by the GWR$_1$ and the GWR$_2$ networks. We refer to this performance measure as the P.E since it evaluates directly the prediction accuracy of the *P-GWR* network while removing the quantization error propagated from the first two layers.

The flow of the overall MSE during training and the neural growth of the GWR networks composing the architecture are reported in Fig. 5. The moment in which we introduce a new motion sequence is marked by a vertical dashed line. As expected, the C.P.E increases as soon as a new sequence is introduced [leading to the high peaks in Fig. 5(a)], for then decreasing immediately. However, the error does not grow but remains constant even though new knowledge is being added every 50 learning epochs. This is a desirable feature for an incremental learning approach. In Fig. 5(b), we observe that with the introduction of a new motion sequence there is an immediate neural growth of the three GWR networks followed by the stabilization of the number of neurons indicating a fast convergence. This neural growth is an understandable consequence of the fact that the movement sequences are very different from each other. In fact, the GWR$_1$ network, performing quantization of the spatial domain, converges to a much lower number of neurons, whereas the higher layers, namely the GWR$_2$ and the *P-GWR* network, have to capture a high variance of spatiotemporal patterns. However, the computational complexity of a prediction step is $O(n)$, where n is the number of neurons. Thus, the growth of the network does not introduce significant computational cost.



(a)



(b)

Fig. 5. (a) C.P.E averaged over all learned sequences up to each learning epoch (in blue) and the P.E computed between the predicted sequence and the sequence represented by the architecture (in red). (b) Average and standard deviation of the neural growth of the three GWR networks during learning.

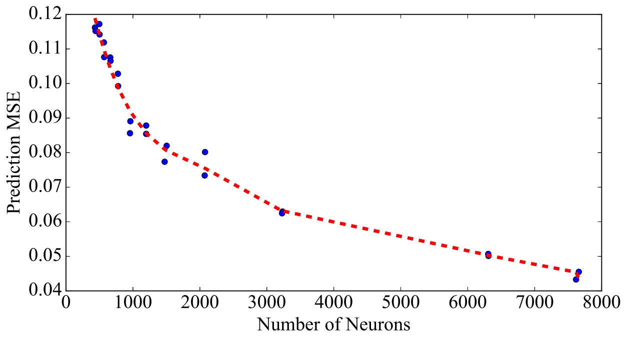


Fig. 6. Prediction MSE versus the number of neurons in the *P-GWR* network.

3) Impact of Activation Threshold: In the described experiments, we set a relatively high activation threshold parameter a_T which led to a continuous growth of the GWR networks. Thus, we further investigated how a decreased number of neurons in the *P-GWR* network would affect the overall P.E. For this purpose, we fixed the weight vectors of the first two layers after having been trained on the entire dataset, and ran multiple times the incremental learning procedure on the *P-GWR* network, each time with a different activation threshold parameter $a_T \in \{0.5, 0.55, 0.6, \ldots, 0.9, 0.95, 0.99\}$. We observed that a lower number of neurons, obtained through lower threshold values, led to quite high values of the MSE (Fig. 6). However, due to the hierarchical structure of our architecture, the quantization error can be propagated from layer to layer. It is expected that similar performances can be reproduced with a lower number of neurons in the *P-GWR* network when a lower quantization error is obtained in the preceding layers.

4) Sensitivity to Prediction Horizon: We now take the architecture trained on the whole dataset and evaluate its prediction accuracy while increasing the prediction horizon up to 20 frames, which correspond to 2 s of a video sequence. For achieving multistep-ahead prediction, we compute the predicted values recursively as described in Section III-E. In Fig. 7, we report the mean absolute error and the standard deviation in radians in order to give a better idea of the error range. The results show a relatively higher magnitude of error for prediction horizons bigger than ten frames. This should come as no surprise since producing accurate long-term predictions is a challenging task when dealing with human-like motion sequences. However, it seems that on average the error does not grow linearly but remains under 0.25 radians.

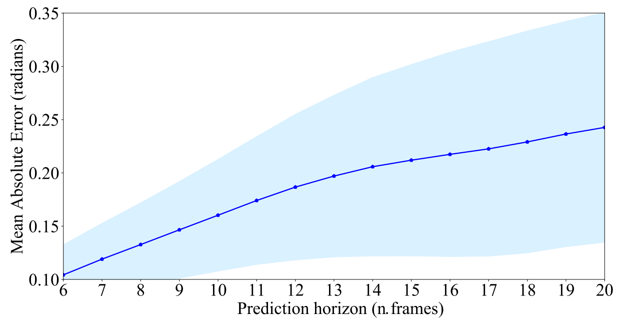


Fig. 7. Mean absolute error (in radians) for increasing values of prediction horizons (expressed in frames). In our case, 20 frames correspond to 2 s of a video sequence.

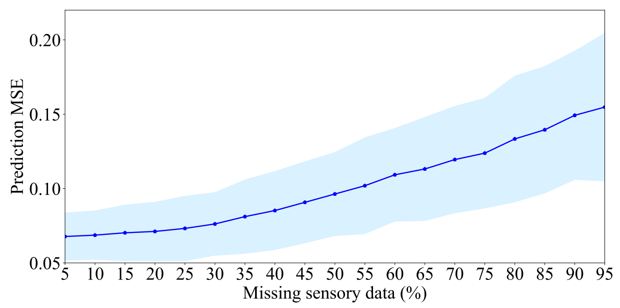


Fig. 8. Prediction MSE averaged over 50 epochs of training on each motion pattern. Up to 30% of data loss, the MSE does not grow linearly but rather stays mostly constant. From this point on, the increasing percentage of data loss leads to an inevitable growth of the P.E.

C. Learning With Missing Sensory Data

In the following experiment, we analyze how the predictive performance of the network changes when trained on input data produced by a faulty visual sensor. We simulate an occurring loss of entire input data chunks in the following way: during the presentation of a motion pattern, we randomly choose video frames where a whole second of data samples (i.e., ten frames) is eliminated. The network is trained for 50 epochs on a motion sequence, each time with a different missing portion of information. We repeat the experiment thereby increasing the occurrence of this event in order to compromise up to 95% of the data and see how much the overall P.E increases. Results are averaged over epochs and are presented in Fig. 8. As it can be seen, the prediction MSE stays mostly constant up to 30% of data loss. This means that the network can still learn and predict motion sequences even under such circumstances.

D. Compensating Variable Delay

Experimental results reported so far have accounted for compensating a fixed time delay which has been measured empirically by generating motor behavior with the real robot. However, the proposed architecture can also be used when the delay varies due to changes in the status of the hardware. In this case, given the configuration of the robot at time step t in terms of joint angle values $J_\xi(t)$, where ξ is the time delay estimation, the optimal predicted angle values to execute in the next step can be chosen in the following way:

$$P^* = \arg\min_{i\in[0,h]} ||J_\xi(t) - P(t+i)|| \tag{12}$$

where $P(t+i)$ are the predictions computed up to a maximum h of the prediction horizon.

The application of this prediction step requires a method for the estimation of the time-delay ξ, which is out of the scope of this paper. Current time-delay estimation techniques mainly cover constant time delays, random delay with a specific noise characteristic, or restricted dynamic time delay [48], which nonetheless do not address uncertainty affecting real-world robot applications. Computational models inspired by biology have also been proposed for the time-delay estimation [48]. However, these models assume knowledge of the sensorimotor dynamics.

VI. Discussion

A. Summary

In this paper, we presented a self-organized hierarchical neural architecture for sensorimotor delay compensation in robots. In particular, we evaluated the proposed architecture in an imitation scenario, in which a simulated robot had to learn and reproduce visually demonstrated arm movements. Visuomotor sequences were extracted in the form of joint angles, which can be computed from a body skeletal representation in a straightforward way. Sequences generated by multiple users were learned using hierarchically arranged GWR networks equipped with an increasingly large temporal window.

The prediction of the visuomotor sequences was obtained by extending the GWR learning algorithm with a mapping mechanism of input and output vectors in the spatiotemporal domain. We conducted experiments with a dataset of ten arm movement sequences showing that our system achieves low P.E values on the training data and can adapt to unseen sequences in an online manner. Experiments also showed that a possible system malfunction causing loss of data samples has a relatively low impact on the overall performance of the system.

B. Growing Self-Organization and Hierarchical Learning

The building block of our architecture is the GWR network [15], which belongs to the unsupervised competitive learning class of artificial neural networks. A widely known algorithm of this class is the SOM [21]. The main component of these algorithms are the neurons equipped with weight vectors of dimensionality equal to the input size. Through learning, the neurons become prototypes of the input space while preserving the input's topological features, i.e., similar inputs are mapped to neurons that are near to each other. In the case of SOMs, these neurons are distributed and remain fixed in a 2-D or a 3-D lattice which has to be defined *a priori* and requires an optimal choice of its size. In the GWR network, the topological structure of the neurons is dynamic and grows to adapt to the topological properties of the input space. In this regard, the GWR network is similar to the GNG algorithm [49], another widely used growing self-organizing neural network. However, the neural growth of the GWR algorithm is not constant, as in the case of the GNG, but rather depends on how well the current state of the network represents the input data. Thus, from the perspective of incremental learning, the GWR algorithm is more suitable than the GNG since new knowledge can be added to the network as soon as new data become available.

The hierarchical arrangement of the GWR networks equipped with a window in time memory is appealing due to the fact that it can dynamically change the topological structure in an unsupervised manner and learn increasingly more complex spatiotemporal dependencies of the input data. This allows for reuse of the neurons during sequence encoding, having learned prototypical spatiotemporal patterns out of the training sequences. Although this approach seems to be quite resource-efficient for the task of learning visuomotor sequences, the extent to which neurons are reused is tightly coupled with the input distribution. In fact, in our experiments with input data samples represented as multidimensional vectors of both arms' shoulder and elbow angles, there was little to no overlap among training sequences. This led to significant growth of the network with each iteration over unseen sequences.

The parameters modulating the growth rate of each GWR network are the activation threshold and the firing counter threshold. The activation threshold a_T establishes the maximum discrepancy between the input and the prototype neurons in the network. The larger we set the value of this parameter, the smaller is the discrepancy, i.e., the quantization error of the network. The firing counter threshold f_T is used to ensure the training of recently added neurons before creating new ones. Thus, smaller thresholds lead to more training of existing neurons and the slower creation of new ones, favoring better network representations of the input. Intuitively, the less discrepancy between the input and the network representations, the smaller the input reconstruction error during the prediction phase. However, less discrepancy means also more neurons. This proved to be not the main issue in our experiments since the number of neurons did not affect significantly the computational complexity of the predicted values.

A limitation of the sliding time-window technique for the encoding of temporal sequences is the high computational cost it introduces due to data's higher dimensionality. However, in our case using angles as body pose features leads to a low-dimensional input compared to, e.g., images.

So, the training with long time windows does not pose a computational challenge. Furthermore, it has been shown that long-term predictions based on a sliding window are more accurate than recurrent approaches [50].

The use of joint angles as visuomotor representations may seem to be a limitation of the proposed architecture due to the fact that it requires sensory input and robot actions to share the same representational space. For instance, in an object manipulation task, this requirement is not satisfied, since the visual feedback would be the position given by the object tracking algorithm. This issue can be addressed by including both position information and corresponding robot joint angles as input to our architecture. Due to the generative nature of self-organizing networks and their capability to function properly when receiving an incomplete input pattern, only the prediction of the object movement patterns would trigger the generation of corresponding patterns of the robot behavior.

C. Future Work

An interesting direction for future work is the extension of the current implementation toward the autonomous generation of robot movements that account for both delay compensation as well as reaching a given action goal. For this purpose, the implementation of bidirectional Hebbian connections would have to be investigated in order to connect the last layer of the proposed architecture with a symbolic layer containing action labels [16], [51] and explore how such symbolic layer can modulate the generation of the movement patterns when diverging from the final goal.

Future studies with the real robot will address the introduction of overall body configuration constraints for learning the perceived motion. The visual body tracking framework becomes unreliable in certain conditions, e.g., when the demonstrator is sitting or is touching objects in the background. In these cases, the provided body configurations may become unrealistic and cannot be mapped to the robot, or, in the worst case, when mapped to the robot may lead to hardware break. For this reason, outlier detection mechanisms should be investigated in order to discard these unrealistic body configurations during training.

The imitation scenario studied in this paper was carried out offline, i.e., the synchronization was evaluated on an acquired data set of motion patterns. Future experiments will comprise and HRI user study in which participants will be able to teach the motion patterns directly to the robot. Moreover, the current results encourage further experiments toward *learning by demonstration* scenarios, whereby demonstrated motion patterns are stored and then recalled for the execution of different tasks with a robotic platform.

References

[1] J. Mainprice, M. Gharbi, T. Siméon, and R. Alami, "Sharing effort in planning human-robot handover tasks," in *Proc. IEEE RO-MAN*, 2012, pp. 764–770.

[2] J. Zhong, C. Weber, and S. Wermter, "A predictive network architecture for a robust and smooth robot docking behavior," *Paladyn*, vol. 3, no. 4, pp. 172–180, 2012.

[3] R. Saegusa, F. Nori, G. Sandini, G. Metta, and S. Sakka, "Sensory prediction for autonomous robots," in *Proc. IEEE-RAS Humanoid Robots*, Pittsburgh, PA, USA, 2007, pp. 102–108.

[4] T. Lorenz, A. Mörtl, B. Vlaskamp, A. Schubö, and S. Hirche, "Synchronization in a goal-directed task: Human movement coordination with each other and robotic partners," in *Proc. IEEE RO-MAN*, Atlanta, GA, USA, 2011, pp. 198–203.

[5] A. Bahill, "A simple adaptive Smith-predictor for controlling time-delay systems: A tutorial," *IEEE Control Syst. Mag.*, vol. CSM-3, no. 2, pp. 16–22, May 1983.

[6] A. Gloye *et al.*, "Predicting away robot control latency," Free Univ. Berlin, Berlin, Germany, Rep. B-08-03, Jun. 2003.

[7] R. Nijhawan and S. Wu, "Compensating time delays with neural predictions: Are predictions sensory or motor?" *Philosoph. Trans. Royal Soc. London A Math. Phy. Eng. Sci.*, vol. 367, no. 1891, pp. 1063–1078, 2009.

[8] R. C. Miall, D. J. Weir, D. M. Wolpert, and J. F. Stein, "Is the cerebellum a Smith predictor?" *J. Motor Behav.*, vol. 25, no. 3, pp. 203–216, 1993.

[9] D. Kerzel and K. R. Gegenfurtner, "Neuronal processing delays are compensated in the sensorimotor branch of the visual system," *Current Biol.*, vol. 13, no. 22, pp. 1975–1978, 2003.

[10] M. Rohde, L. C. van Dam, and M. O. Ernst, "Predictability is necessary for closed-loop visual feedback delay adaptation," *J. Vis.*, vol. 14, no. 3, p. 4, 2014.

[11] C. de la Malla, J. López-Moliner, and E. Brenner, "Dealing with delays does not transfer across sensorimotor tasks," *J. Vis.*, vol. 14, no. 12, p. 8, 2014.

[12] J. Mainprice and D. Berenson, "Human-robot collaborative manipulation planning using early prediction of human motion," in *Proc. IEEE/RSJ IROS*, 2013, pp. 299–306.

[13] M. Ito and J. Tani, "On-line imitative interaction with a humanoid robot using a mirror neuron model," in *Proc. IEEE ICRA*, vol. 2. New Orleans, LA, USA, 2004, pp. 1071–1076.

[14] S. Levine, P. Pastor, A. Krizhevsky, and D. Quillen, "Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection," in *Proc. Int. Symp. Exp. Robot. (ISER)*, 2016, pp. 173–184.

[15] S. Marsland, J. Shapiro, and U. Nehmzow, "A self-organising network that grows when required," *Neural Netw.*, vol. 15, nos. 8–9, pp. 1041–1058, 2002.

[16] G. I. Parisi, J. Tani, C. Weber, and S. Wermter, "Emergence of multimodal action representations from neural network self-organization," *Cogn. Syst. Res.*, vol. 43, pp. 208–221, Jun. 2016.

[17] L. Mici, G. I. Parisi, and S. Wermter, "Recognition of transitive actions with hierarchical neural network learning," in *Proc. ICANN*, 2016, pp. 472–479.

[18] D. Vasquez, T. Fraichard, O. Aycard, and C. Laugier, "Intentional motion on-line learning and prediction," *Mach. Vis. Appl.*, vol. 19, nos. 5–6, pp. 411–425, 2008.

[19] A. M. Schaefer, S. Udluft, and H.-G. Zimmermann, "Learning long-term dependencies with recurrent neural networks," *Neurocomputing*, vol. 71, nos. 13–15, pp. 2481–2488, 2008.

[20] G. A. Barreto, "Time series prediction with the self-organizing map: A review," in *Perspectives of Neural-Symbolic Integration*. Berlin, Germany: Springer-Verlag, 2007, pp. 135–158.

[21] T. Kohonen, *Self-Organization and Associative Memory*. Heidelberg, Germany: Springer-Verlag, 1993.

[22] G. Simon, J. A. Lee, M. Cottrell, and M. Verleysen, "Forecasting the CATS benchmark with the double vector quantization method," *Neurocomputing*, vol. 70, nos. 13–15, pp. 2400–2409, 2007.

[23] N. Johnson and D. Hogg, "Learning the distribution of object trajectories for event recognition," *Image Vis. Comput.*, vol. 14, no. 8, pp. 609–615, 1996.

[24] N. Sumpter and A. Bulpitt, "Learning spatio-temporal patterns for predicting object behaviour," *Image Vis. Comput.*, vol. 18, no. 9, pp. 697–704, 2000.

[25] W. Hu, D. Xie, T. Tan, and S. Maybank, "Learning activity patterns using fuzzy self-organizing neural network," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 34, no. 3, pp. 1618–1626, Jun. 2004.

[26] J. Walter, H. Riter, and K. Schulten, "Nonlinear prediction with self-organizing maps," in *Proc. IEEE IJCNN*, San Diego, CA, USA, 1990, pp. 589–594.

[27] J. Vesanto, "Using the SOM and local models in time-series prediction," in *Proc. Workshop Self Organizing Maps*, 1997, pp. 209–214.

[28] G. Billard, S. Calinon, and R. Dillmann, "Learning from humans," in *Handbook of Robotics*, 2nd ed. Secaucus, NJ, USA: Springer, 2016, pp. 1995–2014.

[29] D. Kulić, W. Takano, and Y. Nakamura, "Incremental learning, clustering and hierarchy formation of whole body motion patterns using adaptive hidden Markov Chains," *Int. J. Robot. Res.*, vol. 27, no. 7, pp. 761–784, 2008.

[30] D. Kulić, C. Ott, D. Lee, J. Ishikawa, and Y. Nakamura, "Incremental learning of full body motion primitives and their sequencing through human motion observation," *Int. J. Robot. Res.*, vol. 31, no. 3, pp. 330–345, 2012.

[31] W. Takano and Y. Nakamura, "Humanoid robot's autonomous acquisition of proto-symbols through motion segmentation," in *Proc. IEEE-RAS Humanoid Robots*, 2006, pp. 425–431.

[32] A. G. Billard, S. Calinon, and F. Guenter, "Discriminative and adaptive imitation in uni-manual and bi-manual tasks," *Robot. Autonom. Syst.*, vol. 54, no. 5, pp. 370–384, 2006.

[33] S. Ekvall, D. Aarno, and D. Kragic, "Online task recognition and real-time adaptive assistance for computer-aided machine control," *IEEE Trans. Robot.*, vol. 22, no. 5, pp. 1029–1033, Oct. 2006.

[34] K. R. Dixon, J. M. Dolan, and P. K. Khosla, "Predictive robot programming: Theoretical and experimental analysis," *Int. J. Robot. Res.*, vol. 23, no. 9, pp. 955–973, 2004.

[35] T. Ogata, S. Sugano, and J. Tani, "Open-end human robot interaction from the dynamical systems perspective: Mutual adaptation and incremental learning," in *Proc. IEA-AIE*, 2004, pp. 435–444.

[36] S. Calinon and A. Billard, "Incremental learning of gestures by imitation in a humanoid robot," in *Proc. ACM/IEEE HRI*, Arlington, VA, USA, 2007, pp. 255–262.

[37] S. M. Khansari-Zadeh and A. Billard, "BM: An iterative algorithm to learn stable non-linear dynamical systems with Gaussian mixture models," in *Proc. IEEE ICRA*, Anchorage, AK, USA, 2010, pp. 2381–2388.

[38] T. Cederborg, M. Li, A. Baranes, and P.-Y. Oudeyer, "Incremental local online Gaussian mixture regression for imitation learning of multiple tasks," in *Proc. IEEE/RSJ IROS*, 2010, pp. 267–274.

[39] G. I. Parisi, S. Magg, and S. Wermter, "Human motion assessment in real time using recurrent self-organization," in *Proc. IEEE RO-MAN*, New York, NY, USA, Aug. 2016, pp. 71–76.

[40] M. A. Giese and T. Poggio, "Neural mechanisms for the recognition of biological movements," *Nature Rev. Neurosci.*, vol. 4, no. 3, pp. 179–192, 2003.

[41] G. I. Parisi, C. Weber, and S. Wermter, "Self-organizing neural integration of pose-motion features for human action recognition," *Front. Neurorobot.*, vol. 9, p. 3, Jun. 2015.

[42] P. A. Estévez and J. R. Vergara, "Nonlinear time series analysis by using gamma growing neural gas," in *Proc. Adv. Self Organizing Maps*, 2013, pp. 205–214.

[43] G. D. A. Barreto, A. F. Araújo, and H. J. Ritter, "Self-organizing feature maps for modeling and control of robotic manipulators," *J. Intell. Robot. Syst.*, vol. 36, no. 4, pp. 407–450, 2003.

[44] K. Shockley, D. C. Richardson, and R. Dale, "Conversation and coordinative structures," *Topics Cogn. Sci.*, vol. 1, no. 2, pp. 305–319, 2009.

[45] G. Tessitore, R. Prevete, E. Catanzariti, and G. Tamburrini, "From motor to sensory processing in mirror neuron computational modelling," *Biol. Cybern.*, vol. 103, no. 6, pp. 471–485, 2010.

[46] M. A. Livingston, J. Sebastian, Z. Ai, and J. W. Decker, "Performance measurements for the microsoft kinect skeleton," in *Proc. IEEE VRW*, Costa Mesa, CA, USA, 2012, pp. 119–120.

[47] J. Han, L. Shao, D. Xu, and J. Shotton, "Enhanced computer vision with microsoft kinect sensor: A review," *IEEE Trans. Cybern.*, vol. 43, no. 5, pp. 1318–1334, Oct. 2013.

[48] A. Sargolzaei, M. Abdelghani, K. K. Yen, and S. Sargolzaei, "Sensorimotor control: Computing the immediate future from the delayed present," *BMC Bioinf.*, vol. 17, no. 7, p. 245, 2016.

[49] B. Fritzke, "A growing neural gas network learns topologies," in *Advances in Neural Information Processing Systems*, vol. 7. Cambridge, MA, USA: MIT-Press, 1995, pp. 625–632.

[50] J. Bütepage, M. Black, D. Kragic, and H. Kjellström, "Deep representation learning for human motion prediction and classification," in *Proc. CVPR*, Jul. 2017.

[51] F. Schrodt and M. V. Butz, "Just imagine! Learning to emulate and infer actions with a stochastic generative architecture," *Front. Robot. AI*, vol. 3, p. 5, Mar. 2016.

Luiza Mici received the B.S. and M.S. degrees in computer engineering from the University of Siena, Siena, Italy. She is currently working toward the Ph.D. degree with the Knowledge Technology Group, University of Hamburg, Hamburg, Germany.

Since 2015, she has been a Research Associate with the Knowledge Technology Group, University of Hamburg, where she was a part of the Research Project Crossmodal Learning. Her current research interests include perception and learning, neural network self-organization, and bio-inspired action recognition.

German I. Parisi received the B.S. and M.S. degrees from the University of Milano-Bicocca, Milan, Italy and the Ph.D. degree from the University of Hamburg, Hamburg, Germany, in 2017, all in computer science.

In 2015, he was a Visiting Researcher at the Cognitive Neuro-Robotics Laboratory, Korea Advanced Institute of Science and Technology, Daejeon, South Korea. Since 2016, he has been a Research Associate of the International Project Transregio TRR 169 on Crossmodal Learning with the Knowledge Technology Institute, University of Hamburg, where he was a part of the Research Project Cognitive Assistive Systems and the International Ph.D. Research Training Group Cross-Modal Interaction in Natural and Artificial Cognitive Systems. His current research interests include neurocognitive systems for human–robot assistance, computational models for multimodal integration, neural network self-organization, and deep learning.

Stefan Wermter received the M.Sc. degree from the University of Massachusetts, Amherst, MA, USA and the Ph.D. (Habilitation) degree from the University of Hamburg, Hamburg, Germany, both in computer science.

He is a Full Professor at the University of Hamburg and the Director of the Knowledge Technology Institute. He has been a Research Scientist at the International Computer Science Institute at Berkeley, Berkeley, CA, USA, before leading the Chair of Intelligent Systems, University of Sunderland, Sunderland, U.K. His current research interests include neural networks, hybrid systems, neuroscience-inspired computing, cognitive robotics, and natural communication.

Dr. Wermter is an Associate Editor of the *Transactions of Neural Networks and Learning Systems*, *Connection Science*, the *International Journal for Hybrid Intelligent Systems and Knowledge*, and *Information Systems*. He is on the Editorial Board of *Cognitive Systems Research*, *Cognitive Computation*, and *Journal of Computational Intelligence*. He has been the General Chair of the International Conference on Artificial Neural Networks 2014. He currently serves as a co-coordinator of the DFG-funded SFB/Transregio International Collaborative Research Center on Crossmodal Learning, and a coordinator of the European Training Network SECURE on Safe Robots.