Efficient exploration and learning of whole body kinematics

Matthias Rolf, Jochen Steil, Michael Gienger

2009

Preprint:

This is an accepted article published in Proceedings of the The 8th International Conference on Development and Learning. The final authenticated version is available online at: https://doi.org/[DOI not available]

Efficient exploration and learning of whole body kinematics

Matthias Rolf, Jochen J. Steil and Michael Gienger

***Abstract*—We present a neural network approach to early motor learning. The goal is to explore the needs for bootstrapping the control of hand movements in a biologically plausible learning scenario. The model is applied to the control of hand postures of the humanoid robot ASIMO by means of full upper body movements. For training, we use an efficient online scheme for recurrent reservoir networks consisting of supervised backpropagation-decorrelation output adaptation and an unsupervised intrinsic plasticity reservoir optimization. We demonstrate that the network can acquire accurate inverse models for the highly redundant ASIMO, applying bi-manual target motions and exploiting all upper body degrees of freedom. We show that very few, but highly symmetric training data is sufficient to generate excellent generalization capabilities to untrained target motions. We also succeed in reproducing real motion recorded from a human demonstrator, massively differing from the training data in range and dynamics. The demonstrated generalization capabilities provide a fundamental prerequisite for an autonomous and incremental motor learning in an developmentally plausible way. Our exploration process – though not yet fully autonomous – clearly shows that goal-directed exploration can, in contrast to "babbling" of joints angles, be done very efficiently even for many degrees of freedom and non-linear kinematic configurations as ASIMOs.**

***Index Terms*—Motor Learning, Neural Networks, Generalization, Humanoid Robots**

I. Introduction

In early childhood, motor learning is an essential and crucial part of development. According to one hypothesis of developmental psychology, which has been explored in recent years, this learning may be based on so called "motor babbling" [1]. In robotics, this concept is mostly picked up as a random exploration of joint angles [2], [3]. On the other hand, it was observed that small children rather act goal directed and direct their movements towards salient objects in their field of view [4]. Different models for motor babbling and early movement learning have therefore been proposed mostly concerned with the acquisition of reaching skills by learning the mapping from hand and/or objects perception to motor variables. In this paper, we investigate hand movements and concentrate on bimanual actions, which involve coupled motion of the whole upper body and – as it turns out – can not be simply combined from learning both arm movements separately.

M. Rolf and J.J. Steil are with the Research Institute for Cognition and Robotics (CoR-Lab), Bielefeld University, Germany, e-mail: {mrolf,jsteil}@CoR-Lab.Uni-Bielefeld.de

Michael Gienger is with the Honda Research Institute Europe, Offenbach, Germany, e-mail: michael.gienger@Honda-RI.de

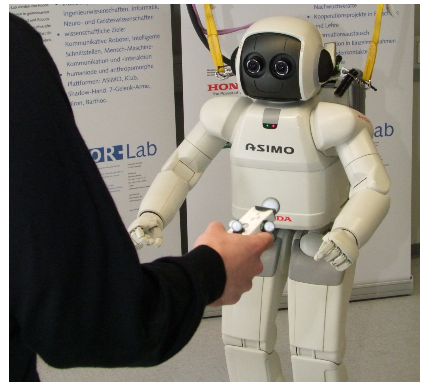


Fig. 1: A user can demonstrate target movements to ASIMO, which are then reproduced with a learned inverse kinematics.

We take an experimental and computational approach to investigate such exploration and learning by means of modelling in an artificial neural network applied to control the motion the humanoid robot ASIMO [5].

To this end, we introduce a very efficient neural learning scheme, which can cope with the typical constraints in developmental learning. There are three major requirements: data have to be processed as perceived (online learning), data are presented in temporal order as they occur in the real world (temporally correlated presentation), and learning needs to be efficient and biologically plausible. Note that the desired combination of online learning and temporal presentation of data rules out most of the standard learning schemes including feedforward multi-layer networks, regression, and other batch schemes.

Neural and biologically plausible learning of motor behavior has been a long standing topic at the border of neuroscience and robotics and inspired a huge amount of research in both domains. It is widely recognized that the cerebellum plays a central role in motor learning (see e.g. the recent review [6]) and it has been argued that forward models using efference copies of motor control signals [7] exist. Though we do not aim at directly modelling a cerebellar network, we argue that the recent approach to recurrent networks known as reservoir computing offers an appealing approach to model motor learning, a connection, which has been explored to some detail in [8]. Basically, the intrinsic architecture of the cerebellum constists of three layers. Evidence from neurophysiology exists that supervised learning takes place in the output layer [6].

The fundamental computational principle of reservoir com-

978-1-4244-4118-1/09/$25.00

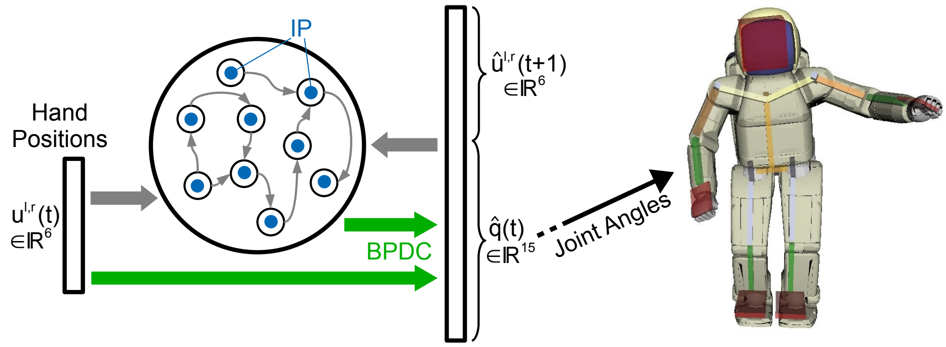


Fig. 2: Target positions of both hands are fed into the recurrent neural network. The network is trained towards an inverse kinematics solution by BPDC adaption of output weights and an Intrinsic Plasticity rule within the reservoir. The estimated joint angles are applied on ASIMO in order to follow a target movement.

puting is to generate a complex nonlinear transformation of the input signal into a high dimensional vector of the recurrent network neuron states. Functionally the state vectors can be regarded as nonlinear temporal feature vectors encoding information about the the inputs, which enable linear learning methods at the output neurons. We will show in this paper that a reservoir network can learn the complex inverse kinematics needed to control hand movements, i.e. we apply the network to an intrinsically static inverse kinematics task, which however becomes temporal by the requirement of correlated presentation of data and online learning.

In related work, learning of the inverse kinematics model as a combination of locally linear low dimensional models has been very successfully demonstrated in [3], [9]. Learning redundancy resolution and disambiguation for following a figure-eight is performed for the humanoid robot at ATR, which shows that locally the inverse model is sufficiently smooth to be approximated by a simple function estimator. The same weighted regression scheme has also been used in [10] to learn goal directed movements based on parameterizing a predefined dynamical system or nonlinear oscillators for rhythmic movements [11]. By definition, these schemes are local and can not extrapolate to untrained regions of the mapping. The approach to learn or design restricted behavior primitives has been criticized in [12], [13], because generalization to new patterns is practically impossible. More recent work applies this approach for imitation learning of simple manipulations with a small torso robot [14]. While our approach also predicts tasks and motor outputs and assumes that sensori-motor patterns are stored in a combined model, it differs in several important aspects from previously introduced schemes. Like in [13], we learn a distributed representation, but rather in a fully recurrent neural network and not using explicit motor inputs. Most important, our learning scheme is completely online and efficient enough to learn while running in a behaving robot. We demonstrate that our network can learn the inverse kinematics with a high degree of accuracy and, if training data are suitable, can generalize to structurally similar motions and even to completely untrained inputs.

II. Task and Neural Network Approach

A. Inverse Kinematics and Body Couplings

Given some target coordinates of end-effectors $u \in \mathbb{R}^n$, an inverse kinematics gives the joint angels $q \in \mathbb{R}^m$ for the robot that apply the end-effector targets. On ASIMO, we control the 3D cartesian positions of both hands by a six-dimensional input variable $u(t) = u^{l,r}(t) \in \mathbb{R}^6, l = left, r = right$. The outputs are the control variables for a total number of 15 degrees of freedom ($m = 15$). Each arm is moved by controlling three rotational degrees of freedom in the shoulder, one in the ellbow and one in the wrist. Therefore each arm is – regarding the task to position the hand – redundant on its own. Additionally, we control four degrees of freedom in the hip: its height over ground and the rotation around all three spatial axes. The last degree of freedom is the head's pan-orientation that is without effect on the task, but also controlled and learned. The hip-control not only introduces additional redundancy in the control problem, but kinematically couples both arms: a change in the hip configuration moves both end-effectors. If for instance the right hand shall be moved utilizing the hip motion, but the left hand shall be fixed, also the left arm's joints have to be moved. This whole-body motion is neither trivially to control nor to learn. However, including the body movements enlarges the total operational range of the hands since the upper body can e.g. be leant in the direction of targets. Second, it allows more play for subsequent movements since joint limits can be effectively avoided.

B. Neural Architecture

The whole body inverse kinematics is learned by a recurrent neural network which receives subsequent target coordinates for both hands as input $u(t) = u^{l,r}(t) \in \mathbb{R}^6$. As output, the network shall produce the set of joint angles/control variables $q(t)$ such that the target position is reached ($F^{l,r}(q(t)) = u^{l,r}(t)$), where $F^{l,r} : \mathbb{R}^{15} \rightarrow \mathbb{R}^6$ denotes the forward kinematics of the system. We also learn a next-step prediction of the target positions $u^{l,r}(t+1)$, which can be used for autonomous behavior generation, but this application is beyond the scope of the current paper. The overall target output is

Connection	Sparseness	Init. range
Input-Reservoir	0.2	0.1
Input-Output	1.0	0.1
Reservoir-Reservoir	0.02	0.02
Reservoir-Output	0.2	0.1
Output-Reservoir	0.2	0.1

BPDC-Learning		IP-Learning	
Rate-Start	0.15	Rate-Start	0.015
Rate-End	0.025	Rate-End	0.0025
ϵ	0.002	μ	0.1

TABLE I: Network structure and learning parameters

Algorithm 1 Online reservoir learning algorithm

Require: set network size, sparsity, initial weights and states
for $k <$ iTrainIterations **do**
  get input $\mathbf{u}(k)$
  get target $\mathbf{d}(k)$
  execute network forward iteration (1),(2)
  apply BPDC rules $(5), (6)$
  apply IP rules $(8), (9)$
end for

$d(t) = (u^{l,r}(t+1)\ ;\ q(t)) \in \mathbb{R}^{21}$. The actual network outputs are denoted with $(\hat{u}^{l,r}(t+1)\ ;\ \hat{q}(t))$.

The respective network consists of 6 input-, 21 output- and 300 hidden "reservoir"-neurons. The output nodes receive the neuron activities from both input and reservoir (see. fig. 2). The reservoir receives the values of input and – in a recurrent loop – from the output nodes. Input is fully connected to the output, while the remaining connections are sparse with only 20% of the possible connections present. The reservoir is internally connected with sparsity 2%.

Formally, we consider the recurrent reservoir dynamics

$$\mathbf{x}(k+1) = \mathsf{W}^{net}\mathbf{y}(k) + \mathsf{W}^{in}\mathbf{u}(k), \quad (1)$$

$$\mathbf{y}(k) = \mathbf{f}(\mathbf{x}(k)), \quad (2)$$

where k is the discrete time step, $x_i, i = 1 \dots N$ are the neural activations, and $\mathbf{y} = \mathbf{f}(\mathbf{x})$ is the vector of neuron activities obtained by applying parameterized Fermi functions component-wise to the vector $\mathbf{x}$ as

$$y_i = f_i(x_i, a_i, b_i) = (1 + \exp^{(-a_i x_i - b_i)})^{-1}. \quad (3)$$

The R-dimensional vector $\mathbf{u}(k) = (u_1(k), \dots, u_R(k))^T$ denotes the inputs at time step k. We assume that the neurons are enumerated such that the first $O = 21$ neuron activations $x_i, i = 1 \dots O$ serve as output values. In our setting we can thus write

$$\mathbf{x}(k) = \left(\hat{u}^{l,r}(k+1)^T, \hat{q}(k)^T, x_{O+1}(k) \dots x_N(k)\right)^T \quad (4)$$

C. Online Learning Rules

Our setup involves two learning rules that work in parallel. Connections to the output nodes are adapted with the supervised *Backpropagation-Decorrelation* (BPDC) rule (see fig. 2). All other connections are randomly initialized from a uniform distribution and stay fixed (see tab. I). Additionally, an unsupervised *Intrinsic Plasticity* (IP) rule is applied in the reservoir neurons in order to optimize information transmission.

Backpropagation-Decorrelation: Output connections are adapted by the backpropagation-decorrelation (BPDC) learning rule, which has been introduced in [15], [16]. It can cope with feedback from output to the internal neurons [17]. A further strength of the BPDC rule is its capability to train the input-to-output weights comprising the linear feedforward path in a common framework together with the reservoir-to-output weights. A formalism to derive the corresponding weight updates $\Delta\mathbf{w}(k)$ was given in [15], [16] and leads to the following BPDC rule [15] for reservoir-to-output connections, i.e. for $i = 1 \dots O$, $j = 1 \dots N$

$$\Delta w_{ij}(k) = \bar{\eta}(k) f(x_j(k-1)) \gamma_i(k) \quad (5)$$

and input-to-output connections, i.e. for $i = 1 \dots O$, $r = 1 \dots R$ respectively

$$\Delta w_{ir}(k) = \bar{\eta}(k) u_r(k-1)) \gamma_i(k). \quad (6)$$

For time k and error $e_i(k) = (x_i(k) - d_i(k))$ we have

$$\gamma_i(k) = -e_i(k) + \sum_{s \in 0} w_{is} f'(x_s(k-1)) e_s(k-1). \quad (7)$$

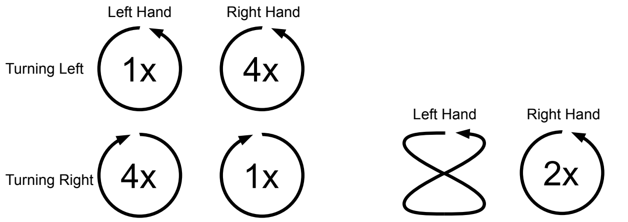
The time-dependent parameter

$$\bar{\eta}(k) = \eta \frac{1}{\|\mathbf{f}(\mathbf{x}(k-1))\|^2 + \|\mathbf{u}(k-1)\|^2 + \varepsilon}$$

can be interpreted as a time and context dependent learning rate. It scales the BPDC learning rate η with a factor dependent on the overall network and input activities and a small regularization constant ε, e.g. $\varepsilon = 0.002$. The BPDC learning constitutes an error correction rule with time-dependent learning rate $\bar{\eta}(k)$, input predicate $y_j(k) = f_j(x_j(k))$, and modified error $\gamma_i(k)$.

BPDC learning is a supervised learning technique, and has linear complexity $O(N)$ in the number of neurons N. Thus, the combination of IP reservoir adaptation (described below) and BPDC learning stays in the $O(N)$ complexity range, such that the forward iteration of the recurrent network (1), (2) becomes the computationally limiting factor ($O(N^2)$ for a fully connected network). In fact, our implementation utilizes sparse matrix multiplication, such that the overall execution and training of sparse networks with a limited number of connections per node also scales linear in the number of nodes. This efficiency renders the scheme ideally suited for implementation on robots for execution and learning in real time.

Intrinsic Plasticity: In order to improve the reservoir online, we use an adaption rule introduced by Triesch [18] and adopted for reservoir optimization in [19]. It is an unsupervised, efficient self-adaptation rule to optimize information transmission of the reservoir neurons by adapting the parameters a_i, b_i of the Fermi transfer functions $f_i(x, a_i, b_i) = (1 + \exp(-a_i x - b_i))^{-1}$ used in the network equation (3). The parameters a, b are updated by the following online gradient



(a) First training pattern, combined out of concurrent circular movements, first to the left, then to the right.

(b) Second pattern, consisting out of a figure "eight" for the left, and circles for the right hand

Fig. 3: Training patterns used for the training.

rule with learning rate ζ in time step k:

$$\Delta b(k) = \zeta \left(1 - \left(2 + \frac{1}{\mu}\right) y(k) + \frac{1}{\mu} y(k)^2\right), \qquad (8)$$

$$\Delta a(k) = \zeta \frac{1}{a(k)} + x(k) \Delta b(k). \qquad (9)$$

We call the joint application of rules (8), (9) to the non-output reservoir neurons intrinsic plasticity (IP) learning. It is local in time and space and therefore efficient to compute and is applied at each time step k after iteration of the network dynamics (1), (2), as shown in algorithm 1.

III. RESULTS

A. Exploration

In order to explore the kinematics and to acquire ground truth training data, we apply an explicit exploration phase. In that exploration phase, we use an analytic velocity-based feedback controller. That whole body motion (WBM) controller [20] uses all upper body degrees of freedom of ASIMO to perform a target motion of both hands. In resolving the redundancy it selects one particular out of the infinite number of solutions based on additional criteria like distance to the joint limits [20]. It is important to note here that the goal of learning is not to replicate the velocity mapping, that is solved by the existing controller. Rather, we learn a pure feedforward control, that solves the inverse kinematics directly on position-level. Thus, the joint angles necessary to realize a desired hand position are immediately available. This is not the case for the velocity-based feedback controller, which has to iteratively approach the target.

To acquire data, we first choose a target motion of the hands $x^{l,r}(t),\ t = 1, ..., T$. We provide this trajectory as target to the analytic WBM controller, that applies it on the real robot with a rate of 5Hz. During that execution the robots joint angles are recorded, i.e. at each t, the current joint values $q(t)$ are memorized. This recording yields a full set of training data $u(t) = x^{l,r}(t)$ and $d(t) = (x^{l,r}(t+1)\ ;\ q(t))$. We only train on one temporal sequence, which can have arbitrary length and form. Notably, training data generated in this fashion are *imperfect*, but realistic. Since the demonstration and execution of targets to the controller is also temporal, a target $x^{l,r}(t)$ is never exactly reached when the current joint angles are measured, so that $x^{l,r}(t)$ and $q(t)$ do not exactly correspond. Nevertheless we will show that this restriction does – depending on the target data – not seriously harm the learned kinematics solution.

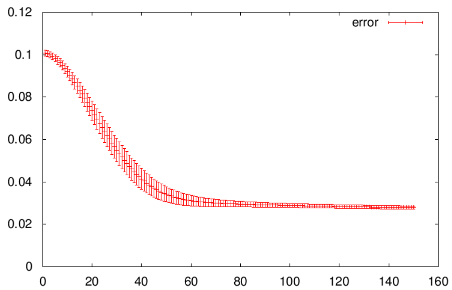
In the current setup, we only apply motions within a fixed vertical plane in front of ASIMO, i.e. the forward/backward component stays fixed. We show results for two distinct training targets $x^{l,r}(t)$. The first is a combination of circular movements (see figure 3a) with different speeds and directions with the plane. The motion roughly captures all top/down and left/right combinations of left and right hand and has a total length of $T = 256$ samples. For comparison, we use a figure "eight" for the left hand and circles for the right hand (figure 3b) with a total length of $T = 64$ timesteps. Here, the left/right orientations of both arms are exactly coupled. Thus it contains no situations where both hands point towards the middle or both hands point away from it.

B. Training

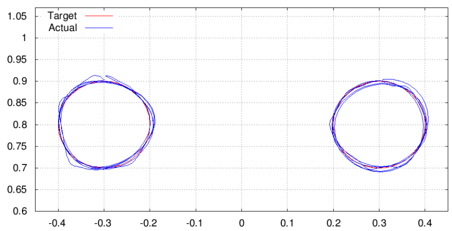
For systematic evaluation, the training procedure is organized in epochs and cycles, where a cycle is one full temporal presentation of the training motion $x^{l,r}(t)$. In each epoch we first re-initialize the network-state randomly and present one cycle to the network without training. Subsequently we show the complete pattern five times with enabled learning: after the presentation of each new target position $x^{l,r}(t)$, the output connections are adapted towards the target output $d(t) = (x^{l,r}(t+1)\ ;\ q(t))$ using the BPDC update rule, and the reservoir neurons are updated with the IP rule. A final cycle is used to estimate the error of the output joint angles $\hat{q}(t)$, while learning is disabled. To obtain an interpretable and realistic error measure, we do not directly compare against the joint-values in the training data. Instead we compute the hand positions that would correspond to the estimated joint angles using a analytical forward kinematic for left and right hand ($F^l(\cdot)$ and $F^r(\cdot)$). The measured error is then the mean euclidean distance $||\cdot||$ between desired and actual hand positions in meters:

$$err = \frac{1}{2} \sum_{t=1}^{T} \left(||F^l(\hat{q}(t)) - x^l(t)|| + ||F^r(\hat{q}(t)) - x^r(t)|| \right) \qquad (10)$$

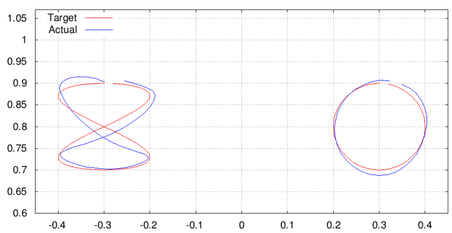
The error curve for the first training pattern over 150 epochs is shown in figure 4a. The second pattern is trained for 600 epochs (see fig. 5a) such that – due to the different pattern lengths – the number of training steps is identical. During these epochs, the learning rates of both BPDC and IP are continuously decreased following an exponential function from a defined start to a defined end value (see table I). This scheduling is not strictly necessary, but improves the performance. The plots show the mean error and std.-deviation over ten different parameter initializations. The two curves have almost identical characteristics: the first epochs have an error of approx. $10cm$, as the network starts to put each hand into the center of gravity of its target motion, which have a radius of $10cm$. The error reaches $2.79cm$ for the first, and $2.95cm$ for the second training pattern, but is still slightly decreasing in both cases.



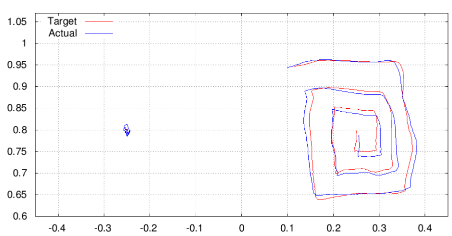
(a) Error within the evaluation cycles over 150 learning epochs. The plot shows mean and std. deviation over ten initializations.



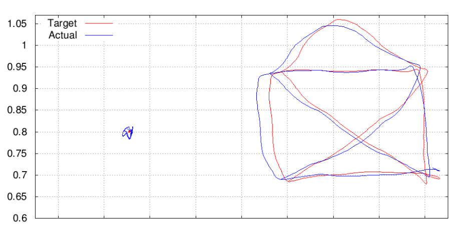
(b) Test with the trained synthetic pattern.



(c) Test with the untrained synthetic pattern.

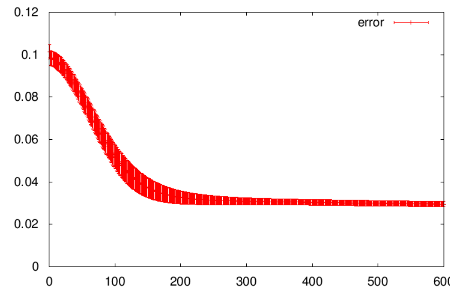


(d) Test with user generated target movements.

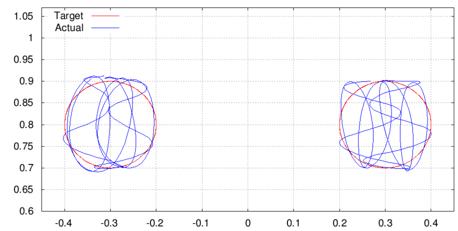


(e) Test with user generated target movements.

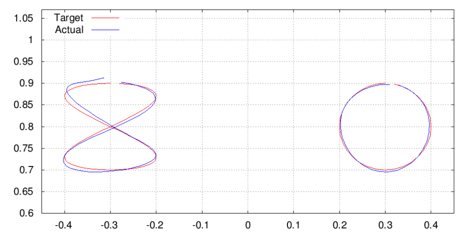
Fig. 4: Training with the first training pattern (fig. 3a)



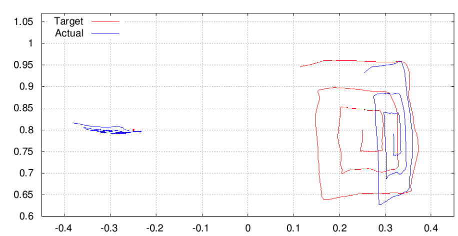
(a) Error within the evaluation cycles over 600 learning epochs. The plot shows mean and std. deviation over ten initializations.



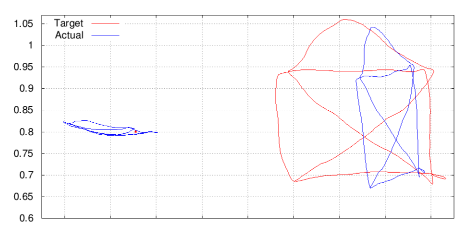
(b) Test with the untrained synthetic pattern.



(c) Test with the trained synthetic pattern.



(d) Test with user generated target movements.



(e) Test with user generated target movements.

Fig. 5: Training with the second training pattern (fig. 3b)

C. Evaluation

We tested the learned inverse kinematics on the real robot. Therefore a target motion is iteratively presented to the neural network and the estimated joint angles $\hat{q}$ are applied on the robot. The rate of presentation, and thus control, is thereby $10Hz$. Figure 6 shows ASIMO controlled by a network trained with the first, circular pattern. The target motion is the second training pattern which is completely new for that particular neural network. The images show subsequent stages of the movement in one second intervals and illustrate that all degrees of freedom (including the hip) are productively used. The robot hands' physical movement is plotted in figure 4c. Figure 5c shows the performance of a network trained with the second pattern. The displayed deviations from the target motion thereby arise from a combination of imperfect training data, the remaining training error of the neural network, but also the physical dynamics of ASIMO. In both cases the the training pattern can be reproduced within a reasonable accuracy. However, only the network trained with circular pattern is able to deal with the respectively other synthetic pattern. The network trained with the "eight"/circle combination fails to apply the first pattern as it has not learned an independent left/right movement of both hands.

D. User Interaction

In order to minimize blind spots in the generalization testing, we incorporated a motion tracking system [21], such that also naive users can demonstrate target movements just by holding a markered device in hand (see figure 1). Since it is – for humans – practically impossible to voluntarily control both hands at the same time, we restricted the demonstration to the right hand. However, the left hand is not free on ASIMO since we apply a constant target on that side. Trying to keep the left hand fixed does indeed not simplify the task, as left and right hand motions are coupled by the hip movements. Also, trying to stabilize one hand was not part of the training data.

Our setup involves the complete chain of exploration, learning, user demonstration of a target movement and its execution on ASIMO, using the recurrent network. Exploration and learning can be done within less than two minutes. A demonstrated target movement can be reproduced immediately and easily be executed in real time. The figures 4d, 4e, 5d and 5e show two user generated movements and their execution on ASIMO, each under the two discussed training conditions. The demonstrated movements differ massively from the training data in both range and dynamics. The network trained with the purely circular pattern is able to reproduce these movements with high accuracy even in regions that are spatially distant from the training pattern. Also the stabilization of the left hand – though completely untrained – is done with deviations less than 2cm. Again, the network trained with the "eight"/circle pattern fails to reproduce the movements. The bad stabilization performance of the left hand reflects the inability to move both hands independently along the left/right axis, which was here not part of the training. Here the left/right leaning degree of freedom in the hip can not be generalized to the users motion. Nevertheless the independent top/down movement, which *was* part of the training data works accurately.

IV. Discussion and Outlook

We demonstrated an inverse kinematics task for the highly redundant humanoid robot ASIMO. In order to apply a target motion of both hands at the same time, we can exploit all upper body degrees of freedom. The coordinated use of this non-linear kinematic chain thereby widely exceeds the difficulties in (even redundant) single robot arms. Our learning technique is able to deal with temporally correlated data and online learning, which are fundamental prerequisites to enable an incremental acquisition and also an ongoing refinement of motor skills. Also, it is fast enough to be used in real time on a real robot system.

We showed that – given proper machine learning techniques – reasonable generalization is possible even without excessive sampling of the joint space, but from few, systematically chosen samples in task coordinates. Our training setup showed that a spanning of all independent movement axes of both hands is necessary, but also rather sufficient to acquire an accurate inverse model for a high range of target positions. A random sampling of joint values would not only require a lot of samples in order to cover task relevant dimensions, but also evokes the fundamental problem of different, inconsistent solutions for the same target position [22]. With our approach we can not only learn one functioning, but also a near optimal solution in terms of comfortable postures since we learn from a near optimal teacher. This allows very smooth and natural movements. Both redundancy resolution and optimality are therefore a direct result of the task/goal directed exploration and do not naturally arise from a random joint space exploration. Besides the exploration or rather spanning of all possible movement axes, the successful training with the circular movement thereby only follows the principle of symmetry: each observed (e.g. top/down) combination of hand positions has a counterpart where the orientations of both hands are flipped.

Future work will address an exploration without analytic inverse kinematic as teacher. The current setup is plausible in the sense of kinesthetic teaching [23]: the robot's hands are guided by a human tutor, while learning takes place with kinesthetic information. Thereby an "optimization" is naturally applied by the physics. However, it is desireable to handle exploration also without teacher and thus in a fully autonomous way. The main problems here are (*i*) to deal with the huge redundancy in humanoid systems and to learn at least on correct solution out of infinitely many [3] and (*ii*) to cover the task relevant space in an efficient way, avoiding to get stuck in postures that will never be used.

Acknowledgements

We would like to thank Sebastian Wrede, Michael Götting and the whole CoR-Lab ASIMO-team for their support and invaluable work on our ASIMO and user interaction setup. Without their spent effort this scenario would not have been realizable.

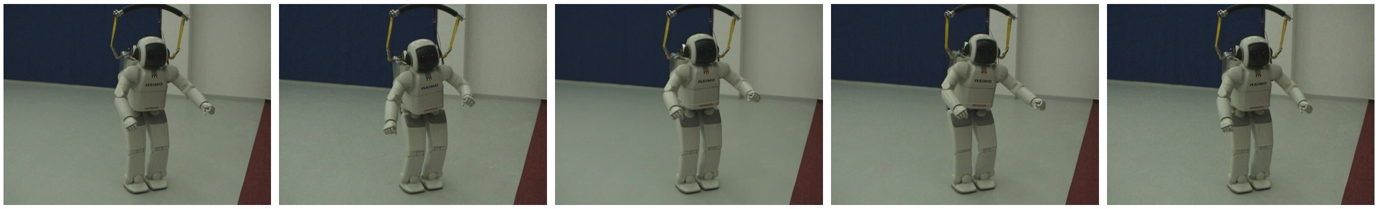


Fig. 6: ASIMO controlled with a learned inverse kinematics solution. The network was trained with a first training pattern and is here tested on the second pattern. The network exploits all degrees of freedom in arms and hip.

REFERENCES

[1] A. Meltzoff and M. Moore, "Explaining facial imitation: A theoretical model," *Early Development and Parenting*, vol. 6, pp. 179–192, 1997.

[2] Y. Demiris and A. Dearden, "From motor babbling to hierarchical learning by imitation: A robot developmental pathway," in *Proceedings of the fifth international workshop on epigenetic robots (EPIROB): modeling cognitive development in robotic systems*, 2005, pp. 31–37.

[3] S. V. Aaron D'Souza and S. Schaal, "Learning inverse kinematics," in *Proc. IROS*, 2001, pp. 298–303.

[4] C. von Hofsten, "An action perspective on motor development," *Trends in Cognitive Sciences*, vol. 8, no. 6, pp. 266–272, 2004.

[5] Y. Sakagami, R. Watanabe, C. Aoyama, S. Matsunaga, N. Higaki, and K. Fujimura, "The intelligent asimo: system overview and integration," in *Intelligent Robots and System*, 2002, pp. 2478–2483.

[6] N. Ramnani, "The primate cortico-cerebellar system: anatomy and function," *Nature Reviews Neuroscience*, vol. 7, no. 7, pp. 511–522, 2006.

[7] R. C. Miall and D. M. Wolpert, "Forward models for physiological motor control," *Neural Networks*, vol. 9, no. 8, pp. 1265–1279, 1996.

[8] T. Yamazaki and S. Tanaka, "The cerebellum as a liquid state machine," *Neural Networks*, vol. 20, no. 3, pp. 290–297, 2007.

[9] S. Vijayakumar, A. D'Souza, and S. Schaal, "Incremental online learning in high dimensions," *Neural Computation*, vol. 17, no. 12, pp. 2602–2634, 2005.

[10] A. Ijspeert, J. Nakanishi, and S. Schaal, "Learning rhythmic movements by demonstration using nonlinear oscillators," in *IROS*, 2002.

[11] ——, "Movement imitation with nonlinear dynamical systems in humanoid robots," in *ICRA*, 2002, pp. 1398–1403.

[12] M. Ito and J. Tani, "On-line imitative interaction with a humanoid robot using a dynamic neural network model of a mirror system," *Adaptive Behavior*, vol. 12, no. 2, pp. 93–115, 2004.

[13] J. Tani, M. Ito, and Y. Sugita, "Self-organization of distributedly represented multiple behavior schemata in a mirror system: reviews of robot experiments using rnnpb," *Neural Networks*, vol. 17, pp. 1273–1289, 2004.

[14] R. Yokoya, T. Ogata, J. Tani, K. Komatani, and H. Okuno, "Experience based imitation using rnnpb," *Advanced Robotics*, vol. 21, no. 12, pp. 1351–1367, 2007.

[15] J. J. Steil, "Backpropagation-decorrelation: Recurrent learning with O(N) complexity," in *Proc. IJCNN*, vol. 1, 2004, pp. 843–848.

[16] J. Steil, "Online stability of backpropagation-decorrelation recurrent learning," *Neurocomputing*, vol. 69, no. 7–9, pp. 642–650, 2006.

[17] F. R. Reinhart and J. J. Steil, "Attractor-based computation with reservoirs for online learning of inverse kinematics," in *ESANN*, April 2009 2009, pp. 257–262.

[18] J. Triesch, "A gradient rule for the plasticity of a neuron's intrinsic excitability," in *Proc. ICANN*, 2005, pp. 65–79.

[19] J. J. Steil, "Online reservoir adaptation by intrinsic plasticity for backpropagation-decorrelation and echo state learning," *Neural Networks, Special Issue on Echo State and Liquid State networks*, 2007, under review.

[20] M. Gienger, H. Janssen, and C. Goerick, "Task-oriented whole body motion for humanoid robots," in *Proceedings of the IEEE-RAS/RSJ International Conference on Humanoid Robots*, 12 2005.

[21] VICON Motion Tracking Systems: http://www.vicon.com.

[22] I. Jordan and Ruemelhart, "Supervised learning with a distal teacher," *Cognitive Science: A Multidisciplinary Journal*, vol. 16, no. 3, pp. 511–522, 2006.

[23] S. Calinon and A. Billard, "Active teaching in robot programming by demonstration," in *16th IEEE International Symposium on Robot and Human interactive Communication*, 2007, pp. 702–707.