Learning Flexible Full Body Kinematics for Humanoid Tool Use

Matthias Rolf, Jochen Steil, Michael Gienger

2010

Learning Flexible Full Body Kinematics for Humanoid Tool Use

Matthias Rolf, Jochen J. Steil and Michael Gienger

***Abstract*—We show that inverse kinematics of different tools can be efficiently learned with a single recurrent neural network. Our model exploits all upper body degrees of freedom of the Honda's humanoid robot research platform. Both hands are controlled at the same time with parametrized tool geometry. We show that generalization both in space as well as across tools is possible from very few training data. The network even permits extrapolation beyond the training data. For training we use an efficient online scheme for recurrent reservoir networks utilizing supervised backpropagation-decorrelation (BPDC) output adaptation and an unsupervised intrinsic plasticity (IP) reservoir optimization.**

***Index Terms*—Full Body Kinematics, Neural Networks, Tool Use, Humanoid Robots**

I. Introduction

The flexibility in which humans use and extend their motor repertoire is still outstanding compared to those of humanoid robots. Humans can control their hands with enormous flexibility in order to manipulate objects, for instance by using different fingers or the complete palm to manipulate objects. While using tools humans can even control and utilize e.g. the tip of a screwdriver. A remarkable example of tool control is the use of laser pointers, where a held object is used to control the position of a light point on a wall.

The learning of tool use has been increasingly studied in robotics research over the last decade. Behavioral aspects have been investigated by Stoytchev *et al.* under the notion of tool affordances [1], [2]. Also the important problem of visual recognition and control of tool-tip points has been subject to several studies [3], [4]. However, it is hardly understood how humans are able to voluntary control not only one particular tool, but many different tools in a flexible way – and how robots can do so. In this study, we focus on the kinematics perspective of flexible tool control: how does a robot need to coordinate its body in order to use and control tools? Different tools, as well as the use of different fingers, must each be described by different kinematic functions. When, for instance, a button is pressed with the tip of the index finger, this requires a joint configuration different from pressing it with the thumb or even hitting it with a tool. Each of these scenarios must be described with a separate inverse kinematic function in order to control the actual position of the effector.

In the forward kinematic case, this scenario can be analytically described by a variable geometric offset from the robots

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

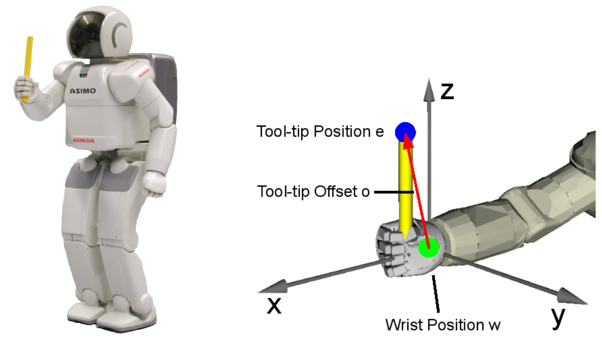


Fig. 1: Honda's humanoid robot with a tool in the right hand. The position of the tool-tip is modeled as an offset from the wrist, that is specified in wrist-centered coordinates.

wrist (see Fig. 1). This offset describes where the current *tool-tip* is located in relation to the wrist. Each tool has a specific offset due to its geometry and length, and how it is held in hand. Within that notion, also the own finger is seen as "tool", as the finger tip has a specific geometric offset from the wrist. Although feedback-based control laws can be formulated based on the forward kinematics equation, a direct inverse function is not trivially to get.

A. Related Machine Learning Approaches

The learning of such motor skills has been identified as one of the key ingredients to intelligent behavior in both robotics and computational neuroscience [5]. Machine learning techniques have been very successfully applied to specific inverse kinematics problems [6]. Several approaches have been used in order to increase flexibility in such learning systems. Under the notion of extendable or adaptive body schemata, several studies investigated how motor and control knowledge can be re-learned for the case of tool use [7], [8]. This approach does justice to the fact that tools are apparently "integrated" into human body schemata [9]. An obvious drawback of the re-learning approach is that only one kinematic function can be represented at a time. Therefore even the own body model is forgotten once a tool is learned. Representing different kinematic functions becomes possible with modular approaches. The MOSAIC model for instance has been discussed as a model for human cerebellar motor learning [10]. Each single module can represent the kinematic function of a particular finger or tool. Alternatively, it was proposed to use parametric models. One module is responsible for several mappings, a particular mapping is selected by parameters, which are externally given to the system. Such models have been successfully applied to the problem of autonomous behavior generation [11]. Parametrized models provide a straight forward way to generalization. New parameters can be fed to the network in order to generate new behavior [12].

B. Approach

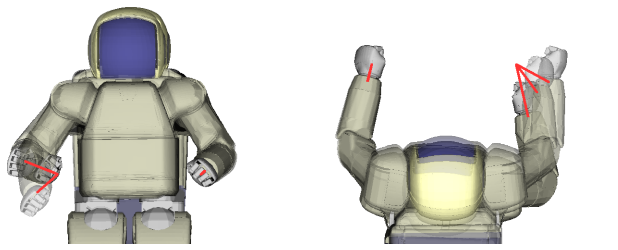
We apply this parametrization approach to the problem of tool kinematics. We train a single neural network to represent inverse kinematics. The geometry of the tool is parametrized, such that different tool-tip offsets can be fed into the network, invoking the corresponding inverse kinematic function. In order to learn and represent the parametrized kinematics, we use a recurrent neural network model, developed under the notion of *reservoir computing* [13], [14]. It has been argued that this learning architecture can serve as model for learning in the human cerebellum [15]. Physiologic evidence shows that inverse models of the body [16] and even tool control [17] are located and also learned in cerebellum. Hence, our model is biologically highly plausible.

In contrast to local learning approaches [6] reservoir computing provides a rather holistic approach to tool and full body kinematics. This approach has been shown previously to allow excellent generalization across different target positions of the effector [18]. The holistic scheme is very efficient, both in terms of samples needed for the training and efficiency of computation. In this paper we show that, using the parametrization approach, also generalization across different tools is possible, whereas the generalization in space is maintaned. As platform, we use the Honda's full body humanoid robot [19]. We apply bi-manual target motions with variable tools, utilizing whole body motion.

II. Kinematic Problem

Forward and inverse kinematics describe the relation between the joint angles $\vec{q} \in \mathbb{R}^n$ of a robot and the corresponding position of the effector $\vec{e} \in \mathbb{R}^m$, e.g. the position of a tool-tip. We denote the forward kinematic function with $\vec{e} = F(\vec{q})$, which uniquely determines the tool-tip position given the joint angles. When the effector must be moved to some desired position, the inverse function F^{-1} is needed. However, that function is not unique in the case of redundancy ($n > m$) which is the standard case for humanoid robots.

On the humanoid robot, we currently control $n = 15$ degrees of freedom. Five joint angles are controlled in each arm: three rotational joints in the shoulder, one in the ellbow, and the rotation of the hand around the forearm axis. Four virtual joints are controlled in the hip: its orientation around all three spatial axes and the height over ground. The hip degrees of freedom are realized by means of leg motion [20]. As additional (not task-relevant) degree of freedom, the head-pan direction is controlled. The target positions are the 3D tool-tip positions for both hands $\vec{e}^{l,r} = \vec{e} \in \mathbb{R}^6$ in world coordinates. We use the wrist positions $\vec{w}^{l,r} = \vec{w} \in \mathbb{R}^6$ (see Fig. 1) as reference


(a) The same target position is applied for different tool-tip offsets (red).

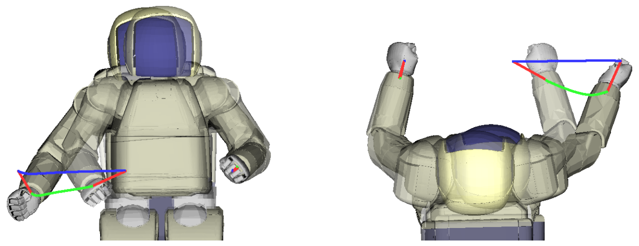

(b) The offset (red) is kept constant, while the tool-tip is moved along a straight line (blue). The position of the wrist point during that motion in green.

Fig. 2: Examples of humanoid and tool kinematics.

vector for the tool-tip position. We denote the forward wrist kinematics by: $\vec{w} = F^{\vec{w}}(\vec{q})$. The tool-tip position is defined by adding a *tool-tip offset* $\vec{o}^{l,r} = \vec{o} \in \mathbb{R}^6$ in wrist centered coordinates W. Since the orientation of the wrist in space changes when the limbs are moving, also the orientation of the offset vector $\vec{o}$ in world coordinates changes. We denote the orientation transformation from wrist to world coordinates by $T^0_W(\vec{q})$. The forward kinematics equation for the tool-tip position is:

$$\vec{e} = F^{\vec{e}}(\vec{q}, \vec{o}) = F^{\vec{w}}(\vec{q}) + T^0_W(\vec{q}) \cdot \vec{o} \tag{1}$$

The offset $\vec{o}$ now acts as parameter to describe various forward kinematic functions.

Interesting about the humanoid's kinematics, and a challenge for the learning, is that both arms are coupled by the upper body motion. If the hip is moved, both arms change their position in space. If one arm shall be moved utilizing the hip degrees of freedom, the other arm also changes its position, except the upper body motion is actively compensated by the other arm to maintain its position. If upper body motion is utilized, no separate kinematic function for left and right arm exist. They must be considered together. The additional redundancy allows smoother movements and a better avoidance of joint limits due to more flexibility in the system. Also, the upper body widens the total range of operation for the hands. Targets that are out of range without upper body motion can be reached, for instance, by leaning forward or backward.

Figures 2a and 2b show several ground truth examples of the inverse kinematics. Figure 2a shows an overlay of three different tool-tip offsets (see Table II, right), while the tool-tip position in world coordinates is identical. This corresponds to fulfilling the same task (e.g. pressing a button) with various tool or also fingers. When the tool is e.g. elongated along the forearm-/x-axis, hand and arm must be positioned backwards to compensate the offset. Due to the kinematic coupling also the upper body is moved which in turn requires slightly different postures for the left arm. In Figure 2b the tool-tip offset is constant, and a linear movement of the tool-tip from right to left is generated. As the posture changes, also the orientation of the offset vector in world coordinates changes, which requires a highly non-linear trajectory of the wrist in space.

III. Learning Architecture

The whole body inverse kinematics is learned by a recurrent neural network. It receives subsequent target tool-tip coordinates for both hands and the current tool-tip offsets as input $u(t) = (\vec{o}(t), \vec{e}(t)) \in \mathbb{R}^{12}$. As output, the network shall produce the set of joint angles/control variables $\vec{q}(t)$, such that the target position is reached ($F^{\vec{e}}(\vec{q}(t), \vec{o}(t)) = \vec{e}(t)$). The network target output is denoted with $d(t) = \vec{q}(t) \in \mathbb{R}^{15}$. The actual network outputs are denoted with $\hat{\vec{q}}(t)$. Therefore, the network is trained to approximate an inverse kinematic function $\hat{F}^{-1}(\vec{o}(t), \vec{e}(t)) = \hat{\vec{q}}(t)$. Throughout our experiments, the offset vector $\vec{o}$ is assumed to be known and acts as a parameter for the estimated inverse kinematics.

The network consists of 12 input-, 15 output- and 500 hidden "reservoir"-neurons. Output nodes receive the neuron activities from both input and reservoir (see Fig. 3). The reservoir receives the values of input and – in a recurrent loop – from the output nodes. The input is fully connected to the output, while the remaining connections are sparse with only 20% of the possible neuron-to-neuron connections present. The reservoir is internally connected with sparsity 2%. As shown in [21], the recurrent architecture of the reservoir implements attractor dynamics, which enables efficient learning of static mappings like inverse kinematics with a dynamical system.

Formally, we consider the recurrent reservoir dynamics

$$\mathbf{x}(k+1) = \mathsf{W}^{net}\mathbf{y}(k) + \mathsf{W}^{in}\mathbf{u}(k), \qquad (2)$$

$$\mathbf{y}(k) = \mathbf{f}(\mathbf{x}(k)), \qquad (3)$$

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

$$y_i = f_i(x_i, a_i, b_i) = (1 + \exp^{(-a_i x_i - b_i)})^{-1}. \qquad (4)$$

The inputs are fed into the network with the weight matrix W^{in}, while W^{net} denotes the weights between neurons inside the network.

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 $O = 15$ neuron activations $x_i, i = 1 \dots O$ serve as output values. In our setting we can thus write

$$\mathbf{x}(k) = \left(\hat{q}(k)^T, x_{O+1}(k) \dots x_N(k)\right)^T \qquad (5)$$

Our setup involves two learning rules that work in parallel. Connections to the output nodes are adapted with the supervised *Backpropagation-Decorrelation* (BPDC) rule [22] (see Fig. 3). All other connections are randomly initialized from a uniform distribution (see Tab. I) and stay fixed. Additionally, an unsupervised *Intrinsic Plasticity* (IP) rule [23], [14] is applied in the reservoir neurons in order to optimize information transmission.

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	0.25	Rate-Start	0.025
Rate-End	0.001	Rate-End	0.0001
ϵ	0.002	μ	0.1

TABLE I: Network structure and learning parameters

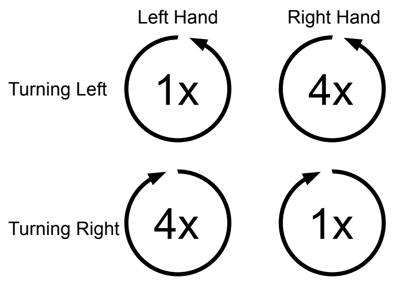


Fig. 4: Training pattern used for the training: a combination of concurrent circular movements. Different speeds of the movements provide all necessary combinations of the spatial axes.

IV. Experiments

In order to explore the kinematics and to acquire ground truth training data, we use the analytic feedback controller [24] based on the effector Jacobian. The goal of learning is not to replicate the velocity mapping that is implemented by the analytic 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 tool-tip position are immediately available.

To acquire data, we first choose a target motion of the tool-tip positions $\vec{e}(t),\ t = 1, ..., T'$. We train on one temporal sequence, which can have arbitrary length and form. We provide this trajectory as target to the analytic controller, that applies it on the real robot with a rate of 5Hz for a certain effector offset $\vec{o}_i$. During that execution, the robots joint angles $\vec{q}_i(t)$ are memorized at each t. This recording yields a set of training data $u_i(t) = (\vec{o}_i; \vec{e}(t))$ and $d_i(t) = \vec{q}_i(t)$ for the offset $\vec{o}_i$.

In the current setup, we apply tool motions within a fixed vertical plane in front of the robot. Our training pattern is a combination of circular movements with different speeds and directions within that plane (see Figure 4). The motion roughly captures all top/down and left/right combinations of left and right tool-tip and has a total length of $T' = 256$ samples. The

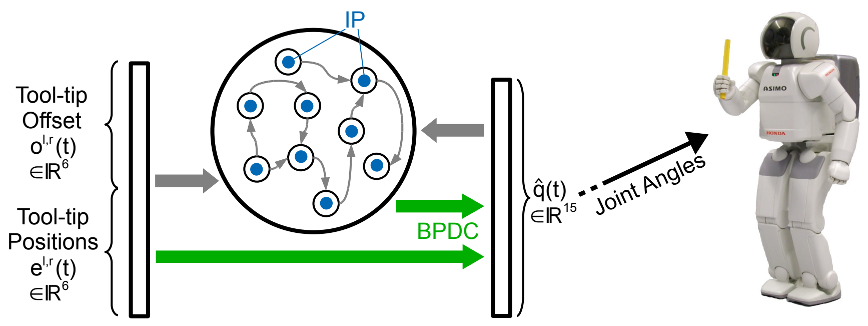

IP
Tool-tip Offset $o^{l,r}(t)$ $\in \mathbb{R}^6$
Tool-tip Positions $e^{l,r}(t)$ $\in \mathbb{R}^6$
BPDC
$\hat{q}(t)$ $\in \mathbb{R}^{15}$
Joint Angles

Fig. 3: Target positions of both tool-tips and the tool-tip offsets 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 (IP) rule within the reservoir. The estimated joint angles are applied on robot in order to follow a target movement.

Training offsets			
	x	y	z
`Default`	0.05	0	0
`x-off`	0.1	0	0
`y-off`	0.05	0.05	0
`z-off`	0.05	0	0.05

Test offsets			
	x	y	z
`x-test`	0.15	0	0
`y-test`	0.05	0.1	0
`z-test`	0.05	0	0.1

TABLE II: Tool-tip offsets of the right hand. The default offset vector is 0.05m straight from the wrist point, in the center of the palm. Each one additional training offset is chosen 0.05m distant from it along each axis. The offsets used for testing are 5cm more distant (see Figures 2a, 2b, 6a and 6b). The left tool-tip's offset is fixed at $(0.05; 0; 0)$.

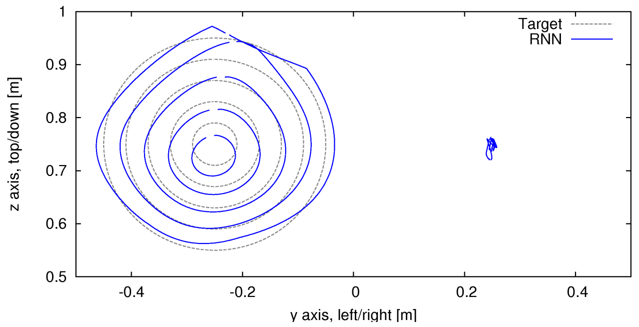
circles have a radius of each 10cm, the center is placed 75cm over ground, and $+25$cm and -25cm distant from the body axis to the left and right respectively.

Besides this sampling of target tool-tip positions in space, different tool-tip offsets $\vec{o}$ need to be explored. We use four different offsets for the training, which are listed in Table II. For this experiment we only vary the offset of the right hand. The left hand receives a constant offset. For the three-dimensional offset space of the right hand, four offset samples are notably the absolute minimum of samples in order to span the full space. The sample offsets are very close ($5cm$) to the hand center point, such that they basically incorporate the size of the hand itself.

We require the analytic controller to produce the same circular tool-tip trajectory $\vec{e}(t)$ for each of the offsets. Due to the different kinematic structure, the recorded joint angles $\vec{q}_i(t)$ differ for each offset $\vec{o}_i$. As overall result of this exploration procedure, we gain the set of network inputs $u(t)$ and target outputs $d(t)$ simply by concatenating the data sets of the four different tool-tip offsets in time. For all offsets together, the training data has $T = 1024$ samples.

A. Training

For systematic evaluation, the training procedure is organized in epochs and cycles, where a cycle is one full temporal presentation of the training input pattern $u(t)$. 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 $u(t)$, the output connections are adapted towards the target output $d(t)$ using the BPDC update rule. The reservoir neurons are updated with the IP rule. 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.



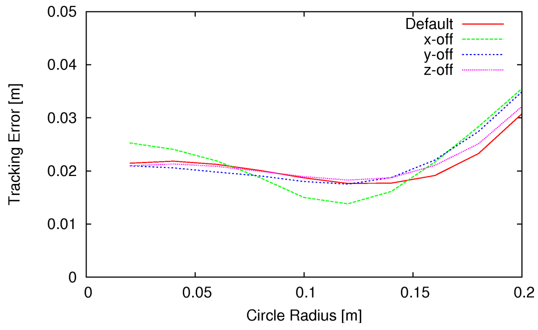


Fig. 5: **Top**: Result trajectories for circle-targets for the right tool-tip and a fixed target position for the left tool-tip. The circle radius is scaled up iteratively. **Bottom**: Tracking error dependent on circle radius, shown for all four training offsets.

B. Spatial Generalization

Previous studies already showed that a high degree of generalization and even extrapolation capabilities to unseen target coordinates is possible from few data with the proposed recurrent network scheme [18], [21]. Here we can show that this generalization is preserved with the additional difficulty of a parametrized and variable tool-tip offset. Figure 5 (top) shows the spatial generalization for the trained tool-tip offset `x-off`. The target tool-tip movement is a circle for the right hand, whereas the left hand receives a fixed target coordinate. This fixed target does in fact not ease the task for the neural network. This behavior is completely untrained and requires the network to compensate upper body movements with the left arm to maintain the position. Step by step the radius of the right hand's circle is increased up to 20cm, which is twice the radius contained in the training data.

A quantitative evaluation is shown in Fig. 5 (bottom). For the scaled circular movements, the tracking error is plotted against the radius of the circle. To obtain an interpretable and realistic error measure, we do not directly compare against ground truth joint-values. Instead we measure the tool-tip positions that are result of the estimated joint angles and compare it to the desired tool-tip position. The measured error is then the mean euclidean distance $||\cdot||$ between desired and actual tool-tip positions in meters:

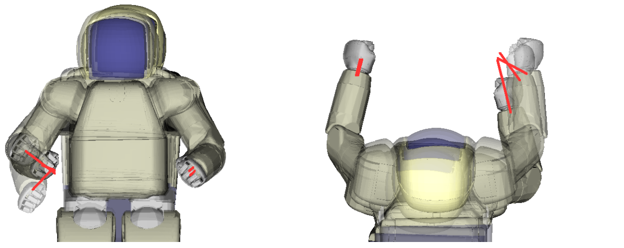
$$err = \frac{1}{2}\sum_{t=1}^{T}\left(||F_l^e(\hat{q}(t)) - e^l(t)|| + ||F_r^e(\hat{q}(t)) - e^r(t)||\right) \quad (6)$$

The plot shows the radius-scaling results for all four tool-tip offsets that were part of the training. The error is rather constant and very low up to a circle radius of 15cm. Afterwards the error increases but still remains very moderate. Even for 20cm radius, where the network has to extrapolate along twice the training radius, the error is clearly below 4cm which is remarkable for the case of open loop control.

C. *Tool Generalization*

The main point of the present study is the flexibility introduced by a parametrized tool-tip offset. The results presented in the previous section show that the inverse kinematic functions for several trained tools can be stored within a single neural network. No re-learning is necessary. However, it is also crucial to deal with different offsets than those present in the training. In the training, only four different tool-tip offsets were presented to the network, spanning the three dimensional offset space for the right hand. Here we show that generalization to untrained offsets is possible with our model from such few samples. Even extrapolation widely beyond the training samples is possible.

Figure 6a repeats the examples from Fig. 2a, but with the trained neural network instead of the analytic feedback controller. The offsets have twice the distance from the wrist than those present in the training (see Tab. II). Nevertheless the network is able to extrapolate, such that the target tool-tip position is reached very accurately. For instance, the network has learned to put the arm back for a long tool on the x-axis. Figure 6b shows the execution of a linear target motion with extrapolated tool-tip offset (compare to Fig. 2b). Although the resulting trajectory shows some curvature, the kinematic solution is remarkably accurate. The stabilization of the left hand is almost perfect in this case.


(a) The same target position is applied for different tool-tip offsets (red).

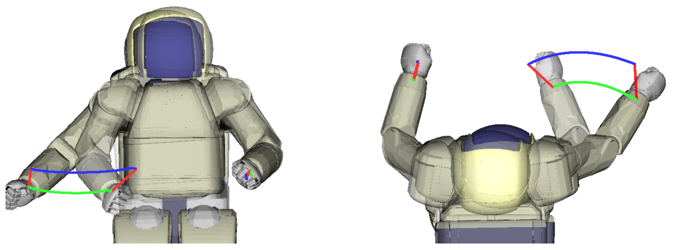

(b) The offset (red) is kept constant, while the tool-tip shall be moved along a straight line (blue). The position of the wrist point during that motion in green. The offset for the right hand is `z-test` $= (0.05, 0.0, 0.1)$.

Fig. 6: Extrapolation performance of a trained neural network for different tool-tip offsets.

Figure 7 shows an experiment in which we keep the target motion constant, but scale the tool-tip offset. The target motion is the same circle as in the spatial generalization experiment, but fixed to radius 10cm. For the right hand, we systematically shift the tool-tip offset away from the default point:

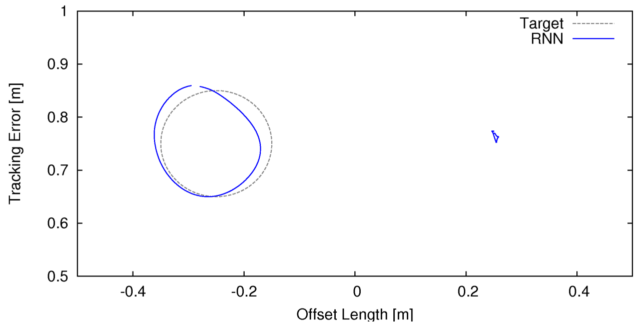
$$\vec{o}^{right} = (0.5; 0.0; 0.0) + length \cdot \frac{\Delta}{|\Delta|}$$

We tested along four direction: pure shift along x ($\Delta = (1; 0; 0)$), pure along y ($\Delta = (0; 1; 0)$), pure along z ($\Delta = (0; 0; 1)$), and mixed ($\Delta = (\frac{1}{3}; \frac{1}{3}; \frac{1}{3})$). Each direction is tested up to a length of 15cm, where only 5cm distance from the `Default` point were present in the training data. Figure 7 shows the tracking error (see equ. 6) depending on the offset length. The network yields a very good kinematic solution up to a scaling length of 10cm, which displays a high degree of generalization also across tools.

V. Discussion and Outlook

The experiments show that our methodology allows to learn kinematics of different tools within a single neural network. The network can learn to coordinate the whole body for doing so. The network's ability to generalize is remarkable. One smooth sample motion, consisting out of 256 closely connected samples, is sufficient to learn the whole body kinematics for a given tool. To generalize across tools, we trained four different offsets for the right hand, which is the minimum to span the 3D offset space at all. In both cases (spatial and tool generalization) the network gives accurate results up to twice the amplitude of the training data.

As the complete problem is learned and represented within a single network, no re-learning is necessary to use a tool – and thus no forgetting of the own body model. Contrary



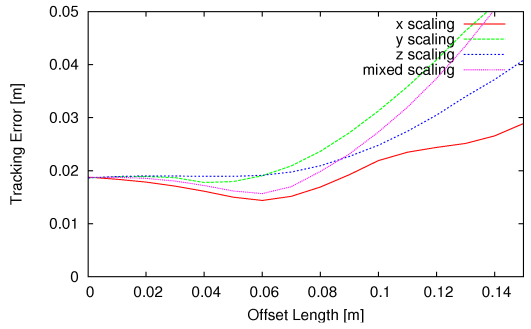


Fig. 7: **Top**: Circular target for the right hand with untrained offset `y-test`. **Bottom**: Tracking error of the circle, dependent on the length of the tool-tip offset.

to modular approaches, expertice and computational resources are holistically distributed in the network and can be efficiently used for all different tools. Undoubtedly, the necessity for different modules will arise at a certain degree of complexity. Imamizu *et al.* could for instance show separate modules in the human cerebellum when subjects were faced with qualitatively different control problems [17], [25]. However, we could show empirically that this point of complexity is not yet reached for quantitative differences like varying tool-tip offsets.

Evenmore, the generalization from offsets very close to the hand to more distant offsets indicates that the control of an external tool is possible once the coordination of the own, full body is mastered in a flexible manner.

A. Outlook

In the present study we learned a feedforward control. Such feedforward control is very usefull in the sense that it allows an immediate estimate of where to place the joints and is not subject to the stability issues of feedback control. However, the integration of an additional feedback control loop will be subject to further research in order to implement a fine-tuning of the kinematic solution and improve the accuracy. Hybrid control architectures have e.g. been studied in feedback-error learning [26] or more recently in [27].

Currently, we generate training data with an existing analytic controller. This prerequisite will be relaxed in further studies. In fully autonomous motor learning scenarios, generalization has been shown to be crucially important [28]. Therefore we are convinced that the current learning architecture will also be a useful method for tackling such further questions.

Acknowledgments

Matthias Rolf gratefully acknowledges the financial support from Honda Research Institute Europe for the Project "Neural Learning of Flexible Full Body Motion".

References

[1] A. Stoytchev, "Behavior-grounded representation of tool affordances," in *ICRA*, April 2005, pp. 3060–3065.

[2] S. Griffith, J. Sinapov, M. Miller, and A. Stoytchev, "Toward interactive learning of object categories by a robot: A case study with container and non-container objects," in *ICDL*, 2009.

[3] C. C. Kemp and A. Edsinger, "Robot manipulation of human tools," in *ICDL*, 2006.

[4] A. Edsinger and C. C. Kemp, "What can i control? a framework for robot self-discovery," in *EpiRob*, 2006.

[5] D. M. Wolpert, Z. Ghahramani, and J. R. Flanagan, "Perspectives and problems in motor learning," *Trends in Cog. Sci.*, vol. 5, no. 11, pp. 487–494, 2001.

[6] A. D'Souza, S. Vijayakumar, and S. Schaal, "Learning inverse kinematics," in *Proc. IROS*, 2001, pp. 298–303.

[7] C. Nabeshima, Y. Kuniyoshi, and M. Lungarella, "Adaptive body schema for robotic tool-use," *Advanced Robotics*, no. 20(10), 2006.

[8] M. Hersch, E. Sauser, and A. Billard, "Online learning of the body schema," *International Journal of Humanoid Robotics*, no. 5(2), 2008.

[9] A. Maravita and A. Iriki, "Tools for the body (schema)," *Trends in Cog. Sci.*, vol. 8, no. 2, pp. 79–86, 2004.

[10] D. Wolpert and M. Kawato, "Multiple paired forward and inverse models for motor control," *Neural Networks*, pp. 1317–1329, 1998.

[11] 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.

[12] H. Arie, T. Endo, T. Arakaki, S. Sugano, and J. Tani, "Creating novel goal-directed actions using chaotic dynamics," in *IEEE 8th International Conference on Development and Learning*, 2009.

[13] H. Jaeger, "Adaptive nonlinear system identification with echo state networks," in *NIPS*, 2002, pp. 593–600.

[14] 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.

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

[16] N. Ramnani, "The primate cortico-cerebellar system: anatomy and function," *Nature Reviews Neuroscience*, no. 7(7), 2006.

[17] H. Imamizu, S. Miyauchi, T. Tamada, Y. Sasaki, R. Takino, B. Ptz, T. Yoshioka, and M. Kawato, "Human cerebellar activity reflecting an acquired internal model of a new tool," *Nature*, vol. 403, no. 6766, pp. 192–195, 2000.

[18] M. Rolf, J. J. Steil, and M. Gienger, "Efficient exploration and learning of full body kinematics," in *ICDL*, 2009.

[19] 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.

[20] T. Takenaka, "The control system for the honda humanoid robot," *Age and Ageing*, pp. ii24–ii26, 2006.

[21] F. R. Reinhart and J. J. Steil, "Attractor-based computation with reservoirs for online learning of inverse kinematics," in *ESANN*, 2009.

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

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

[24] M. Gienger, H. Janssen, and C. Goerick, "Task-oriented whole body motion for humanoid robots," in *Int. Conf. Humanoid Robots*, 2005.

[25] H. Imamizu, T. Kuroda, S. Miyauchi, T. Yoshioka, and M. Kawato, "Modular organization of internal models of tools in the human cerebellum," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 100, no. 9, pp. 5461–5466, 2003.

[26] M. Kawato, K. Furukawa, and R. Suzuki, "A hierarchical neural-network model for control and learning of voluntary movement," *Journal Biological Cybernetics*, vol. 57, no. 3, 1987.

[27] F. Nori, L. Natale, G. Sandini, and G. Metta, "Autonomous learning of 3d reaching in a humanoid robot," in *IROS*, 2007.

[28] T. D. Sanger, "Failure of motor learning for large initial errors," *Neural Computation*, vol. 16, no. 9, pp. 1873–1886, 2004.