

See discussions, stats, and author profiles for this publication at: <https://www.researchgate.net/publication/334363920>

# Building Neural Representations for Flexible Tool-Use

Preprint · July 2019

DOI: 10.13140/RG.2.2.31554.94401

---

CITATIONS


0

---

READS

24


2 authors, including:

 **Jochen J. Steil**  
Technische Universität Braunschweig

227 PUBLICATIONS 2,899 CITATIONS

SEE PROFILE

Some of the authors of this publication are also working on these related projects:

 AMARSi View project

 CogIMon View project

# Building Neural Representations for Flexible Tool-Use

Klaus Neumann and Jochen J. Steil

Research Institute for Cognition and Robotics (CoR-Lab)  
Bielefeld University - Germany

**Abstract.** We introduce a novel approach to model tool-use for humanoid robots such as iCub. The learning represents the highly redundant coupling of the inverse kinematics and the motions imposed by the tool in a flexible way. It is shown that the manipulation kinematics can be learned from few ground-truth examples using an efficient extreme learning machine framework. The experiments reveal that this approach exhibits human-like behavior (motor hysteresis effect) without learning from human data.

**Keywords.** Neural network, learning, extreme learning machine, inverse kinematics, robotics, redundancy, tools, dynamic system.

## 1 Introduction

The ability to use tools is one of the cornerstones of behavioral intelligence and fundamental to human life: tools are used to extend the reach, to amplify the physical strength, and to overcome many limitations induced by the human anatomy. With the advent of autonomous and highly redundant humanoid robots such as iCub [1], machines begin to display an unprecedented dexterity and start to feature very flexible motor capabilities with a high precision. Because of their humanoid anatomy such robots are expected to handle tools in a way similar to humans in a large variety of tasks. A predefined parametrization or hard-coding of arbitrary constraints introduced by a tool is not feasible in this scenario. Learning of the skill will be more efficient than a situation dependent reprogramming. But can humanoid robots exhibit human-like behavior when dealing with tools? And if so, how can this emergence help to understand human behavior?

The implementation of control strategies for robots via computational models acting in flexible environments has developed an outstanding history in recent years. Several machine learning techniques have been applied very successfully to specific inverse kinematics problems [2]. Under the notion of expendable or adaptive body schemata, several studies investigate how motor and control knowledge can be relearned for the case of tool-use [3, 4, 5]. Bi-manual pose-constraint movements for tool-use were main subject in [6], but without a focus on the multiplicity of different solutions. The integration of arbitrary constraints has been analyzed for controlling specific actions, e.g. by Howard et al. [7]. However, learning the incorporation of arbitrary constraints and resolving problems which high redundancy into voluntary control is not well investigated.

One of the most promising candidates for computational motor control and emergence of complex and human-like behavior is the dynamical systems approach. It emphasizes motor behavior as a process of self-organization between the robot and its environment. Besides their undoubted modeling power, finding appropriate dynamical systems for a given complex behavioral phenomenon is difficult due to their inherent unpredictability. Thus, modeling is either left to the intuition of the expert or reduced to the baseline behaviors in order to solve the problem with focus on practicability. This course of action almost eliminates all unexpected emergent behavior which is of great interest in sciences building models for cognition and therefore undesirable. In many cases, cognitive effects are explicitly integrated with the use of human data in the learning process or manual tuning of the model which prevents a deeper understanding of the emergence of intelligent behavior. Programming by demonstration is such a technique to bootstrap learning for robots with the help of a human demonstrator [8, 9]. Despite the impressive success of programming by demonstration in many applications like robotics [10, 11], it is not surprising that systems trained in this manner show human-like behavior. But is it also possible for a computational framework to emerge human-like behavior when learning from robot data which was produced by a robot inspired by human-beings? And if so, what kind of behavior is expected to emerge when robots learn to handle tools in a way similar to humans?

It is well known that reaching movements towards tools require a series of complex transformations between sensory and motor coordinate systems. Despite the possibly infinite number of movements, only those with a high degree of structure and invariance in the kinematical parameters remain as candidates for movement selection. Accumulated experimental evidence suggests the motor hysteresis effect (MHE) [12]. This effect was first shown by Rosenbaum in the early nineties [13, 14] in experimental setups concerning motor control strategies where the participants were supposed to perform grasp and manipulation tasks. Such tasks are carried out in the context of ongoing sequences of behavior: ongoing grasp selection was highly influenced by grasps used in the previous trial. Besides many plausible explanations Rosenbaum argued that movement generation causes less cognitive load when generated by small variations to the previous plan than the creation from scratch [15] causing better energy efficiency. This effect has been reproduced in a number of experiments on humans [16, 14, 17].

In this paper, we propose a flexible tool-use framework for the humanoid robot iCub to perform reaching movements and will focus on complex emergent behavior induced by learning. Many practical manipulation tasks can be treated as imposing a certain constraint on the motion of the robot's hands. Examples are use of a stick or a steering wheel, where the hands become coupled with respect to both orientation and position. Inherent in the use of tools by humanoids is the high redundancy of the manipulation scenario. There are several different ways to handle a tool, either caused by its construction or the humanoid robot, that need to be taken into account. We employ a framework consisting of two coupled extreme learning machines (ELM) [18] for learning which structurally

separates the tool from the kinematics of the robot. This model will be used to induce dynamical systems programmed by a scheme called state prediction (SP) introduced in particular for recurrent neural networks in [19]. The geometry of the given tool and the kinematical structure of the robot are only implicitly available through training examples which can be obtained by kinesthetic teaching or through motor babbling.

The experiments will reveal that the proposed dynamical system approach is sufficient to encode the infinite number of grasping solutions and that a reproduction of the constraint is achieved when generalizing to unseen targets. We additionally show that motor hysteresis emerges as control strategy in the proposed framework without explicit incorporation when using the iCub data.

## 2 Neural Learning Approach: Extreme Learning Machine and State Prediction

We first introduce the neural model implementing the dynamical system. Then, programming of multi-stable dynamics into the neural model is presented.

### 2.1 Network Architecture: Extreme Learning Machine

We consider the network architecture depicted in Fig. 1 called extreme learning machine (ELM) [18]. The network comprises three different layers:  $\mathbf{u} \in \mathbb{R}^I$  collects the input,  $\mathbf{h} \in \mathbb{R}^R$  the hidden, and  $\mathbf{y} \in \mathbb{R}^O$  the output neurons. The input is connected to the hidden layer through the input matrix  $W^{\text{in}} \in \mathbb{R}^{R \times I}$  which remains fixed after random initialization and semantically separates the input  $\mathbf{u}^k = (\mathbf{p}, \mathbf{x}^k)^T$  into parameters  $\mathbf{p} \in \mathbb{R}^P$  and network state  $\mathbf{x}^k \in \mathbb{R}^X$  at time step  $k$ , where the input is satisfying  $P + X = I$ . The read-out is linear in the parameters given by the matrix  $W^{\text{out}} \in \mathbb{R}^{O \times R}$  subject to supervised learning. The calculation for the  $i$ th output neuron for input  $\mathbf{u}^k$  is thus given by:

$$y_i(\mathbf{u}^k) = \sum_j W_{ij}^{\text{out}} f(a_j \sum_n W_{jn}^{\text{in}} u_n^k + b_j) \quad (1)$$

where  $a_j, b_j$  are slope and bias parameterizing the component-wise applied Fermi function  $f(x) = \frac{1}{1+e^{-x}}$ , and  $j = 1 \dots R$  is the index of the hidden layer neuron. The output of the ELM is then used to compute the next state of the network:

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \Delta t \cdot \mathbf{y}(\mathbf{u}^k) , \quad (2)$$

where  $\Delta t$  is a time constant for discretization of the continuous dynamics,  $k$  is the current time step of the iteration, and  $\mathbf{y}(\mathbf{u}^k)$  is the output of the ELM with

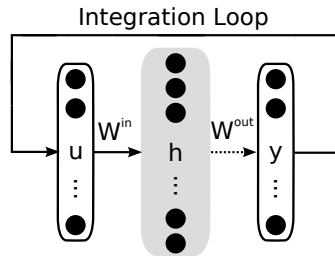


Figure 1: ELM with integration loop.

input  $\mathbf{u}^k$ . The initial state of the network is denoted by  $\mathbf{x}^0$ .

## 2.2 Programming Dynamics: State Prediction

In order to imprint suitable dynamics into the ELM, the programming dynamics approach recently introduced in [19] called state prediction (SP) is used and adapted for the model shown in Fig. 1. This approach extends the training data with synthesized sequences to facilitate attraction to the data samples. Sequences  $\hat{\mathbf{x}}_n^s(k) = \mathbf{x}_n + (1 - \frac{k}{K})^2 \nu^s$  are generated for each input sample pair  $\mathbf{p}_n$  and  $\mathbf{x}_n$  in the data set;  $s$  denotes the index of the  $S$  sequences,  $\nu^s$  is a small perturbation,  $k = 1, \dots, K$  are the time steps, and  $n = 1, \dots, N$  is the index of the training samples. This imprints  $\mathbf{x}_n$  as a valid attractor of the dynamical system.

The corresponding hidden network states  $\mathbf{h}(\mathbf{p}_n, \hat{\mathbf{x}}_n^s(k))$  for each sequence are collected. Attraction to the desired outputs is enforced by mapping states  $\mathbf{h}(\mathbf{p}_n, \hat{\mathbf{x}}_n^s(k))$  to outputs  $\hat{\mathbf{x}}_n^s(k+1) - \hat{\mathbf{x}}_n^s(k)$ . This can be accomplished efficiently by applying ridge regression:

$$W^{\text{out}} = (H^T H + \alpha \mathbb{I})^{-1} H^T \hat{Q}, \quad (3)$$

where

$$H = \begin{pmatrix} \mathbf{h}(\mathbf{p}_1, \hat{\mathbf{x}}_1^1(1))^T \\ \vdots \\ \mathbf{h}(\mathbf{p}_n, \hat{\mathbf{x}}_n^s(k))^T \\ \vdots \\ \mathbf{h}(\mathbf{p}_n, \hat{\mathbf{x}}_N^S(K-1))^T \end{pmatrix} \text{ and } \hat{Q} = \begin{pmatrix} \hat{\mathbf{x}}_1^1(2)^T - \hat{\mathbf{x}}_1^1(1)^T \\ \vdots \\ \hat{\mathbf{x}}_n^s(k+1)^T - \hat{\mathbf{x}}_n^s(k)^T \\ \vdots \\ \hat{\mathbf{x}}_N^S(K)^T - \hat{\mathbf{x}}_N^S(K-1)^T \end{pmatrix} \quad (4)$$

collects the hidden states and the desired outputs in the respective matrices and  $\alpha$  modulates the trade-off between weight decay and training error. The latter is added for the sake of a better generalization [20, 21] - a well established procedure in neural networks learning (see, e.g. [22]). The perturbations are chosen randomly by a multivariate Gaussian distribution with  $\Sigma = 0.001 \cdot \mathbb{I}$  centered around the desired point of attraction  $\mathbf{x}_n$ .

The presented approach differs from the original work in [19] by using a relative encoding of the output. The network therefore induces continuous dynamics whose discretization is explicitly controlled through  $\Delta t$ . We yield  $S = 7$  sequences with  $K = 5$  time steps for training in the experiments, which enlarges the training set by a factor of  $S \cdot (K - 1)$ . However, learning is still feasible due to the efficient learning scheme of the ELM approach.

## 3 Dynamical Tool-Use for iCub

Given a robot, the forward kinematics function  $F : \mathbb{R}^m \rightarrow \mathbb{R}^n$  is uniquely defined. It converts a set of joint angles  $\mathbf{q} \in \mathbb{R}^m$  into the corresponding end effector configuration  $\mathbf{e} \in \mathbb{R}^n$ . An inverse kinematics function  $F^{-1}$  of a robot is defined

by the forward kinematics in the following equation:  $F(F^{-1}(\mathbf{e})) = \mathbf{e}$ . It maps a configuration of the end effector to the corresponding joint angles of the robot. Note that there is no unique inverse kinematics function in the case of iCub because of the redundancy. A tool is interpreted as a function  $C : \mathbb{R}^p \rightarrow \mathbb{R}^n$  which maps the actual tool configuration  $\mathbf{p} \in \mathbb{R}^p$  to an end effector configuration  $\mathbf{e}$  such that valid grasps  $\mathbf{q}$  are selected.

### 3.1 Tools as Kinematic Constraints for iCub

The end effector configuration of the humanoid robot iCub contains two subsets: the left and the right hand. The hand center points  $\mathbf{p}^{\text{L,R}}$  are described in Cartesian coordinates  $x$ ,  $y$  and  $z$  with respect to the world coordinate system. The orientations of the hands are expressed as spatial orientations of the grasp axis  $\mathbf{d}^{\text{L,R}}$  with a normalization  $\|\mathbf{d}^{\text{L,R}}\| = 1$ , see Figure 2 (right). Thus the end effector configuration is a  $n$ -dimensional ( $n = 12$ ) input variable

$$\mathbf{e} = (\mathbf{e}^{\text{L}}, \mathbf{e}^{\text{R}})^T = (\mathbf{p}^{\text{L}}, \mathbf{d}^{\text{L}}, \mathbf{p}^{\text{R}}, \mathbf{d}^{\text{R}})^T \in \mathbb{R}^n, \quad (5)$$

where L stands for left and R for right. The iCub cartesian solver, which will be used to generate the ground truth examples, operates on two times seven degrees of freedom ( $m = 14$ ). Each arm is moved separately by controlling two rotational degrees of freedom in the shoulder, two in the elbow and three in the wrist. However, the arms are coupled through the hip and whole body motion but without consideration in this paper.

A tool is defined by a constraint function  $C : \mathbb{R}^p \rightarrow \mathbb{R}^n$ , which restricts the hand configurations to a subset of the reachable task space. The constraint is parameterized by the tool’s position and tool’s orientation in Euler angles:  $\mathbf{p} = (s_1, s_2, s_3, \theta_1, \theta_2, \theta_3) \in \mathbb{R}^p$ . Given this  $p$ -dimensional ( $p = 6$ ) input vector  $\mathbf{p}$  the set of suitable bi-manual end effector configurations  $C(\mathbf{p}) = E_{\mathbf{p}} \subset \mathbb{R}^n$  is uniquely defined through the variety of possible grasps. Such a set is transformed by the inverse kinematics to a set of corresponding joints  $Q_{\mathbf{p}} \subset \mathbb{R}^m$  such that  $F(Q_{\mathbf{p}}) = E_{\mathbf{p}}$ . For the control of the robot, a mapping  $T : \mathbb{R}^p \rightarrow \mathbb{R}^m$ , choosing a robot’s joint configuration  $T(\mathbf{p}) = \mathbf{q} \in Q_{\mathbf{p}}$  by coupling the tool  $C$  and the inverse kinematics  $F^{-1} : \mathbb{R}^n \rightarrow \mathbb{R}^m$  is needed:

$$T(\mathbf{p}) = F^{-1}(C(\mathbf{p})) = F^{-1}(\mathbf{e}) = \mathbf{q} \in Q_{\mathbf{p}}. \quad (6)$$

Fig. 3 shows some examples of iCub grasping a one meter long stick bi-manually controlled by the introduced dynamical system. Given the position and orientation of the tool as input variable  $\mathbf{p}$ , the introduced neural network architecture (after learning as described below) computes joint angles  $\mathbf{q}$  to grasp the stick appropriately. In Fig. 3 (left) iCub performs a bi-manual grasp with the same end effector configuration  $\mathbf{e} \in E_{\mathbf{p}}$  for  $\mathbf{p} = (0, \dots, 0)^T$  but with different

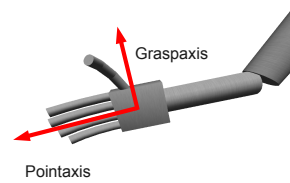


Figure 2: iCub’s right hand.

joint solutions. In Fig. 3 (right) iCub produces two grasps with both different hand center points and grasp axes:  $\mathbf{p}_1^{L,R} \neq \mathbf{p}_2^{L,R}$  and  $\mathbf{d}_1^{L,R} \neq \mathbf{d}_2^{L,R}$ .

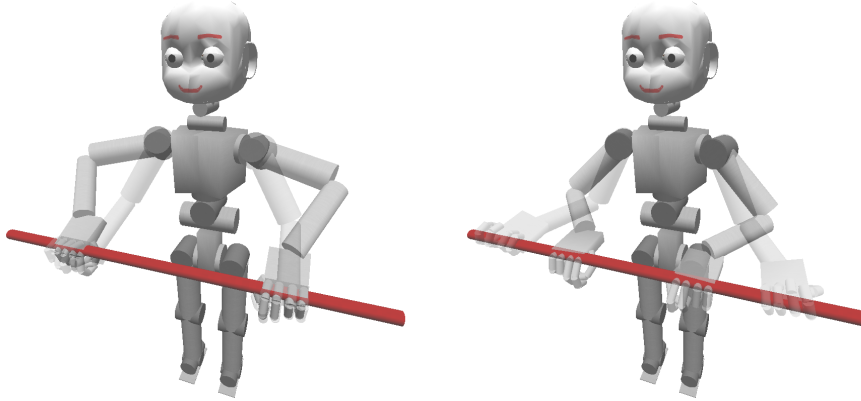


Figure 3: iCub grasping a stick bi-manually in four different ways. Different solutions through the redundancy of iCub (left). Different solutions through the redundancy of the stick manipulation scenario (right).

### 3.2 Tool-Use: Building Neural Representations

At first, the neural representation of the tool is constructed by the ELM model described in Sect. 2. Only the left hand is modeled for simplicity. This reduces the forward kinematics to the forward function of the left arm:  $F^L(\mathbf{q}^L) = \mathbf{e}^L$ .

The parametrization  $\mathbf{p} = (s_1, s_2, s_3, \theta_1, \theta_2, \theta_3) \in \mathbb{R}^6$  for the tool ELM represents the tool configuration, which consists of the three-dimensional position  $\mathbf{s} = (s_1, s_2, s_3) \in \mathbb{R}^3$  in Cartesian coordinates and the orientation  $\theta = (\theta_1, \theta_2, \theta_3) \in \mathbb{R}^3$  in roll-pitch-yaw angles. The state  $\mathbf{x} = \mathbf{e}^L = (\mathbf{p}^L, \mathbf{d}^L) \in \mathbb{R}^6$  of this ELM is consisting of the end effector position  $\mathbf{p}^L$  and the coordinates of the grasp axis  $\mathbf{d}^L$ . The network is used to produce valid grasps of the tool with respect to the tool’s configuration. While the dynamics of the ELM are iterated, a trajectory  $\mathbf{x}^0 \rightarrow \dots \rightarrow \mathbf{x}^k \rightarrow \dots \rightarrow \mathbf{x}^\infty$  towards a proper grasp is generated while convergence of the network to a fixpoint attractor state.

The goal is to use such states as targets for an arbitrary inverse kinematics solver in order to maintain the joint angles for the robot used in the tool manipulation scenario. This solver could be a simple non-dynamic function as in [6] or a model suited to represent the high redundancy involved in humanoid robotics; such as in [19]. One advantage of this modularly approach is that the modules for tool or robot kinematics are easily exchangeable and the dynamics become interpretable in comparison to a direct encoding from tool configuration to joint angle trajectories.

However, we also use the ELM model to represent the inverse kinematics described in Sect. 2. This ELM gets the state of the first ELM  $\mathbf{e}^L$  as parametriza-

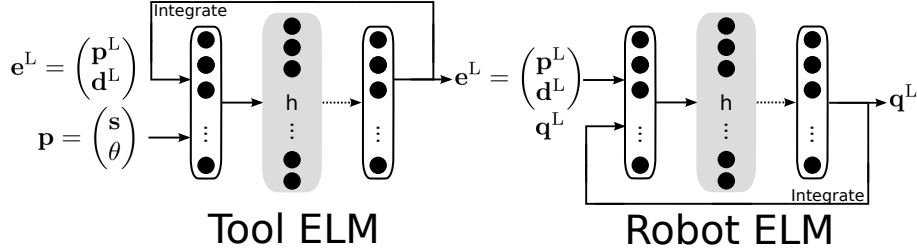


Figure 4: Two ELMs to implement the tool kinematics and the robot’s inverse kinematics separately.

tion. The state of this ELM is collection of joints  $\mathbf{q}^L$ , which is supposed to converge to a solution of the inverse kinematics. The modular architecture for neurally representing tool-use is depicted in Fig. 4. The following update equations are obtained for a time dependent tool configuration  $\mathbf{p}^k$ :

$$\mathbf{e}^{L,k+1} = \mathbf{e}^{L,k} + \Delta t \cdot \hat{\mathbf{v}}_1(\mathbf{p}^k, \mathbf{e}^{L,k}) \quad (7)$$

$$\mathbf{q}^{L,k+1} = \mathbf{q}^{L,k} + \Delta t \cdot \hat{\mathbf{v}}_2(\mathbf{e}^{L,k}, \mathbf{q}^{L,k}) , \quad (8)$$

where  $\hat{\mathbf{v}}_{1,2}$  are the output of the first and the second ELM. In addition, the vector of the direction is normalized:  $\|\mathbf{d}^{L,k}\| = 1 : \forall k$ .

In order to train the models, a data set doublet  $D = (D^1, D^2)$  is needed.  $D^1 = (\mathbf{p}_i, \mathbf{e}_i^L)_{i=1, \dots, N_{\text{tool}}}$  contains a set of tool configurations  $\mathbf{p}_i$  and a set of corresponding end effector configurations  $\mathbf{e}_i^L$  when grasping the tool.  $D^2 = (\mathbf{e}_i^L, \mathbf{q}_i^L)_{i=1, \dots, N_{\text{robot}}}$  comprises end effector configurations  $\mathbf{e}_i^L$  and the respective joint angles  $\mathbf{q}_i^L$ . Supervised learning is efficiently done by separating the learning for the modules and using the state prediction approach. The following sections will suggest methods to implement the dynamics of an one meter long stick.

## 4 iCub Handling a Tool: The Stick Example

This section describes the training for the dynamical systems driving iCubs kinematics towards tool-use. A stick is used as an example in the scenario. We additionally show that also the inverse kinematics of iCub can be learned in the same framework.

### 4.1 Learning the Tool Kinematics of a Stick

In order to learn the tool kinematics of a stick, data is generated by uniformly sampling the space of possible tool configurations in the range  $s_{1 \dots 3} \in [-0.1m, 0.1m]$  and  $\theta_{1 \dots 3} \in [-30^\circ, 30^\circ]$  with  $N_{\text{tool}} = 1000$  samples. For each tool configuration, one valid grasp is randomly placed on the one meter long stick given by the end effector position  $\mathbf{p}^L$ , the grasp axis orientation  $\mathbf{d}^L$ , and the corresponding joint angles  $\mathbf{q}^L$ .



We additionally enforce a morphology of the tool which forces iCub at the left end of the stick ( $-0.5m$  to  $-0.25m$ ) to point with its thumb towards the right end and at the right end of the stick ( $0.25m$  to  $0.5m$ ) to point with its thumb towards the left end (e.g. by an ergonomic handle). This is done by representing only the respective solution for the grasp axis at the ends of the stick. In between is a range of indifference ( $-0.25m$  to  $0.25m$ ) where both grasp axes were presented during learning.

The data for training are obtained through the iCub simulation, but it is also possible to obtain the data by kinesthetic teaching.

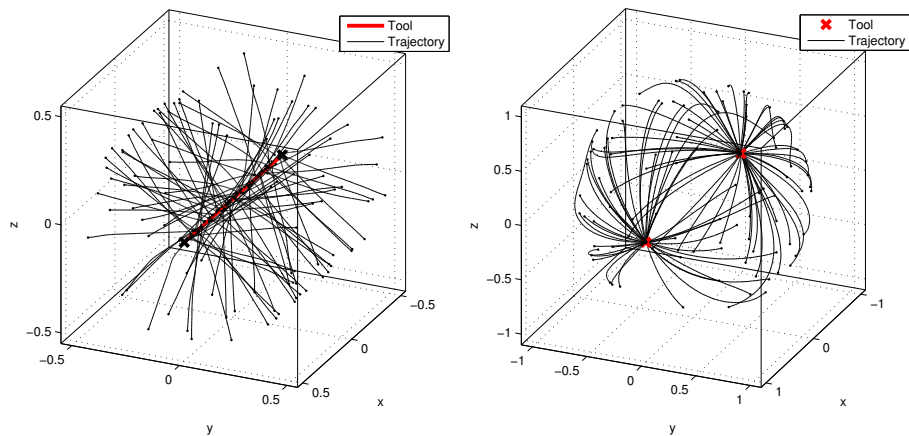


Figure 5: Position and directions of stick grasps obtained by the neural representation.

The left part of Fig. 5 shows the implemented continuous attractor with finite length representing the tool for configuration  $\mathbf{s} = \theta = (0, 0, 0)^T$ . It is demonstrated that various initial end effector configurations converge to tool grasp positions on an almost straight trajectory. The figure demonstrates that the network connects the attractors to a continuous manifold, although only fixpoint attractors are implemented by SP. Fig. 5 (right) shows the two possible grasp axis orientations - either iCub’s thumb points towards the left or the right end of the stick. Note, that the grasp axis trajectories are in the space of the unit sphere due to the restriction  $\|\mathbf{d}^L\| = 1$ .

Fig. 6 illustrates trajectories for the tool ELM for different parameter settings projected onto the x-z-plane. A finite-length continuous attractor is build (red stroke in the figure) which can be moved and rotated by changing the parameterization in the input vector. Parameters are  $\mathbf{s} = (-0.05, 0, 0)^T$  and roll angle =  $20^\circ$  (left) and  $\mathbf{s} = (0.1, 0, 0.15)^T$  and roll angle =  $-25^\circ$  (right).

Tab. 7 shows the average error and its standard deviation for 100 grasp trials. For each trial, a tool configuration in the range of the training data and a starting points uniformly initialized in  $[-1, 1]^3$  is given. After 1000 convergence steps,

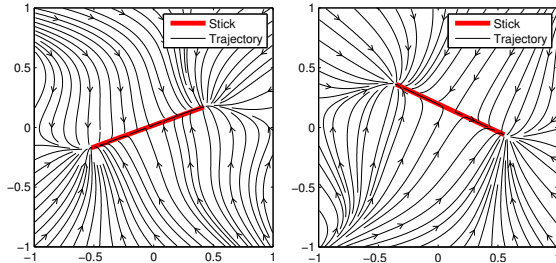


Figure 6: Dynamic flow fields of the tool ELM projected to the x-z-plane.

Left Solution	
Pos. [cm]	$0.47 \pm 0.18$
Ori. [ $^\circ$ ]	$0.45 \pm 0.17$
Right Solution	
Pos. [cm]	$0.46 \pm 0.19$
Ori. [ $^\circ$ ]	$0.45 \pm 0.18$

Figure 7: Mean and standard deviation of the position and orientation error on test data.

the distance from the end effector position to the stick and the distance between grasp axis and stick axis are calculated and recorded. The errors are separated into left and right solution which means that the thumb is pointing towards the respective side. The table demonstrates that the errors are small enough for a grasping since the average error is significantly lower than a typical diameter for a stick used for iCub.

#### 4.2 Learning the Inverse Kinematics of iCub

The inverse kinematics mapping will also be implemented by a dynamically applied ELM as described in Sect. 2. The respective parameterization is given by the state of the ELM representing the tool dynamics, which is the target of the end effector:  $\mathbf{e}^L = (\mathbf{p}^L, \mathbf{d}^L)$ . The state of this ELM is programmed such that it converges to a solution of the inverse kinematics:  $\mathbf{q}^L : F(\mathbf{q}^L) = \mathbf{e}^L$ .

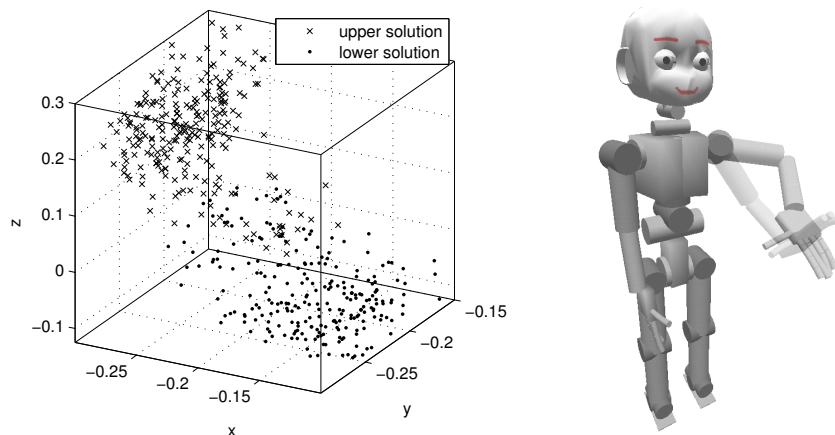


Figure 8: Data set used for the learning of iCub's inverse kinematics mapping (left). iCub performing the same grasp with two different solutions (right).

The used network approach is sufficient to prevent averaging of ambiguous data samples. Therefore, coping with the high redundancy of the inverse kinematics mapping of iCub. The data set used for learning iCub’s inverse kinematics mapping was produced by the simulation and contains  $N_{\text{robot}} = 1000$  samples. The data set contains two different solutions for this mapping, achieved by setting the point axis to an angle of either  $\frac{\pi}{4}$  or  $-\frac{\pi}{4}$  with respect to the y-axis. The solutions are called upper solution for  $\frac{\pi}{4}$  and lower solution for  $-\frac{\pi}{4}$ . Fig. 8 (left) shows the samples of the data set, demonstrating that the upper solution is mainly present in the upper part of iCub’s working space and the lower solution in the lower part. The right side of Fig. 8 shows iCub performing the same grasp with the two different solutions.

In order to verify the generalization ability of the presented approach, 300 additional training samples are produced in the same way. Tab. 1 comprises the errors for an ELM averaging over this 300 data samples. The accuracy of the different solutions are assigned separately. Furthermore, the table also contains the training errors which are slightly lower than the test errors. Besides the fact, that the upper solution is approximated more precisely than the lower solution, both errors stay below 1 cm in average. The introduced model is therefore sufficient to cope with the inverse kinematics mapping of the humanoid robot iCub.

Pos. [cm]	Upper Sol.	Lower Sol.
Train	$0.11 \pm 0.13$	$0.67 \pm 1.18$
Test	$0.23 \pm 0.27$	$0.83 \pm 1.34$
Ori. [ $^{\circ}$ ]	Upper Sol.	Lower Sol.
Train	$0.27 \pm 0.37$	$1.63 \pm 3.01$
Test	$0.45 \pm 0.67$	$2.01 \pm 3.23$

Table 1: Mean and standard variation of the position and orientation error on training and test data.

## 5 Emergence of Motor Hysteresis from Robot Data

As already mentioned, Rosenbaum [13, 14] performed a remarkable experiment dealing with terminal postures in grasping and object manipulation, where the end-state comfort effect and the motor hysteresis effect (MHE) were revealed. Participants were supposed to grasp a rod that was horizontally supported by a cradle. The goal was to place its right or left end against one of 14 targets, while the targets were arranged vertically on the shelves of a bookcase and had to be contacted in either ascending or descending order. High targets corresponded to low numbers and low targets were denoted by high numbers. Satisfying the end-state comfort effect, about 80 percent of the subjects used the overhand grip when the goal was to bring the right end of the bar to the high targets, and 90 percent used the underhand grip when the goal was to bring the right end to the low targets. Besides this, the end-state was significantly affected by the history of previously performed grasps; which is of main interest in this section.

## 5.1 Motor Hysteresis

It was shown that the MHE observed in human movement studies is subject to the fact that such tasks are carried out in the context of ongoing sequences of behavior and can be observed in experimental setups concerning motor control strategies. It was demonstrated that ongoing grasp selection was highly influenced by grasps used in the previous trial: Participants persisted in using an overhand grasp in the ascending target condition and an underhand grasp in the descending target condition - named MHE by Kelso et al. [12]. The explanation for the MHE given by Rosenbaum et al. [13] postulates a range of indifference, where it is equally comfortable for the participants to use either an overhand or an underhand grasp. Hence, a new movement plan can be generated by small variations to the previous plan, causing less cognitive load than the creation from scratch [15]. Rosenbaum formulated the MHE in the following words: “the sequential effect can be summarized by saying that subjects persisted in using the grip they used before” [13]. The MHE and the end-state comfort effect were studied in further detail for continuous task spaces in [23].

## 5.2 Motor Hysteresis in the Computational Model

Interestingly, the neural approach introduced in this paper shows an effect in the tool manipulation scenario which is similar to those for humans. The plots

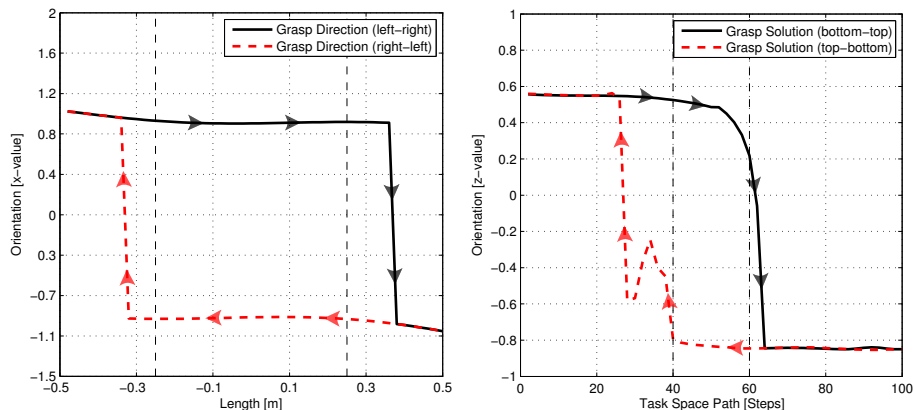


Figure 9: Position and directions of stick grasps obtained by the neural representation.

in Fig. 9 summarizes two experiments with iCub.

In the first experiment, iCub is supposed to move its left hand from the left side (indicated by  $-0.5m$ ) of the stick towards the right side (indicated by  $0.5m$ ) and back. This is done by changing  $e_1^L$  linearly from  $-0.5m$  to  $0.5m$  and back while iterating the update Eqs. (7) and (8). We recorded  $d_1^L$  in this iteration phase and show the results in Fig. 9 (left). The morphology of the tool forces

iCub at the left end of the stick to point with its thumb towards the right end and at the right end of the stick to point with its thumb towards the left end. The plot shows that in the middle region both solutions are encoded in the network and stable but selection is solely determined by the previous state. A clear hysteresis effect shows up, which is similar to the motor hysteresis effect triggered by the tool and following the psychophysical intuition that “a subject persisted in using the grip it used before” ([13]).

The second plot (Fig. 9, right) shows an experiment where iCub is supposed to move its left hand on a target trajectory. The trajectory started in  $\mathbf{e}^{L,0} = (-0.11; -0.2; -0.1)^T$ , passes  $\mathbf{e}^{L,50} = (-0.225; -0.2; 0.025)^T$  and ends at  $\mathbf{e}^{L,100} = (-0.29; -0.2; 0.29)^T$ . These points correspond to steps 0, 50, and 100, respectively. The z-values of the hand orientations  $\hat{e}_6^{L,k} = F_6^L(\mathbf{q}^{L,k})$  indicating the selected inverse kinematics solution are recorded and visualized in Fig. 9 (right). When looking at the data in Fig. 8, lower solutions for the inverse kinematics occur with a higher probability than upper solutions in regions where the z-value is high and vice versa. This is due to iCub morphology. Thus iCub prefers upper solutions (1-25) in the upper part of the task space and lower solutions (65-100) in the lower part. In the middle of the workspace (25-65), both solutions are represented in the data and successfully imprinted into the neural architecture such that iCub persists in using the already used solution for the inverse kinematics. The selected solution depends mainly on the history of the previous performed grasps. Interestingly, this effect is not explicitly implemented but arises naturally through the data production which is in turn affected by the robot's morphology.

## 6 Conclusion

We present a neural learning approach to cope with bi-manual tool-use for the humanoid robot iCub. Despite the big data sets obtained by SP, learning stays efficient due to the ELM scheme. The idea is to separate the representation of the tool and the robot's inverse kinematics in order to profit from the modular architecture in terms of substitutability and interpretability. It is shown that a tool and the inverse kinematics can be stored in the network to a high accuracy. Also the high redundancy involved in the task is represented, which can be resolved dynamically.

Interestingly, the introduced approach also shows motor hysteresis effects typically observable in psychophysical experiments on human manipulation tasks. It is demonstrated that this typical human-like behavior arises in a natural way without explicit integration into the learning.

## References

- [1] Giulio Sandini, Giorgio Metta, and David Vernon. The icub cognitive humanoid robot: An open-system research platform for enactive cognition. *50 Years of Artificial Intelligence*, pages 358–369, 2007.

- [2] Aaron D’Souza, Sethu Vijayakumar, and Stefan Schaal. Learning inverse kinematics. In *Proc. IROS*, pages 298–303, 2001.
- [3] Cota Nabeshima, Yasuo Kuniyoshi, and Max Lungarella. Adaptive body schema for robotic tool-use. *Advanced Robotics*, 20(10), 2006.
- [4] M. Hersch, E. Sauser, and A. Billard. Online learning of the body schema. *International Journal of Humanoid Robotics*, 5(2), 2008.
- [5] Alexander Stoytchev. Computational model for an extendable robot body schema. Technical Report GIT-CC-03-44, Georgia Institute of Technology, 2003.
- [6] Klaus Neumann, Matthias Rolf, Jochen J. Steil, and Michael Gienger. Learning inverse kinematics for pose-constraint bi-manual movements. *Proc. SAB*, pages 478–488, 2010.
- [7] Matthew Howard, Stefan Klanke, Michael Gienger, Christian Goerick, and Sethu Vijayakumar. A novel method for learning policies from variable constraint data. *Autonomous Robots*, 27(2), 2009.
- [8] S. Schaal. Is imitation learning the route to humanoid robots? *Trends in Cognitive Sciences*, 3(6):233–242, 1999.
- [9] A. Billard, S. Calinon, R. Dillmann, and S. Schaal. Robot Programming by Demonstration. In Bruno Siciliano and Oussama Khatib, editors, *Springer Handbook of Robotics*, chapter 60, pages 1371–1394. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.
- [10] S.M. Khansari-Zadeh and A. Billard. Learning stable non-linear dynamical systems with gaussian mixture models. *IEEE Transaction on Robotics*, 27(5):943–957, 2011.
- [11] M. Hersch and A. G. Billard. A biologically-inspired controller for reaching movements. In *Proc. BIOROB*, pages 1067–1071, 2006.
- [12] J. A. S. Kelso, J. J. Buchanan, and T. Murata. Multifunctionality and switching in the coordination dynamics of reaching and grasping. *Human Movement Science*, 13(1):63–94, 1994.
- [13] David A. Rosenbaum and Matthew J. Jorgensen. Planning macroscopic aspects of manual control. *Human Movement Science*, 11(12):61–69, 1992.
- [14] David A. Rosenbaum, Frank Marchak, Heather Jane Barnes, Jonathan Vaughan, James D. Siotta, and Matthew J. Jorgensen. Constraints for action selection: Overhand versus underhand grips. In *Attention and Performance*, volume 13, pages 321–345. Lawrence Erlbaum Associates, 1990.
- [15] David A. Rosenbaum, R. G. Cohen, S. A. Jax, D. J. Weiss, and R. Van der Wel. The problem of serial order in behavior: Lashley’s legacy. *Human Movement Science*, 26(4):525–554, 2007.

- [16] Martin W. Short and James H. Cauraugh. Planning macroscopic aspects of manual control: End-state comfort and point-of-change effects. *Acta Psychologica*, 96(12):133–147, 1997.
- [17] Ronald G. Marteniuk and Eric A. Roy. The codability of kinesthetic location and distance information. *Acta Psychologica*, 36(6):471–479, 1972.
- [18] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew. Extreme learning machine: A new learning scheme of feedforward neural networks. In *International Joint Conference on Neural Networks*, Budapest, Hungary, 2004.
- [19] R.F. Reinhart and J.J. Steil. Neural learning and dynamical selection of redundant solutions for inverse kinematic control. pages 564–569, 2011.
- [20] A. N. Tikhonov and V. Y. Arsenin. Solutions of ill-posed problems. *soviet Math. Dokl.*, (4):1035–1038, 1963.
- [21] A. N. Tikhonov. Solution of incorrectly formulated problems and the regularization method. *W. H. Winston, Washington, D. C.*, 1977.
- [22] C. M. Bishop. Training with noise is equivalent to tikhonov regularization. *Neural Computation*, 7:108–116, 1994.
- [23] C. Schuetz, M. Weigelt, D. Odekerken, T. Klein-Soetebier, and T. Schack. Motor control strategies in a continuous task space. *Motor Control*, 15:321–341, 2011.