Encoding Complete Body Models Enables Task Dependent Optimal Behavior

Oliver Herbort and Martin V. Butz

***Abstract*— Many neural network models of (human) motor learning focus on the acquisition of direct goal-to-action mappings, which results in rather inflexible motor control programs. We propose a neural network architecture (SURE_REACH) that acquires complete body models through unsupervised learning. It encodes redundancy on the kinematic and on the motor command level in order to exert highly flexible, task-dependent optimal control. This paper shows that our approach accounts for two forms of effective human behavior based on exploiting kinematic redundancy. First, depending on the starting posture, hand targets are pursued in different ways optimizing movement efficiency. Second, the arm posture at the end of a movement can be aligned anticipatorily to facilitate a subsequent movement. A discussion of computational implications and relations to behavioral and neurophysiological findings concludes the paper.**

I. Introduction

Any behaving system needs to be able to control its own body goal-directedly. This is especially true for animals and humans, since motor behavior is the only option to interact with the world. A seemingly very simple form of goal directed behavior is moving ones hand to a particular position, such as reaching for an object or pointing. Still, reaching requires that a goal representation, for example the desired hand position in visual space, is transformed into a series of motor commands that move the hand swiftly to the goal. The neural representations of these goal-to-action mappings are termed internal *inverse models* [1]. In recent decades, motor cortical and cerebellar neural network models have been proposed for the acquisition of such inverse models, e.g. [2], [3]. They differ in many aspects but have one thing in common: They gather information during motor learning and aggregate the information by encoding a single, preferably optimal action for each potential goal (and each body state). While this aggregation allows for a compact representation, it also yields several severe limitations.

First, storing only the best action for pursuing each goal requires an environment in which the optimal action always stays the same. However, in most circumstances the optimality criteria, which determine the optimal action, change over time. For example, sometimes a movement has to be carried out as quickly as possible, whereas at other times it requires great accuracy. If the action that was optimal during motor learning was the only one represented in an internal model, then movements cannot be adjusted flexibly to changing optimality criteria. Even worse, if only one action is stored to pursue a given goal, and this action cannot be carried out any more, due to, for example, injuries or obstacles, the goal cannot be at all reached, not even in a suboptimal way.

Several models have shown the benefits of storing multiple actions. The MOSAIC model is able to quickly adjust to new dynamical contexts, because an array of different controllers is trained [3]. The controllers that are most suitable in the current context can be activated on the fly. However, no redundant actions within a single context are encoded. A more radical approach has been taken by the *posture based motion planing theory* [4]. It does not rely on trained goal-to-action mappings, but uses a complete body model which provides all possible actions and their properties (e.g. distance to the goal, movement cost). Hence, an action can be selected that is optimal for the current task. The model accounts in detail for a wide range of human behavior, such as grasping or obstacle avoidance. However, the model is highly abstract and the body model underlying action generation is not learned, but rather prewired. In conclusion, these and other approaches [5] show that representing more than just the optimal action for each goal enables much higher robustness and flexibility.

Besides this behavioral inflexibility, the storage of gathered information into one goal-to-action mapping assumes that all potential goals are already known during motor learning. In most models, it is assumed that the potential goals are the different sensory states the organism can perceive. The learning mechanisms then strives to find for each sensory state the action that optimally moves the body so that the respective sensory state is actually perceived. Thus, only those potential goals are represented that are anticipated during motor learning. Later on, the controller would have difficulties of processing novel, maybe less constrained goal representations—a capability that would also enhance behavioral robustness and flexibility. On the one hand, the possibility to set an underconstrained goal enables the control of only those aspects of a movement that are relevant for the task, resulting in behavior that is more efficient and less prone to noise [6]. On the other hand, controllers like the MMC model [5] have shown that relying on a prewired complete kinematic body model instead of a goal-to-action mapping enables the processing of underconstrained goals. This capability is especially important for human motor control because most goals are underconstrained due to motor redundancy. To summarize, if a complete body model is available, not only those goals that were anticipated during motor learning can be processed but also many other more general goal representations, thus increasing robustness and

The authors are with the Department of Cognitive Psychology, University of Würzburg, Germany, (phone: +49 931 312808; fax: +49 931 312815; email: oliver.herbort@psychologie.uni-wuerzburg.de).

flexibility.

Most neural network models of motor learning and control aggregate the gathered information compactly in a many-to-one goal-to-action mapping. This enables a compact goal representation but makes motor control inflexible and sub-optimal in an environment that requires the quick adaptation to novel task demands, often from one movement to another. However, approaches that rely on a representation of a complete body model are restricted in their learning capabilities (e.g. [4]). In this paper, we propose a new neural network model of motor learning and control, called SURE_REACH[1], which grounds task-dependent optimal control on a body model, which is acquired through learning experience [7].

In the following, we briefly describe the neural network model. Then, two behavioral findings in the domain of reaching are replicated. First, we show that the final arm state of a movement toward a specific desired hand location depends on the initial arm position, thus minimizing movement costs. Second, we simulate anticipatory behavior in the sense that the end posture of one movement depends on the requirements of a subsequent task, if this task is incorporated into the goal representation. This finding can hardly be accounted for by other neural network models of motor learning because it requires an explicit representation of the redundant solutions of the inverse kinematics. To our knowledge, this task has not been simulated before. A final discussion about the implications of the proposed work concludes the paper.

II. SURE_REACH

SURE_REACH is a modular hierarchical architecture that solves the inverse problem of generating a sequence of motor commands that moves the hand to a desired hand location. It is divided into two modules that are trained with unsupervised, associative learning rules.

First, the *posture memory (PM)* addresses the inverse kinematics problem. It transforms a hand location into a set of arm postures that realize the respective hand location. Second, the *motor controller (MC)* generates motor commands that move the arm toward the goal posture set, provided by PM. Thereby MC is able to generate movements toward redundant, underconstrained goal specifications. Even more so, the postures encoded in the goal representation can be weighted if not all end postures are equally useful outcomes of the movement. MC consists of several motor-command-dependent body models, which encode the movements of the arm in posture space, given a certain motor command is executed.

Before a movement can be performed, MC prepares a state-to-action mapping by means of *dynamic programming* based on the learned body models. This mapping provides suitable motor commands to move a simulated arm from each possible posture toward the desired hand location and can be considered an online generated inverse model. The dynamic programming approach is also one of the key differences to previous models. Whereas other models encode a single inverse model during motor learning, which is used for all reaching movements later on, SURE_REACH generates an individual inverse model for each newly presented target. This enables incorporating task dependent constraints and optimality criteria by adjusting the model used by dynamic programming to the current task's demands. For example, as we have demonstrated elsewhere, SURE_REACH avoids obstacles in hand space, regards novel cost functions, or controls an arm despite a disabled joint—all without having been in either of these situations and without the necessity to relearn [7]. In the following, the applied arm model, body space representations, and neural network structures are briefly presented. Figure 1 shows the basic architecture. A detailed evaluation and discussion of SURE_REACH can be found in [7].

A. Arm Model

To simulate reaching experiments we implemented a model of a three joint planar arm that roughly approximates the kinematic features of a human arm that is restricted to the transverse plane. The lengths of the upper arm, forearm, and hand were $l_1 = 32cm$, $l_2 = 25cm$ and $l_3 = 18cm$, respectively. The shoulder, elbow and wrist joints were allowed to move within $-60° \leq \phi_1 \leq 120°$, $-160° \leq \phi_2 \leq 0°$, and $-80° \leq \phi_3 \leq 60°$, respectively. Two antagonistic "muscles" were attached to each limb. Each muscle was activated by motor commands ranging between $0.0 \leq mc_i \leq 1.0$. To compute the final movement of a joint ϕ_i, activation of antagonistic motor commands was subtracted and the result was multiplied by a gain factor $g = 2.25°$:

$$\phi_i(t+1) = \phi_i(t) + g(mc_{2i-1} - mc_{2i}), i = 1, 2, 3$$

Albeit this arm model is very simple, it has the critical property that most hand locations of the arm can be realized by an infinite number of arm postures.

B. Space Representation

In the architecture, extrinsic hand location space and intrinsic arm posture space are represented. Hand coordinates were encoded by a population of neurons H. Each neuron h_i of H fired if the hand coordinates (x, y) are close enough to the neuron's preferred hand location (h_i^x,h_i^y):

$$h_i = \max(1.0 - \frac{|x - h_i^x|}{3.0}; 0) \cdot \max(1.0 - \frac{|y - h_i^y|}{3.0}; 0)$$

The preferred hand locations were arranged in a $51 \times 26 = 1326$ grid with $3cm$ distance, covering a $150cm \times 75cm$ rectangle, which covered the upper half of the arm's work space. The shoulder joint was centered on the lower line of this rectangle (dashed rectangle in figure 1). Arm postures were encoded in a similar population of neurons P, where each neuron p_i was activated according to the following equation:

$$p_i = \prod_{j=1}^{3} \max(1.0 - \frac{|\phi_j - p_i^{\phi_j}|}{20.0°}; 0),$$

[1]SURE_REACH is an acronym for sensorimotor unsupervised redundancy resolving architecture.

where $p_i^{\phi_j}$ are the preferred joint angles of each neuron p_i, which were arranged in a $10 \times 9 \times 8 = 720$ grid covering the entire posture space. The distance between two adjacent neurons was $20°$. Hence, in both representations only a few neurons were active at the same time, indicating the current location of the hand or the current arm posture.

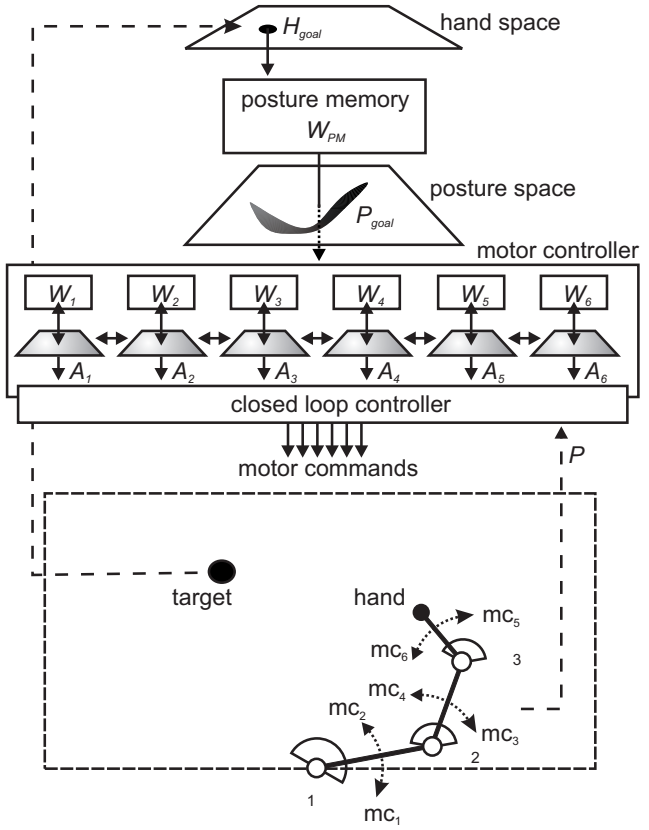

Fig. 1. If an external target appears, it is transformed into a desired hand location H_{goal}. The posture memory converts it into a redundant goal representation in posture space P_{goal}. Starting from P_{goal} the motor controller propagates neural activation through the neuron layers A_i based on the recurrent neural connections W_i and lateral interconnections. The differences in the neuron layer activities A_i are then used to generate motor commands based on the current posture P, resulting in effective closed loop control of the three joint arm. The dashed rectangle shows the area of external hand space that can be encoded. The dotted arrows show the effect of the motor commands and the slices at each joint show the reachable joint angles.

C. Posture Memory

The posture memory (PM) was implemented by a fully connected single layer neural network which maps from extrinsic hand space to intrinsic posture space by means of a 1326×720 weight matrix W_{PM}. During motor learning, in each time step of the simulation the current hand (H) and arm state (P) were associated by Hebbian learning:

$$W_{PM}(t) = W_{PM}(t-1) + \epsilon P H^T,$$

where $\epsilon = 0.001$ is the learning rate. To obtain *all* the arm postures that were associated with a single hand location during learning, a desired hand location H_{goal} was fed into the network:

$$P_{goal} = W_{PM} \times H_{goal}.$$

The output activation P_{goal} then represents the redundant arm postures that are suitable to move to the desired hand location. This set of postures is further processed by the motor controller.

D. Motor Controller

The motor controller (MC) extracts complete body models from movements made in a motor babbling phase, during which random motor commands were executed. This information was encoded motor-command-dependently in $n = 6$ recurrent interconnected neural networks. Each network was associated to a certain motor command and stored the transitions in posture space that occurred if the respective motor command was executed. This information was then used to generate posture-to-action mappings dynamically if a new target is presented.

1) Motor Learning: Each of the six neural networks consisted of a single layer of interconnected neurons A_i. The neuron layers A_i had the same size as the posture representation P and thus consisted of 720 neurons each. Each neuron in a layer was connected to itself and all other neurons of the same layer by a 720×720 synaptic weight matrix W_i. During learning, neural layers A_i had the following dynamics:

$$A_i(t) = \rho A_i(t-1) + mc_i(t-1)P(t-1),$$

where P is a representation of the arm posture, ρ is a decay coefficient that enabled the learning of temporally far reaching posture transitions by maintaining a trace of past posture representations, and mc_i is the activation of the i-th motor command during learning. Neural network weights were updated according to a the following associative learning rule:

$$w_i^{jk}(t) = w_i^{jk}(t-1) + \delta a_i^j(t) p^k(t)(\theta - w_i^{jk}(t-1)),$$

where w_i^{jk}, a_i^j, and p^k are single values of the weight matrices, neuron layers and the representation of the current posture, respectively, δ is the learning rate that exponentially decreased from $\delta_0 = 0.1$ to $\delta_{1{,}000{,}000} = 0.01$ during learning, and $\theta = 0.1$ is a ceiling value that prevented weights from increasing infinitely. The motor babbling phase in which MC and PM were trained lasted for 1,000,000 time steps. Thereby, in random intervals of 1 to 8 time steps, a new set of motor commands was generated by setting each motor command to 1.0 with a probability of $p = 0.3$ and to 0.0 otherwise. This procedure was repeated until at least one motor command was set to 1.0.

2) Dynamic Programming: When MC is used for control, a posture-to-action mapping is prepared by dynamic programming that associates a set of motor commands to each possible arm state. Once this mapping is built it is used to direct the arm by means of closed loop control to the goal. The dynamic programming is based on the connectivity

between the different neurons of a neural network (W_i) and interconnections between the neurons in different neural networks associated to the same posture. In each time step, the activity levels A_i of the neurons were updated by the following equations:

$$A_i^* \leftarrow \max\{\beta(\gamma\frac{\sum_j^{j\neq i} A_j}{n-1} + (1-\gamma)A_i), P_{goal}\}$$
$$A_i \leftarrow A_i^* + W_i \times A_i^*,$$

where n is the number of neural networks, max returns the entry-wise maximum of two vectors, $\beta = 0.17$ reduces neural activity, $\gamma = 0.43$ specifies the intensity of crosstalk between networks, and P_{goal} is the representation of suitable goal postures normalized so that single values add up to 1.0. The activation of the goal representation is constantly injected into the neural networks (max operator). Activities in the neural networks spread out from this goal activation, resulting in a stable state which yields different activity patterns in different networks. Due to the acquired body model encoded in the synaptic weights, activation is propagated preferably to those neurons that represent postures from which the goal can be easily reached, if the motor command associated to the respective neural network is executed. Thus, it is possible to determine which motor command is suited best to pursue the current goal from the current arm state by comparing the activation levels in neurons that encode the current posture and approach the goal by means of closed loop control. This was computed by the following equations:

$$mc_i^* = P^T A_i,$$
$$mc_i = \frac{\max(mc_i^* - mc_{anta(i)}^*; 0)}{\sum_{i=1\ldots6}\max(mc_i^* - mc_{anta(i)}^*; 0)},$$

where P is the current posture, and $mc_{anta(i)}$ is the antagonistic motor command to mc_i. This resulted in a normalized set of motor commands that moved the arm for $2.25°$ in posture space (d1-norm).

III. Two Behavioral Findings

In this section, we report simulated behavioral data from psychological experiments with SURE_REACH. First, by default, stored kinematic redundancy is exploited to move to a goal as quickly as possible. We show that, depending on the start position, movements to the same hand location end with different postures. Second, by interaction between MC and PM, the model is able to acquire a goal with a posture that facilitates the execution of a subsequent movement.

A. Start Posture Dependency

If humans are instructed to move the hand to a specific location the final arm posture of the movement depends on the initial arm posture. This posture dependency enables humans to exploit the kinematic redundancy of their arms to exert more efficient movements [8], [9]. To evaluate if SURE_REACH can account for this finding, 10 controllers were independently trained. After learning, each controller had to move to 25 random goal locations. Each goal was pursued starting from two randomly selected arm postures. A movement was allowed to take maximally 100 time steps. In sum, 500 movements to 250 different hand goals were performed. Of the 500 movements, 94.8% reached a position within a 3cm radius around the target within $38.8(SD = 19.5)$ time steps on average. The remaining 5.2% movements had an average error of $6.65cm(SD = 3.77cm)$ in hand space.

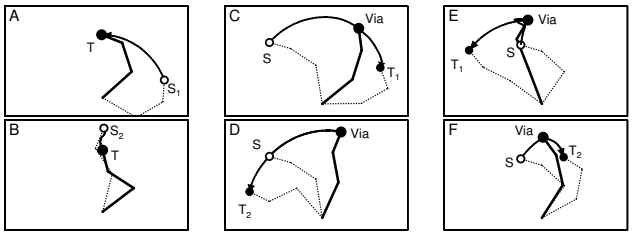

Fig. 2. A, B) Movements starting from different locations (S_1,S_2) to the same target (T) result in different end postures. C-F) The end postures of movements from identical start postures (S) to identical targets (Via) can be adjusted so that good starting positions for movements to subsequent targets (T_1,T_2) are assumed.

To determine if the end posture of a movement depends on the starting posture, the difference between the final postures (after 100 time steps) were computed (d2-norm) for each pair of movements. On average, postures differed by $39.2°(SD = 48.5°)$. Figure 2A,B shows an example. To assure that this optimizes control, because end postures are close to initial postures, the following efficiency measure was computed for each pair:

$$E = \Delta\Phi_{1,2} + \Delta\Phi_{2,1} - \Delta\Phi_{1,1} - \Delta\Phi_{2,2},$$

where the $\Delta\Phi_{i,f}$ are the posture differences between the initial posture of the movement with the index (i) and the final posture of the movement with the index (f). If movement end postures are not particularly close to initial postures, the posture differences in actually made movements ($\Delta\Phi_{1,1}$, $\Delta\Phi_{2,2}$) should not differ systematically from posture differences between the start posture of one movement and the end posture of another ($\Delta\Phi_{1,2}$, $\Delta\Phi_{2,1}$). E should thus be close to 0.0. However, E is positive if the transitions of actually made movements are shorter than the transitions of the 'virtual' movements. The average E for each controller was computed and compared to 0.0 with a t-test, which revealed a significant positive value ($E = 27.1°, SD = 9.60°, t(9) = 8.91, p > 0.001$). The results show that, as in humans, end postures of movements to the same hand location depend on the starting posture in a way that optimizes movement efficiency.

B. Anticipatory Posture Selection

Most movements in every day life are part of a larger sequence. For example, grasping a cup is often followed by moving the cup to the mouth. Hence, in movement sequences, motor redundancy should be exploited in a way so that the outcome of one movement is a good starting point for the subsequent one. Indeed, this has been shown in numerous

experiments. For example, data from humans that had to sequentially reach different hand targets clearly revealed that the arm posture at an intermediate target location depends on the subsequent target [8].

Computational models of motor learning and control that do not encode redundant solutions for the inverse kinematics problem are unlikely to account for this finding. In SURE_REACH, the redundant postures that are represented for each hand target can be weighted, dependent on their utility to reach the next goal. This was simulated by a two step process involving both PM and MC. First, activation maps for moving to the second of two targets were generated for 25 time steps without actually moving the arm. Second, this activation was combined with a goal representation for the first target (P_{goal}) to generate a target representation (P^*_{goal}) for the first movement, that incorporates demands for the second movement.

$$p^{*j}_{goal} = p^j_{goal} \times (10^{-\alpha} + \max(a^j_{i,0\leq i\leq n})) \qquad (1)$$

where p^{*j}_{goal} are the components of P^*_{goal} and p^j_{goal} are the components of P_{goal}, and α is a weighting parameter that determines the amount to which the compound target representation is influenced by the subsequent target. The larger α, the higher the influence of the second target. The a^j are the components of the activation maps A_i, which indicate the closeness of the associated posture to the subsequent target, assuming that the i-th motor command is activated. Thus, the largest a_i indicates closeness, assuming that the optimal motor command is activated.

The anticipatory capabilities of the controller were tested with the ten independent controllers mentioned in the previous section. Each controller had to perform 50 sets of movements. A set consisted of four movements to random locations Via, T_1, and T_2: (1) a movement from a starting posture S to a via target Via anticipating a subsequent target T_1, (2) a subsequent movement to T_1, (3) a movement from S to Via anticipating T_2, and (4) the subsequent movement to T_2. Thereby the goal representation for movements to the via target (1,3) were determined by equation 1. Figure 2C-F shows example movements ($\alpha = 6.0$). Each of the movement sets was simulated with four different settings of α ($\alpha = 3, 4, 5, 6$) and a control setting, in which movements to the via target were carried out independently of the subsequent goal. The hand location at S and the targets T_1, T_2, and Via were separated by at least 20cm from each other.

For each controller and each of the four settings of α, the average difference between end postures of movements to the same via location but with different anticipated subsequent targets (1,3) was computed (d2-norm) to determine how α affects the dependency of a movement's end posture on a subsequent task. In the control setting the postures at the via location did not depend on a subsequent target. Figure 3A shows that increasing impact of the anticipated goal increases the posture difference at the via location as well. A one-way ANOVA revealed a significant effect of α ($F(3, 36) = 70.2, p < 0.001$).

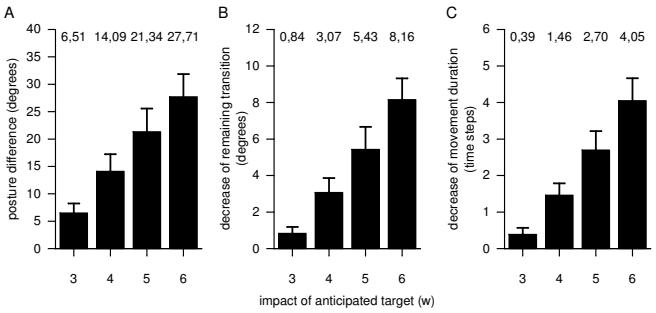

Fig. 3. A goal representation in SURE_REACH can be adjusted to facilitate an anticipated subsequent movement. A) The more the goal of a subsequent movement is incorporated (α) into a goal representation, the more different the end postures of movements to the same location, given different subsequent targets. The difference between B), the required transition in posture space, and C), the duration between movements that independently followed another movement and movements that have been anticipated in the goal representation of the preceding movement, increase with increasing α. Error bars show standard deviations of the average performances of ten individually trained controllers.

To assess if this effect was accompanied by an increase in efficiency we further analyzed two performance measures for movements (2) and (4): the joint angle transitions (d2-norm) made during the movement and its duration[2]. We compared the performance of movements in the anticipatory conditions to those in the control condition by subtracting the former from the latter. Positive values indicate lower joint angle transitions and faster movement times compared to control movements. Figures 3B, C show that the efficiency of movements (2) and (4) increase if the goals of these movements are more strongly incorporated in the goal representations of the preceding movements. One-way ANOVAs revealed a significant impact of α on the remaining joint angle transition ($F(3, 36) = 108, p < 0.001$) and on movement duration ($F(3, 36) = 127, p < 0.001$).

The simulated experiments show that SURE_REACH exploits kinematic redundancy to incorporate demands of the subsequent task in its goal representation. By doing so, the subsequent movement can be carried out faster because it starts from an advantageous posture. The suitability of a posture to serve as starting posture for a movement to a particular hand target is provided by the sensorimotor grounded distance measures in the motor controller. Similar behavior in humans has been found in reaching tasks [8] but also in other domains like bimanual object manipulation [10] or speech production [11]. Additionally, the more complex movement preparation process is in line with experimental findings, which show an increase in preparation time for the initiation of the first movement of a sequence of aiming movements [12]. In conclusion, the availability of redundant postures provides the flexibility to align movements to the demands of future tasks.

[2]The movement duration was considered the time between the onset of a target and the number of time steps required to move to an area within 5cm of the target. Movement sets were excluded from the computation of the movement duration if at least one movement didn't reach the 5cm criteria (6.0%) to obtain valid results.

IV. Discussion

We outlined an unsupervised learning architecture for goal directed behavior that grounds behavioral flexibility on learned body models. Unlike many accounts for motor learning that lack behavioral flexibility due to highly aggregated goal-to-action mappings, SURE_REACH strives to develop complete kinematic and sensorimotor models. It extends previous models that account for the readily incorporation of task-specific constraints and optimality criteria by learning the necessary body models from sensorimotor interaction.

In the introduction, we criticized the goal-to-action mapping approach due to its incapability to encode redundant actions and cope with novel goal representations. The simulations reveal that SURE_REACH can use both to enhance control performance. The posture memory (PM) provides many redundant postures for each desired hand location. These redundant, possibly weighted, goal representations can be processed by the motor controller (MC), even if these representations were never used during learning. The simulations confirmed that SURE_REACH exploits the encoded kinematic redundancy by default to generate short efficient movements. Moreover, the explicit representation of redundancy enables the model to adjust the final posture of a movement in a way that facilitates the execution of a subsequent movement, if this is required by the current task.

In contrast to many other models, SURE_REACH relies on an unsupervised learning scheme that connects neurons that encode different body configurations. This is very appealing from a computational and neuroscientific point of view. On the one side, body states or movement plans are likewise represented by populations of neurons in different motor areas [13]. Theoretical considerations suggest that this form of representation is not only robust but enables advanced information processing [14]. On the other side, associative learning mechanisms were substantiated in motor-cortical areas [15]. Furthermore, a problem of error-based motor learning approaches is avoided because associative learning does not require the transformation of an externally represented error signal into an error signal in motor command space.

The reported simulations clearly show that the representation of complete kinematic and sensorimotor body models enables the quick adaptation to novel tasks and optimality criteria. Whereas striving to encode redundancy may be very economic in the sense that most of what is experienced is also retrievable, it requires far more complex neural networks to encode the body models and to generate motor commands task-dependently than neural networks that encode a direct goal-to-action mapping. Hence, to apply this architecture to control a body with many more degrees of freedom, the learning mechanisms and body representations have to be refined. First, many independent low dimensional body models could be stored in a modular architecture, separating, for example, arm, hand, and finger representations. Second, the now hard-wired population encoding could be improved by self-organizing maps that optimally cover the relevant work space. Third, local learning rules and sparse neural networks could replace the now fully connected neural networks. By including these enhancements we are confident that SURE_REACH will be able to control more complex and dynamic bodies.

In conclusion, learning mechanisms that encode only a single goal-to-action mapping are too restricted to account for the high flexibility in human motor behavior. Although it is not yet sufficiently well understood how the brain adjusts motor control from one moment to the next matching different task requirements, the presented neural network architecture suggests one possible solution by encoding redundancy and resolving redundancy task-constrained on the fly.

Acknowledgments

The authors are grateful for the fruitful discussions with their colleagues in Würzburg. This work was supported by the European commission contract no. FP6-511931.

References

[1] M. Kawato, "Internal models for motor control and trajectory planning," *Current Opinion in Neurobiology*, vol. 9, pp. 718–727, 1999.

[2] D. Bullock, S. Grossberg, and F. H. Guenther, "A self-organizing neural model of motor equivalent reaching and tool use by a multijoint arm," *Journal of Cognitive Neuroscience*, vol. 5, no. 4, pp. 408–435, 1993.

[3] M. Haruno, D. M. Wolpert, and M. Kawato, "Mosaic model for sensorimotor learning and control," *Neural Computation*, vol. 13, no. 10, pp. 2201–2220, 2001.

[4] D. A. Rosenbaum, L. D. Loukopoulos, R. G. J. Meulenbroek, J. Vaughan, and S. E. Engelbrecht, "Planning reaches by evaluating stored postures," *Psychological Review*, vol. 102, no. 1, pp. 28–67, 1995.

[5] H. Cruse, U. Steinkühler, and C. Burkamp, "MMC - a recurrent neural network which can be used as manipulable body model," in *From Animal to Animats 5*, R. Pfeifer, B. Blumberg, J.-A. Meyer, and S. Wilson, Eds. Cambridge, MA: MIT Press, 1998, pp. 381–389.

[6] E. Todorov and M. I. Jordan, "Optimal feedback control as a theory of motor coordination," *Nature Neuroscience*, vol. 5, no. 11, pp. 1226–1235, 2002.

[7] M. V. Butz, O. Herbort, and J. Hoffmann, "Exploiting redundancy for flexible behavior: Unsupervised learning in a modular sensorimotor control architecture," 2007, (Manuscript submitted for publication).

[8] M. H. Fischer, D. A. Rosenbau, and J. Vaughan, "Speed and sequential effects in reaching," *Journal of Experimental Psychology: Human Perception and Performance*, vol. 23, no. 2, pp. 404–428, 1997.

[9] J. F. Soechting, C. A. Buneo, U. Herrmann, and M. Flanders, "Moving effortlessly in three dimensions: Does donders' law apply to arm movement?" *Journal of Neuroscience*, vol. 15, pp. 6271–6280, 1995.

[10] M. Weigelt, W. Kunde, and W. Prinz, "End-state comfort in bimanual object manipulation," *Experimental Psychology*, vol. 53, no. 2, pp. 143–148, 2006.

[11] G. S. Dell, F. Chang, and Z. M. Griffin, "Connectionist models of language production: Lexical access and grammatical encoding," *Cognitive Science*, vol. 23, no. 4, pp. 123–147, 1999.

[12] A. Lavrysen, W. F. Helsen, L. Tremblay, D. Elliott, J. J. Adam, P. Feys, and M. J. Buekers, "The control of sequential aiming movements: The influence of practice and manual asymmetries on the one-target advantage," *Cortex*, vol. 39, pp. 307–325, 2003.

[13] A. P. Georgopoulos, "Current issues in directional motor control," *Trends in Neuroscience*, vol. 18, no. 11, pp. 506–510, 1995.

[14] D. C. Knill and A. Pouget, "The bayesian brain: The role of uncertainty in neural coding and computation," *Trends in Neurosciences*, vol. 27, no. 12, pp. 712–719, 2004.

[15] A. Jackson, J. Mavoori, and E. E. Fetz, "Long-term motor cortex plasticity induced by an electronic neural implant," *Nature*, vol. 444, pp. 56–60, 2006.