Grounding an Internal Body Model of a Hexapod Walker
Control of Curve Walking in a Biologically Inspired Robot

Malte Schilling[1,2], Jan Paskarbeit[2], Josef Schmitz[2], Axel Schneider[2], and Holk Cruse[2]

***Abstract*— While internal models are a prerequisite for higher-level function, they have to be grounded in lower-level function serving sensorimotor control. In this paper we introduce an internal body model for the control of a hexapod walker. The internal model deals with a highly complex robotic structure of 22 degrees of freedom and coordinates the single joint movements to achieve an overall stable and adaptive walking behavior. It is implemented as a hierarchical recurrent neural network consisting of different levels of abstraction which are tightly intertwined. We demonstrate the feasibility of the concept by applying the model to a simulated robot and show how the different levels of the body model interact and how this allows to scale the model even further. While the internal model is used in this context explicitly for motor control, it is also a predictive model and can be applied for sensor fusion. We discuss how in this way such an internal model offers the flexibility to be utilized in motor control and to be used for planning ahead by a cognitive expansion of the movement controller.**

I. INTRODUCTION

Cognitive function have been found to be based on lower-level and motor control related function [13]. Planning ahead—following [16] as the defining characteristic of cognition—in this sense recruits [1] internal models originally co-evolved in the context of action [26] in a form of internal simulation [11]. While this provides a grounding for internal models, i.e., they are directly anchored in the sensorimotor system, the internal models are in addition required to be flexible. Only a flexible model allows to be applied in a new context. In this way, for example, a predictive model can be exploited as an internal simulator to anticipate consequences of possible actions and to select only an appropriate one when facing a never experienced and possibly harmful situation.

An internal model of the body [4] can be assumed as a central representation and has been found active in quite diverse tasks besides motor control [28], perception of movements [14], planning [9] and, in the case of humans, even language [19]. The body model appears to serve different functions. First, in motor control it has to solve the inverse kinematic problem to coordinate whole body movements. Second, as there are multiple noisy, but redundant sensory information in biological systems, it is important to integrate the redundant information and to fuse the sensory data. Third, predictive models are already important in motor control to account for the delay of sensory feedback. But in addition they are a prerequisite for the ability to plan ahead.

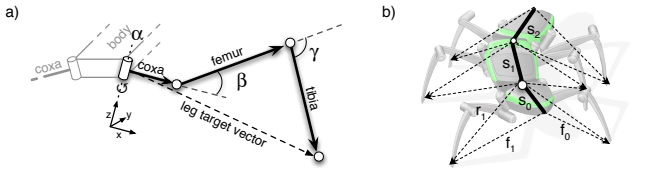

Fig. 1. The body model for the six-legged walker is divided into two layers. The lower layer contains six networks, each representing one leg (a). The upper layer (b) represents the body and the six legs, which are only represented by vectors pointing towards the tip of each leg. On this level the leg is described with reference to the respective body segment. Both layers are connected via the shared representation of the target position of the leg and are implemented as recurrent neural networks. α, β, and γ denote the three joint angles.

In this paper, we present a recurrent neural network approach for an internal model of the body which addresses all three tasks mentioned. We want to focus here on the inverse function and show how this type of model can be applied to the control of a simulated six-legged walking robot. As this robot has a high number of degrees of freedom, the overall complexity will be distributed onto two levels of representation. We will explain how these levels are interconnected. The work presented demonstrates the feasibility of the approach for very complex manipulator structures and how an internal model of the body can be grounded that can be also used for prediction and sensor fusion.

II. MMC BODY MODEL

The task of the body model is to control the walking of a six-legged robot. Each leg consists of three joints. The central body of the robot consists of three segments that are coupled via two universal joints. Therefore, the robot has 22 degrees of freedoms in total. During walking not all the legs are on the ground at the same time. But even with only three legs on the ground—as during tripod walking—there are still at least 13 degrees of freedom which have to be controlled and which are directly coupled. Therefore, it is helpful to distribute the complexity onto different levels of representation (Fig. 1). In our approach we divide the complexity onto two levels. On the lower level, each leg is represented. The kinematics of the leg are described through

*This work has been supported by the Center of Excellence Cognitive Interaction Technology (EXC 277), by the EC-IST EMICAB project # FP7–270182 and by a DAAD postdoctoral fellowship.

[1]M. Schilling is with the International Computer Science Institute Berkeley (CA), 1947 Center Street, Suite 600, 94704 Berkeley (CA), USA malteschilling at googlemail.com

[2]M. Schilling, J. Paskarbeit, J. Schmitz, A. Schneider, and H. Cruse are with the Center of Excellence Cognitive Interaction Technology, University of Bielefeld, Germany

the three joints and the lengths of the segments. Together, these elements constitute a target vector of the leg. On the higher level, only the position of the foot point with respect to the body segment is represented abstracting away the detailed joint angle representation required to actually control the leg itself. The level of body representation is constituted by vectors representing the foot points of the legs with respect to the body and by the representation of the single body segments (see Fig. 2). All of these vectors are represented as three dimensional vectors. The coupling between the levels is given through the shared representation of the target position of the leg.

The body model is represented by a recurrent neural network based on the MMC principle. In the following, we will first explain the general MMC approach and will then introduce the MMC body model for the six legged walker. We will briefly explain the leg networks, but want to focus mainly on how the different leg networks can be integrated in the body model.

A. The general MMC approach

The Mean of Multiple Computation (MMC) principle [5] allows to find solutions for kinematic problems. Already quite simple manipulator structures can be intractable from a mathematical point of view and no closed solution can be found. For example, a human arm can be considered to consist of three segments (upper arm, lower arm and hand) and having seven degrees of freedom. The arm is redundant, meaning there are many possible solutions to reach for a point in three dimensional space and one can not provide a direct mathematical solution. This is a problem for many control approaches and usually requires the introduction of additional constraints. In contrast, the MMC approach exploits the redundancy and as it distributes the complexity onto a local level, it still can find solutions for manipulators with a—in principle unlimited—high number of degrees of freedom.

In the Mean of Multiple Computations approach the kinematics are decomposed into local relationships. A local relationship consists usually only of three variables and describes how these are related. In the example of a manipulator arm, one could think of these variables as the vectors of the segments. Relating two variables often requires the introduction of additional auxiliary variables, e.g., when relating the first and second segment one has to introduce a diagonal vector as a new auxiliary variable. In the end, one can derive from the kinematics of a manipulator a large set of equations and each variable (including the auxiliary variables) is contained in multiple of these computations. Next, to come up with a Mean of Multiple Computation network, one has to take for each variable all the computations that contain this variable and solve the equations for this variable. This results in Multiple Computations for each variable. For a given manipulator configuration all this multiple equations will give the same result. But when one changes via an external input this change is propagated through the equations also onto other connected variables. In this case, the Multiple Computations may provide different solutions. The different solutions are simply integrated through calculation of the Mean value of all the Multiple Computations.

In this way, the Mean of Multiple Computation approach constitutes an iterative approach to calculate the kinematics of a manipulator. Usually, one wants to include into the integration of the multiple computations also the current value of the variable. This introduces low pass properties into the system and prevents oscillations.

The computation of the equations and the integration can be at the same time interpreted as a recurrent neural network. The weight matrix of this network is constituted by the equations which describe the relations between the variables. The whole network is acting as an autoassociator and the attractor space is formed by the encoded kinematic constraints. In a stable state an MMC net represents a manipulator configuration and all the equations for one variable yield the same result. But the network can now be used to solve any kinematic problem. For example, to solve the inverse kinematic problem only a target position for the manipulator has to be enforced onto the network. This disturbs the stable state of the network, but the disturbance is spread to multiple connected equations and all variables are slightly changed to account for the introduced error. Over time, the network settles again in a harmonic state. The network minimizes the overall error of the network which leads to a solution of the inverse kinematics, or—if there is no solution—at least the network comes as close to a solution as possible (When thinking of reaching movements of an arm, the target position could be simply out of reach. The MMC network would provide an arm configuration in which the arm is pointing towards the target.). In this way, MMC networks have been used for the solution of inverse, forward and any mixed kinematic problems (for details on the three segmented arm example and the complete derivation of a network for the example see [23]; there it is also shown how the MMC principle can be applied to complex representations of joint angles as typically used in robotics).

B. Overview of the MMC Leg Level

The MMC body model consists of two layers (see Fig. 1). On the lower level, each leg is represented as a kinematic chain consisting of three leg segments. The three segments are interconnected and attached to the body through three joints each providing one degree of freedom. The leg networks for the control of the leg are implemented as MMC networks using joint angle representations. In general, this can be realized by employing a transformation representation like dual quaternions to represent the kinematics of the manipulator [23]. We applied this to the insect leg in the past [21], but as the insect leg only consists of three degrees of freedom, it is possible to even simplify such an MMC network using redundant trigonometric relationships. Importantly, both types of MMC leg networks—and MMC networks in general—are forming recurrent neural networks acting as autoassociators.

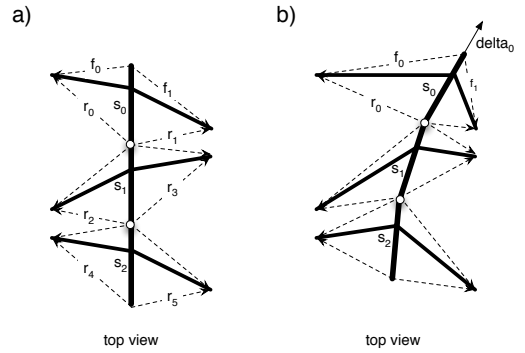

Fig. 2. Vectors constituting the body model (view from above). In a) the two vectors describing the foot point with respect to the segments are shown (dashed lines). Leg and body segments shown in bold. b) shows how the vectors are changed when the model is pulled at the front ($delta$ vector) and the foot points are kept in place.

Such an MMC network has several advantages compared to an explicit computation. First, the network is able to solve forward, inverse and any mixed kinematic problems in a few iteration steps. Second, explicit computations of inverse kinematics—if possible at all—often involve the application of inverse functions of sine and cosine functions which require case distinctions or lead to singularities. These are not required in the case of the MMC leg networks. Third, for cases where no solution is possible (e.g., when a target point of the leg is situated outside the workspace and therefore unreachable), the net still converges to a stable and geometric valid solution which is minimizing the error (in the example, this would be the leg pointing into the direction of a far away target).

The leg networks constitute the lower level of the body model. The MMC networks compute the control signals for the joint motors during stance movement, i.e., when the leg is on the ground and therefore, the leg forms a closed kinematic chain with all the other legs currently touching the ground. A vector pointing to the foot of the leg is provided by the higher level body network to the leg network which calculates corresponding joint angles as control signals and pushes these down to the joint motors. In the next section, we will explain how the target vectors of the leg networks are integrated into the higher level body model.

C. The Higher-level Body Network

We will use a complex MMC body model of a six-legged walker (that is a hierarchically organized model of the legs and the body) to control targeted movements of the leg for the stance movement. The function of the body model is to mediate the coupling between the single legs. During a movement these vectors have to be moved in a coherent way. While the body moves to the front, the feet should stay on the same place on the ground, i.e., the relative position between the feet should not change. As the model still has a high number of degrees of freedom, this is still a hard problem and not directly computable. Therefore, we apply the idea of the passive motion paradigm to this problem. The body model can be thought of as a real stick model of the manipulator. When we now pull this model at the front with a rubber band, the front segment will move in this direction and the other body segments, the legs and the interconnecting joints will just follow the movement [17].

We use the MMC principle to encode the constraints describing the kinematics of the body model as described above (for details on the derivation of the equations from the kinematic structure see [23]). A movement is now produced by pulling at the tip of the modeled body which disturbs the state of the network. The network distributes the disturbance onto all variables and all vectors have to be changed in order to account for the error. This relaxation of the network is constrained and driven by the encoded kinematic constraints that form the attractor space of the network. The network settles in a state that minimizes the overall error and at the same time represents a valid configuration for the robotic structure. This configuration can be used to control the simulated robot, i.e., we use the resulting representation of the target position of the leg as an input for the lower level representation of the leg geometry that produces joint movements.

In the following, we want to explain some details of the body model. As the hexapod walker moves around, there is no fixed global reference system inherently given. Approaches in the past have applied the MMC principle for a hexapod walker with respect to a fixed world coordinate system [20]—we will deviate from this solution as it is one of our goals to develop from the bottom-up how representations are grounded. A body-centered reference system appears to be an early representation and only later-on other representations grounded in the former might have been developed [4]. Besides, the solution presented here uses a much smaller number of involved equations and is therefore simpler. As a consequence, the vectors in the body model (see Fig. 2 and 4) are not representing absolute positions, but relative ones.

When setting up the Mean of Multiple Computation network it is difficult to encode the leg target vectors (as the vectors going from the base of the first joint down to the end point of the leg, see Fig. 1 a) as such as the body segments themselves can be moved with respect to each other while these leg target vectors are attached to a fixed part of the segment. We would have to ensure this as a constraint during every iteration. Therefore, we describe the relative position of a foot point with respect to the segment by introducing two vectors. First, the f vector starts at the front of the respective body segment and goes down to the foot point. Secondly, the r vector connects the rear of the body segment with the foot point of the leg. These two vectors are used inside the MMC body network as the representation of the target position of the leg (Fig. 1 b). Together with the body segment it is straightforward to derive the leg target vector which is needed on the lower-level to compute the corresponding joint angles. This transformation can be given as a simple neural network, but has to be applied only once when the information of the foot position is passed down from the body network to the leg level. We have trained

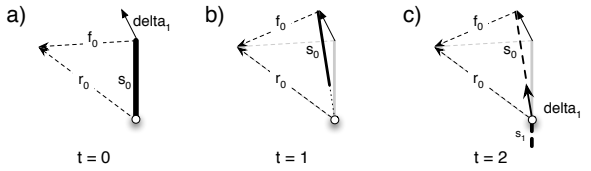

Fig. 3. Introduction of the $delta_0$ vector into the equations: $delta_0$ describes a displacement—a relative pull onto the body segment (a). In (b) it is shown how this affects the different equations: the segment vector is directly altered by this $delta_0$ vector and the f_0 vector, too. The r_0 vector remains the same. The overall error stemming from the $delta_0$ vector can be propagated to the next segment which then is also pulled and also participating in compensating for the disturbance (c). In this way the whole network representing the body model can be driven.

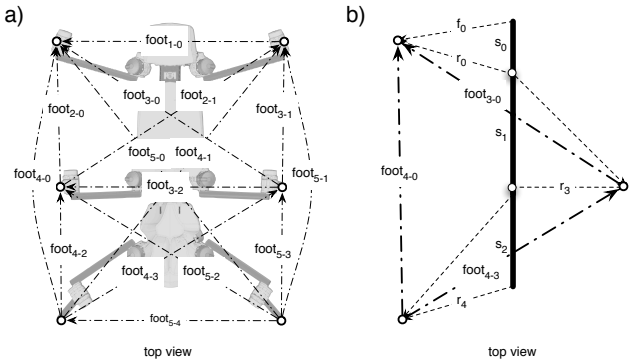

Fig. 4. Introduction of the vectors connecting the foot points. a) vectors connecting the foot points of the legs. b) example configuration during walking, only three legs are on the ground (front left, middle right, hind left. Bold line: body segments.

simple feedforward networks for such simple transformations in the past [20].

As we are only using relative and local relations, we cannot specify from the outside a new target position for the robot in world coordinates and also can not just simply specify a new target vector with respect to the current body reference system as this changes over time and we would have to keep track of such a target position. Therefore, we take the idea of the pulled movement literally and introduce a pull vector ($delta_0$ for the first segment) which is attached to the front body segment. In a settled state of the network this vector is zero. But when we introduce a constant pull, it adds a displacement during each iteration and disturbs the equations. It constantly drives all the connected variables and acts like an explicit error term. This term is included in the equations containing the first body segment. Fig. 3 illustrates how now for the f_0-vector a new value is calculated after an iteration step (this is done in the same way for the other legs, too).

An equation taking part in the computations of an MMC network represents how the new value of a variable can be calculated from current values of related variables. Such equations are deduced from closed chains of vectors describing the kinematics and we restrict ourselves only to use local relationships, i.e., relationships consisting of a small number of variables (usually three, forming all together a triangle described by the vectors). Here, we introduce a new variable which represents an explicit error or displacement term ($delta_0$). As shown in Fig. 3 a) $delta_0$ explicitly represents the displacement of segment number zero in between two time steps. From this, we can derive equations for the depending variables for the next time step (Fig. 3 b):

$$\begin{aligned} s_0(t+1) &= s_0 + delta_0(t) \\ f_0(t+1) &= -delta_0(t) - s_0(t) + r_0(t) \\ r_0(t+1) &= -s_0(t) + f_0(t) = r_0(t) \end{aligned} \quad (1)$$

Importantly, only the f_0 and segment (s_0) vector are affected by the $delta_0$ vector. But this change spreads through the equations to all other connected variables (including the r_0 vectors). These equations are now integrated into the set of equations used to calculate a new value for a variable.

In principle, in MMC networks all variables can be changed freely. But for some of the variables certain restrictions should apply, e.g., in Fig. 3 b) one can see that the new segment vector s_0 has become longer than the segment. Therefore, one can introduce external constraints which are applied after every iteration. In the case of the segment length, the s_0 vector will have to be shortened which introduces another disturbance into the network (the network is in a non-harmonic state and is still moving through the attractor space). When—at some point in time—we set the $delta_0$ vector to zero, we delete all external disturbances and the network will settle during a few iteration steps and will compensate also for the internal disturbance which stems from the application of the non-linear constraints.

We can also exploit this internally produced error: Until now, we only have explained how the first segment is pulled through the explicit disturbance vector introduced by us. But how are the other body segments adopt the movement, what is pulling the next body segment? Actually, the first segment pulls at the second segment (in direction of its movement)—and this pull corresponds to the error vector from above (Fig. 3 c). The vector that we use for compensation of the length change can be also used as a displacement for the following body segment (and in the same way a $delta$ vector can be derived for the last body segment):

$$delta_1(t+2) = r_0(t+1) - f_0(t+1) - s_0(t+1) \quad (2)$$

An important aspect of the MMC body network are the relations between the foot points of the walker (Fig. 4). As we have argued above, we use relative relations between all the participating variables. These describe, on the one hand, the embodied relations, i.e., a leg or a segment. On the other hand, the foot point relations relate the body model to the environment: These vectors are situated and they provide a connection from the body to the surroundings. The foot point vectors constitute something like a world for the agent. They provide an embodied frame of reference to which the world can relate and in which global references can be anchored. One important aspect is that these vectors are not allowed to change because as the insect can attach with their tarsi to the

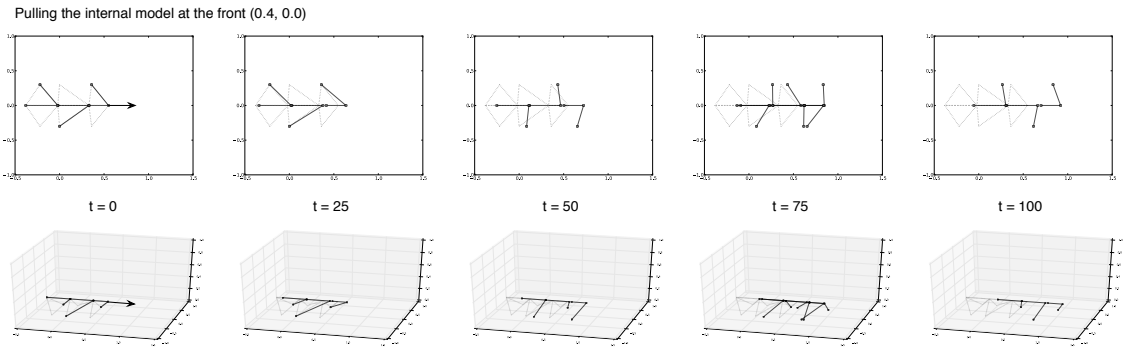

Fig. 5. Example run of the internal body model (time from left to right), shown are different snapshots of the simulation (in the top a view from above, below a perspective view from the side and top). Shown are only the f vectors for the legs which are currently on the ground (producing the stance movement). In the fourth picture all the legs are on the ground while they are all switching from stance to swing (swing to stance respectively) behavior.

ground the feet are fixed on the ground. We can derive all possible connections between the feet (15 overall, shown in Fig. 4 a) and from this we can derive closed chains of vectors which can be used to set up equations (e.g., for the left front leg and the right middle leg, we can setup an equation as illustrated in Fig. 4 b): $f_0 = -s_0 + f_3 + foot_{3-0}$). From this, we achieve for each leg variable (both, the r and the f vectors) 5 additional equations, but most of the time only a few of these can be used. When the robot is walking, part of his legs are in the air while making a swing movement. For these legs, we cannot exploit the foot vectors as the feet are not fixed anymore and the relative position between the feet changes (for an example during tripod walking, see Fig. 4 b).

Therefore, the controller [8] which selects for each leg which movement to produce (there are only two possibilities: a swing movement (for details on how the swing movement is controlled see [25]) moving the leg to the front or the stance movement pushing the leg backwards and moving the body forward) has to inform the body model about which feet are on the ground and thus shall be included in the computations of the body model. As all the relations are described directly as part of a neural network, this modulation through the higher level controller can easily be included—a leg in the air is inhibiting all its contributions in the body model.

The MMC network is constituted by all these equations describing multiple computations for each variable. The multiple computations are integrated through mean calculation.

III. SIMULATION RESULTS

The body model has been implemented as described above. The internal model was used to compute the joint trajectories of the legs being in stance mode (for details on how the swing movement is controlled see [25]). As explained, the internal model consists of two layers—a body layer and a layer consisting of the six leg networks. This model is pulled from the outside in the walking direction and provides corresponding representations of the target position of the leg (and segment orientations) to the overall movement of the body. A disturbance vector (describing the displacement) is set in the body model and the network is advanced one iteration step. In our simulations, the body model has not to settle to a complete harmonic state before the foot point position can be passed down from the body level to the leg networks. One iteration step per control step has shown to be sufficient, as the network continues to relax to a stable state over the following control steps applied, even though these might introduce new disturbances. The resulting foot point positions are translated into leg target vectors and these are forced onto the MMC leg models which are then advanced further. The leg networks settle quickly (a couple of iteration steps) into a stable state and provide corresponding values for the joints. When a leg is in stance mode, these values are taken to control the joints of the simulated robot (these values are used as target values for the individual joint motor, for details on the actuation of the individual joint see [18]).

In Fig. 5 we show an example of a straight movement (to the right) as represented by the neural network model. The explicit pull vector is set and drives the activation of the network, first it moves the front segment and the attached legs. Then the activation is propagated to the other segments. As can be seen, the body adopts the movement while the feet keep in place. Shown are only the f vectors for the legs on the ground. These vectors are the vectors which are shared with the lower leg level and which are pushed onto the leg networks as the leg target vectors. The leg network then

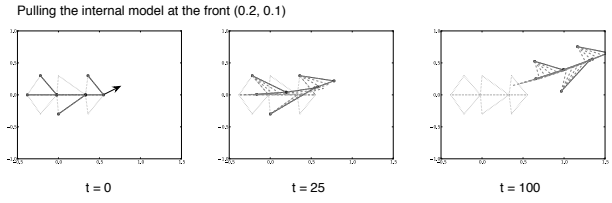

Fig. 6. Example run of the internal body model (time from left to right), shown are different snapshots of the simulation when the body model is pulled to the front and left. The segments are following this direction and the legs support the movement. Shown are only the f vectors for the legs that are currently on the ground (producing the stance movement).

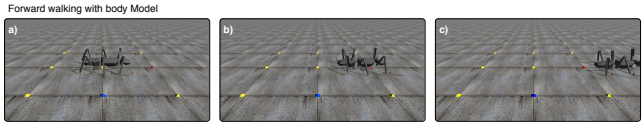

Fig. 7. Simulated robot walking forward using the internal body model. Red and green vertical bars illustrate the position of the anterior extreme position (AEP) and posterior extreme position (PEP) of each leg, respectively.

produces the corresponding stance movement. Fig. 6 shows another simulation, this time the body model is pulled to the front and the left. Again, the model follows the induced movement.

We have applied the model to a robot in a dynamic simulation. The robot is controlled by a biological inspired controller termed Walknet [8]. The general idea is that each leg has its own controller deciding which movement to perform. This decision is based on the current sensory state of the leg, but also influenced by the state of the neighboring legs. It has been proven in the past to lead to stable walking behaviors and be especially good at adapting to even severe disturbances [24]. While the swing net is controlled by a simple neural network [25], the stance movement is controlled by the body model. The Walknet controller has to inform the body model about which legs are currently fixed on the ground. These legs constitute the current manipulator configuration that is realized by the network. Fig. 7 shows a picture series of the walking robot using the internal body model.

To use the body model to negotiate a curve, only the pull vector acting on the body model has to be adjusted and has to point in the direction the animal should walk to. The body model is pulled (at the front of the first segment) into this direction and all the standing legs are following as well as the segments. In Fig. 8 we show an example run in which the pull vector is set to the front and left of the animal (left and down in the figure). As can be seen, the robot is taking a slight turn which is initiated by the body segments turning slowly into the direction of the curve and which is supported by the adapted leg trajectories.

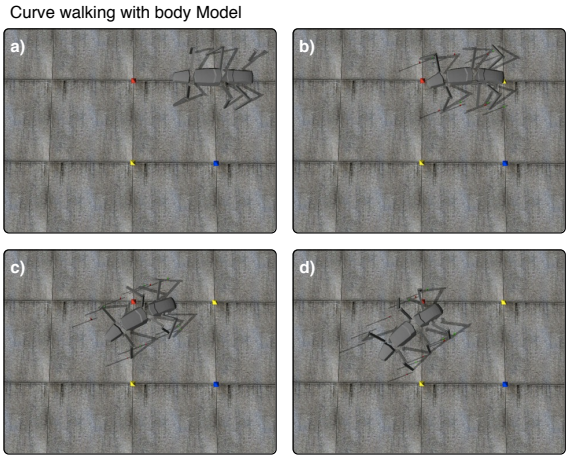

Fig. 8. Simulated robot walking a turn to the left using the internal body model. The internal body model is constantly pulled to the front and the left. Red and green vertical bars illustrate the position of AEP and PEP of each leg, respectively.

IV. DISCUSSION

Different notions of body representations can be found in the literature (for a recent review see [7]). It is usually distinguished between body schema and body image. The body schema is the sensorimotor representation that is based on afferent and efferent information and that is utilized in guiding action. The body image, on the other hand, subsumes all other types of representations not involved in action, e.g., conceptual or emotional ones and it is usually associated with conscious aware representations (as the notion of a body image is quite broad it has been proposed to differentiate more and to partition the concept of body image in multiple representation [7]).

The model presented here falls into the category of body schema. It is important to note that the current geometrical state of the body is represented as an activation of the network and is not encoded at some place of the recurrent network. The MMC model captures the general functional and geometric relationships of the body (for an extension integrating also dynamic characteristics of body movements see [22]). It does not represent specific segment sizes. In this way, the model provides a set of general bodily characteristics which allows to be applied on different bodies. This flexibility makes the body model applicable for diverse tasks. It can be used in different contexts, e.g., for the control of the own body or for the perception of someone else's body.

In a recent review, Hoffmann et al. [12] summarized work on the application of body representation in robotics concluding that body schemas can be used to improve the behavior of robots. Body schemas are applied in robotics in different forms. While some are completely predetermined, it has become more and more important that body schemas can adapt to structural changes and are—at least partially—learnt. In self-calibrating approaches the general topological structure of a body is given and only the parameters are learned (e.g., in [10] from the topology of the robot and parts of the kinematics the robot is able to learn the rest of the kinematic parameters through self-observation). Other approaches try to come up with a body schema starting from scratch (an impressive example is the work by Bongard [2] in which a starfish like robot synthesizes a body scheme and a simple walking behavior and can adapt both even when one leg of the robot is shortened). Most of these approaches have been applied only to quite simple robotic structures (Bongard's robot consists of eight degrees of freedom) and lots of the work focus on the learning of visuomotor coordination of arm movements [12]. The presented MMC model deals with a much more complex body structure and provides a hierarchically organized model which can be further extended. We are not dealing with learning as such of the model, as the underlying structure is assumed partly innate [3] and is nonetheless quite flexible.

Besides that, typical approaches of employing a body schema usually only concentrate on task specific models serving a specific function (the inverse kinematic function) or explicitly distinguish between inverse and forward kinematic models [27]. While this not only requires a large number of—partially redundant—models, it also hinders transfer of knowledge between models [6]. In contrast, the MMC model is a flexible body model which can be recruited for different function (forward and inverse). At the same time, it provides a principle to exploit redundancy, not only from the kinematic description, but also from sensory input. This appears to be an important property as most biological systems consist of a multitude of sensors which often provide redundant data, e.g., the hand position measurement is influenced by skin, joints, muscle, eyes, and even the ears [15]. It has been proposed that the brain integrates these different multimodal influences as a form of weighted mean computation [15] like is done in the MMC model.

V. CONCLUSIONS

We have introduced an internal model of the body implemented as a recurrent neural network. The body model consists of two layers which are connected through a shared representation of the target position of the leg. A detailed description of the leg has been provided in [23] showing how an MMC type model can be used to mediate between joint angle representations and a Cartesian target space for the leg. Here, we presented how a body model can be structured onto different levels and how the different layers can be integrated. We have applied the body model for a hexapod walker with overall 22 degrees of freedom and used the body model to coordinate the movement of all legs touching the ground and to control the actuation of the joints between the body segments. We demonstrated how the body model can be used to negotiate the joint movements for forward walking and for walking in curves.

We are currently applying the leg model to integrate multimodal sensory data and in this way to cancel out noise by exploiting the encoded structural information and the predictive information on the anticipated movements. We will use the body model in a next step also as a forward model for planning ahead. The body model will be driven in an internal simulation loop [11] as a replacement for the body itself, allowing to try out possibly dangerous behaviors in simulation before applying them to the real robot. This would allow the robot to cognitively solve movement problems without getting harmed.

REFERENCES

[1] M. Anderson, "Neural reuse: A fundamental organizational principle of the brain," *Behavioral and Brain Sciences*, vol. 33, pp. 254–313, 2010.

[2] J. Bongard, V. Zykov, and H. Lipson, "Resilient machines through continuous self-modeling," *Science*, vol. 314, no. 5802, pp. 1118 – 1121, November 2006.

[3] G. Carruthers, "Types of body representation and the sense of embodiment," *Consciousness and Cognition*, vol. 17, no. 4, pp. 1302 – 1316, 2008.

[4] H. Cruse, "Feeling our body - the basis of cognition?" *Evolution and Cognition*, vol. 5, no. 2, pp. 162–173, 1999.

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

[6] P. R. Davidson and D. M. Wolpert, "Internal models underlying grasp can be additively combined." *Experimental Brain Research*, vol. 155, no. 3, pp. 334–340, Apr 2004.

[7] F. de Vignemont, "Body schema and body image–pros and cons," *Neuropsychologia*, vol. 48, no. 3, pp. 669 – 680, 2010.

[8] V. Dürr, J. Schmitz, and H. Cruse, "Behaviour-based modelling of hexapod locomotion: Linking biology and technical application," *Arthropod Structure & Development*, vol. 33, no. 3, pp. 237–250, 2004.

[9] D. T. Gilbert and T. D. Wilson, "Prospection: Experiencing the future," *Science*, vol. 317, no. 5843, pp. 1351–1354, 2007.

[10] M. Hersch, E. L. Sauser, and A. Billard, "Online learning of the body schema," *I. J. Humanoid Robotics*, vol. 5, no. 2, pp. 161–181, 2008.

[11] G. Hesslow, "Conscious thought as simulation of behaviour and perception," *Trends in Cognitive Sciences*, vol. 6, no. 6, pp. 242–247, 2002.

[12] M. Hoffmann, H. Marques, A. H. Arieta, H. Sumioka, M. Lungarella, and R. Pfeifer, "Body schema in robotics: a review," *IEEE Trans. Auton. Mental Develop.*, vol. 2, no. 4, pp. 304–324, December 2010.

[13] M. Jeannerod, *Motor Cognition — What Action tells the Self*. University Press, Oxford, 2006.

[14] F. Loula, S. Prasad, K. Harber, and M. Shiffrar, "Recognizing people from their movement," *Journal of Experimental Psychology: Human Perception and Performance*, vol. 31, no. 1, pp. 210–220, 2005.

[15] T. R. Makin, N. P. Holmes, and H. H. Ehrsson, "On the other hand: Dummy hands and peripersonal space," *Behavioural Brain Research*, vol. 191, no. 1, pp. 1–10, 2008.

[16] D. McFarland and T. Bösser, *Intelligent behavior in animals and robots*. MIT Press, Cambridge, MA, 1993.

[17] F. Mussa-Ivaldi, P. Morasso, and R. Zaccaria, "Kinematic networks distributed model for representing and regularizing motor redundancy," *Biological Cybernetics*, vol. 60, no. 1, pp. 1–16, 1988.

[18] J. Paskarbeit, J. Schmitz, M. Schilling, and A. Schneider, "Layout and construction of a hexapod robot with increased mobility," in *IEEE/RAS-EMBS International Conference on Biomedical Robotics and Biomechatronics*, 2010, pp. 621–625.

[19] F. Pulvermüller, "Brain mechanisms linking language and action," *Nature Reviews Neuroscience*, vol. 6, no. 7, pp. 576–582, July 2005.

[20] M. Schilling and H. Cruse, "Hierarchical MMC Networks as a manipulable body model," in *Proceedings of the International Joint Conference on Neural Networks (IJCNN 2007), Orlando, FL*, 2007, pp. 2141–2146.

[21] M. Schilling, J. Paskarbeit, A. Schneider, and H. Cruse, "Flexible internal body models for motor control–on the convergence of constrained dual quaternion mean of multiple computation networks," in *Proceedings of the International Joint Conference on Neural Networks, 2012*, in-press.

[22] M. Schilling, "Dynamic equations in MMC networks: Construction of a dynamic body model," in *Proc. of The 12th International Conference on Climbing and Walking Robots and the Support Technologies for Mobile Machines (CLAWAR)*, 2009.

[23] ——, "Universally manipulable body models — dual quaternion representations in layered and dynamic MMCs," *Autonomous Robots*, vol. 30, no. 4, pp. 399–425, 2011.

[24] M. Schilling, H. Cruse, and P. Arena, "Hexapod Walking: an expansion to Walknet dealing with leg amputations and force oscillations," *Biological Cybernetics*, vol. 96, no. 3, pp. 323–340, 2007.

[25] M. Schumm and H. Cruse, "Control of swing movement: influences of differently shaped substrate," *Journal of Comparative Physiology [A]*, vol. 192, no. 10, pp. 1147–1164, 2006.

[26] L. Steels, "Intelligence with representation," *Philosophical Transactions: Mathematical, Physical and Engineering Sciences*, vol. 361, no. 1811, pp. 2381–2395, 2003.

[27] D. M. Wolpert and M. Kawato, "Multiple paired forward and inverse models for motor control," *Neural Networks*, vol. 11, no. 7-8, pp. 1317–1329, 1998.

[28] D. Wolpert, R. Miall, and M. Kawato, "Internal models in the cerebellum," *Trends in Cognitive Sciences*, vol. 2, no. 9, pp. 338–347, 1998.