A Competitive Mechanism for Self-Organized Learning of Sensorimotor Mappings

Nikolas J. Hemion
Emergentist Semantics
CoR-Lab
Bielefeld University
Email: nhemion@cor-lab.uni-bielefeld.de

Frank Joublin
Honda Research Institute Europe GmbH
Email: Frank.Joublin@honda-ri.de

Katharina J. Rohlfing
Emergentist Semantics
CoR-Lab
Bielefeld University
Email: kjr@uni-bielefeld.de

***Abstract*—How can a robot learn sensorimotor knowledge in a developmental way based on its own experiences solely? An important step is the acquisition of a *body-schema*—learning about the physical behavior of its own body, and how incoming sensory stimuli can be put in relation to the own body. In this work, we study how a competitive learning mechanism, which is related to the EM algorithm, can help to simplify the learning problem. We demonstrate how a robot can learn the way visual stimuli move as a consequence of the robots own actions of moving its camera or moving its end-effector in front of its camera. We show how the robot can discriminate stimuli originating from these two kinds of actions and learn the position of the end-effector in its visual input. Previous approaches have relied on a preprocessing step to "self-detect", which we find is not necessary. The robot acquires a set of sensorimotor estimates, which could later be used, e.g. in visually guided reaching.**

I. Introduction

A *body-schema*, originally introduced by Head and Holmes [1], refers to the ability of organisms to organize incoming sensory stimuli in a way that allows them to bring the stimuli into relation to their own body. For example, a body schema enables to use proprioceptive information to determine the positions of the limbs. In many applications, robots also require this ability to reason about the relation of their body to stimuli originating from the environment, e.g. when planning collision-free motion. Here, a body-schema is usually seen as a model of the body of the robot, which is available to the robot itself. In most existing robots, such a model is pre-specified by the human designer. However, robots with pre-designed internal models fail to adapt to changes applied to their body, e.g. as a consequence of physical damage. Also, in complex robotic setups, it can become difficult or even impossible to design an accurate model of the body, which also gives a motivation for the autonomous acquisition of a body-schema by the robot.

Previous work on the learning of a body-schema in robotics can be subdivided into *explicit* and *implicit* models (see [2] for a recent review). In explicit models, the problem is reduced to calibrating a set of parameters, such as the length of segments. In many cases, this is the easier and more favorable solution but it cannot be applied to all robotic setups. The problem of learning an implicit model, which we also want to consider in this work, can be seen as the more general case, where the model of the body is learned as a sensorimotor mapping. Here, the robot is not given any prior information about the relation between its effectors and sensors. Its task is to retrieve this relationship from the information that is available to it, namely the sensory inputs and the motor commands that it generates.

Most existing approaches to learning a body schema as an implicit model use an illustrative setup with a robot arm and a camera (fixed or movable) observing a visual scene in which the robot's own arm is also located. The body-schema is learned by fitting a model onto training data using standard machine learning techniques. For example, Metta et al. considered the task of letting the robot learn to reach at the point in space where it is looking with its cameras [3]. In their approach, the robot repeatedly first fixated a target visual stimulus, then tried to reach it with its end-effector and then, after performing the arm movement, fixated with its cameras on the visual location of the end-effector. This way, training data could be generated to acquire a "motor-motor" map, associating camera postures with corresponding arm postures. Consequently, the robot could learn to reach at the location where it is looking.

Similarly, Gaskett and Cheng used a pair of a closed loop controller, which was employed for visually guided reaching, and an open loop controller, which could bring the end-effector into the visual field of the robot, where the open-loop controller was trained during operation using self-organizing maps [4].

These works make two assumptions on the robot: On the one hand, the robot is already able to detect the position of its end-effector in its visual field (e.g. by giving it an easily detectable color), and on the other hand, the robot relies on an existing closed-loop controller for calibrating the arm and camera posture space. In this work, we want to show how a robot can learn the necessary mappings without relying on these assumptions.

Several methods have been proposed to let the robot "self-detect" its end-effector in the camera input other than using an easily detectable color, mostly using some combination of movement detection and temporal contingency estimation between the sending of motor commands and the detection of movements. These can be prepended to the learning system to provide the necessary input, i.e. the position of

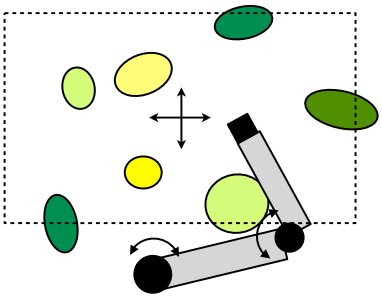


Fig. 1. A visualization of the simulation that was used in the experiments. The simulated robot is equipped with a 2DOF planar arm with a fixed shoulder position and rotational joints in the shoulder and the elbow, and with a 2DOF movable camera (indicated by the dashed rectangle as its field of view), which could move horizontally and vertically in a plane parallel to the robots workspace. Randomly placed objects compose the visual background.

the end-effector. For example, Stoytchev demonstrated how the characteristic delay between sending a motor command and observing change in the image can be estimated using temporal contingencies [5]. Fitzpatrick and Arsenio proposed to use a mechanism of correlating changes in arm postures with changes in pixel values, while performing rhythmic movements to locate the visual region of the end-effector in the camera image [6]. Kemp and Edsinger [7] used mutual information to estimate the amount to which the robot's actions had control over image patches of different appearances and showed that their robot could learn the appearance of its end-effector and that it was controllable. Gold and Scassellati trained a graphical model to use probabilistic reasoning over time to classify image regions as belonging to the robot or being animate or inanimate "other" [8]. In contrast, we aim at showing that it is not necessary to implement a separate self-detection step into the learning process, and that the process of learning the body-schema can self-organize.

II. Simulation

We used a simulated robot in our experiments, as depicted in Figure 1. The robot possesses a two-degrees-of-freedom (2DOF) planar arm and a movable camera, also with 2DOF. It is assumed that the robot is equipped with controllers that can stably drive its effectors (i.e. arm and camera) into desired postures, specified as coordinates in the respective motor spaces.

As sensory input the robot receives proprioceptive feedback and visual input. The proprioceptive feedback provides the robot with information on the current positions of the camera and arm (in the same format as the desired postures that are sent to the controllers, i.e. coordinates in the respective motor spaces). The visual input provides the location of one object in the robot's visual field, as a coordinate in the image frame. Note that several objects can be in the visual field simultaneously. We assume some kind of attention mechanism is working in the robot, by which one of the visible objects is randomly selected to be attended. This can either be an object (or salient region) in the visual background scene, or the visual percept of the robot's end-effector. They are not treated any differently in the input, but are all subject to being attended. More specifically, we simply generate a fixed number of objects in the visual field of the robot and let the system select one of them or the robot's end-effector, all with equal probability.

Thus, it is assumed that the vision problem is solved insofar as it is abstracted from raw camera input to the coordinate of the attended visual stimulus, which can either be in the background or on the robot's end-effector. In our simulation, we only require that the robot is able to visually track a stimulus while moving and to recognize, when the stimulus has disappeared. This could be implemented for example by using an optical flow based method, or by using a saliency-based visual attention system based on proto-object descriptors (e.g. [9]).When assuming that objects in the world only change their appearance slowly (e.g. while being rotated), simply performing a nearest neighbor search in the proto-object feature space between frames should suffice, with a predefined threshold for rejection.

In the remainder of this paper, we use mathematical notations to refer to the inputs and outputs of the robot. Table I is an overview of the notations used for description.

TABLE I
List of notations

$\mathcal{S}_c \subset \mathbb{R}^2$	Camera motor space
$\mathcal{S}_a \subset \mathbb{R}^2$	Arm motor space
$\mathcal{S}_v \subset \mathbb{R}^2$	Visual location space
$x_c \in \mathcal{S}_c$	Proprioceptive feedback of the camera posture
$u_c \in \mathcal{S}_c$	Desired camera posture sent to controller
$x_a \in \mathcal{S}_a$	Proprioceptive feedback of the arm posture
$u_a \in \mathcal{S}_a$	Desired arm posture sent to the arm controller
$x_v \in \mathcal{S}_v$	Position of attended proto-object

A. Problem Statement

In a robotic setup as the one described above, the problem of learning a body-schema can be described as follows. As the robot should learn, how its actions directly influence the input from its sensors, there are two contiguities to be learned:

1) *Saccades* – Movement of the camera causes the location of visual stimuli to shift.
2) *Camera-arm coordination* – Movement of the arm causes the visual perception of the end-effector to move.

Also, of course, simultaneous movement of both arm and camera results in a superposition of the two changes. The proprioceptive input resultant from the robot's movements does not have to be learned, as it is identical to the desired postures sent to the controllers, plus some motor error, which can be assumed to be small.

Given a static environment, the robot should thus learn a mapping of the kind

$$f\left(x_c(t), u_c(t), x_a(t), u_a(t), x_v(t)\right) = x_v(t_c), \qquad (1)$$

which governs how the position of a visual stimulus (end-effector or environment) is changed as a consequence of an action of the robot. $t_c > t$ denotes the point in time when all controllers of the robot have converged to their target positions.

An important point to note here is that the visual stimulus input x_v can either originate from a background object or from the robot's end-effector. Formally we can say that there is a discrete latent variable, which indicates wether the robot is attending its own end-effector or a part of the background scene. This means that the robot has not yet learned about the appearance of its end-effector or how to discriminate it from other visual stimuli. The procedure presented in this work could actually be used to also bootstrap the learning of the appearance of the end-effector, but we do not consider this here.

B. Training Data

The robot should only use training data which it can acquire on its own. We let the robot perform random movements ("motor babbling") by consecutively generating random target postures in the robot's camera and arm motor spaces. Before these are sent to the controller, we also let the system randomly select one of the objects in the camera's visual field to be "attended", i.e. tracked during the movement.

Thus, the training data that the robot generates is composed of the state of the robot's sensors and the motor targets before the movement,

$$X^t = \langle x_c(t), u_c(t), x_a(t), u_a(t), x_v(t) \rangle \tag{2}$$

and the state of the robot sensors after the movement,

$$Y^t = x_v(t_c). \tag{3}$$

As stated above, $x_c(t_c)$ and $x_a(t_c)$ are assumed to be sufficiently close to $u_c(t)$ and $u_a(t)$ respectively, since the robot generates movements by specifying a target posture for the controller, and thus these do not have to be learned by the robot.

The training set is the set of all training samples $\langle X^t, Y^t \rangle$,

$$S = \left\{ S^t, t = 1 \dots T \right\} \tag{4}$$

$$= \left\{ \left\langle X^t, Y^t \right\rangle, t = 1 \dots T \right\}. \tag{5}$$

III. Computational Model

Directly trying to learn an estimate of the function f (cf. Equation 1) using the whole training set S is bound to produce bad results, since the information of the latent variable is missing: The system does not know a priori, when it is looking at its own end-effector and when it is looking at the background scene. However, the outcome $x_v(t_c)$ of the robots movement depends on this latent variable, and as a consequence the data represents samples from two different underlying functions in one space. Therefore, the system requires some kind of mechanism to classify the training samples, which is effectively why previous approaches have used a self-detection mechanism, so that only examples of the one kind are being generated for the training (cf. Section I). In contrast, we propose to let the system *self-organize* in classifying the training samples, using a mechanism that is functionally related to the well known *EM* algorithm [10].

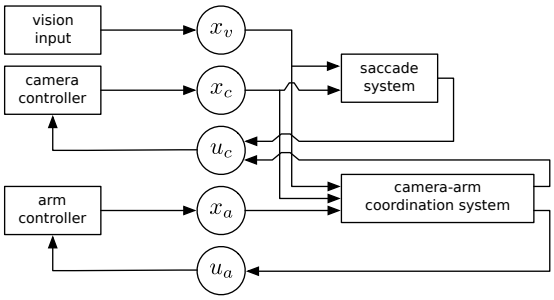


Fig. 2. Schematic overview of the computational model.

Our computational model therefore consists of two systems (see Figure 2), which should learn estimates of the mappings between sensory inputs and motor commands. On the one hand, the *saccade system* receives vision input and proprioceptive feedback from the camera, and generates targets for the camera controller. This system should enable the robot to learn how it can control the positions of static objects in its visual input by moving its camera and how to look at a target visual stimulus. On the other hand, the *camera-arm coordination system* additionally receives proprioceptive feedback from the arm and also generates targets for the arm controller. This system should learn how the visual perception of the robot's own end-effector can be controlled by the robot's motors (either camera or arm), and how to look at the position of the own end-effector or to bring the end-effector to a target visual location.

The two systems consist of estimates of forward mappings $f(\cdot)$, which transform current sensory inputs and motor commands into predicted sensory stimuli, and inverse mappings $g(\cdot)$, which return the motor commands necessary to transform a currently perceived situation into a goal situation, specified by a desired sensory value. In the saccade system, the two estimates

$$f_{\text{sacc}}\left(x_c(t), u_c(t), x_v(t)\right) = x_v(t_c) \tag{6}$$

and

$$g_{\text{sacc}}\left(x_c(t), x_v(t), x_v^*(t)\right) = u_c(t), \tag{7}$$

are estimated, where x_v^* is the desired visual location used for the inverse mapping.

The camera-arm coordination system is composed of the forward estimate

$$f_{ca}\left(u_c(t), u_a(t)\right) = x_v(t_c), \tag{8}$$

and the two inverse estimates

$$g_{ca}^a\left(x_c(t), x_v^*(t)\right) = u_a(t), \tag{9}$$

and

$$g_{ca}^c\left(x_a(t), x_v^*(t)\right) = u_c(t), \tag{10}$$

where g^a_{ca} generates an arm posture that will bring the end-effector to a desired visual location $x^*_v(t)$, given the camera posture $x_c(t)$, and g^c_{ca} generates a camera posture given a desired visual location $x^*_v(t)$ and an arm posture $x_a(t)$.

A. Bootstrapping the Learning Process using Preliminary Model Predictions

Similar to how the learning of a body-schema is related to the question of which input signals are *controllable* by which effectors, the problem of dividing up the training samples is equivalent to the question of which samples are *predictable* by which forward estimates. By having split up the input space of f and only keeping those dimensions that are necessary to compute the estimates (e.g. omitting the arm target u_a in the input for the estimate f_{sacc}), we have projected the training samples into lower-dimensional subspaces. This has the effect that for example in the input space for the saccade system, only the training samples belonging to the kind "looking at the visual background" can be described as a function. All other training samples are uncorrelated noise, as they depend on omitted input dimensions: They are *unpredictable* on the basis of this subspace.

To make this idea clearer, let us consider a simple example in three dimensions (see Figure 3). Let $f_1 \colon \mathbb{R} \to \mathbb{R}$ and $f_2 \colon \mathbb{R} \to \mathbb{R}$ be two arbitrary functions. We generate a training set by consecutively drawing two random inputs $x_1, x_2 \in \mathbb{R}$ from some distribution (e.g. uniformly distributed in the interval $[-5, 5]$ in our example). We then compute y based on a latent random variable either as $y = f_1(x_1)$ or as $y = f_2(x_2)$ and combine the three values into a training sample, $\langle x_1, x_2, y \rangle$. If we now omit one input by projecting the training samples into a subspace, e.g. by only considering the training samples as pairs $\langle x_1, y \rangle$, all examples that were generated by the function f_2 are randomly distributed across the input space. Only those examples that were generated by f_1 are samples from a function that can be learned.

We exploit this property of the data in the following way: We train estimates for the two functions (e.g. using multi layer perceptrons) and compute predictions of the target value y for the whole training set for each of the estimates and compute the squared error between the predicted and observed values. We then update each estimate using only those training samples, where the estimate produced the lowest error. These two steps are then repeated for a given number of times or until the mean squared error falls below a desired threshold.

Coming back to our robot scenario, we now want to formalize this procedure. Let X^t_{sacc}, X^t_{ca}, Y^t_{sacc} and Y^t_{ca} be the components of X^t and Y^t, corresponding to the inputs and outputs of the mappings f_{sacc} and f_{ca}. For each training sample S^t, predictions are generated using the preliminary estimates and compared to the actually observed data. For this we use the squared error for the forward estimates in each of the two systems, i.e. for $s \in \{\mathrm{sacc}, \mathrm{ca}\}$,

$$E^t_s = (Y^t_s - f_s(X^t_s))^2. \tag{11}$$

Based on these error values we want to estimate, whether the training sample S^t is an observation of the kind represented by the saccade system (i.e. the robot looked at the static background and moved its camera) or the camera-arm coordination system (i.e. the robot looked at its end-effector and moved its camera and/or its arm). For this, we introduce the binary weighting scheme

$$w^t_s = \begin{cases} 1 & \text{if } \nexists\, r \in \{\mathrm{sacc}, \mathrm{ca}\}, r \neq s : E^t_r < E^t_s, \\ 0 & \text{otherwise.} \end{cases} \tag{12}$$

Thus, an example is only used for training an estimate if the associated weight w^t_s equals 1.

Figure 3(b)-(d) is a visualization of the first four iteration steps of this mechanism for the simple three-dimensional example.

B. Using the Learned Estimates

After the training, the robot could use the learned estimates in a way related to employing a body-schema, e.g. for reaching for a visually perceived target, in the following way: When the robot's camera is in some posture x_c and the robot sees an object at a position x_v in its visual input space, these values could be provided as input for the inverse estimate g^a_{ca}, which would in turn provide the necessary arm motor command to bring the end-effector to the position of the object. The other way around, when the robot has its arm in some configuration x_a and should move its camera to a position that brings the end-effector to a position x_v in its camera input, the inverse mapping g^c_{ca} could be employed, yielding the necessary camera motor command.

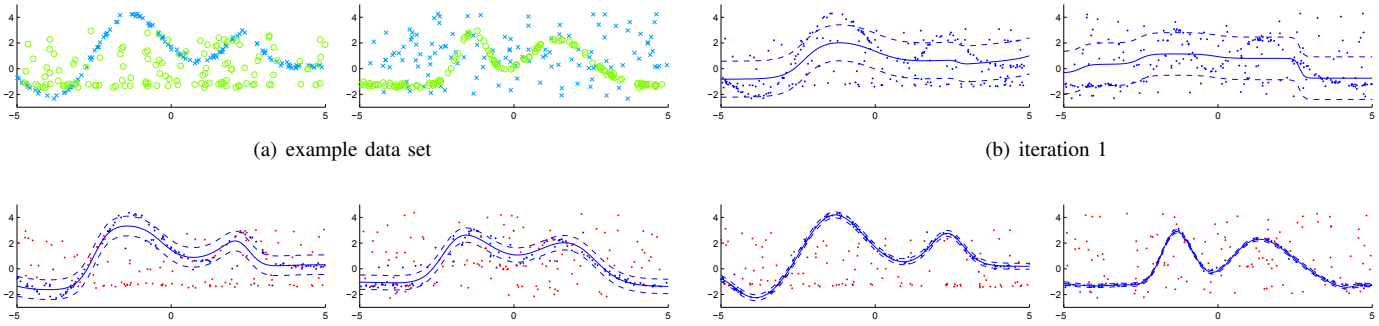
IV. Simulation Results

To test the proposed mechanism, we used multilayer perceptrons (MLPs) for estimating the forward and inverse mappings, each with 15 neurons in one hidden layer. However, in principle any regression method can be used.

We generated a training set S by letting the robot perform random movements. Before each movement, four objects were placed randomly in the visual scene observed by the robot's camera. On average, the robot thus had a 20% chance to look at its own end-effector and an 80% chance to look at an object belonging to the visual background scene.

We then iterated the training process outlined above 10 times. At each iteration step, we randomly drew 1000 training samples from S and used the estimates to generate predictions for those samples. The samples were then assigned to the saccade system and the camera-arm coordination system, based on the rule defined in Equation 12. The assigned training samples were further separated into a training set, a validation set and a testing set, all three equally large. The training set was used to train each MLP in the system using the Levenberg-Marquardt algorithm for optimization, and using the validation set to prevent over-fitting.

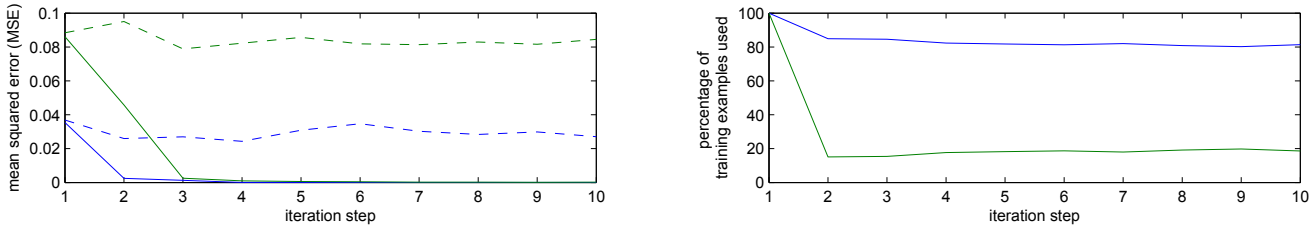
Figure 4(a) shows the development of the mean squared errors computed for the forward estimates across the training



(a) example data set (b) iteration 1

(c) iteration 2 (d) iteration 4

Fig. 3. (a) The two plots show an example training set where green circles correspond to one kind of observation and blue crosses correspond to another kind of observation. The two plots represent projections of the training samples into two-dimensional subspaces, with a one-to-one correspondence of points in the two plots. It can be seen that the green circles are random noise (with some non-trivial distribution in the vertical dimension) in the left plot, whereas the blue crosses are random noise in the right plot. (b)-(d) Using the mechanism described in this work, the training of estimates for the two functions can self-organize to associate the correct training samples for training the estimates. Blue dots represent training samples that are used to update the estimate, red dots the remaining training samples that are discarded for the training of the estimate. Solid blue lines show the estimates, dashed lines represent the mean squared error of the current estimate for the associated training samples. Initially, also wrong training samples are used for the training. However, the closer the estimates approach the true functions, the better their prediction becomes, and thus the training input for the estimates has less and less noisy data.



(a) (b)

Fig. 4. (a) The mean squared error of the forward estimates in the saccade system and the camera-arm coordination system. Solid lines show the development of the mean squared error when the proposed mechanisms are employed. Dashed lines show the development of the error when training without using the mechanisms. (b) The number of training examples that were assigned to the saccade system and the camera-arm coordination system. The true average numbers of 80% and 20% are quickly approached.

iterations using the testing set. In the first iteration, both estimates are trained using all training samples, resulting in a first rough estimation of the underlying functions. In the following iteration steps, both forward estimates quickly approach the underlying true function, as the mean squared error drops toward zero. In comparison, continuing to acquire training examples and training both estimates using all observations does not improve the performance at all: the mean squared error in this case remains at the same level as after the first iteration step.

Figure 4(b) plots the number of training examples that were assigned to the two systems. It can be seen that initially too many examples are used, but after only a few iterations, the training process is able to provide the two systems with approximately the correct number of training examples, i.e. 80% for the saccade system and 20% for the arm-camera coordination system.

V. Discussion

As we have shown in our simulation experiment, a robot can learn about the way its effectors can influence its sensory inputs as in a body-schema without the need to previously apply a separate self-detection step. We have used a competitive training mechanism, which is related to the idea of the EM algorithm: The training set is partitioned based on the performance of preliminary estimates, which are then updated based on the partitioned training data.

We have proposed to project the training data into lower-dimensional subspaces before using the predictions to separate the examples. Applying the same mechanism by training competing estimates in the whole input and output space would also allow to accurately represent the training data. However, the estimates would separate the input space in an arbitrary manner, not capturing the true underlying separation. We therefore propose that in a more complex scenario, in which there are more than just two relationships to be learned, the system should learn mappings on increasingly higher subspaces. Our intuition is that the same competition that we have show in this work to be helpful in discriminating two kinds of data points could also help in learning higher-dimensional problems by filtering out input that can be explained by lower dimensional estimates, and thus reducing the amount of noise in the training data for higher-dimensional estimates.

A. Relationship to the Concept of Schema

The idea of the mechanism that we have outlined, namely using preliminary predictions of competing systems to separate inputs, is to some degree congruent with a concept that was originally formulated in Psychology, which is the *schema*.

The concept of schema is an approach to give an explanation for the way that knowledge is organized in the human brain. Schemata provide a framework in which experiences are stored and abstracted to derive a definition of what is relevant, e.g. important sensory information in a certain situation, relevant next actions, or relevant properties of objects, etc. Even though the concept of schema has been widely used across disciplines for almost a century (e.g. [11], [12], [13]), definitions have still not converged.

However, a central aspect of the concept of schema that we want to highlight, is that knowledge is stored in a very structured manner. The work of Minsky on *frames* [14], which was very influential not only in the field of artificial intelligence, captures this idea in schma-like symbolic structures to implement knowledge in artificial systems.

More recent work on the topic is often focusing on giving embodied approaches for schemata. E.g. in psychology, Barsalou gave a comprehensive account for human cognition based on an embodied approach of acquiring and representing knowledge in schema-like mental structures, which he called "perceptual symbols" [15]. The accompanying concept of *image-schema* is often used for giving an account for abstract thought on the basis of embodied experiences (e.g. [16]).

In neuroscience, the concept of schema also has been used, where correlates for the structure provided by schemata on a conceptual level are sought for on a neural level [13], [17]. Often this is done in the context of discussing a *grasp* concept and put in relation to the mirror neuron system [18], where a grasp schema is described as connecting motor programs to the sensory context relevant for the action.

One important property of the concept of schema, which we see as to some degree congruent with the mechanism described in this work, is the active nature of schemata as they are being used in perception. Schemata are thought of as active elements of the cognitive apparatus, which actively seek the information which is relevant for them. For example, whenever we are confronted with the image of a car, our past experience with cars, organized in a *car schema*, becomes active and governs the way we reason about the stimulus being presented to us and puts it in relation to our experiences and knowledge [15]. For example, we will immediately know that we can use the car for driving. Piaget called the ability to map new percepts of objects onto existing schemata "assimilation" [12]. In more technical terms, Rumelhart [19] therefore described schemata as "recognition devices whose processing is aimed at the evaluation of their goodness of fit to the data being processed (p. 169)."

If schemata are seen as structural elements associating information in a framework for knowledge, and these structural elements dynamically compete in explaining the input that is presented to the cognitive system, then we can conclude that this competition could aid the system in organizing the information, at least on the level of low complexity as in our simulation experiments. When interpreted from a schema theorists point of view, our results can be seen as an instance where the assimilation property of schemata helps in organizing and learning about incoming information.

ACKNOWLEDGMENT

Nikolas J. Hemion gratefully acknowledges the financial support from Honda Research Institute Europe.

REFERENCES

[1] H. Head and G. Holmes, "Sensory disturbances from cerebral lesions," *Brain*, vol. 34, no. 2-3, pp. 102–254, 1911.

[2] M. Hoffmann, H. Marques, A. Arieta, H. Sumioka, M. Lungarella, and R. Pfeifer, "Body schema in robotics: A review," *IEEE Transactions on Autonomous Mental Development*, vol. 2, no. 4, pp. 304–324, 2010.

[3] G. Metta, G. Sandini, and J. Konczak, "A developmental approach to visually-guided reaching in artificial systems," *Neural Networks*, vol. 12, no. 10, pp. 1413–1427, Dec. 1999.

[4] C. Gaskett and G. Cheng, "Online learning of a motor map for humanoid robot reaching," in *Proc. of the 2nd Int. Conference on Computational Intelligence, Robotics and Autonomous Systems*, 2003.

[5] A. Stoytchev, "Self-detection in robots: A method based on detecting temporal contingencies," *Robotica*, vol. 29, pp. 1–21, 2011.

[6] P. Fitzpatrick and A. Arsenio, "Feel the beat: using cross-modal rhythm to integrate perception of objects, others, and self," in *Proc. of the 4th Int. Workshop on Epigenetic Robotics*. Genoa, Italy: Lund University Cognitive Studies, 2004.

[7] C. C. Kemp and A. Edsinger, "What can I control? a framework for robot self-discovery," in *Proc. of the 6th Int. Conference on Epigenetic Robotics*, Paris, France, 2006.

[8] K. Gold and B. Scassellati, "Using probabilistic reasoning over time to self-recognize," *Robotics and Autonomous Systems*, vol. 57, no. 4, pp. 384–392, 2009.

[9] M. Wischnewski, A. Belardinelli, W. X. Schneider, and J. J. Steil, "Where to look next? combining static and dynamic proto-objects in a TVA-based model of visual attention," *Cognitive Computation*, vol. 2, no. 4, pp. 326–343, 2010.

[10] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 39, no. 1, pp. 1–38, 1977.

[11] F. C. Bartlett, *Remembering*. Oxford, England: Oxford University Press, 1932.

[12] J. Piaget, *Biology and knowledge: an essay on the relations between organic regulations and cognitive processes*. Chicago: University of Chicago Press, 1971.

[13] M. A. Arbib, "Schema theory," in *The handbook of brain theory and neural networks*. Cambridge, Mass.: MIT Press, 1998, pp. 993–998.

[14] M. Minsky, "A framework for representing knowledge," in *The Psychology of Computer Vision*, P. H. Winston, Ed. New York [u.a.]: McGraw-Hill, 1975, pp. 211–277.

[15] L. W. Barsalou, "Perceptual symbol systems," *Behavioral and Brain Sciences*, vol. 22, no. 4, pp. 577–660, 1999.

[16] J. M. Mandler, "Perceptual and conceptual processes in infancy," *Journal of Cognition and Development*, vol. 1, no. 1, pp. 3–36, 2000.

[17] V. Gallese and G. Lakoff, "The brain's concepts: the role of the sensory-motor system in conceptual knowledge," *Cognitive Neuropsychology*, vol. 22, no. 3, pp. 455–479, 2005.

[18] M. A. Arbib, "From Monkey-Like action recognition to human language: An evolutionary framework for neurolinguistics," *Behavioral and Brain Sciences*, vol. 28, no. 2, pp. 105–124, 2005.

[19] D. E. Rumelhart, "Schemata and the cognitive system," in *Handbook of social cognition*. Lawrence Erlbaum Associates, 1984, vol. 1, pp. 161–188.