Bootstrapping Cognition from Behavior—A Computerized Thought Experiment

Ralf Möller, Wolfram Schenck

Computer Engineering Group, Faculty of Technology, Bielefeld University

Received 16 March 2006; received in revised form 6 October 2007; accepted 29 May 2007

Abstract

We show that simple perceptual competences can emerge from an internal simulation of action effects and are thus grounded in behavior. A simulated agent learns to distinguish between dead ends and corridors without the necessity to represent these concepts in the sensory domain. Initially, the agent is only endowed with a simple value system and the means to extract low-level features from an image. In the interaction with the environment, it acquires a visuo-tactile forward model that allows the agent to predict how the visual input is changing under its movements, and whether movements will lead to a collision. From short-term predictions based on the forward model, the agent learns an inverse model. The inverse model in turn produces suggestions about which actions should be simulated in long-term predictions, and long-term predictions eventually give rise to the perceptual ability.

Keywords: Cognitive architecture; Situated cognition; Computer simulation

1. Introduction

To illustrate the problem our brain is facing when interpreting sensory information, Dennett (1978) asks the reader to imagine being locked in the windowless control room of a giant robot, walls covered with lamps indicating the sensory state of the robot and with buttons connected to its effectors. None of the lamps and buttons is labeled, but the operator nonetheless has to make sense of the flickering lights and press the right buttons to bring the robot through the day. Phrased like this, the task appears to be close to impossible, even if some lamps would represent the results of a sophisticated perceptual analysis of sensory signals. So, how can the brain learn to assign meaning to sensory signals, how can it “bootstrap its way to understandings it was not born with” (Dennett, 1998, p. 73)? In this paper, we use a computer simulation to exemplify a sensorimotor theory of cognition tackling this question. We will return to the control room metaphor after outlining the assumptions of this theory.

Correspondence should be addressed to Ralf Möller, Computer Engineering Group, Faculty of Technology, Bielefeld University, D-33594 Bielefeld, Germany. Web site: ti.uni-bielefeld
The cognitive sciences have long been dominated by a “disembodied” view on perceptual and cognitive abilities. Brains were thought of as devices for abstract information processing, being only loosely connected to the outside world through senses and actuators. If behavior was considered at all, the units responsible for producing appropriate actions were seen as a final and separate stage of processing which would merely transform the results of cognitive analysis into actions. In the last two decades, however, an alternative perspective has emerged (or, in some aspects, resurfaced) which acknowledges that perceptual and cognitive abilities are tightly interwoven with behavior. Experiments in cognitive psychology, for example, revealed a strong mutual interaction between perceptual processes and action planning. These observations led to the formulation of “common coding” theories stating that perceived events and planned actions share a common representational framework. More specifically, common coding theories rest on the assumption that actions are represented by their anticipated sensory effects (Prinz, 1997; Hommel et al., 2001). Anticipation of action effects requires an integration of perceptual states and motor commands at the input of “forward models” (internal predictors of action effects), a structural property which departs from the strictly linear processing in the above-mentioned “sense-think-act” concept.

A similar, sometimes even more radical paradigm change could be observed in artificial intelligence (AI) research after it transpired that the view of intelligence as symbol-based reasoning has not led to truly intelligent behavior despite decades of research, and might therefore be fundamentally flawed (Brooks, 1991; Pfeifer & Scheier, 1999). Most alternative approaches emerging after the downfall of “classical” AI share the idea that having a body and interacting with the world is an indispensable prerequisite for achieving cognitive competences; the rather diverse approaches are therefore subsumed under the labels “behavior-based” or “embodied” AI. Wilson (2002) reviews some specific claims underlying embodied concepts of cognition. While many approaches stress the execution of behavior as prerequisite for cognition, some accept that “even when decoupled from the environment, the activity of the mind is grounded in mechanisms that evolved for interaction with the environment” (Wilson, 2002, p. 626). Also the model presented in this paper is an instance of such “offline” approaches to behavior-based cognition.

We feel that, although it was not born in this framework, our model can best be understood as an instance of Gibson’s “ecological” approach to cognition (Gibson, 1979), one of the early sensorimotor theories of perception; a clear account of its basic assumptions is given by Norman (2002). Central to Gibson’s ideas is the notion of “affordances”: By picking up certain “invariants” from the visual information, an observer directly perceives the behavioral meaning of an object. Interestingly, Norman (2002) points out that Gibson’s examples of “affordances”—a chair being “sit-on-able,” a stair being “step-on-able”—always include some action by the observer, like sitting down, stepping on, etc. Pursuing this line of thought, we assume that affordances are revealed by a process of internal simulation of action consequences. Based on the present visual information, the observer internally simulates sequences of actions and anticipates their outcome. Given the image of a chair, this perceptual process might predict the tactile impressions and the feeling of support perceived as result of the behavioral sequence of sitting down, and it is these predictions that give the observer the impression of the chair being “sit-on-able.” Thus, in contrast to Gibson’s ideas on how affordances are perceived, the behavioral meaning of the object is acquired actively by
a sensorimotor simulation process that is influenced by visual information, rather than being directly obtained from the visual image by a passive analysis of sensory information.

As pointed out above, the internal simulation takes place completely “offline”: No action is overtly executed during the simulation, but the simulated actions are only driving the anticipation process. Looking at an object will automatically trigger this simulation process, far below the level of conscious engagement with the sensory information. Support for this idea comes from several experiments in visual priming: Even if the action is completely irrelevant for the task at hand, seeing an object seems to automatically “potentiate” actions involving this object, like left- vs. right-handed manipulation or precision vs. power grip (Tucker & Ellis, 1998; Ellis & Tucker, 2000; Tucker & Ellis, 2001). The potentiation effect is revealed since the pre-activated actions interfere with other actions which have to be executed by the experimental subjects. It is conjectured that intentional behavior could just pick from the “afforded” actions, finally resulting in the execution of an appropriate action (Tucker & Ellis, 1998). That motor representations are involved in the perception of objects is also indicated by neuroimaging studies (Grèzes & Decety, 2002).

Several authors have put forward similar “simulation theories” of cognition (Hesslow, 2002; Cruse, 2003; Wolpert et al., 2003; Holland & Goodman, 2003; Grush, 2004). These theories attempt to explain perceptual, cognitive, and behavioral competences including mental imagery, detection of anomalies, development of motor plans, foresighted selection of behavior, understanding of observed actions, imitation, social interaction, and even conscious thought. Moreover, our ideas have close correspondences to the theory of “sensorimotor contingencies” advocated by O’Regan and Noë (O’Regan & Noë, 2001; Noë, 2004), although their theory is not directly related to internal simulation and aims primarily at an explanation of qualia. The theory described in this paper is restricted to a very basic perceptual ability, the understanding of the spatial arrangement and the shape of objects. What it ultimately strives to explain is the ability to “perceive” or “understand” the behavioral meaning of objects from their visual appearance. In this attempt, the theory deviates from common coding theories (Hommel et al., 2001) and from Grush’s “amodal spatial imagery” which both presume some level of perceptual processing like object recognition to which the simulation process can resort. Our approach, in contrast, deliberately works on “raw” sensory information like pixels, edges, or, in our case, image regions (“modal emulation” in Grush’s terminology). It thus takes an extreme stance by claiming that even some very basic perceptual competences rest on sensorimotor simulation, and higher cognitive abilities like object recognition emerge from these basic competences (Möller, 1999).

Simulation theories are closely linked to the concept of internal models, structures within the central nervous system which either model the causal relationship between actions and their sensory effects (forward models), or are able to suggest motor commands for achieving a desired sensory effect in a given situation (inverse models) (Wolpert & Ghahramani, 2000; Karniel, 2002). Both types of models can be learned by neural network approaches in the interaction with the environment. Compared to the learning of inverse models which is usually hampered by the missing teacher and by the necessity to explore high-dimensional spaces (Hoffmann et al., 2005), forward models can simply be acquired by behaving randomly and observing the sensory consequences. In this paper, the simulated agent uses multilayer perceptrons to predict the image of a panoramic camera from the previous image and the
movements of the agent (visual forward model), and to predict collisions from the same input signals (tactile forward model).

In our concept of “action potentiation,” the visual impression of an object does not only trigger a single sequence of simulated actions, but a multitude of different sensorimotor sequences. It is evident that an exhaustive simulation of all possible action sequences is out of the question. However, if intentional behavior should be able to later select one of the simulated sequences, it has to be guaranteed that “useful” sequences are contained in the simulation process. The problem becomes even more severe when a specific goal should be reached. There are two solutions to avoid exhaustive search. With a goal state given, the arising search problem can be reformulated as an optimization problem: Some distance measure between predicted and desired final state is minimized over the space of motor sequences (Hoffmann & Möller, 2004; Hoffmann, 2005). Alternatively, upholding the idea of searching through the sensorimotor space, the search can be restricted to the “most promising” (Gross et al., 1999) or “typical” actions (Möller, 1999). These actions are suggested by an internal structure which can be interpreted as an inverse model: At each point in the sensorimotor simulation process, the inverse model suggests one or a small number of actions which previously proved to be successful. In this paper, the inverse model is also learned. Training data are produced by internal simulations of sensorimotor sequences that become possible once the visual and tactile forward model are available. Since we assume that the inverse model does not have to perform complex perceptual processing to suggest typical actions, linear functions are used as modules of the inverse model; the linear approximations of the training data are produced by partial least squares regression, a method from chemometrics (Wold, 1975; Frank & Friedman, 1993; Garthwaite, 1994; Helland, 2001). This inverse model suggests a single action, and the search through the space of sensorimotor sequences is accomplished by disobeying its suggestion in some randomly chosen situation in the sequence and replacing it with some random action.

Since shape recognition of complex objects as the ultimate test case for our approach is not yet in reach, this paper demonstrates the concept of anticipatory shape “recognition” by means of a dead end recognition task that we had introduced as a thought experiment in our previous work (Möller, 1996, 1999). The task for the agent is to recognize an arrangement of obstacles in front as either a “dead end”—an arrangement with the behavioral meaning that an escape in the current direction of movement is impossible—or as a “corridor”—an arrangement which can be passed without turning around. Despite its simplicity, we think that the task bears strong similarities to shape recognition of objects, since it is the overall shape of the obstacle arrangement which determines its behavioral meaning. By random movements in a simple training environment, the agent acquires knowledge on how the visual image of its panoramic camera changes in response to its own movements. In addition to the visual sense, the agent can feel collisions with obstacles; a collision is encoded by a single binary signal regardless of its location on the agent’s body.

Computer simulations and robot experiments are meanwhile established as a standard method for the investigation of cognitive models; the following studies are closest to our approach. Mel (1988, 1989) employed a kinematic visuomotor forward model together with a differential inverse model to guide a robot arm to a target position, even in the presence of obstacles. Gross et al. (1999) and Stephan and Gross (2003) describe a mobile robot which learned to predict optical flow patterns and used the prediction for foresightedly avoiding
precarious situations in an obstacle avoidance task and for distinguishing between self- and externally-generated optical flow. Moreover, the anticipatory ability helped to overcome disturbances of the flow detection caused by fluctuating ambient light. Also in the domain of mobile robot control, Hoffmann & Möller (2004) and Hoffmann (2006) demonstrated that chains of forward models can be used for the mental transformation of visual information and for the goal-directed planning of action sequences. Datteri et al. (2004) apply visuomotor forward models in a block-pushing task performed by a robot manipulator to explore ways for replacing perceptual processing by prediction. While also using a form of forward models, the work by Tani and Nolfi (1999) and Ziemke et al. (2005) goes into a somewhat different direction from our approach: In their work, the robot’s tasks and its internal models are always related to a specific environment, while in our model the internal models can be applied to arbitrary sensory situations. Oudeyer et al. (2007) use forward models to guide the learning process of an artificial agent. The curiosity of the agent is influenced by an internal model which predicts the error made by the forward model. This evaluation keeps the robot away from situations which are already predictable, but also from those which appear to be too unpredictable. Elements of our approach are also present in the recent work of Shanahan (2006): Decisions taken by inverse models can be vetoed by an internal simulation of action consequences based on affective responses. The main goal of this work seems to be an exploration of a neural architecture based on the “global workspace theory” which promises to explain the distinction between conscious and non-conscious thought processes.

We now return to the question raised in the beginning: How can the brain make sense of sensory signals? We would like to continue from the control room metaphor by suggesting that the only way to assign meaning to the flickering lamps will be to observe how the lamps change when the different buttons are pressed. Causal relationships between actions and their sensory consequences should be relatively easy to detect and can be incorporated into a forward model by some generalizing learning method. However, the prediction alone will not bring the robot through the day: The selection of promising actions for execution will require some given assessment of the value of a true or predicted situation (or of an entire sequence). Thus some of the lamps should actually be labeled “good” or “bad” to indicate whether the situation is desirable for the robot or not. In the simulated agent, these quality-indicating lamps are connected to the tactile sensor of the robot—collisions are “bad”—and to its motor commands—moving forward is “good,” switching between translatory movements and turning is “bad.” We consider the assessment of tactile (rather than visual) signals and of actions to be an essential feature of this approach. The tactile sense is a proximal sense: Touch has a relatively direct behavioral meaning for a living being—think of food being inserted into the mouth or of getting bruises from colliding with obstacles. The same holds for actions which have immediate meaning since they are connected with costs. Vision, on the contrary, is a distal sense, and visual information often has no intrinsic value for the agent. The internal simulation is a “projection” from the distal visual sense onto the proximal tactile sense and onto motor commands and from there onto the assessment signals, and this projection is what ultimately bestows the visual image with behavioral meaning.

The remainder of the paper is organized as follows: The simulated agent and its environment is described in section 2. The following sections present the structure of the internal models (forward models in section 3, inverse model in section 4), the data collection, and
the training and test of these models. How forward and inverse models are applied for the internal simulation process is described in section 5. Section 6 demonstrates the perceptual abilities arising from the internal simulation process for the task at hand. The implications of the experiments are discussed in section 7.

2. Simulated agent and environment

The agent’s environment is an infinite horizontal plane on which cylindric obstacles are standing; Fig. 1 (left) shows a top view. All obstacles are of the same height (0.5 units), but have different diameters (varying from 0.5 to 0.9 units). To simplify the visual preprocessing, all obstacles in a given environment are colored differently. The simulated agent has a round body shape with a diameter of 1 unit. In a single step, it can either translate forward or backward by 0.2 units, or turn left or right around its center by approximately 15.3°. When a translatory movement would lead to a collision, it is not executed, but the collision sensor is activated.

A panoramic camera is mounted in the center of the agent’s body in 1.5 units height above the ground. The camera captures panoramic monochrome images (700 × 280 pixel) spanning the full 360°; an example is shown in Fig. 1 (top right). The forward direction of the agent corresponds to the horizontal center of the image. The camera position was deliberately set to be higher than the obstacles so that the arrangement of all obstacles is fully represented in a single camera image. The horizon is located at the upper margin of the image, the height of the image was chosen such that even nearby obstacles are completely visible in the camera.

Fig. 1. Left: Example environment of the agent in top view. The agent is shown as a circle with the forward direction indicated by a line. The ten cylindric obstacles are numbered to establish a link to the images on the right. The area is 8 × 8 units wide. Top right: Panoramic image obtained from the agent’s camera in this situation. The numbers correspond to the top view. Object outlines were added for clarity. Bottom right: Visual representation derived from the panoramic image. A segmentation procedure extracts the bottom center point and the width of each region (gray), visualized as black circles.
The rendering method is using simple trigonometric relations to approximate the perspective projection. The projection of each horizontal circular plane (a cylinder comprises two of them) is approximated by two half-ellipses centered at the cylinder. The body between the two planes appears as rectangle. Vertical view angles below the horizon are passed through the sum of a linear and a cubic function; this expands nearby cylinders in apparent height. Objects are drawn back to front to handle occlusions. The floor is uniformly white.

As we already mentioned in the introduction, our model assumes that there is a hierarchy of cognitive abilities which is grounded in a sensorimotor simulation process. The simulation process provides the agent with an understanding for the shape and function of objects which we see as a basic cognitive ability. Higher cognitive abilities like recognition of tools (objects with typical functions) or symbolic categorization rest on this basic level of spatial understanding. This assumption prohibits complex sensory processing beyond the level of the extraction of simple visual features from the image, since advanced sensory processing—like recognition of complex features or entire objects—would constitute a jump to higher levels of the hierarchy. In our example, “objects” are arrangements of multiple cylindrical obstacles, thus we can consider the image of each obstacle as an elementary visual feature (and not as image of an object). In the same way in which all visual features belonging to different parts of a chair have to be considered to understand the behavioral meaning of the chair as a whole, all single obstacle features determine the meaning of an obstacle arrangement in our simplified case. Therefore the only visual preprocessing performed in the agent’s brain is a segmentation procedure which returns the bottom center point and the width of each colored region (in the following called a “blob”); all obstacles being colored differently simplifies this step. In Fig. 1 (bottom right), each blob is visualized by a black circle: its lowest point is located at the bottom center point of the image region, its diameter is corresponding to the region’s width. In the implementation, each blob is characterized by its horizontal and vertical position in the image (bottom center point) and its size which is set to half of the region’s width.

3. Forward models

Our agent acquires two different forward models, a visual forward model which allows the agent to predict the next image from the current image and the given movement, and a tactile forward model which predicts from an image whether the execution of the given movement would cause a collision. Both the visual and tactile forward models are neural networks trained with data obtained from executing random movements in a training environment and registering both changes in the image and collisions. We describe the representation chosen for the image, the architecture of the multi-layer perceptrons used as predictors, the collection of the training examples, and the training regime of the networks, and evaluate the performance obtained from the trained predictors.

3.1. Representation

The agent translates and rotates by only small distances and angles, thus all visual features are only shifting by small distances within the image. Under this assumption, a visual forward
model could comprise a retinotopic array of local predictors, each observing a small region of the previous image and predicting whether a blob will appear in the center of its visual field and which size the blob would have. This would require one to train each local predictor separately and to update the entire array in each prediction step. To avoid the large computational effort, we simplified the predictor array in the simulation. A visual scene is represented by a blob list, with blobs being described by their position \((x, y)\) and size \((s)\). Each blob is passed separately through a single predictor, and a new blob list describing the entire predicted image is assembled from all predicted blobs. The predictor takes the three values \(x, y, s\) as input and predicts how this blob will change in position and size \((\Delta x, \Delta y, \Delta s)\) under the given movement. Fig. 2 (top left) shows the inputs and outputs of the abstract visual forward model.

The tactile forward model predicts from the given image and the planned movement whether a collision will occur after the movement was executed. Such a tactile predictor would have the entire image as its receptive field. In our simulation, the tactile predictor takes a blob as input rather than an image. A collision is predicted for a given image if a collision is predicted for at least one of the blobs. As will be shown below, the overall effect of this mechanism can be visualized as a receptive field spanning the entire image. We would like to point out, however, that the training is simplified by the fact that only a single blob is present in each
training image (see below); with multiple blocks it would not be known which blob caused the collision.

3.2. Network

Both visual and tactile forward model are implemented as multi-layer perceptrons. We first describe the implementation of the visual forward model. Just encoding the blob’s position and size and the movement at the input, and the change in the blob’s position and size at the output (as visible in the abstract model in Fig. 2, top left) did not produce sufficiently precise predictions. Therefore separate perceptrons receiving the same inputs are used to predict each output variable ($\Delta x$, $\Delta y$, $\Delta s$) for each movement direction (forward, backward, left, right); see Fig. 2 (top right). Moreover, the input image was split into an upper and lower portion which is captured by separate networks. The splitting is motivated by the large difference in image shifts for nearby and distant objects. Separating the two regions improves the accuracy of the prediction of small shifts for distant objects. In the training, blobs in image row $0 \leq y < 160$ are assigned to the “top” network, blobs in image row $120 \leq y \leq 279$ to the “bottom” network. The overlap between the two regions was introduced to avoid border effects. In the application of the networks, the border between the two regions is at row $y = 140$; blobs with $y < 140$ are handled by the top network, blobs with $y \geq 140$ by the bottom network.

Since the panoramic images are closed in $x$-direction, we provided two additional input signals that reflect this periodicity: The signal $x$-cos is the cosine of the horizontal view angle, the signal $x$-sin its sine. Thus, the network is given the freedom to either consider the absolute coordinate $x$ or the two circular values $x$-cos and $x$-sin for the prediction. At the output of the network, we decided to encode the changes in position and size rather than the position and size of the predicted blob. For the usually small changes in position and size, the network would otherwise have to represent a function which is almost an identity mapping. This turned out to be difficult for multi-layer perceptrons for some reason; directly encoding the deviations improved the prediction accuracy.

The tactile forward model uses the same input encoding as the visual forward model and four separate perceptrons, one for each movement. The collision is encoded as a single output value.

The same network architecture was used for all component networks of both visual and tactile forward model: 5 input neurons, 2 hidden layers with 5 neurons each, 1 output neuron. Adjacent layers are fully connected. Neurons in the hidden and output layers have the hyperbolic tangent as output function (output range $-1 \ldots 1$).

3.3. Data collection

The training data are collected in environments with only a single object. To determine the change in position and size of a blob resulting from a movement, the blob has to be identified in the image before and after the movement. In environments with multiple objects, each object could be identified by its unique color; even without unique coloring, the agent’s movements are small enough so that neighboring blobs extracted from the two images are likely to correspond to the same object if the scene is not too crowded. However, occlusions
would introduce errors in the training data, and if the object colors would not be unique, occasional mismatches could not be avoided. Since the performance of the forward model is crucial for the success of the internal simulation, we decided to exclude this potential source of performance degradation by only placing a single object in the training environment.

At the beginning of each run, the agent is positioned in the center of its arena, facing right. An object is positioned either before, behind, to the left, and to the right of the agent. Object diameters vary from 0.5 to 0.9 units in steps of 0.1. The object is placed at a distance of 1 unit from the agent (repeated 3 times for each direction), and at a distance of 2, 3, 5, and 7 units (each distance repeated once). In each situation, the agent executes a sequence of 100 random steps (forward, backward, left, and right movements are chosen with the same probability); this is repeated 10 times for each situation. Fig. 3 visualizes the movements of the agent in one training situation.

In each movement step, data for both the visual forward model and for the tactile forward model are collected. For the visual forward model, the blob’s location and size in the image is stored together with the change of the blob’s position in horizontal and vertical direction and its change in size, and together with the movement that was executed (one of the four movements forward, backward, left, and right). For the tactile forward model, the blob’s position and size, the binary collision signal, and the movement is registered.

![Image](image.jpg)

Fig. 3. Data collection for the forward models: The agent 10 times executes 100 movement steps, each sequence starting from the center of the arena. In this example, the obstacle (diameter 0.7 units) is initially located behind the agent. Lines depict the agent’s trajectory, and small dots (in the vicinity of the obstacle) mark locations from which the randomly chosen movement caused a collision.
3.4. Training

Inputs and outputs are linearly transformed such that the minimal value is encoded as $-0.5$ and the maximal value as $0.5$. For the visual forward model, minima and maxima are determined separately over all component networks responsible for the top part and over all component networks responsible for the bottom part. This also means that minima and maxima are determined over all four movements; this avoids problems with zero data range, as it occurs, for example, in $\Delta y$ for rotations (always 0). For the tactile forward model, the minima and maxima are determined over all available data.

For each movement, approx. 24 000 training examples are available for the top part of the visual forward model and approx. 12 000 for the bottom part. The data set for the tactile forward model is strongly imbalanced: For translatory movements, only approx. 1 000 collisions were registered, compared to approx. 35 000 non-collisions. Since this would overemphasize non-collisions, we randomly select as many non-collision examples as we have collision examples for the training. For rotations, no collision examples are available at all. Here we randomly select 1 000 training examples from the non-collision data.

All networks were trained with “vanilla” error backpropagation (Rumelhart et al., 1986) (learning rate 0.02, no momentum term). In each of the 500 training epochs, all training data were presented in randomly scrambled order. Weights were adapted immediately after the presentation of a data point (direct learning).

3.5. Test

To evaluate the performance of the blob predictor (visual forward model), we position the agent in the center of the arena (facing right) and place an object with an diameter of 0.7 units (an intermediate value from our range of obstacle diameters) at all possible positions on a $41 \times 41$ grid covering an area of $-10 \ldots 10$ units in each direction. The agent performs a movement, and the error between predicted and true blob size after the movement is registered. Table 1 presents the results for one translatory and one rotatory movement. For each predicted variable, the average absolute error, the minimal error, and the maximal error are specified. It is apparent that the average absolute error is well below one pixel for all variables, and even

<table>
<thead>
<tr>
<th>Movement</th>
<th>Variable</th>
<th>Avg. abs. error</th>
<th>Min. error</th>
<th>Max. error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Forward</td>
<td>$x$</td>
<td>0.36</td>
<td>$-1.44$</td>
<td>1.18</td>
</tr>
<tr>
<td></td>
<td>$y$</td>
<td>0.40</td>
<td>$-1.38$</td>
<td>1.38</td>
</tr>
<tr>
<td></td>
<td>$s$</td>
<td>0.18</td>
<td>$-0.94$</td>
<td>1.01</td>
</tr>
<tr>
<td></td>
<td>$x$</td>
<td>0.42</td>
<td>$-0.37$</td>
<td>0.76</td>
</tr>
<tr>
<td>Left</td>
<td>$y$</td>
<td>0.01</td>
<td>$-0.14$</td>
<td>0.04</td>
</tr>
<tr>
<td></td>
<td>$s$</td>
<td>0.003</td>
<td>$-0.012$</td>
<td>0.017</td>
</tr>
</tbody>
</table>
the maximal and minimal deviations are only in the order of one pixel. For the left rotation, errors in the \( y \) and \( s \) variable are very small since these values are not changing under rotation.

Fig. 4 gives a visual impression of the quality of a recursive prediction where, starting from a true image, the next predicted image is obtained from the previously predicted image, thus there is no feedback from the environment during this simulation process. In a recursive prediction, errors are slowly accumulating, but it is apparent that the overall arrangement of the objects is preserved even after 88 steps. Besides the performance limits of the predictor, deviations are also caused by ambiguities in the visual information. For example, an obstacle with a diameter of 0.7 units appears in the same size from a distance of 5.2 to 4.6 units, thus it is impossible to exactly predict the size of the obstacle. Also the coordinates of an object are rounded to integer values when the image is generated which prevents a precise prediction of its position. Since all three variables are inputs to the image predictor, there is also a cross interference between position and size.

As already mentioned above, the function of the tactile forward model can be visualized as receptive field covering the entire image. Fig. 5 shows the output of the collision predictor when a blob with size \( s = 40 \) is presented as input at all positions \( (x, y) \) in the image. The left image shows the output for a forward movement, the right image for a backward movement. The predictor plausibly predicts collisions (dark region) for a forward movement when the obstacle is positioned between approx. \(-90^\circ\) and \(90^\circ\) from the forward direction, and for a backward movement when the obstacle is positioned in the same range relative to the backward direction. The output for other blob sizes \( s \) is visually almost indistinguishable from the diagrams presented; since it is the bottom center point of a region that is encoded in a blob, the size of an obstacle has only a small influence on the predicted collision, as can be confirmed by a simple geometric analysis.

4. Inverse model

The inverse model’s task is to produce foresighted obstacle avoidance behavior. It suggests a movement (here only forward, left, right) given an image and the previous movement. If the actions suggested by the inverse model would immediately be executed, the model should be able to guide the agent through an arrangement of obstacles while trying to move as straight as possible. In our experiments, however, the suggested actions are not executed, but are guiding the internal simulation process. This focuses the search for desirable actions on the most promising ones and therefore restricts the sensorimotor search space. To achieve foresighted obstacle avoidance, the suggested movement should not end in a collision and should be chosen such that following the policy of the inverse model over a short period results in small movement costs; in our case, forward movements should be preferred over rotations, and switching between movements should be avoided. Although the inverse model only suggests a single movement, the movement has to be selected with the future costs in mind. Future costs can be predicted if the visuo-tactile forward model is available, so the training data for the inverse model can be obtained by selecting the best sequence of movements based on the predictions of the visuo-tactile forward model over a short prediction horizon. The inverse model comprises four linear modules, three for selecting the movement, one for avoiding
Fig. 4. Application of the visual forward model. The agent is moved manually through some arbitrary trajectory shown in the left column. The prediction starts from a single true visual image; in all subsequent steps, the next image is predicted from the previous predicted image. Simultaneously, the agent is executing the same movements to provide the true images for comparison. In the images in the right column, the prediction (predicted blobs displayed as black circles) is overlaid on the true visual image (gray regions) after 10, 26, 47, 68, and 88 steps. It is apparent that errors are slowly accumulating in the prediction process.
Fig. 5. Application of the tactile forward model. A blob with $s = 40$ was presented to the tactile forward model for each position $(x, y)$ in the image. The output of the tactile forward model for forward and backward movements is encoded as gray value, black corresponding to 0.6, white corresponding to $-0.6$. Horizontal ticks have a distance of 90°. Diagrams for other blob sizes $s$ are virtually identical.

collisions. We use the partial least squares regression method to determine the linear function approximations. In the following, the material is presented in the same order as for the forward models.

4.1. Representation

Since the inverse model has to take the overall arrangement of objects into account, its input has to be the entire image rather than a single blob. Therefore it is necessary to transform the blob list representation obtained from the visual preprocessing or from the visual forward model into an image; Fig. 6 shows an example. The coarse-grained image shown on the right is obtained in the following way: First, the blobs are drawn as filled circles (fill value 1.0, background value 0.0) into an image of the same resolution as the original input image plus a lower margin that was added to avoid border effects from the filtering (700 × 310 pixel). Then the image is filtered with a $3 \times 3$ binomial filter and subsampled by a factor of 2; this step is repeated three times which results in a final image of 88 × 39 pixels. The final image size is a trade-off between computational effort and precision.

4.2. Network

The desired foresighted obstacle avoidance behavior emerges from two criteria: The agent should not collide with obstacles, and its movements should cause minimal motor costs over

Fig. 6. Input representation for the inverse model: From the blob-list representation on the left side, the coarse-grained input image for the inverse models on the right side is obtained as described in the text. Black pixels correspond to 1, white pixel to 0.
a short period. The motor cost function defined below rewards forward movements. This is related to the perceptual task the agent has to solve: To distinguish between a corridor and a dead end requires mental movements that explore the given arrangement of obstacles; for example, just mentally rotating on the spot will not be helpful for understanding the behavioral meaning of the image. Moreover, the decisions of the inverse model should consider future costs over a short prediction horizon (in our case 7 steps). This is required to take “reasonable” decisions close to obstacles: If the agent sees nearby obstacles in front and on the left, a foresighted decision would be to turn right for some steps since afterwards the agent could continue with forward movements, whereas a left turn would subsequently require a large number of costly left turns before the agent could move forward again. Taking foresighted decisions in this way, the agent will exhibit wall-following behavior when it comes close to rows of obstacles. The same holds for recesses formed by some obstacles: Since the agent will have to perform a costly half-turn of approx. 180° once it has entered a recess, a foresighted agent should already turn off in some distance from the recess.

To produce this foresighted obstacle-avoidance behavior, the inverse model only has to consider few aspects of a visual scene, mostly nearby obstacles. We therefore expect that a relatively simple model structure should be sufficient to accomplish this task. We select linear models and a regression method as building blocks for the inverse model. Each linear model is trained with the univariate partial least squares method (PLS), a regression method widely used in chemometrics (Wold, 1975; Frank & Friedman, 1993; Garthwaite, 1994; Helland, 2001); PLS is meanwhile also applied as component of online learning methods in robot motor control (Vijayakumar et al., 2005). Univariate PLS is especially suited for problems where the number of training examples is small compared to the number of input variables. In situations with many input dimensions, there are often strong correlations between input variables which cause problems in ordinary least-squares regression. PLS produces a small number of components (usually much smaller than the number of input variables) that capture most of the information in the input variables which is useful to predict the output (Garthwaite, 1994). PLS then performs ordinary single-variable regression between each component and the output variable. The components are obtained by projecting the input data onto orthogonal directions in the input space. These directions are determined in an iterative procedure such that they have both high variance in input space and high correlation with the output variable (Hastie et al., 2001, p. 67).

The inverse model consists of two parts. The first part (called selector) comprises three regression modules which are used to select the most advisable action from the three candidate movements forward, left, and right. The second part of the inverse model is a single regression module employed as collision predictor: In situations where the selector suggests a forward movement, the collision predictor is invoked to check whether the movement might cause a collision. If this is the case, the next best suggestion of the selector (left or right turn) is used instead. The function of the collision predictor could of course also be performed by the tactile forward model, however we preferred to use a module that takes the same information as input as the modules of the selector.

The selector contains three regression modules referred to as forward-left, forward-right, and left-right. The first movement $m_1$ in each combination $m_1$-$m_2$ (with $m_1, m_2 \in \{\text{forward, left, right}\}$) is assigned the output value $q = 1$ in the regression, the second move-
Fig. 7. Linear approximation of goodness in the selector module, visualized for a simple two-dimensional data set. Left: Training data for the three movements forward (dark gray dots), left (light gray), and right (white) are distributed in the data space of the input image (in the example two-dimensional, in the real selector high-dimensional). Three linear functions are learned by assigning output values of 1 and 0 to the samples in each pair of movements. The 0.5 iso-lines of the three linear approximations are shown, the + and − signs indicate the direction of growing/falling output values. Right: The goodness of the left turn \( g(\text{left}) \) as obtained from equation (2). Two linear approximations are joined at the dashed ridge by a minimum operation.

For movement \( m_2 \) the output value \( q = 0 \). Regression prediction in each module produces an output value \( \hat{q}(m_1, m_2) \) that can be used to decide between the two movements. Given an image \( x \) (organized as vector), the average of the training images \( \bar{x}(m_1, m_2) \), and the regression coefficients \( \beta(m_1, m_2) \) for the movement combination \( m_1-m_2 \), the output value is obtained from

\[
\hat{q}(m_1, m_2) = 0.5 + \beta(m_1, m_2)^T \cdot [x - \bar{x}(m_1, m_2)].
\]

In the reverse direction, we set \( \hat{q}(m_2, m_1) = 1 - \hat{q}(m_1, m_2) \). Equation (1) describes a plane for each pair of movements. In Fig. 7 (left), the 0.5 iso-lines of the three planes left-right, forward-right, and forward-left are shown. We can use the plane equation to decide which of the movements \( m_1 \) and \( m_2 \) should be executed for a given image \( x \). Now, each of the three alternative actions is assigned a goodness \( g(m) \) from

\[
g(m_1) = \min_{m_2 \neq m_1} \hat{q}(m_1, m_2).
\]

This operation joins two linear functions by forming a ridge. While equation (1) decides between two movements, the goodness computed in equation (2) establishes a border between the movement \( m_1 \) and the two other movements. The movement suggested by the selector is the one with maximal goodness \( g \). Fig. 7 (right) visualizes the principle: The goodness of a left turn is obtained from equation (2) as \( g(\text{left}) = \min\{\hat{q}(\text{left}, \text{forward}), \hat{q}(\text{left}, \text{right})\} \), where we have to determine the first term from \( \hat{q}(\text{left}, \text{forward}) = 1 - \hat{q}(\text{forward, left}) \) since the linear approximation is only determined for the opposite direction (forward-left). If \( g(\text{left}) \) would be larger than both \( g(\text{right}) \) and \( g(\text{forward}) \), the selector would suggest a left turn.
The collision predictor is produced from a PLS regression applied to a non-collision set and a collision set of images, both obtained for forward movements. Here, collision images are associated with the output variable \( q = 1 \), non-collision images with \( q = 0 \). If the selector chooses a forward movement but the regression prediction in the collision predictor delivers a value above 0.3, the forward movement is revoked and replaced by the rotation with the best goodness.

Besides the selector and the collision predictor, the inverse model had to be complemented by an anti-oscillation module. In some cases, the inverse model suggests a turn in one direction in the given situation and a turn in the opposite direction in the next one; this leads to infinite oscillations. To prevent these, the anti-oscillation module replaces a rotation by a rotation in the opposite direction if two subsequent movements suggested by the selector are rotations in opposite direction.

4.3. Data collection

Training data for the selector modules in the inverse model are obtained from an internal simulation process using the visuo-tactile forward model. Short sequences of movements starting from the true image are simulated, the sequence with minimal costs is selected, and the first movement of the sequence is stored together with the true image. The total costs of a sequence contain collision and motor costs. A sequence with collisions can never be the one with minimal costs and is therefore discarded. The motor costs of a simulated sequence are determined from the sum of the costs of each movement in the sequence (forward 0, rotations 20), and the sum of the costs when two subsequent movements differ (10 when switching from translation to rotation or vice versa, 1000 when switching between rotations).

Even for a relatively short prediction horizon, the number of possible movement sequences is too large to perform an exhaustive search for the best. For 3 possible movements, a prediction horizon of only 7 steps would already require an internal simulation of \( 3^7 = 2187 \) different sequences of 7 steps which amounts to more than 15 000 applications of the visuo-tactile forward model. We reduce the effort by three different means. First, we restrict the search to sequences composed of maximally 3 subsequences. Each subsequence is a series of identical movements. For example, the sequence LLLFFRR (F = forward, L = left, R = right) with 3 subsequences would be included, but the sequence LFLLFRR (5 subsequences) would not. Second, we can sort all sequences in ascending order of their motor costs once in the beginning and test the sequences in this order. If a sequence turns out to be collision-free and the next sequence to be tested has higher precomputed motor costs, we can terminate the search. Third, predicted collisions immediately abort the test of a sequence. The first two measures usually keep the number of tested sequences below 100 even close to obstacles; in larger distance from the obstacles, often only a single sequence (7 forward movements) has to be tested.

The first movement of the best sequence found is stored together with the initial image of this sequence in the training set for the selector modules. If multiple sequences exhibit the same minimal cost value, a sequence is chosen randomly. Then the agent performs a real movement and initiates a new search. A random decision is taken whether the real movement
will coincide with the best movement found in the search (probability 0.7), or whether a randomly selected exploratory movement is inserted to increase the variety of the training data (probability 0.3).

Data for the collision predictor in the inverse model are collected in each real situation. The tactile forward model is invoked to find out whether a forward movement in the current situation would cause a collision, and the image of the situation is stored together with the collision prediction.

Fig. 8 visualizes the six training environments used for collecting data; in each environment, an example trajectory of 100 steps followed during data collection is shown. The upper three environments are “open,” the lower three “closed.” In each run, the agent is initially positioned at the center of the arena, facing in one of 8 different directions. It then performs 100 real movements as shown in Fig. 8. In each real situation, it searches for the best sequence of 7 steps length which continues from the real situation as described above. If, in an open environment, the real movement leaves the arena, the agent is turned towards the center.

4.4. Training

For the training of the selector modules, approx. 3,500 data points for forward movements and approx. 700 for each left and right turns were collected. For the regression to equally consider both classes (output value 0 and 1) in each combination of movements $m_1\cdot m_2$, the smaller sets were expanded to the size of the larger set by duplicating samples. The

![Fig. 8. Environments used to collect data for the training of the inverse model. In each environment, one example training trajectory (100 steps) is shown.](image)
same was done for the collision predictor where approx. 4,500 non-collisions but only 400 collisions were available. For each module of the inverse model (three selector modules and collision predictor module), 3 PLS components were determined (3 orthogonal directions in input space). The mean images $\bar{X}(m_1, m_2)$ and the coefficients $\beta(m_1, m_2)$ determined from the PLS components—as used in equation (1)—are visualized in Fig. 9 as images. The mean images $\bar{X}(m_1, m_2)$ show obstacles (white regions) on the right if a left turn is involved (forward-left, left-right) and obstacles on the left if a right turn is involved (forward-right, left-right); in the left-right module, obstacles are present on both sides in the mean image. The coefficients $\beta(m_1, m_2)$ found by PLS are plausible and can at least partly be interpreted from the images: The coefficients in the forward-left selector module contain a black region (negative values) close to the center. If an obstacle is present in this region, the module will favor left turns over forward movements. In the left-right module, obstacles on the right favor left turns (white region) and vice versa (black region). In the collision predictor, the coefficients exhibit a white (positive) region in the center; if objects are present there, the module will predict a collision. Other features in the coefficient images may be more difficult to interpret, though.

4.5. Test

To test the inverse model, the agent is placed into a closed environment different from the training environments; see Fig. 10. In each step, the agent follows the suggestions of the inverse model (selector, collision predictor, anti-oscillation module). Fig. 10 compares the behavior of two different inverse models. The model used in the left diagram was obtained by anticipating only 3 steps in the collection of the data, the model in the right diagram by anticipating 7 steps. Both models produce the expected obstacle-avoidance and wall-following behavior. The model with the larger prediction horizon of 7 steps (right diagram) keeps the agent farther away from walls and out of recesses (except for the top-left one), thus, apparently, the model has acquired a more foresighted policy. Following such a strategy in the internal simulation may help to restrict the simulated trajectories to the more promising ones with respect to the task of finding a straight route through (or a way out of) an arrangement of obstacles. In the

![Fig. 9. Modules of the inverse model (three selector modules and collision predictor module) obtained from PLS (prediction horizon in the data collection: 7 steps). The top row shows the average of the training data $\bar{X}(m_1, m_2)$ for each module; the correlation coefficients $\beta(m_1, m_2)$ are visualized in the bottom row; see equation (1). Positive values are encoded by white, negative values by black, zero by gray pixels.](image)
Fig. 10. Test of the inverse model in an unknown environment. The agent starts in the center of the arena (facing downward) and follows the suggestions of the inverse model for 500 steps. Left: Trajectories obtained from an inverse model trained for a prediction horizon of 3 steps. Right: Trajectories for a prediction horizon of 7 steps.

experiments described below, we always use the inverse model with a prediction horizon of 7 steps.

5. Internal simulation

The internal simulation process is an interplay of visuo-tactile forward model and inverse model as visualized in Fig. 11. The simulation starts with the true visual image $S(t)$. The inverse model takes this image and the last executed movement (in our experiments the last movement is always assumed to be a forward movement) and suggests a movement $M(t)$. This movement is not executed, but only drives the prediction process: From $S(t)$ and $M(t)$, the visual forward model predicts the next image $S(t+1)$, and the tactile forward model predicts the tactile (collision) signal $C(t+1)$. Based on the predicted image $S(t+1)$, the inverse model suggests the next movement $M(t+1)$, the forward models produce a visual and a tactile prediction and so on. Movements and tactile signals are included in an assessment of the quality of the sequence. In our experiments, the prediction step is repeated 60 times. In the rare cases where the inverse model suggests a movement that would lead to a collision according to the tactile forward model, the sequence is aborted. If the simulation of a sequence is complete, its motor commands are analyzed (motor assessment).

Since the inverse model suggests just a single movement in each step, only a single sensorimotor sequence is produced. If this sequence is not “desirable” according to some criterion, alternative sequences have to be simulated. Here we are using a simple random
strategy for varying the first simulated sequence, i.e., the one suggested by the inverse model. Each modified sequence is identical to the first sequence up to a randomly selected step within the first two thirds of the sequence. At that point, the sequence is modified by performing 3 subsequent rotations either to the left or to the right (approx. 45°). For all subsequent steps, the internal simulation again follows the suggestions of the inverse model. In our experiments, the first sequence is repeatedly modified in this way until a modified sequence is found that fulfills the criterion.

6. Object recognition by internal simulation

The perceptual task for our simulated agent is to decide between a corridor and a dead end when looking at an arrangement of obstacles. The result of the perceptual analysis could be a decision to either enter the obstacle arrangement if it is a corridor, or to turn around and look for an alternative route if it is a dead end (here we just report the decision and do not simulate the resulting actions). The motor decisions taken in the internal simulation are not executed but are just driving the prediction process; meanwhile, the agent is standing close to the entrance to the obstacle arrangement and is looking at the scene without showing any overt behavior. The image seen from this position is the initial sensory information $S(t)$ used in the simulation process.

According to our simulation theory of perception, an object (here an arrangement of obstacles) is not recognized by directly analyzing its visual features, but by predicting how the
agent will interact with the object. Such a prediction includes visuo-tactile information and movements, and all predicted sensory and motor information could in principle be considered for a behavioral decision or a perceptual judgment. However, as already mentioned in the introduction, it is likely that ultimately it will not be the predicted visual information which allows the agent to interpret the original image, but the tactile information or, in our example, even only the motor commands. In contrast to a dead end, a corridor is an arrangement of obstacles through which the agent can pass without too many turns and without changing the overall direction of movement too much: If $n_l$ is the number of left turns and $n_r$ the number of right turns taken in the simulation, we define a corridor as a sequence where $n_l + n_r < 20$ and $|n_l - n_r| < 10$. If no corridor sequence can be found among 20 different sequences, the arrangement is classified as dead end.

To test whether the perceptual decision can reliably be taken by the agent based on the internal simulation, we first designed 5 different dead end situations (top row of Fig. 12 and 14; object diameters ranging from 0.5 to 0.9 units) and then modified each arrangement in 6 different ways by deleting or moving obstacles such that it is turned into a corridor (column underneath each dead end situation in Figs. 12 and 14). Some objects are partially covered by others in the original image (Fig. 14), but all arrangements were designed in a way that the blob of a partially covered object could still be determined correctly in the visual preprocessing; this guarantees that the arrangement is fully recognizable from this image. Since the prediction sticks to the blob-list representation and never reconstructs an image, predicted blobs can overlap without interference.

Since the panoramic view is unfamiliar, deciding between dead ends and corridors by looking at the images in Fig. 14 is not easy for a human observer (and it would be even more difficult if the image would be distorted in some way). The agent, however, “understands” the behavioral meaning of all situations by revealing their “affordances” through a process of internal simulation: All situations are correctly classified as dead ends or corridors. The maximal 20 simulation sequences are visualized as trajectories in Fig. 12. For the dead ends (top row), 20 sequences are simulated, but none of the sequences is found to fulfill the “corridor criterion.” For some corridors, already the first simulated sequence fulfills the corridor criterion so that the simulation can be terminated. In other, more complex situations, several modified sequences have to be simulated before one fulfilling the corridor criterion can be found. In all corridor cases, one trajectory is visible that leaves the arrangement of obstacles. It is clear, though, that the random search could also have produced 20 sequences which all would be missing the “exit” from the corridor arrangement; in this case, the corridor would have been misclassified as dead end.

It is apparent that the simulated trajectories are always embedded in the obstacle arrangement, thus the forward model proves to be sufficiently precise to predict the changing visual situation over 60 steps (see section 5). Only some trajectories would cause collisions if the agent would really execute the movements of the simulated sequence; these collisions were not predicted due to accumulating errors in the visual long-term prediction; see collision situations marked with small dots in Fig. 13. Accumulating errors may also result in false estimates on the width of gaps between obstacles, thus it is possible that arrangements are misclassified because, in the simulation, non-existing gaps are traversed in dead ends or existing gaps missed in corridors.
Fig. 12. Object recognition by internal simulation. Top view of the 30 test environments (top row: dead ends; column below each dead end: corridors obtained from modifying the dead end) and simulated trajectories starting from the true position of the agent (circle with line). Small dots mark agent positions where the simulated movement would actually have caused a collision.
Fig. 13. Enlarged view of two of the simulation trajectories in Fig. 12 (left: dead end in fourth column, first row; right: corridor in second column, fifth row).

7. Discussion

7.1. Bootstrapping cognition from behavior

7.1.1. Bootstrapping in our approach

The term “bootstrapping” allegedly goes back to the tale about Baron Münchhausen, who claimed to have pulled himself out of a swamp by his own hair or, in a different version, by his own bootstraps (Wikipedia). Generally, the expression refers to a process where complex competences emerge from some initially more or less blank state. We assume the term is used by Dennett (1998) in a similar sense when he addresses the question how the brain can learn to assign meaning to sensory data (in our case visual information) if the meaning is not given from the beginning. Our answer to Dennett’s question—which we tried to substantiate by the computerized thought experiment presented in this paper—is that cognitive or perceptual competences can ultimately emerge from an evaluation system that assesses signals which have immediate behavioral relevance; these signals can be tactile sensory impressions or information on the executed motor commands. What is required to carry over the immediate “meaning” of these signals to the visual sense is a process that predicts the sensory consequences of actions in a given visual situation, among them those with predefined value. Thus, in addition to the existence of an evaluation system, we have to assume that the agent can learn to predict the sensory consequences of its own actions.

Given the value system and the forward model, the agent is capable of characterizing the visual scene by a number of simulated sequences. These sequences will reveal some affordances, but other, behaviorally important affordances may not be found in the huge search space of all possible sensorimotor sequences. The search space has to be restricted in a way that is related to the task at hand (an attempt to understand the shape of a graspable object will require an internal simulation of hand movements towards the object whereas simulating
leg movements will not be helpful). We therefore suggest that the bootstrapping process is a staged procedure: *Long-term predictions* (in our case required for distinguishing between dead ends and corridors) are based on simulated decisions taken by means of *short-term predictions* (in our case, the decisions should produce foresighted obstacle-avoidance behavior).

Decisions on the short-term level could in principle be obtained in each situation by internally simulating a number of sensorimotor sequences with short-term prediction horizon and executing the commands of the best sequence found. However, a more efficient way is to train an inverse model with the decisions derived from the short-term simulation process. The inverse model should at least be able to suggest an appropriate action in typical and relatively simple situations. Once the inverse model is trained, it will guide the long-term prediction process. The value systems used to assess the short-term and long-term predictions are both based on tactile and motor information but are somewhat different from each other in our case: In the short term, the agent should move straight without collisions (see section 4.3), while in the long term, it should avoid getting caught in dead ends as expressed by the “corridor criterion” (see section 6). It is clear, though, that the short-term criterion which is used to shape the inverse model has to be closely related to the perceptual or behavioral decision taken by the long-term simulation. In the task at hand, both allow the agent to cover a large spatial distance in a given number of steps.

The cognitive ability to distinguish between corridors and dead ends emerges without the need for complex sensory processing: Only elementary “blob” features are extracted from the image, but there is no instance where these low-level features are combined to detect higher-level features like “walls” or even entire “corridor” prototypes. We expect that it would
indeed be difficult to train an adaptive model to classify images as the ones shown in Fig. 14 as dead ends or corridors.

To summarize, our experiments show that cognitive competences can be “bootstrapped” from low-level visual mechanisms, a simple value system, and adaptive structures capable of predicting the sensory consequences of actions. To focus the internal simulation on actions related to the task at hand, knowledge can be transferred from the forward models and the value system into an inverse model.

7.1.2. Bootstrapping in developmental robotics

The idea that cognitive competences can be bootstrapped from behavior is a tenet of an emerging research field which is known under different labels as “embodied artificial intelligence,” “embodied cognition,” “developmental robotics,” or “epigenetic robotics” (reviews: Lungarella et al., 2003; Pfeifer and Iida, 2004). Our own research could certainly be subsumed under one of these labels, since it rests on the assumption that “brain, body and environment are reciprocally coupled, and that cognitive processes arise from having a body with specific perceptual and motor capabilities” (Lungarella et al., 2003). However, when it comes to the question of cognitive competences, most other approaches assume that actions are directly involved in the cognitive processes. Pfeifer and Iida (2004) call this the “principle of sensory-motor coordination” or the “principle of information self-structuring” and write: “Through the—physical—interaction with the environment, the agent induces or generates sensory stimulation [...]. The thus generated stimulation will typically be more structured, and will contain correlations within and between sensory channels that greatly facilitate the problem of focusing on the relevant stimulation and is in fact the enabler for learning.” By contrast, in our approach, as pointed out in section 1, the cognitive process itself is completely passive in the sense that no behavior is overtly executed.

Despite the fact that overt behavior is assumed to be part of the perceptual process in these approaches, we see some interesting correspondences to our work insofar as the categorization problem is transferred from the sensory domain to the motor domain. Pfeifer and Scheier (1997), for example, provide a mobile robot with a “circling behavior” which allows it to distinguish between cylinders of different size just by observing the angular velocity of its movements. This is very similar to our motor-based “corridor criterion” (see section 7.2.1), with the difference that in our case the behavior is only simulated within the agent’s “nervous system.” Beer (2003) describes a simulated agent equipped with an array of distance sensors which is able to discriminate between falling circles and diamonds (based on an evolved control system). Also in this task the active scanning movements of the agent are inextricably involved in the perceptual process (which would otherwise not even be possible given the coarse resolution of the sensor). Moreover, the perceptual decision is directly reflected in the behavior of the agent (the agent catches circles and avoids diamonds). Beer (2003) observes that “since we conceptualize the object discrimination task in terms of a ‘decision’ to ‘approach’ ‘circles’ and ‘avoid’ ‘diamonds’ [...], we expect to find representations of such things within the agent, as well as processes that manipulate and transform these representations. [...] We have found no evidence for circle detectors or corner detectors within this agent’s evolved nervous system [...].” The same is true for our agent: There is no trace of a sensor-based detector for corridors or dead ends, but just a motor-based “corridor criterion” (although this is explicitly given...
whereas in Beer’s evolved controller, decisions are taken in a complex interplay of sensing and acting).

In the following, we will focus on the shared idea of bootstrapping cognition from a minimal set of premises, where newly acquired competences and structures are resting on already mastered competences and existing structures. Approaches from developmental robotics typically start with a small number of built-in mechanisms (innate behaviors, reflexes, drives, learning mechanisms) and then let the agent “discover” relations between actions and sensations by interacting with the world; these relations are in turn used to supercede the built-in mechanisms and thus reach a first level of cognitive competences (Blank et al., 2005). Letting the robot discover sensorimotor relations in a learning process rather than having a human designer providing precast representations avoids an “anthropomorphic bias” (Blank et al., 2005) and allows the robot to adapt to changing conditions (Olsson et al., 2006). In the extreme case, not only relations between sensory state and actions are learned, but also a model of the sensory apparatus. An example is the work by Olsson et al. (2006): The informational relationships between sensors (which are reflected in their physical arrangement) are learned and expressed in “sensoritopic maps.” These maps can then be used to learn sensorimotor models by some optical flow method.

Most approaches assume some hierarchy of levels where higher levels rest on the representations provided by lower levels. An incremental learning sequence proceeding from lower to higher level seems to be more viable than attempting to directly learn complex relationships or concepts (Kuipers et al., 2006). In a sequence of learning stages, it should be possible to bridge the wide gap between “the blooming buzzing confusion” on the level of raw sensory and motor signals (Kuipers et al., 2006, citing William James) and “useful higher-level representations for space, time, objects, actions, etc.” (Kuipers et al., 2006). In the work by Blank et al. (2005), for example, higher levels are supposed to take abstracted, “chunked” representations from lower levels as input, and form representations of sensorimotor sequences from subsequences represented in the previous level. For a single level of this hierarchy, Blank et al. (2005) could demonstrate that the compact representations derived from the sensorimotor exploration are sufficient to guide a robot to a goal location along a given path. Kuipers et al. (2006) go in a similar direction also in the context of a navigation task by learning sensory state abstractions together with high-level, temporally extended actions. Reducing the number of actions (the “task diameter”) on a higher representational level allows them to reduce the effort for learning action policies (see also Provost et al., 2006).

It is apparent that these ideas from developmental robotics—grounding cognitive competences in built-in mechanisms, superceding built-in mechanisms by learned sensorimotor relationships, and staged learning procedures—are also characteristics of our approach. Let us focus on hierarchies which are present or conceivable in three forms for our approach. First, we notice a hierarchy with respect to the length of the prediction horizon—short-term predictions are used for training the inverse model, and long-term predictions are based on the trained inverse model. Second, as we will discuss in section 7.2.1, also a hierarchy of inverse models is conceivable—inverse models for foresighted obstacle avoidance, giving rise to inverse models for traversing gaps—, although this hierarchy may be limited by the fundamental assumption of our approach that complex sensory representations have to be avoided. Third, we assume that, when this approach should be applied to real-world problems,
a hierarchy of forward models would have to be formed, with higher-level forward models operating on more abstract sensory representations and on a coarser time scale; presently, our model contains only a single level of forward models.

7.2. Perceptual competences

7.2.1. Perception through anticipation

Interpreting visual information is hard. There is still no technical system that could recognize arbitrary objects as instances of some class, for example an arbitrary chair it has never seen before, and even recognizing a known object from different view points turns out to be a difficult problem (but see section 7.2.2). Our simple computerized thought experiment is of course still miles away from accomplishing such feats, but we think our approach offers a new perspective on the problem of invariant recognition. A chair may actually not directly be recognized by the sum of its visual features, but from its behavioral meaning that can be derived from the visual features by means of an internal, sensorimotor simulation. The internal simulation unravels the chair’s “affordances,” its behavioral meaning, for example the affordance of being “sit-on-able.” Our simulated agent can distinguish between corridors and dead ends because the internal simulation process reveals that some arrangements are “passable” while others are not. An external observer would attribute perceptual abilities to the agent, although no perceptual processing in the usual sense is taking place. There is no sensory classifier module for dead ends and corridors, but only a “corridor criterion” which is related to the motor commands executed in the internal simulation.

Depending on the complexity of the situation, the internal simulation can be very fast: In some corridor arrangements, the actions suggested by the inverse model are “mentally” guiding the agent out of the obstacle array, thus it can be classified as a corridor after simulating only a single sequence. Generally we assume that better suggestions produced by the inverse model will reduce the number of simulated sequences required to unravel certain affordances. There appears to be some trade-off between the complexity of the inverse model and the temporal effort for the simulation. For example, it would reduce the simulation effort if the inverse model would turn the agent towards larger gaps in the walls it is just passing (see Fig. 12: 6th row, 2nd and 4th corridor; 7th row, 3rd and 4th corridor). This modification would already transform the inverse model into a very simple “corridor classifier” which is able to recognize, at least from a short distance, that an escape is possible in a certain direction. In an extreme case, a very complex inverse model could take the decision whether to enter an obstacle array or not without any simulation steps; such an inverse model would actually be a sensory classifier. However, this classifier would have to accomplish invariant recognition of arbitrary dead ends and corridors based on the sensory information alone which is probably a non-trivial task.

In our approach, specific perceptual competences seem to be related to the inverse model and to the evaluation criterion; the visuo-tactile forward model, in contrast, can be involved in a wide variety of perceptual tasks since it does not contain any task-specific components. A very simple perceptual task would be to estimate the distance to an obstacle by simulating how many translatory steps can be executed before the agent collides with the obstacle; a sequence of translations would be simulated and the collision-free steps would be counted. Hoffmann &
Möller (2004) use a visuo-motor forward model to decide whether the agent is located in the center of an array of obstacles; there, a sequence of rotations is simulated and the distance to the objects is analyzed. Also spatial relations between objects like “object A is located behind object B” could be revealed by applying the forward model together with mechanisms for obstacle avoidance and target approach: If, in the internal simulation, the agent has to take a detour to reach the target object, the object is standing behind another. In all these applications, the same forward model could be applied, but inverse model and evaluation criteria will differ. Rephrased in the context of Gibson’s ideas, an internal simulation guided by a specific inverse model and a specific value system is able to reveal specific affordances. Without this guidance, the simulation will reveal a multitude of different affordances, but these are not necessarily useful for the agent.

Note that the perceptual competences depend on the senses, on the motor abilities, and on the body shape of the agent. The property of being a dead end or a corridor depends on the diameter of the agent’s body: With a smaller diameter, some dead ends would become corridors, with a larger diameter, some corridors would turn into dead ends. An agent that cannot turn on the spot may interpret some corridors (in the judgment of our agent) as dead ends since it cannot traverse tight corners. Fajen and Turvey (2003) use the terms “body-scaled affordances” (such as passages) and “action-scaled affordances” (in our example, the action being the ability to turn on the spot). Perceptual competences which are unrelated to the abilities of the agent cannot be attained at this basic level: Our agent would not be able to find out whether a cylinder is “graspable” since it lacks the ability to grasp and can therefore not acquire the internal models needed for this task.

7.2.2. Other approaches to function recognition

The level of cognitive competences we attempt to address with our approach can best be described by the term function recognition. It is fundamentally different from most approaches to visual object understanding—a term encompassing recognition, identification, and categorization (review: Palmeri & Gauthier, 2004)—which are based on purely sensory processing, regardless of the details of the presumed processing mechanisms (image-based vs. structural descriptions, exemplar- vs. rule-based categorization, etc.). Typically, these approaches accomplish recognition by a hierarchy of sensory processing stages from the retinal image to object-tuned neural units (see, e.g., Riesenhuber & Poggio, 2002); one could use the term appearance recognition for these approaches. Invariance—both in the sense of being tolerant against changes in perspective and illumination, and of being able to assign differently shaped objects to the same category—has to be accomplished at some stage of this hierarchy; we think this is the crucial and still unsolved problem of vision.

Along with Norman (2002) we conjecture that the mechanisms of function categorization we are suggesting have parallels in the function of the “dorsal” system, while the above-mentioned appearance-based approaches are related to the “ventral” pathway and may therefore rest on completely different mechanisms. Nonetheless, the ability to recognize the function of objects that we ultimately hope to achieve with our approach—e.g., recognizing a chair in its function—is strongly overlapping with or even surpassing the abilities of the purely sensor-based approaches: An object is recognized as member of a (function) class regardless of whether the same object has been seen before; however, the result of function recognition
will usually be to take a behavioral decision rather than to assign a label or symbol to the object.

Some approaches to function recognition from computer vision exhibit similarities to our approach (although publications in this field appear to have thinned out in the last few years). An example is the work by Stark and Bowyer and colleagues who attempt to “define an object category in terms of the function properties shared by all objects in the category” (Stark and Bowyer, 1991) rather than by a direct recognition process based on exact geometrical models of objects. Following a very similar argumentation to ours, Stark et al. (1996) point out that to achieve similar capabilities as a function-based classification, a direct classification of objects based on geometrical models would need to “introduce explicit allowed ranges of variation in geometry or structure. However, it can be extremely difficult to anticipate and specify all elements of parametrization needed to make the class of shapes represented by a parametrized prototype equivalent to that captured in a human category concept.”

Stark and Bowyer (1991) categorize objects given in form of a geometrical model (list of faces and vertices) as chairs of different types or as non-chairs. Each class of chairs is described by a functional description comprising a set of function labels (e.g., “provides sittable surface”) which in turn are based on one of a number of procedural knowledge primitives (e.g., “relative orientation” or “stability”); these primitives can be extracted from the geometrical model. They apparently succeeded to correctly recognize chairs with very different visual appearance, although in practice it may be difficult to obtain a clear geometrical description of objects. In later work (Stark et al., 1996; Sutton et al., 1998), geometrical descriptions (so-called “object plus unseen space models”) were obtained from laser range finder data, and action plans to confirm the object’s function could be derived from the inferred functionality. Characteristic for this approach is that the functional reasoning is entirely based on the geometrical description, thus possible actions of the observer are only implicitly encoded in the functional description. This is different from the work by Bogoni and Bajcsy (1995) and Bogoni (1998a,b) who present a system which learns to associate the visual appearance of a tool-like object with its suitability for piercing and chopping. The function is inferred from chopping and piercing interactions of the tools with different materials.

Since our approach is presently far from the ability to recognize chairs or tools it is difficult to judge to which extent a practical application would have to incorporate elements from the above-mentioned systems for function recognition; this especially concerns geometrical descriptions of objects (which, from our perspective, re-introduces aspects of appearance recognition into approaches to function recognition). Hallmark of our approach is the attempt to keep the object representation at the lowest possible level, and transfer the burden to infer the object’s function to the sensorimotor anticipation process. Thus, in our approach, action plans are not derived from the result of the function recognition, but are part of the recognition process. Whether these assumptions will at some point allow to surpass the abilities of the above-mentioned approaches has to be seen.

7.3. Computerized thought experiment: Assumptions and insights

In our simulation, the problem of “function recognition” (see section 7.2.2) is obviously simplified and idealized in many ways; in what follows we attempt to motivate and justify the
underlying assumptions and the decisions taken in the design of our computerized thought experiment.

With respect to the overall perceptual and behavioral task, our goal was to “animate” our original thought experiment (Möller, 1999). This thought experiment should be easy to envision for the reader, but is also of a manageable complexity that allowed us to derive a computerized version. As Beer (2003) points out, models in such simple domains primarily serve for “conceptual clarification,” and Kuipers et al. (2006) see their own, closely related work as “a gedankenexperiment or ‘intuition pump’ to help us develop useful insights at this early stage of our research enterprise.” It would certainly be much more convincing if we could demonstrate the feasibility of our approach for more demanding (and familiar) tasks like the recognition of furniture or dishes. This, however, is not yet in reach since in our approach it would require to solve numerous problems on the way, like movement control (locomotion, reaching, grasping), tactile processing, prediction of image data, robust learning mechanisms capable of dealing with high-dimensional data, etc. We also feel that moving from the prototypical “chairs” example to the simpler “dead end” task might help to look at the problem of perception from a completely novel perspective (which is what we intend to do with our approach), since it is somewhat removed from the traditional concept of recognizing “objects.” Moreover, we assume that there will be a large increase in complexity when moving from anticipation models dealing with locomotion (as in our example) to models dealing with manipulation (e.g., dishes and chairs). For many of the problems that would have to be solved for anticipation in the context of manipulation we simply cannot present solutions at the present stage. For example, the prediction of the visual state would have to consider the structure of objects (i.e., physical connections: moving the handle of a hammer will also move its head) and the physics of interaction (predicting the consequences of a missing leg in the process of sitting down on a chair).

The next set of assumptions concerns the agent's view of the world. The agent is equipped with a panoramic camera, and the obstacle arrangements are chosen such that the bottom center points of all objects are seen, thus the agent has full information on the visual scene. Thus, in a certain sense, the agent has a complete (external) model of the object-to-recognize at its disposal; nonetheless, the components of this model are not linked to each other as in the visual object representation of the appearance-based recognition models described in section 7.2.2 (first paragraph). Partial views and occlusions would require to add mechanisms for active perception and visual short-term memory. Although it might be possible to incorporate such mechanisms into our simulation with reasonable effort, it is not clear in which respect they would contribute to the conceptual statements we were aiming at. Another simplification is that the vertical position of the obstacles in the image is related to their distance from the agent (because the camera is mounted above the ground plane), thus no mechanism for depth perception (binocular stereo, motion stereo) is required (however, a plethora of such methods is available in computer vision, so this assumption could be lifted for other application domains). All objects are colored differently and can clearly be distinguished from the background to simplify the segmentation process (and, as pointed out above, we thus manage to get around the problem of segmentation that will become even more severe when moving from locomotion to manipulation; but these problems are also affecting the classical, appearance-based vision approaches). The visual preprocessing and the type of
features (blobs) extracted are extremely simple and will definitely have to be replaced by more complex (maybe even shape-based) descriptions of objects (see section 7.2.2); we feel that discussing the details of these representations would be too speculative at the present stage of our work.

The motor abilities of the agents were chosen from pragmatic motives. The size of the rotational and translational movements is small enough to establish links between the same object in the image before and after the movement, even if we would use training environments with multiple objects, and large enough to keep the number of steps in the internal simulation small. Note that also the agent’s size and the size of the environment were chosen in relation to the motor step size (or vice versa). The decision to use a discrete set of movements allowed us to split the forward model into separate parts which proved to be easier to train than with a continuous set of movements as input signal (see next paragraph).

The hardest practical problem in our approach is to achieve the required precision of the forward model in order to obtain a sufficiently long prediction horizon. Over the past few years, we have been investing a large portion of our work into this endeavor (but we are not aware of other work going in the same direction). In our first attempts, we tried to directly predict pixel values of the next image from image regions in the previous image. This appears to produce reasonably good results in restricted domains, especially for uniformly colored, blob-like landmarks, although even then additional effort has to be invested to project the predicted images back onto the manifold of training images after each prediction step (Hoffmann, 2006). For other domains, e.g., for visual forward models for a robotic camera head, this strategy produced unsatisfying results (Große, 2005). We recently managed to train high-precision forward models in the context of visual saccades (Schenck & Möller, 2007) by turning towards a different form of representation: Instead of trying to predict pixel values, the forward model learns the coordinates of the source pixel in the previous image and assigns its pixel value to the destination pixel in the next image. The forward model used in this work is based on the same idea: changes in the coordinates and the size are predicted rather than image pixels. All decisions on the encoding of inputs (additional cos/sin input signals, splitting of the image into an upper and lower part) and outputs (changes instead of absolute values) and on the overall architecture of the network (splitting the network into separate parts for movements, image part, and output variable) were taken in a long period of experimentation just to produce a forward model with sufficiently high precision. The same holds for the network type (multilayer perceptron), network size (number and size of layers), learning methods (vanilla backpropagation) and learning parameters (learning rates, number of epochs). Our crude method for balancing collisions and non-collisions in the training set of the tactile forward model (section 3.4) could be replaced by more advanced methods to eliminate the intervention of the designer (see, e.g., Blank et al., 2005). Imbalanced data sets are a common problem in learning; we just opted for the simplest solution. To conclude this paragraph: The success of the object recognition experiment shows that forward models with sufficient prediction accuracy can in principle be learned, and from our other work we are optimistic that this is also possible in more realistic domains.

The training environment for the forward model is characterized by two properties: Objects of different diameters are presented, and the training environment contains only a single object. The first property turned out to be crucial for the training of the forward model which also
predicts the apparent size of the objects. Training the network with only a single diameter of
objects produced large errors in the prediction even when tested with objects of exactly the
same diameter, probably because small deviations in the previous prediction step brought the
network in the next step into a region of the input space where no training data were available.
The decision to use only a single object was already motivated in section 3.3, but we expect
no serious degradation in performance if this assumption would be lifted.

Also the architecture of the inverse model was selected after some experimentation with
different network architectures. We decided to use a regression method since the encoding of
inputs and outputs was simpler than in other network types (e.g., vector quantization methods).
PLS appeared to be the most suitable regression method for this problem; we provided some
arguments in section 4.2. We already motivated the necessity of the inverse model in section
7.1.1: Deriving the decision from an inverse model is much more efficient than obtaining
it from an internal simulation process; we also think that this is a requirement of a staged
learning approach (see section 7.1.2). An efficiency argument also applies to the question why
the training data for the inverse model are derived from internal simulations rather than from
real actions in the world: Once the forward model is available, it is much cheaper for the
agent to simulate several dozens of alternative behavioral sequences than the actually execute
them. Moreover, it may seem to be a waste to train the inverse model only with the initial
move of a sequence, although an entire optimal plan for the next few steps is available from
the search process (see section 4.3). Ideally, the inverse model should in each step take a
decision that is identical to a previously formed plan (except maybe in the light of new data),
but the necessity of an anti-oscillation module (section 4.2) proves that this is not the case.
A systematic analysis of inverse models suggesting single moves in comparison to inverse
models suggesting entire plans would be necessary to resolve the different aspects of this
question (quality of prediction, effects of conflicting data in the training sets etc.). To reveal
different affordances, different plans would have to be executed; in our system, these plans
would correspond to different inverse models. For example, the intention to analyze an object
with respect to the affordance of being “sit-on-able” would activate a different inverse model
than the affordance of being “graspable.” Since our agent only analyzes a single affordance
(“passable” or not), it only needs a single inverse model. Finally, the different representations
used for the forward model (blob lists) and the inverse model (images) lead to a duplication
of the collision detector; however, we found it conceptually more consistent that all parts of
the inverse model take the same representation as input (images; see section 4.2). The specific
cost function associated with the inverse model (section 4.3) is certainly one of many different
possible solutions: It only has to guarantee that the agent ultimately produces foresighted
obstacle-avoidance behavior since this behavior is related to the affordance that should be
revealed. Similar arguments hold for other properties like the exploration probability.

Although we had to invest some effort into finding the appropriate architecture and learning
methods for both forward and inverse model, we do not consider the specific solutions
(network type, encoding, etc.) as an important contribution of this paper; without doubt,
more sophisticated architectures and learning methods do exist (recurrent neural networks
could, for example, be a solution to handle occlusions in image-like representations of the
visual scene).
The interplay of forward and inverse models in the *internal simulation* is based on the overall system architecture shown in Fig. 11, regardless to which domain our approach is applied. Many design decisions, however, are again specific for the thought experiment of dead end recognition, and some are just the simplest solutions that came to mind. The latter is for example the case for the decision to abort failed sequences instead of backtracking. Backtracking would have required a complex data structure (or a recursive implementation); our method only needs some random manipulation of the decisions. It is obvious that backtracking would be more efficient, but our re-iteration method did the job just good enough. Parameters, like the number of maximally 20 sequences, were selected from the observation of the decision process: 20 was enough to reliably find gaps in the obstacle arrangement (see Fig. 12). The most critical question in this context may be what would be the origin of the “corridor criterion” if it would not be specified by the designer, since this criterion is the keystone of the entire perceptual process that reveals the affordance. We can only speculate that this criterion may be part of a general cost function that would be the result of some evolutionary process: avoiding unnecessary detours will be beneficial for a mobile agent.

Generally we feel that the multitude of underlying assumptions and design decisions are not a special feature of only our approach, but are affecting all “computerized thought experiments.” Depending on the time scale of the processes that are addressed, the designer has to take certain commitments: Either the architecture of the system has to be designed, or initial conditions, learning methods, and developmental processes, or evolutionary algorithms and morphogenetic processes (see Table 1 in Pfeifer and Iida, 2004). Dealing with all levels at the same time and growing a solution “from scratch” is simply not feasible.

We already mentioned that numerous additional mechanisms, many of them in the sensorimotor domain, would be required to proceed from our simple example to more realistic applications. Since our approach strives to keep purely sensory representations on the lowest possible complexity, its explanatory value will stand and fall with the type of sensory processing required for tasks like furniture recognition. Complete object descriptions as in the function recognition approaches described in section 7.2.2 should be avoided since this would introduce features of appearance-based approaches which we attempt to replace by our model. Nevertheless, more abstract representations of spatial (“Gestalt”) or temporal chunks will have to be learned in a staged process as suggested by the bootstrapping approaches from developmental robotics (see section 7.1.2). Active vision could become involved for visual segmentation, and selective attention mechanisms may be required to sequentially analyze the physical structure of an object (e.g., the connections between seat and legs of a chair). The problem will further be complicated by noisy sensor data, restricted view, and uncertain effects of motor commands. Nonetheless, these prospective difficulties should not be used as an argument to throw away the entire approach—classical approaches to computer vision have been striving to solve the same problem for decades, so far with only partial success.

By turning our thought experiment into a computer simulation we could demonstrate that the internal models required in our approach can actually be learned by the agent and successfully applied to solve a simple perceptual task. The essential building blocks (forward model, inverse model, value system) were already present in our thought experiment
(Möller, 1999), but working on the simulation brought about several novel ideas, specifically the method for training the inverse model based on internal simulations, the methods for restricting the search space in the short-term predictions (although this might be task-specific), and the idea that the perceptual ability actually rests on some motor criterion rather than a sensory criterion. Our future work will try to scale up this approach to more realistic domains; in a recent project, we already moved on to simple manipulation tasks (block pushing; Sinder, 2006).

8. Conclusions

Our computerized thought experiment shows that visual perception can be bootstrapped from a simple non-visual value system, low-level visual feature extraction, and the internal simulation of action effects. It provides support for our theory that affordances are revealed by an internal simulation of action sequences and their sensory effects. Our present results are a “proof of principle.” In our future work we hope to demonstrate that robotic agents are able to interpret real-world visual information based on this approach; a successful application in autonomous robots would also lend credibility to our theory as a possible explanation for the cognitive abilities of biological agents.

Notes

1. Note that this knowledge is not restricted to the training environment, but can be applied to arbitrary visual situations (of course only within our model’s domain): The forward model is not a model of a specific environment—as it could, for example, be used for navigation purposes—but a description of how aspects of an arbitrary visual scene are changing under movements.

2. The agent is not equipped with neural mechanisms which one would usually associate with cognitive competences (e.g., neural networks for the categorization of sensory data), but its cognitive abilities are based on internal sensorimotor models (forward and inverse models): It categorizes images of obstacle arrangements as dead ends or corridors in a process of sensorimotor simulation. Although this cognitive ability is based on sensorimotor models, it does not require motor activity.

3. It is, however, possible that such competences could be acquired at some higher level of mechanisms, for example by observing actions of other agents.

4. Although some approaches stress the importance of sensorimotor processes in learning to recognize objects, they still adhere to the concept of purely sensor-based processing stages (Almássy et al., 1998; Almássy & Sporns, 2001).
Acknowledgments

We are grateful to three referees for their thorough reviews. Many arguments presented in section 7, including numerous references, were incorporated based on their helpful suggestions.

References


