Perception through Visuomotor Anticipation in a Mobile Robot ⋆,⋆⋆

Heiko Hoffmann [1]

Cognitive Robotics, Max Planck Institute for Human Cognitive and Brain Sciences, Amalienstr. 33, 80799 Munich, Germany

Abstract

Several scientists suggested that certain perceptual qualities are based on sensorimotor anticipation: for example, the softness of a sponge is perceived by anticipating the sensations resulting from a grasping movement. For the perception of spatial arrangements, this article demonstrates that this concept can be realized in a mobile robot. The robot first learned to predict how its visual input changes under movement commands. With this ability, two perceptual tasks could be solved: judging the distance to an obstacle in front by 'mentally' simulating a movement toward the obstacle, and recognizing a dead end by simulating either an obstacle-avoidance algorithm or a recursive search for an exit. A simulated movement contained a series of prediction steps. In each step, a multi-layer perceptron anticipated the next image, which, however, became increasingly noisy. To denoise an image, it was split into patches, and each patch was projected onto a manifold obtained by modeling the density of the distribution of training patches with a mixture of Gaussian functions.

Key words: sensorimotor anticipation, vision, forward model, mobile robot, perception, multi-layer perceptron, image denoising, Gaussian mixture model

⋆ I am grateful to Ralf Möller for introducing me to the area of perception through sensorimotor anticipation, for numerous discussions, and for comments on the manuscript. Bruno Lara, Helmut Radrich, Karl-Heinz Honsberg, and Fiorello Banci helped setting up the robot. Furthermore, I thank the two anonymous reviewers, whose comments helped to improve the manuscript.

⋆⋆Running title: Visuomotor Anticipation

Email address: Heiko.Hoffmann[at]ed.ac.uk (Heiko Hoffmann).

[1] Present address: IPAB, School of Informatics, University of Edinburgh, EH9 3JZ, UK

submitted to Neural Networks 1 Feb 2006, revised 14 Jun 2006, accepted 10 Jul 2006

1 Introduction

Mammals learn to see by actively exploring their environment (Held and Hein, 1963; Gregory, 1998). Actions like crawling, walking, turning, poking, and grasping cause specific changes in the sensory signals of the actor, and these causal relationships can be learned. Several authors suggested that these learned relationships are the basis of certain perceptual qualities, like understanding the spatial arrangement of obstacles, the function of tools, and the material properties of objects (Möller, 1996, 1999; Gross et al., 1999; O'Regan and Noë, 2001; Grush, 2004; Hoffmann and Möller, 2004).

Psychological experiments indeed show that active movement alters or interferes with perception (Held and Freedman, 1963; Prinz, 1997; Rossetti et al., 1998), but psychology does not give an insight into the working machinery. This gap may be filled by robot experiments, which offer the possibility to study the interplay of action and perception since both the behavior and the 'neural processing' can be controlled experimentally (Webb, 2001; Verschure et al., 2003).

Robot experiments already demonstrated that action can lead to object recognition: objects could be separated from background if poked by a robotic arm (Metta and Fitzpatrick, 2003), and a mobile robot detected the size of objects by circling around them (Pfeifer and Scheier, 1999, p. 407ff.). In these studies, however, the robots need to be active during the perceptual task itself; humans can perceive passively. A solution to this direct dependence on action may be sensorimotor anticipation. A so-called 'forward model' predicts the sensory effects caused by motor commands (Wolpert et al., 1995). Thus, overt motor commands can be replaced by covert ones (Hesslow, 2002). Such a sensorimotor anticipation was successfully applied to reinforcement learning (Sutton, 1992; Schaal, 1997) and to mobile-robot navigation (Tani, 1996; Tani and Nolfi, 1999; Gross et al., 1999; Ziemke et al., 2005).

The work presented here combines these two approaches: action-based object recognition and sensorimotor anticipation. A mobile robot with omnidirectional vision 'mentally' explored certain movement plans and tested their effect on a single image from the robot's camera. Specifically, by simulating a forward movement, the robot judged the distance to an obstacle in front. Furthermore, the robot decided if it faced a dead end or a passage by simulating two alternative movement strategies: an obstacle-avoidance algorithm and a recursive search for an exit. This work thus complements a thought experiment in which Möller (1999) suggested that a dead end can be recognized by anticipating the result of a simulated recursive search.

Perceptual judgment based on sensorimotor simulation has two main advan-

tages. First, perception is directly linked to the observer's body size and behavior. Distance, for example, is not a metrical measurement, but understood as the time-to-contact given a certain velocity (Mallot et al., 1992). Furthermore, a dead end is understood as an arrangement of obstacles that cannot be passed through (Möller, 1999). This understanding automatically considers the observer's physical properties: the obstacles may form a dead end only for a robot of a certain size; a smaller robot would pass between all obstacles. As second advantage, sensorimotor simulation provides a solution to viewpoint invariance. With anticipation, a dead end is recognized depending on its behavioral meaning and independent of the observer's perspective (Möller, 1999).

Here, sensorimotor anticipation is used to make sense out of the sensory input, which in the case of vision only provides a distorted map of the outside world; for example, lines are generally not mapped onto lines on the retina (O'Regan and Noë, 2001)—the same holds for our omni-directional vision system. Thus, here, we want to demonstrate that the geometric properties of the outside world can be understood using visuomotor anticipation, even if we predict only the (almost) raw visual input, a gray-scaled image as seen through the camera. Working with such an image representation avoids that the problem of making sense out of the sensory input is partially solved by a human engineer (Brooks, 1986), for example, by predicting object positions in image coordinates. Working with images will further allow us to visually illustrate the internal simulation.

Different from other robotic studies on sensorimotor anticipation, which were limited to navigational tasks (Tani, 1996; Tani and Nolfi, 1999; Gross et al., 1999; Ziemke et al., 2005), in the present study, a spatial arrangement is understood by a passive observer. A dead end could be recognized by a robot that stands still. Thus, the sensorimotor simulation must start from a single image. Analyzing only a single image prohibits the use of flow fields as in Gross et al. (1999) and the use of Elman recurrent neural networks for prediction (Tani, 1996; Tani and Nolfi, 1999; Ziemke et al., 2005). Instead, here, given only the current image, a forward model predicts the next image.

Images were predicted using a set of multi-layer perceptrons (MLP). Each pixel was computed by one three-layer perceptron. The MLP output, however, turned out to be noisy, and this noise accumulated catastrophically after just a few prediction steps. Thus, after each step, an image had to be denoised.

Images are elements in a high-dimensional space—if represented by a vector that contains each pixel's color. Usually, however, the distribution of images is locally restricted to a few dimensions. In our case, a small enough image patch shows one or two obstacles that vary only in location and orientation. Therefore, most of the noise can be removed by projecting the high-dimensional

image patch onto a low-dimensional manifold that models the distribution of training images (Mika et al., 1999; Shi et al., 2005). For learning a manifold, we already have a training set, namely the images that were collected to train the forward model.

To learn such a non-linear manifold, other studies used kernel principal component analysis (Mika et al., 1999) and locally linear embedding (Shi et al., 2005). These two methods, however, require all training patterns for denoising a single image. Thus, they are too slow for our purpose. Therefore, here, a denoising method is introduced in which the distribution of image patches is modeled with a mixture of principal component analyzers (Tipping and Bishop, 1999). The training patterns are then replaced by a set of spatially-localized subspaces.

The remainder of this article is organized as follows. Section 2 explains how perception through sensorimotor anticipation is realized in the robot. Section 3 describes the methods for learning to anticipate. Section 4 describes the experiments on perceptual judgment and shows results. Section 5 contains the discussion, and section 6 concludes the article.

2 Perception through sensorimotor anticipation

In the presented framework for perception through sensorimotor anticipation, a robot understands a specific property of the surrounding by simulating a specific movement plan. For example, in a previous study (Hoffmann and Möller, 2004), a robot simulated a turning movement; and after observing that the predicted visual input stayed constant, the robot deduced that it stood in the center of a rotation-symmetric surrounding. There, the plan was a fixed movement; but, generally, movement plans can be more flexible. In the present article, a movement plan allows a limited number of motor commands at each moment of the anticipation. The choice of motor commands depends on the current estimate of the visual input. Thus, the overall procedure of perception contains the following three parts: choosing an appropriate movement plan, carrying out the simulation, and during this simulation, mapping the predicted visual input onto a behavioral command.

A movement plan must be provided by the experimenter and depends on the desired perceptual quality, but see section 5 for a biological interpretation. The following three movement plans were simulated:

(1) To understand the distance to an obstacle, the robot moves forward uniformly until it bumps into the obstacle (see section 4.1).
(2) To understand if an arrangement of obstacles in front of the robot forms

a dead end or a passage, the robot simulates an obstacle-avoidance algorithm. The simulation halts when the front of the robot either is blocked by obstacles or faces free space. Depending on these two situations, the arrangement is recognized as a dead end or a passage (see section 4.2).

(3) For the same dead-end / passage classification task, the robot simulates an alternative strategy, a recursive search for an exit. At each prediction step, two movements are possible: a left turn or a right turn. Given this limitation, the robot tries all movement combinations; it travels along the branches of a search tree (Fig. 1). If the robot finds a path through the obstacles, the arrangement is recognized as a passage, otherwise as a dead end (see section 4.3).

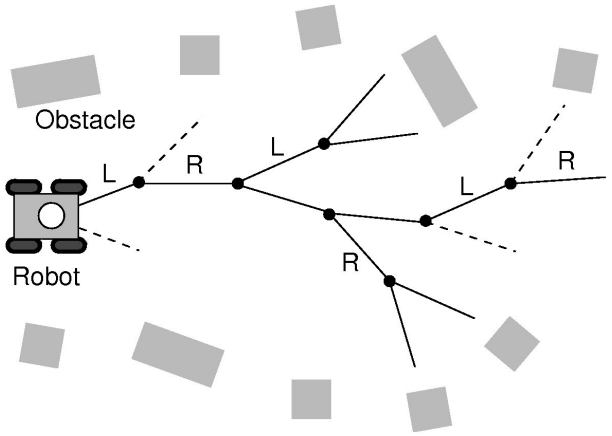

Fig. 1. Recursive search for an exit. At each black dot, the lines show the two movement alternatives for the robot. The first choice of movement is indicated by the letter L or R. Unused alternatives are shown by dashed lines.

In each plan, a movement starts from a single image. Given this image and the motor commands provided by the movement plan, a forward model predicts the next image, which is denoised subsequently. These steps are repeated until the movement plan gives a stop signal. Figure 2 shows the corresponding information flow.

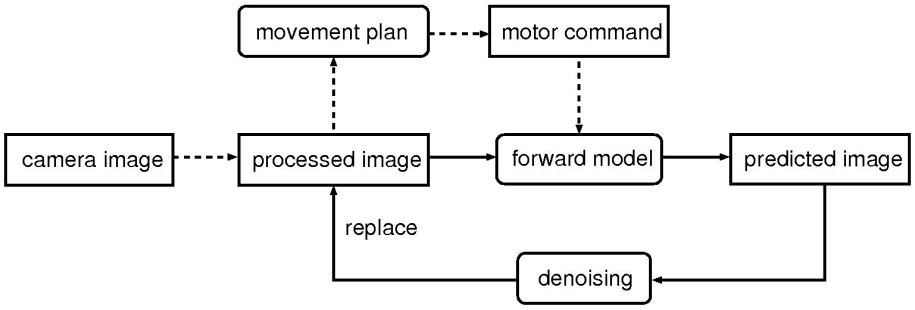

Fig. 2. Information flow for anticipation. Initially, a single camera image is given. After pre-processing and guided by a movement plan, the image is manipulated in a sequence of anticipation steps (solid lines). Each step consists of a forward model and a denoising model.

The movement commands chosen by the plan depend on the current mental image. In the present study, for simplicity, the mapping from this image onto the velocity commands is hard coded (in a more biologically-plausible version, the robot may learn this association with a neural network—see section 5). Within each image, the center of the robot is identified. From this center, the distance to obstacles was computed in three sectors in front of the robot (see Fig. 12 as an example). Depending on these three distances, the robot decides to turn left, to turn right, or to stop, as specified by the movement plan.

3 Methods for learning to anticipate

The robot learns to anticipate by observing the consequences of its actions within its environment. Here, learning comprises the following steps: collecting training data, image processing, learning a forward model, and learning a denoising model.

3.1 Collecting training data

To collect training data, the robot randomly explored its surroundings. A Pioneer 2AT four-wheeled mobile robot collected data within four different arrangements of obstacles, which were bricks wrapped in red paper (Fig. 3). Motor commands are the wheel speeds, which were chosen independently for the left and right side.

The random exploration was split into movement sequences. At the beginning of each sequence, the robot randomly chose a velocity pair (v_L, v_R) from the set $\{\,(20i, 20j)\,|\,i = -3, -2, \ldots, 3;\ j = -3, -2, \ldots, 3;\ |i| + |j| > 0 \wedge [\,ij \geq 0 \vee (|i| < 3 \wedge |j| < 3)\,]\,\}$ in the unit mm/sec (the last condition avoids fast turns, for which the MLP failed to produce reliable predictions—see section 3.3). The chosen speed was maintained for a maximum of five 2-second intervals. Such a sequence of constant velocity removes the dependence on the acceleration. A sequence ended prematurely if the robot got to close to an obstacle (Hoffmann and Möller (2004) describe the distance-detection mechanism used here). Afterwards, a new pair of velocities was chosen as described above.

For each obstacle arrangement, the robot moved for about 2 400 intervals, in total 9 443. At the end of each interval, an image was taken by the robot's camera. No image was taken from the beginning of a sequence to discard the effect of the first interval, which depends on acceleration. Two consecutive images and the corresponding velocity make one training pattern. The above

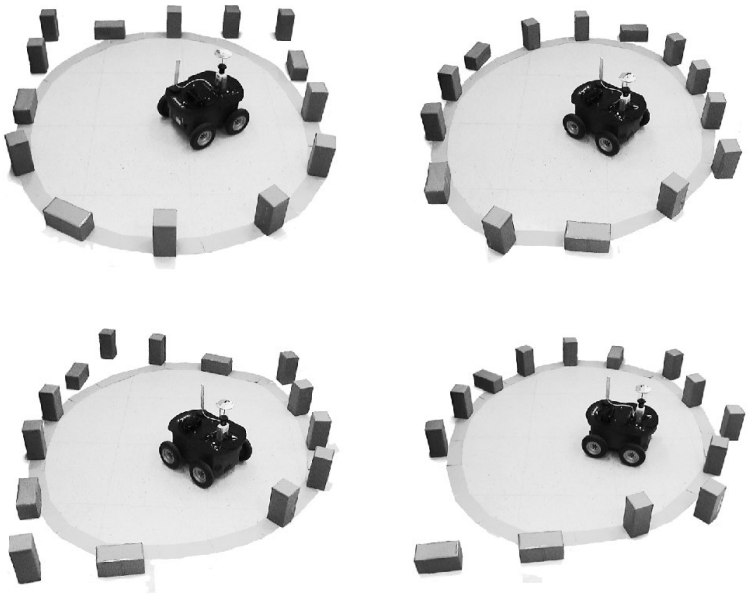

Fig. 3. Training environments. During training, the robot moved only within the oval shaped border (gray curve).

choice of velocities omits the zero-velocity case. To include this case, training patterns were added by using the first image of each constant-velocity sequence as a start and end image of a movement interval. This inclusion improved the performance of the network.

3.2 Image processing

An omni-directional vision system [2] provided the sensory input. The camera images were processed to emphasize the obstacles and to reduce the number of pixels (Fig. 4).

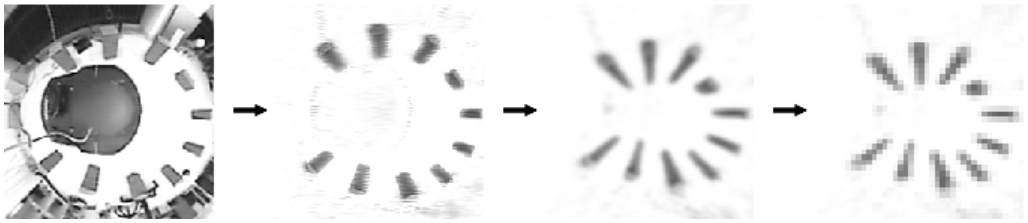

Fig. 4. Steps of image processing.

The processing comprised the following three steps. First, for each pixel, a gray value was computed by evaluating R-(G+B)/2 (R, G, and B stand for the red, green, and blue values). Second, the image was transformed such that the new

[2] The vision system comprised an Accowle hyperbolic mirror (middle size, wide view), a Pentax TS2V314A lens, and a DFK 4303/P camera.

distance of a pixel to the image center was proportional to the square of the original distance. This step counterbalanced an effect of the omni-directional mirror: objects farther away shrink over-proportionally; thus, the prediction of how an object moves within the camera image is much harder for larger distances. Third, the image was blurred and sub-sampled down to 40×40 pixels. The gray value of a pixel was scaled to the interval $[0, 1]$, with 0 being white and 1 being black.

3.3 Learning a forward model

The forward model predicts an image given the current processed image and the wheel velocities. Each image pixel was predicted using an MLP. For every pixel, a separate network was used. The network's input comprised the two wheel velocities and an image region (11×11 pixels) centered around the location of the output pixel (Fig. 5). When this region extended over the margin of the image, the input neurons on the outside were set to zero. The size of the input region limited the turning speed of the robot (see section 3.1), because a pixel is unpredictable if the movement of obstacles between two consecutive images is larger then the radius of the pixel's input region.

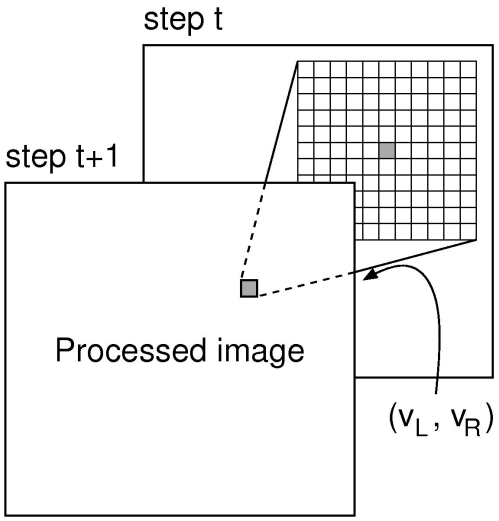

Fig. 5. Forward model. At step $t + 1$, a multi-layer perceptron predicts each pixel (gray square) given its surrounding (11×11 pixels) at step t and a motor command (v_L, v_R).

Each MLP has a simple three-layer structure with one hidden layer; the network's function $f(\mathbf{x})$ is

$$f(\mathbf{x}) = \alpha + \sum_{i=1}^{n_H} \beta_i \Big/ \left(1 + \exp\left(-\sum_{j=1}^{n_I} \phi_{ij} x_j + \theta_i\right)\right) \quad . \tag{1}$$

The vector $\mathbf{x}$ contains the input values x_j. The parameter n_H is the number of hidden neurons ($n_H = 15$) and n_I the number of input neurons ($n_I = 123$). The variables ϕ_{ij} and θ_i are the weights and biases between input and hidden layer, and β_i and α are the weights and bias between hidden layer and output neuron. All weights and biases were initialized randomly to lie within the interval $[-0.1, 0.1]$. On the data collected and processed as described above, the MLPs were trained using 2 000 epochs of resilient propagation (Riedmiller and Braun, 1993).

3.4 Learning a denoising model

To denoise an image, it was split into overlapping patches. A grid of 10×10 tiles (each 4×4 pixels large) was put on top of the image. A patch consists of the area of a tile together with a two-pixels wide border (in total: $d = 8 \times 8$ pixels). If the border reached out of the image, the affected pixels were set to zero (white). Each patch was denoised separately (Fig. 6). The overlap avoided discontinuities between tiles.

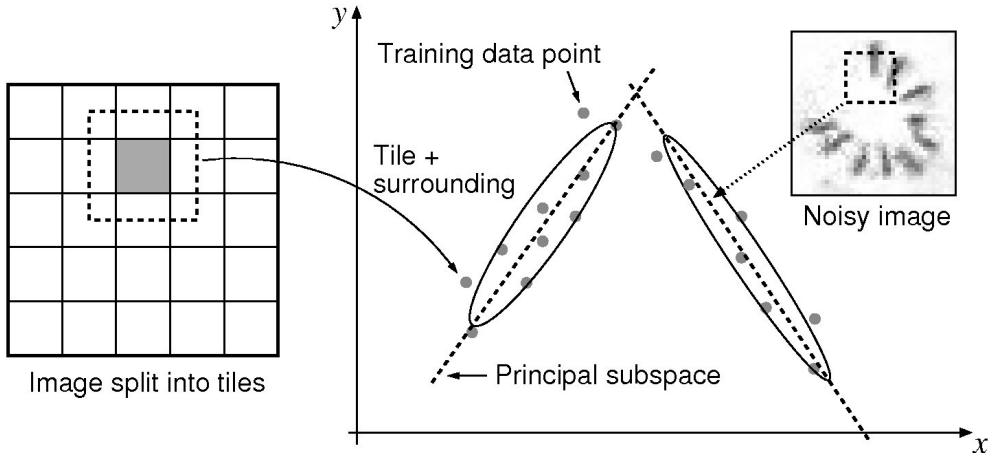

Fig. 6. Image denoising. (Left) Images are split into tiles, and each tile together with its surrounding (region inside the dashed square) is denoised separately. (Right) The distribution of such image patches (gray dots) is modeled by a mixture of Gaussians (ellipses). For each Gaussian, a small number of principal components spans a subspace (dashed lines). To denoise a patch, it is projected (dotted arrow) onto the principal subspace of the closest ellipsoid.

The training patterns were extracted from the same images as used for the forward-model learning, with the only difference that not all data were used. Instead, the number N of training patterns varied between tile locations. For some locations, most patches were almost white. Thus, to improve the computation speed, patches in which all pixels had a gray-value below 0.2 were removed from the training set.

For each tile location, the training patterns were modeled by a mixture of probabilistic principal component analyzers (Tipping and Bishop, 1999). The probabilistic framework allows to compute multi-variate Gaussians by evaluating only a few eigenvectors. Their number q, in the present article, was set to 5 unless otherwise noted. The number m of Gaussians in the mixture was set to $N/100 + 1$. Compared with Tipping and Bishop (1999), two improvements were necessary (Hoffmann, 2005). First, the Gaussian centers $\mathbf{c}_i$ were initialized with the vector quantizer 'Neural Gas' (Martinetz et al., 1993). Afterwards, as usual, expectation and maximization (EM) steps alternate; here, 30 EM steps were computed. Second, to increase robustness, after each EM step, a Gaussian was removed if its prior probability fell below $(q + 2)/N$, because $q + 2$ is the minimum number of data points required to compute q principal components and the residual variance. In the case of removal, to keep m constant, the Gaussian with the largest prior was split in two (Hoffmann, 2005).

Denoising a patch happened in two steps (Algorithm 1). First, from the mixture model, the Gaussian j was chosen for which the noisy patch had the smallest normalized Mahalanobis distance p_j. Such a distance value was computed from the eigenvectors $\mathbf{W}_j$ (a $d \times q$ matrix containing the eigenvectors in its columns), the eigenvalues $\mathbf{\Lambda}_j$ (a diagonal matrix), and the residual variance per dimension σ_j^2 as obtained from a spatially-localized probabilistic principal component analysis, which is part of the above EM-algorithm (Tipping and Bishop, 1999; Hoffmann et al., 2005). Second, the image patch was reconstructed based on the principal components of the chosen local model (Fig. 6 and line 9 in Algorithm 1). Finally, the border part of a patch was removed again, and the whole image was reconstructed from the tiles.

Algorithm 1 Image denoising

1: **for** each tile k **do**
2: from the image, extract the vector $\mathbf{x}_k$ that contains the tile k and its border
3: **for** $i = 1$ to m **do**
4: $\boldsymbol{\xi} = \mathbf{x}_k - \mathbf{c}_{k,i}$
5: $\mathbf{y}_i = \mathbf{W}_{k,i}^{\mathrm{T}} \boldsymbol{\xi}$
6: $p_i = \mathbf{y}_i^{\mathrm{T}} \mathbf{\Lambda}_{k,i}^{-1} \mathbf{y}_i + (\boldsymbol{\xi}^{\mathrm{T}} \boldsymbol{\xi} - \mathbf{y}_i^{\mathrm{T}} \mathbf{y}_i)/\sigma_{k,i}^2 + \ln \det \mathbf{\Lambda}_{k,i} + (d - q) \ln \sigma_{k,i}^2$
7: **end for**
8: $j = \arg\min_i p_i$
9: $\mathbf{x}_k^D = \mathbf{W}_{k,j} \mathbf{y}_j + \mathbf{c}_{k,j}$
10: **end for**
11: remove the border from each patch $\mathbf{x}_k^D$ and compose image from tiles

This denoising algorithm greatly reduced the noise in the predicted images (Fig. 7). The computational complexity for one denoising step is $O(n_T \langle m \rangle q\, d)$, where n_T is the number of tiles ($n_T = 100$), and $\langle m \rangle$ is the average number

of models in the mixture ($\langle m \rangle = 63.3$). In comparison, the complexity for one forward-model prediction step is $O(n_P\, n_H\, n_I)$, where n_P is the number of pixels in the whole image ($n_P = 1600$). For our parameters, these two complexity values were of the same order: $n_T\, \langle m \rangle\, q\, d \approx 2 * 10^6$ and $n_P\, n_H\, n_I \approx 3 * 10^6$. In the experiments, both forward and denoising step took about 0.05 sec each (Athlon 1800+ CPU, implementation in C++ based on the Basic Linear Algebra Subprograms—BLAS).

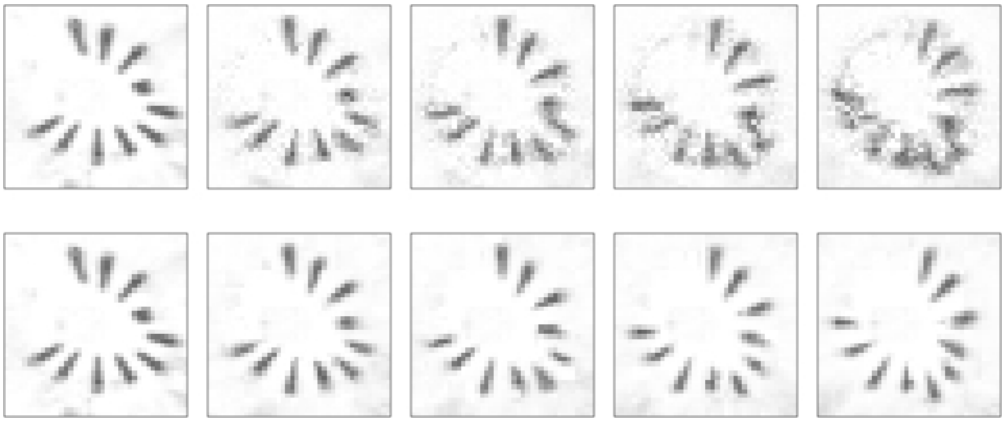

Fig. 7. Comparison of a forward prediction without (top row) and with (bottom row) denoising. Starting with the images on the left, each following image is obtained by predicting the sensory consequences of a turn with $v = (25, -25)$.

4 Experiments

In the experiments, the acquired ability to anticipate is exploited for perceptual judgment. Three tasks were studied: distance estimation, recognizing a dead end through a simulated obstacle avoidance, and recognizing a dead end through a simulated recursive search.

4.1 Distance estimation

The robot has to estimate the distance to an obstacle in front. Figure 8 shows the two setups used for testing. For each setup, the distance between obstacle and robot was varied in steps of 5 cm either from 20 cm to 85 cm (setup 1) or from 22.5 cm to 82.5 cm (setup 2). For each distance, the robot simulated a forward movement at a speed of 50 mm/sec. The simulation stopped when within the predicted image the distance to the obstacle reached a predefined threshold (7 pixels from the robot center). Figure 9 shows two sample simulations.

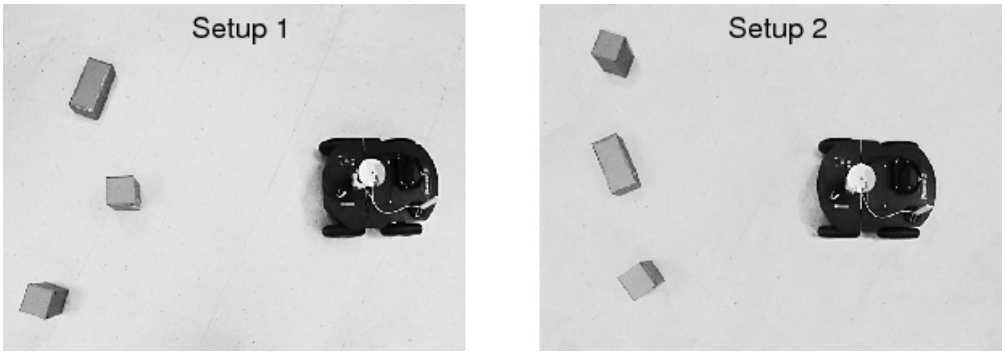

Fig. 8. Obstacle setups for distance estimation. The robot has to estimate the distance to the obstacle in front (the one in the middle). The two other obstacles only serve as a possible distraction.

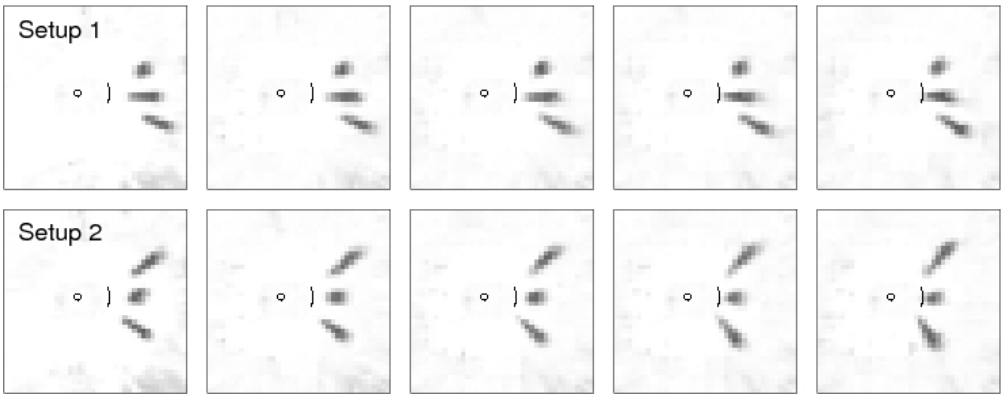

Fig. 9. Prediction of a forward movement toward the obstacle in front. Each row shows a simulation, which starts with the image on the left. The distance between robot and obstacle is 60 cm in *setup 1* and 57.5 cm in *setup 2*. A small circle marks the center of the robot in each image. The simulation stops when part of an obstacle reaches the circle segment.

For the quantitative results, the forward model was trained twice and the mixture model three times. Thus, taken together, we have six training-run combinations. Averages were taken over all six training runs.

The number of prediction steps required to reach an obstacle varied only little between these training runs (Fig. 10). For each distance, apart from the farthest, the difference between minimal and maximal number of prediction steps was within one step.

The average number of prediction steps was up to a constant offset proportional to the real distance (Fig. 10). Moreover, the slope of the linear increase of steps versus real distance was close to the expected slope (if the robot would actually move with the given speed).

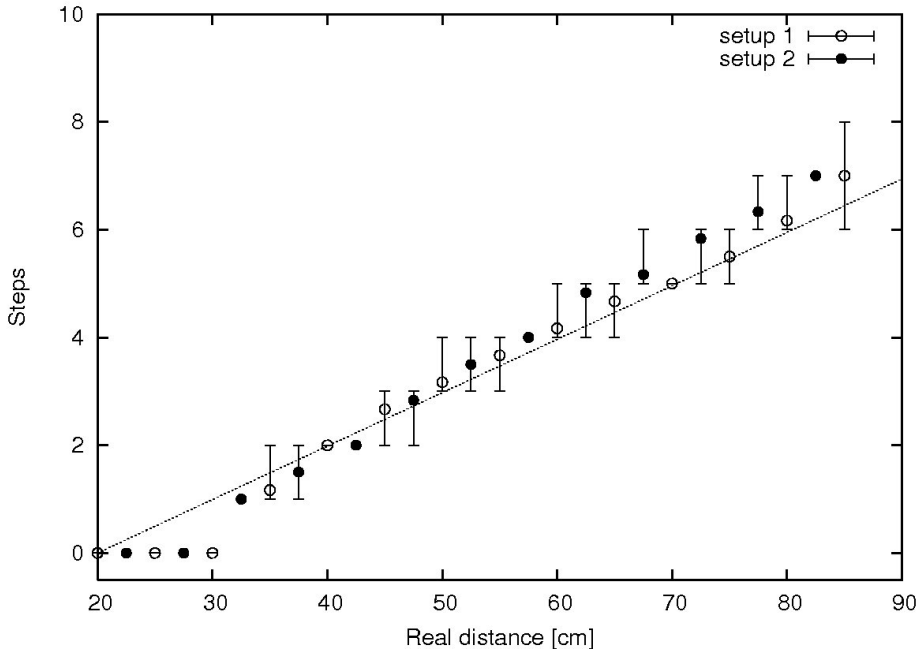

Fig. 10. Distance estimation. The graph shows the number of prediction steps needed to reach the obstacle as a function of the obstacle's distance. The average number of steps and the min/max values over six training runs are shown. In the absence of error bars, min and max values coincide. The dotted line shows the expected increase given a forward speed of 50 mm/sec.

4.2 Recognizing a dead end through simulated obstacle avoidance

Facing an arrangement of obstacles, the robot simulated an obstacle avoidance algorithm to recognize a dead end or a passage. This simulation was tested on 16 setups (Fig. 11). In the obstacle avoidance algorithm, the choice of velocities was derived from the mental image. Within such an image, the distance from the robot center to an obstacle was computed in three sectors (36° each, see Fig. 12). Depending on these distances, the robot either chose new velocities for the next prediction step or stopped the simulation and classified the arrangement. This choice was made according to the following rules:

- If the distance in the front sector is 6 pixels or less, or in all three sectors 8 pixels or less, then stop and classify the arrangement as 'dead end'.
- If the distance in the left sector is 7 pixels or less and smaller than in the right sector, then make a left[3] turn, $(v_L, v_R) = (-30, 30)$ in mm/sec (analogue for the right side).
- If the distance in the left sector is between 8 and 11 pixels and smaller than in the right sector, then turn leftward, $(v_L, v_R) = (20, 50)$ in mm/sec

[3] The image is mirror reversed.

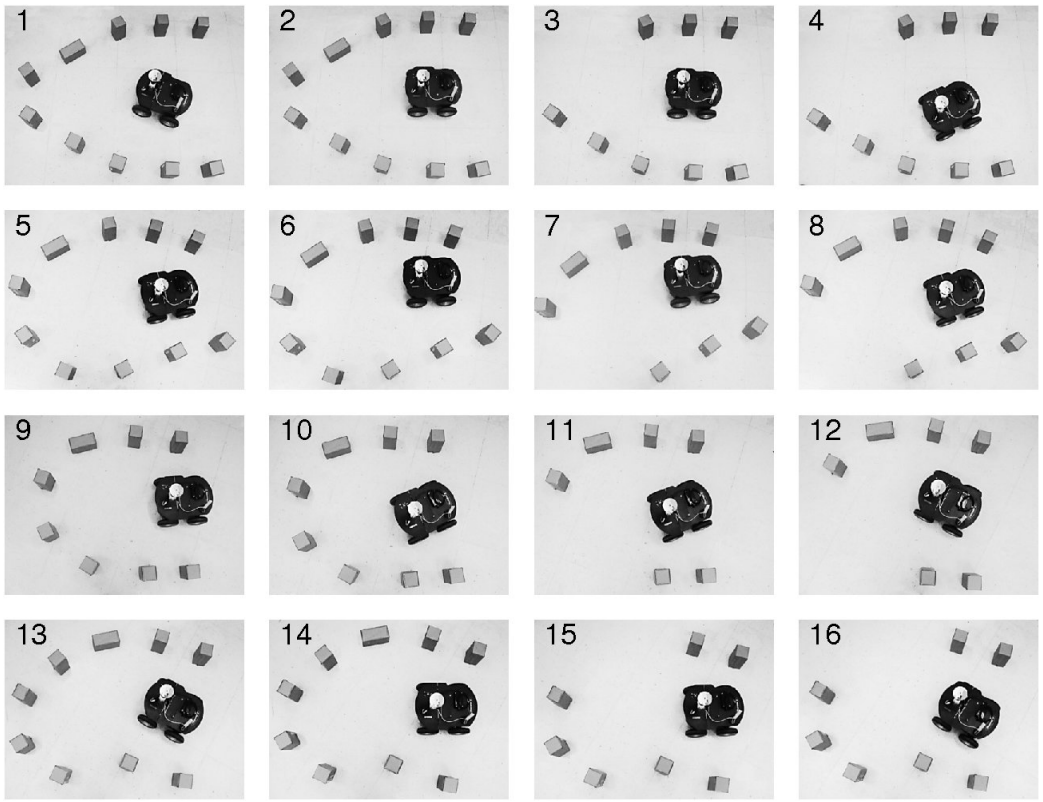

Fig. 11. Setups for the dead-end/ passage classification task. The dead ends are in the first two columns. For each obstacle arrangement, two setups were used with the robot facing in different directions.

(analogue for the right side).

- If the distance in all sectors is 20 pixels or more, then stop and classify the arrangement as passage.

In all other cases, the robot moved forward with 60 mm/sec. These rules restrict the movement of the robot such that in non of the test cases, the robot could turn around and leave the obstacle arrangement through the entrance. Figure 12 shows an example of a successful simulation run.

For the forward and denoising model, the same six training runs were tested as in section 4.1. In four out of six runs, all 16 setups were classified correctly (Fig. 13 illustrates the result of one of these four runs). In one run, one passage and in another run, four passages were misclassified. Figure 14 shows an example from a failed classification trial; a new obstacle appeared out of noise, transforming a passage into a dead end. New obstacles also appeared at a large distance behind the robot, as in Fig. 12 and Fig. 17. This error, however, did not disturb this anticipation. For all 16 setups, the robot computed on average 265 prediction steps (min: 228, max: 293).

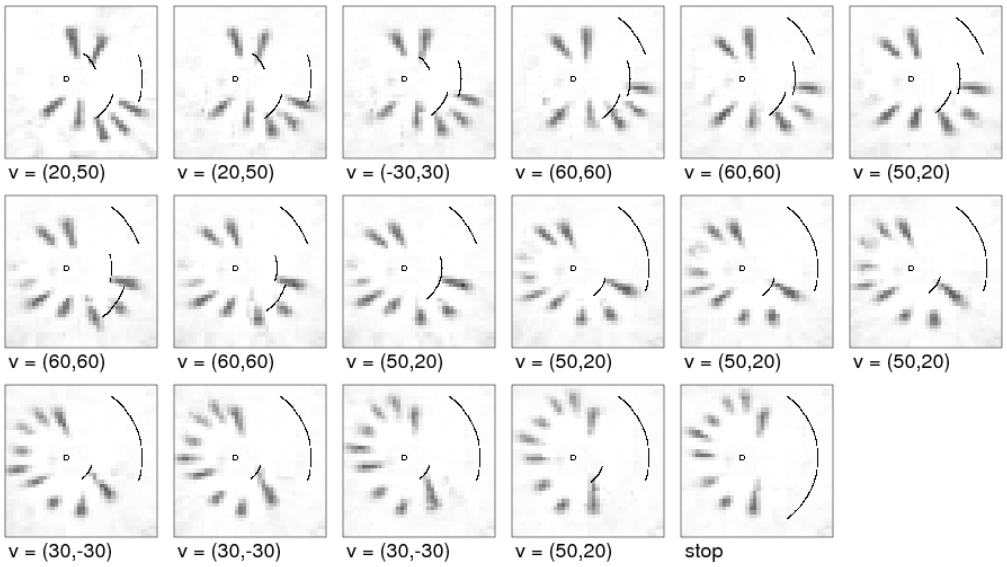

Fig. 12. Example of a simulated obstacle-avoidance algorithm (setup 16 in Fig. 11). The simulation starts from the top left. In each image, a small circle shows the center of the robot. Three circle segments mark the distances to the obstacles in front. Based on these distances, the velocity for the next time step is chosen (given below each image in mm/sec).

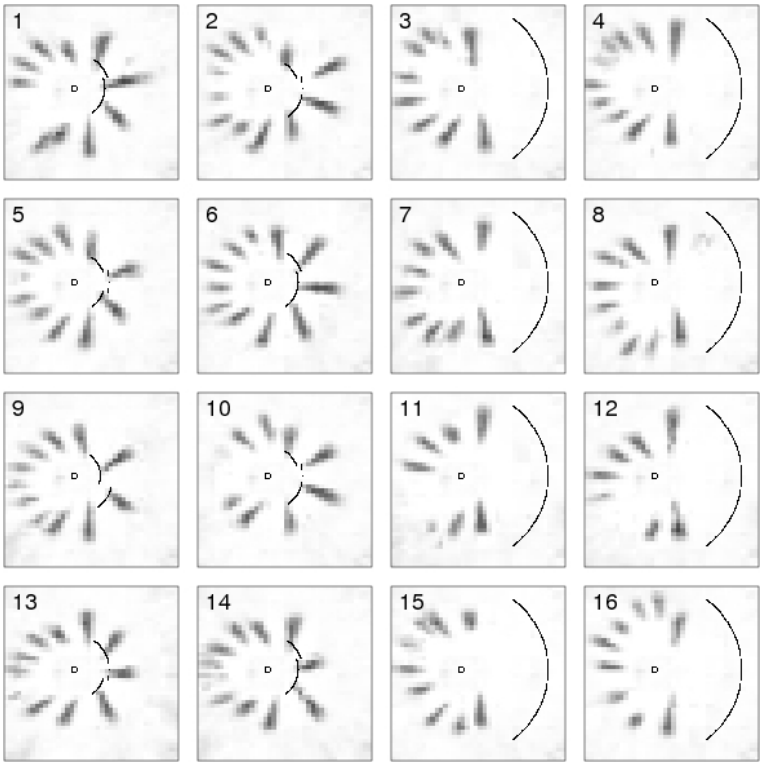

Fig. 13. Final images of the simulated obstacle-avoidance algorithm for all setups (numbered as in Fig. 11). Circles and circle segments are set as in Fig. 12.

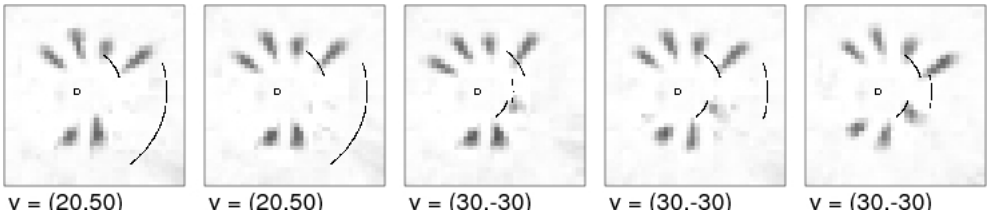

Fig. 14. Sequence showing how an obstacles emerges out of noise.

The proper function of the denoising algorithm depends on the number q of eigenvectors (Fig. 15). For a q-value between 1 and 5, the classification performance was good. However, if $q = 0$ (here, a noisy image patch was projected onto the center of the nearest Gaussian), denoising was too strong: obstacles disappeared occasionally or stood still at one position within the image; thus, dead ends turned into passages (Fig. 15, left). On the other hand, if q was too large, denoising was insufficient: new obstacles appeared occasionally, turning passages into dead ends (Fig. 15, right).

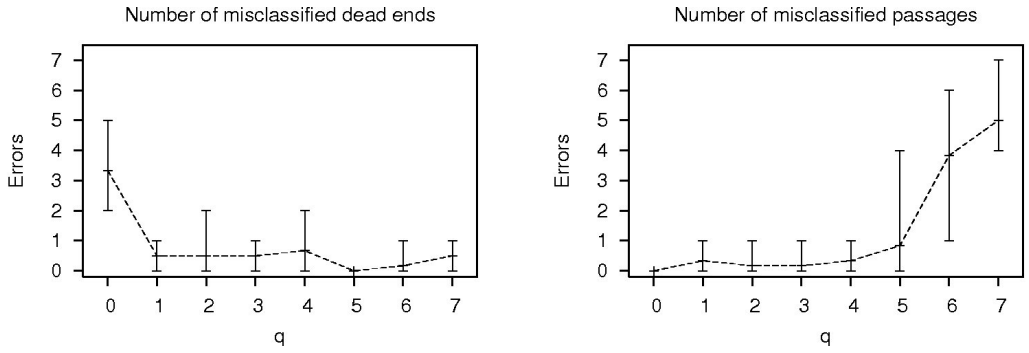

Fig. 15. Classification errors over all dead ends (Left) and passages (Right) depending on the number of principal components q used for denoising. The mean errors over six training runs are shown. Error bars show min and max values.

4.3 Recognizing a dead end through simulated recursive search

As an alternative to the obstacle-avoidance algorithm, the robot simulated a recursive search for an exit. At each branching point in the search, two velocity combinations were possible: $(v_L, v_R) = (20, 50)$ or $(v_L, v_R) = (50, 20)$ in mm/sec. These velocities forbade the robot to turn around and exit through the entrance. Choosing just two combinations limits the exponential growth of possibilities. To limit this growth further, at each branching point, two subsequent prediction steps were computed; that is, the movement intervals lasted 4 seconds, which reduced the required search depth (the number of decisions along a movement). The direction that was chosen first at a branching point depended on the search depth. With increasing depth, this direction altered between left and right (Fig. 1). Thus, the robot tries first a zig-zag

forward movement, which, in case of a passage, reduced the search time. Stop conditions were derived from the mental image as in section 4.2.

By simulating a recursive search, the robot could detect dead ends and passages. Results were again averaged over the six training runs as in section 4.1. The average classification error over all 16 setups was less than 1.0 using a maximum search depth between 9 and 12 (Fig. 16). Outside this range, the error increased. Figure 17 shows a sample movement found using a maximum search depth of 10. With this search depth, the robot computed on average 8001 prediction steps for all 16 setups (min: 6236, max: 10446).

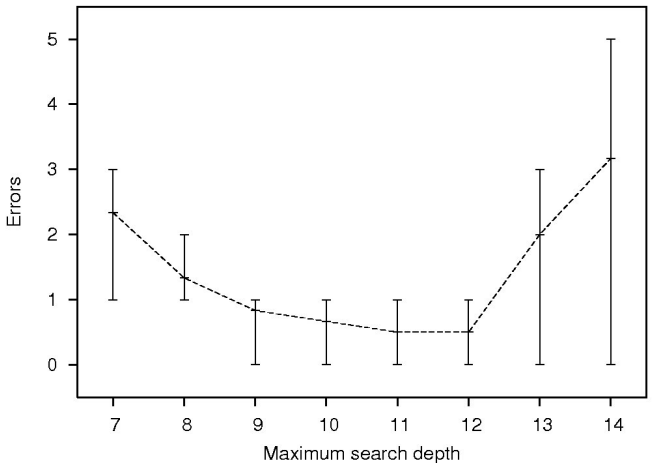

Fig. 16. Classification errors over all 16 setups depending on the maximum search depth for the recursive search. The mean errors over six training runs are shown. Error bars show min and max values.

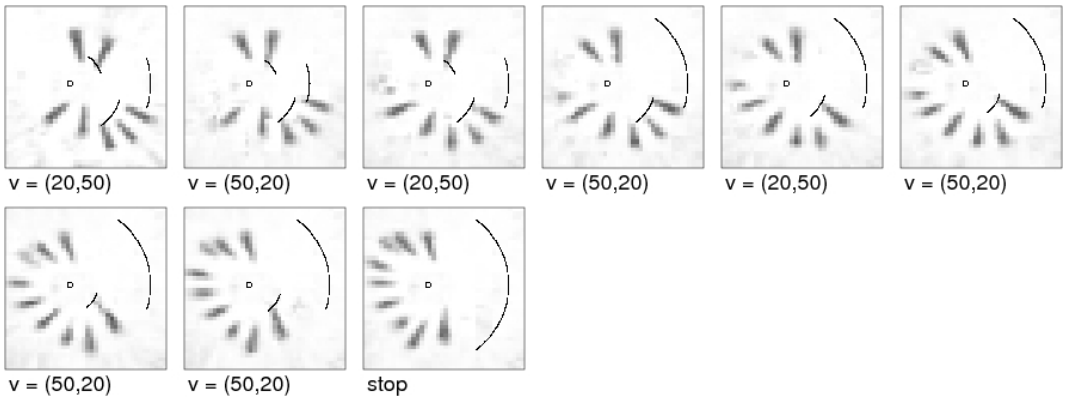

Fig. 17. Sample result of an internal simulation obtained through a recursive search (setup 16 in Fig. 11)—see also Fig. 12.

5 Discussion

The experiments demonstrated that a spatial arrangement of obstacles can be understood in its behavioral meaning based on sensorimotor anticipation. A robot learned to anticipate by moving actively and observing the sensory effect of its motor commands. Such a dependence of perception on active movement explains a behavioral experiment by Held and Hein (1963): kittens that grew up being only passively moved fail to avoid a steep cliff and thus do not understand depth. Here, depth perception might result from a simulated movement towards a target, as in the distance-estimation experiment (section 4.1).

The presented approach is biological relevant from the following perspective. An animal model was presented, which does not correspond to any real animal, but faces problems also the nervous system of animals and humans has to cope with: given the sensory input and the control over motor units, how can we infer properties of the outside world? This article demonstrated that this inference can be achieved based on visuomotor anticipation, which itself can be based on a forward model that was learned in an unsupervised way. Neurophysiological and psychophysical studies indeed indicate that such internal models exist in the brain (Kawato, 1999; Wolpert and Flanagan, 2001). The forward model was trained using standard machine-learning techniques (multi-layer perceptrons and Gaussian mixture models) rather than using biologically-realistic networks, because the focus was on the mechanisms behind perceptual understanding rather than on the learning techniques themselves.

Since the goal was to show how simulated action can lead to perceptual understanding, this work did not try to engineer the best possible solution for dead-end detection. For example, a more reliable mechanism might use a laser range sensor to compute the gap between obstacles and compare the gap size with the robot's size. From a biological perspective, however, this engineered solution conceals the real challenge because the distance to obstacles and the robot's size are represented in some abstract measure, which is a priori inaccessible to an animal and which requires calibration.

Sensorimotor simulation provided an efficient way to classify an image; such a classification would be difficult on a purely visual level. First, a teacher would be required to label dead ends and passages. Second, on the pixel level, the difference between two images showing a dead end and a passage could be smaller than the difference between images of the same class.

A mentally simulated sequence of movements could be, in principal, directly executed (without visual feedback). In reality, however, two technical problems arise. First, the predicted positions of the obstacles have small mistakes,

which implies that also the predicted position of the robot is erroneous. This error impairs action execution, but not perceptual judgment. Second, the commanded wheel velocity differs from the actually observed velocity (on the Pioneer 2AT, the velocity fluctuates around ±6 mm/s). Again, this error did not disturb perception since an MLP averages over the fluctuations in the training set. With this error, however, a lengthy movement sequence—like the one in Fig. 12, which shows how to move through a passage—cannot be successfully executed (Hoffmann and Möller, 2004). To work around these problems, the robot may execute only the first few steps of a simulation, take a new picture, run the simulation again, and so forth.

Prediction errors result mainly from the pixel-based sensory representation. Predicting the whole image means predicting a position in a high-dimensional space; and in each dimension, the position may be erroneous (Fig. 7). Here, this error could be fixed by part with a denoising step. For more complex environments, however, particularly, for the visual input of mammals, such a sensory representation is inefficient. A representation on a higher level is needed: for example, a location coding in place cells, as found in the hippocampus of mice, could be suitable for the spatial-perception tasks. In the present study, the pixel-based representation was chosen to demonstrate that sensorimotor anticipation can lead to an understanding of the structure behind the sensory input (a specifically engineered representation would therefore weaken this demonstration).

A dead end could be recognized by two different movement strategies: search recursively for an exit and avoid obstacles. The recursive search has the advantage that the choice of velocities does not depend on the distance to obstacles apart from the stop conditions. This independence saves parameters that need to be tuned in the obstacle-avoidance algorithm. Furthermore, a recursive search would also work if a dead end has a branching point. Disadvantages, however, are the dependence on the maximal search depth (Fig. 16) and the computational complexity. First, the lower bound on the maximal search depth depends on the length of a passage, and the upper bound depends on prediction errors: with increasing number of movement combinations also the chance of a failure (obstacle removal or emerging) increases. Second, the number of forward prediction steps was on average about 30 times higher than for obstacle avoidance. This imbalance makes the recursive search biologically implausible.

The mapping from a mental image onto the velocity commands may be learned beforehand; that is, the robot autonomously learns to avoid obstacles, and engineered rules as in section 4.2 can be omitted. To learn this mapping, the robot might move randomly and observe the visual effect of collisions with obstacles. Tactile sensors could detect these collisions. An example for carrying out such a learning presented Pfeifer and Scheier (1999): a robot learns to avoid

obstacles based on collision and proximity sensors and a simple neural network with Hebbian learning. Alternatively, a genetic algorithm may find a mapping by minimizing the robot's path length through a passage. These additions would replace the hard-wired mapping from image to motor commands and lead to a more biologically-plausible model.

Distances were estimated by simulating a movement towards an obstacle and counting the movement steps. In setup 2, for larger distances, one more prediction step was required than expected (Fig. 10). The reason might be the lying obstacle in setup 2 (Fig. 8). This obstacle appears to be smaller than a standing one within the camera image (Fig. 9). When moving towards obstacles, the learned mapping of the forward model has two effects: obstacles grow bigger (see the distant obstacles in Fig. 12) and move closer to the robot (Fig. 9). For large distances, the lying obstacle could not move closer during one prediction step because the image resolution was too low. Thus, to judge the distance, the robot has to use also the obstacle's apparent size. Since we work only with a two-dimensional projection of the three-dimensional world, the robot faces a size-distance ambiguity, and the lying obstacle appeared to be farther away than it actually was.

A perceptual quality was directly linked to a specific movement plan: for example, to understand a dead end, the robot needs to simulate an obstacle-avoidance algorithm. Here, no mechanism was presented on how to obtain an appropriate movement plan in a given situation. The execution of a movement plan may be triggered by other brain processes, which are influenced by action goals or sensory input, as the following two examples illustrate. (1) An animal wants to reach a distant food source. The goal to reach the food triggers simulations of the obstacle-avoidance algorithm. (2) We want to understand the function of a cup. The handle of a cup associates a grasping posture (Hoffmann et al., 2005). This grasping posture starts the simulation of various manipulations (for example, tilting and turning). Here, a cup might be understood based on these simulated manipulations.

Sensorimotor simulation appears to be well suited to extract geometric properties from the outside world; apart from dead ends, also symmetry can be understood (O'Regan and Noë, 2001; Philipona et al., 2004; Hoffmann and Möller, 2004). Not all kinds of perception, however, can be based on sensorimotor simulation. For example, in humans, information processing in an animal/non-animal classification task is too fast (150 ms) for recurrent processes (Thorpe et al., 1996), which are needed for mental simulation.

We do not know if the brain actually uses sensorimotor anticipation for perceptual judgment. Since sensorimotor simulation consumes a lot of time, a critical test could be to measure the processing speed of certain brain functions and the processing speed for perceptual judgment. If such a measurement would

rule out that perception is directly resulting from sensorimotor anticipation, it could still be the means to *learn* certain perceptual qualities.

6 Conclusions

This article presented experiments with a visually-guided mobile robot that carried out perceptual judgment based on visuomotor anticipation. The robot learned a forward model by moving randomly within arrangements of obstacles and by observing the changing visual input. For perceptual judgment, the robot stood still, observed a single image, and internally simulated the changing images given a sequence of movement commands as specified by a certain movement plan. With this simulation, the robot judged the distance to an obstacle in front and recognized in an arrangement of obstacles either a dead end or a passage. Thus, the experiments demonstrated that sensorimotor simulation makes the geometry of the outside world accessible to an observer—see also O'Regan and Noë (2001); Philipona et al. (2004).

To my knowledge, the presented article is the first, in which a real mobile robot infers properties as complex as the state 'dead end' or 'passage' based solely on a single image and a previously learned forward model. Thus, this work is significant for the understanding of how cognition emerges from sensorimotor models.

An image was predicted by computing each pixel with a multi-layer perceptron given as input the previous image and a movement command. This prediction, however, was noisy. Thus, a denoising method was introduced that greatly reduced the noise within an image (Fig. 7). This method described image patches with a Gaussian mixture model and mapped noisy patches onto the corresponding mixture of principal subspaces. Such a denoising method might be also helpful for other applications in which noise-free image patches are restricted to a low-dimensional non-linear manifold.

The mobile-robot experiment verified a thought experiment by Möller (1999): a robot can indeed detect a dead end through sensorimotor anticipation. In addition to the thought experiment, the robot study shows two points. First, sensorimotor anticipation still needs a suitable sensory representation to be computationally robust. A pixel-based representation is vulnerable to noise. Environments that are more complex than the one presented here require a higher-level dimension-reduced representation. Second, a real robot requires a specific simulated movement plan, and experiments show which plan is more efficient. A recursive search, as suggested by Möller (1999), is computationally expensive and prone to prediction errors for large search depths. Thus, sensorimotor anticipation by itself cannot generate a behavior from sensory

input by testing all possible movement variants. Instead, other processes need to trigger a suitable movement plan like, for example, avoiding obstacles.

Future work will study different sensory representations and will extend the anticipation approach to object manipulation using a robotic arm. Thus, objects or object properties will be recognized based on sensorimotor simulation.

References

Brooks, R. A., 1986. Achieving artificial intelligence through building robots. Tech. Rep. A. I. Memo 899, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, USA.

Gregory, R. L., 1998. Eye and Brain. Oxford University Press, UK, Ch. 8, pp. 136–169.

Gross, H.-M., Heinze, A., Seiler, T., Stephan, V., 1999. Generative character of perception: A neural architecture for sensorimotor anticipation. Neural Networks 12 (7-8), 1101–1129.

Grush, R., 2004. The emulation theory of representation: Motor control, imagery, and perception. Behavioral and Brain Sciences 27 (3), 377–442.

Held, R., Freedman, S. J., October 1963. Plasticity in human sensorimotor control. Science 142 (3591), 455–462.

Held, R., Hein, A., 1963. Movement-produced stimulation in the development of visually guided behaviour. Journal of Comparative and Physiological Psychology 56 (5), 872–876.

Hesslow, G., 2002. Conscious thought as simulation of behaviour and perception. Trends in Cognitive Sciences 6 (6), 242–247.

Hoffmann, H., 2005. Unsupervised Learning of Visuomotor Associations. Vol. 11 of MPI Series in Biological Cybernetics. Logos Verlag Berlin, PhD thesis (2004), Bielefeld University, Germany.

Hoffmann, H., Möller, R., 2004. Action selection and mental transformation based on a chain of forward models. In: Schaal, S., Ijspeert, A., Billard, A., Vijayakumar, S., Hallam, J., Meyer, J.-A. (Eds.), From Animals to Animats 8, Proceedings of the Eighth International Conference on the Simulation of Adaptive Behavior. MIT Press, Los Angeles, CA, pp. 213–222.

Hoffmann, H., Schenck, W., Möller, R., 2005. Learning visuomotor transformations for gaze-control and grasping. Biological Cybernetics 93 (2), 119–130.

Kawato, M., 1999. Internal models for motor control and trajectory planning. Current Opinion in Neurobiology 9, 718–727.

Mallot, H. A., Kopecz, J., von Seelen, W., 1992. Neuroinformatik als empirische Wissenschaft. Kognitionswissenschaft 3, 12–13.

Martinetz, T. M., Berkovich, S. G., Schulten, K. J., July 1993. "Neural-Gas" network for vector quantization and its application to time-series prediction. IEEE Transactions on Neural Networks 4 (4), 558–569.

Metta, G., Fitzpatrick, P., June 2003. Early integration of vision and manipulation. Adaptive Behavior 11 (2), 109–128.

Mika, S., Schölkopf, B., Smola, A. J., Müller, K.-R., Scholz, M., Rätsch, G., 1999. Kernel PCA and de-noising in feature spaces. Advances in Neural Information Processing Systems 11, 536–542.

Möller, R., 1996. Wahrnehmung durch Vorhersage—eine Konzeption der handlungsorientierten Wahrnehmung. Ph.D. thesis, Faculty of Computer Science and Automation, Ilmenau Technical University, Germany.

Möller, R., 1999. Perception through anticipation—a behavior-based approach to visual perception. In: Riegler, A., Peschl, M., von Stein, A. (Eds.), Understanding Representation in the Cognitive Sciences. Plenum Academic / Kluwer Publishers, New York, pp. 169–176.

O'Regan, J. K., Noë, A., 2001. A sensorimotor account of vision and visual consciousness. Behavioral and Brain Sciences 24, 939–1031.

Pfeifer, R., Scheier, C., 1999. Understanding Intelligence. MIT Press, Cambridge, MA.

Philipona, D., O'Regan, J. K., Nadal, J.-P., Coenen, O. J.-M. D., 2004. Perception of the structure of the physical world using unknown multimodal sensors and effectors. In: Advances in Neural Information Processing Systems. Vol. 16. MIT Press.

Prinz, W., 1997. Perception and action planning. European Journal of Cognitive Psychology 9 (2), 129–154.

Riedmiller, M., Braun, H., 1993. A direct adaptive method for faster backpropagation learning: The RPROP algorithm. In: Proceedings of the IEEE International Conference on Neural Networks. San Francisco, CA, pp. 586–591.

Rossetti, Y., Rode, G., Pisella, L., Farné, A., Li, L., Boisson, D., Perenin, M.-T., September 1998. Prism adaptation to a rightward optical deviation rehabilitates left hemispatial neglect. Nature 395, 166–169.

Schaal, S., 1997. Learning from demonstration. Advances in Neural Information Processing Systems 9, 1040–1046.

Shi, R., Shen, I.-F., Chen, W., Yang, S., 2005. Manifold learning for image denoising. In: The Fifth International Conference on Computer and Information Technology (CIT'05). pp. 596–602.

Sutton, R. S., 1992. Reinforcement learning architectures. In: Proceedings ISKIT'92 International Symposium on Neural Information Processing. Fukuoka, Japan.

Tani, J., 1996. Model-based learning for mobile robot navigation from the dynamical systems perspective. IEEE Transactions on Systems, Man, and Cybernetics—Part B 26 (3), 421–436.

Tani, J., Nolfi, S., 1999. Learning to perceive the world as articulated: An approach for hierarchical learning in sensory-motor systems. Neural Networks 12, 1131–1141.

Thorpe, S., Fize, D., Marlot, C., June 1996. Speed of processing in the human visual system. Nature 381, 520–522.

Tipping, M. E., Bishop, C. M., 1999. Mixtures of probabilistic principal component analyzers. Neural Computation 11, 443–482.

Verschure, P. F. M. J., Voegtlin, T., Douglas, R. J., October 2003. Environmentally mediated synergy between perception and behaviour in mobile robots. Nature 425, 620–624.

Webb, B., 2001. Can robots make good models of biological behaviour? Behavioral and Brain Sciences 24, 1033–1050.

Wolpert, D. M., Flanagan, J. R., 2001. Motor prediction. Current Biology 11 (8), R729–R732.

Wolpert, D. M., Ghahramani, Z., Jordan, M. I., 1995. An internal model for sensorimotor integration. Science 269, 1880–1882.

Ziemke, T., Jirenhed, D.-A., Hesslow, G., 2005. Internal simulation of perception: A minimal neuro-robotic model. Neurocomputing 68, 85–104.