

# Unsupervised learning of reflexive and action-based affordances to model navigational behavior

DANIEL WEILLER<sup>1</sup>, LEONARD LÄER<sup>1</sup>, ANDREAS K. ENGEL<sup>2</sup>, PETER KÖNIG<sup>1</sup>

<sup>1</sup>Institute of Cognitive Science  
Dept. Neurobiopsychology  
University of Osnabrück  
49069 Osnabrück  
Germany

<sup>2</sup>Dept. of Neurophysiology and Pathophysiology  
University Medical Center Hamburg-Eppendorf  
20246 Hamburg  
Germany

## Abstract

Here we model animat navigation in a real world environment by using place cell as a sensory representation. The cells' place fields divided the environment into discrete states. The robot learns knowledge of the environment by memorizing the sensory outcome of its motor actions. This was composed of a central process, learning the probability of state-to-state transitions by motor actions and a distal processing routine, learning the extent to which these state-to-state transitions are caused by sensory-driven reflex behavior (obstacle avoidance). Navigational decision making integrates central and distal learned environmental knowledge to select an action that leads to a goal state. Differentiating distal and central processing increases the behavioral accuracy of the selected actions. We claim that the system can easily be expanded to model other behaviors, using alternative definitions of states and actions.

## Introduction

Navigation refers to the practice and the skill of animals as well as humans to find their way and to move from one place to another by any means (Wilson and Keil, 1999). The ability of animals to navigate in essentially two-dimensional maze environments has been studied extensively (Olton and Samuelson, 1976; Morris, 1984). Navigation involves cognitive processes like sensory processing, actions execution and decision-making. Here we propose a cognitive model of these processes implemented on a robot faced with the task to navigate in a four-arm-maze environment. To solve this task the robot learns the sensory outcome of its actions, thus acquiring of environmental properties. This knowledge was used to plan and execute actions to solve the navigational task. We claim that the introduced cognitive model is not restricted to a navigational behavior and can easily be generalized to model other behavior.

Navigation can be defined by executing the appropriate action at the right location in an environment to move towards a goal. In the rodents hippocampus O'Keefe (O'Keefe et. al., 1971) found place cells, which encode the position of the animal. These cells fire only when the animal is located in a certain region of the environment, defined as the cell's place field. Although the contribution of these cells to the animal's behavior has still not been fully understood, it is assumed that these cells constitute a cognitive map (O'Keefe and Nadel, 1978) of the environment and are, thus, the bases of navigation. Wyss and coworkers (Wyss et al., 2006) have recently shown that place cells can be understood as an optimally stable representation of the visual input of a behaving robot in a hierarchical network. This implies that unsupervised learning of the sensory input results in a reorganization of the sensory space, spanned by its visual input, which has a spatial meaning. We used this fact by using place cells to locate the robot in its environment. We simplified this task by approximating the firing properties of place cells by Gaussian function and distributed the corresponding place fields in the whole environment. These place fields correspond to the robots internal states and represent the positions between which it is able to differentiate. Hence, in order to enable the cognitive model controlling the robot to perform navigational behavior, we chose place cells as the representation of the environment.

To navigate in the environment the robot first has to learn the sensory outcome of its actions and second to plan its actions according to its knowledge. Learning and planning are both done in its state space, spanned by the place fields. The robot learns local state transitions caused by its action execution. Because the execution of the same actions in a state can result in a transition to different states, the information gained from these local transitions is stored as transition probabilities in a probabilistic directed graph. The robot has also to avoid obstacles. We implemented a reflexive obstacle avoidance behavior controlled by the robots proximity sensors. In case the robot used its reflexive behavior during a transition between two states, we memorized the occurrence of such an event in so called reflex factors. Here the architecture of the cognitive model differentiated between central processing, responsible for state transition memorization and distal processing, responsible for reflex factor learning. The transition probabilities and the reflex factors reflect the environmental properties in relation to the robot's actions. Thus by random action execution, the robot learns an approximation of the environmental affordances (Gibson, 1977), defined the action possibilities afforded by the environment. The robot plans goal directed actions by integrating the information gained by central and distal processing in a local decision-making process. This integration results in a quantitative measure how reliably each executable action leads towards the goal. Overall, the key components of our cognitive model are (i) a high-level representation (place fields) of sensory input space, (ii) the knowledge of environmental properties acquired by active

exploration of local state transitions and (iii) a decision-making process driven by this knowledge.

Here we show that using the described cognitive model, a robot can successfully navigate to different goals within a four-arm-maze environment. As expected, the differentiation between central and distal processing reduces the negative effect of the obstacle avoidance behavior on navigational performance. We claim that by redefining the states and actions the introduced model can be expanded to model other behaviors.

## **Methods**

### *Overview of the architecture*

Our cognitive model learns the properties of the environment and plans its action to move towards a goal, based on the state space which is spanned by the spatial representation of place field (state). We divided the four-arm-maze environment (Fig. 1A) into compact discrete states (Fig. 1B) similar to place fields. The architecture of the cognitive model consisted of central and distal components. The central component captures the transition between states, caused by the robots action execution in these states. In contrast the distal component accounts for the usage of distal sensors, like infrared-sensors, facilitating obstacle avoidance during the robots state transition. Here the obstacle avoidance behavior is defined as reflexive behavior. While the central component accounts for any of the robots transition, the distal component constitute only transitions combined with reflexive behavior. Thus the transitions and the transitions combined with reflexive behavior represent the robots locally learned environmental properties according to the robots actions. To navigate to a particular target within the environment, the model chooses during the decision-making process the action that maximally increases the probability of reaching the respective spatial position.

### *Sensory processing*

We chose place cells as a representation of the environment. In a previous study it has been shown how such place cell properties can be acquired by mobile robots using unsupervised learning in a hierarchical network (Wyss et al. 2006). Because here our main purpose is to model behavior we deliberately used predefined place cells to simplify this task. We approximated the firing properties of place cells as a function of the robots position by Gaussian functions (standard deviation: 0.04 m). To cover the whole four-arm-maze environment we randomly distributed 72 of these Gaussian functions. Hence, for each of the robots possible position we obtain the activity of each place cell. A winner-take-all process extracted the robots position in the state space from the activity of the place cells. Accordingly the robot was located in the state (place fields) corresponding to the most activated place cells. (The used distribution of these states is shown in Figure 1B.) In order to determine the robots current state we had to extract its position and calculate the place cells activity using the distributed Gaussian functions as described above. Hence the robot was tracked by a Color Cmos Camera 905C (Analog Camera), which was attached above the environment as shown in Figure 1A. The analog camera signal was digitized by a Hauppauge WinTV Express card. With the help of the camera and the color code attached on top of the robot, its position and orientation were calculated. Thus, the place cell represents a mapping from the position space where the robot is navigating, to the state space of the agent, controlling the robot. The agent uses only positional information provided by this state space (place fields).

### Action execution

The robot was able to execute eight different actions in order to restrict the number of transition needed to learn the environmental properties. Each of these actions consisted of a certain orientation followed by a straight movement of the robot. The corresponding orientations were equally spaced from 0 to 325 degrees. As a result of executing such an action in a state (source), the robot will reach a different state (endstate) and thus results in a transition between states. The endstate is defined by the winner-take-all process calculating the current state, being dominated by another place cell. The position within a place field a transition is completed is defined by the place cell's activity not increasing anymore as the robot moves further and thus a local maximum of the activity is reached. The local maximum is defined by the derivative of the robots obtained activity of the place cells being zero. The frequencies of the transitions from source  $i$  with action  $k$  to endstates  $j$  is stored in the experience matrix  $EM_{i,j,k}$ .

### Reflexes

To prevent the robot hitting one of the mazes boundaries, a reflexive obstacle avoidance behavior was implemented. The proximity sensors (Fig 1C) were used in order to perform this behavior. If the robot had to use its obstacle avoidance behavior during action execution, the system associated the preceding state and action with the occurrence of a reflex event. The frequencies of co-occurrence of the reflexive event and a particular state ( $j$ ) – action ( $i$ ) combination is stored in the reflex matrix  $RM_{j,i}$ .

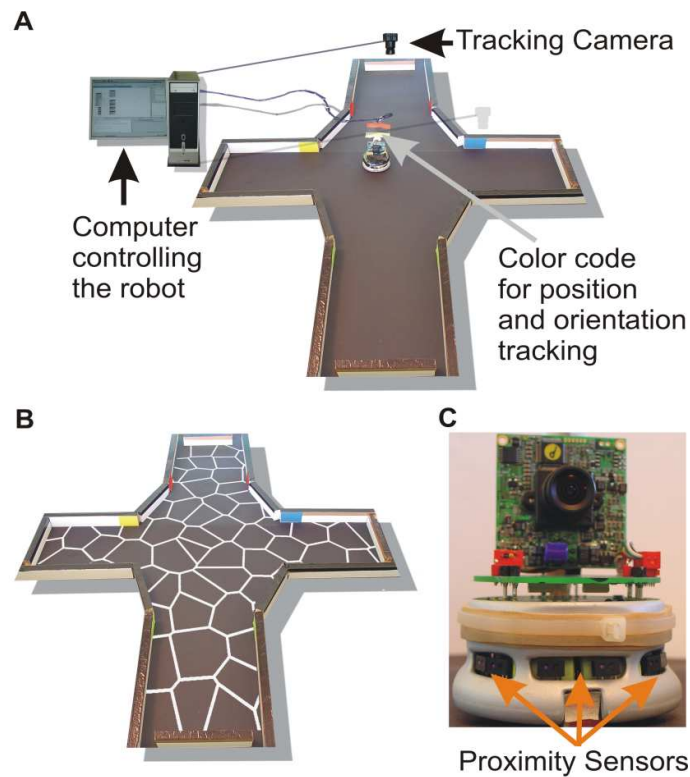


Figure 1: (A) We chose a four-arm-maze environment in order to test the model. A computer controlled the Khepera robot and extracted the robot's position and orientation, using the over head camera and the color code attached on top of the robot. (B) We subdivided the sensory space (position) into states. In the experiments we used this state distribution. The white boundaries assign for the region, one place cells is most activated. (C) Khepera robot used in the navigation experiments.

## Decision making

The properties of the environment (boundaries, obstacles, etc.) determine how likely it is that a certain state transition will occur given a chosen action. These state transitions are approximated and learned by the agent as it explores its maze environment and are stored in a transition matrix (Figure 2A). The transition matrix consists of a 2D matrix for each action  $i$   $TM_i$ . The row index determines the source  $j$ , the state where an action was executed and the row index represents the endstate  $k$  of this action. Thus the transition probability defined by source  $j$ , endstate  $k$  and action  $i$  is stored in the transition matrix  $TM_{i,j,k}$  shown in Figure 2A. Hence summing of the transition matrix over the endstates  $k$  (rows) is normalized to one for each action and sources. For the experiments described below, the robot learned the transition probabilities based on 240 minutes of random exploration.

Next we address the problem of choosing the action with the most desirable outcome to move towards the goal. To accomplish this task an iterative reverse flooding approach was introduced, which integrates the environmental properties (Fig. 2B). The properties gained from the central component of the architecture are stored in the transition matrix. This matrix consisted of eight 2D matrices  $TM_i$ , one for each action, which share similarities with a directed graph. The vertices of this graph correspond to the states, the edges correspond to the transitions and the edges weight to the transition probabilities. This results in 8 directed graphs equivalent to the eight possible actions. In each of the iteration steps of reverse flooding the activation of the state corresponding to the goal states is one. The activation of activated states is propagated through the graph by passing the activity weighted with the corresponding transition probability to the states with transitions to the activated one (reverse direction of the directed edges). Technically spoken the activation is propagated from the endstates to the sources weighted by their transition probability, representing a backward flooding. This process gives rise to 8 different activity values for each state. Thus up to now only the learned environmental properties resulting from the central process were considered during the flooding process. To integrate also the learned properties caused by distal processing we introduced reflex factors. The reflex factor is proportional to the percentage of actions  $i$  combined with a reflexive event at source  $j$ :

$$rf_{j,i} = 1 - \left( \frac{RM_{j,i}}{\sum_k EM_{i,j,k}} \right) \cdot \frac{5}{6}$$

The weighting factor of  $5/6$  was introduced to prevent zero activation at an action which is combined only with obstacle avoidance behavior. Thus during each iteration step the eight activations of state  $j$  corresponding each to one of the eight actions  $i$  were multiplied by the corresponding reflex factor  $rf_{j,i}$ . After each of the iteration steps the maximum of the eight activations of a state were accounted as the states activation for the next iteration step. This iterative process was continued until the states activity converged. In order to select an action on a state to move towards the goal we considered the eight different incoming state activation values which resulted from the activation propagation of the eight actions. The robot chose the actions which resulted in the highest incoming activation of a state.

Furthermore we introduced a decay factor  $df$  which was here 0.9. After each iteration step, the states activation was multiplied by this factor. As more transitions are needed to reach the goal states as more the decay factor is taken into account and thus decreases the states activity. Hence, the decay factor penalized these trajectories to the goal state with more transitions to the goal.

Here the flooding algorithm defined in the last section was implemented with the help of matrices.

$$act_j(0) \begin{cases} 0 & j \neq l \\ 1 & j = l \end{cases}$$

represented the activation at the 0'th activation propagation, where the goal was located at state  $l$ .

$$\overline{act}(t + 1) = \max_i((TM_i \cdot \overline{act}(t)) \cdot rf) \cdot df + \overline{act}(0)$$

where  $\overline{act}(t)$  is the vector of activation values for the states after  $t$  iteration steps.  $df$  represents the decay factor.

### Robot Setup

To test the model in a real-world environment we used Khepera II robots (K-Team). The robot was equipped with 8 proximity sensors, which emitted infrared light and measured the strength of its reflection, and two wheels, each controlled by one motor (Fig. 1C). For implementation and flexible programming, we used MicroPsi (Bach, 2003; Bach and Vuine, 2003), an Eclipse-based Java programming environment, as an interface to the robot. The agent that controlled the robot's behavior was implemented in this framework. The real-world environment was a four-arm maze with boundaries built from white wooden pieces (Fig. 1B). Each arm had a width of 0.21 m and a length of 0.28 m. The four-arm maze environment fitted into an area of one meter squared.

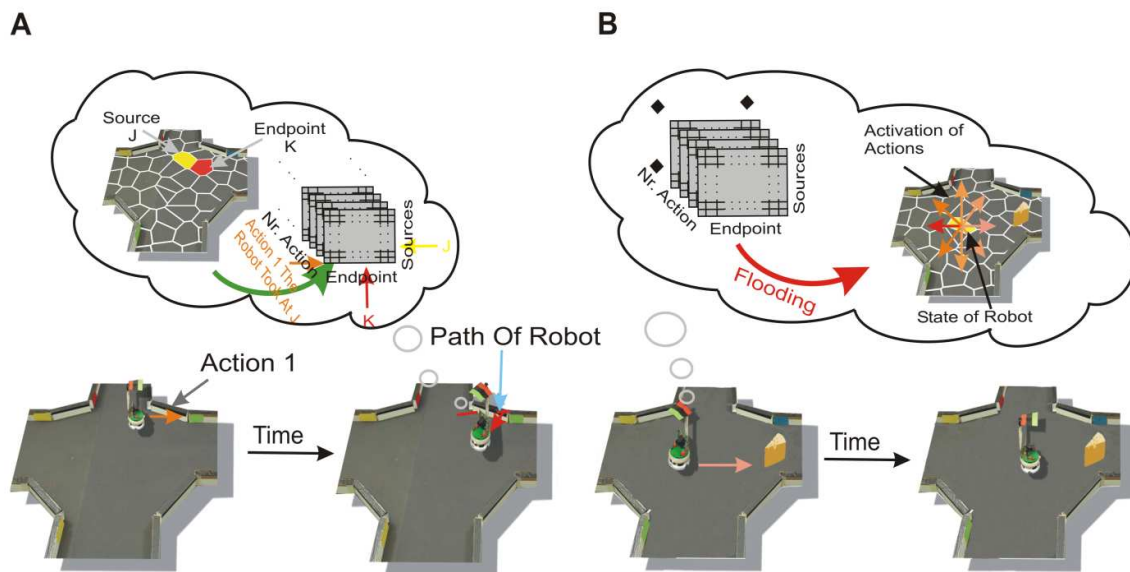


Figure 2: (A) Learning of the properties of the environment. The robot is on a certain state, defined here as Source J (yellow labeled) and randomly chooses an action (Action 1). The execution of the action results in another state, defined as endstate K (red labeled). This transition was stored in a three dimensional matrix, called the experience matrix, with the dimensions sources, endpoints and actions. The number of action executions combined with obstacle avoidance from a source were stored separately. (B) The robot moving to a goal (the "cheese" for the artificial "rodent"). His choice is a consequence of the flooding of the transition matrix, resulting in an activation of the different actions, shown as colored arrows. The action with the strongest activation was chosen.

## Analysis

As a means of comparison, a simulated robot was implemented using MATLAB (Version 7.0 (R14), Mathworks). The same navigational and experience algorithm was used as described above. The obstacle avoidance behavior was implemented by setting the angle of reflection equal to the angle of incidence to the boundary, with a random scatter of 10 to  $-10$  degrees added.

To compare the navigational behavior and the learned transition of the robot we introduce the geometrical transition matrix. It takes into account only the topographical properties of states in the environment. In order to experience the transition probabilities, based on the topographical properties, we let the simulated robot execute every action on each x/y position within the state given by the resolution of the tracker. Thus the geometrical transition matrix only takes the topographical distributions of states into account. Because the robot chose a new action according to the local maximum of the place cells activity, we weighted an actions transition by the probability of the robot choosing an action at the corresponding cell's activity. The execution of the different actions on each position within a state is due to transition probabilities resulting from an infinite experience time of the robot and thus represents the true underlying transition probabilities.

In order to compare the outcome of the different actions of one state learned by the robot, we measured the correlation coefficient of the transition probabilities of these actions on each state. We correlated the transition probabilities represented by a row vector of the Transition matrix of action  $i$ ,  $TM_i$ , with the same row vector of the Transition matrix of action  $j$   $TM_j$ . Before calculating the correlation coefficients between the two vectors we reduced the transition probabilities in the row vector by the average of these transition probabilities. This average was calculated by averaging over the transition probabilities of the topographical next neighbors. Thus two actions are equivalent when their correlation coefficient is 1.0; they are linearly uncorrelated when the correlation coefficient is 0.0.

To characterize the predictability of an actions' transition to a state we defined a second measure: The predictability of action  $i$  in state  $j$  is given by the maximum transition probability stored in the row vector  $j$  of the Transition Matrix  $TM_i$ . This maximum transition probability was reduced by the probability of transferring to one of the connected states by chance.

$$Pr_{i,j} = \max_k (TM_{i,j,k}) - \frac{1}{conn_{i,j}}$$

$Pr_{i,j}$  corresponds to the predictability of action  $i$  in state  $j$  and  $conn_{i,j}$  is the number of states the robot can transfer by executing action  $i$  on state  $j$ .

In order to evaluate the decision making process we analyzed the activation of each action calculated by the flooding process. We chose the normalized activity as an appropriate measure to characterize the selection of an action during navigating to a goal. This activity is defined as the most activated action on a state normalized by the sum of the incoming activity and the decay factor.

$$NormAct_j = \frac{\overline{act}_j}{\left( \sum_i ((TM_i \cdot \overline{act}(t)) \cdot rf) \cdot df + \overline{act}(0)_j \right) \cdot (1 - df)}$$

$act_j$  represents the converged activity of state  $j$  after the flooding process. The denominator corresponds to the sum of the activation of a state  $j$  over all actions; the activity of state  $j$  as

well as the sum of activities is given by the converged activity resulting from the flooding process. In order to reduce the dependency of the normalized activation onto the decay factor we multiplied the denominator by this decay factor. Thus normalized activity ranged from 0 to 10.

## Results

Here we investigated the robots navigational performance and how the central processes, namely the transition probabilities, as well as the distal processes, defined by the reflex factors, contributed to the decision-making process.

### *Navigation performance*

We investigated the navigation performance of the robot by analyzing its path to a number of different target sites in the environment. In each of the measured trials, the robot was placed on one of five possible starting positions and given one out of four target locations. In order to obtain a comparable measure we normalized the length of the robot's path by the *direct path*. The direct path represented the shortest traversable distance from the robot's starting point to the goal state. Figure 3 shows a path traveled by the robot (yellow line) and the corresponding direct path (light gray line). Overall the robot's median path length across 20 trials was 1.71 with a standard deviation of 0.47. This represents an increase of 71% ( $\pm 47\%$ ) when compared to the direct path. For all configurations of the start positions and targets, the robot was able to reach the target in a reasonably short amount of time.

This relative increase of the robots path length might have multiple causes: the division of the environment into discrete states (place fields), the robots learned transitions and the robots behavior while navigating through the environment. First we investigated the contribution of the discrete states in the lengthening of the robot's path to the targets. To provide a first approximation of this increase, we simulated the robot's behavior using the same navigational algorithm as described in the Methods section. The simulation used the geometrical transition matrix to navigate from the same start positions to the same goal states as the real robot. The transition probabilities of the geometrical transition matrix take only the topographical distribution of states into account (see Method section). Figure 3 shows a path of the simulated robot to a goal (red line). This simulation resulted in a median increase of 19% ( $\pm 9\%$ ) compared to the direct path. Thus, the discrete states used here to represent the environment did not greatly contribute to the lengthening of the robot's path to a goal.

How can we interpret the robot's navigational behavior? Approximately a quarter of the increase of the robot's path to a goal was caused by the usage of discrete states as a representation of the environment. Another quarter of the lengthening can be explained by the differences between the robots learned and the geometrical properties, stored in the robot's experienced and the geometrical transition matrix (data not shown). Further we analyzed the effect of obstacle avoidance onto the robots navigational performance. The agent engaged its obstacle avoidance behavior in 60% of the trials independent of the particular combination of starting and goal states. Analyzing only the trials in which the agent did not engage obstacle avoidance we obtained a median of 1.36 ( $\pm 0.23$ ). Thus the largest share of the lengthening of the robot's path compared to the direct path is due the obstacle avoidance behavior. In all trials, the robot was able to find its goal in a reasonably short amount of time, with the main increase in path length arising from the necessity of navigating through the narrow arms of the maze, where obstacle contact occurs most frequently.

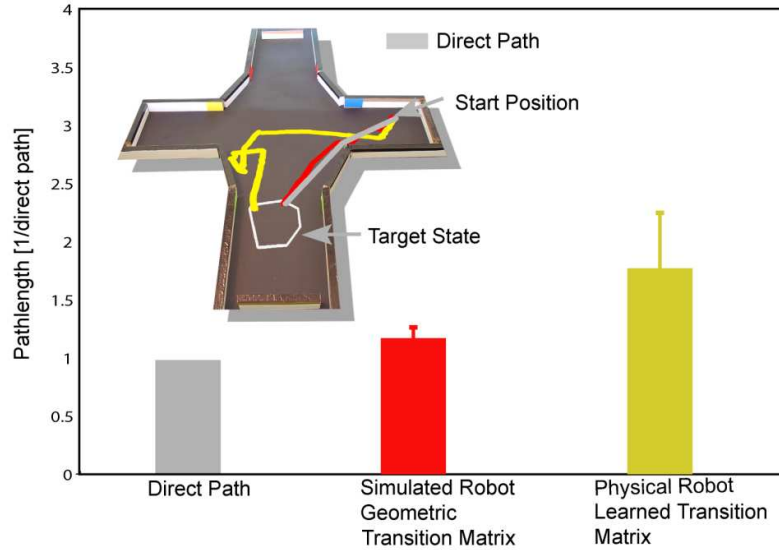


Figure 3: Navigational behavior of the robot was investigated by measuring the length of the path to different goals. The direct path, defined as the shortest traversable path from the start point to the goal state (shown as the gray line in the upper part), was used to normalize the length of the robot's path (yellow line) to the goal. The red line corresponds to the length of a path by a simulated robot by taking the topographical distribution of states (geometric transition matrix) into account. The bars represent the median length between different starting and goal states and their standard deviation.

### *Characteristics of the learned transition matrix*

The robot's navigational performance results from the decision-making process. This process is based on the learned transition and the learned reflex factors, representing the learned environmental properties. Here we investigated the characteristics of the robots learned transitions. We investigated first, the differences between the transitions of different actions on a state; second the influence of the used topographical distribution of states on the learned transitions and third the predictability of the state to which one action execution transit to. Fourth we examined the effect of the robots limited time to learn the environmental properties on the learned transition probabilities. For the most part we analyzed the transition matrixes characteristics by comparison to the simulation, based on the geometrical transition matrix (see Method section), representing the transitions based on the used topographical distribution of states. By this comparison we investigate the extent to which the topographical distribution of states gives rise to the investigated characteristics of the transition matrix.

Here we analyzed the similarity between the transitions of different actions, defined as the redundancy of the robot's possible actions on a state, by comparing the transition probabilities associated with these actions. For this purpose we computed correlation coefficients (see Methods and Figure 4A,B) between the transition probabilities of the different actions on each state. Higher correlation coefficients ( $>0.5$ ) were more frequently observed in the experienced transition matrix (44%) than in the geometrical case (25%), (Figure 4A). Thus the robot's real world action execution resulted in more similar outcomes and thus resulted in a higher redundancy of the actions compared to the geometrical case. Most (93%) of the highly correlated actions in the experienced case were obtained for states at the boundaries of the environment, and so were primarily due to the robot's obstacle avoidance behavior elicited by wall contact. The robot's action execution resulted in more similar transitions compared to the transitions based on the topographical distribution of states.

Next we investigated the influence of the topographical distribution of states on the robot's learned state transitions. Because the used topographical properties of states are fully represented by the geometrical transition matrix (see Method section), we calculated for each state and action the correlation coefficient between the transition probabilities stored in the geometrical and the robot's experienced matrix. Across all actions and states a mean correlation coefficient of 0.56 ( $\pm 0.52$ ) was obtained. Although these correlation coefficients are low it should be considered that these coefficients are calculated only for neighboring states and thus a conservative estimate. While different actions executed by the robot resulted in similar transitions more often than expected when only the topographical properties of the states are taken into account, the topographical state distribution nevertheless had an influence on the robot's learned transitions.

We then analyzed the predictability of action outcomes. Predictability defines the ability to predict the state to which one action execution makes a transition. In order to evaluate the actions' predictability we introduced predictability values (see Method section), proportional to maximum transition probability of an action. Figure 4D shows the occurrence of predictability values for the experienced and geometric transition matrices. Lower predictability values ( $< 0.3$ ) of the actions occurred more often in the experienced case (37%) compared to the geometric one (13%). Thus in general the robot's actions are equally likely to reach a number of spatially adjacent states. This is due to the actions transition probabilities characterized by a non-sparse probability distribution. Furthermore we investigated the influence of the obstacle avoidance behavior on the action predictability of the experienced transition matrix. Most (84%) of the low predictability values are due to actions for which the robot had to use its obstacle avoidance at least once. Thus obstacle avoidance reduced predictability of the action result. In most cases we obtained a lower predictability of the robot's resultant state, than we would have expected by the topographical distribution of place fields.

Are the differences between the robot's learned and the geometrical properties due to the robot's limited experience time? We generated the transition probabilities of the geometrical transition matrix by simulating the execution of each action on each position within a state (see Method section). Devolving this procedure to the robots learning of the transition probabilities, it has to experience its environment for an infinite time. In contrast, the robot's experienced transition matrix is based on executing each action on each state 11.54 times on average. Here we investigated the influence of the robots limited experience to the mean correlation coefficient between the geometrical and the robots experienced transition probabilities ( $0.56 \pm 0.52$ ). In order to investigate the influence of the robot's limited experience time on the difference between the geometrical and learned matrix, we compared generated geometrical transition matrices to the geometrical transition matrices. The generated geometrical transition matrices were calculated like the geometrical transition matrix; the only difference in the generated case is the number of actions executed on each state restricted to the one of the robots and thus was less than for the geometrical transition matrix. We simulated 300 generated transition matrices. In order to compare these matrices we correlated the transition probabilities for each action and state of the generated matrices with the geometric one. We averaged these correlation coefficients for each generated transition matrix. This yielded a distribution of averaged correlation coefficients with a mean value of 0.86 ( $\pm 0.1$ ). Thus, the averaged correlation coefficient of 0.56 ( $\pm 0.52$ ) between the geometric and the robots learned transition probabilities were lower than the correlation coefficients between generated and geometrical transition matrix. Thus the difference between the robot's experienced transition matrix and the geometrical transition matrix is dominated by the robots behavior and not due to limited time the robot experienced the environment.

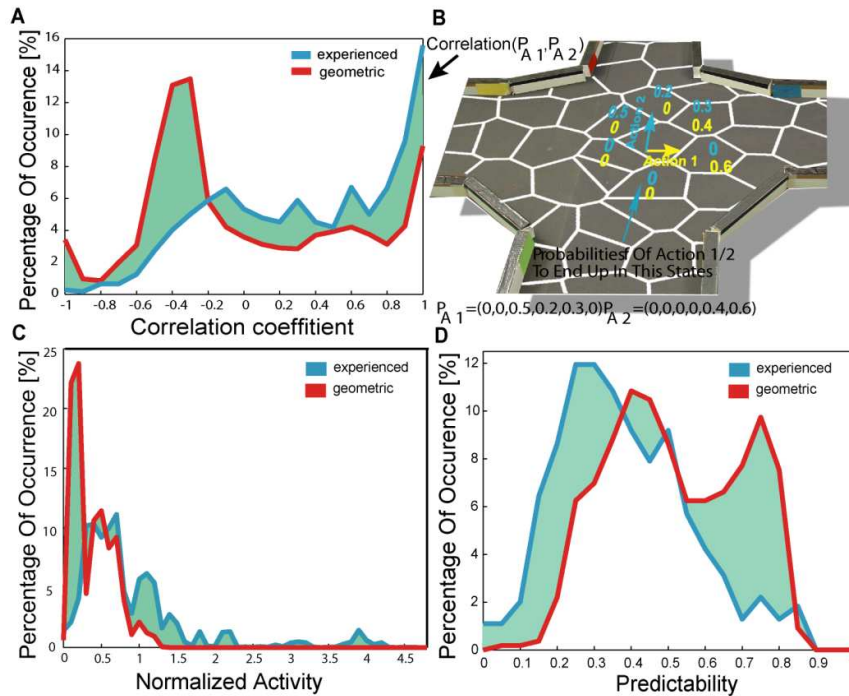


Figure 4: (A) The percentage of occurrence of the different correlation coefficients between the transition probabilities of different actions of a given state. The coefficients for the geometrical and the robot's experienced transition matrix are shown. (B) An example of calculating the correlation coefficients displayed in A.  $P_{A1/2}$  represents the probabilities of the action 1/2. (C) Here the ratio of the most activated action to the sum of all incoming activation (normalized activity (see Method section)) of a state is shown. The higher reflex factors in the experienced case (data not shown) increase the normalized activity compared to the geometric case. (D) Distribution of maximum transition probabilities of all experienced actions and states.

Here we have investigated the properties of the robots learned transition probabilities in the Transition matrix. We obtained a similarity between the transitions of different actions in the robot's experienced compared to the geometrical transition matrix. Also in general a lower predictability of the transition of the robot's actions to a state was obtained compared to the geometrical case. Thus both properties of the robots transition matrix are not fully caused by the topographical distribution of states. The obstacle avoidance behavior gives rise to this lower predictability as well as the similarity of the action results. Neglecting the reflex factors during the decision making process, which navigational behavior would result by only taking the learned transition into account? We would expect that it is not important for the robot to choose a precise action while moving towards a goal, caused by low action predictability as well as the high similarity between the transition probabilities of the different actions. However the transition matrix is influenced by the geometrical distribution of the place fields, while the obstacle avoidance behavior causes a similarity between the actions and a low predictability of an action's resultant state.

### Reflexes

Next we investigate the impact of the distal processing on the agent's decision making process, which involves the selection of actions in order to move to a goal. The flooding processes integrates the distal components in the decision making process with help of the reflex factors. After flooding (see Method section), the agent selects the action most highly activated at the state corresponding to the robot's location. Here we investigate the impact of the reflex factors on this process by analyzing the normalized activation (see Method section). This measure is proportional to the ratio of the highest action's activation to the sum of the

other action's activation on a state. Thus a low normalized activation describes a decision making process with the execution of different actions would result in a similar navigational performance. In contrast, high values define a decision making process in which the agent chooses a precise action in order to move to the goal and thus executing different actions than the most activated one would result in different navigational performances. Taking only the transition matrix during the flooding process into account and thus neglecting the reflex factors, based on the properties of the transition probabilities investigated above, we would expect lower normalized activities compared to the geometrical transitions. In contrast, taking the reflexes into account, this normalized activation was higher for the experienced than for the geometric transition matrix (Figure 4C). This implies that the robot chose a precise action in order to move to a goal and thus executing a different action than the highest activated one result in a worse navigational performance. The higher normalized activations for the experienced transition matrix are due to higher reflex factors compared to the geometrical transition matrix (data not shown). Thus taking the reflexes into account reduces the effects of the obstacle avoidance behavior on the learned transitions during the decision-making process and results in a more precise action selection in order to successfully reach a goal.

How do the different components of the algorithm influence the behavior of the robot? Here we analyzed the contribution of the central processes and distal processes to the robots decision-making process. Taking only the central processes, namely the state transitions, for the decision-making into account, different actions executions would result in similar navigational performances, although navigation in the narrow arms required a precise action in order to reduces hits against the walls and thus reduce the path length to goals. In contrast, integrating the distal learned environmental properties, namely reflexes into the decision-making process the robot has to execute one precise action to navigate towards the goal. Thus as we expected, taking the distal processing into account reduces the effects of reflexive behavior and allows the robot to successfully navigate in the environment.

## **Discussion**

Here we have introduced and implemented a model that allows a robot to navigate through an environment. The model learns the environmental properties, in an unsupervised manner by randomly executing the robot's actions possibilities. Because the robots learning process was done in a finite time period, the robots knowledge of its actions possibilities only approximates its environmental affordances. The architecture of this model differentiated between central processing versus distal processing. The distal processing is defined by the state transitions where reflexive behavior of the sensory-driven obstacle avoidance occurred. The central processing is represented by all learned transitions between the states. The reflexive behavior acts upon the robots learned transitions, resulting in uniformly distributed and less predictable actions outcomes than we would have expected by looking at the used topographical distribution of place fields. However, as expected the integration of the information gained by the reflexive and central processing in the decision-making process reduced the impact of sensory-driven behavior on the navigational performance. Consequently the robot was able to successfully navigate in the environment in a short amount of time.

The cognitive model is based on a sensory representation composed of discrete states. In this state space First the robot learned the sensory outcomes of its actions execution, namely the state transition and the reflex factors. Thus, the robot learned the environmental properties with respect to its actions. Based on these results, the robot planned its action in order to move

to the goal state in its internal state space. We defined the states such that they are equivalent to place cells place field, providing a representation of body position within the external space. These place cells can be understood as an optimally stable sensory representation of the visual input given by a robot moving in an environment (Wyss et al., 2006). The unsupervised learning resulted in a reorganization of the sensory space spanned by the robots visual input, leading to a low dimensional representation of the sensory input with a spatially meaning. In order to model other behavior, we have to choose an appropriate organization of the sensory space. On this sensory representation states can be defined, resulting in a state space. Further a definition of actions has to be done, which is adapted to the behavior to be modeled. Corresponding to these actions the sensory outcome in the state space can be learned. Differentiating between distal, namely the transitions influenced by the sensory driven behavior, and the central processes, the state transitions, would result in a better performance of the system to reach a certain goal state. Using a different sensory representation and other definition for the possible actions, different behaviors can be modeled.

Different studies have modeled navigational behavior by using place cells as a representation of the environment. Here we divide the different approaches into two group characterized by the type of learning used: Hebbian learning or reinforcement learning. The first type of learning exploits the fact that while moving in the environment, more than one place cell is active at the rodent's location, caused by the overlapping place fields of the corresponding cells. The Hebbian learning approach takes this fact and applies the biologically motivated principles of LTP and LTD, resulting in a strengthening of the connections between place cells which were active in a certain time interval. These cells and their connections between each other represent a cognitive map (Gerstner and Abott, 1996; Blum and Abott, 1996; Gaussier et al., 2002). Other studies introduced a cell type - goal cells - representing the goal of the navigational task (Burgess et al., 1997; Truellier and Meyer, 2000). The connections between the place and the goal cell encode the place cell's direction to the goal. The strength of connections between these two cell types was also modulated by Hebbian learning. In contrast to our model, the mentioned approaches rely on a global orientation and a metric, measuring the directions and distance to the goal at a given location within the environment. The global orientation used by these studies is defined using the same frame of reference over the whole environment. In contrast, we wanted the robot to learn the topology of the environment and thus did not introduce global variables as orientation or a metric. Furthermore, some of the mentioned studies (Stroesslin et al., 2005; Forster et al, 2000; Gerstner and Abbott, 1997; Burgess et al., 1997; Truellier and Meyer, 2000) used population coding to encode the position or direction to the goal. The population vector approach is based on the assumption of place fields and rodent's orientations having separate topologies. Thus to decode the robot's position or orientation the weighted average of place cells or orientations has to be calculated. This incorporates knowledge of the topology in the decoding scheme and impedes a generalization to other action repertoires. In contrast we defined the actions independently of each other so that the action repertoire can easily be expanded, for example including the action of lifting an object. Other branches of studies (Forster et al., 2000; Aleo and Gerstner, 2000; Stoesslin et al., 2005) used reinforcement learning (Sutton and Barto, 1997) to perform a navigational task. The concepts of Markov Decision Process and value iteration (Sutton and Barto, 1997) are commonalities between reinforcement learning and our approach, while in our model the value iteration was expanded by reflexes. A pure reinforcement learning approach involves learning the properties of the environment by using an explicit reinforcement signal, given by a goal state; while in the presented model these properties are latently learned (Tolman, 1948), resulting in a global strategy for navigation in this environment. In contrast to other studies, here we presented a cognitive model that is able to learn the topology and properties of the environment in a latent manner

and can be expanded to model other behaviors by redefining the meaning of the actions and states.

We introduced a cognitive architecture in order to model animal-like behavior and tested it in a navigational framework. The navigational performance given by this architecture is not constrained to a specific setup because the behaviorally interpreted properties of the environment are self-learned and not predefined. Here we showed that differentiation between central and distal processing routines resulted in a better navigational performance. We argued that this cognitive model can be expanded to model other behavior.

## References

- Arleo A, Gerstner W. (2000) Spatial cognition and neuro-mimetic navigation: a model of hippocampal place cell activity. *Biol Cybern*, 83(3):287-99.
- Bach, J. (2003) The MicroPsi Agent Architecture *Proceedings of ICCM-5, International Conference on Cognitive Modeling, Bamberg, Germany* (pp. 15-20)
- Bach, J., Vuine, R. (2003) Designing Agents with MicroPsi Node Nets. *Proceedings of KI 2003, Annual German Conference on AI. LNAI 2821, Springer, Berlin, Heidelberg.* (pp. 164-178)
- Blum KI, Abbott LF. (1996) A model of spatial map formation in the hippocampus of the rat. *Neural Comput.* 8(1):85-93.
- Burgess N, Donnett JG, Jeffery KJ, O'Keefe J. (1997) Robotic and neuronal simulation of the hippocampus and rat navigation. *Philos Trans R Soc Lond B Biol Sci.* 29:352(1360):1535-43.
- Foster DJ, Morris RG, Dayan P. (2000) A model of hippocampally dependent navigation, using the temporal difference learning rule. *Hippocampus.* 10(1):1-16
- Gaussier P, Revel A, Banquet JP, Babeau V. (2002) From view cells and place cells to cognitive map learning: processing stages of the hippocampal system. *Biol Cybern.* 86(1):15-28.
- Gerstner W, Abbott LF. (1997) Learning navigational maps through potentiation and modulation of hippocampal place cells. *J Comput Neurosci.* 4(1):79-94.
- Gibson James J. (1977) The Theory of Affordances. In *Perceiving, Acting, and Knowing, Eds. Robert Shaw and John Bransford*
- Morris R (1984) Developments of a water-maze procedure for studying spatial learning in the rat. *J Neurosci Methods* 11 (1): 47-60.
- Olten, D.S., & Samuelson, R.J. (1976) Remembrance of places passed: Spatial memory in rats. *Journal of Experimental Psychology: Animal Behavior Processes*, 2, 97-116.
- O'Keefe J, Dostrovsky J. (1971) The hippocampus as a spatial map. Preliminary evidence from unit activity in the freely-moving rat. *Brain Res.* 34(1):171-5.
- O'Keefe J, Nadel L. (1978) *The Hippocampus as a Cognitive Map.* Oxford University Press.
- Sutton, Richard S.; Andrew G. Barto (1998) *Reinforcement Learning: An Introduction.* MIT Press.
- Strosslin T, Sheynikhovich D, Chavarriaga R, Gerstner W. (2005) Robust self-localisation and navigation based on hippocampal place cells. *Neural Netw.* 18(9):1125-40. Epub 2005 Nov 2.
- Tolman EC. (1948) Cognitive Maps in Rats and Man. *Psychological Review* 55: 189-208.
- Trullier O, Meyer JA. (2000) Animat navigation using a cognitive graph. *Biol Cybern.* 83:271-85.
- Wilson R.A., Keil F.C. (1999) *The MIT encyclopedia of the cognitive sciences.* MIT Press
- Wyss R, König P, Verschure PF. (2006) A model of the ventral visual system based on temporal stability and local memory. *PLoS Biol.* ;4(5):e120.