COBOSLAB
Cognitive Bodyspaces: Learning and Behavior

Technical Report No. CoboslabY2010N001
March 18, 2010

# Motivated TGNG: Algorithm and Performance Evaluations

Martin V. Butz & Kevin Reif

COBOSLAB, Department of Psychology
University of Würzburg
Röntgenring 11
97070 Würzburg, Germany
http://www.coboslab.psychologie.uni-wuerzburg.de

# Motivated TGNG: Algorithm and Performance Evaluations

Martin V. Butz[*] Kevin Reif[†]

**Abstract**

This technical report describes the underlying algorithms of the Motivated Time Growing Neural Gas architecture and gives a detailed overview over the parameters used and their effects on the architecture's performance. An in-depth discussion of Motivated TGNG can be found in (Butz, Reif, & Shirinov, in revision).

## 1 Introduction

Based on our Time Growing Neural Gas algorithm (Butz, Reif, & Herbort, 2008), we extended the existing architecture to include a motivational module (Shirinov & Butz, 2009; Butz et al., in revision). This module allows an animat (1) to learn a sensorimotor cognitive map self-motivatedly and (2) to use it to generate flexible goal-directed behavior based on internal curiosity, fear, and hunger drives.

In this report we first give a detailed algorithmic description of the architecture's TGNG component as well as the new motivational module. We further outline the basic parameter setting used in our performance evaluations. Finally, we analyze different parameter dependencies and their effects on the animat's goal reaching performance and its choice of path through different maze environments.

In our experiments we assessed the animat's performance in five different mazes ranging in their complexity from very simple (an empty room) to very demanding (an intricate maze with teleporters). However, if not stated otherwise, the results for the different mazes were very similar. Therefore, in this report we only discuss the graphs for the first (complex) maze as a representative example.

[*] butz@psychologie.uni-wuerzburg.de
[†] kevin.reif@stud-mail.uni-wuerzburg.de

## 2 Algorithmic Description of Motivated TGNG

In this section, we give a comprehensive overview of the Motivated TGNG algorithm. The algorithmic description of Motivated TGNG is shown in Alg. 1 and Alg. 2.

### 2.1 TGNG Part

Alg. 1 mainly specifies the cognitive map learning aspect of the algorithm. After initializing the motivational drives and the empty list of nodes for the hunger drive, the main loop of the Motivated TGNG animat is activated (steps 1-4).

Each iteration, the current sensation vector $\vec{s}$ and the executed movement $\vec{m'}$ are perceived. The previous winner is set to $W_2$, and a new current winner node $W_1$ is determined, which is the node with the sensory vector $\vec{s_W}$ closest to the current sensation using the Euclidean distance (step 5-6). Next, the global error $\epsilon$ is updated based on the Euclidean difference between $\vec{s_{W_1}}$ and $\vec{s}$ (step 7).

$$\epsilon \leftarrow \epsilon(1-\delta) + \delta\sqrt{\sum_i (s_i - s_{W_1 i})^2}, \tag{1}$$

If the global error exceeds the vigilance threshold $\theta_\epsilon$, then the movement is recorded in the previous winner (step 9) and $W_1$ is replaced with a new node, whose center is set to the current sensation. Moreover, an edge is created that connects the previous winner $W_2$ with the new $W_1$ and stores the last executed motor vector (steps 10-11). Moreover, the global error is rest to zero (step 12).

If the global error threshold was not exceeded in this iteration but a new winner was determined, then the movement is also recorded in the previous winner (step 14) and an edge is either created if none existed so far, or the existing edge is updated setting its age to zero and adjusting its motor vector (steps 15-19). Updating the motor vector and experience of an edge is realized as follows:

$$\begin{aligned} \vec{m_E} &\leftarrow \begin{cases} \frac{e_E \cdot \vec{m_E} + \vec{m}}{e_E + 1} & \text{if } e_E < \frac{1}{\beta_m} \\ \frac{\vec{m_E} + \vec{m} \cdot \beta_m}{1 + \beta_m} & \text{otherwise} \end{cases}, & (2) \\ e_E &\leftarrow e_E + 1, & (3) \end{aligned}$$

Moreover, the center of the new winning node is adjusted towards the current perception (step 20) and the ages of all edges that depart from the new winner node are increased by one (step 21). If one of those ages reaches the maximum age $\theta_a$, then the edge is deleted. Given a node is consequently not connected to any other node any longer, also the node is deleted (step 22).

To inhibit apparently unreliable connections, if the edge that connects $W_2$ to $W_1$ was actually not the desired transition, then activity flow through this edge is inhibited for $\iota_E$ iterations (steps 23-25). Moreover, to prevent node circling, recently visited nodes are inhibited with a linearly decreasing strength for the next $\iota_N$ iterations (step 26).

After the adaptions due to the motivation module and the activity propagations through the network (step 28—see details below), the best action is chosen to execute given $W_1$ (step 29). Here, either a directional action is executed that has been executed less than $\theta_c$ times in this node before given the node's histogram of already executed actions, or the action in that edge is chosen that connects $W_1$ with the neighboring node that shows strongest activity.

## 2.2 Motivational Module

The functionality of the motivational module is shown in Alg. 2. The module first updates the hunger motivation. Given the presence of a food item, the food is consumed and the hunger reservoir filled (step 2). If the memory list of hunger-related nodes already contains the currently active node, then the inhibition level of the node is reset to zero, otherwise the node is added to the list with a zero inhibition level (steps 3-7). Given no food was encountered, the hunger reservoir is decreased according to

$$\sigma_h \leftarrow \max(\sigma_h - v_h; 0), \tag{4}$$

(step 9) where $v_h$ is the amount of decrease (set to $v_h = 0.002$ in our simulations) and the memory list of the hunger motivation is checked for a corresponding node, which is inhibited if present (steps 10-12). Independent of if food was encountered, the inhibition values of all entries in the memory list are slightly relieved (step 14).

Next, the security drive is updated given the current sensation $\vec{s}$ according to

$$\xi = \max(1 - \text{dist}_{min} \cdot d_\xi; 0), \tag{5}$$

(step 15) where $d_\xi$ (set to $d_\xi = 0.0045$) is a dimensioning parameter, which scales the sum of five distances to an appropriate range. The activations of the motivational drives are induced onto the TGNG network and are propagated for one iteration throughout the network (steps 16-17). Finally, the propagated activity values are combined

$$v_i = \frac{w_h \cdot h_i + w_f \cdot f_i + w_c \cdot c_i}{w_h + w_f + w_c}, \tag{6}$$

**Algorithm 1** Motivated TGNG

```
1: Initialize motivational drives and create an empty activation list for the hunger drive.
2: Start with one node $N$, setting its center to the system's initial sensations $\vec{s}$.
3: Set both winner nodes ($W_1$ and $W_2$) to $N$.
4: while not externally terminated do
5:     Perceive current sensation $\vec{s}$ and perceived movement $\vec{m'}$.
6:     Set previous winner to $W_2$ and determine the new winner node $W_1$ given $\vec{s}$.
7:     Update the global error $\epsilon$ (Eq. 1) based on $\vec{s}$ and the center $\vec{s_{W_1}}$.
8:     if global error $\epsilon$ exceeds threshold $\theta_\epsilon$ then
9:         Add executed movement $\vec{m}$ to histogram of node $W_2$
10:        Create a new node setting it to $W_1$ with its center at the current sensation $\vec{s}$.
11:        Create a new edge $E$ from $W_2$ to $W_1$ with motor vector $\vec{m_E} = \vec{m'}$.
12:        Reset the global error $\epsilon$ to 0.
13:    else if $W_1 \neq W_2$ then
14:        Add executed movement $\vec{m}$ to histogram of node $W_2$
15:        if Edge $E$ exists from $W_2$ to $W_1$ then
16:            Update edge $E$ by setting its age $a_E = 0$, increasing its experience $e_E$, and updating its motor vector $\vec{m_E}$ given $\vec{m'}$ (Eq. 2).
17:        else
18:            Create a new edge $E$ from $W_2$ to $W_1$ with motor vector $\vec{m_E} = \vec{m'}$.
19:        end if
20:        Move the center $\vec{s_{W_1}}$ towards the input signal $\vec{s}$ by fraction $\varepsilon_w$.
21:        Increment the ages $a_E$ of all edges $E$ emanating $W_1$.
22:        Delete all edges older than maximum age $\theta_a$ and delete nodes without any emanating edges.
23:        if Edge $E$ from $W_2$ to $W_1$ was not the desired transition then
24:            Inhibit activity transitions through $E$ for $\iota_E$ iterations.
25:        end if
26:        Inhibit activity propagation in $W_1$ linearly decreasing over the next $\iota_N$ iterations.
27:    end if
28:    UPDATE Motivation Module given $W_1$.
29:    Choose and execute the best movement direction $\vec{m}$ given current winner $W_1$.
30: end while
```

(step 18) where $w_h$, $w_f$, and $w_c$ reflect the respective strengths of the hunger, fear, and curiosity drives. State value $v_i$ denotes the resulting internal activity of a node, which is used for the determination of goal-directed behavior, as specified above. As a result, the external motivation-dependent activations are iteratively kept up-to-date while the propagated activations continuously change over time given the up-to-date external activations.

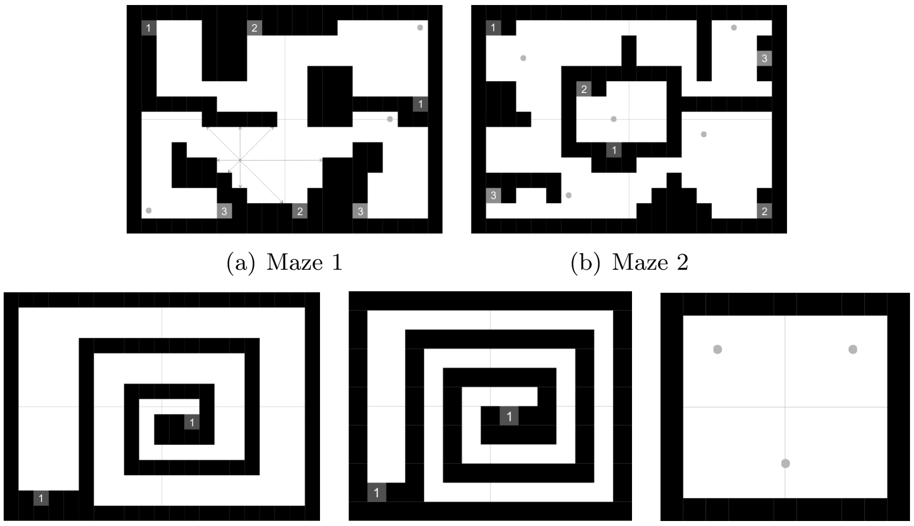
**Algorithm 2** UPDATE Motivation Module

```
1: if food is located within a certain range from the current position then
2:   Reset hunger reservoir to 1.
3:   if the hunger memory list already contains an entry $x$ for node $W_1$ then
4:     Reset inhibition level $\iota_{Rx}$ of list entry $x$ to zero.
5:   else
6:     Create a new entry $x$ in the activation list linking it to node $W_1$.
7:   end if
8: else
9:   Decrease the hunger reservoir value.
10:  if node $W_1$ is linked in a memory list entry $x$ then
11:    Set the inhibition level $\iota_{Rx} = 0$.
12:  end if
13: end if
14: Decrease the inhibition level $\iota_{Rx}$ of each hunger activation list entry $x$ by multiplying it with the relief rate $\rho_\iota$.
15: Update the internal security drive given current sensation $\vec{s}$.
16: Apply external activation patterns to the neural network.
17: Perform a step of activity propagations for each node in the neural net for all three motivation-dependent activities.
18: Combine the propagated activity values into one overall activity in each node.
```

## 3 Environment

The environment used in our experiments is a two-dimensional continuous maze environment that consists of squared sub-areas. These subareas can be either *empty*, *solid*, or *teleporter* areas. The agent moves within and between empty areas. Solid areas prevent passage, leading the agent along the wall of the blocked area. Teleporter areas come in pairs. Once the animat enters a teleporter area, it is immediately transferred to the corresponding position at the open side of the paired teleporter area. Food sources are placed in the environment where the hunger reservoir is filled once entering the food area, which is four units in radius. After food consumption, which is simply realized by entering the food area, the food is removed. It reappears at the same location after 1000 time-steps.

The size of the mazes was set such that each empty or filled squared subarea has a side length of 10 (continuous) units. Fig. 1 shows the different mazes used in our evaluations. The teleporters are introduced to show that TGNG is neither dependent on a sufficient sensory proximity given motor proximity for successful learning nor is the system restricted to learn in Euclidean (e.g. metrical 2D or 3D) spaces.



(a) Maze 1 (b) Maze 2

(c) Maze 3 (d) Maze 4 (e) Maze 5

**Figure 1:** The five maze environments used in our goal-reaching evaluations. Teleporter pairs are marked with numbers. Light brown circles indicate food sources, where applicable.

## 4 Parameter Settings

For the TGNG learning algorithm, we use the following parameter settings:

$$\delta = 0.5, \theta_\epsilon = 5, \beta_m = 0.1,$$

$$\varepsilon_w = 0.05, \varepsilon_n = 0, \theta_a = 100, p = 0.2$$

For the motivation module and the activity propagations, the parameter settings were as follows (unless stated otherwise):

$$\gamma = 0.99, \beta_a = 10, \beta_c = 0.5, d_\xi = 0.0045, \theta_\xi = 0.5, \rho_\iota = 0.99995,$$

$$p_h = 1, p_f = 0.5, p_c = 0.1, v_h = 0.002, v_f = 0.01, v_c = 0.001, N_c = 8, \theta_c = 1.$$

Note that the hunger priority is double the size of the fear priority. This assures that the robot will always search for food first, provided that the hunger level is high enough and at least one food location is known.

## 5 Performance Study of Motivated TGNG

In this section we study and highlight the major parameter dependencies that Motivated TGNG depends on. First, however, we study the noise robustness of the system. In all graphs, we report average performance and normalized standard deviation values over ten independent runs, which were initialized with different random seeds. We focus on evaluations in Maze 1. However, similar results were achieved within the other mazes, if not stated differently.

### 5.1 Noise Robustness

As shown in Fig. 2 the architecture is very robust with regard to noise added to the executed motor vectors. Even noise levels of 2.5 units in standard deviation (std), which is 2.5 times higher than the actual step made, only have a slight impact on the animat's goal reaching performance. However, as expected, the animat deviates more and more from the ideal path with higher noise values.

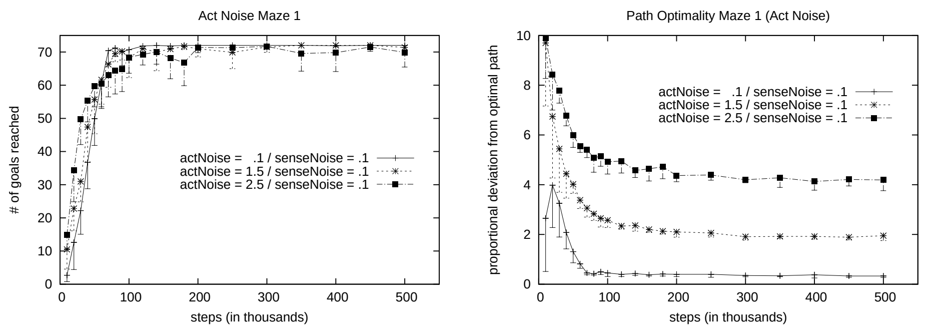


**Figure 2:** Increasing action noise makes the paths to the goals more noisy but does only marginally affect goal reaching performance.

Even though adding noise to the sensory inputs (as opposed to the motor vectors) of the animat has a smaller impact on path optimality (c.f. Fig. 3), higher noise levels delay learning, thus delaying and partially preventing the system from reaching all goals reliably. In addition, the size of the network (the number of edges) increases dramatically for noise levels higher than 1 unit in std resulting in an unacceptable high run time (c.f. Fig. 4, no standard deviations here since sensory noise with std=1.5 is an exemplary run). This is due to the fact that due to the high noise multiple nodes are needed to cover the same location in the maze, since the $\theta_\epsilon$ threshold will trigger node creations solely due to the high perceptual noise. Consequently, the number of edges between neighboring locations in the maze grows quadratically with the number of nodes needed to cover one location.

As shown in Fig. 5 also the simultaneous increase of both noise levels still

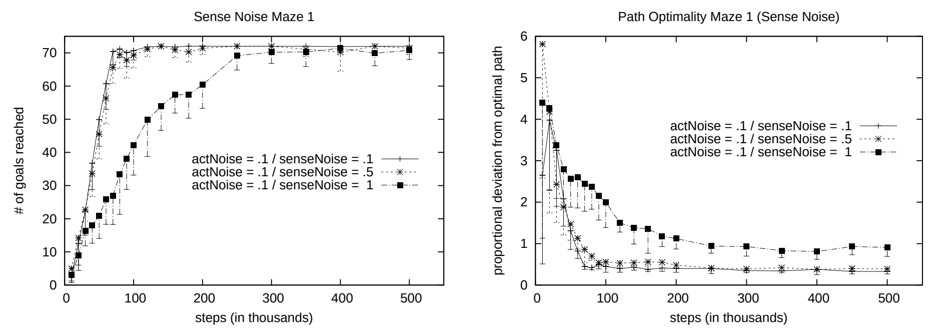


**Figure 3:** Increasing sensory noise delays learning but still yields robust goal reaching performances with good reasonable path qualities up to a certain degree.

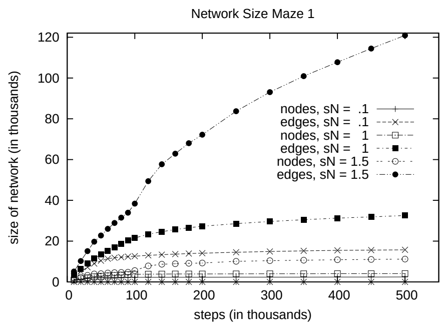


**Figure 4:** The number of nodes and particularly edges increases significantly once the sensory noise is often higher than the error threshold $\theta_\epsilon$.

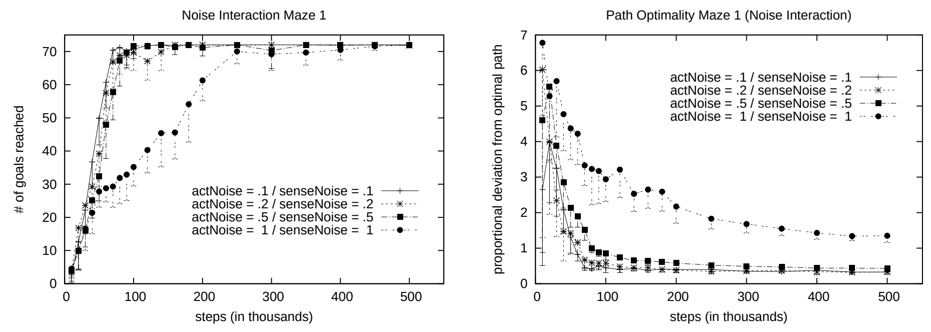


**Figure 5:** Sensory and action noised appear to have rather independent influences on the performance of Motivated TGNG.

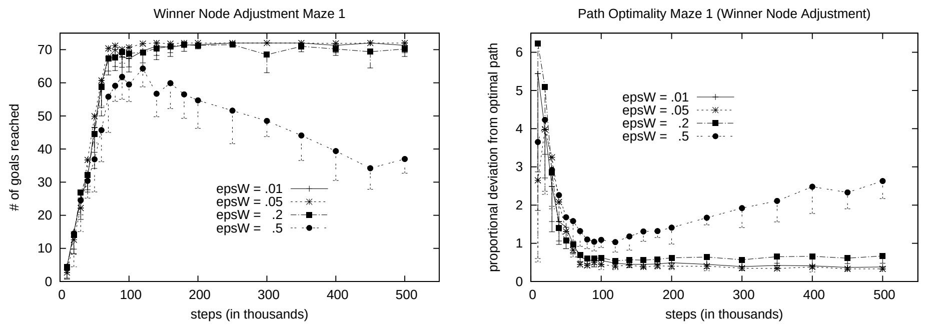


**Figure 6:** Only very strong node adjustments yield unstable system behavior.

yields very good results for values below 1 unit std. As the path optimality mainly depends on the noise level added to the motor vectors, it decreases in much the same way as without the additional noise in the animat's sensory input (c.f. Fig. 2).

## 5.2 Parameter Dependencies

### 5.2.1 Error Threshold $\theta_\epsilon$

The error threshold $\theta_\epsilon$ is mainly responsible for the network's granularity. Large values yield a very coarse distribution of nodes resulting in a faster learning of the environment's topology with the drawback of less efficient paths and possible aliasing errors for values too large considering the evaluated environment. Small values on the other hand lead to a high number of nodes and thus a finely grained network. This results in a delayed learning and a quickly increasing computational run time, however, it assures a very high success rate. (c.f. Butz et al. (in revision) for a more detailed evaluation of $\theta_\epsilon$).

### 5.2.2 Adaptation of Node Centers

With regard to the adjustment rate $\varepsilon_w$ of the current winner node towards the current input signal $\vec{s}$, Motivated TGNG performs very well for all values up to .2 considering both goal-reaching performance and path optimality (c.f. Fig. 6). Moving the current winner towards the input signal is essential to develop a balanced distribution of nodes. However, too strong adjustments result in a network that is never fully stable.

As for the winner's neighboring nodes, even slight adjustments with $\varepsilon_n > 0$ towards the current input signal $\vec{s}$ lead to drastic decreases in performance and path optimality as shown in Fig. 7. This also holds true for other complex maze configurations. Moving neighboring nodes towards the current input signal destabilizes the network since (1) only the nodes but

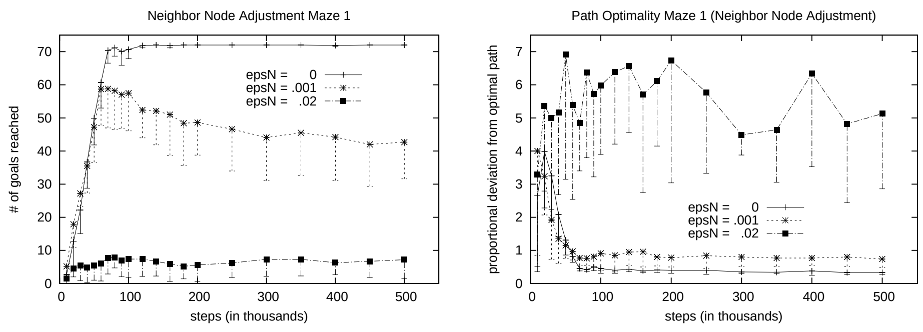


**Figure 7:** Adjustments of the centers of neighboring nodes have a strong negative effect on performance.

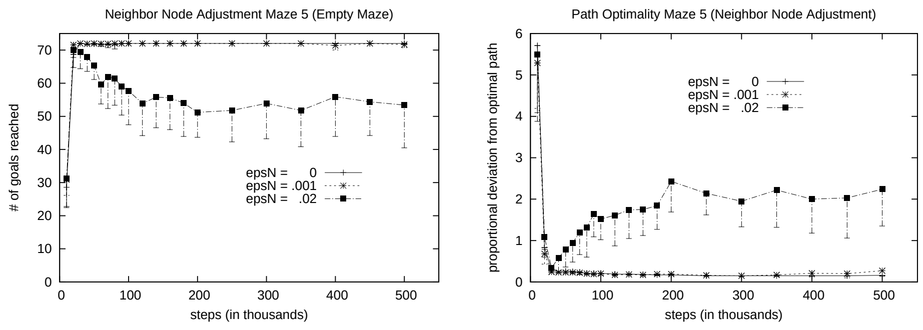
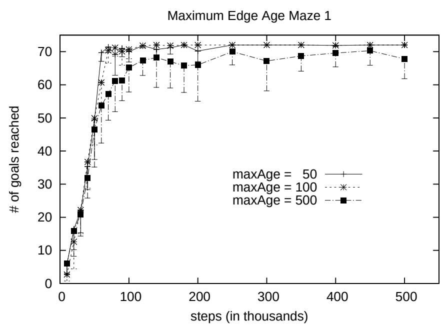


**Figure 8:** In the empty Maze 5, neighbor node adjustments do not have an equally negative effect as observed in Maze 1. However, also in Maze 5 updates of neighboring nodes do not yield any performance benefits.

not the stored motor vectors in the edges are adjusted and (2) a winner's neighboring nodes are not necessarily comparable with regard to the stored sensations as we create nodes based on proximity in time not space. Experiments with the empty maze (Maze 5), however, showed that for small adjustment values (up to $\varepsilon_n = 0.001$) performance and path optimality remain constant at an excellent level. This is due to the fact that in this empty maze there are no teleporters and no concave edges, so that there are no nodes whose immediate neighbors differ greatly in their stored sensations (cf. Fig. 8).

### 5.2.3 Edge Deletion

The results for different maximum ages of the network's edges suggest that values around 100 yield the best performance. However, the differences in performance and path optimality are rather small as can be seen in Fig. 9.



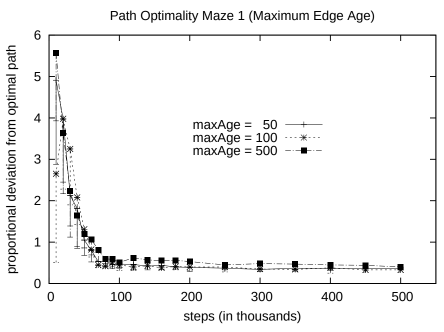
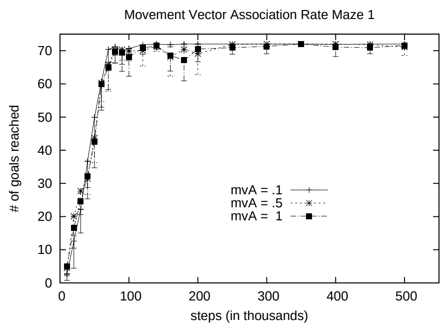


**Figure 9:** The maximum age $\theta_a$ for edge deletions has only a minor performance impact.



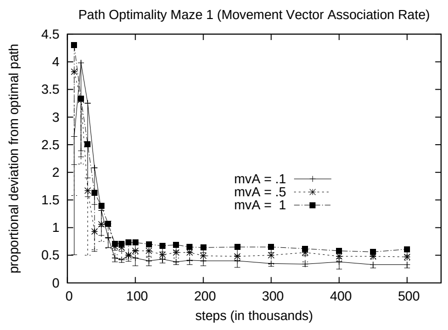


**Figure 10:** The movement vector update rate $\beta_m$ affects performance only marginally.

### 5.2.4 Movement Vector Adaptation Rate

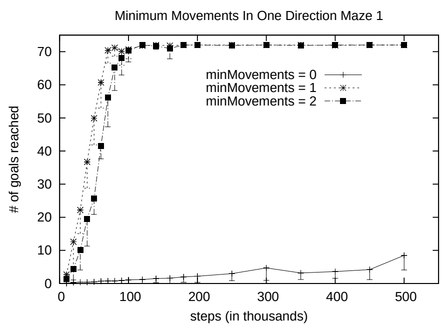
Different values for the movement vector association rate $\beta_m$ hardly yield any impact on the animat's performance and path optimality. Fig. 10 shows that a value of $\beta_m = .1$ yields the best performance among generally good results.

### 5.2.5 Curiosity Definition

Fig. 11 shows that it is crucial for the animat's performance to execute at least $\theta_c = 1$ movement in each direction from any given node before switching to normal activity-based action selection. Otherwise it is impossible to learn the layout of the maze and thus any goal positions reliably.

Increasing the directional partitions $N_c$ and thus the directions to be explored from every given node, results in a small delay in learning without improving the animat's path optimality as shown in Fig. 12. Since the system still reaches all 72 goals in both cases, an increase of $N_c$ is not necessary but only delays learning progress.

As with increasing the directional partitions, a higher time delay pro-



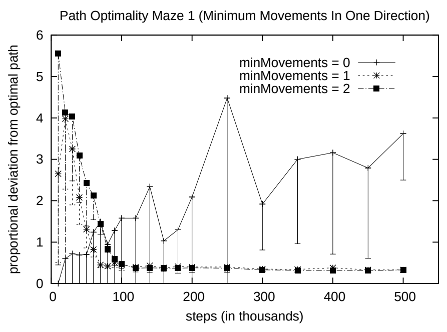
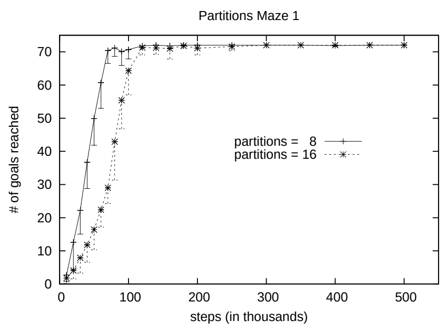


**Figure 11:** The initial curiosity based on unknown movement effects is highly important to ensure initial effective maze exploration.



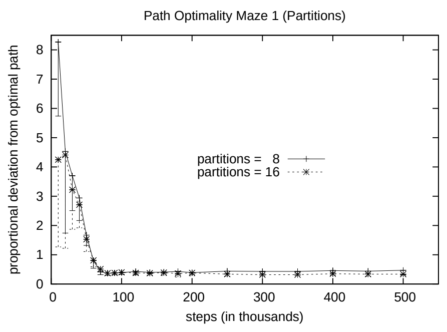


**Figure 12:** A larger directional partition of the movement histogram delays learning progress. Path optimality is only slightly improved.

portion $\beta_c$ in the animat's curiosity drive leads to a small learning delay but to no improvements with regard to path optimality. Again, performance converges to reaching all 72 goals reliably with a good path efficiency, as shown in Fig. 13.

### 5.2.6 Temporary Node and Edge Depression

As shown in Fig. 14, setting depression times for the network's nodes and edges to values $> 0$ increases the overall performance significantly, especially when both values are set to $\iota_N > 0$ and $\iota_E > 0$. Thus, both node and edge depressions are important to balance exploration and to prevent ineffective circling behavior. However, there is no best ratio, since all combinations of values $\iota_N >= 10$ and $\iota_E >= 100$ yield excellent results. The same holds true for the path optimality results.

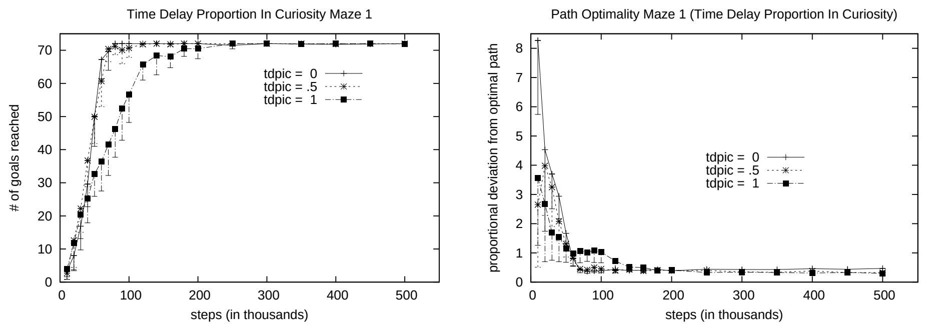


**Figure 13:** When increasing the influence of the time delay effect on curiosity (parameter $\beta_c$) learning speed decreases but still a similar performance level is reached ultimately. No time delay influence, on the other hand, slightly decreases path optimality later on.

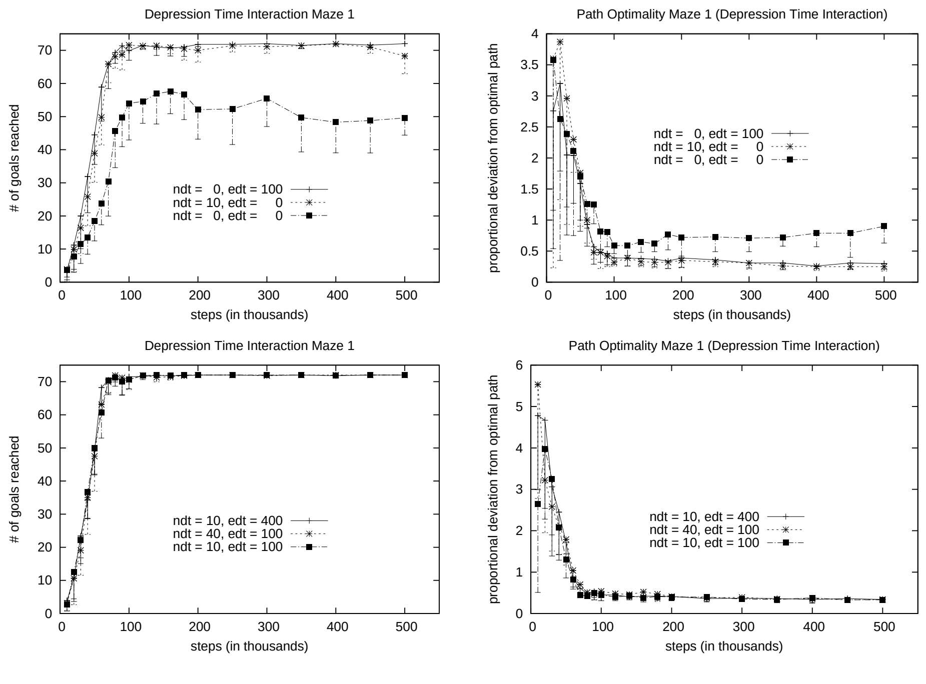


**Figure 14:** As long as both inhibition methods are activated ($\iota_N >= 10$ and $\iota_E >= 100$) Motivated TGNG does hardly ever get stuck in circles so that the cognitive map is learned well and consequently all goals are reached on an effective path.

## 6 Conclusion

As the experimental evaluations show, the architecture's performance remains largely constant for a rather wide range of different parameter values. In light of this robustness, it is relatively easy to find a parameter setting that yields optimal results when evaluating a new environment. Moreover, it shows that the underlying algorithms do not only allow the animat to develop goal directed behavior based on its inherent motivational drives, but to do so flexibly and largely independent of outer circumstances such as noise or different environmental layouts. We believe therefore that this architecture constitutes an excellent basis for future studies on self-motivated cognitive map learning and goal-directed behavioral control.

Finally, it should be emphasized that Motivated TGNG is not restricted to maze learning tasks. It can be expected to yield interesting behavioral patterns and effective goal directed behavioral performances in other environments in which different movements and sensations are possible as long as the sensations yield a Markov environment. For example, arm control tasks are imaginable in which the system perceives arm postures and connects these postures given appropriate arm motor commands. In our future research we intend to apply Motivated TGNG to such tasks.

## References

Butz, M. V., Reif, K., & Herbort, O. (2008). Bridging the gap: Learning sensorimotor-linked population codes for planning and motor control. *International Conference on Cognitive Systems*, *CogSys 2008*.

Butz, M. V., Reif, K., & Shirinov, E. (in revision). Self-organizing sensorimotor maps plus internal motivations yield animal-like behavior. *Adaptive Behavior*.

Shirinov, E., & Butz, M. V. (2009). Distinction between types of motivations: Emergent behavior with a neural, model-based reinforcement learning system. *2009 IEEE Symposium Series on Artificial Life (ALIFE 2009) Proceedings*, 69-76.