Adaptive Landmark-Based Navigation System Using Learning Techniques

Bassel Zeidan[1], Sakyasingha Dasgupta[2], Florentin Wörgötter[2], and Poramate Manoonpong[2,3]

[1] Faculty of Mathematics and Computer Science, Institute of Computer Science, University of Göttingen, D-37077 Göttingen, Germany
[2] Bernstein Center for Computational Neuroscience (BCCN) University of Göttingen, D-37077 Göttingen, Germany
[3] The Mærsk Mc-Kinney Møller Institute, University of Southern Denmark, 5230 Odense M, Denmark
`bassel.zeidan@gmail.com;{dasgupta,worgott}@gwdg.de;poma@mmmi.sdu.dk`

Abstract. The goal-directed navigational ability of animals is an essential prerequisite for them to survive. They can learn to navigate to a distal goal in a complex environment. During this long-distance navigation, they exploit environmental features, like landmarks, to guide them towards their goal. Inspired by this, we develop an adaptive landmark-based navigation system based on sequential reinforcement learning. In addition, correlation-based learning is also integrated into the system to improve learning performance. The proposed system has been applied to simulated simple wheeled and more complex hexapod robots. As a result, it allows the robots to successfully learn to navigate to distal goals in complex environments.

Keywords: Goal-directed behavior, Sequential reinforcement learning, Correlation based learning, Neural networks, Walking robots.

1 Introduction

Attempts to create autonomous robots that can move around and navigate toward a (distal) goal have been ongoing for over 20 years [12]. A lot of effective robotic navigation systems have been proposed. Some of them use internal representations (e.g. generalized voronoi diagrams [9], place cells [10], etc). Others use potential fields [11]. Reinforcement learning (RL) has been widely used for navigation learning [2,10]. The learning process of RL systems is guided by reward signals and based on a trial and error mechanism. This mechanism gives the system an adaptive property to cope with different situations and unexpected scenarios. However, RL systems require many learning trials to solve problems that have large state spaces.

In this paper we present an adaptive navigation system based on sequential reinforcement learning. It is inspired by animal navigation behavior where animals including gerbils [13] and ants exploit environmental features (landmarks) to find the right direction. The presented system treats environmental landmarks as subgoals that guide the navigation process. In contrast to other subgoal-based navigation systems [7,8], the presented system relies on exploiting the local property of radial basis function networks to solve subgoal-based tasks. These networks enable the system to perform such tasks efficiently in large continuous spaces while keeping the system structure simple. Furthermore, the special feature of our approach is the integration of correlation-based learning (ICO learning) into the system. ICO learning acts here as adaptive exploration which improves learning performance. The proposed system is tested on a simple wheeled robot and a more complex hexapod robot. It allows the robots to effectively navigate to distal goals in complex environments.

2 Sequential Learning Strategy

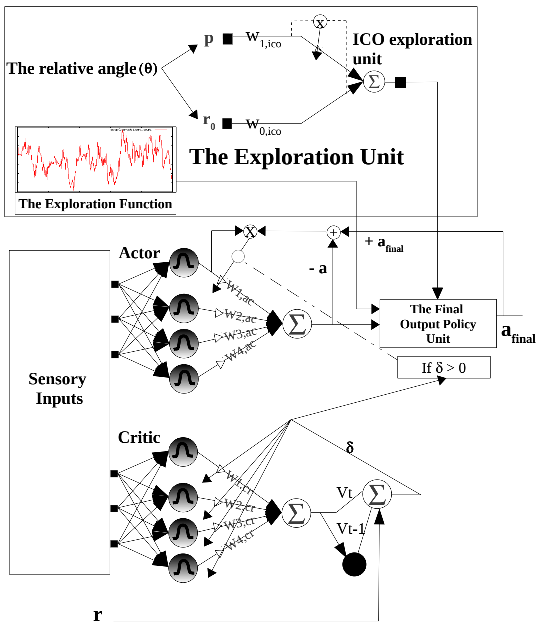
The presented navigation system operates based on a sequential learning algorithm which treats environmental landmarks as subgoals. By learning to reach these subgoals in the right order, the system learns an entire trajectory which leads to the final goal. This is done by performing a sequence of RL learning phases. The term "learning phase" is given to a learning process that enables a robot to move from one subgoal to the next. The robot receives a positive reward each time it reaches a subgoal in the correct order. This order is defined by the system designer. The system's behavior is checked after a fixed number of trials in a special test trial in which no exploration signals are produced to test the learned policy. The policy improvements stop and the exploration signals are turned off after a test trial for each learning phase that its related subgoal was reached in the right order during the test trial (see Algorithm 2).

Using a normal RL system is insufficient for sequential learning. This is because it uses the same representation of the state space for different learning phases. This causes the system during a learning phase to overwrite what has been learned in other learning phases. Our approach to overcome this problem is to feed the index of the current targeted subgoal as an additional input called the Subgoal Definer (SD) input to the RL system. This enables the system to become aware of the change that happens after reaching a subgoal. One dimension in the input space is sufficient to enable the system to cope with any number of subgoals. Using this additional input gives a better representation of the state space where the same state of the robot in the environment is represented differently for different learning phases.

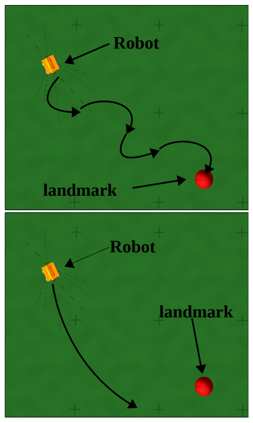
3 Adaptive Landmark-Based Navigation System

The learning process of the presented navigation system is an actor-critic method [1], a special type of temporal difference (TD) RL. This method has an ability to produce a smooth control signal because of its ability to handle continuous action spaces. In addition, it is based on a biological learning model [6].

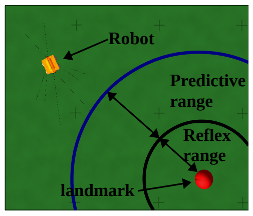
The proposed system consists of the following four units: 1) the actor 2) the critic 3) the exploration unit 4) the final output policy unit. (see Fig. 1a).



(a)



(b)



(c)

Fig. 1: (a) The proposed system's structure and units. The TD error effect on the system's structure. (b) The upper/lower picture shows the effect of using an insufficient big/small value respectively for the exploration unit's parameter ($w_{0,ico}$). (c) The input signals' ranges of the ICO exploration system.

3.1 The Actor and Critic

The actor is responsible for the policy. The critic estimates the expected total payoff (V) value. During learning, the TD error drives the learning process by guiding the actor's policy and improving the critic's estimation. The TD error is calculated by:

$$\delta_t = r_t + (\gamma V_t) - V_{t-1}, \qquad (1)$$

where δ is the TD error. γ is the discount factor $\in$ [0, 1]. V is the expected total payoff value (V value). r_t is the received reward at time step t.

Radial basis function (RBF) networks are used as function approximators to give the system the ability to handle continuous spaces and to produce a smooth control signal. Each RBF network consists of two layers. The first layer consists of hidden neurons that have radial basis activation functions. The second layer consists of output neurons. The output of each output neuron is calculated as the weighted linear combination of the hidden neurons' outputs and given by:

$$y_j(x(t)) = \sum_{i=1}^{H} w_{i,j} v_i(x(t)), \tag{2}$$

where y_j is the output of the j-th output neuron. x(t) is the presented input signal at time step t. $w_{i,j}$ is the weight that connects the i-th hidden neuron with the j-th output neuron. H is the number of hidden neurons. v_i is the output of the i-th hidden neuron which is constructed from the normalized Gaussian basis functions as:

$$v_i(x(t)) = \frac{a_i(x(t))}{\sum\limits_{j=1}^{H} a_j(x(t))}, \tag{3}$$

$$a_i(x(t)) = \mathrm{e}^{-||S_i^T(x(t)-C_i)||^2}. \tag{4}$$

S_i is the diagonal matrix of the inverse covariance of the RBF neural network. C_i is the function center of the i-th hidden neuron.

The RBF centers and variances don't get modified during learning. However, the network's weights get updated according to a specific learning rule. Updating a certain weight connected to a hidden neuron will have an effect on a local area in the input space that is dominated by that neuron. Because of this local property, our system is able to perform multiple learning phases by using only one RL structure. RBF networks are used to implement the actor and the critic. Using nonlinear approximators for both enables the system to cope with any environment configurations. The weights of the critic RBF network are updated by [1]:

$$\frac{dw_{i,cr}(t)}{dt} = \lambda \delta v_{i,cr}(x(t)), i = 1, 2, 3, ..M \tag{5}$$

where λ is the critic learning rate. $v_{i,cr}(x(t))$ is the output of the critic network's i-th hidden neuron. M is the number of hidden neurons in the network.

The actor improves the policy by learning previous taken actions that yielded good outcomes. A class of algorithms called CACLA is used to update the actor [3]. CACLA helps RL systems to perform stable learning in continuous action spaces with a good convergence property. It exhibits better performance in comparison to other update rules. The actor network's weights are updated by:

$$\frac{dw_{i,ac}(t)}{dt} = \eta(a_{final,t} - a_t) v_{i,ac}(x(t)), i = 1, 2, 3, ..N \tag{6}$$

where η is the actor learning rate $\in$ [0,1]. $a_{final,t}$ is the actual (final) action value of the system at time step t. a_t is the actor's output (action value) at time step t. $v_{i,ac}(x(t))$ is the output of the actor network's i-th hidden neuron. N is the number of hidden neurons in the actor RBF network.

The goodness of a taken action can be assessed by checking the TD error sign at the next time step. Here, a positive TD error indicates that the previously taken action was good while a negative TD error indicates that it was bad. This way, the actor learns towards a taken action only if the TD error is positive. On the other hand, the previously taken action is ignored if the TD error is negative.

3.2 The Exploration Unit

This unit is responsible for producing an exploration signal. This signal is combined with the actor's output during learning to produce the system's final output (action value). It allows the system to discover new areas in the state space. The function shown in Eq. 7 is used to produce the exploration signal [1].

$$\epsilon_t(v) = \zeta \Phi_t min[1, max[0, \frac{V_{max} - V_t}{V_{max} - V_{min}}]], \tag{7}$$

where $\epsilon_t(v)$ is the exploration signal at time step t. ζ is a scale factor. Φ_t is a Gaussian distribution noise with the mean of zero and the standard deviation of one. V_t is the V value at time step t. V_{max} and V_{min} are the maximum and minimum observed V values and they are assigned dynamically during learning.

A new exploration unit is integrated into the system to improve the performance. This unit is added based on the idea of using landmarks as attraction locations since they lead to the right direction. It produces a continuous signal which pulls the robot towards a nearby landmark if it is located within a certain range. Correlation-based learning, called input correlation learning (ICO learning) [4], is used to implement this unit. ICO exploration unit takes two input signals: a predictive signal and a reflex signal and both are the relative angle between the robot and the nearby landmark. However, they are received at different times. The predictive signal is received first then the reflex signal comes later (see Fig. 1c). The output of the ICO exploration unit is given by:

$$\epsilon_{ico} = p w_{1,ico} + r_0 w_{0,ico}, \tag{8}$$

where p is the predictive input. $w_{1,ico}$ is the plastic synapse of the predictive input. r_0 is the reflex input. $w_{0,ico}$ is the synaptic strength of the reflex input.

ICO learning correlates between these two input signals which gives the system after learning the ability to predict the reflex signal before it happens and to prevent it from happening. $w_{1,ico}$ changes according to the ICO learning rule:

$$\frac{dw_{1,ico}(t)}{dt} = \mu p \frac{dr_0(t)}{dt}, \tag{9}$$

where μ is the learning rate $\in$ [0, 1]. The learning stops when the system is able to pose the robot directly towards a nearby landmark within the predictive range. The robot enters the reflex range after learning with the relative angle of zero degree from the landmark to the robot (no learning occurs).

The reflex signal represented by $r_0 w_{0,ico}$ in Eq. 8 is able by itself to produce this behavior. However, this is possible only if $w_{0,ico}$ is tuned carefully. The tuning should enable the system to produce an output which is suitable for the range of the input control signal and also for the frequency with which it is fed to the robot. Choosing a bad value for $w_{0,ico}$ causes undesirable behaviors (see Fig. 1b). Tuning this parameter requires detailed information about the robot's structure. In addition, fixing it causes the loss of the adaptivity property and the lack of flexibility against changes. ICO learning provides a powerful mechanism to tune the system. $w_{0,ico}$ is fixed to a constant value[1] and $w_{1,ico}$ is modified by ICO learning until it converges to an optimal value which generates the desired behavior while avoiding bad situations.

This unit could pull the robot blindly towards a wrong landmark because it is not able to assess the quality of performing this behavior [2]. On the other hand, the RL system assesses each action before it gets learned by the actor. The assessment process is done using the received TD error. Based on that and since the ICO system learns faster than the RL system, the unit's output signal is used as an additional exploration signal which aids the system during learning. However, it doesn't contribute in the final output of the system after learning.

3.3 The Final Output Policy Unit

The way the exploration signal is being used has a profound impact on the performance. This unit combines other units' outputs to produce the final output (action value) of the system. The final output is determined as shown in Algorithm 1. The algorithm produces the final output during learning. However, after learning the actor's output is the only one considered as a final output.

Algorithm 1: Determine the final output of the system

```
if a landmark is in the predictive or reflex range then $a_{final,t} \leftarrow \epsilon_{ico,t}$
else if the trial number mod 2 = 0 then $a_{final,t} \leftarrow \epsilon_t(v) + a_t$
else $a_{final,t} \leftarrow \epsilon_t(v)$
return $a_{final,t}$
```

4 Experimental Results

The proposed system has been applied to two simulated robots: the wheeled robot NIMM [2] and the hexapod robot AMOS [5].

4.1 Experiment 1: NIMM in an Environment with Three Subgoals

The goal of this experiment is to investigate the system's efficiency and to assess the benefit of using the ICO exploration unit. The robot should learn to reach two subgoals (1 and 2) and the goal (3) in the right order sequentially (see Fig. 2*a*). The robot receives +1 reward at each subgoal if it is reached in the correct order. The robot receives four input signals: the relative angle from the three (sub)goals to the robot and the SD input. The system produces one control signal to control the robot. The sign and the amplitude of the control signal

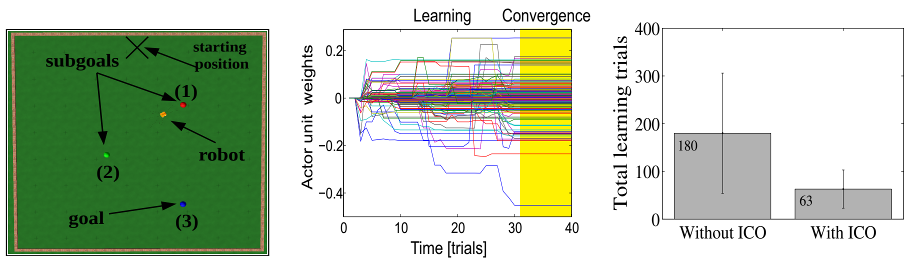
[1] Based on the range of the input control signal, the system designer specifies a reasonable value for $w_{0,ico}$. Even if this value is not optimal and causes undesirable reflex behavior, this will not affect the ICO system final output after learning. This is because this reflex signal will be eliminated when the ICO learning process ends.

Algorithm 2: Sequential reinforcement learning algorithm

```
Initialization: the exploration is ON for all learning phases.
set $w_{i,ac}$, $w_{i,cr}$ and $w_{1,ico}$ to 0.0
repeat
    $SD \leftarrow 0$
    repeat
        $x(t) \leftarrow$ sensory inputs & SD
        if a test trial OR
           the exploration is OFF for the current learning phase then
            $a_{final,t} \leftarrow a_t \leftarrow \sum_{i=1}^{N} w_{i,ac} v_{i,ac}(x(t))$
        else
            $a_t \leftarrow \sum_{i=1}^{N} w_{i,ac} v_{i,ac}(x(t))$, $V_t \leftarrow \sum_{i=1}^{M} w_{i,cr} v_{i,cr}(x(t))$
            $\epsilon_t(v) \leftarrow \zeta \Phi_t min[1, max[0, \frac{V_{max}-V_t}{V_{max}-V_{min}}]]$, $\epsilon_{ico,t} \leftarrow p w_{1,ico} + r_0 w_{0,ico}$
            if a landmark is in the reflex range then
                $w_{1,ico} \leftarrow w_{1,ico} + \mu p (r_{0,t} - r_{0,t-1})$
            $a_{final,t} =$ determine the final output of the system (Algorithm 1)
            $\delta_t = r_t + (\gamma V_t) - V_{t-1}$
            $w_{i,cr} = w_{i,cr} + \lambda \delta_t v_{i,cr}(x(t-1))$
            if $\delta_t > 0$ then $w_{i,ac} = w_{i,ac} + \eta(a_{final,t-1} - a_{t-1}) v_{i,ac}(x(t-1))$
        if the next subgoal is reached then $SD \leftarrow SD + 1$
    until the termination of the current trial
    if a test trial then turn the exploration off for all achieved learning phases
until the actor's policy converges to an optimal one (i.e. the exploration is off
for all learning phases)
```

determine the steering direction (left or right) and the steering angle of the robot, respectively. The number of hidden neurons in the actor and critic RBF networks is set to 625. The number of bases on each input dimension of the actor and critic is set to 5. The width of the Gaussian basis functions is set to twice the distance between its center and the center of its nearest neighbor. The actor learning rate η is set to 0.02 and the critic learning rate λ to 0.05. The discount factor γ is set to 0.9999. For the ICO learning parameters, the ICO learning rate μ, the predictive range and the reflex range are set to 0.4, 0.06 and 0.023, respectively. For the exploration function parameters, the scale factor ζ is set to 0.5. V_{min} is set to 0 and V_{max} is assigned dynamically during learning.

The results shown in Figs. 2*b and* 2*c* show that the system is reliable and effective to allow the robot to perform the sequential navigation task. It can be observed in Fig. 2*c* that using the ICO unit has a significant impact on the system performance. Based on this result, this unit will be integrated into the system and used as an essential element of the system for any further experiments. The



(a) (b) (c)

Fig. 2: (a) The set up of experiment 1. (b) The actor RBF network's weights during learning until the policy converges to an optimal one after 32 trials. (c) The average number of trials needed until an optimal policy is obtained. The experiment was performed 60 times. In 30 of them, the system operates without the ICO exploration unit and 30 of them with it.

speedup factor of using the ICO exploration unit is calculated as follows:

$$The\ speedup\ factor = \frac{learning\ time\ (trials)\ without\ ICO}{learning\ time\ (trials)\ with\ ICO} = \frac{180}{63} = 2.85$$

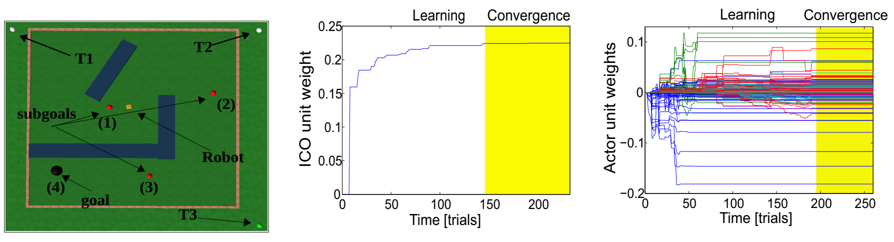
4.2 Experiment 2: NIMM Robot in a Complex Environment

In this experiment a more complex environment is used where the subgoals are placed far from each other. In addition, obstacles are installed between them to make them harder to be reached. The robot should learn to reach all subgoals (1, 2 and 3) and the goal (4) in the right order (see Fig. 3*a*). The robot gets rewarded with +1 at each subgoal if it is reached in the correct order and with -1 if it hits an obstacle. The robot detects obstacles using its infrared sensors. The sensors' signals are not used as inputs to the system. However, when the sensors are activated, a negative reward signal is sent to the system. This usage of the sensors' signals is sufficient to enable the system to achieve the task. The system produces one control signal to control the robot and it receives four input signals: two inputs as the distance between the robot and the two transmitters T1 and T2. One input as the relative angle from T3 to the robot and the SD input. Determining the type of the input signals and the locations of the signal transmitters is not arbitrary. These signals provide for the system a representation of the robot's states in the environment. To perform a successful learning process, this representation must be sufficient to cover all states and it must be unique for different states. In the presented experiments, these conditions are fulfilled. The number of hidden neurons in the actor and critic RBF networks is set to 875. The number of bases on each input dimension of the actor and critic is set to 5, 5, 5 and 7, respectively. The width of the Gaussian basis functions is set to twice the distance between its center and the center of its nearest neighbor. The actor learning rate η is set to 0.025 and the critic learning rate λ to

0.06. The discount factor γ is set to 0.9999. For the ICO learning parameters, the ICO learning rate μ, the predictive range and the reflex range are set to 0.6, 0.055 and 0.025, respectively. The exploration function is modified to produce more curvy exploration trajectories as follows:

$$\epsilon_t(v) = \zeta \Phi_t(min[1, max[0, \frac{V_{max} - V_t}{V_{max} - V_{min}}]])^2, \tag{10}$$

The scale factor ζ is set to 23. V_{min} and V_{max} are assigned during learning. As observed from Figs. 3*b and* 3*c* the system successfully enabled the robot to perform the navigation task even in a complex environment.

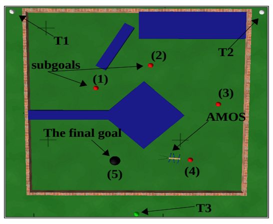


(a) (b) (c)

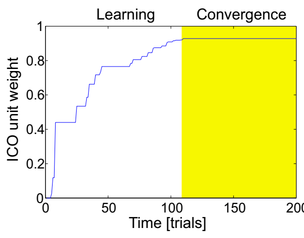
Fig. 3: (a) The set up of experiment 2. (b) The ICO exploration unit weight during learning. It converges to an optimal value after 144 trials. (c) The actor RBF network's weights during learning until the policy converges to an optimal one after 183 trials. The average number of trials that are needed until an optimal policy is obtained is 258±161. A video of this experiment can be seen at [http://www.manoonpong.com/SAB2014/SVideo1.mpg]. In addition, another experiment for long-distance navigation learning can be also seen at [http://www.manoonpong.com/SAB2014/SVideo2.mpg].

4.3 Experiment 3: AMOS in a Multiple-goal Environment

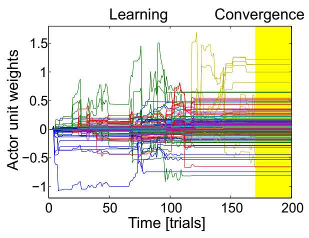
The proposed system is used to control the simulated hexapod robot AMOS [5]. AMOS's walking ability relies on the movements of its six legs which are controlled by signals received at the legs' joints and produced by a two neurons oscillator (CPG)[5]. The presented system produces one output signal. The sign and the amplitude of this signal determine the steering direction (left or right) and the steering angle of the robot, respectively. This signal is used to modify the amplitude of the oscillator's signals and thus it enables AMOS to turn left or right. The robot should learn to reach all subgoal and the goal in the right order which is 1, 2, 3, 4, 5 (see Fig. 4*a*) The robot receives four input signals: two inputs as the distance between the robot and the two transmitters T1 and T2. One input as the relative angle from T3 to the robot and the SD input. The robot receives +1 reward at each subgoal if it is reached in the correct order and with -1 if the robot hits an obstacle. The robot detects obstacles using its ultrasonic sensors. The number of hidden neurons in the actor and critic RBF networks is set to 1512. The number of bases on each input dimension of the actor and critic is set to 6, 6, 6 and 7 respectively. The width of the Gaussian basis functions is set to twice the distance between its center and the center of its nearest neighbor. The actor learning rate η is set to 0.03 and the critic learning rate λ to 0.05. The discount factor γ is set to 0.9999. For the ICO learning parameters, the learning rate μ, the predictive range and the reflex range are set to 0.6, 0.13 and 0.07, respectively. For the exploration function parameters, the scale factor ζ is set to 0.04. V_{min} and V_{max} are assigned dynamically during learning. Figs. 4*b and* 4*c* demonstrate that the system successfully enabled the hexapod robot to perform the navigation task.



(a)



(b)



(c)

Fig. 4: (a) The set up of experiment 3. (b) The ICO exploration unit weight during learning until it converges to an optimal value after 109 trials. (c) The weights of the actor RBF network during learning until the policy converges to an optimal one after 172 trials. The average number of trials that are needed until an optimal policy is obtained is 346 ±265. A video of this experiment can be seen at [http://www.manoonpong.com/SAB2014/SVideo3.mpg]

5 Conclusions

In this paper a powerful landmark-based navigation system is proposed. It is based on sequential reinforcement learning. The experimental results show that the system enables robots to successfully learn to navigate in complex scenarios with high performance and a 100% success rate. The system is flexible to cope with different environments as well as transferable to different robots. The integration of the ICO exploration unit into the system shows an impressive and profound impact on the system performance.

In fact, using sequential RL provides a proof of concept template to solve tasks that have a sequential nature (e.g. teaching a robot arm to make a cup of coffee). A sequential task is a task that can be divided into smaller assignments that should be performed in a specific order. In the future work, we will test the navigation system on our real wheeled and legged robots. Furthermore, we

will look into the possibility of extending the system to give it the ability to determine an optimal sequence of subgoals among different valid sequences. In addition, we will investigate using new methods that enable the system to overcome partially observable MDP cases (e.g. using recurrent neural networks[14]).
Acknowledgements: This research was supported by Emmy Noether grant MA4464/3-1 and BCCNII grant 01GQ1005A (project D1).

References

1. Doya, K.: Reinforcement Learning in Continuous Time and Space. Neural Comput. 12(1), 219-245 (2000)
2. Manoonpong, P., Kolodziejski, C., Woergoetter, F., Morimoto, J.: Combining Correlation-based and Reward-based Learning in Neural Control for Policy Improvement. Advances in Complex Systems, volume 16, issue 02n03 (2013) doi:10.1142/S021952591350015X
3. Hasselt, H., Wiering, M.: Reinforcement Learning in Continuous Action Spaces. Proceedings of the 2007 IEEE Symposium on Approximate Dynamic Programming and Reinforcement Learning, ADPRL (2007)
4. Porr, B., Woergoetter, F.: Strongly Improved Stability and Faster Convergence of Temporal Sequence Learning by Utilising Input Correlations Only. Neural Comput. 18, 1380-1412 (2006)
5. Manoonpong, P., Pasemann, F., Woergoetter, F.: Sensor-driven Neural Control for Omnidirectional Locomotion and Versatile Reactive Behaviors of Walking Machines. Robotics and Autonomous Systems, Elsevier Science, vol. 56(3), pp. 265-288 (2008)
6. Woergoetter, F., Porr, B.: Temporal Sequence Learning, Prediction, and Control - a Review of Different Models and Their Relation to Biological Mechanisms. Neural Comp. 17, 245-319 (2005)
7. Bakker, B., Schmidhuber, J.: Hierarchical Reinforcement Learning with Subpolicies Specializing for Learned Subgoals. Proceedings of the 2nd IASTED International Conference on Neural Networks and Computational Intelligence, p. 125-130 (2004)
8. Botvinick, M. M., Niv Y., Barto, A. C.: Hierarchically Organized Behavior and Its Neural Foundations: A Reinforcement Learning Perspective. Cognition 113(3), p. 262-280 (2009) doi:10.1016/j.cognition.2008.08.011
9. Masehian, E., Naseri, A.: Mobile Robot Online Motion Planning Using Generalized Voronoi Graphs. Journal of Industrial Engineering 5, 1-15 (2010)
10. Sheynikhovich, D., Chavarriaga, R., Strösslin, T., Gerstner, W.: Spatial Representation and Navigation in Bio-inspired Robot. Biomimetic Neural Learning, LNAI 3575, pp. 245-264 (2005)
11. Ge, S.S., Cui, Y.J.: Dynamic Motion Planning for Mobile Robots Using Potential Field Method. Autonomous Robots, volume 13, issue 3 , pp. 207-222 (2002)
12. Arkin, R. C.:Behavior-based Robotics. Cambridge Massachusetts MIT Press (1998)
13. Collett, T.S.: The Use of Visual Landmarks by Gerbils: Reaching a Goal When Landmarks Are Displaced. Journal of Comparative Physiology A, volume 160, issue 1, pp. 109-113 (1987)
14. Dasgupta, S., Woergoetter, F., Morimoto, J., Manoonpong, P.: Neural Combinatorial Learning of Goal-directed Behavior with Reservoir Critic and Reward Modulated Hebbian Plasticity. Systems, Man, and Cybernetics (SMC), 2013 IEEE International Conference on, pp. 993-1000 (2013)