

Learning Robot-Environment Interaction Using Echo State Networks

Mohamed Oubbati, Bahram Kord, and Günther Palm

Institute of Neural Information Processing, University of Ulm
89069 Ulm, Germany

{mohamed.oubbati, bahram.kord, guenther.palm}@uni-ulm.de
<http://www.uni-ulm.de/in/neuroinformatik.html>

Abstract. Learning robot-environment interaction with echo state networks (ESNs) is presented in this paper. ESNs are asked to bootstrap a robot's control policy from human teacher's demonstrations on the robot learner, and to generalize beyond the demonstration dataset. Benefits and problems involved in some navigation tasks are discussed, supported by real-world experiments with a small mobile robot.

Keywords: robot-environment interaction, echo-state network, learning from demonstration.

1 Introduction

Any cognitive activity arises from interaction between bodies, brains and environments [1]. This means that a behaviour of a robot operating in and interacting with an environment cannot be analysed in isolation, since it is a result of properties of the robot itself (embodiment), the environment (situatedness), and the control program (task) the robot is executing. Thus, analysing the triangle (robot-task-environment) as one dynamical system could provide better understanding of phenomena governing the robots behaviours over time. Recurrent neural networks (RNNs) are powerful tools to learn such complex dynamical systems, for two main reasons. First, they are universal approximators of dynamical systems [2]. Second, they can exhibit continuous dynamics; a suitable property to model robot-environment interaction. The second reason is motivated by the fact that dynamics of nervous systems and the physical world are continuous in nature [3]. Under this perspective, many efforts have been done to analyze robot-environment interaction using RNNs. Beer and colleagues carried out rigorous analysis on understanding of a humanoid robot-environment interaction using continuous-time recurrent neural network [4]. Pasemann et. al. have shown that attractors formed in a recurrent neural network (RNN) can be used to characterise robot-environment interaction [5,6]. Their investigation shows that a RNN controller has four relevant attractors, which can be directly mapped to some environmental states like free space, obstacles, or deadlock situation. Tani et. al. explored higher cognitive abilities of robots through a multiple timescales RNN to generate reusable behaviors [7].

In this work, we assume that the robot learns by demonstration. A teacher demonstrates a task using the body of the robot *learner*, then a RNN learns from the robot-environment interaction to derive a control policy for the demonstrated behavior. This

procedure is particularly important when considering real robots. Due to the fact that demonstration dataset are exactly those that the robot would observe/execute, the learning system will be less sensitive to noisy sensors. Also in term of system integration this approach provides fast and efficient way of obtaining the control code, i.e. a nonlinear dynamical model that maps environmental situations to actions. Our learning system is an *Echo State Network* (ESN), which has two principal properties: (1) a large RNN is used as a "reservoir" of excitable complex dynamics; this network will be not trained; (2) only the weights of output connections are to be adjusted [8]. Many dynamical systems, which were difficult to learn with the existing methods, have been easily learnt by ESN [9,10]. Recently, we used ESNs to develop a dynamic controller for mobile robots [11]. The advantage is that no knowledge about the robot model is required; a useful property in practical situations, where the exact knowledge about the physical parameters of the robot is almost unattainable. We also explored the notion that a well trained ESN needs to change only its internal state to change its behavior policy [12,13]. The ESN, in this work, is asked to bootstrap a robot's control policy from teacher's demonstration, and to generalize beyond the demonstration dataset. Naturally, we expect adequate performances only for system conditions that are close to those seen during training, such that valid solutions could be acquired for similar states that may not have been seen during demonstration.

The rest of this paper is organized as follows. Section 2 explains the learning design, and section 3 presents some implementation results. A discussion and conclusion are drawn in section 4.

2 Learning Design

The world (robot-task-environment) consists of states S and actions A , with the mapping between states from which actions being defined by a probabilistic transition function $T(\acute{s}|s, a) : S \times A \times S \rightarrow [0, 1]$. We define the demonstration set D as k paires of observation and actions $D = \{(s_i, a_i)\}$, $s_i \in S$, $a_i \in A$, $i = 1, \dots, k$. The ESN is then provided with the set D , and asked to aquire the task dependent control policy $\pi : S \rightarrow A$ to select desired actions based on current states.

2.1 Echo State Networks

An ESN (Fig. 1) is formed by a so-called "Dynamic Reservoir"(DR), which contains a large number of sparsely interconnected neurons with non-trainable weights. The activation of internal neurons is updated according to

$$X(n+1) = f(W^{in}U(n+1) + WX(n) + W^{back}Y(n+1)) \quad (1)$$

and the outputs are calculated as

$$Y(n+1) = f^{out}(W^{out}(U(n+1), X(n+1), Y(n))) \quad (2)$$

An essential condition for successful using of ESN is the “echo state” property. It is a condition prior to training, where the actual network state is required to be an “echo” of its history. This means that the state of each internal neuron x_i can be mapped by input/output (u/d) histories through a function e_i , i.e. $x_i(n) = e_i(d(n - 1), d(n - 2), \dots, u(n), u(n - 1), \dots)$ [9]. If this condition is met, only weights connections from internal neurons to the output (W^{out}) are to be trained. This could be done by any suitable training method (least squares method, etc.) in a one-shot fashion [8].

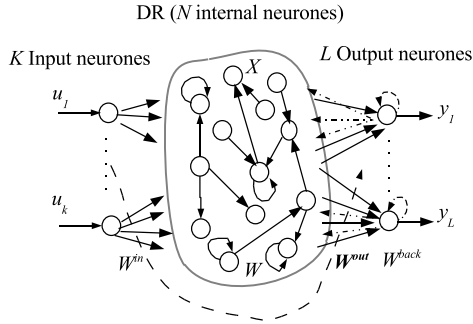


Fig. 1. Basic architecture of ESN. Dotted arrows indicate connections that are possible but not required.

2.2 Deriving Policy with ESNs

During demonstration, a teacher operates the robot while recording its sensor/actuator data. Once the demonstration set $D = \{(s_i, a_i)\}$ have been gathered, the ESN performs a batch learning to derive the control policy $f \approx \pi : S \rightarrow A$ (Fig. 2).

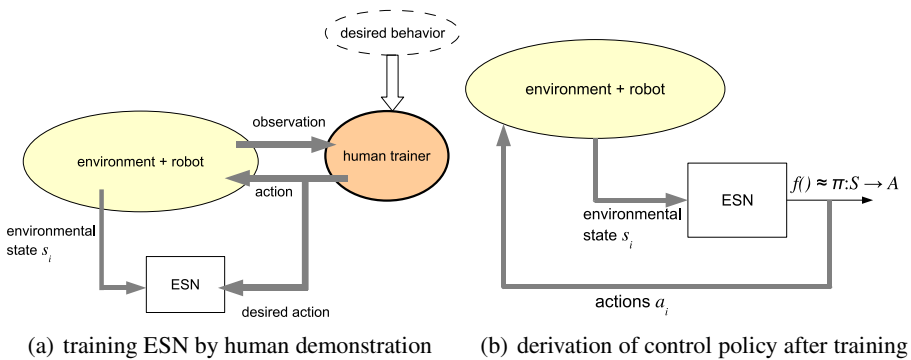


Fig. 2. Learning design

3 Implementation

We implemented the learning design on a small mobile robot, called e-puck [14], equipped with 8 infrared (IR) proximity sensors and two stepper motors. We drove the robot manually several times for the task to be learned. The linear velocity of the robot (v) was kept constant while the angular velocity (w) was controlled by a human operator using a keyboard (Fig. 3). During the movement we collected sensor readings and angular velocities for training. Training was performed using ESNs with 8 inputs (8 IRs), N internal neurons, and one output that represents the desired angular velocity transformed to two wheel speeds. No back-connection from the output to the DR, and no synaptic weight connections from the input directly to the output. The input and the internal synaptic connections weights were randomly initialized from a uniform distribution over $[-1, +1]$. The internal weight matrix W has a sparse connectivity of 20% and scaled such that its maximum eigenvalue $|\lambda_{max}| = \alpha$ (also called spectral radius). After training, the ESN provides the desired angular velocity based on the actual sensor-readings (Fig. 4). We performed several experimental tests; three of them are reported here: *door-passing*, *wall-following*, and *route-learning*. We will show the effect of the internal neurons number N and the spectral radius α on the ESN performance.



Fig. 3. Implementation. (a) demonstration made on the robot learner. (b) after training, the ESN controls the robot.

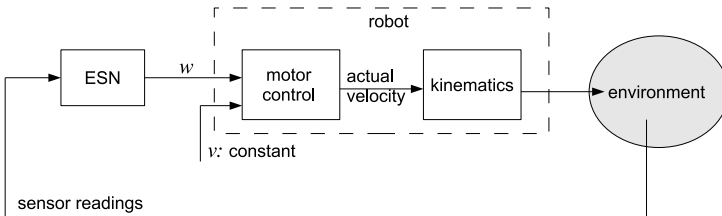


Fig. 4. Robot control with ESN

3.1 Door-Passing

Here, the robot learns how to move through a door. Demonstration data were prepared by driving the robot several times in a door-passing scenario, while collecting sensor readings and correspondent angular velocities (Fig. 5). After training, we put the robot in a scenario 1 where the robot should pass through two openings (Fig. 6. a). The spectral radius $\alpha = 0.8$, and the number of neurons in the reservoir N receives the values $\{6, 8, 12\}$. During this test we were surprised to see that even with 6 internal neurons the ESN could generalize successfully and bring the robot through the two doors. In a more complicated environment (scenario 2), we expect that the robot moves through the openings D_2 and D_4 (Fig. 6. b). The number of neurons in the reservoir is kept to 12, and α is varied from 0.4 to 0.8 in a step of 0.2. We can see that the ESN with $\alpha = 0.8$ performed the best, regarding the smoothness of the robot path obtained with this spectral radius.

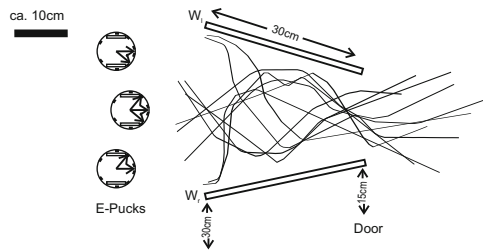


Fig. 5. Collection of training data for door-passing: The teacher moves the robot from different start positions in the environment in order to demonstrate the most possible situations. Demonstrations were in a form of 10 movements producing 750 data pairs sampled with a period of 0.1 second.

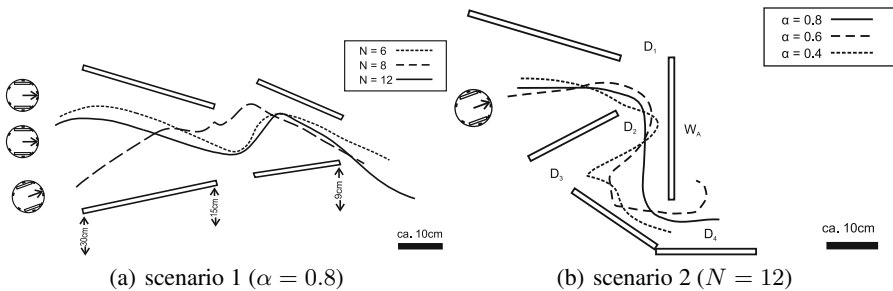


Fig. 6. Door-passing

3.2 Wall-Following

In this task two main situations need to be learned: (1) move parallel to a wall, and (2) turn in a concave corner. We have first collected training data by driving the robot several times in these two situations as illustrated in Fig. 7. Due to the smooth movement of the robot during previous experiment we kept the spectral radius to $\alpha = 0.8$, and we tested the ESN performance with different number of internal neurons $N = \{6, 8, 12\}$ (Fig. 8). To show the degree of similarity between the actual and the demonstrated robot behaviors we did a test in a form of occurrence of the robot distance to the wall in training and test data (Fig. 9). The distance was measured using one infrared sensor (IR_3) on the robot. By merely looking at these results, we can subjectively say that ESNs could reproduce successfully the demonstrated behavior, since the IR_3 delivered values lie often between $[0, 1000]$. Fig. 10 shows a recovery testing situation, in which we put an object (A) at the corner and we made a small opening (B) (2 cm) in the wall. Moving near (A) or (B) means that the ESN has to deal with a convex corner at (A) (not seen before), and maintain its stability in presence of sensory inputs totally beyond the interval of training data at (B). An ESN with small dimension (6 and 8 internal neurons) showed a good performance to recover those perturbations, whereas using large dimension (more than 12 internal neurons) the control loop lost stability at many times. We can see that with 12 neurons the system began to lose its stability, when reaching (B).

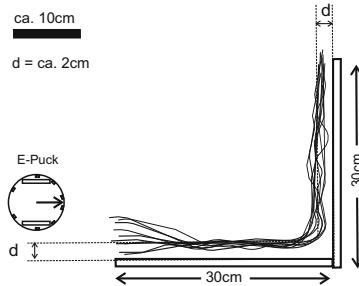


Fig. 7. Demonstration data for the task wall-following. IR sensors and wheel speeds were collected by moving the robot several times to show how to follow a wall and how to deal with a concave corner. 10 movements were collected in a form of 693 pairs sampled with a period of 0.1 second.

3.3 Route-Learning

Here, we demonstrate the ability of an ESN to learn a more challenging task; *learning a route*. The robot was led many times along a desired route to collect rich information about the environment. Fig. 11. (a) shows 20 rounds collected to train the ESN controller, where the two red dashed lines represent the borders of route to be learned. After training process is completed, the acquired perceptions are associated with motor actions, enabling the robot to follow the route autonomously. Fig. 11. (b) shows that the trained robot follows the desired trajectory well, with few deviations. This result shows also how the ESN is reliable and copes with noise and new environmental perceptions.

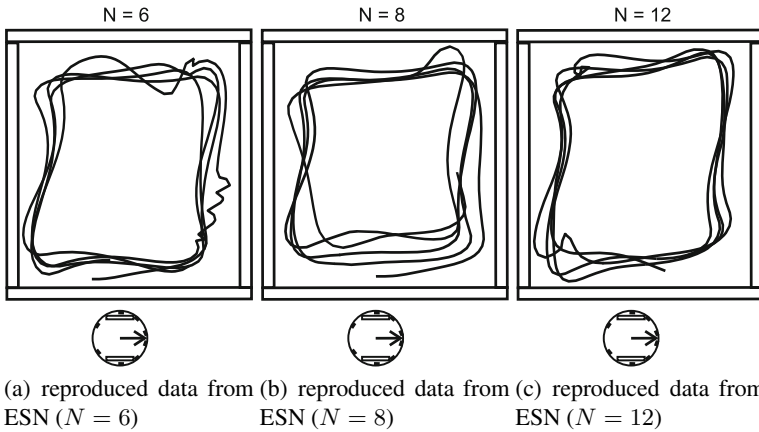


Fig. 8. Wall-following: The results show that the trained ESNs could accomplish this task successfully, but the robot behavior is found to be almost identical for the different neurons N .

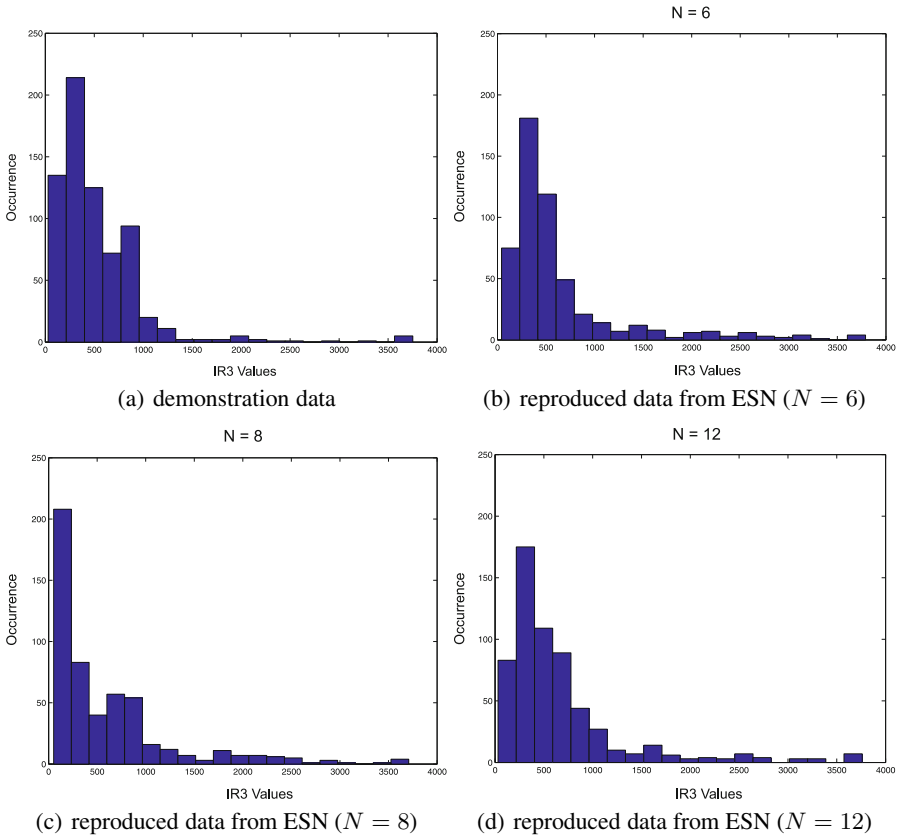


Fig. 9. Occurrences of the robot distance to the wall from demonstrated and reproduced data

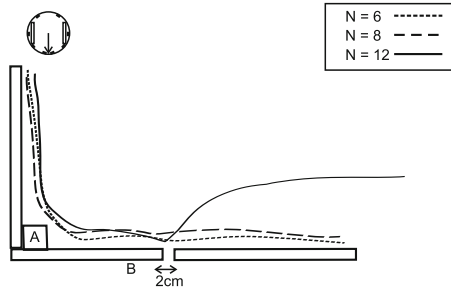


Fig. 10. Wall-following with recovery testing: We put an object (A) at the corner, which means that the robot has to deal also with a convex corner, and we made a small opening (B) (2 cm) in the wall, such that sensory data in those situations are totally beyond the interval of training data. ESNs with 6 and 8 neurons could recover those perturbations learner, whereas with 12 neurons the control loop began to lose stability.

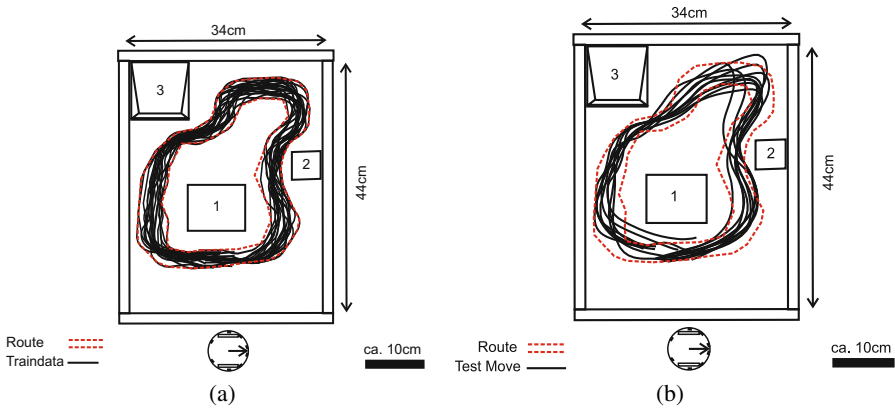


Fig. 11. Route learning with ESN. (a) human teacher led the robot between obstacles (1,2,3) to collect training data in a form of 2390 samples. (b) ESN with 20 internal neurons and $\alpha = 0.7$ controls the robot to reconstruct the desired route.

For example when the robot deviate from the trained route, due to its kinematic constraints, the ESN could generalize, and generate appropriate control signals in order to bring the robot again on its desired path.

4 Discussion and Conclusion

In this paper we have implemented the ESN as a learning system to derive a control policy from robot-environment interaction. A similar work has been done in [15], where the task consists in finding and then getting closer to a possibly moving target. The ESN has shown great performance to solve that task; a result that pushed us to test the ESN in more complicated tasks. Furthermore, we wanted, in this paper, to see the effect of

the internal neurons number N and the spectral radius α on the ESN performance in performing the tasks of *door-passing*, *wall-following*, and *route-learning*. Once these behaviors have been demonstrated using the body of the robot learner, three ESNs performed batch learning on demonstration datasets to derive a control policy for those tasks. The results have shown that an ESN even with 6 internal neurons could reproduce and generalize successfully the door-passing task. Experiments have also shown that an ESN with $\alpha = 0.8$ performed the best, regarding the smoothness of the robot path obtained with this spectral radius. In the wall-following task the robot has learned how to maintain a distance parallel to the wall, and how to deal with a concave corner. The robot reproduced successfully the demonstrated behavior, and also showed a great robustness against new situations, i.e. in presence of convex corner, and an opening in the wall. The task of route-learning was the more challenging to be learned, since each IR-sensor plays an important role in the learning process. The result shows that the trained robot follows the desired trajectory well, with few deviations. These deviations means that the ESN receives completely new environmental perception; a hard generalization test. In those new states the ESN could generalize and generate appropriate control signals in order to bring the robot again on its desired route.

We note that the learning system has also shown poor performances in many situations. We identified three main causes of that. The first cause is due to the dataset sparsity; undemonstrated areas in the state space. Using a keyboard, the human trainer was not able to demonstrate behaviors in all areas of the state space. This has raised the question of how the robot should act when it encounters a state without a demonstration. When novel states (very different from previously demonstrated states) are encountered by the robot, the ESN could not generalize optimally. In those situations, it was necessary to acquire additional demonstrations, and re-make training. The second cause was the quality of the dataset. The poor quality of the employed sensors as well as the kinematic precision of the utilized robot has generated dataset ambiguity in some situations. For example, during multiple execution of a desired task, different actions has been mapped to almost identical states. The third cause was the learning system itself. During preparation of the ESN, it was not easy to find its optimum parameters. Using a “relatively” large dimension (more than 20 internal neurons) the network lost stability at many times. This is possibly due to the high dimensionality of the system, or to the relatively small amount of training data. We have made a similar observation in another experiment, when we trained an ESN as an adaptive velocity controller for an omnidrive mobile robot [13].

There are many research questions to be addressed in the future. We summarize three main points:

- What is the behavioral capacity of ESNs? In this work, we trained separate ESNs for each task. We did not discuss the question whether a single ESN could learn and reproduce all demonstrated behaviors. In a previous work [12] we have shown that an ESN needs to change only its internal state to change its behavior policy. Does learning with fixed-weights perform well in these tasks?
- How to improve generalization? When a completely novel state is encountered by the robot, the ESN is unable to produce an adequate action. In this case, the ESN should be able to recognize the lack of knowledge and solicit help from the teacher.

- How to deal with symetrie in a route learning? In a route learning it is possible to encounter similar situations which need different actions. The short term memory in ESNs might provide a solution.

References

1. Pfeifer, R., Bongard, J.C.: *How the Body Shapes the Way We Think: A New View of Intelligence* (Bradford Books). The MIT Press, Cambridge (2006)
2. Funahashi, K.-i., Nakamura, Y.: Approximation of dynamical systems by continuous time recurrent neural networks. *Neural Network* 6(6), 801–806 (1993)
3. Beer, R.D.: A dynamical systems perspective on agent-environment interaction. *Artif. Intell.* 72(1-2), 173–215 (1995)
4. Beer, R.D.: The dynamics of active categorical perception in an evolved model agent. *Adaptive Behavior* 11(4), 209–243 (2003)
5. Pasemann, F.: Dynamics of a single model neuron. *International Journal of Bifurcation and Chaos* 3, 271–278 (1993)
6. Hülse, M., Zahedi, K., Pasemann, F.: Representing robot-environment interactions by dynamical features of neuro-controllers. In: Butz, M.V., Sigaud, O., Gérard, P. (eds.) *Anticipatory Behavior in Adaptive Learning Systems*. LNCS (LNAI), vol. 2684, pp. 222–242. Springer, Heidelberg (2003)
7. Tani, J., Yamamoto, J.: On the dynamics of robot exploration learning. *Cognitive Systems Research* 3(3), 459–470 (2002)
8. Jaeger, H.: Tutorial on training recurrent neural networks, covering BPPT, RTRL, EKF and the echo state network approach. Technical Report 159, AIS Fraunhofer, St. Augustin (2002)
9. Jaeger, H.: The ‘echo state’ approach to analysing and training recurrent neural networks. Technical Report 148, AIS Fraunhofer, St. Augustin, Germany (2001)
10. Jaeger, H., Haas, H.: Harnessing Nonlinearity: Predicting Chaotic Systems and Saving Energy in Wireless Communication. *Science* (April 2004)
11. Oubbati, M., Schanz, M., Buchheim, T., Levi, P.: Velocity control of an omnidirectional robocup player with recurrent neural networks. In: Bredenfeld, A., Jacoff, A., Noda, I., Takahashi, Y. (eds.) *RoboCup 2005*. LNCS (LNAI), vol. 4020, pp. 691–701. Springer, Heidelberg (2006)
12. Oubbati, M., Schanz, M., Levi, P.: Meta-learning for Adaptive Identification of Non-linear Dynamical Systems. In: *Proc. Joint 20th IEEE International Symposium on Intelligent Control and 13th Mediterranean Conference on Control and Automation*, Limassol, Cyprus. IEEE, Los Alamitos (June 2005)
13. Oubbati, M., Palm, G.: A neural framework for adaptive robot control. *Journal of Neural Computing and Applications* 19(1), 103–114 (2010)
14. Mondada, et al.: The e-puck, a robot designed for education in engineering. In: *Proc. of the 9th Conf. on Autonomous Robot Systems and Competitions*, vol. 1, pp. 59–65 (2009)
15. Hartland, C., Bredèche, N.: Using Echo State Networks for Robot Navigation Behavior Acquisition. In: *ROBIO 2007*, Sanya Chine (2007)