A Connectionist Architecture for Learning to Play a Simulated Brio Labyrinth Game

Larbi Abdenebaoui[1], Elsa A. Kirchner[1], Yohannes Kassahun[1], and Frank Kirchner[1,2]

[1] Robotics Group, University of Bremen
Robert-Hooke-Str. 5, D-28359, Bremen, Germany
[2] German Research Center for Artificial Intelligence (DFKI)
Robert-Hooke-Str. 5, D-28359, Bremen, Germany

1 Introduction

The Brio labyrinth, shown in Figure 1, is a popular game consisting of a board with holes and walls. In addition, the game has knobs which are used to tip the board in two planar directions for controlling the angle of the board. The aim of the game is to maneuver a steel ball along a marked path from a starting position to a final position on the board by tipping it so that the ball moves without falling into any of the holes. The path is partially bordered by the walls. Some of the walls form corners, where the ball can be controlled easily.

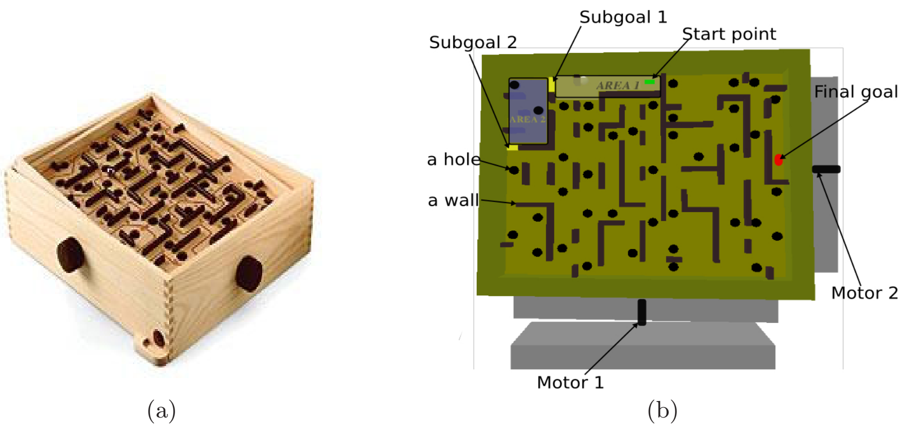
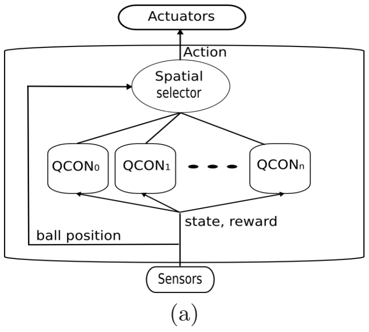


Fig. 1. (a) The Brio labyrinth (b) The simulation with the first two subareas labeled

To enable an artificial agent to play the game, an ODE [3] based simulation of the game has been realized (see Figure 1b). The dimensions of the board and the number of holes and walls of the simulated game correspond to those of the real one. The walls are modeled as rigid bodies represented by boxes connected to the main board with fixed joints. The ODE-based simulation allows a realistic reproduction of the physical properties of the game. Collision-handling routine was called at each step of the simulation using the Coulomb friction model. The knobs are simulated with motors.

2 Learning Architecture

Steering the ball through the whole Brio labyrinth is a very complex task. A human player solves it by breaking it down into several smaller problems. To learn the proper movements a hierarchical learning architecture based on QCON has been developed. The QCON is a connectionist Q-learning model proposed by Lin [1] where each action has a separate network. As observed in human players and following the divide and conqueror paradigm, the labyrinth was subdivided into small regions, where a QCON is assigned to each region (Figure 1b/2a).



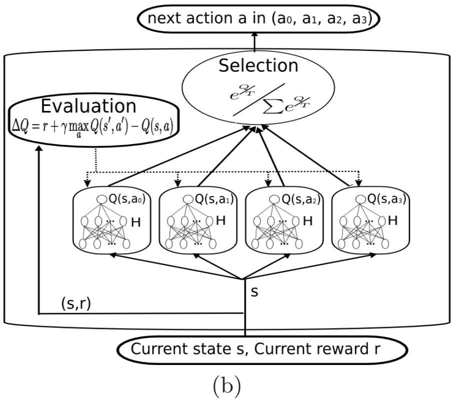

next action a in (a0, a1, a2, a3)
Selection
Evaluation
ΔQ = r + γ max_a Q(s′,a′) − Q(s,a)
Q(s,a0)
Q(s,a1)
Q(s,a2)
Q(s,a3)
H
s
(s,r)
Current state s, Current reward r
(b)

Fig. 2. (a) Illustration of the whole architecture. The current state and reward are inputs into the architecture. The output of the whole architecture is the output of an active QCON. (b) QCON architecture: Each action-value is represented by a feedforward network with one hidden layer that is trained using back-propagation algorithm and Q-learning with the parameters given in Table 1. We have four possible actions a_0, a_1, a_2, a_3 (see Table 1).

First the QCONs are trained separately on their respective subareas. In order to connect two subsequent areas, the subgoal of the first area is used as a starting region for the next one. In the play phase, based on the current position of the ball, a spatial selector module selects the appropriate learned QCON to be active and sends the output of the QCON to the actuators. This solution is inspired by "place cells" [2] found in the hippocampal brain region of rats. Place cells are found to be selectively active when the rat is situated in different locations while performing navigational tasks in a labyrinth environment [4]. The chosen approach has the following advantages: (1) It is easier to achieve a solution with an architecture composed of a committee of QCONs than a monolithic one. (2) The solution scales up easily as the complexity of the game increases. The complexity of the architecture (the number of QCONs and the number of the hidden neurons in each QCON) is directly proportional to the complexity of the game (number of holes and walls).

3 Experiments

The parameters of the experimental setup are shown in Table 1. They are found empirically after performing various experiments on different subareas.

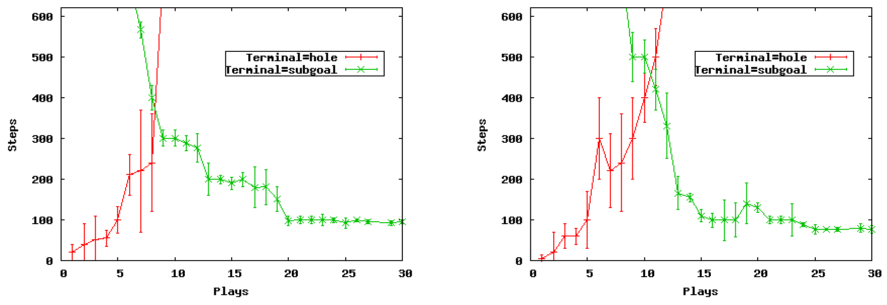
Table 1. Parameters of the experimental setup

Factor	Description
State	state s=$(x, y, V_x, V_y, P_1, P_2)$ (x,y) The ball position on the board (V_x,V_y) The ball velocity on the board (P_1,P_2) Motors position values
Action	For every motor there are two possible rotations: turn clockwise or turn anti-clockwise relative to the current position in steps of 0.1 deg; there are 4 possibles actions $a_0 = (-0.1, 0.1), a_1 = (-0.1, -0.1), a_2 = (0.1, 0.1), a_3(-0.1, -0.1)$
Reward	-0.5 if in hole; 1 if in subgoal; 0 otherwise
Learning	Discount factor γ=0.8; Learning rate α=0.2 Number of hidden units in a QCON net H=10
Actions selection	Stochastic: $\frac{e^{Q/T}}{\sum e^{Q/T}}$ Simulated annealing T:1 $\rightarrow$ 0.0002
Study	Average over 10 experiments in a single area; Play after after each 10 trials with greedy policy Maximum number of steps per play 600

For each study, where a QCON was trained in a given subarea, we performed 10 experiments. An experiment consisted of 300 trials, and after each tenth trial the agent played with the learned greedy policy. A trial begins with a random position and terminates when the ball falls in a hole, or when it attains the subgoal, or when the number of steps is greater than 600. We subdivided the labyrinth manually based on a predefined number of holes on a single subarea. This number is limited to two holes (see Figure 2b).

4 Results

The plots in Figure 3 show the results on the first two subareas. Two subtasks are solved by the agent. The first one is to avoid the holes which needed on average about 100 trials for both areas, and the second one is to attain a subgoal in a given subarea as fast as possible. The second subtask needed about 250 trials for both areas. We have found that the number of trials to learn the two subtasks



First area Second area

Fig. 3. Plots of the number of steps versus plays. One step corresponds to one action, and every play was performed after 10 trials using the learned greedy policy. Red curve (hole): number of steps needed before the ball falls in a hole. Green curve (subgoal): number of steps needed to reach the defined subgoal.

was similar for the other remaining subareas. Once trained, a QCON network does not need further adaptation when playing the game continually from the start point to the final goal in the whole labyrinth.

5 Summary and Outlook

We have presented a connectionist architecture for learning to play a simulated Brio labyrinth game that uses the divide and conquer paradigm inspired by the way a human player plays the game. We have shown that the architecture scales up easily as the number of subareas increases. In the future, we want to develop a way of automatically dividing the board into subareas. We also want to transfer and test the architecture on the real labyrinth and thereby improve its performance.

References

1. Lin, L.-J.: Self-improving reactive agents based on reinforcement learning, planning and teaching. Machine Learning 8(3-4), 293–321 (1992)
2. O'Keefe, J., Dostrovsky, J.: The hippocampus as a spatial map. preliminary evidence from unit activity in the freely-moving rat. Brain Research 34(1), 171–175 (1971)
3. Smith, R.: Open dynamics engine (2005), `http://www.ode.org`
4. Wilson, M.A., McNaughton, B.L.: Dynamics of the hippocampal ensemble code for space. Science 261(5124), 1055–1058 (1993)