accepted as poster to *FNS 1998*

Fuzzy-Neuro Systems 19./20. March 1998 – Munich (FRG)
in: W. Brauer (Ed.) Proc. 5th International Workshop "Fuzzy-Neuro-Systems '98- Computational Intelligence, FNS 98", Munich, pp. 344-351, Proceedings in Artificial Intelligence, infix-Verlag 1998

Medically Motivated Testbed for Reinforcement Learning in Neural Architectures*

Dimitrij Surmeli, Gunther Köhler, and Horst-Michael Gross
{dima,koehler,homi}@informatik.tu-ilmenau.de

Technical University Ilmenau
Department for Neuroinformatics

Abstract We introduce a testbed for performance testing of reinforcement learning algorithms. The testbed is motivated by problems arising from a medical application, therapy of traumatic coma. Results from tests of a Neural Gas as a clusterer for continuous inputs linked with Q-learning are presented as well as detailed plans regarding future experiments.

keywords: reinforcement learning, benchmark testbed, neural nets

1 Introduction

The therapy of traumatic coma is a complicated process characterized by an extreme individual variability of lesion and physiological reaction, enhanced by varying personal and equipment factors. A computer-aided physician's assistant could use available clinical data for adaptive intervention suggestion and permanent individualized monitoring with unwavering attention and quality. The system should make use of existing knowledge, yet also remain adaptive to the individual patient and learn from its own experiences of success and failure, thereby extending and validating instilled knowledge. To that end, our approach uses techniques from reinforcement and neural network learning.

Our preliminary analysis of the medical therapy yielded some general characteristics of patient dynamics. Upon arrival, a patient is in a critical, yet still physiological (and therefore, bounded) state and has to be nursed into a sufficiently better state. During that,

* This work is supported by the Thuringian Ministery for Science, Research and the Arts (project ITHERA, B 511-95005)

executing a null action (meaning, to do nothing at all) does not mean no change in the patient's status, it will usually worsen it significantly. We term this phenomenon 'eigen-dynamics'. Interventions to stabilize or elevate the patient's status not only change the status, but may also alter the reaction: in the same state, the patient may react differently to the same intervention. That means, the effect of interventions depends on the state of the patient, and the kind and recency of preceding interventions. This phenomenon we capture by 'reactive dynamics'. Beyond that, we realize that certain interventions have a time-dependent effect which prevents or supports the effect of same or other interventions. To assess the capabilities of reinforcement learning algorithms, in order to be able to match agents to a specific task, or to break an overall task into managable smaller tasks and investigate the cooperation of the subtask agents, we have created an artificial testbed. It was designed to capture the phenomena mentioned above in an easily accessible and controllable process with known properties.

2 Testbed

The testbed is composed of a ball set into a mountainous environment. There maybe one or more peaks and zero or more valleys. In our understanding, the movement of the ball is controlled by one or more agents. In reinforcement learning terms, we understand 'nursing to a better state' as finding a goal state from a random initial state. Thus, the ball's primary goal consists in getting to the top of the largest peak. To achieve that, the ball is equipped with an 'engine'. In trying to roll up a peak, it has to battle gravity and friction. The ball must thus also have a certain inertia. Therefore, when it executes a certain action, the effect of that action will carry over to the next action in that the ball can accumulate speed.

We decided that an action shall be composed of a thrust and a steering angle for the 'engine', standing in for the dosis and the effect of interventions, respectively. This seemed to correspond better to a medical intervention, which tries to 'nudge' the development of the patient's state with a certain force into a desired direction. Executing a null action means the ball will roll into a valley, which is interpreted as a deteriorating health status. Reaching the top along with rolling downhill should capture the mentioned 'eigen-dynamics'. Currently, the other,'reactive' dynamics are implemented through a function which changes the efficiency of an executed action: the actually effective, oriented thrust $th_{eff} = th_{momentary} + \alpha * th_{old}; \alpha < 1$. 'Oriented' means that the effective steering angle has already been considered in this calculation. th_{eff} should capture that momentary and older actions interact, e.g. actions in different directions. Further, this effective thrust

saturates as it approaches an upper limit: an acceleration in the same direction with the same momentary thrust results in a lesser or no increase of the ball's velocity.

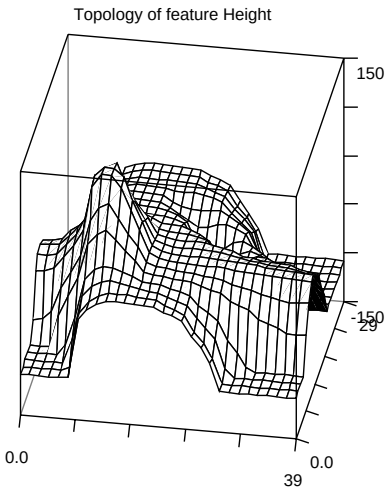

Figure1: An actively moving ball is supposed to learn to climb up to a peak in a mountainous environment. On a slope, gravity will counteract upwards movements by the ball. The boundaries of the environment and the absolutely lowest valley are 'deadly'. The x-, and y-directions of the environment can be viewed as certain measurements characterizing the patient's health, whereas the height mimics the effort necessary to change the patient's status. The interplay of inertia and gravity with the height map symbolize the patient's physiology.

In addition to the feature 'Height', our world can be extended by any number of features, such as 'Energy' or 'Color', to prompt efficient movement and realize landmarks.
Positive external reinforcement is delivered upon reaching maximum height, whereas negative reinforcement is returned upon reaching minimum height, collision with the boundary of the environment or exceeding a threshold on the number of steps in the current trajectory. Here, a trajectory means the movement of the ball in the environment from some inital state until receiving reinforcement. This testbed unifies and extends a number of tasks proposed for reinforcement learning experiments, such as goal finding in grid-world environments with or without static obstacles, the mountain car experiment or pole balancing. [4] describes more realistic experiments similar to ours also in that the 'goal finding' behavior is presented as the search for food units, and probably both the 'eigendynamics' and the 'reactive dynamics' are embodied by the agent's enemies who actively pursue it. From this point of view, our feature 'Color' can also be used as a static obstacle along with the reactive dynamics as dynamic obstacles. However, we regard our setup as difficult as this, and much more plausible in the medical context.
This testbed uses global, GPS-like (GPS=Global Positioning System) sensations as input for the agent. We think physicians actually have GPS-like information about the patient's state.

2.1 Configuration actually used

In this paper, we report on experiments to show the ability of reinforcement algorithms to successfully solve the problems posed above. The experiments gradually become harder,

using only the feature 'Height'. We have subdivided the complexity of the environment, as well as the physics of the ball, into three stages: **environment:** 1)only one peak; 2)one peak and several valleys; 3)one positive peak, several valleys and reactive dynamics; **ball characteristics** I)no inertia, no gravity, no friction; II)no gravity, no friction; III)with inertia, gravity and friction. In the following figures and tables, 'plain' refers to experiments without inertia, gravity or friction. Similarly, 'w/ inertia' marks experiments with inertia invoked, but no gravity or friction. Those take effect in experiments marked 'w/ inertia, grav and frict'.
The power of the installed engine is set to be slightly greater than gravity, such that the ball could just drive straight up a slope.

Our actions are setup as global actions, e.g. 'go South with full thrust'. For these tests, the representation of the action components along with that of the height and height gradients, are discrete. Thrust can take on values between -1 (full back) and 1 (full forward) in 0.5 increments, and the steering angle can range from $-\pi/2$ to $\pi/2$ in 30° increments. Sensations are perceived by the agent in continuous form. When no inertia, gravity or friction are invoked, the input consists only of the position components (x, y). In all other cases, these can be supplemented by the (x, y) components of the speed vector.

2.2 Algorithm actually used

The first algorithm tested and the only one reported here features a Neural Gas (NG in the following) as a clusterer for the continuous inputs onto a fixed number of NG-neurons. Q-learning([6]) is used, such that for every NG-neuron, every executable action is assigned a Q-value representing the utility of execution of that action in this prototypical state. The utility of every state-action pair is calculated from the reinforcement received after the subsequent transition and the discounted maximal utility of an action in the new state. Details of the algorithm can be found in [2].

The agent is trained until the error for the prediction of the Q-values has converged enough to satisfy the progress criterion as defined in [5]. That paper also determines the paradigm of our tests regarding training, validation and test sets and their usage. Those rules are intended to establish a rigorous performance testing for reinforcement learning algorithms according to the spirit governing benchmark tests for supervised learning.

3 Test results

We use the rate of successful trials (the ball is set in and reaches the goal area) within a training epoch and the test error as measures of performance. For comparison, results for a non-learning random walk are included. Of course, the prediction error will be very low, as Q-values do not change and are intitialized equal.

3.1 Results for the different experiments using only position as input, with one positive peak, no valleys

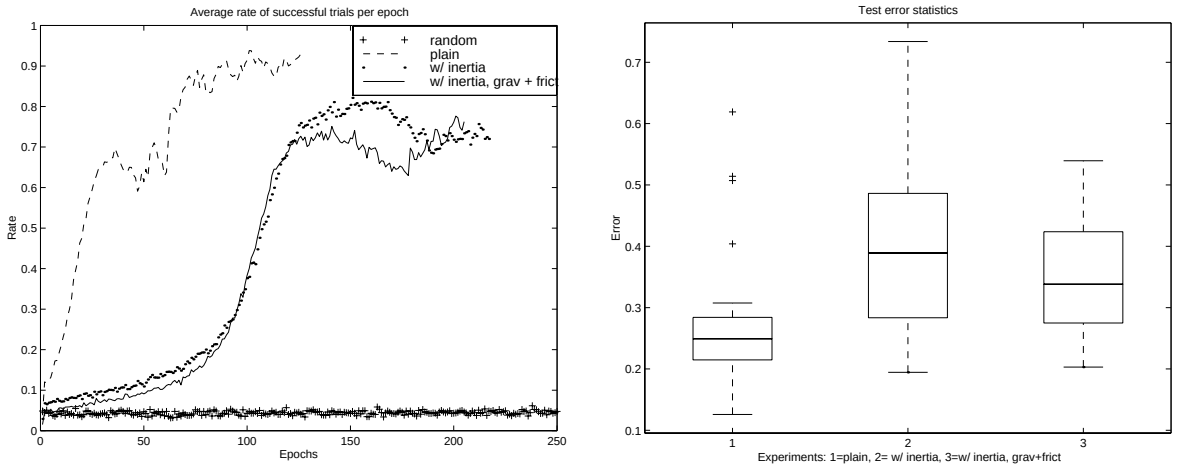

Figure2: Rates of successful trials per training epoch (100 trials) averaged over 50 realizations of each experiment and box-whiskers plots for the prediction error on the test set.

As is shown in (Fig.2), the agent can find the goal. The agent performs slower and worse when inertia and gravity are invoked. Note the extremely high number of training cycles required to solve the problem. This is mainly due to the 'puristic' reinforcement function which rewards only terminal states. The performance as measured by the error in predicting the Q-values for the test set, are provided as quartiles in (Fig.2 and Tab.1). The respective mean and standard deviation are not plotted, as none of the error distributions passed the Kolmogorov-Smirnov-Test([3]) comparing them to a Log-Normal distribution.

testbed physics	mean	Std.Dev	Q1	median	Q3	Min	Max
plain	0.2593	0.0878	0.2125	0.2489	0.2831	0.1258	0.6190
w/ inertia	0.4037	0.1411	0.2835	0.3891	0.4862	0.1945	0.7338
w/ inertia, grav and frict	0.3548	0.0896	0.2750	0.3382	0.4237	0.2031	0.5394

Table1: Statistics for the prediction error on the test set.

3.2 Results with several valleys, one peak

With the more complicated environment, the success rate drops and the prediction error increases compared to the agent with position as only input(Fig.3). The performance between the agent with position-only input and the agent given additional speed input, with or without reactive dynamics, are comparable. This supports the intuition that the additional information about speed disambiguates hidden states (same position, different speed). However, the increased input dimensionality and sparser distribution of the NG neurons more counters that effect: the NG loose their topological coding. It is surprising that the reactive dynamics can not only be learned, but actually lead to faster, yet more unstable learning. Performance measures are provided in (Fig.3 and Tab.2).

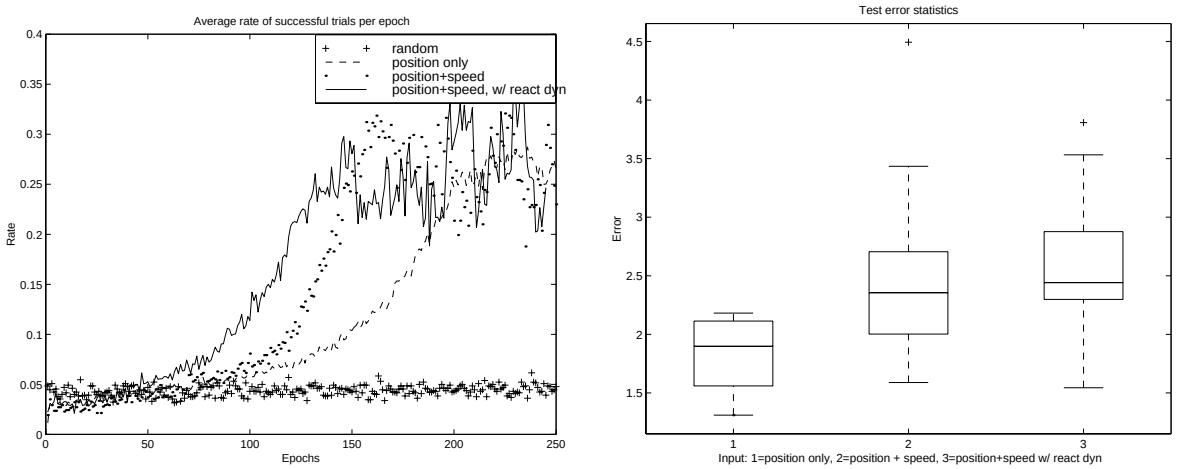

Figure3: Rates of successful trials per training epoch (100 trials) averaged over 50 realizations of each experiment and box-whiskers plots for the prediction error on the test set.

input for agent	mean	Std.Dev	Q1	median	Q3	Min	Max
position only	1.7789	0.2735	1.5985	1.7603	1.9927	1.1085	2.4037
position+speed	2.5153	0.7651	2.0021	2.3551	2.7048	1.5892	4.4936
same + react dyn	2.4602	0.6177	2.1081	2.3891	2.6230	1.5438	3.8071

Table2: Statistics for the prediction error on the test set.

4 Summary

We have shown that the problems embedded in a test environment which is phenomenologically similar to a patient, can be solved at least partially by a reinforcement learning

algorithm utilizing a Neural Gas as input clusterer and Q-learning for the state-action pairs. We also show the degradation of performance with increasing problem complexity, and suggest possible solutions next.

Since the time needed for training the agents is unacceptable, we will first investigate methods to speed up learning by presenting the training set ordered such that first inputs are used which are very close to the goal area, and only later inputs with greater Euclidean distance. Also, the agent will receive an additional immediate internal reinforcement, derived from its sensations and possibly, last actions. Here, the increase in height at the new position could be used. To derive conclusions for the implementation, a trained agent will also be tested with a different height map, analogous to a different patient. We will also extend the range of methods investigated with this artificial setup by directly approximating the mapping of continuous inputs to Q-values with several function approximators. For those tests, we will utilize the residual and residual gradient approaches described in [1]. Regarding the planning problem, we will investigate the utility of action and input traces, i.e. extend the input space by some history.
Using the experience gained with this artificial environment, we are planning to use real clinical data as environment for similar tests, or models that capture actual clinical data adequately and generalize them. This will gradually build the desired physician's assistant system. Finally, the time scales for the environment and the agent shall be desynchronized: in a medical context, the strict binding loop of action-environmental dynamics-sensation-agent-action cannot be upheld. Apart from its non-deterministic and non-stationary characteristics, the environment will then appear also partially non-causal to the agent. We are not aware of any approaches to that problem.

References

[1] L. C. Baird. Residual algorithms: Reinforcement learning with function approximation. In Armand Prieditis and Stuart Russell, editors, *Proceedings of the Twelfth International Conference*, Machine Learning. Morgan Kaufman Publisher, 1995.

[2] H.-M. Gross, V. Stephan, and H.-J. Böhme. Sensory-based robot navigation using self-organizing networks and Q-learning. In *Proc. WCNN'96, World Congress on Neural Networks*, pages 94–99. Lawrence Erlbaum Associates, Inc., Publishers, September 1996.

[3] Steve Lawrence, Andrew D. Back, Ah Chung Tsoi, and C. Lee Giles. On the distribution of performance from multiple neural network trials. *IEEE Transactions on Neural Networks (IEEE TNN)*, accepted, 1997.

[4] Long-Ji Lin. Self-improving reactive agents based on reinforcement learning, planning and teaching. *Machine Learning*, 8:293–321, 1992.

[5] Lutz Prechelt. Proben1 - A set of neural network benchmark problems and benchmarking rules. Technical Report 21/94, Fakultät für Informatik, Universität Karlsruhe, 1994.

[6] C.J.C.H. Watkins and P. Dayan. Q-learning technical note. *Machine Learning*, 8:279–292, 1992.

5 Appendix

This section gives the details about the reinforcement learning architecture used in all experiments. [2] defines an algorithm composed of two modules: a Neural Gas clusterer to reduce continuous or quasi-continuous input spaces to a finite number of typical situations encountered by the agent, and Q-learning for the actions available in each of these situations. Contrary to intuition, stabilizing the locations of the NG-neurons first and then training the Q-values for them, yields worse and slower overall training performance than training them simultaneously, as the agent is trained only on situations it actually and actively encounters, with a realistic frequency.

As a successful strategy regarding the exploration-exploitation problem to determine which action to execute, the currently best or some random other, a Boltzmann mechanism is used. By systematically decreasing its temperature, the selection gradually shifts from random to greedy.

The learning rate for the NG-neurons, their learning radius, the learning rate for the Q-values and the cooling of the Boltzmann-temperature, all undergo an exponential decline from starting value 1 up to the end value 0.01 over a certain number of steps (50,000 for learning rate and learning radius of the NG, 600,000 for Boltzmann-cooling and 750,000 for the Q-learning rate). Note that each NG-neuron features its individual Q-learning rate: the NG-neuron must be best matching for the situation to suggest an action (which here, consitutes a step) and subsequently adapt its Q-learning rate. All Q-values were initialized to 0.5. The discount factor γ was set to 0.8.

We used a world with one peak and (if appropriate) 5 valleys (one in each corner and one more centrally located, see Fig.1) with heights in the range $[-100, 100]$ on a $[0..40, 0..30]$ quasi-continuous gridworld. The target area was located around (10,10), i.e. explicitly not in the very middle of the grid, so as to exclude purely stochastic goal finding. For the Neural Gas, we used 50 neurons with random, uniformly distributed initial positions on the entire world. Each experiment was run at least 50 times.