Annals of DAAAM for 2010 & Proceedings of the 21st International DAAAM Symposium, Volume 21, No. 1, ISSN 1726-9679
ISBN 978-3-901509-73-5, Editor B. Katalinic, Published by DAAAM International, Vienna, Austria, EU, 2010
Make Harmony Between Technology and Nature, and Your Mind will Fly Free as a Bird
Annals of DAAAM International

MULTIPLE Q-UPDATE APPROACH FOR REINFORCEMENT LEARNING USING SIMILARITIES OF ADJACENT STATES

ALBERS, A[lbert]; SCHILLO, S[ebastian] M[athias] & FRIETSCH, M[arkus]

Abstract: *This paper presents a new machine learning approach, based on Reinforcement Learning (RL), to control a highly nonlinear robot demonstrator. An agent controls the movements of the robot by choosing torques for each joint and immediately receives a feedback signal. The implemented SARSA-agent uses an update rule that calculates the estimate of the current state-action by using the current Q-value, the received reward and the following state-action value. To accelerate the learning process, a novel Q-update approach using similarities of adjacent states resp. actions, is presented. In contrast to classic RL agents, where only previously visited states are updated, additional information are stored. By using this multiple update algorithm, additional knowledge is gathered in fewer episodes. The evaluation of this new approach shows that the integration of additional knowledge into the learning process dramatically increases the speed of finding a solution.*

Key words: *Reinforcement Learning, Robotics, Dynamics*

1. INTRODUCTION

Instead of using a conventional control system, the aim of this paper is to develop an advanced control system for a simplified humanoid robot manipulator using Reinforcement Learning algorithms. Machine learning techniques are able to find sophisticated solutions for problems where conventional control algorithms are faced with serious difficulties.

1.1 RL Overview

RL is based on the trial-and-error method: the system learns from gathered experience. To determine how good an action is the agent would have to choose it and observe the reward afterwards. Therefore a value is needed to specify how desirable it is for the agent to take a certain action in a state s. The value under a fixed policy can be computed as:

(1)

is called the action value function for policy . denotes the state, stands for the action at time step t (Watkins, 1989). denotes the expectation for policy . This type of update rule evolved from the idea of dynamic programming which was introduced by (Bellmann, 1954) and (Bertsekas et. al., 1996). In order to overcome local extremes, an optimal compromise between exploring and exploiting actions has to be found. This is achieved by picking random actions with a certain probability. E.g. a ε-greedy policy can be used to choose random action depending on a parameter ε and a random number. Learning is achieved by iteratively updating the Q-Table e.g. by using General-Policy-Iteration (GPI) presented in (Sutton and Barto, 1998). A good survey of RL can be found in (Kaelbling et. al., 1996) or (Smart and Kaelbling, 2002).

1.2 Two Link Planar Robot

In this approach, a humanoid robot manipulator should be controlled by a reinforcement learning agent. To show the feasibility of reinforcement learning methods to control manipulator systems, a 2-DOF-Manipulator system is used as a first simplified platform. A schematic drawing of the robot can be seen in Figure 1. The central goal of this paper is to move the robot manipulator from a start position to a target position. The aim is to create a smooth trajectory using as few control commands as possible.

Thus, the state of the manipulator is described by its two position angles and its two corresponding angular speeds which are discretized:

(2)

The system is described in more detail in (Denzinger and Laureyns, 2008)

2. Q-TABLE

Several problems occur when when a look-up-table is used. (Martin and De Lope, 2007). One major drawback is the large number of states. If an exploring action is chosen, the agent leaves the well-known path with the already well-estimated Q-Values. Hence, without the possibility of making a decision based on Q-Values, all actions are random and the agent does not have any orientation, even if it is very close to the target area. This increases the duration of an episode as well as it hinders a fast and intensive learning process. The basic idea behind the novel Q-update is to store more Q-Values per step and facilitate the agent to make more adequate choices.

If the distance from a specific query point is small enough, the action chosen in the query point is also well suited for a slightly varied state as a first approximation. Hence, the state variables are varied by a small amount Δ as indicated in equation (3):

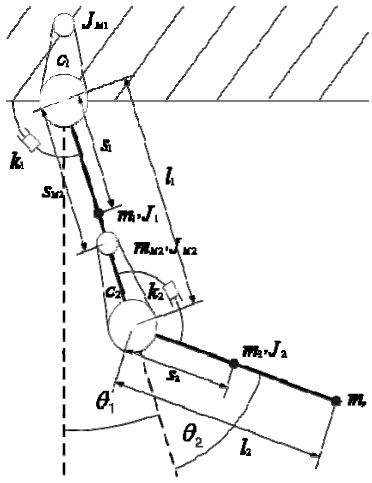

Fig. 1. Planar two link robot model

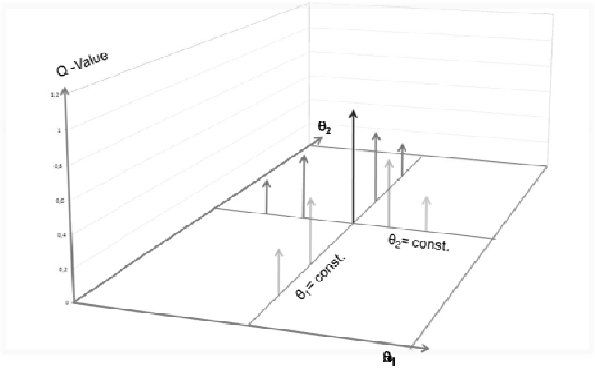

Fig. 2. Visualization of the novel Q-update method

(3)

Only one state variable is changed while the three others (indicated by index j) remain constant. The indeces i, j denote the position of the state variable. The variation is equal to the discretization step size of the state entries —— for . For the varied state , a different Q–Value is stored. This new Q-Value has a smaller value than the original one because the confidence is smaller. The variation of the state by increments of the discretization step size is necessary to ensure that the new value is stored next to the old one instead of overwriting it. Fig. 2 shows a visualization of the basic idea of this method: in addition to the visited state in the middle, adjacent states are updated using a discount factor in order to represent a less confidential update. Note that the Q-Table is a 5-dimensional matrix. The table's velocity , and action entries are considered to be constant in order to display the Q-Values and the angle's , . Furthermore the Q-Values are scalars in the general case; for better readability an additional dimension for is introduced in Fig. 2.

3. EVALUATION

Different points, evenly distributed over the workspace, are chosen to compare the novel approach to two classic Reinforcement Learning agents. All agents use the Elegibility-Traces presented in (Sutton and Barto, 1998). Apart from a geometrical balanced distribution, the distance between start and target position were taken into account as well as the type of movement, such as straight or diagonal. This was done to obtain a profound and clear decision basis.

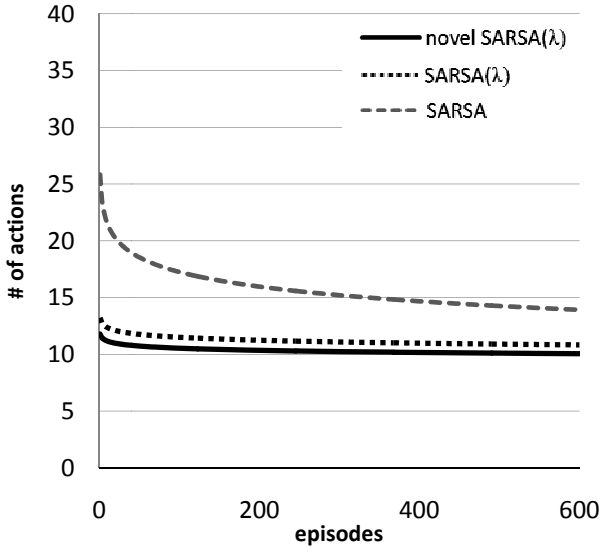

Fig. 3. Comparison of different agents

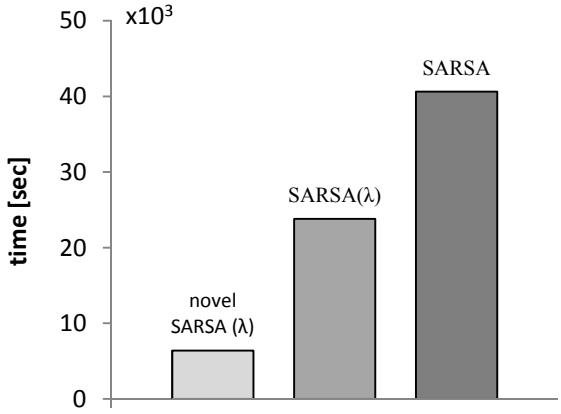

Fig. 4. Elapsed time for learning of different agents

The quality of the solution found by the novel_SARSA(λ) and SARSA(λ) agents is very high at the start of the RL task. That is, the agents need a low number of actions to reach the target position. SARSA's number of actions decreases constantly although the learning process is less intensive compared to the versions of SARSA(λ). The performance of the SARSA(λ) version is a trade-off between good quality of the solution and time. If the number of actions is higher, the agent is reset to the starting position. In addition, the higher computational effort elongates the learning process. To sum up: the novel_SARSA(λ) is the most successful agent, concerning quality as well as the elapsed time.

4. CONCLUSION

A novel update algorithm for Reinforcement Learning agents based on exploiting similarities of adjacent states and actions was presented. An evaluation of the new approach was conducted using point-to-point movements of a 2-DOF manipulator model. The comparison of the novel update algorithm to two classic RL agents clearly showed better results regarding elapsed time and quality. Influences of the new approach on the exploration characteristics of the learning process are object of future research.

5. REFERENCES

Bellman R. (1954). The theory of dynamic programming, *Bulletin of the American Mathematical Society* , 60, 503-515

Bertsekas D. P. & Tsitsiklis J. N. (1996). *Neuro-Dynamic Programming*, Athena Scientific, Belmont, MA

Denzinger J.; Laureyns I. & et.al. (2008). A study of reward functions in reinforcement learning on a dynamic model of a two-link planar robot, The 2nd European DAAAM International Young Researchers´ and Scientists´ Conference

Kaelbling L. P.; Littman M.L. & Moore A. (1996). Reinforcement Learning: A Survey, *Journal of Artificial Intelligence Research* 4 , 237-285

Martin J. A. & De Lope J. (2007). A Distributed Reinforcement Learning Architecture for Multi-Link Robots, *4th International Conference on Informatics in Control, Automation and Robotics* (ICINCO), (pp. 192-197). Angers, France

Smart W. D. & Kaelbling L. (2002). Making Reinforcement Learning Work on Real Robots, *Ph.D. Thesis*, Brown University

Sutton R.S & Barto A.G. (1998). Reinforcement Learning, an introduction, *The MIT press*, MA

Watkins C. (1989). Learning from Delayed Rewards, *Thesis*, University of Cambidge, England