Autonomous Vehicle Steering based on Evaluative Feedback by Reinforcement Learning

Klaus-Dieter Kuhnert[1], Michael Krödel[1]

[1] University of Siegen, Institute of Real-Time Learningsystems,
Hölderlinstrasse 3, D-57068 Siegen / Germany
kuhnert@fb12.uni-siegen.de

Abstract. Steering an autonomous vehicle requires the permanent adaptation of behavior in relation to the various situations the vehicle is in. This paper describes a research which implements such adaptation and optimization based on Reinforcement Learning (RL) which in detail purely learns from evaluative feedback in contrast to instructive feedback. Convergence of the learning process has been achieved at various experimental results revealing the impact of the different RL parameters. While using RL for autonomous steering is in itself already a novelty, additional attention has been given to new proposals for post-processing and interpreting the experimental data.

1. Introduction

The study presented in this paper deals with the concept and the implementation of a system which, based on experience over a period of time, is able to autonomously learn to steer different vehicles and to optimise its behaviour to various possible road courses. This shall be done in a different way than researched in many other works before as described further below.

Key element is the fact that any action (steering, acceleration) is dependent on the situation to which a vehicle is exposed. If a vehicle is exposed to a real environment, situations are subject to permanent changes and therefore any true autonomous system will have to continuously adapt its actions.

Many research projects have been performed based on neural nets and have shown some results, but were always dependent on strong similarities between current environment and previous training pattern. A new situation always needs to be trained if deviating even slightly from previously trained situations.

The model based approach proved better success and is still being pursued in many researches. Even though we also believe in its further success, the parameterisation becomes more and more complex when the number of different situations increases (e.g. when situations are being further examined). This remains the biggest challenge for some time. In this light, an interesting variation has been proposed by using a neural net for learning the parameters of the used model [13], [14].

Altogether, however, both directions are dependent on instructive feedback – therefore they are based on a-priori knowledge resulting from parameters of teaching phases.

This paper therefore describes the research of a third method: Reinforcement Learning (RL). RL-Systems provide capabilities of self-optimising actions based on evaluative feedback. They explore the overall state-space by means of analysing the impact of previously issued actions, coping with delayed feedback as well as coping with disturbed feedback.

Given the above aspects, it should also be noted that RL is not striving to compete with the established approaches like modelling. In lieu thereof, any progress of RL-Systems might be used to enhance the advantages of modelling achieved so far. At the end, a combined system built on modelling and RL might provide better results than each approach alone. In this light, we strongly believe RL-system will play a significant role in the near future in autonomous driving systems.

This paper, however, focuses purely on Reinforcement Learning in order to explore its benefits and limitations.

All in all, the main targets of this research are: steering of an autonomous vehicle along any curvy road, autonomous exploration of new actions for familiar as well as for new situations, therefore autonomous optimization (self-tuning of the system to any combination of environment and vehicle), learning from evaluative feedback (in contrast to instructive feedback/ teaching), coping with delayed feedback (delayed rewarding) as well as non-linearity of true environment, and finally Real-time processing.

2. Related work

Till now the visual control of systems for autonomous vehicle driving with learning components have been implemented in several ways. [2] describes a short direct connection between image processing and soft computing learning method using a neural network. This approach provides good results but only as long as input pictures of the scene are similar to the training patterns. This approach was being enhanced by a multiple neural network [3], but could not completely solve the dependency problem of the taught training patterns. Further developments then included a GPS system [4] to support orientation or enhanced the approach with object-oriented vision in order to distinguish between road following and obstacle detection [5], [6]. In all those variations, however, neural networks with their inherent dependency on training patterns are embedded. Also, as a major difference to the presented research, the established knowledge on vehicle driving is stored within the neural net but not explicitly available, e.g. for optimisation or for further learning processes.

A completely different approach is being followed by using explicit modelling, therefore trying to rebuild a model of the environment as well as the vehicle and to derive proper actions from it. The basic idea of such a model is to try to understand interaction between vehicle and environment and to predict consequences of any behaviour thus allowing to determine a suitable behaviour in a given situation.

The major challenge of this approach is to find a suitable model which approximates the true vehicle behaviour and environment in the best way. Any difference between the model and the real environment/vehicle results in a difference between the calculated behaviour and an optimum behaviour. Any model also needs to be shaped up and tuned with parameters. Usually there are no versatile models, so any change of e.g. vehicle or environment requires a corresponding tuning, respectively adaptation of the model. In other words, any tuned model is valid only for a certain environment or vehicle and is more or less sensible to any change of these. [7] describes an early success with international attention of a vehicle system using a real-time vision system BVV2 [8].

Further developments in this area (e.g. [9]) are being pursued with significant progress, however always dependent on many parameters for the modelling process.

3. General Structure

Figure 1 shows the overall structure of our approach. According to Sutton/Barto [1], a RL-system consists of an RL-Agent and theRL-Environment. The RL-Agent receives input regarding the state (situation) s_t as well as a reward r_t and determines an appropriate action a_t. This action will cause a reaction of the RL-Environment and consequently result in a change of state from s_t to s_{t+1}. Similarly, the RL-Environment will also issue a reward r_{t+1} corresponding to s_{t+1}.

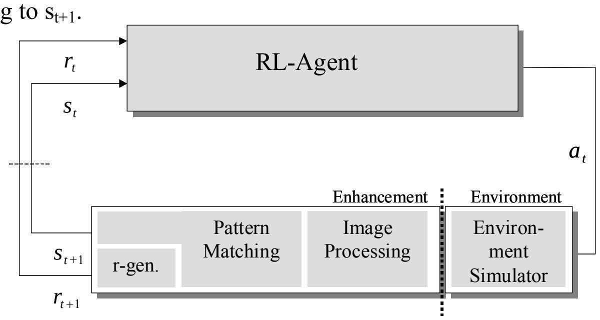

Fig. 1. Structure of the System

Since the determination of the state s and the reward r is required from the RL-Environment and usually not being provided by an environment simulator, our system enhances the RL-Environment and provides methods of Image Processing, Pattern Matching and reward-generation being described more in detail in the next paragraph.

4. Image Processing and Pattern Matching

The proper pre-processing of the incoming data is key to any successful RL-System. One of the major novelties of this research is the determination of a suitable situation description in correspondence to the situation the vehicle is in. In this light, the classical RL-Environment (the lower part of figure 1) has been enhanced in order to provide the RL-agent with defined descriptions of each situation. Any incoming image, along with external information on any appropriate action, is being given to an image processing system, which extracts all relevant information in order to describe the current situation. Such situation description is being referred to as Abstract Complete Situation Description (ACSD). Even though such technique is a significant part of the current research, it shall not be described at this point since public presentations (also available on the web) have been done and has been described in proceeding papers in detail (e.g. [10], [11], [12]). At this point it shall only be emphasized that it makes use of a self-created statistical database storing the conditional probabilities of road mark positions and additionally exploiting the information to extract the road marks faster and more reliable than with many other methods. Such ACSDs are then being stored along with the corresponding action a of the training phase in a database.

When operating the system in driving mode, any incoming image is being converted into an ACSD. Given the ACSD of the current image and the ACSD's in the database, a fast k-nearest neighbour algorithm locates the most similar ACSDs. Such way, the RL-Agent not only receives information regarding the current situation but also information which other similar (or identical) situations experienced before.

In this context, the ACSD explained above is being used as the state s, the action a is basically the steering command (i.e. angle of the steering wheel).

Additionally, a reward r is being determined, which can be a value representing the lateral deviation from the middle of the road if the agent has to learn to follow any road course – however, the reward can also be a timer value measuring the time needed for a road section if the agent is to learn to pass a road section in the shortest possible time.

5. Reinforcement Learning

The basic idea of RL is, that states $s_{t,}$, respectively actions issued at a certain state a_t, are being rated considering the reward r_{t+1} of the situation s_{t+1}. Such rating is being represented by Q(s,a) and is defined to be the sum of future rewards discounted by a discount factor γ. In the beginning only estimates of the q-values exist. Thus, the current Q-values deviate from the converged Q-values by an error TDerr.

$$Q(s_t, a_t) = r_{t+1} + \sum_{i=2}^{\infty} \gamma^i r_{t+i} - TDerr \qquad (1)$$

$$Q(s_t, a_t) = r_{t+1} + \gamma \cdot Q(s_{t+1}, a_{t+1}) - TDerr \qquad (2)$$

$$TDerr = r_{t+1} + \gamma \cdot Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t) \qquad (3)$$

The error TDerr is being used for updating the Q-values (also discounted by the learning rate parameter α) and will lead to a convergence of the Q-values, as long as the reward is deterministic.

$$Q(s_t, a_t) := Q(s_t, a_t) + \alpha \cdot TDerr \qquad (4)$$

The maximum Q-value of a state s across all possible actions shall be:

$$Q_{\mathrm{a-max}} = \max_a Q(s_i, a) \qquad (5)$$

and in combination with the policy π, the system usually selects the action with the highest Q-value, resulting the system to operate in the mode called exploitation mode:

$$\pi(s_i): a = a(Q_{a-\max}) \tag{6}$$

An initial target, however, is to self-optimise behaviour over time. Consequently it is imperative to actively explore the state-action-space in order to search for the best action (and temporarily switching to exploration mode):

$$\pi(s_i): \begin{cases} a = a(Q_{a-\max(i)}) \text{ for } rand() \geq \varepsilon \\ a = rand() \text{ else} \end{cases} \tag{7}$$

with $rand() \in [0,1], \varepsilon \in [0,1]$

In this policy learning and exploitation are randomly mixed. Such way the RL-system also adapts autonomously to a new or even changing environment without any explicit training phase.

Notable, at this point, is also the capability of a RL-system to cope with non-linearity's (e.g. errand rewarding) of the environment. This notation also includes the ability of the system to cope with delayed rewards. Given is for example the state-action relationship as displayed in figure 2. At t=0 the system shall be in state s_i and has the option between two actions: $a_{i\rightarrow j}$ which will cause a transition to state s_j and further-on to state s_k or action $a_{i\rightarrow i}$ which will prevent any change of state. r_j shall be errand and r_k shall be much higher than r_i. Therefore, the system should be able to learn to accept an errand (low) temporary reward at state s_j but to finally reach s_k and should not remain in state s_i.

According to its policy, the system will choose the action with the highest Q-Values. Depending on the rewards and the discount factor, the maximum Q-value at state s_i is given in formula 8. Basically, the closer the discount value get towards "1", the higher will be the preference on long-term-rewards instead of short-term rewards.

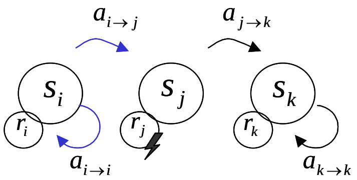

Fig. 2. State sequence of the RL-system while coping with disturbed reward r_j(see flash)

$$Q_{a-\max(i)} = \max\{\frac{r_i}{1-\gamma}; r_j + \gamma\frac{r_k}{1-\gamma}\} \tag{8}$$

6. Experimental results and findings

6.1 Experimental setup

The experiments have been done with a closed-loop-system consisting of two connected computers. The System-PC, processes the video stream of a connected camera and calculates the steering commands for the vehicle. These steering commands are then being given to the Simulator-PC, which is responsible for the environment simulation. A converter box connects both interfaces. The output of the second computer is being given onto its monitor, which again is being read by the video camera – alternatively, the video output of the Simulator-PC is being connected directly to the framegrabber of the system-PC using a S-VHS-cable. Due to this set-up a realistic amount of measurement noise is introduced into the system.

The task of the following experiments has been, to learn the ideal vertical position, respectively, the driving angle of a vehicle driving along a road course. In this light, a simplified system with 11 possible steering commands (equally distributed from sharp left steering to sharp right steering) has been defined. The number of possible situations varies depending on the settings of the image processing part.

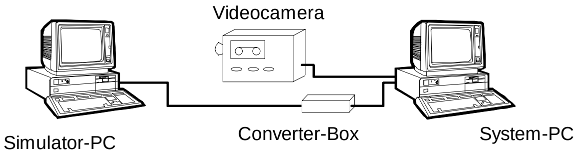

Fig. 3. HW setup for this research using two interconnected computers

At this point it should be noted that all further results have been achieved without any supervised teaching at all! Therefore, the system discovers the whole state space completely on its own – in detail: the appropriateness of every action of every situation. Such extreme exploration of the environment is only possible on a simulator, which is our main reason for choosing such a platform.

6.2 Splining and measurement of convergence

One of the major new and significant findings in this research was, that a criterion is needed as to how much the system converged. Even though the values of TDerr (formula 3) represent the convergence error and is therefore the basis for the update, the chart of TDerr(t) does not express the grade of state-space convergence. Fig. 4 shows lateral converged state-space (optimal lateral position to be learned) after the issuance of approx. 170.000 actions and a chart of TDerr over time – the convergence is not really be recognizable.

Regarding the state-space: the state “R1” is equivalent to being at the left edge of the road; the best action in this situation is the selection of the center situation. The state “R11” is equivalent to being at the right road edge; the best action in this situation is again the selection of the center situation. The action “1” is equivalent to selecting the leftmost situation, the action “11”is equivalent to selecting the rightmost situation.

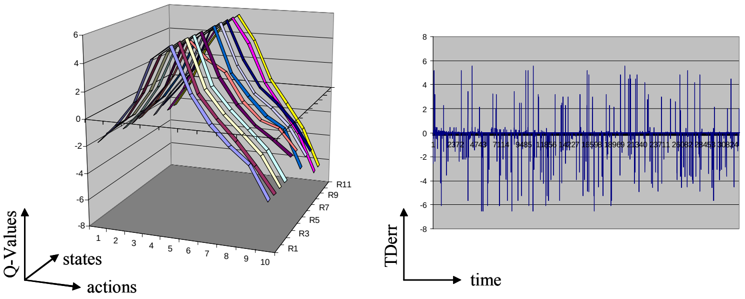

Fig. 4. original state-space (approx 30.000 actions) and TDerr (t)

Therefore the state-space of the Q-values is being approximated by calculating for all situations each one spline over all actions. The cumulated squared difference between all original Q-Values and it’s corresponding splined Q-Value results in the determination of a value “Smoothness”. The splined Q-Values as well as the development of the Smoothness Value during the same testseries as Fig. 4 is shown in Fig. 5 and a clear indication for convergence can be seen. A rising smoothness value indicates the adaptation of the system to its environment (major learning performed), because the action space has to be globally smooth for the chosen system. The smoothness value decreases while the system converges. A complete convergence (therefore smoothness-value equal to zero) will not be achieved since any true environment is also not absolutely deterministic.

However, a disadvantage of the splining, is the distortion of the state-space if some actions did not get issued often enough – resulting in a less often update according to Reinforcement Learning. Fig. 6 shows the original Q-Values as well as the splined Q-Values for a state-space, in which only the middle actions got issued often enough (resulting in a local convergence). As a solution to this dilemma, the number of updates for each Q-Value gets counted and can either be considered during the splining process or used to hide the associated Q-Values.

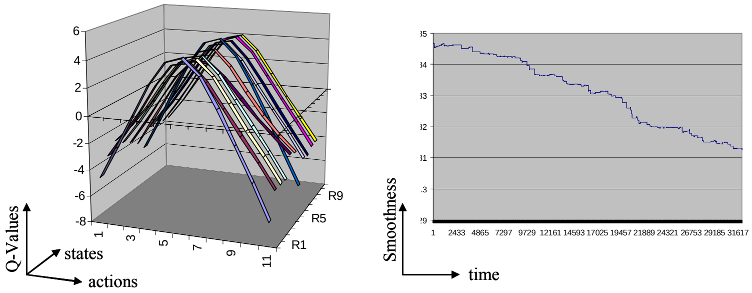

Fig. 5. splined state-space (approx 30.000 actions) and Smoothness(t)

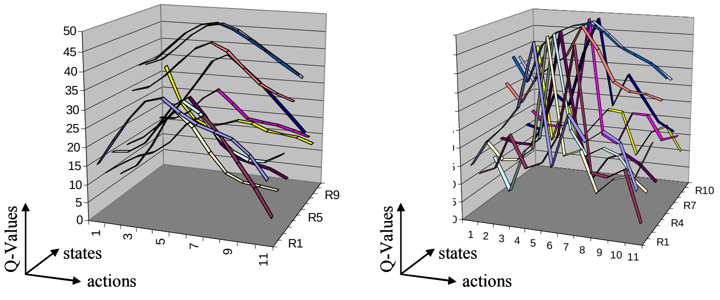

Fig. 6. original state-space and splined state-space

Fig. 6 also shows the impact of reduced exploration. At reduced exploration, some actions might nearly never get issued (since not any action can be issued from any situation, creating a situation-specific dependency). Partially, this can be overcome by longer test-cycles but still, the counted number of updates for each Q-Values needs to be considered for any further analysis.

6.3 Impact of learning parameters

Although some other publications deal with the combination of RL and autonomous driving, the impact of the RL parameters are not yet publicly documented. In consequence, quite some experiments have been spent on such topic and provide for the first time an overview of the impact of the basi RL-parameters. Regarding the learning rate parameter α on the learning process, Fig. 7 and Fig. 8 show two similar testseries which differ only in values of α. A small value for α results in slower, but more stable learning. It should be noted that for those and the further tests, the system had to learn the driving angle, ie. the optimal driving angle is dependant on the situation the vehicle is in resulting in a different position of the maximum Q-Value for each situation.

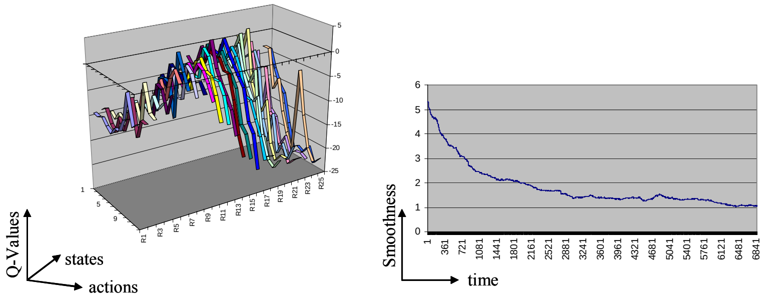

Fig. 7. testseries (approx 7.000 actions) with $\alpha = 0.1$; original state-space and Smoothness(t)

An environment with a higher number of situations lead to a more complex state space. Fig. 9. show corresponding tests; again with different settings for the grade of exploration. All in all, the system performs the learning task quite well – especially, as mentioned above, without any teaching at all. The more complex the environment becomes (dimension of state-space increasing) the test duration needs to be enhanced accordingly. However, even extended test times might run into limitations when the environment gets more and more complex. In those cases, focused exploration (i.e. exploration of only some sub-areas of the whole state-space) are supposed to be a viable solution – further investigation on this matter is planned for the near future.

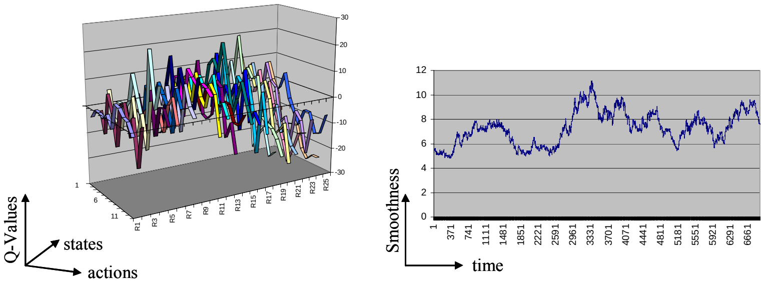

Fig. 8. testseries (approx. 7.000 actions) with α = 0.5; original state-space and Smoothness(t)

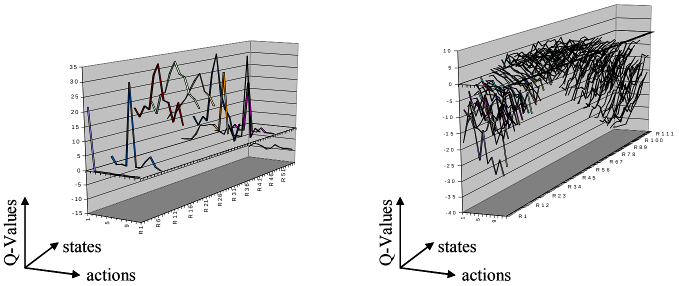

Fig. 9. Impact of exploration: ε = 0,1 (left) resp. ε = 1,0 (right)

7. Summary

Pattern Matching provides capabilities of autonomous driving with knowledge being directly accessible (for further optimization). In addition, Reinforcement Learning allows autonomous optimization of behaviors based on self-created rewards, even if delayed or disturbed. Combining both techniques allows learning and optimizing of visual steering of autonomous vehicles. The current research, will now be further used in more complex environments in order to explore the limiations of exploration in combination to test duration. Also, further aspects regarding coping with delayed rewards will still be focussed on within the current research.

References

[1] Richard Sutton, A. G. Barto, Reinforcement Learning: An introduction, MIT-Press, 2000, Cambridge (USA)

[2] D. A. Pommerleau, Efficient Training of Artificial Neural Networks for Autonomous Navigation, Neural Computation 3, 1991

[3] T.M. Jochem, D.A. Pomerleau, C.E. Thorpe. MANIAC: A Next Generation Neurally Based Autonomous Road Follower, IAS-3, Int. Conference on Intelligent autonomous Systems, February 15-18, 1993, Pittsburgh/PA, USA, F.C.A. Groen, S.Hirose, C.E.Thorpe (eds), IOS Press, Washington, Oxford, Amsterdam, Tokyo, 1993

[4] T.M.Jochem, D.A.Pomerleau, C.E.Thorpe, Vision Guided Lane Transition, Intelligent Vehicles '95 Symposium, September 25-26, 1995, Detroit/MI, USA

[5] S.Baluja, D.A.Pomerleau, Expectation-based selective attention for visual monitoring and control of a robot vehicle, Robotics and Autonomous System, Vol.22, No.3-4, December, 1997

[6] Uwe Franke, Dariu Gavrilla, Steffen Görzig, Frank Lindner, Frank Paetzold, Christian Wöhler, Autonomous Driving Goes Downtown, IEEE Intelligent Vehicles Systems, v.13 n.6, p.40-48, November 1998

[7] E.D.Dickmanns, A.Zapp, Autonomous High Speed Road Vehicle Guidance by Computer Vision, Preprints of the 10th World Congress on Automatic Control, Vol.4, International Federation of Automatic Control, Munich, Germany, July 27-31, 1987

[8] K.-D.-Kuhnert, A Vision System for Real Time Road and Object Recognition for Vehicle Guidance, Proc. Mobile Robots, Oct 30-31, 1986, Cambridge, Massachusetts, Society of Photo-Optical Instrumentation Engineers, SPIE Volume 727

[9] E.D.Dickmanns, R.Behringer, D.Dickmanns, T.Hildebrandt, M.Maurer, F.Thomanek, J.Schiehlen, The Seeing Passenger Car 'VaMoRs-P', Intelligent Vehicles '94 Symposium, October 24-26, 1994, Paris, France

[10] M. Krödel, K.-D. Kuhnert, Pattern Matching as the Nucleus for either Autonomous Driving or Drive Assistance Systems, IEEE Intelligent Vehicle Symposium, June 17-21, 2002, Versailles, France

[11] K.-D. Kuhnert, M. Krödel, Reinforcement Learning to drive a car by pattern matching, Anual symposium of Pattern recognition of DAGM, September 16-18, 2002, Zurich (Switzerland)

[12] K.-D. Kuhnert, M. Krödel, Autonomous Driving by Pattern Matching and Reinforcement Learning, International Colloquium on Autonomous and Mobile Systems, June 25-26, 2002, Magdeburg, Germany

[13] K.-D. Kuhnert, W. Dong, Über die lernende Regelung autonomer Fahrzeuge mit neuronalen Netzen, 18. Fachgespräch Autonome Mobile Systeme (AMS), December 4-5, Karlsruhe, Germany

[14] W. Dong, K.-D. Kuhnert, Robust adaptive control of honholonomic mobile robot with parameter and non-parameter uncertainties, IEEE Transaction on Robotics and Automation, 2004