Informatik an der HAW-Hamburg

FAUST·onyx

Steuerungsoptimierung eines autonomen Fahrzeugs mittels Reinforcement Learning

AW2 Präsentation SS2010

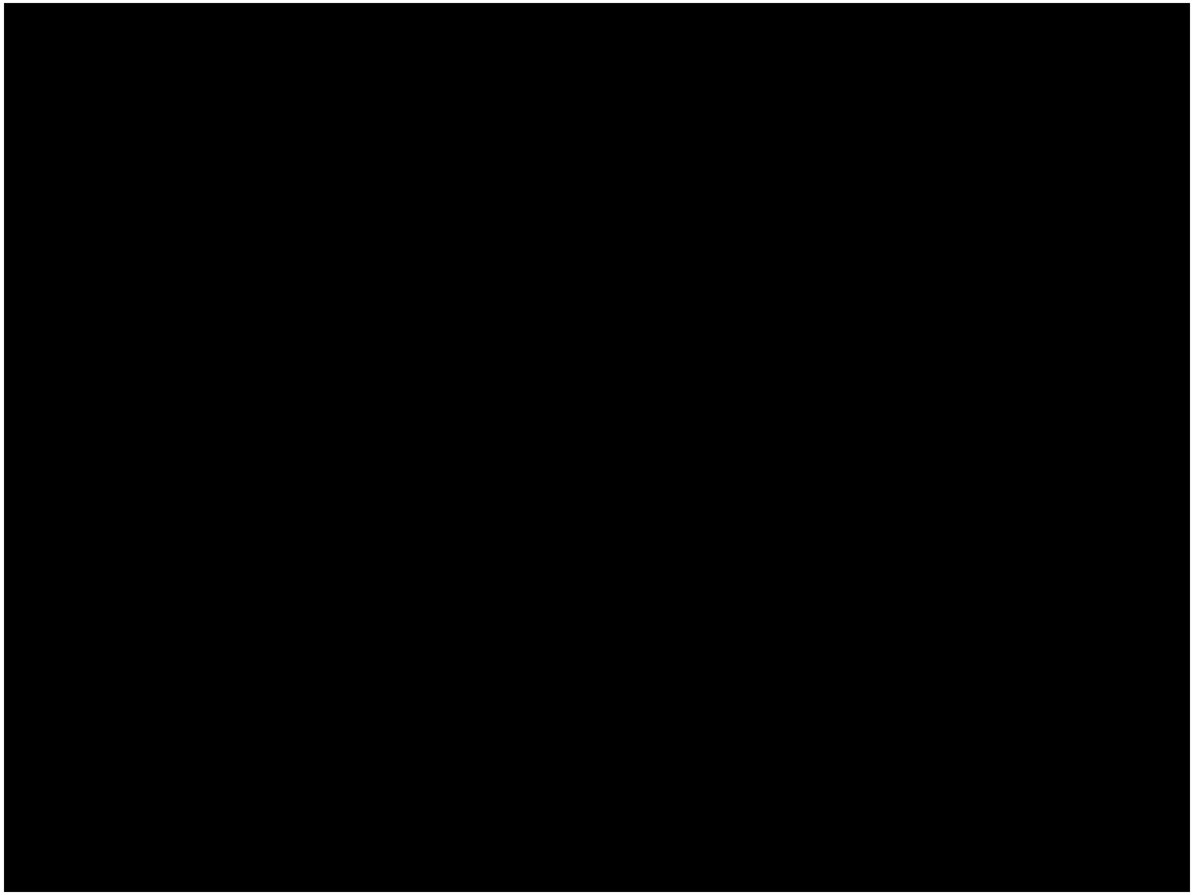
Ivo Nikolov

ivo.nikolov@haw-hamburg.de

Agenda

- Motivation
- Related Work
  - Learning to Drive a Real Car in 20 Minutes
  - High Speed Obstacle Avoidance using Monocular Vision and Reinforcement Learning
  - An Application of RL to Aerobatic Helicopter Flight
- Zusammenfassung und Ausblick

Motivation



Agenda

- Motivation
- Related Work
  - Learning to Drive a Real Car in 20 Minutes
  - High Speed Obstacle Avoidance using Monocular Vision and Reinforcement Learning
  - An Application of RL to Aerobatic Helicopter Flight
- Zusammenfassung und Ausblick

Learning to Drive a Real Car in 20 Minutes

- Autoren:
  - Martin Riedmiller: *Neuroinformatics Group, Univ. of Osnabrueck*
  - Mike Montemerlo und Hendrik Dahlkamp: *AI Lab, Stanford University*
- Vorgestellt:
  - Proccedings of the FBIT 2007 conference, Jeju, Korea
- Ziel:
  - steer a real robot car employing Reinforcement Learning (based on Neural Fitted Q Iteration)

Learning to Drive a Real Car in 20 Minutes

- Q Learning and Neural Networks pros
  - approximate a Q-Function for continuous state spaces
  - nonlinear functions can be better approximated
- cons
  - theoretical convergence do not hold any more, although a lot of successful applications have been reported
  - each update for one state-action pair might induce unforeseeable changes at the Q-values for other state-action pairs

Learning to Drive a Real Car in 20 Minutes

Fitted Q Iteration

Inputs: a set of four-tuples $\mathcal{F}$ and a regression algorithm.
Initialization:
Set N to 0 .
Let $\hat{Q}_N$ be a function equal to zero everywhere on $X \times U$.
Iterations:
Repeat until stopping conditions are reached

- $N \leftarrow N+1$.

- Build the training set $\mathcal{TS} = \{(i^l, o^l), l = 1, \cdots, \#\mathcal{F}\}$ based on the the function $\hat{Q}_{N-1}$ and on the full set of four-tuples $\mathcal{F}$:

$$\begin{aligned} i^l &= (x_t^l, u_t^l), \\ o^l &= r_t^l + \gamma \max_{u \in U} \hat{Q}_{N-1}(x_{t+1}^l, u). \end{aligned}$$

- Use the regression algorithm to induce from $\mathcal{TS}$ the function $\hat{Q}_N(x,u)$.

Learning to Drive a Real Car in 20 Minutes

- Neural Fitted Q Iteration (NFQ)
  - update the Q Function off-line considering the entire set of transition experiences
  - more advanced supervised learning techniques can be used
- Learning cycle
  - data is collected as (s , a, s`)
  - NFQ approximates a Q Function
  - greedy exploit the new Q Function (exploration can be added)

Learning to Drive a Real Car in 20 Minutes

- The RL Controller for Steering
  - costs

$$c(s,u) = \begin{cases} 0 & , \text{ if } |cte| < 0.05m \text{ (success)} \\ +1 & , \text{ if } |cte| > 0.5m \text{ (failure)} \\ 0.01 & , \; else \end{cases}$$

  - inputs: cross track error, velocity, heading error …
  - Q function and policy

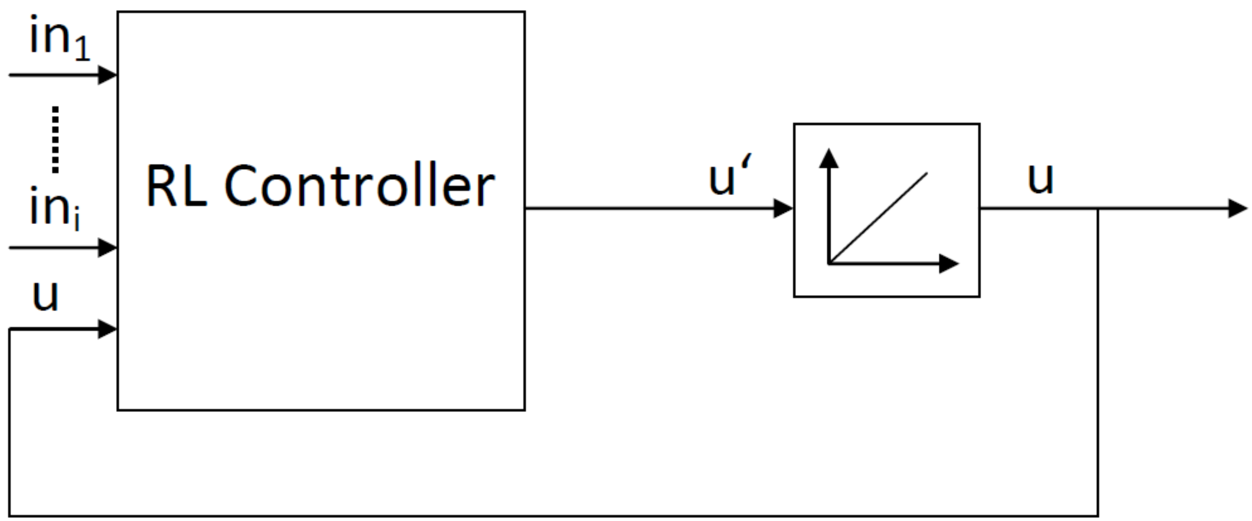
$$\pi(s) = \arg\min_u Q(s,u)$$

  - a discretisation of actions with I-DOE

$$U' = \{\pm 60^o, \pm 10^o, 0\}$$

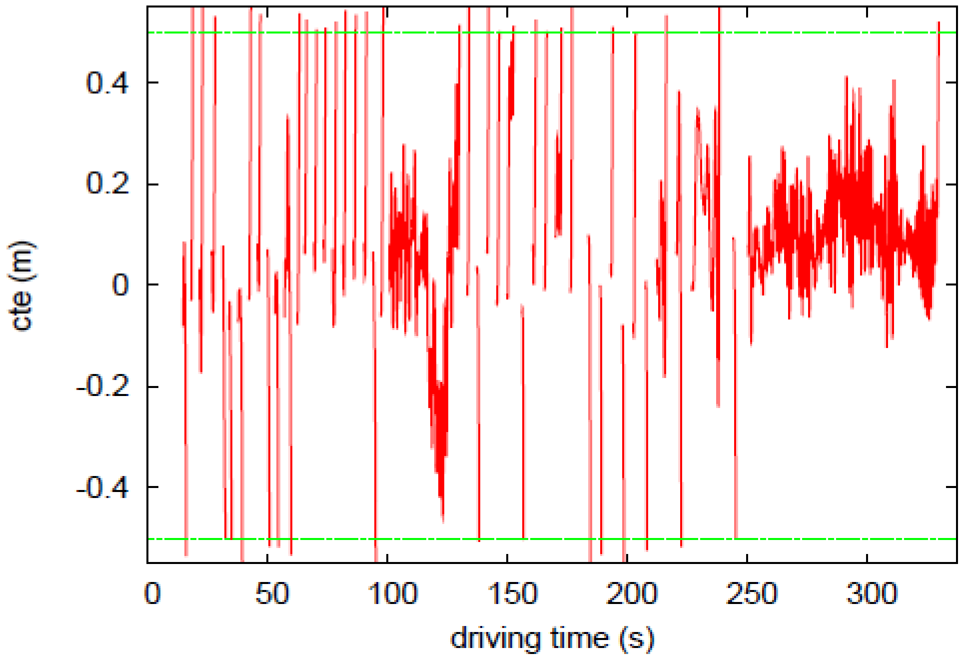
Learning to Drive a Real Car in 20 Minutes

I-DOE (dynamic output elements)



Learning to Drive a Real Car in 20 Minutes

- Results



Agenda

- Motivation
- Related Work
  - Learning to Drive a Real Car in 20 Minutes
  - High Speed Obstacle Avoidance using Monocular Vision and Reinforcement Learning
  - An Application of RL to Aerobatic Helicopter Flight
- Zusammenfassung und Ausblick

High Speed Obstacle Avoidance …

- Autoren:
  - Jeff Michels, Ashutosh Saxena, Andrew Y. Ng: *Computer Science Department, Stanford University, Stanford*
- Vorgestellt:
  - Proceedings of the 22nd International Conference on Machine Learning, Bonn, Germany, 2005
- Ziel:
  - driving a remote control car at high speeds through unstructured outdoor environments

High Speed Obstacle Avoidance …

- model the RC car control problem as a Markov decision process (MDP) with

$$R(s) = -|v_{desired} - v_{actual}| - K.Crashed$$

- learn the parameters of the control policy using PEGASUS policy search algorithm

θ_1: σ of the Gaussian used for spatial smoothing of the predicted distances
θ_2: if $\hat{d}_i(\alpha_{chosen}) < \theta_2$, take evasive action rather than steering towards α_{chosen}
θ_3: the maximum change in steering angle at any given time step
θ_4, θ_5: parameters used to choose which direction to turn if no location in the image is a good steering direction (using the current steering direction and the predicted distances of the left-most and right-most stripes of the image).
θ_6: the percent of max throttle to use during an evasive turn

High Speed Obstacle Avoidance …

PEGASUS

- transform a MDP into an "equivalent" one that has only deterministic transitions
- find a good policy π of a policy class ∏ for the transformed MDP
- if the action space is continuous and ∏ is a smoothly parameterized family of policies, a gradient ascent methods can be used to optimize π

Agenda

- Motivation
- Related Work
  - Learning to Drive a Real Car in 20 Minutes
  - High Speed Obstacle Avoidance using Monocular Vision and Reinforcement Learning
  - An Application of RL to Aerobatic Helicopter Flight
- Zusammenfassung und Ausblick

An Application of RL to Aerobatic Helicopter Flight

- Autoren:
  - Pieter Abbeel, Adam Coates, Morgan Quigley, Andrew Y. Ng: *Computer Science Department, Stanford University, Stanford*
- Ziel:
  - autonomous completion on a real RC helicopter of four aerobatic maneuvers

An Application of RL to Aerobatic Helicopter Flight

- Learning a Model
  - Data Collection using the Apprenticeship Learning Algorithm
  - Learn a Model that predicts accelerations as a function of the current state and inputs
- Controller Design
  - Linear quadratic regulator (LQR)
  - Differential dynamic programming (DDP)

An Application of RL to Aerobatic Helicopter Flight

Linear quadratic regulator (LQR) control problem is a special class of MDPs*(S, A, T, H, s(0), R)* and a dynamics model is given by:

$$s(t+1) = A(t)s(t) + B(t)u(t) + w(t)$$

The reward for being in state s(t) and taking action/input u(t) is given by:

$$-s(t)^T Q(t)s(t) - u(t)^T R(t)u(t)$$

An Application of RL to Aerobatic Helicopter Flight

Differential dynamic programming (DDP) approximately solves general continuous state-space MDPs by iterating the following two steps:

1. Compute a linear approximation to the dynamics and a quadratic approximation to the reward function around the trajectory obtained when using the current policy.

2. Compute the optimal policy for the LQR problem obtained in Step 1 and set the current policy equal to the optimal policy for the LQR problem.

Agenda

- Motivation
- Related Work
  - Learning to Drive a Real Car in 20 Minutes
  - High Speed Obstacle Avoidance using Monocular Vision and Reinforcement Learning
  - An Application of RL to Aerobatic Helicopter Flight
- Zusammenfassung und Ausblick

Zusammenfassung und Ausblick

- Related Work
- Ausblick
  - Projekt 1
  - Proekt 2

Vielen Dank für die Aufmerksamkeit!

References

[1] Martin Riedmiller, Mike Montemerlo, Hendrik Dahlkamp: Learning to Drive a Real Car in 20 Minutes , Proccedings of the FBIT 2007 conference, Jeju, Korea

[2] Damien Ernst, Pierre Geurts, Louis Wehenkel: Tree-Based Batch Mode Reinforcement Learning

[3] Martin Riedmiller: Neural Fitted Q Iteration - First Experiences with a Data Efficient Neural Reinforcement Learning Method

[4] Je Michels, Ashutosh Saxena, Andrew Y. Ng: High Speed Obstacle Avoidance using Monocular Vision and Reinforcement Learning. Proceedings of the 22nd International Conference on Machine Learning, Bonn, Germany, 2005

[5] Andrew Y. Ng, Michael Jordan: PEGASUS: A policy search method for large MDPs and POMDPs

[6] Pieter Abbeel, Adam Coates, Morgan Quigley, Andrew Y. Ng: An Application of Reinforcement Learning to Aerobatic Helicopter Flight

[7] Brian D. O. Anderson, John B. Moore: Optimal Control-Linear Quadratic Methods

[8] Stanford University, Autonomous Helicopter Project: http://heli.stanford.edu/

[9] Carolo Cup 2010 Faust Videos: http://www.youtube.com/results?search_query=carolo+cup+2010+faust&aq=f