HACKATHON: LECTURE

Optimization and Machine Learning for Automated Cars

Prof. Dr. habil. Vadim Azhmyakov

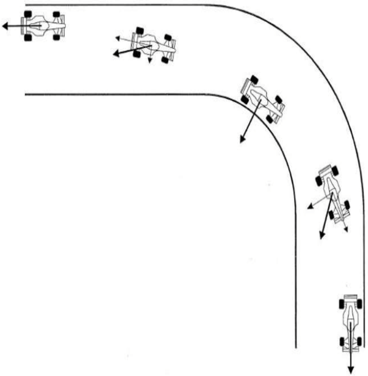
DOCET TI
and
National Research University Higher School of Economics
Moscow, Russia

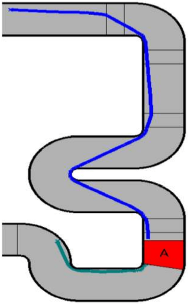
Moscow,
December 2021

Outline

Introduction: Optimization and Automated Automotive Industry

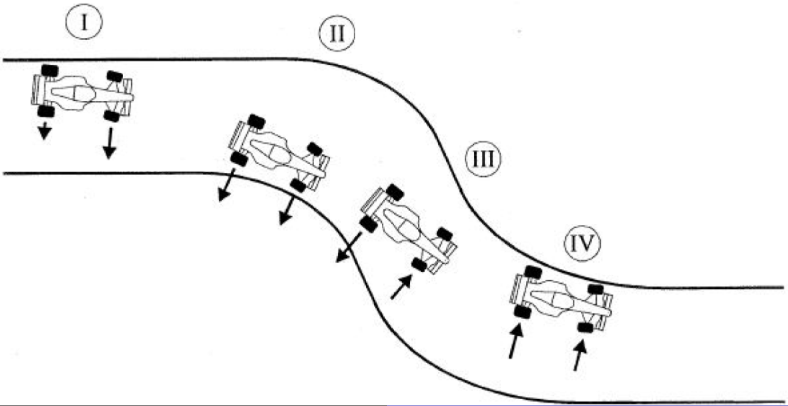
how to follow the given track as fast as possible?





Introduction: Optimization and Automated Automotive Industry

a closer tracking during a curvy maneuver $\Rightarrow$ loss of the velocity



Introduction: Optimization and Automated Automotive Industry

two main challenges in automotive development

- Modern automated (intelligent) cars;
- Machine Learning (ML) algorithms for the trajectory optimization (smart driving).

types of the optimization approaches

- A priori - an optimal path finding $=$ classic optimization;
- Real-time feedback-based optimization $=$ Reinforcement Learning (RL).

Obstacle avoidance! Best reference trajectory tracking! Minimum driving time (maximum average speed)!

RL as an Optimization Algorithm

mathematical model of a racing car

State : $\xi := (x, y, \psi, v)^T$, Euclidean coordinates (x, y), car orientation ψ, velocity v,

Action (control) : $u := (\delta, d)^T$ steering angle δ, duty cycle of the motor d,

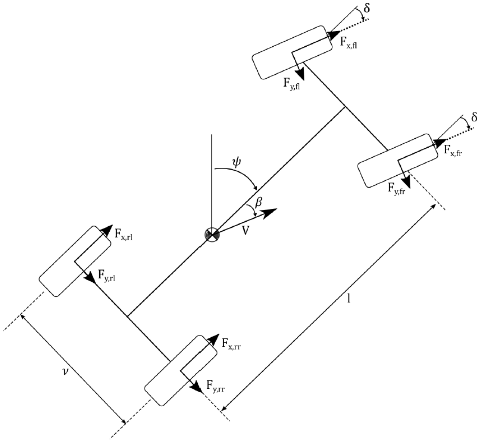
Environment (dynamic system) : $\dot{\xi}(t) = f(\xi(t), u(\cdot)),\ \xi(0) = \xi_0 \in \mathbb{R}^4,$ (1)

where

$$f(\xi, u) := \begin{pmatrix} v \cos \psi \\ v \sin \psi \\ \frac{1}{L} \delta v \\ (C_1 - vC_2)d - C_d v^2 - C_r - C_\delta \delta^2 v^2 \end{pmatrix}$$

RL as an Optimization Algorithm

a more sophisticated car model



RL as an Optimization Algorithm

a simplified mathematical model (kinematics) for data simulation

$$
\begin{aligned}
&\textbf{state}: \ \xi := (x,y)^T, \ \textbf{action}: \ u := (v,\psi)^T, \\
&\textbf{kinematic environment}: \\
&\dot{x}(t) = v\cos\psi, \ x(0) = x_0 \in \mathbb{R}, \\
&\dot{y}(t) = v\sin\psi, \ x(0) = y_0 \in \mathbb{R}.
\end{aligned}
\tag{2}
$$

the concept **Reward of Stage** (objective)

$$
R(\xi(t), u(\xi(t))) := v(t)\cos\psi(t) - w_1 v(t)\sin\psi(t) - w_2 g^2(x(t), y(t)),
$$

where $(\xi, u)^T$ is the state-action pairs, $w_j, \ j = 1,2$ are weights associated with the Reward terms and $g(x,y) = 0$ is the reference trajectory (set point) for a given smooth function $g(\cdot)$. The Reward should not only encourage high speed along the track, but also punish speed vertical to the track as well as deviation from the track.

RL as an Optimization Algorithm

reward optimization, Q-equation, learning, training, ...

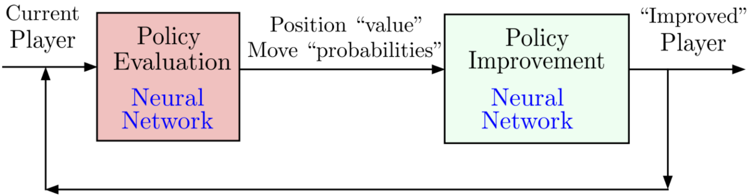
$$\text{minimize} \sum_{t \in [0,t_f]_G} R(\xi(t), u(\xi(t))) \tag{3}$$

$$\begin{aligned}&\mathbf{Q-equation}: \; Q(\xi(t), u(\xi(t))) = R(\xi(t), u(\xi(t))) + \\ &\alpha \max_{u_{t+1}(\xi)} Q(\xi(t+1), u(\xi(t+1))).\end{aligned} \tag{4}$$

solution $Q^{opt}(\cdot)$ of (4) $\Rightarrow$ policy iteration $u^{opt}(\xi(t+1))$.

Python for Optimization and ML

the ML solution to the Q-equation (4) The essence of Deep Q-Learning is the estimation of $Q^{opt}(\xi, u)$ using a type of neural network called a Deep Neural Network (also Q-Network) (DNN) parameterized by a specific vector θ. A "training" is used for the numerical definition of this parameter vector θ.



Python for Optimization and ML

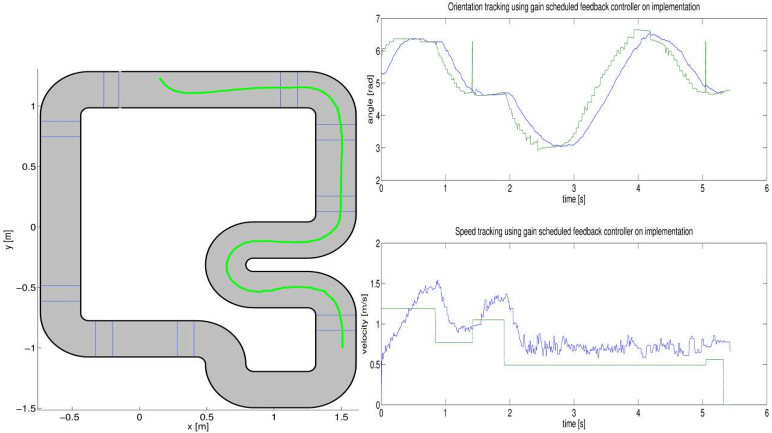
the Pythod RL frameworks

OpenAI Gym, Keras (rl.agents, KerasRL), PyTorch, Pyqlearning, TensorFlow (Tensorforse), RLCoach, TFAgents, RLlib and others.

the Python optimization packages

scipy.optimize (unconstrained and constrained optimization), scipy.optimize.minimize and others.

Some Computational Results



THANKS!