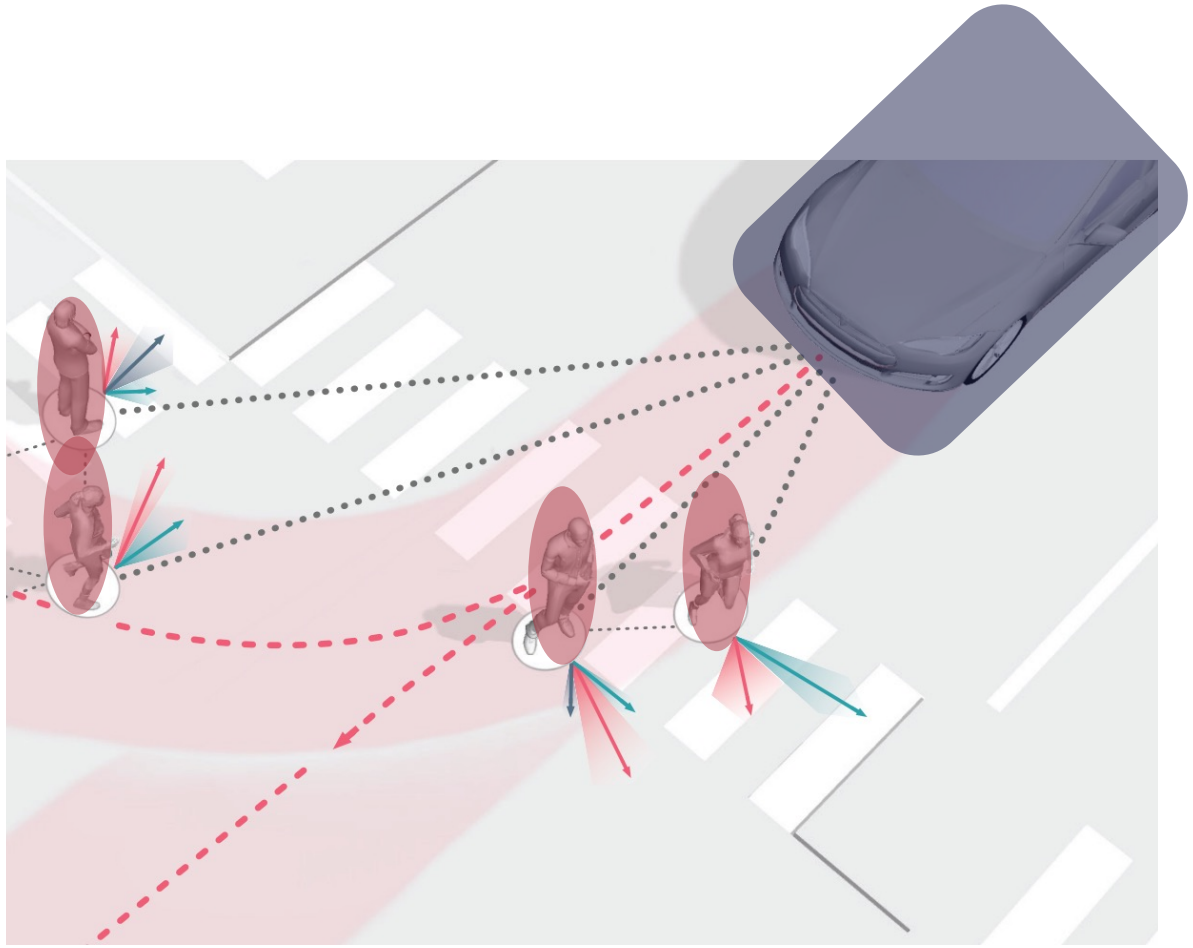
KI WISSEN
Automotive AI Powered by Knowledge

KI Wissen Final Event | 21-22 March 2024

Model Predictive Control under Temporal Logic Specifications

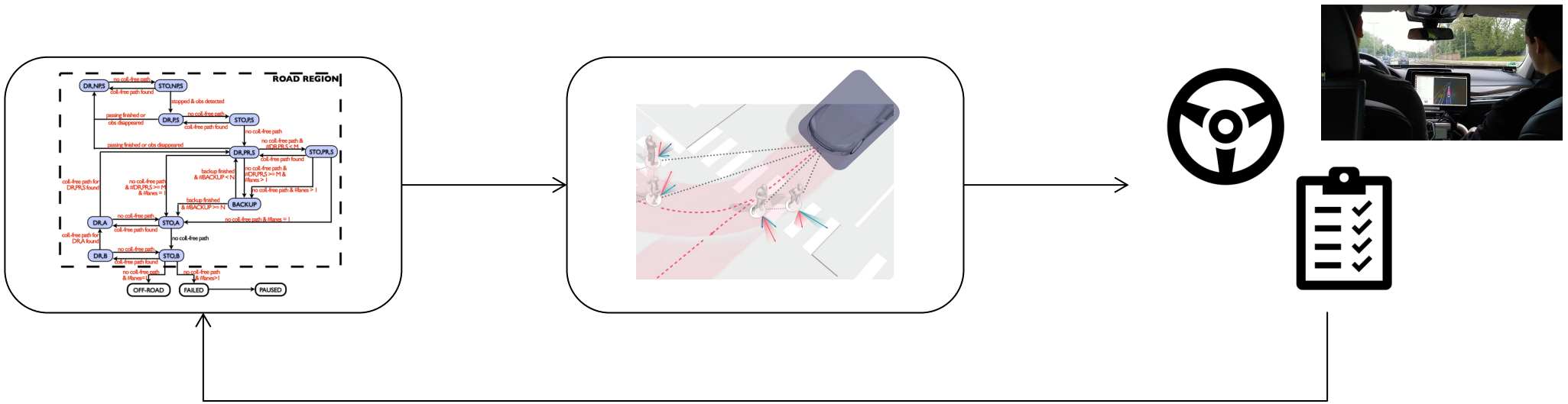
Etienne Bührle | FZI

Motivation



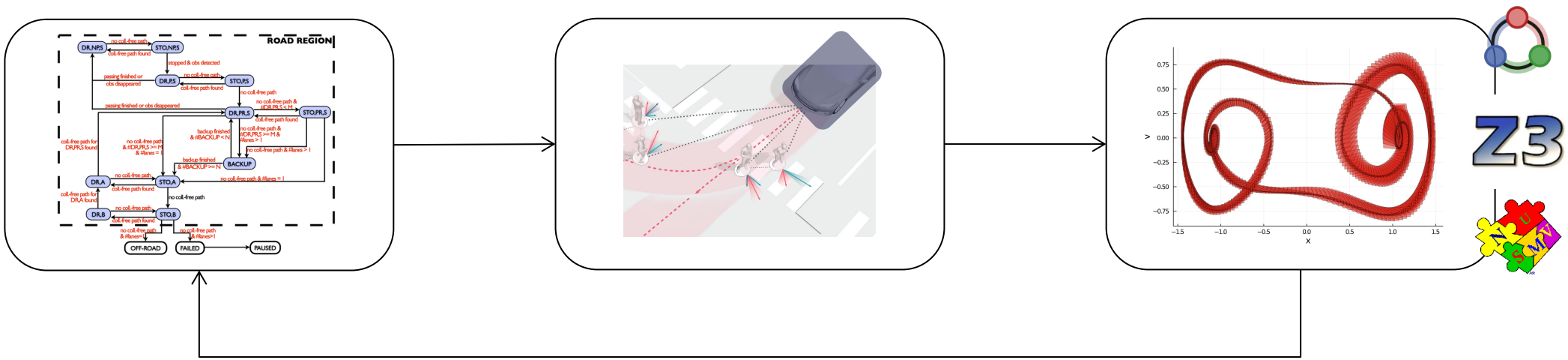
Salzmann et al.

Verification



Salzmann et al.; Wongpiromsarn; Prism; Z3; nuSMV

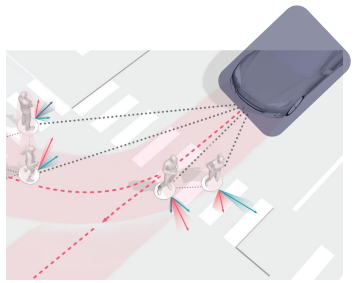
Verification

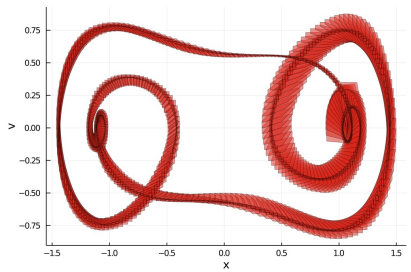


Salzmann et al.; Wongpiromsarn; JuliaReach; Prism; Z3; nuSMV

Verification

Can we synthesize controllers from specifications?

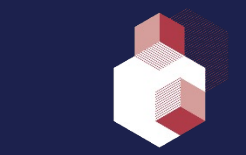




Salzmann et al.; Wongpiromsarn; JuliaReach; Prism; Z3; nuSMV

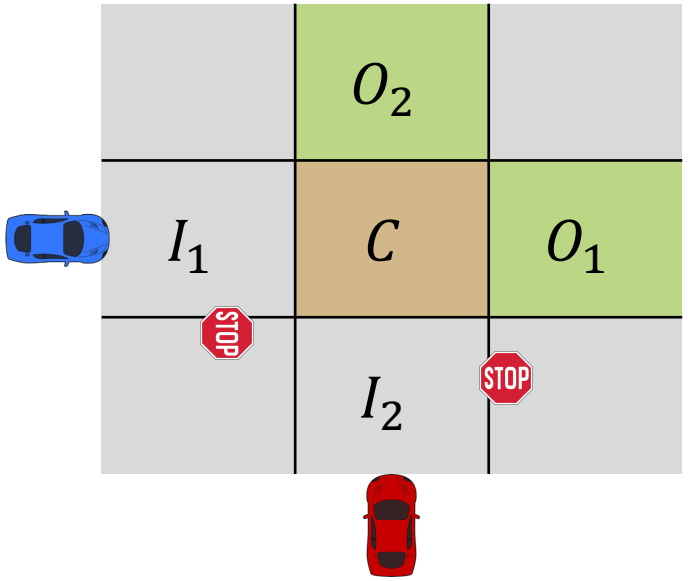
Agenda

1. Formal Synthesis for Linear Systems
2. Multiagent Control
3. Control Barrier Safety Layer



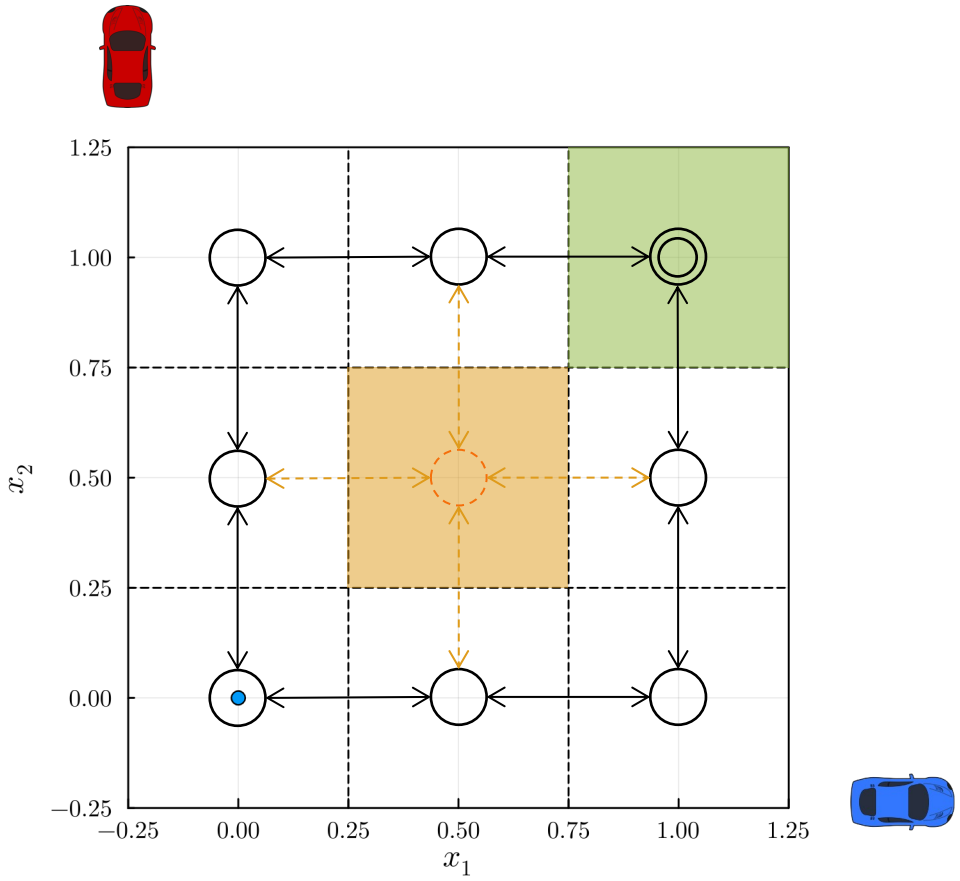
Formal Synthesis for Linear Systems

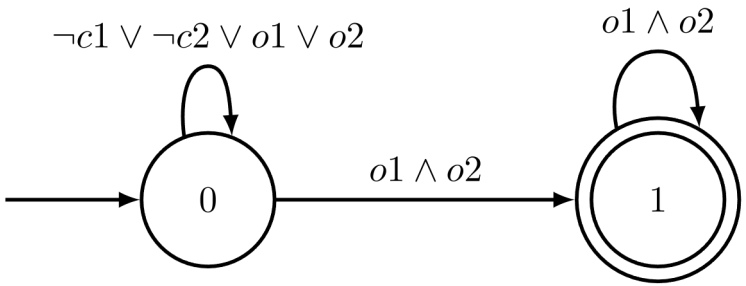
Formal Synthesis for Linear Systems



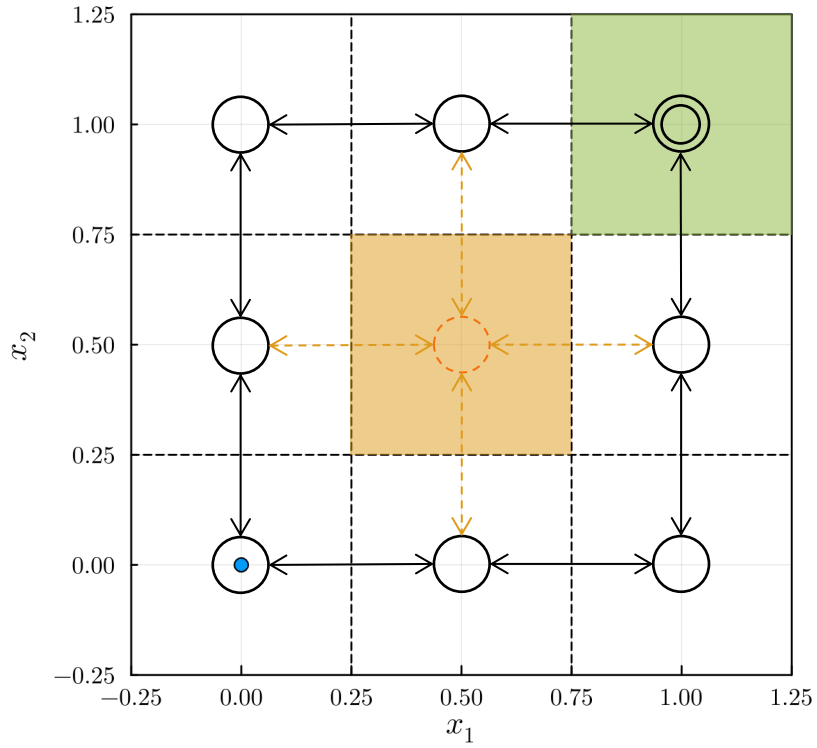
$$G\neg(x_1 \in C \wedge x_2 \in C)$$
$$\wedge F(x_1 \in O_1)$$
$$\wedge F(x_2 \in O_2)$$

Formal Synthesis for Linear Systems





Quasi-Infinite Horizon OCP



$$\min_{x,u,q} \int_0^T c(x,u)dt + V(x(T))$$

$$s.t. \quad x(T) \in \mathcal{T} \qquad D(q) \leq 0$$

$$C(x,u) = 0 \qquad S(q,x) \leq 0$$

Value Function

Moment LP

$$\min_{\mu,\mu_T} \langle l, \mu \rangle$$

$$\text{s.t.} \quad div(f\mu) + \mu_T = \mu_0$$

$$\mu_0, \mu_T, \mu \in M(X \times U)$$

Dual Moment LP

$$\max_{v} \langle v, \mu_0 \rangle$$

$$\text{s.t.} \quad l + \nabla v \cdot f \in C(X \times U)$$

$$l_T - v \in C(X_T)$$

Positivity constraints

Value Function

Moment LP

$$\min_{\mu,\mu_T} \langle l, \mu \rangle$$

$$\text{s.t.} \quad div(f\mu) + \mu_T = \mu_0$$

$$\mu_0, \mu_T, \mu \in M(X \times U)$$

Dual LP

$$\max_{v} \langle v, \mu_0 \rangle$$

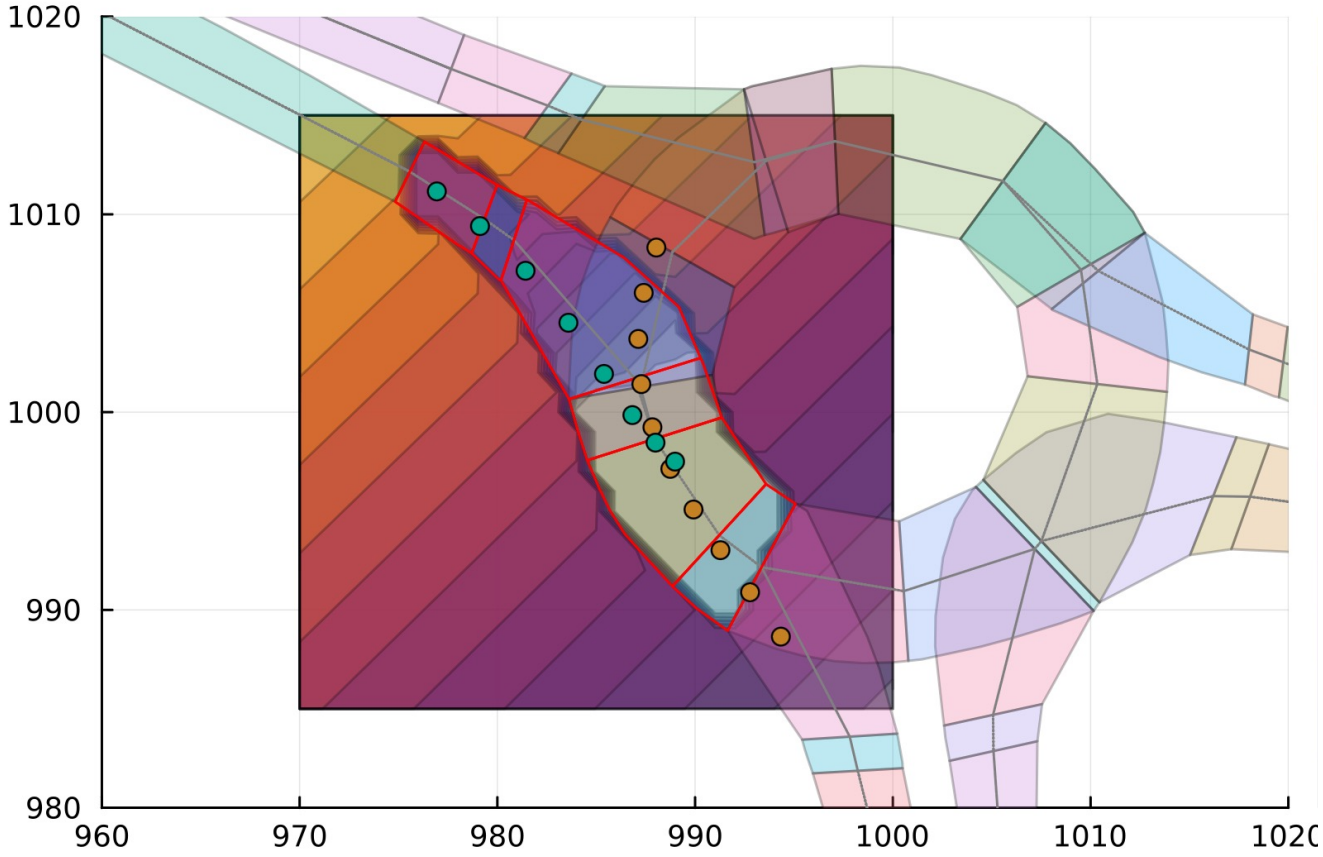
Approximate via sampling

$$\text{s.t.} \quad l + \nabla v \cdot f \geq 0 \quad \forall x, u \in S(X \times U)$$

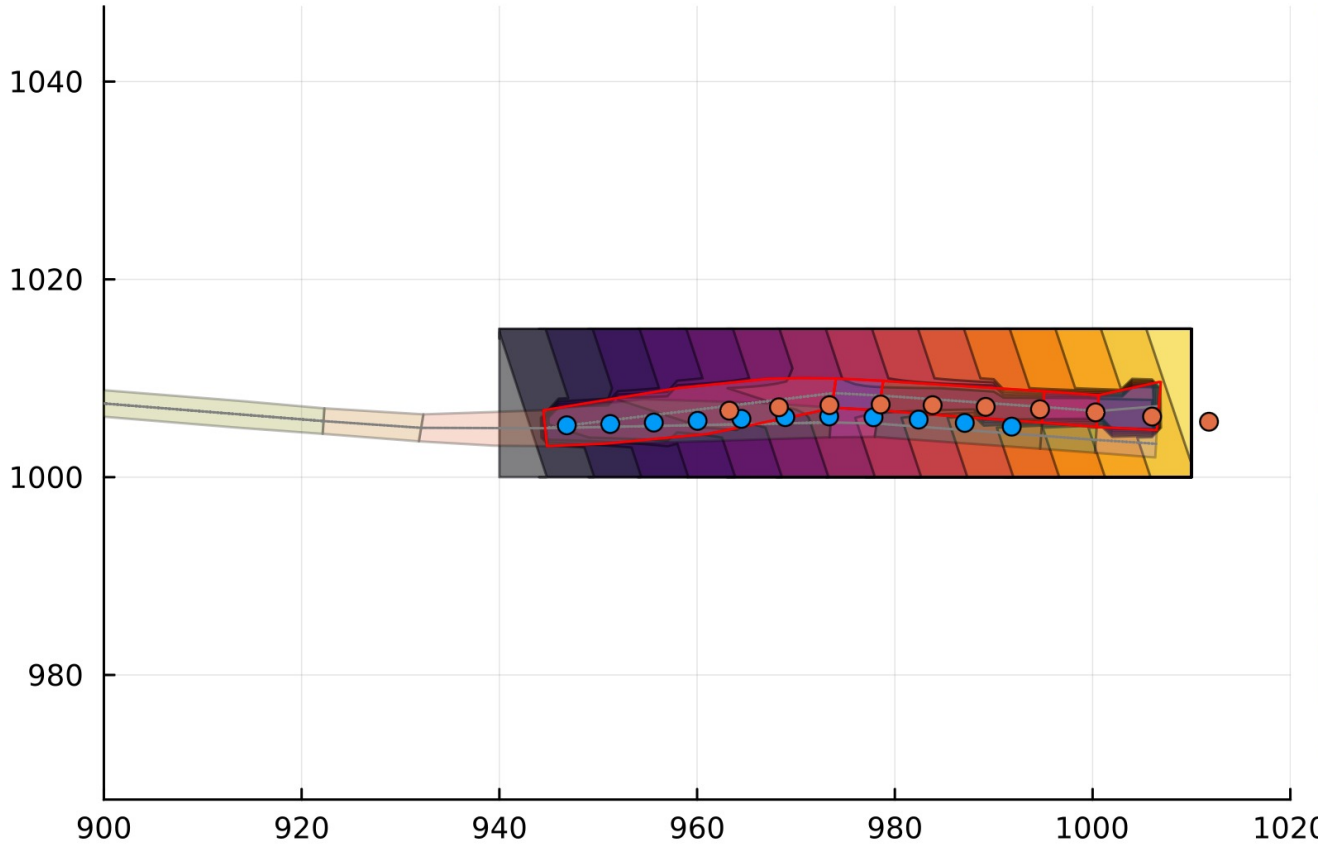
$$l_T - v \geq 0 \quad \forall x \in C(X_T)$$

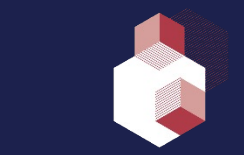
=> Linear Program

Results



Results





Multiagent Control

Method

Optimize individually

$$\min_{x,y,q} J_x(x,y) + \lambda_1(y - y^0) \qquad \min_{y,x,q} J_y(y,x) + \lambda_2(x - x^0)$$

$$s.t.\ S_x(x,y) = 0 \qquad s.t.\ S_y(y,x) = 0$$

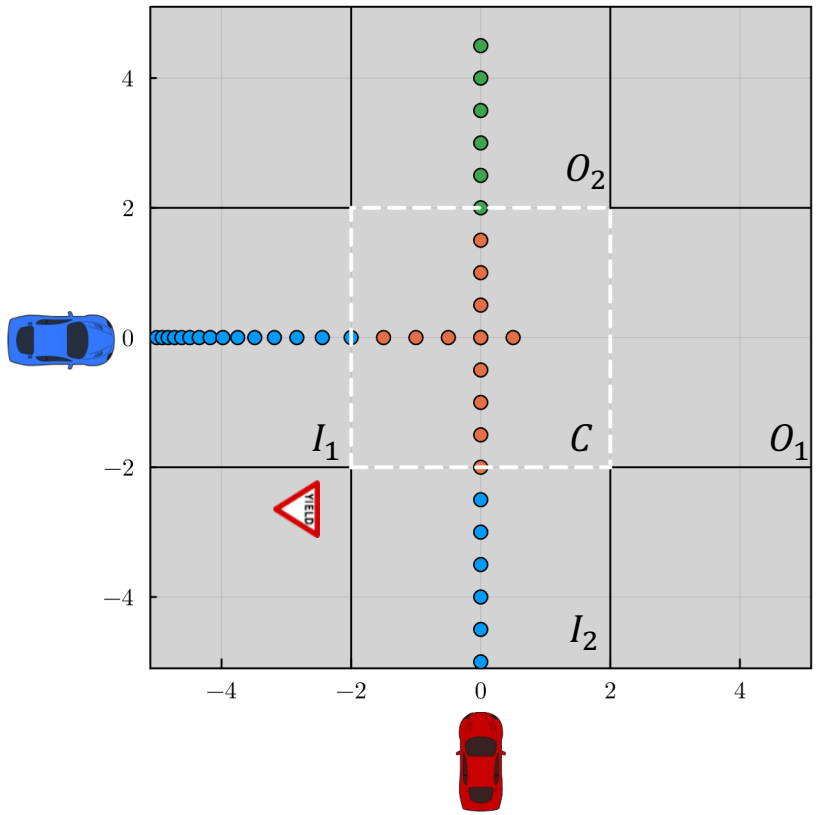
Penalize deviation of shared variables

$$\lambda_1' = \lambda_1 + \alpha(y - y^0) \qquad \lambda_2' = \lambda_2 + \alpha(x - x^0)$$

Update shared variables, iterate

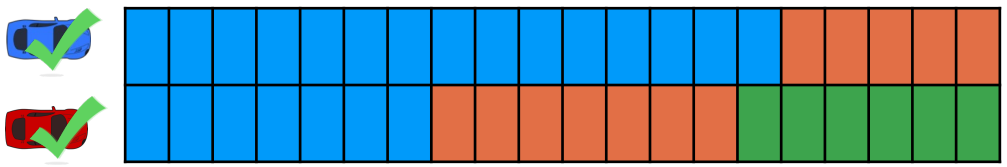
Update y^0 Update x^0

Example

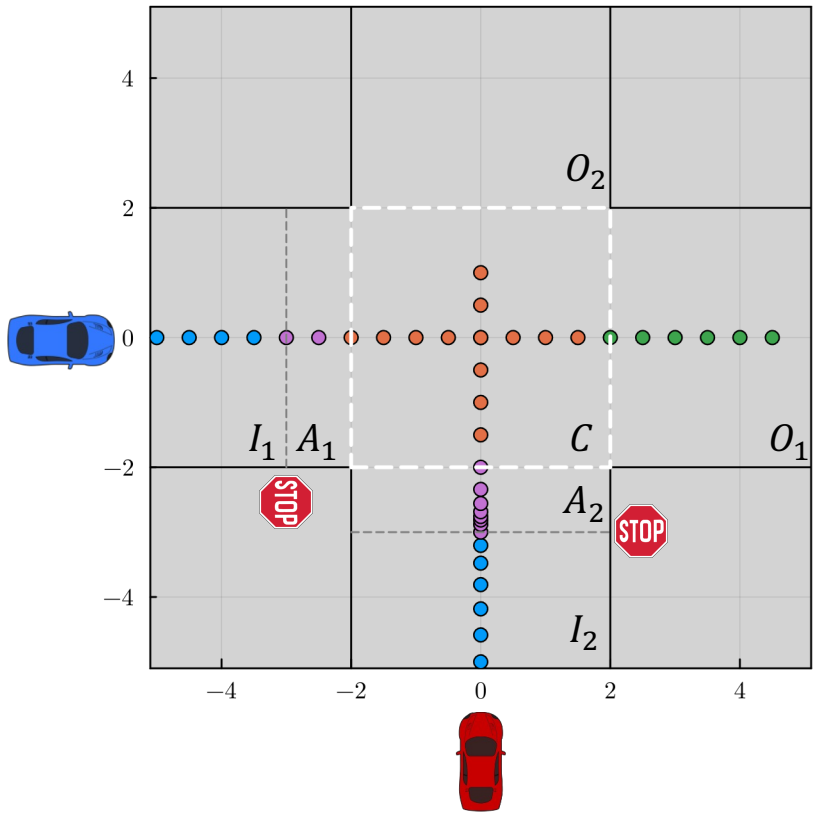


$G\neg(x_1 \in C \wedge x_2 \in C)$

$true$

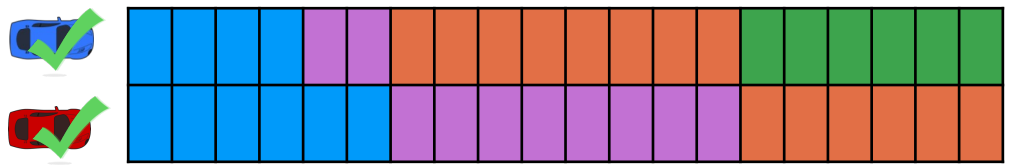


Example



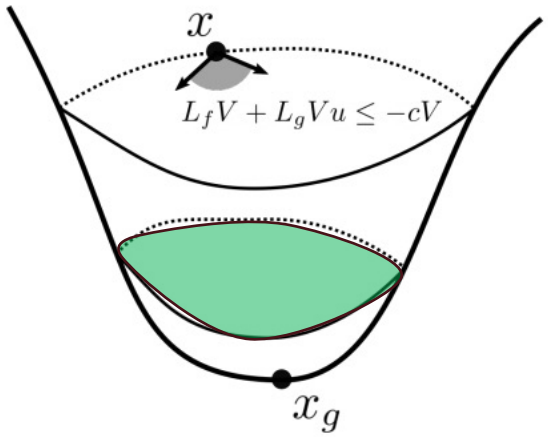
$G((x_1 \in I_1 \wedge x_2 \in A_2) \Rightarrow G\neg(x_1 \in C \wedge x_2 \in C))$

$G((x_2 \in I_2 \wedge x_1 \in A_1) \Rightarrow G\neg(x_2 \in C \wedge x_1 \in C))$



Control Barrier Safety Layer

Control Barrier Functions



- Given system dynamics $\dot{x} = f(x, u)$ and barrier function

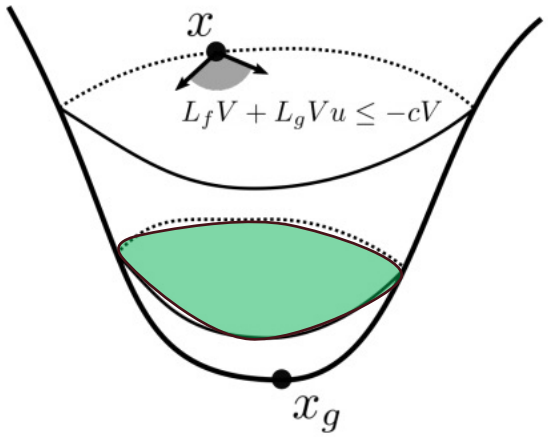
$$h(x) \leq 0, \qquad x \in X_{safe}$$
$$h(x) > 0, \qquad x \notin X_{safe}$$

- We want the system trajectories to stay inside the safe set

$$\dot{h} = \nabla_x h \cdot f(x, u) \leq -c \cdot h(x)$$

Gao et al.

Control Barrier Functions



- Given system dynamics $\dot{x} = f(x, u)$ and barrier function

$$h(x) \leq 0, \qquad x \in X_{safe}$$
$$h(x) > 0, \qquad x \notin X_{safe}$$

- For input-affine systems, these constraints are affine in u!

$$\nabla_x h \cdot (f(x) + g(x) \cdot u) \leq -c \cdot h(x)$$

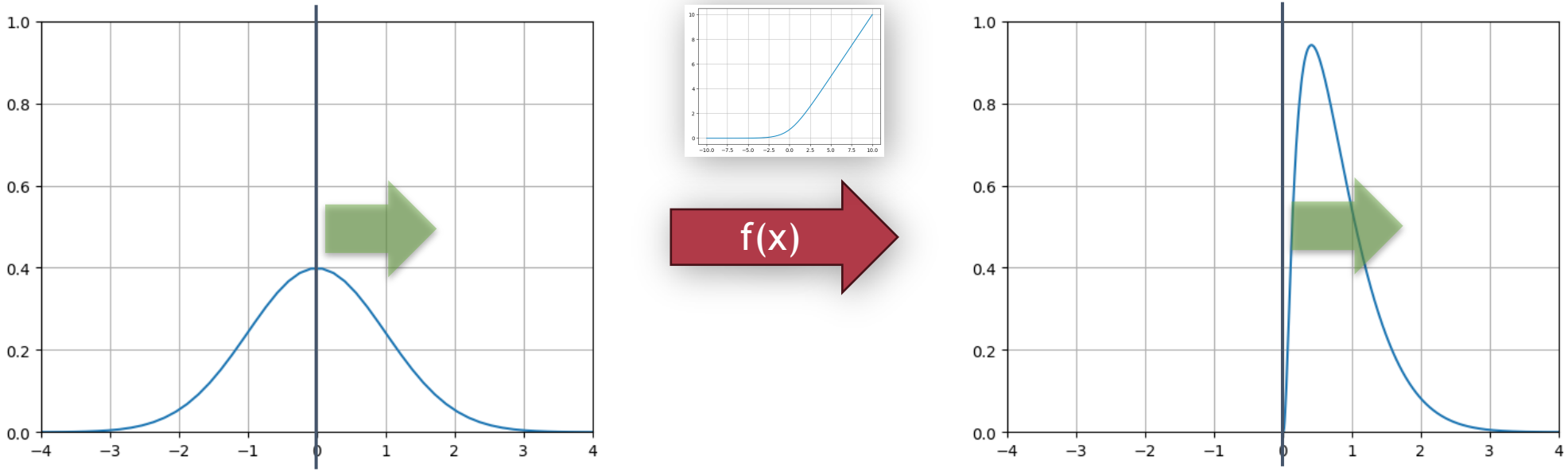
Gao et al.

Projection Layer

- Project density into safe set using change of variables

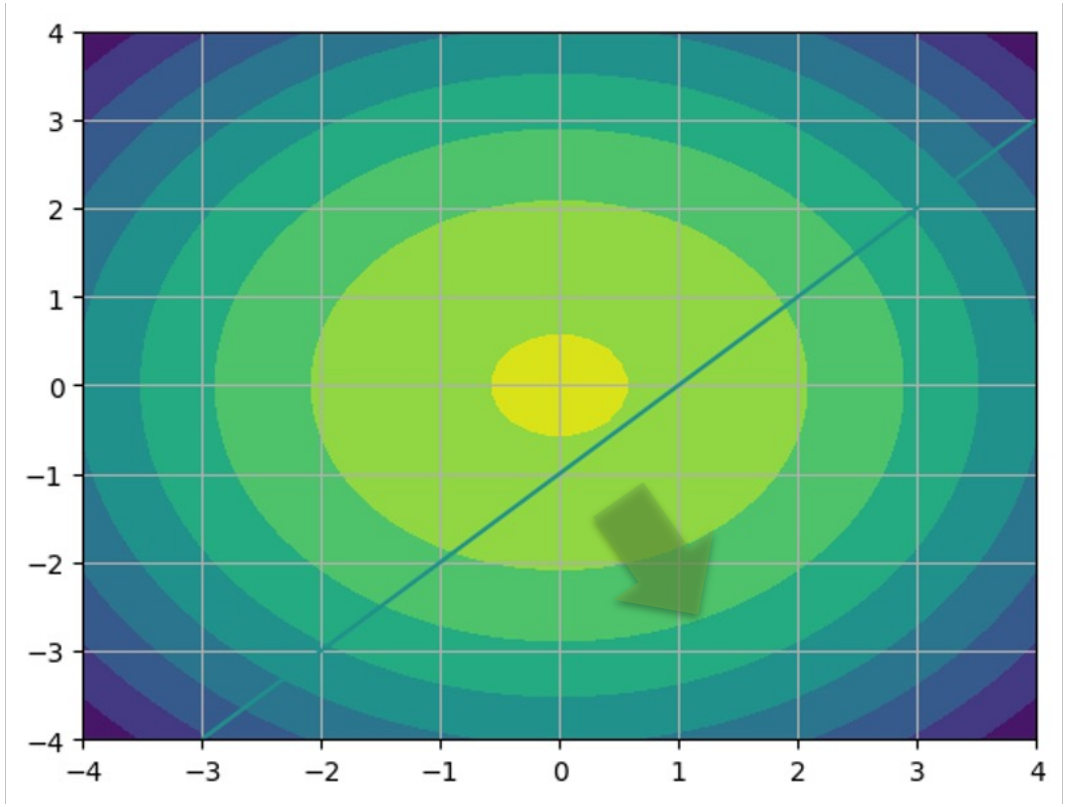
$$p_x(x)$$

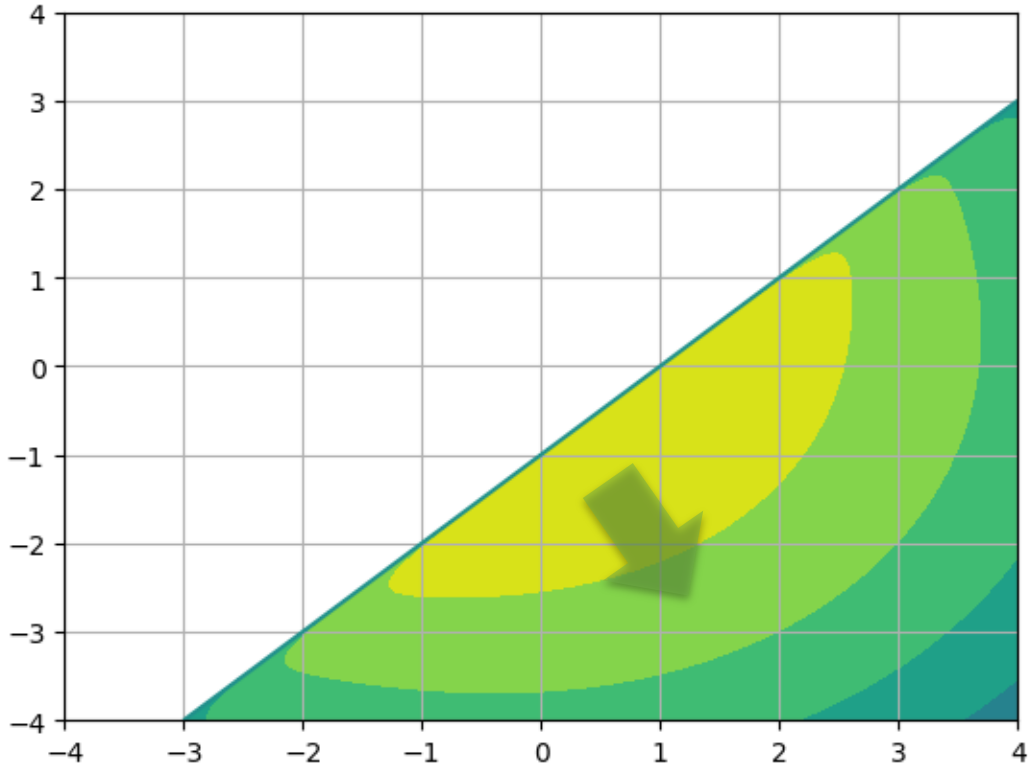
$$p_z(z) = p_x\left(f^{-1}(z)\right)\left|\det\frac{\partial f}{\partial x}\right|$$



Projection Layer

- Project density into safe set using change of variables





Vanilla Policy Gradient

Algorithm 1 Vanilla Policy Gradient Algorithm

1: Input: initial policy parameters θ_0, initial value function parameters ϕ_0
2: **for** $k = 0, 1, 2, \ldots$ **do**
3: Collect set of trajectories $\mathcal{D}_k = \{\tau_i\}$ by running policy $\pi_k = \pi(\theta_k)$ in the environment.
4: Compute rewards-to-go $\hat{R}_t$.
5: Compute advantage estimates, $\hat{A}_t$ (using any method of advantage estimation) based on the current value function V_{ϕ_k}.
6: Estimate policy gradient as

$$\hat{g}_k = \frac{1}{|\mathcal{D}_k|} \sum_{\tau \in \mathcal{D}_k} \sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t)|_{\theta_k} \hat{A}_t.$$

7: Compute policy update, either using standard gradient ascent,

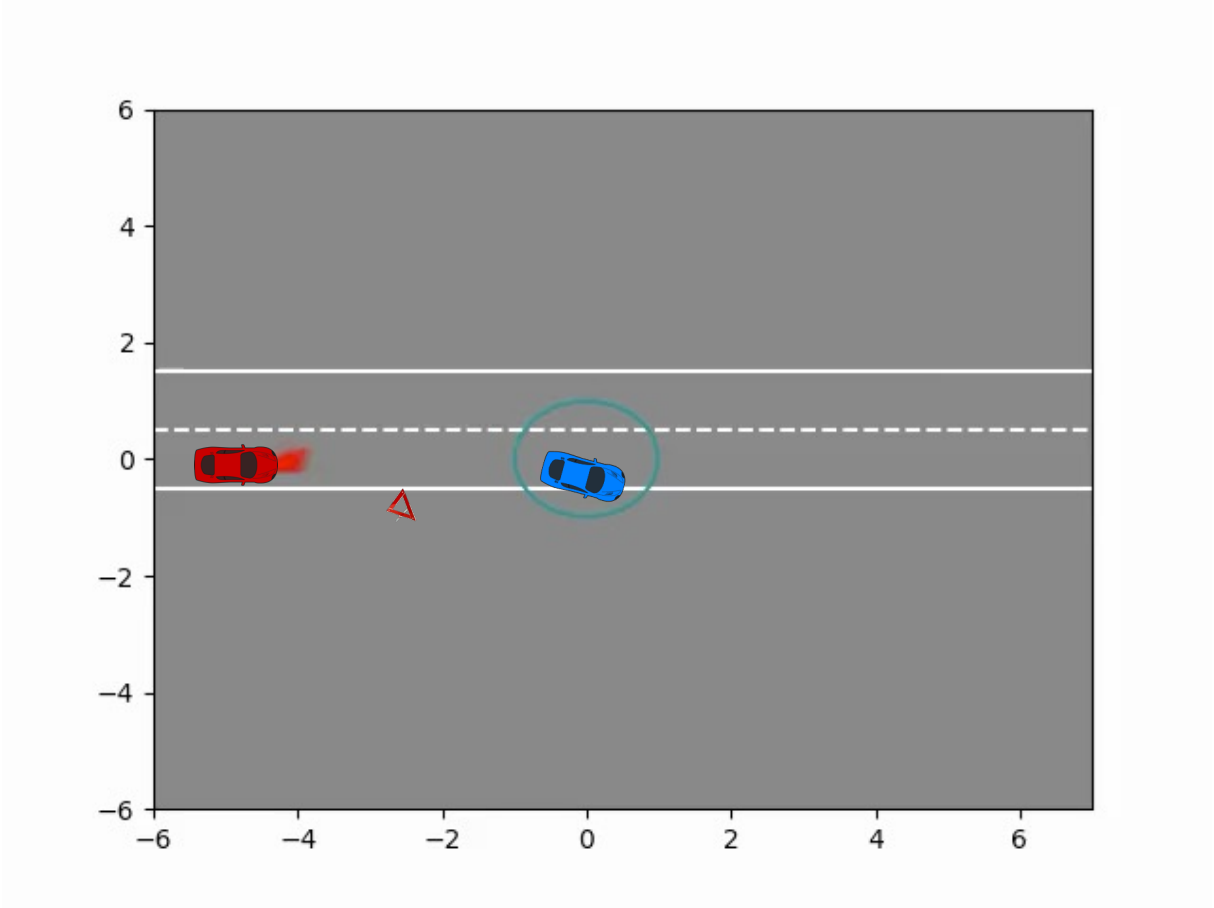
$$\theta_{k+1} = \theta_k + \alpha_k \hat{g}_k,$$

or via another gradient ascent algorithm like Adam.
8: Fit value function by regression on mean-squared error:

$$\phi_{k+1} = \arg\min_{\phi} \frac{1}{|\mathcal{D}_k| T} \sum_{\tau \in \mathcal{D}_k} \sum_{t=0}^{T} \left(V_\phi(s_t) - \hat{R}_t \right)^2,$$

typically via some gradient descent algorithm.
9: **end for**

Example



KI WISSEN

Automotive AI Powered by Knowledge

Etienne Bührle| FZI | buehrle@fzi.de

KI Wissen is a project of the KI Familie. It was initiated and developed by the VDA Leitinitiative autonomous and connected driving and is funded by the Federal Ministry for Economic Affairs and Climate Action.

KI FAMILIE

Supported by:

Federal Ministry for Economic Affairs and Climate Action

on the basis of a decision by the German Bundestag

kiwissen.de

@KI_Familie

KI Familie