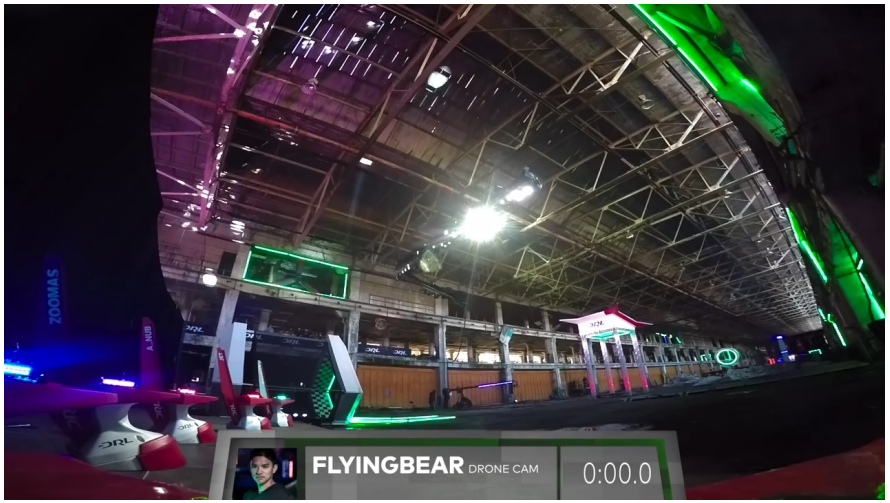
Augmented Lagrangian Methods for Online Game Theoretic Planning with Constraints

ICRA 2021 Tutorial on Game Theoretical Motion Planning

Mac Schwager
Aeronautics and Astronautics
Stanford University



FPV Human Drone Racing



GTP Autonomous Drone Racing

Stanford University

Generalized Nash Style Game

Each Robot has its own objective:

Robot i's strategy

All other robot's strategies

$$\min_{\theta_i \in \Theta} J_i(\theta_i, \theta_{-i})$$

$$\text{s.t.} \quad g(\theta_i, \theta_{-i}) \leq 0$$

Coupled constraints
Generalized Nash Game

Generalized Nash Equilibrium:

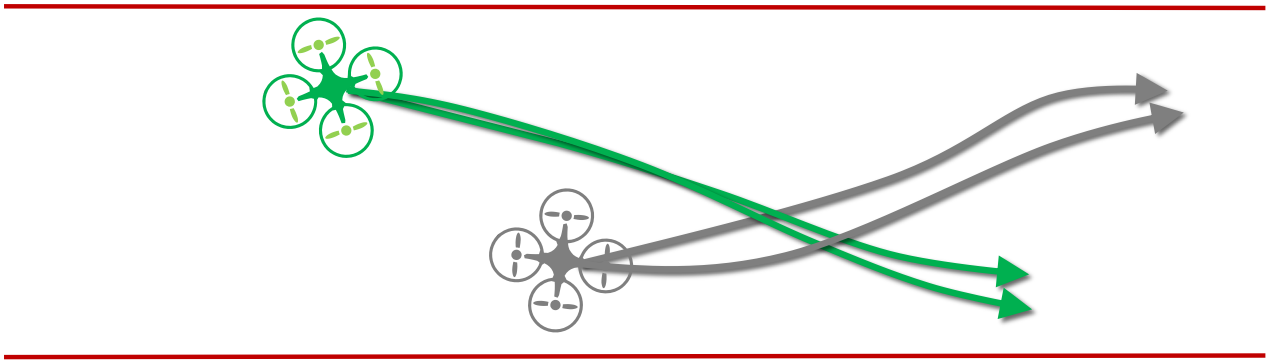
For all robots:

$$\theta_i^* = \arg\min_{\theta_i \in \Theta} J_i(\theta_i, \theta_{-i}^*)$$

$$\text{s.t.} \quad g(\theta_i, \theta_{-i}^*) \leq 0$$

No robot can unilaterally improve its objective.

Iterative Best Response (IBR)



- Nash Equilibrium is a fixed point of the best response map.
- If IBR converges, it converges to a Nash equilibrium!
- In practice, does not work with constraints (Generalized Nash Games)

ALGAMES: A Fast Solver for General Sum Games

For each Robot:

$$\min_{\theta_i \in \Theta} J_i(\theta_i, \theta_{-i})$$

$$\text{s.t.} \quad g(\theta_i, \theta_{-i}) \leq 0$$

$$\mathcal{L}_i(\theta_i, \theta_{-i})$$ Augmented Lagrangian



Simon LeCleac'h



Zac Manchester

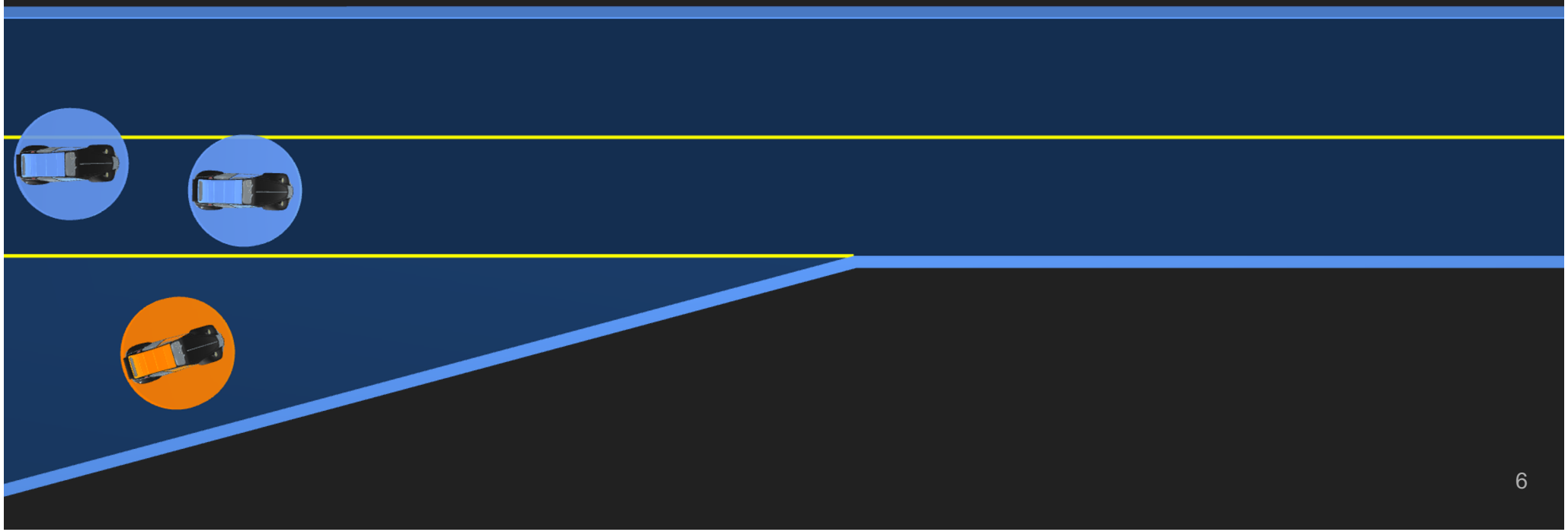
Directly solve first order optimality conditions with Newton's method

$$\nabla_{\theta_i} \mathcal{L}_i(\theta_i, \theta_{-i}) = 0 \quad \forall i$$

LeCleac'h et Al., RSS 2020
LeCleac'h et Al., ArXiv (long form)

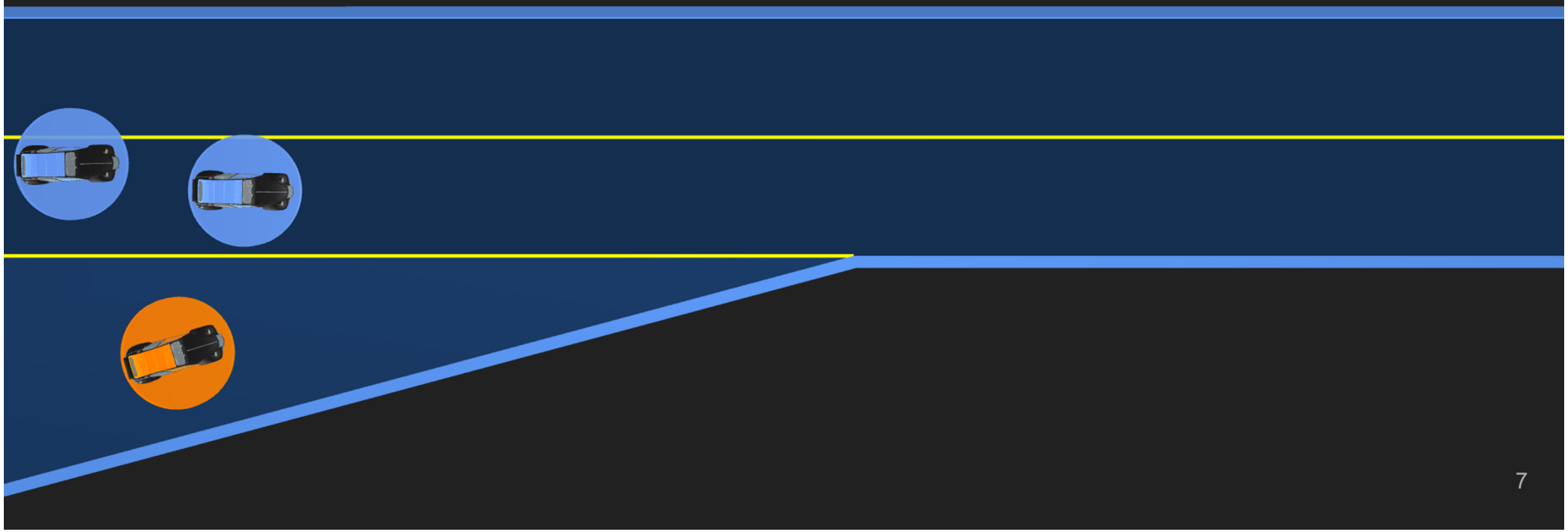
Solves in 14 ms for two players

ALGAMES algorithm



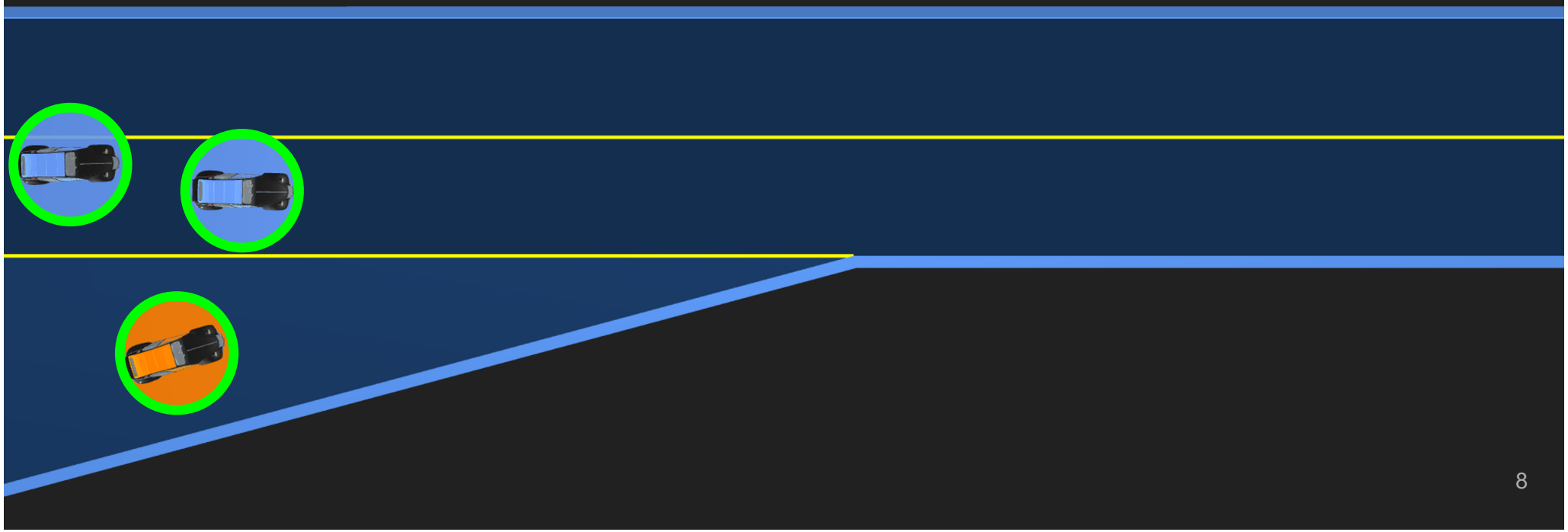
ALGAMES algorithm

Vehicle collision avoidance



ALGAMES algorithm

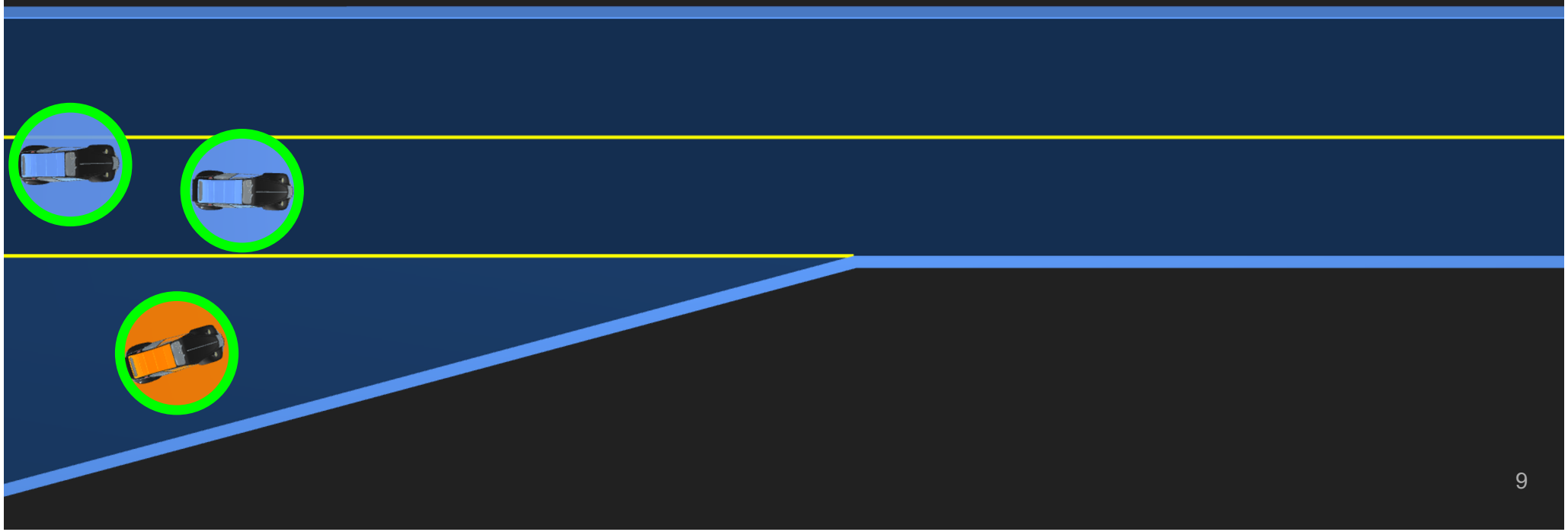
Vehicle collision avoidance



ALGAMES algorithm

Vehicle collision avoidance

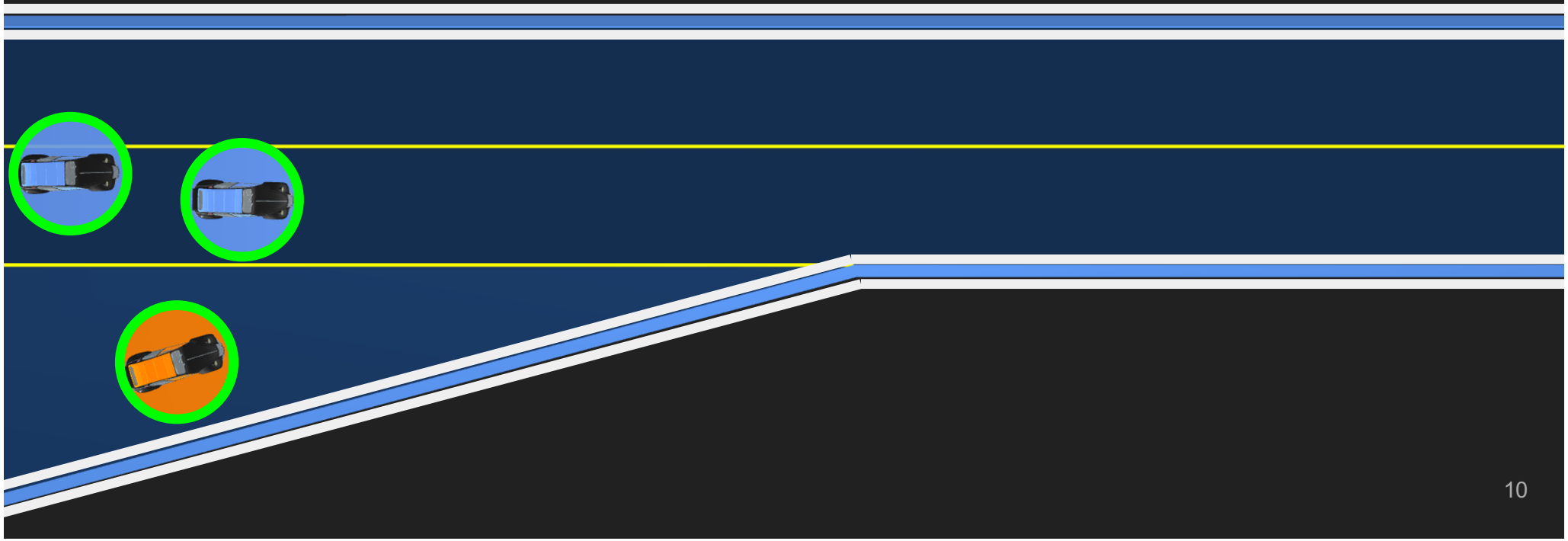
Boundary collision avoidance



ALGAMES algorithm

Vehicle collision avoidance

Boundary collision avoidance



Dynamics

$$x_{k+1} = f(\overbrace{x_k}^{\text{Joint state}}, \underbrace{u_k^{(1)}}_{\text{Player 1}}, \underbrace{u_k^{(2)}}_{\text{Player 2}})$$

Write as: $D(X, U) = 0$

Objective for One Player

$$\min_{X, U^{(1)}} \; J^1(X, U^{(1)})$$

$$\text{s.t.} \quad D(X, U) = 0 \qquad \text{Dynamics}$$

$$C(X, U) \leq 0 \qquad \text{Constraints}$$

Coupled Objectives

$$\min_{X,U^{(1)}} \; J^1(X,U^{(1)}) \qquad \min_{X,U^{(2)}} \; J^2(X,U^{(2)})$$

$$\text{s.t.} \quad D(X,U) = 0 \qquad \text{s.t.} \quad D(X,U) = 0$$

$$C(X,U) \leq 0 \qquad C(X,U) \leq 0$$

Augmented Lagrangian

Lagrange multiplier for dynamics

$$L^{\nu}(X,U) = J^{\nu} + \mu^{\nu T} D + \lambda^{T} C + \frac{1}{2} C^{T} I_{\rho} C$$

Lagrange multiplier for constraints

Quadratic penalty for constraints

where

$$I_{\rho,kk} = \begin{cases} 0 & \text{if } C_k(X,U) < 0 \wedge \lambda_k = 0 \\ \rho_k & \text{otherwise,} \end{cases}$$

- Rigorously enforces constraints (from Lagrange multiplier)
- Better numerical conditioning (from quadratic penalty)

Newton's Method

Gradient for player v: $\nabla_{X,U^\nu} L^\nu(X,U,\mu^\nu) = G^\nu(X,U,\mu^\nu)$

Stacked gradient and dynamics: $G(X,U,\mu) = [(G^1)^T, \ldots, (G^M)^T, D^T]^T$

Stacked Hessian: $H = \nabla_{X,U,\mu} G = \nabla_y G$

Search Direction: $\boxed{\delta y = -H^{-1} G}$

Iterations: $y \leftarrow y + \alpha \delta y$

With backtracking line search for α

Newton's Method

In practice, we find search direction with a backward substitution through time, similar to a Riccati recursion.

$$\delta y = -H^{-1} G$$

Converges to dynamically feasible trajectory, but with constraint violation!

ALGAMES

Until Convergence **do**

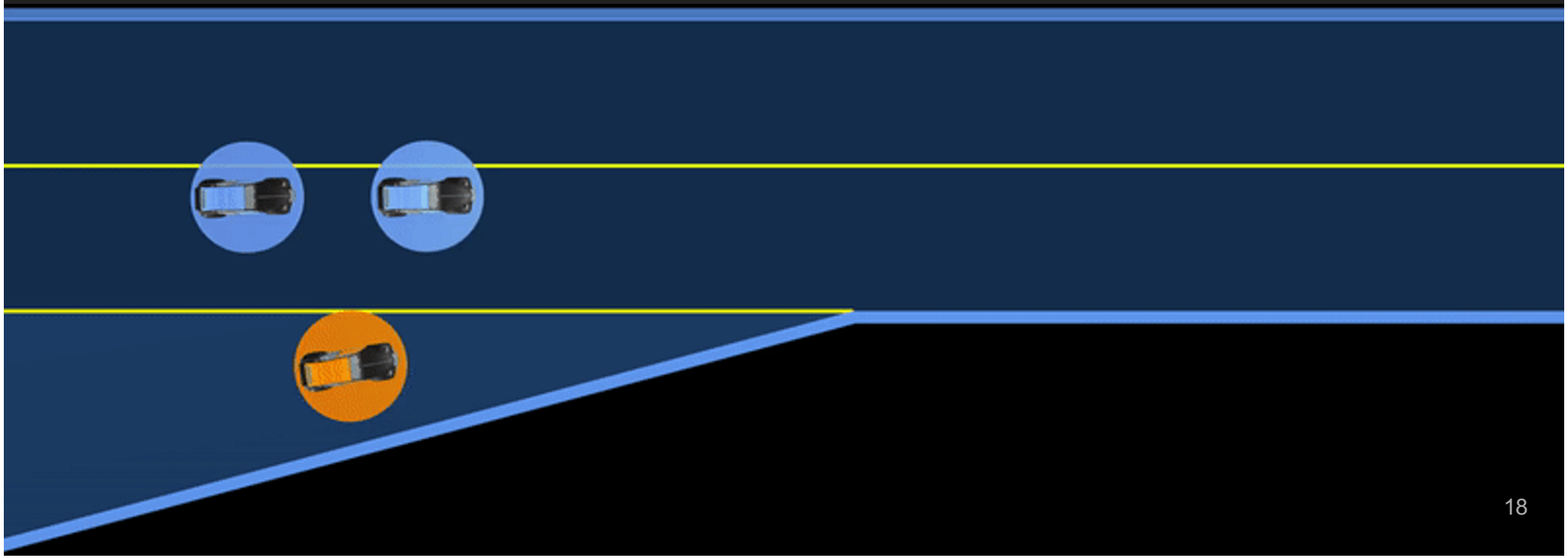
$y \leftarrow \text{NEWTON'SMETHOD}(y)$

$\lambda \leftarrow \text{DUALASCENT}(y, \lambda, \rho)$,

$\rho \leftarrow \text{INCREASINGSCHEDULE}(\rho)$,

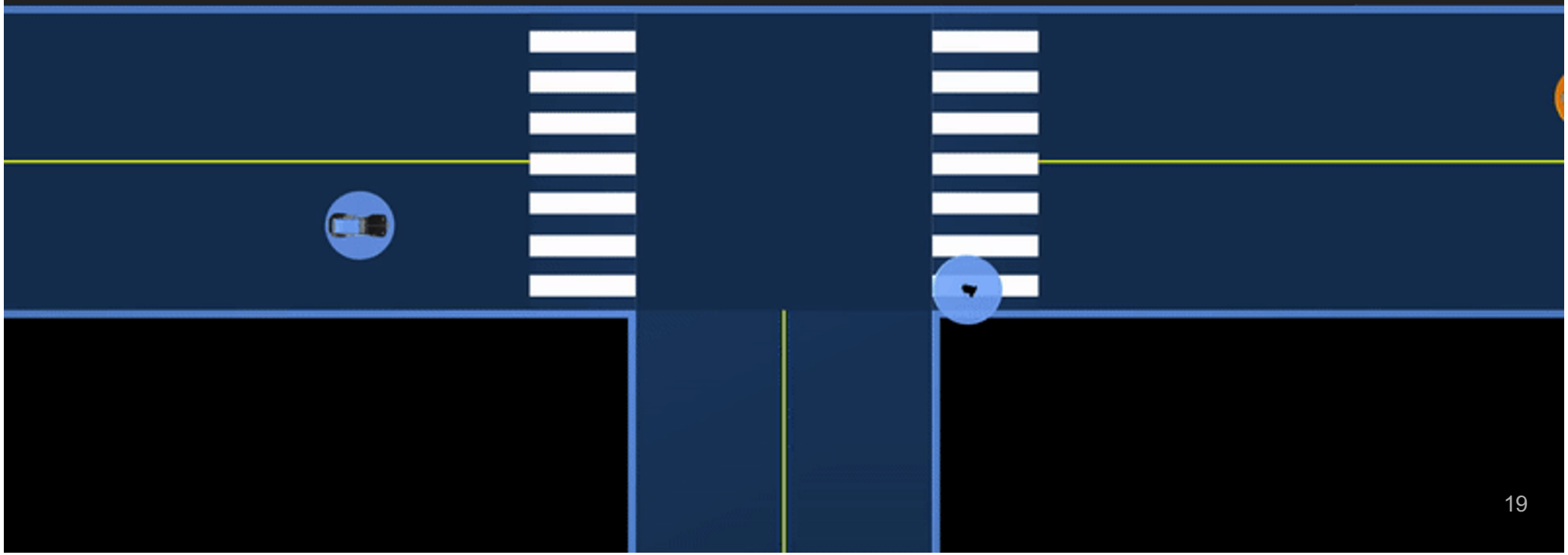
Applications

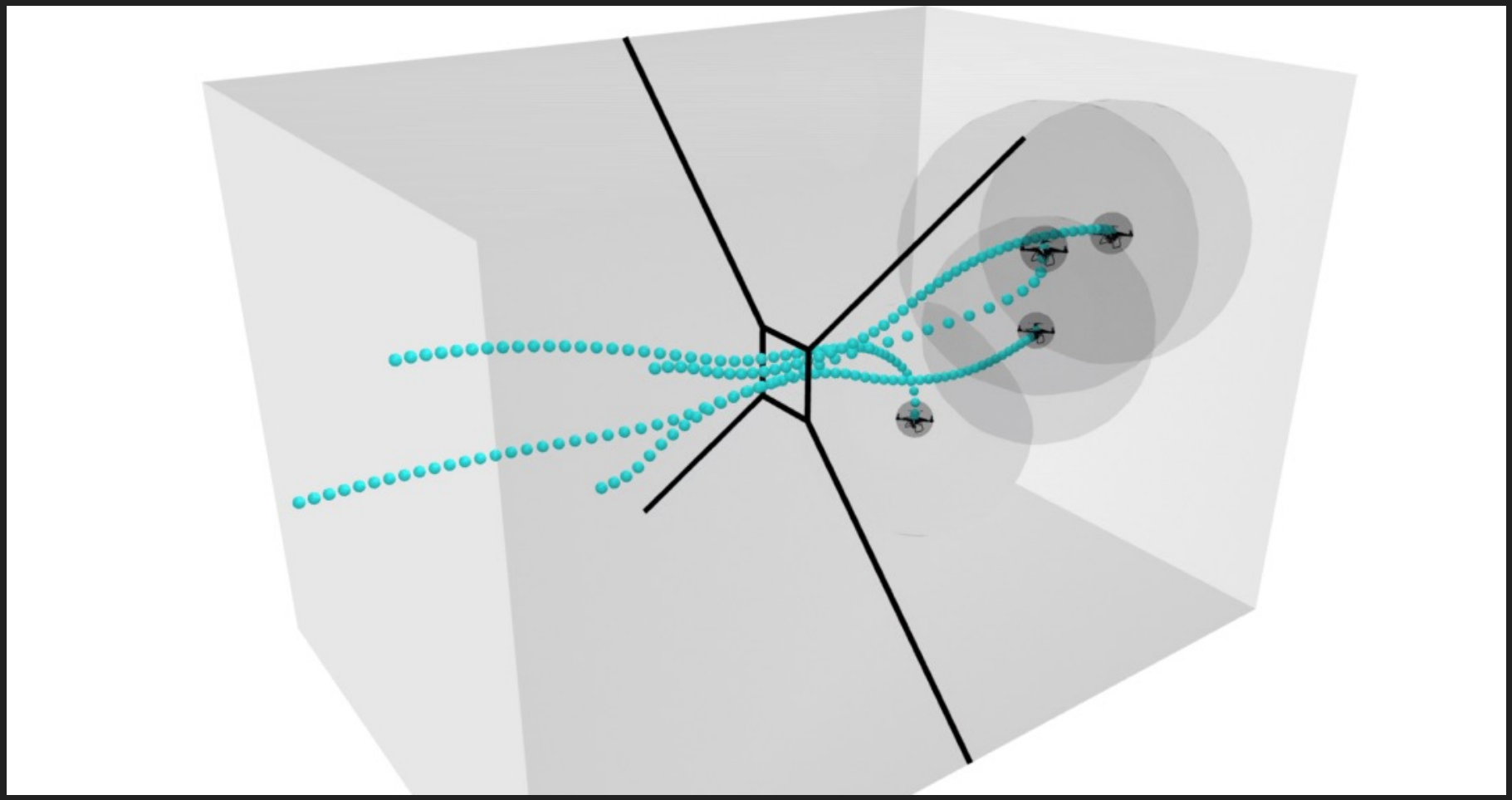
Ramp merging



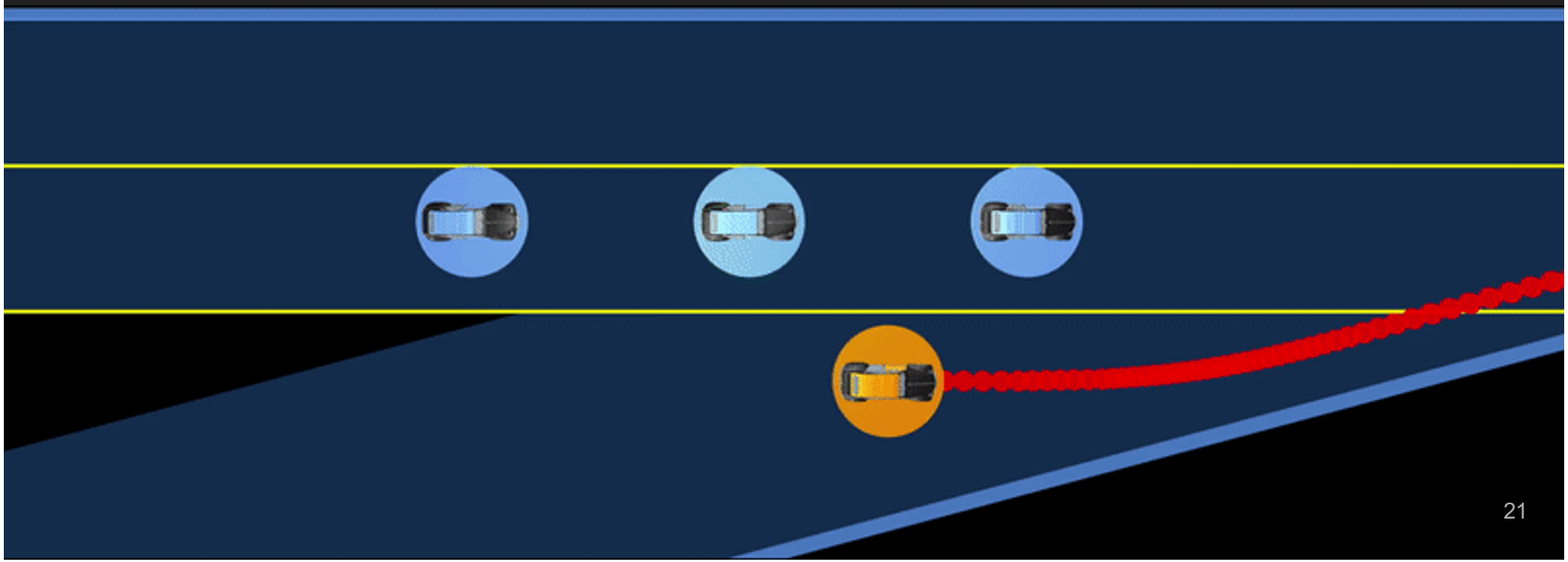
Applications

Intersection crossing

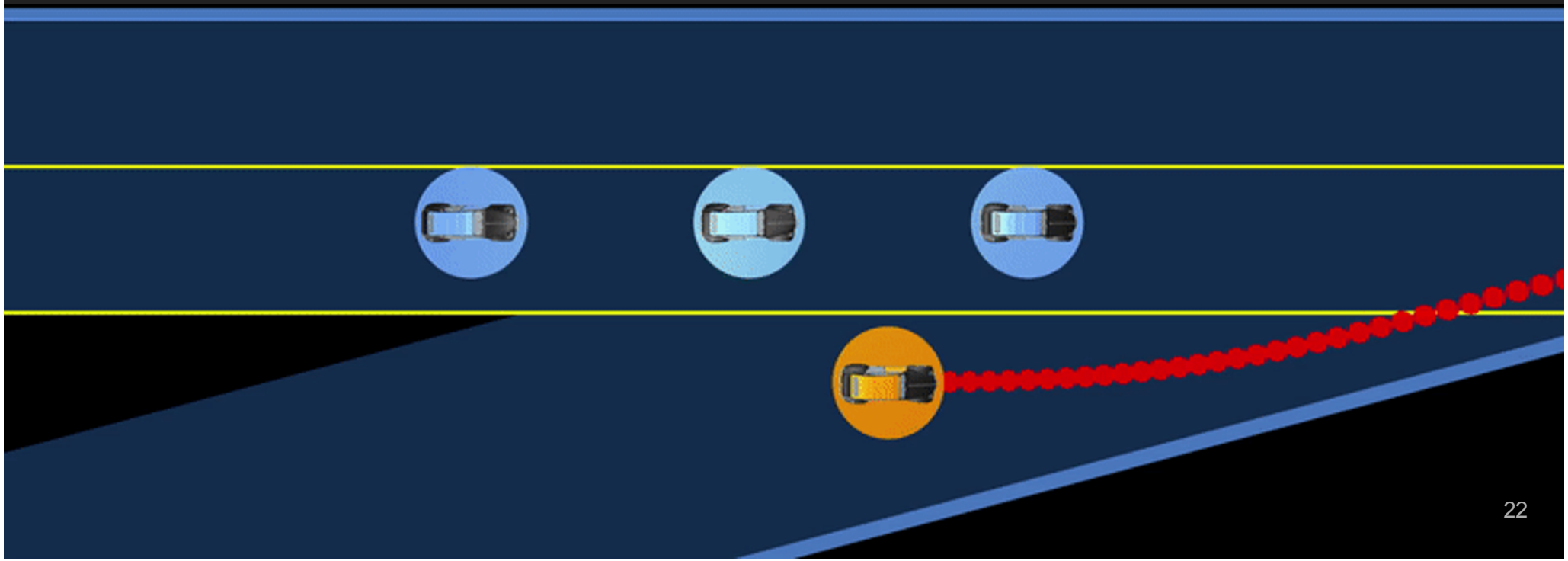




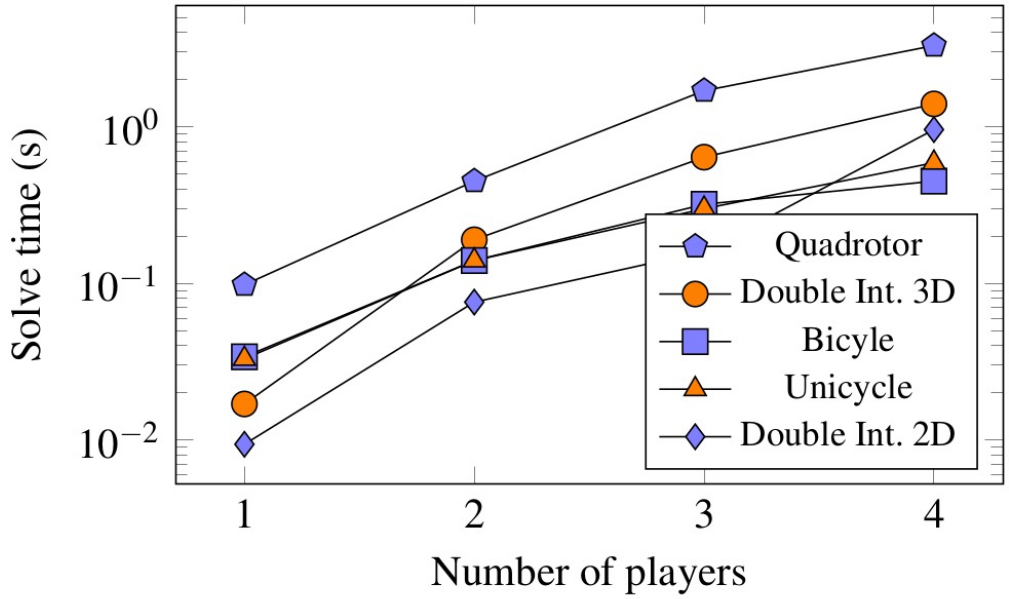
Non game theoretic MPC



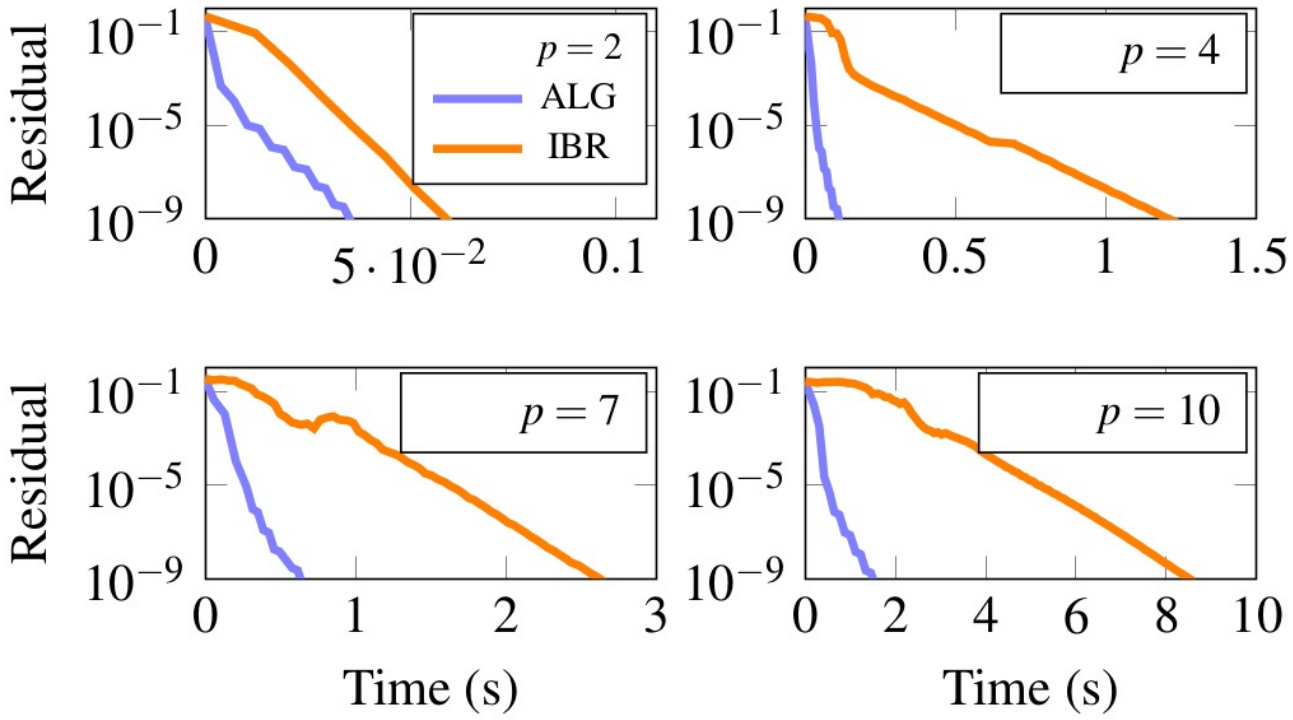
ALGAMES



Model	State dim.	Control dim.
Quadrotor	12	4
Double Integrator 3D	6	3
Bicycle	4	2
Unicycle	4	2
Double Integrator 2D	4	2

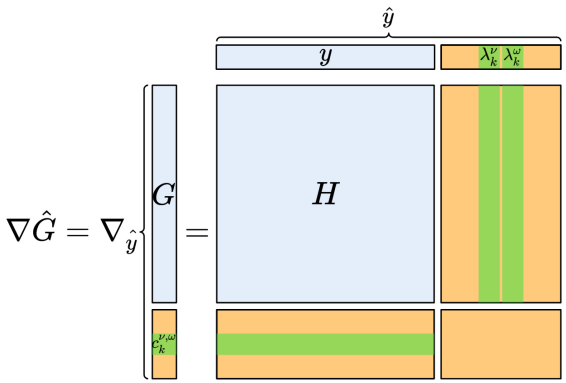


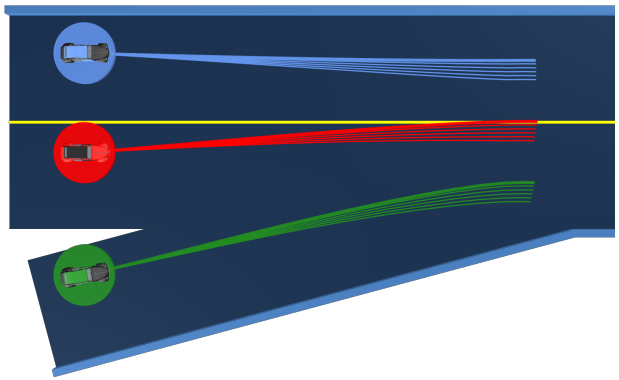
Scenario	# Players	ALGAMES	iLQGames
Ramp Merging	2	**38 ± 10ms**	$104 \pm 23ms$
	3	**89 ± 14ms**	$197 \pm 15ms$
	4	$860 \pm 251ms$	**705 ± 209ms**
Intersection	2	**50 ± 11ms**	$752 \pm 168ms$
	3	**116 ± 22ms**	$362 \pm 93ms$
	4	**509 ± 33ms**	$1905 \pm 498ms$

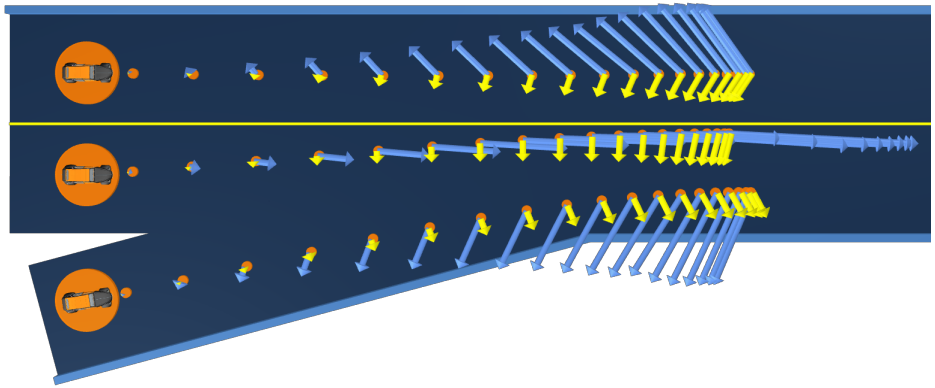


(b) Convergence of Algames and IBR on an unconstrained Nash equilibrium problem with different number of players.

Continua of Generalized Nash Equilibria

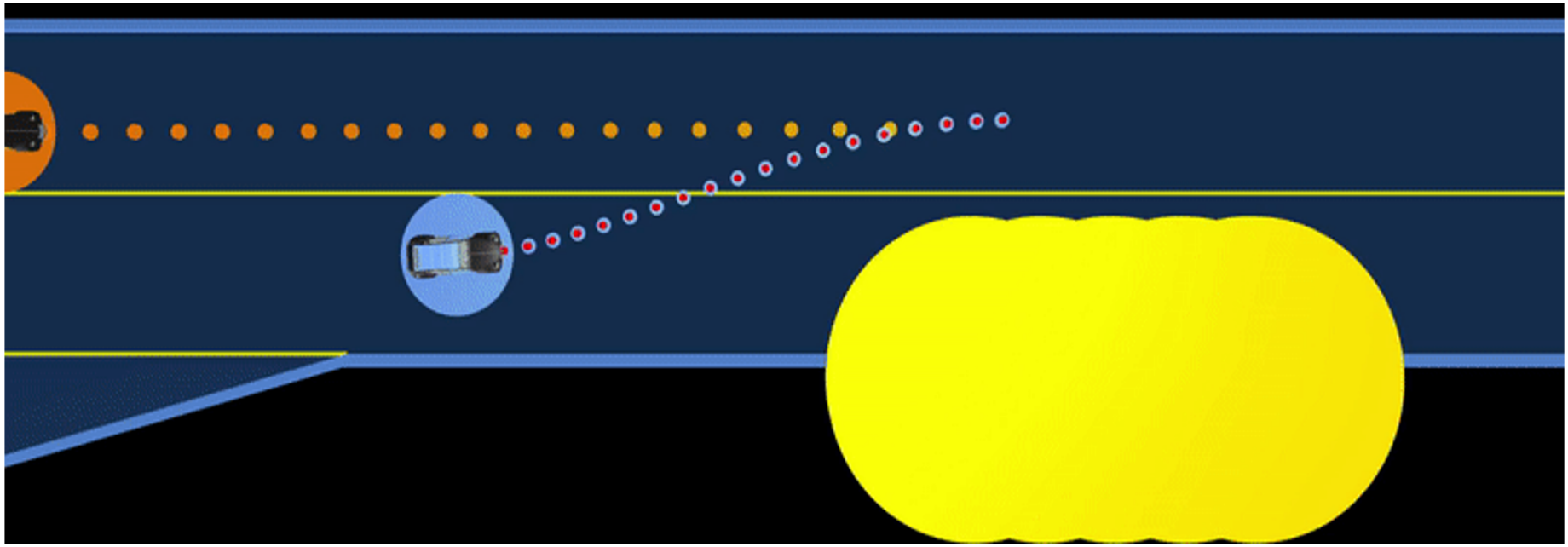






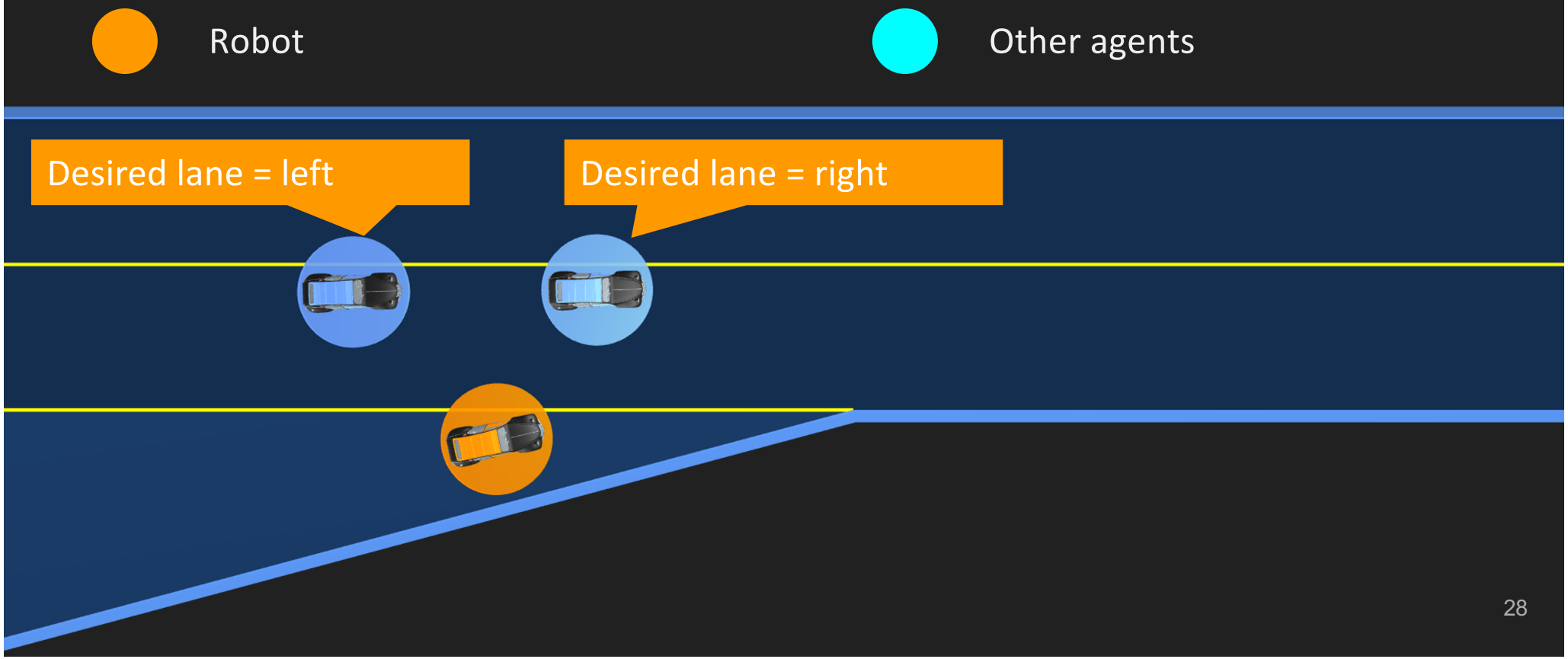
ALGAMES finds **Normalized Nash Equilibrium:** multipliers equal for different players on Shared constraints.

LUCIDGames

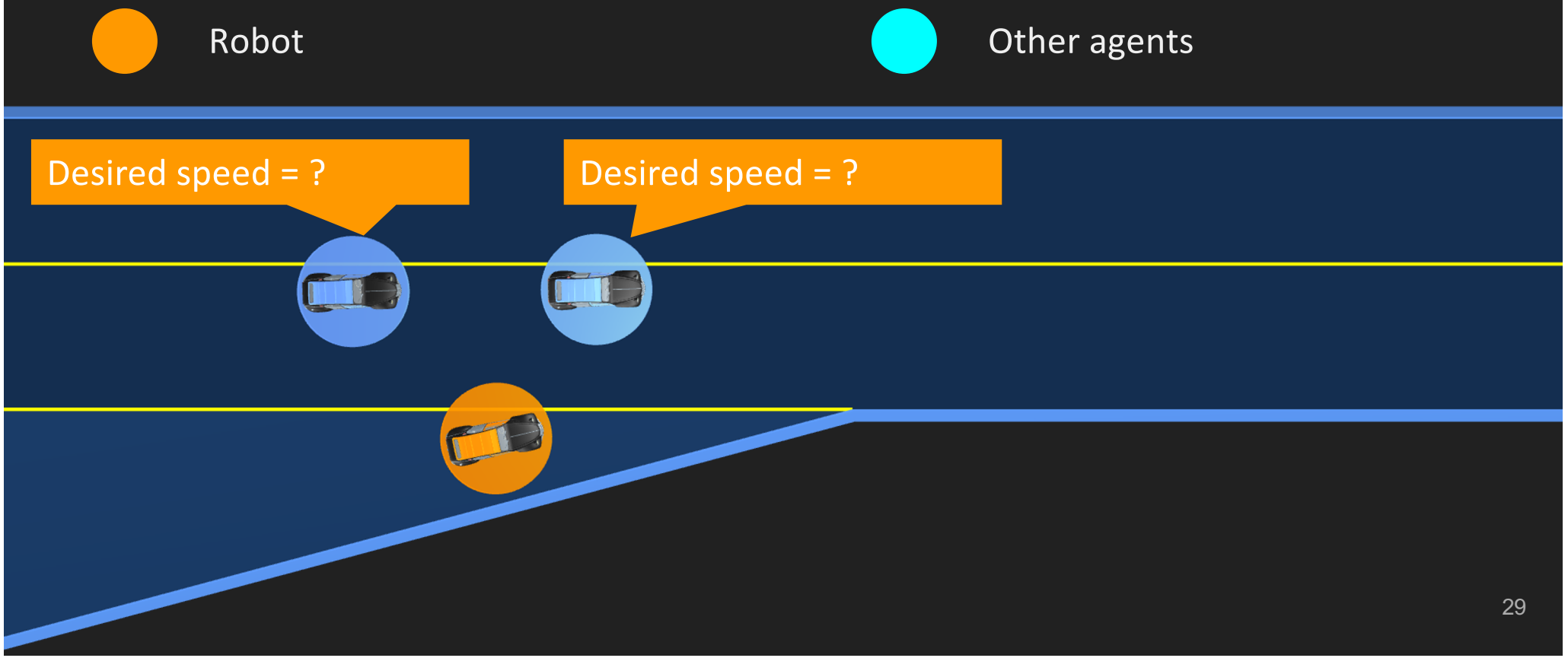


LeCleach, Schwager, Manchester, RA-L 2021.

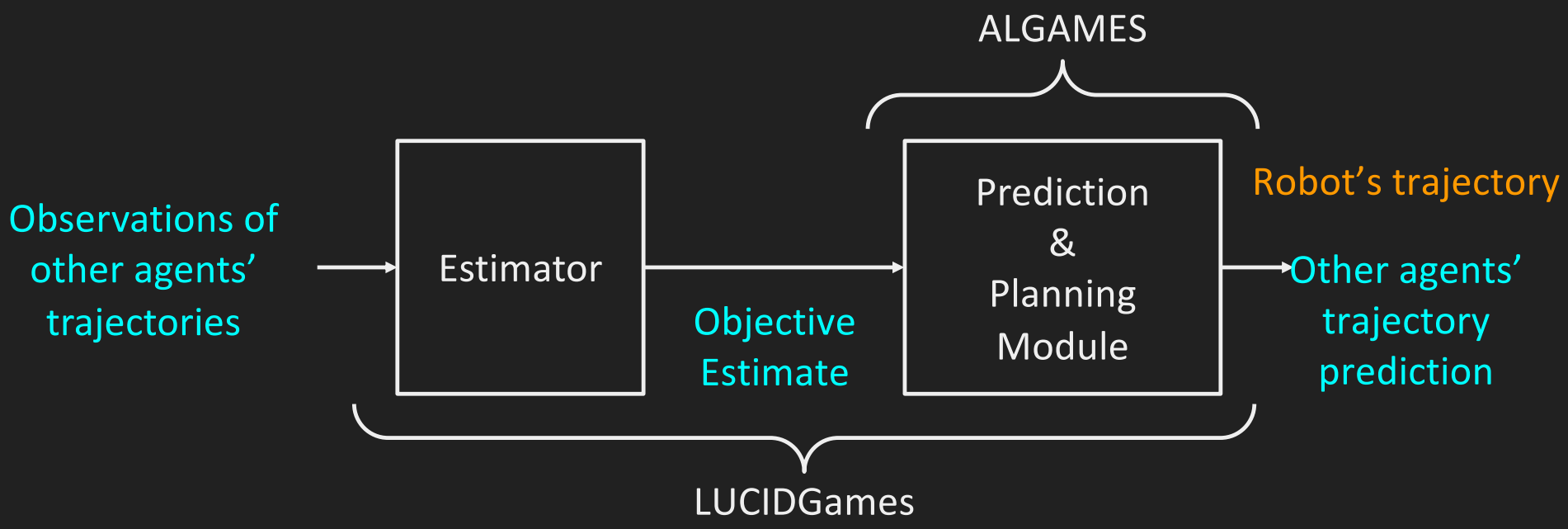
Online Objective Estimation



Online Objective Estimation



Overview


ALGAMES
Observations of other agents' trajectories
Estimator
Objective Estimate
Prediction & Planning Module
Robot's trajectory
Other agents' trajectory prediction
LUCIDGames

Objective Parameterization

Unknown

Human

$$\begin{cases} \min\limits_{X, U^{(1)}} & J^1(X, U^{(1)}), \\ \text{s.t.} & C(X, U) \leq 0. \end{cases}$$

Robot

$$\begin{cases} \min\limits_{X, U^{(2)}} & J^2(X, U^{(2)}), \\ \text{s.t.} & C(X, U) \leq 0. \end{cases}$$

Objective Parameterization

$$J^1(X, U^1) = \theta^T \phi(X, U^1)$$

Objective Parameterization

Features

$$J^1(X, U^1) = \theta^T \overbrace{\phi(X, U^1)}$$

Objective Parameterization

Features

$$J^1(X, U^1) = \underbrace{\theta^T}\overbrace{\phi(X, U^1)}$$

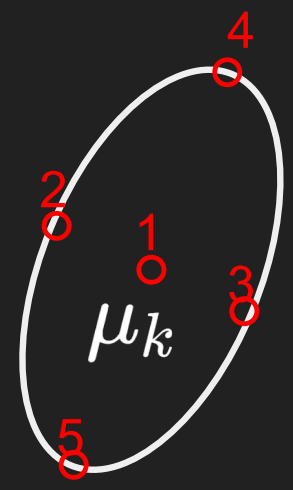
Weighting vector

UKF Algorithm

Update estimate $\theta_{k+1} \sim \mathcal{N}(\mu_{k+1}, \Sigma_{k+1})$

1. Sample sigma-points from $\theta_k \sim \mathcal{N}(\mu_k, \Sigma_k)$

1. Predict $\hat{x}_k$ given x_{k-1} and a sigma-point



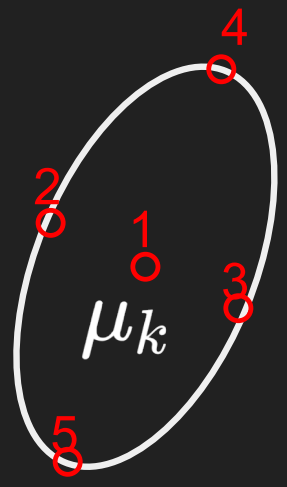


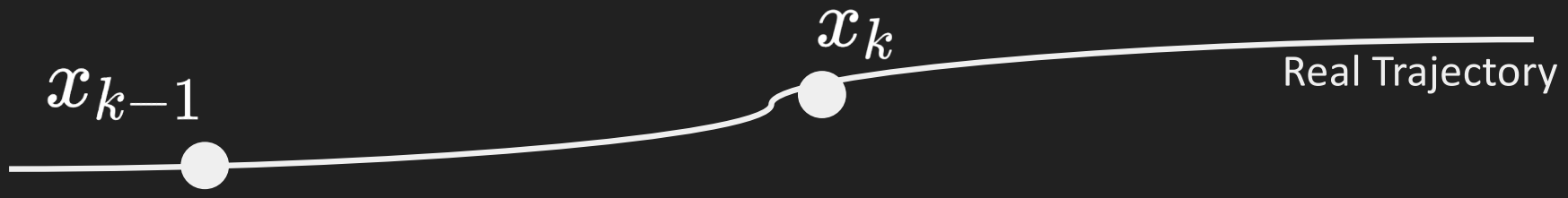
UKF Algorithm

Update estimate $\theta_{k+1} \sim \mathcal{N}(\mu_{k+1}, \Sigma_{k+1})$

1. Sample sigma-points from $\theta_k \sim \mathcal{N}(\mu_k, \Sigma_k)$

1. Predict $\hat{x}_k$ given x_{k-1} and a sigma-point

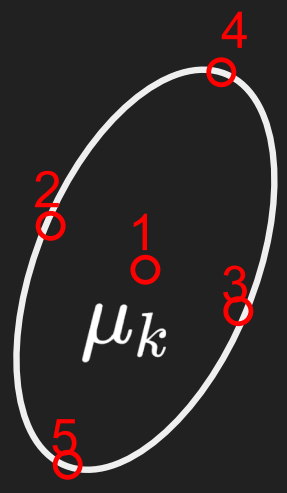


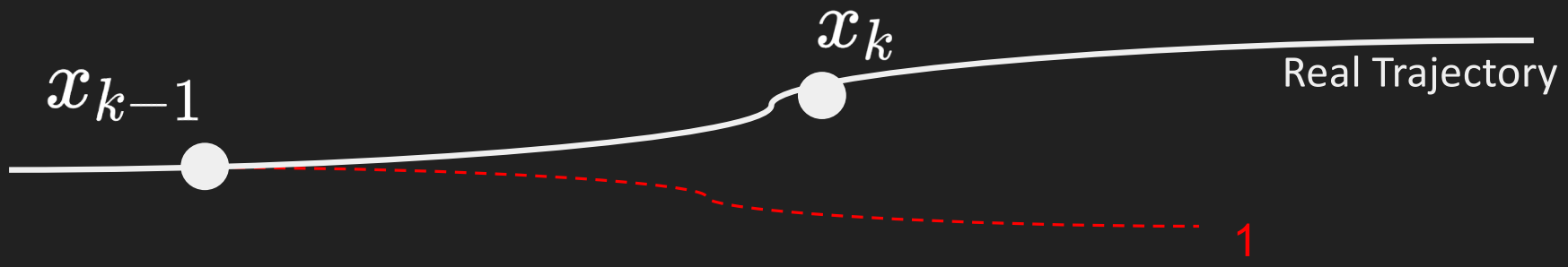


UKF Algorithm

Update estimate $\theta_{k+1} \sim \mathcal{N}(\mu_{k+1}, \Sigma_{k+1})$

1. Sample sigma-points from $\theta_k \sim \mathcal{N}(\mu_k, \Sigma_k)$
1. Predict $\hat{x}_k$ given x_{k-1} and a sigma-point

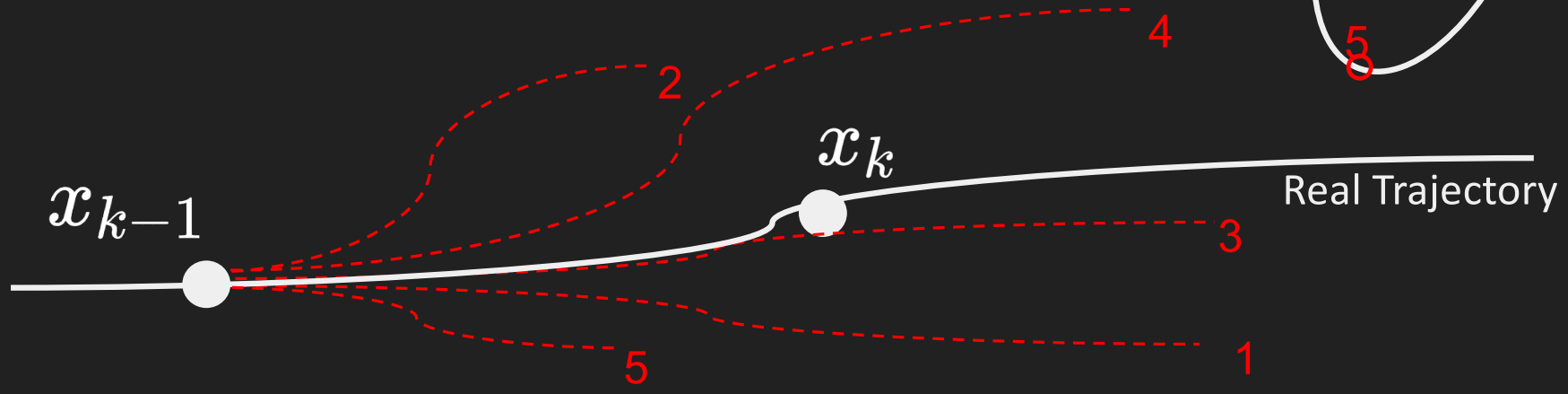




UKF Algorithm

Update estimate $\theta_{k+1} \sim \mathcal{N}(\mu_{k+1}, \Sigma_{k+1})$

1. Sample sigma-points from $\theta_k \sim \mathcal{N}(\mu_k, \Sigma_k)$
1. Predict $\hat{x}_k$ given x_{k-1} and a sigma-point

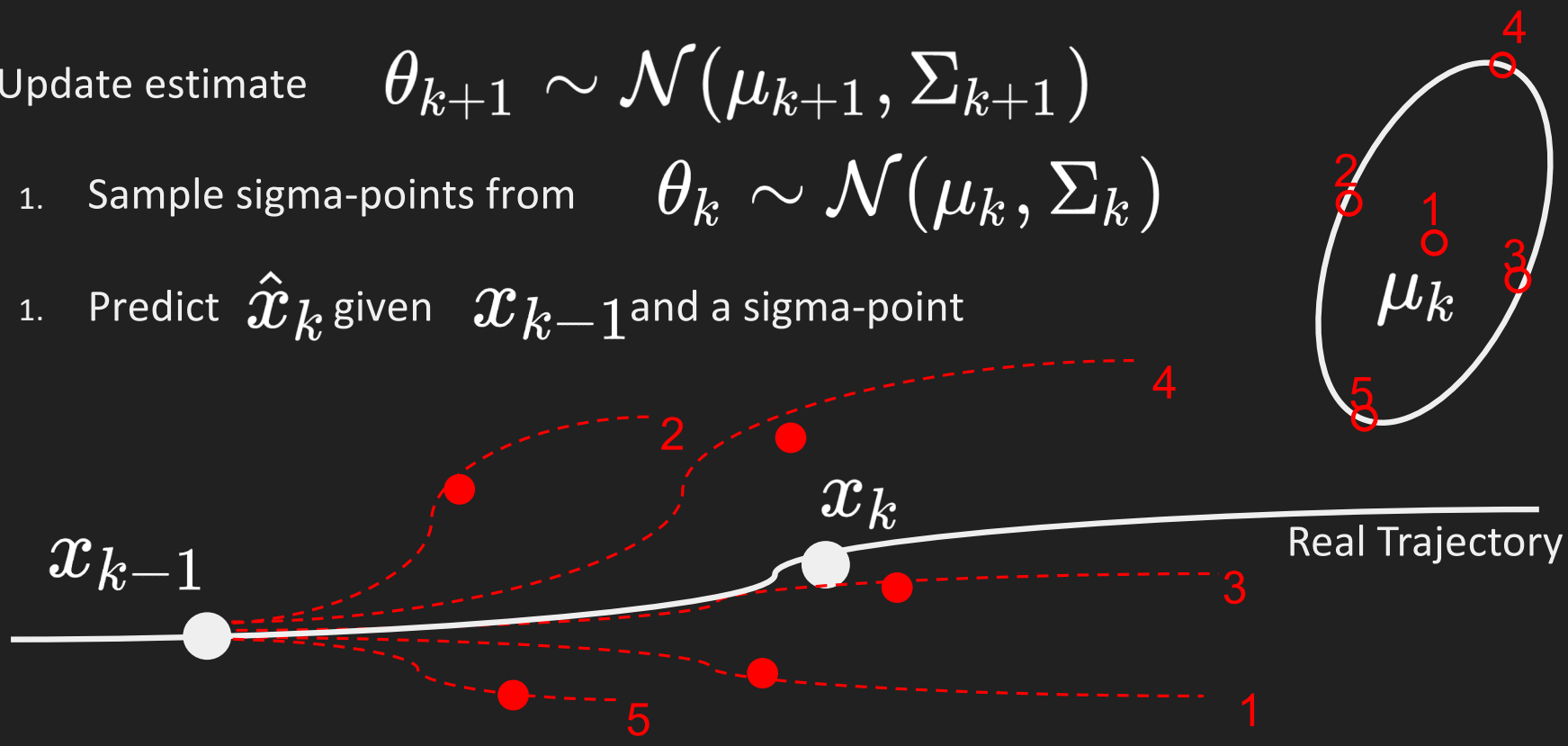


UKF Algorithm

Update estimate $\theta_{k+1} \sim \mathcal{N}(\mu_{k+1}, \Sigma_{k+1})$

1. Sample sigma-points from $\theta_k \sim \mathcal{N}(\mu_k, \Sigma_k)$

1. Predict $\hat{x}_k$ given x_{k-1} and a sigma-point

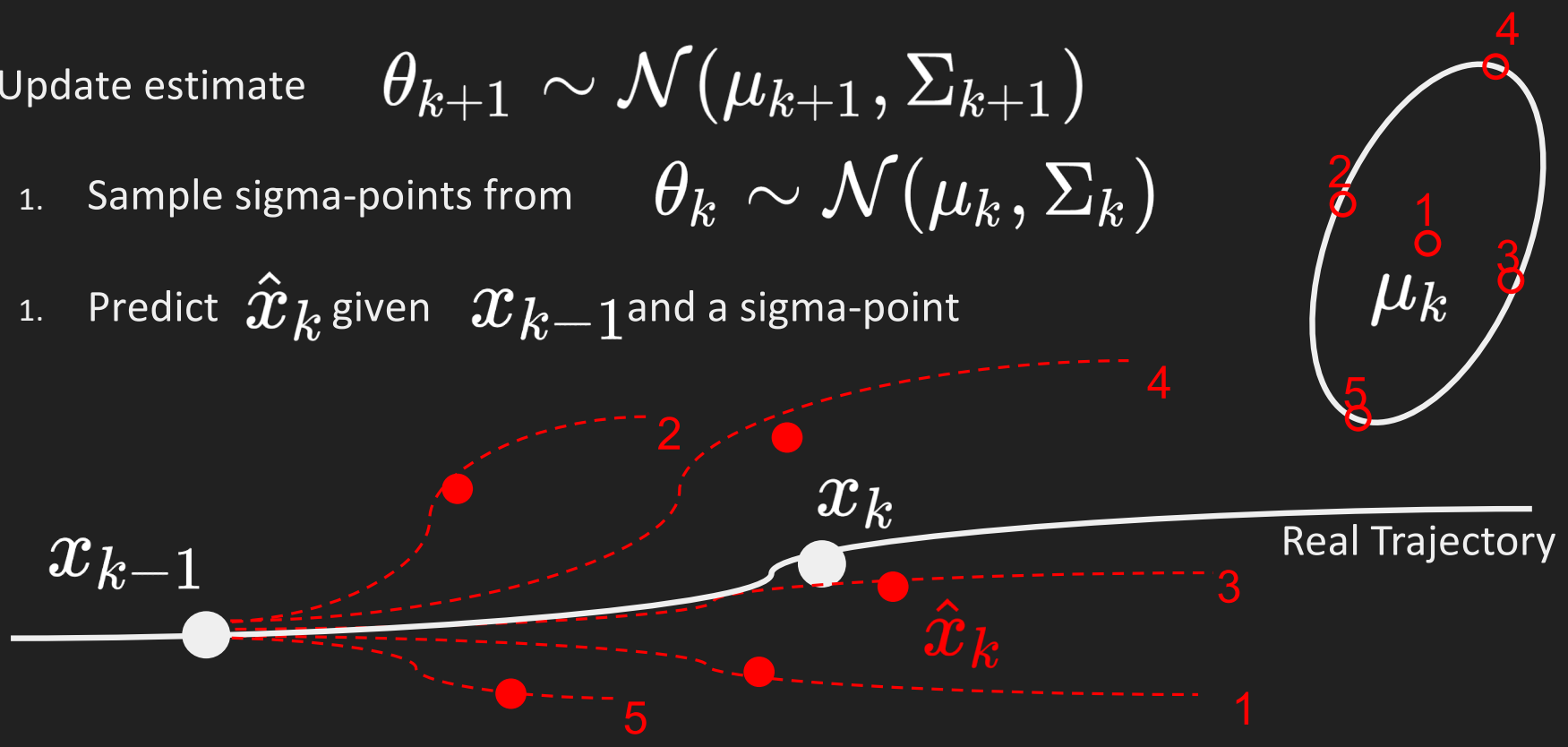


UKF Algorithm

Update estimate $\theta_{k+1} \sim \mathcal{N}(\mu_{k+1}, \Sigma_{k+1})$

1. Sample sigma-points from $\theta_k \sim \mathcal{N}(\mu_k, \Sigma_k)$

1. Predict $\hat{x}_k$ given x_{k-1} and a sigma-point

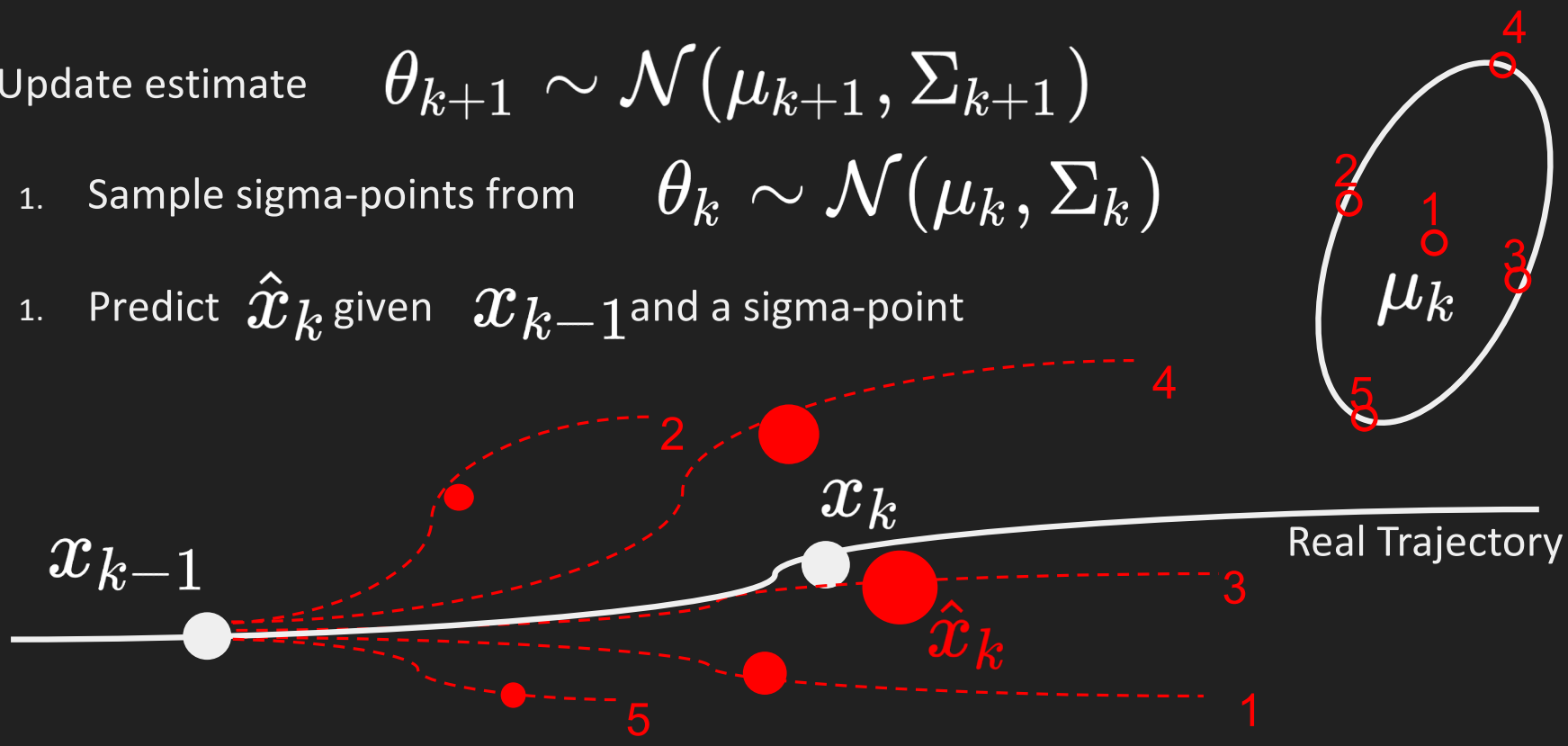


UKF Algorithm

Update estimate $\theta_{k+1} \sim \mathcal{N}(\mu_{k+1}, \Sigma_{k+1})$

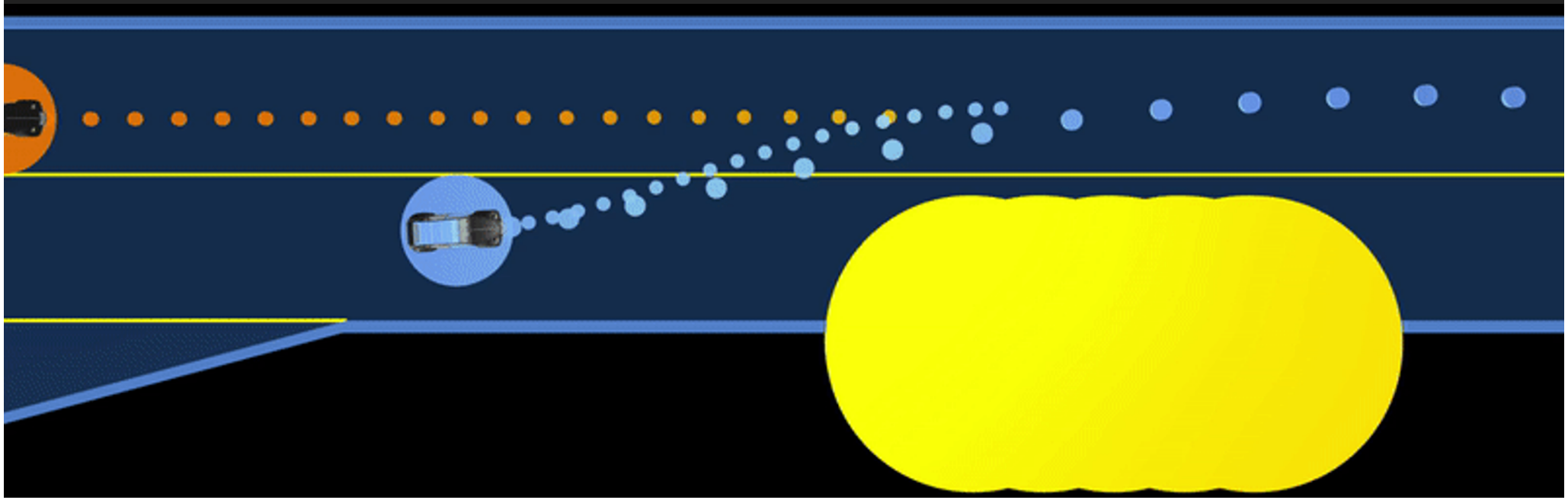
1. Sample sigma-points from $\theta_k \sim \mathcal{N}(\mu_k, \Sigma_k)$

1. Predict $\hat{x}_k$ given x_{k-1} and a sigma-point



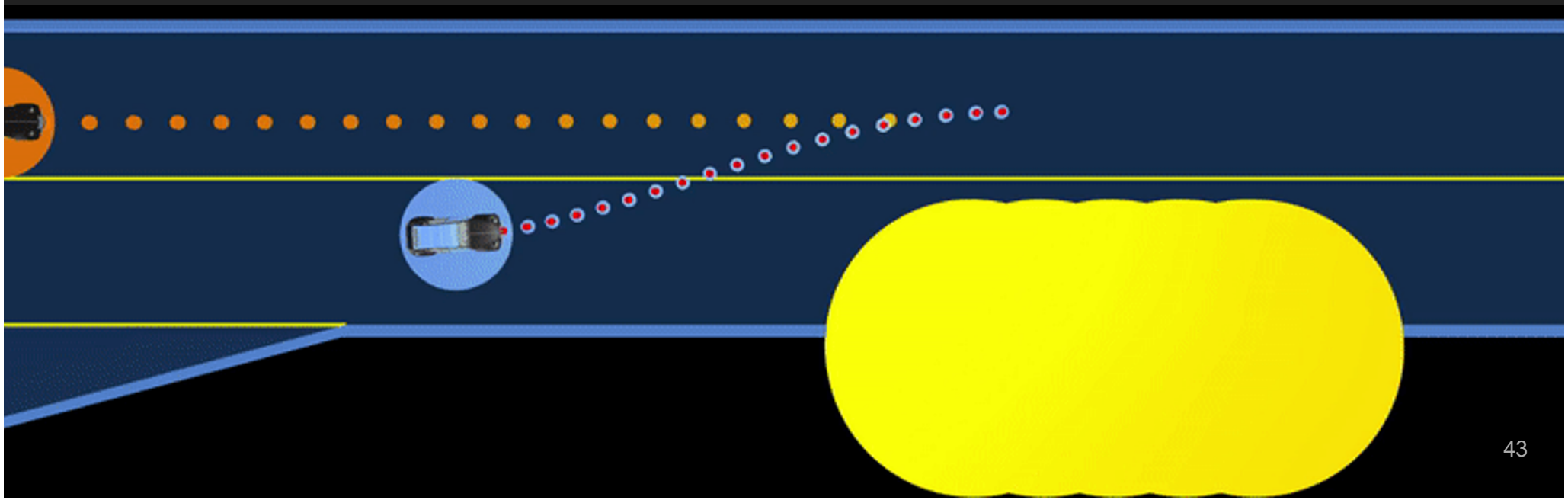
Results

Obstacle avoidance scenario



Results

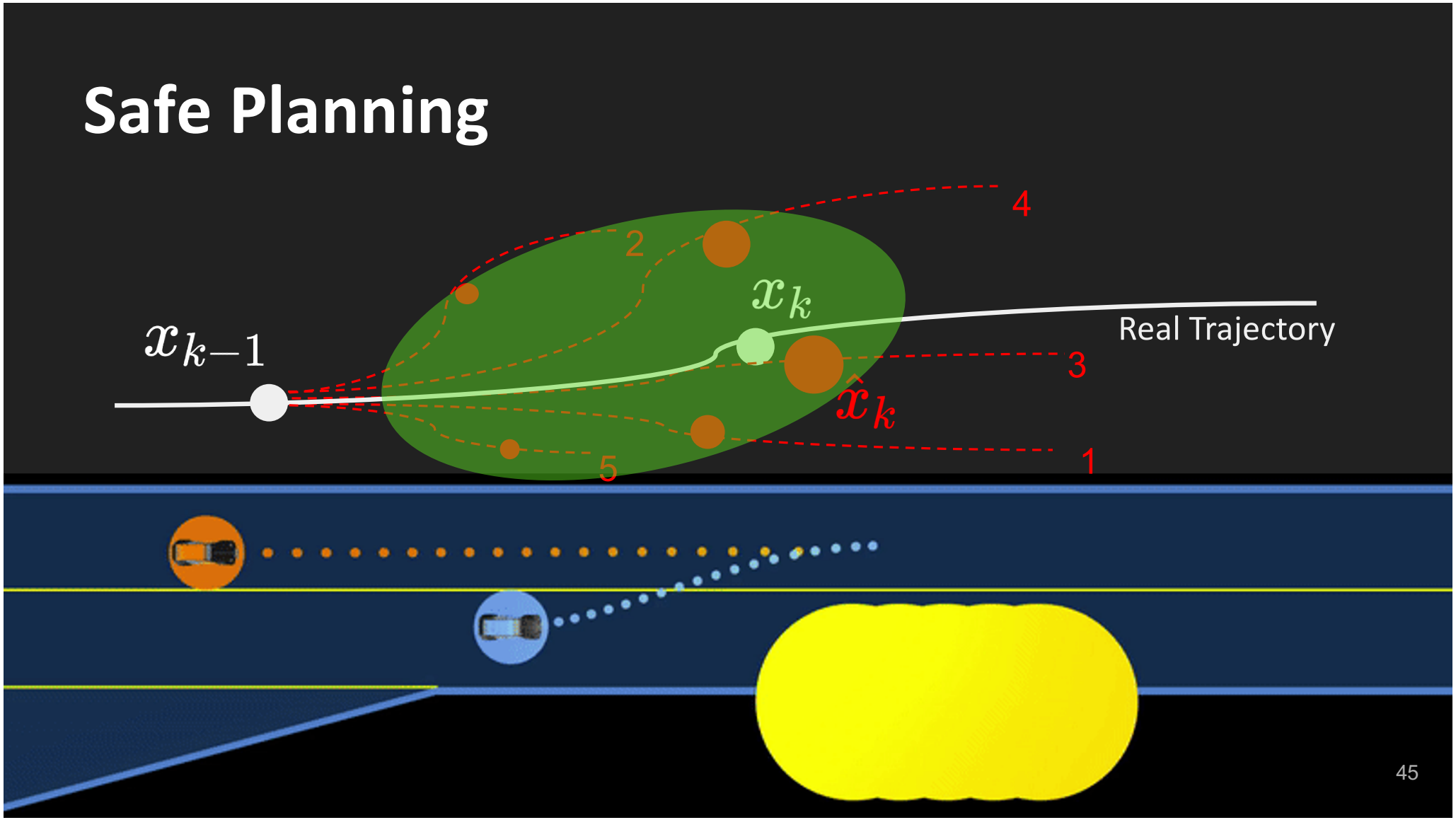
Obstacle avoidance scenario



Tractability

# of Players	Freq. in Hz	$\mathbb{E}[\delta t]$ in ms	$\sigma[\delta t]$ in ms
2	132	7.55	16.7
3	38.0	26.3	37.7
4	11.4	87.3	94.0

Safe Planning





Simon Le Cleac’h
simonlc@stanford.edu



Mac Schwager
schwager@stanford.edu



Zachary Manchester
zacmanchester@stanford.edu

https://github.com/RoboticExplorationLab/ALGAMES.jl.git