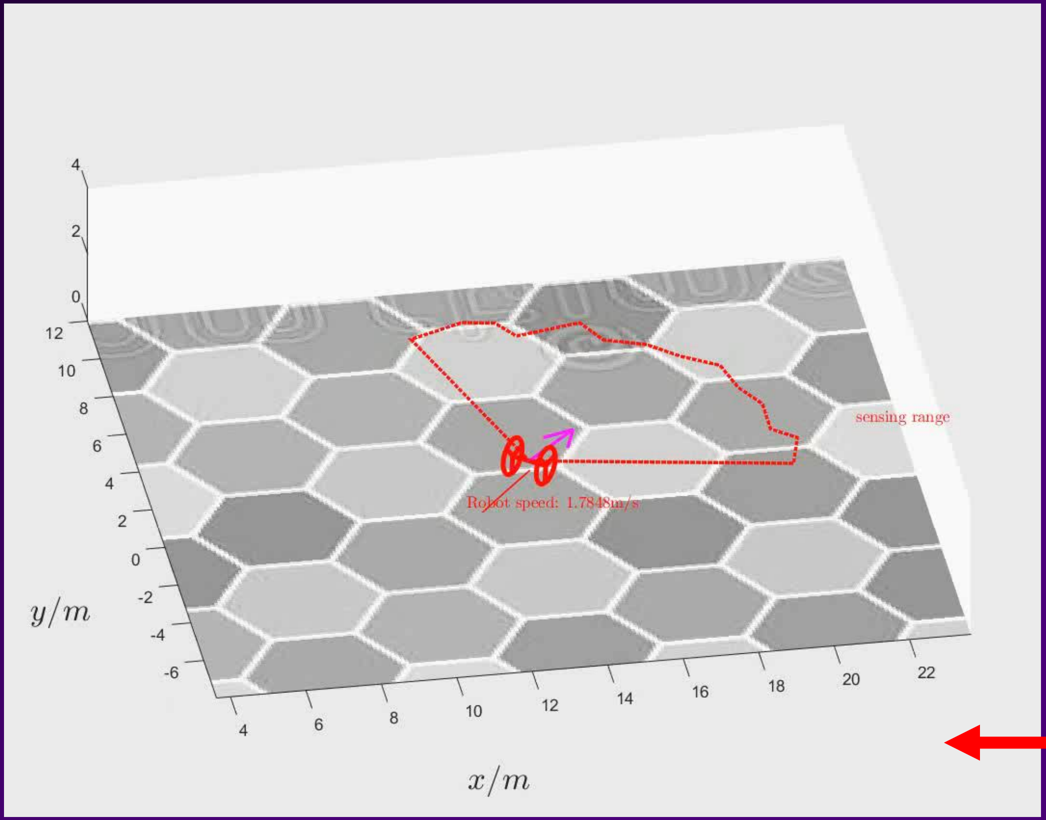
SAFE OPTIMAL CONTROL OF AUTONOMOUS VEHICLES WITH CONTROL BARRIER FUNCTIONS

C. G. Cassandras

Division of Systems Engineering
Dept. of Electrical and Computer Engineering
Center for Information and Systems Engineering
Boston University

https://christosgcassandras.org

MOTIVATION: SAFE-CRITICAL TASKS



Reach **destination**…

…**safely**

(avoid obstacles with unknown locations)

…with **minimal** effort

(minimize energy)

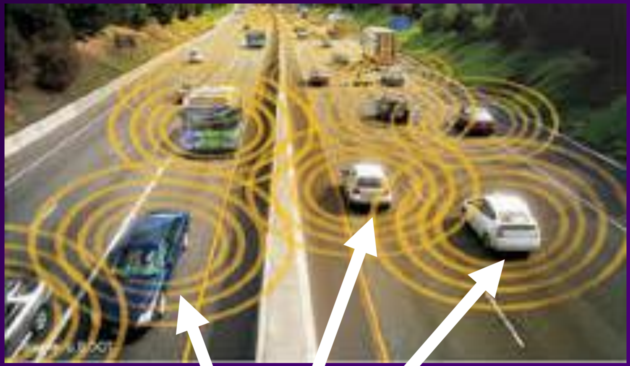
Actual execution using CBF-based optimal control

Design autonomous systems which can *guarantee*

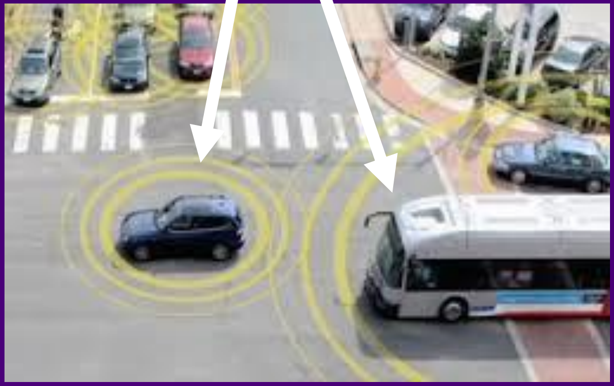
- **SAFE** and **OPTIMAL** control actions
- …throughout a task
- …with minimal or no supervision

MOTIVATION – THE INTERNET OF CARS

GAME-CHANGING OPPORTUNITY:
CONNECTED AUTONOMOUS VEHICLES (CAVs)



CAVs



FROM (SELFISH) “DRIVER OPTIMAL” TO (SOCIAL) “SYSTEM OPTIMAL” TRAFFIC CONTROL

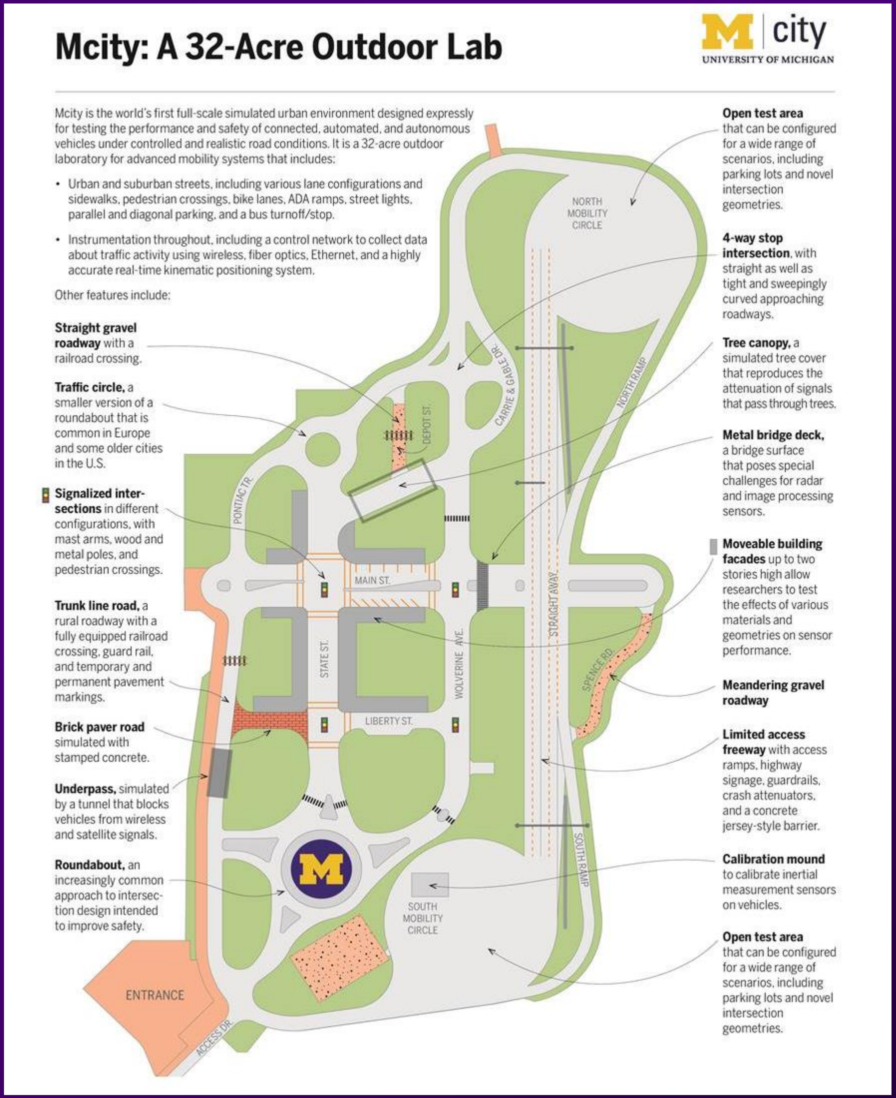
THE “INTERNET OF **CARS**”

NO TRAFFIC LIGHTS, NEVER STOP…

A TEST BED FOR CAVs

Mcity test bed, U. Michigan

2016 Audi A3 e-tron PHEV (ARPA-E project)



CONTROL ZONES

CONTROL ZONE (CZ):

Vehicles can cooperate to achieve desirable performance

Minimize Travel Time through CZ

+

Minimize Energy through CZ

+

Maximize Passenger Comfort

+

Guarantee Safety

FROM TRAJECTORY PLANNING TO EXECUTION: CONTROL BARRIER FUNCTIONS (CBFs)

Ames, Grizzle, Tabuada, CDC 2014; Xiao, Cassandras, Belta, *Automatica, 2021*

OPTIMAL CONTROL v REAL-TIME EXECUTION

OPTIMAL CONTROL

MODEL PREDICTIVE CONTROL

Optimality *BUT…*

- Hard to obtain solutions when constraints active
- Time consuming to compute
- Models are too simple
- Not robust to disturbances

- **Easy to solve *BUT…***

- Suboptimal
- Linearization errors
- Models still too simple
- Computationally expensive

OPTIMAL CONTROL v REAL-TIME EXECUTION

OPTIMAL CONTROL

CONTROL BARRIER FUNCTIONS

Optimality *BUT…*

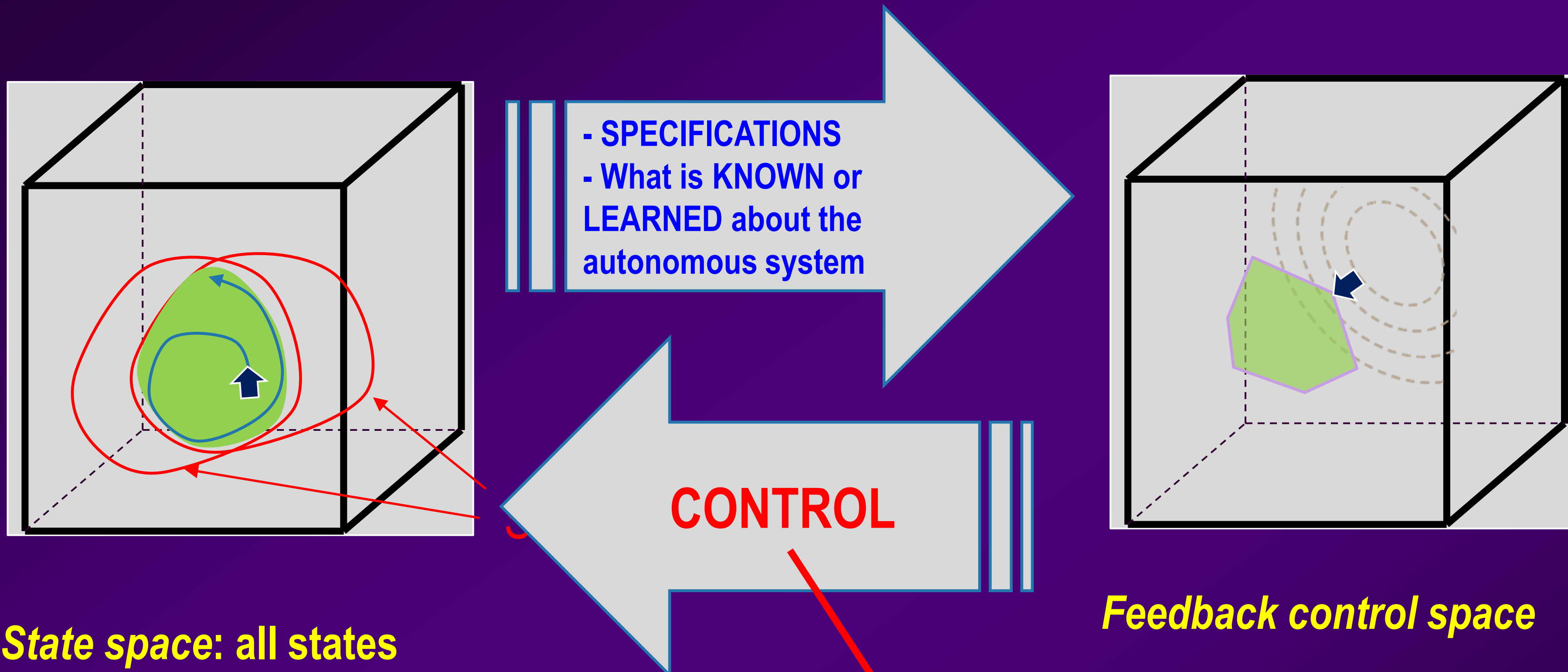
- Hard to obtain when constraints active
- Time consuming to compute
- Models are too simple
- Not robust to disturbances

- Easy to solve
- *Forward Invariance*
- Nonlinear Dynamics and Constraints
- Complex objectives
- Noise in dynamics

BUT…

- Suboptimal
- Possibly conservative, myopic

CONTROL BARRIER FUNCTIONS - OVERVIEW



State space: all states that autonomous system can access (e.g., positions, speeds,…)

FORWARD INVARIANCE:
Controller guarantees SAFETY over all future times

→ *Control Barrier Functions* (CBFs)

CONTROL BARRIER FUNCTIONS - OVERVIEW

- **System dynamics:** $\dot{x} = f(x) + g(x)u$
- **Control Barrier Functions:** (continuously differentiable *state* constraints) $b_k(x) \geq 0$
- **Map each CBF onto another constraint on the *control* input such that the satisfaction of this new constraint implies the satisfaction of the original constraint:**

IMPLIES

CBF constraints: $\boxed{L_f b_k(x(t)) + L_g b_k(x(t))u(t) + \gamma(b_k(x(t))) \geq 0}$

Lie derivatives along f and g

Class K function: strictly increasing, $\gamma(0) = 0, \quad \gamma(t) = \rightarrow \infty \text{ as } t \rightarrow \infty$

$$L_f V(x) \triangleq \frac{\partial V}{\partial x} f(x) \quad \text{and} \quad L_g V(x) \triangleq \frac{\partial V}{\partial x} g(x)$$

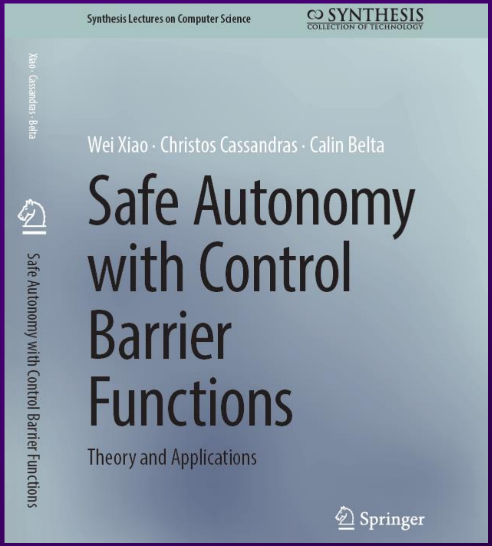
CONTROL BARRIER FUNCTIONS - OVERVIEW

***CBF constraint*:** $$L_f b_k(x(t)) + L_g b_k(x(t)) u(t) + \gamma(b_k(x(t))) \geq 0$$

Linear in u

This inequality provides a set of feedback controls that guarantee $b_k(x) \geq 0$ for all future times ⇨ FORWARD INVARIANCE

Details, Proofs, Applications in….



CONTROL BARRIER FUNCTIONS - OVERVIEW

- **Control Lyapunov Functions**:
  (can be used to stabilize one or more state variable)

CLF constraints: $L_f V(x(t)) + L_g V(x(t))u(t) + \varepsilon V(x(t))) \leq \delta(t)$

Example: $(v_i(t) - v_{ref}(t))^2$

Soft constraint parameter

CONTROL BARRIER FUNCTIONS - OVERVIEW

- **Solve optimization problem:**

$$\min_{\boldsymbol{u}(t),\delta(t)} \int_0^T ||\boldsymbol{u}(t)||^2 + p\delta^2(t)dt$$

subject to

$$\dot{x} = f(x) + g(x)u$$

State dynamics

$$L_f b_k(x(t)) + \ldots(t))u(t) + \gamma(b_k(x(t))) \geq 0$$

HARD safety constraints

CBF constraints

$$L_f V(x(t)) + L_g V(x(t))u(t) + \varepsilon V(x(t)) \ldots$$

SOFT constraints

CLF constraints

$$u_{min} \leq u \leq u_{max}$$

Control constraints

CONTROL BARRIER FUNCTIONS - OVERVIEW

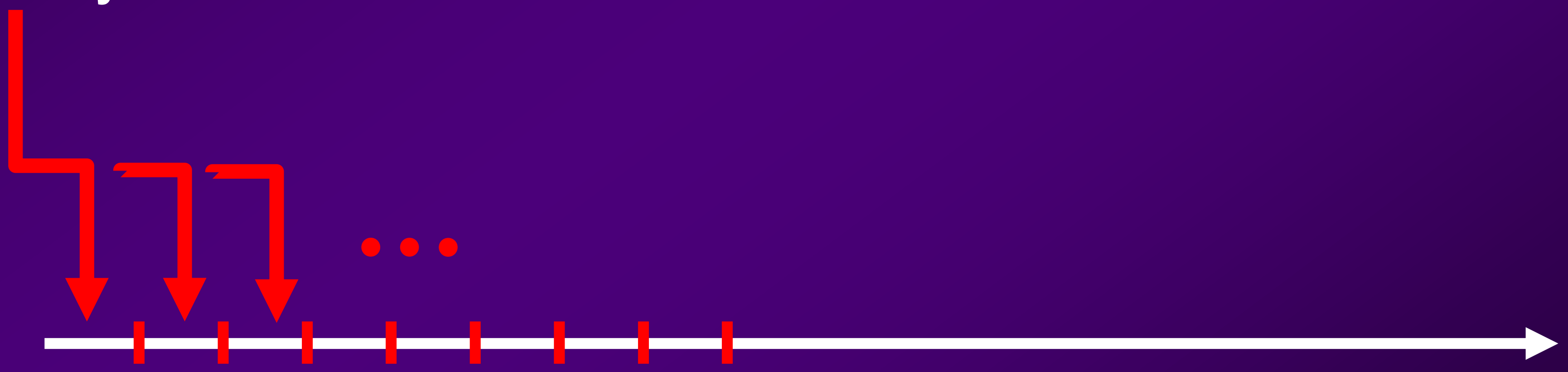
- **Discretize time and solve a sequence of Quadratic Programs (QPs):**

$$(\boldsymbol{u}^*(t_k), \delta^*(t_k)) = \arg \min_{\boldsymbol{u}(t_k), \delta(t_k)} \|\boldsymbol{u}(t_k)\|^2 + p\delta^2(t_k)$$

s.t. Vehicle Dynamics, CBF constraints, CLF constraints

Linear in u_k

- **Easy to solve**



SEVERAL ISSUES TO RESOLVE…

1. What if control $u(t)$ does not show up in CBF constraint?

$$L_f b_k(x(t)) + \underbrace{\boxed{L_g b_k(x(t))}}_{=0\ ???} u(t) + \gamma(b_k(x(t))) \geq 0 \qquad (*)$$

2. Is each QP problem always **FEASIBLE** ?

3. Since CBF constraint is only a sufficient condition for the original constraint, how **CONSERVATIVE** are the controls that come from $(*)$?

4. What if the environment/system is **TIME-VARYING**?

5. What if **DYNAMICS ARE UNKNOWN**?

ISSUE 1: HIGH ORDER CBFs (HOCBFs)

KEY IDEA: Keep taking derivatives of $b(x)$ until u shows up

Define a sequence of functions $\psi_i(x)$:

Class K function

$$\psi_0(\boldsymbol{x}) := b(\boldsymbol{x})$$

$$\psi_i(\boldsymbol{x}) := \dot{\psi}_{i-1}(\boldsymbol{x}) + \alpha_i(\psi_{i-1}(\boldsymbol{x})), \quad i \in \{1, \dots, m\}$$

m **: Relative degree of $b(x)$ = number of times it needs to be differentiated along its dynamics until the control u explicitly shows in the corresponding derivative.**

***HOCBF constraints*:**

$$\sup_{\boldsymbol{u} \in U} [L_f^m b(\boldsymbol{x}) + [L_g L_f^{m-1} b(\boldsymbol{x})]\boldsymbol{u} + S(b(\boldsymbol{x})) + \alpha_m(\psi_{m-1}(\boldsymbol{x}))] \geq 0$$

Lie derivatives along f and g

Remaining Lie derivatives

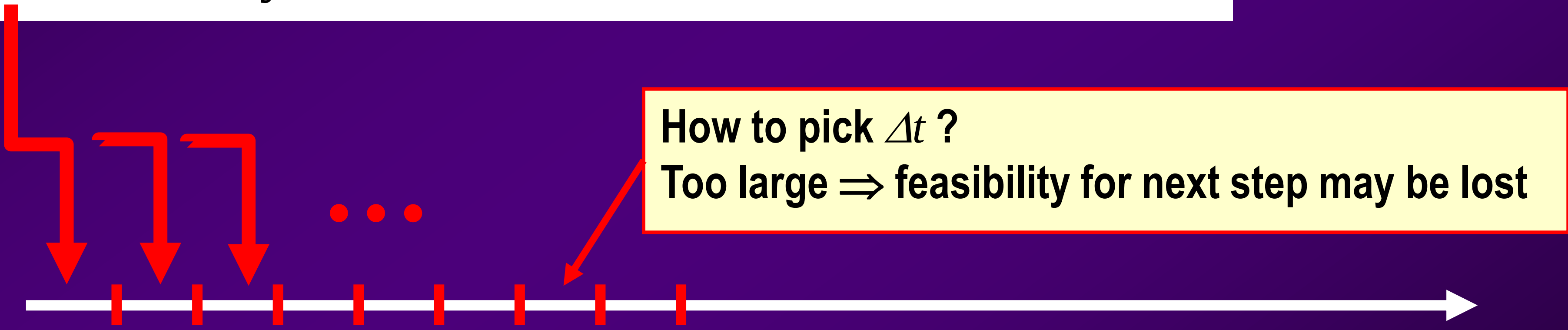
Class K function

ISSUE 2: QP FEASIBILITY → EVENT-DRIVEN CONTROL

KEY IDEA: REPLACE TIME-DRIVEN CONTROL BY EVENT-DRIVEN CONTROL

$$(\boldsymbol{u}^*(t_k), \delta^*(t_k)) = \arg \min_{\boldsymbol{u}(t_k),\delta(t_k)} \|\boldsymbol{u}(t_k)\|^2 + p\delta^2(t_k)$$

s.t. Vehicle Dynamics, CBF constraints, CLF constraints

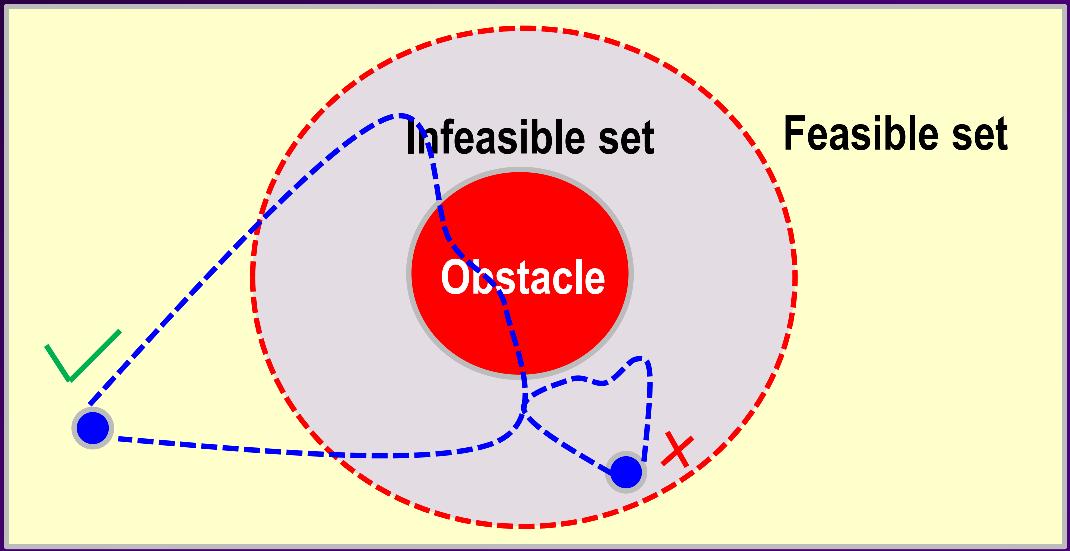


Instead, trigger the next QP based on appropriate EVENTS
Side benefit: increased unpredictability ⇒ extra security !

Xiao, Belta, Cassandras, *IEEE Trans. on Automatic Control*, 2023

ISSUE 3: CONSERVATIVENESS → ADAPTIVE CBFs

KEY IDEA: ADAPTIVE CBFs TO MAXIMIZE FEASIBILITY ROBUSTNESS



If constraint becomes active before detection of obstacle, then safety is not guaranteed *(e.g., too late to decelerate/turn away)*

Can we shrink infeasible set? (which may be hard to identify…)
$\Rightarrow$ Make constraint active **as late as possible**

Define a FEASIBILITY ROBUSTNESS metric and adapt CBF constraint to maximize it

ADDED CHALLENGE: *Learn INFEASIBLE SETS*

ISSUE 4: TIME-VARYING ENVIRONMENTS → ADAPTIVE CBFs

KEY IDEA: ADAPTIVE CBFs

Revisit ADAPTIVE CBFs with more flexible forms of adaptivity, e.g.,

$$L_f^m b_j(\boldsymbol{x}) + L_g L_f^{m-1} b_j(\boldsymbol{x})\boldsymbol{u} + O(b_j(\boldsymbol{x})) + \alpha_m(\psi_{m-1}(\boldsymbol{x})) \geq 0$$

$$\psi_i(\boldsymbol{x}) := \dot{\psi}_{i-1}(\boldsymbol{x}) + p_i \psi_{i-1}^{q_i}(\boldsymbol{x}), \quad i \in \{1, \dots, m\}$$

Non-negative parameters

Learn optimal parameters

Xiao, Belta, Cassandras, *Automatica*, 2022

ISSUE 5: UNKNOWN DYNAMICS → EVENT-DRIVEN CONTROL

1. Define **ADAPTIVE AFFINE DYNAMICS** $\dot{\bar{\boldsymbol{x}}} = f_a(\bar{\boldsymbol{x}}) + g_a(\bar{\boldsymbol{x}})\boldsymbol{u}$

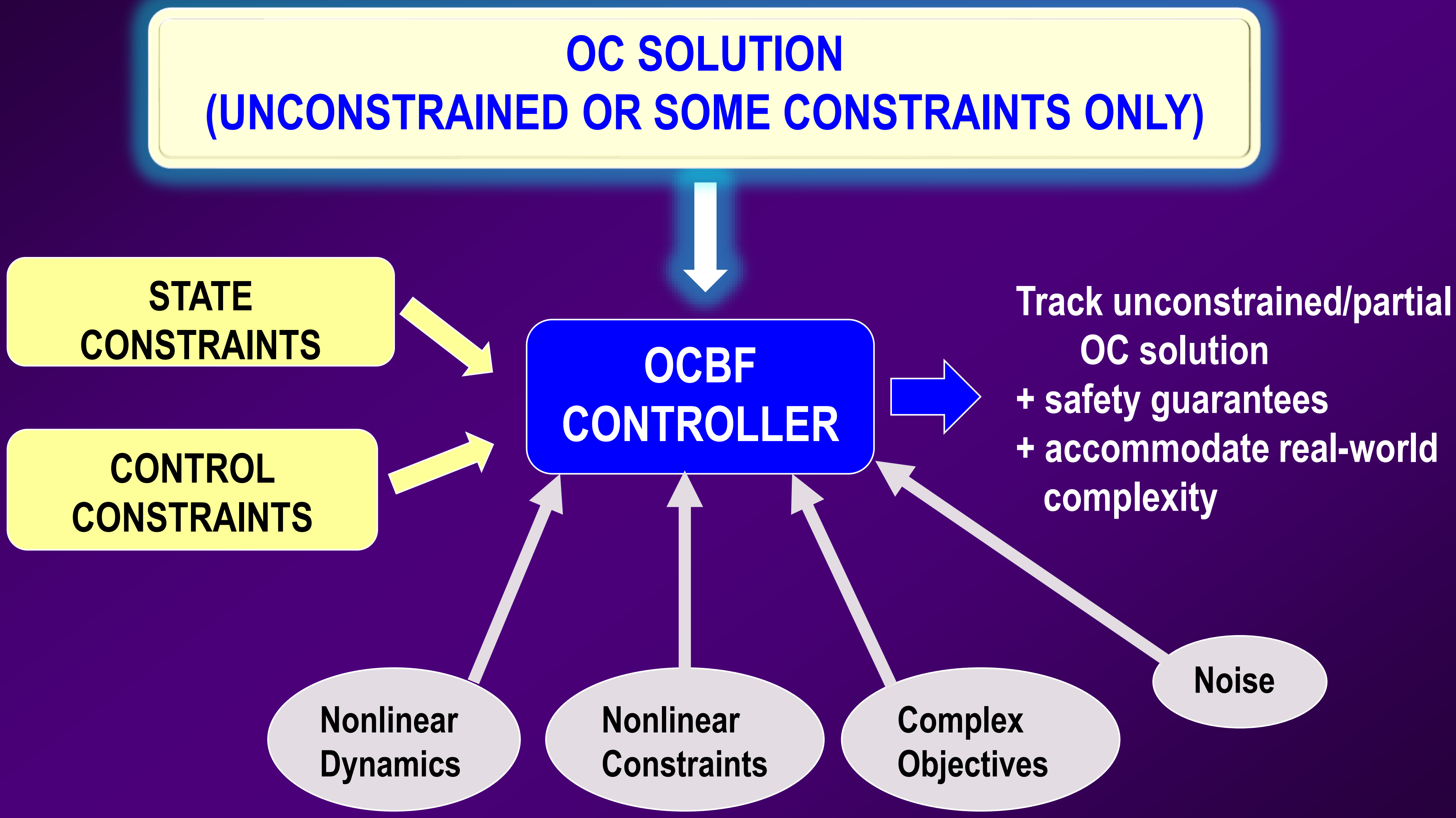
2. Find HOCBF that guarantees $b(\boldsymbol{x}(t)) \geq 0, \forall t \in [0, T]$

3. Formulate CBF-based QP

4. Determine EVENTS required to solve QPs + condition that guarantees $b(\boldsymbol{x}(t)) \geq 0, \forall t \in [0, T]$ between events

Xiao, Belta, Cassandras, *IEEE Trans. on Automatic Control*, 2023

OC + CBF APPROACH (OCBF): OVERVIEW



OPTIMAL CONTROL

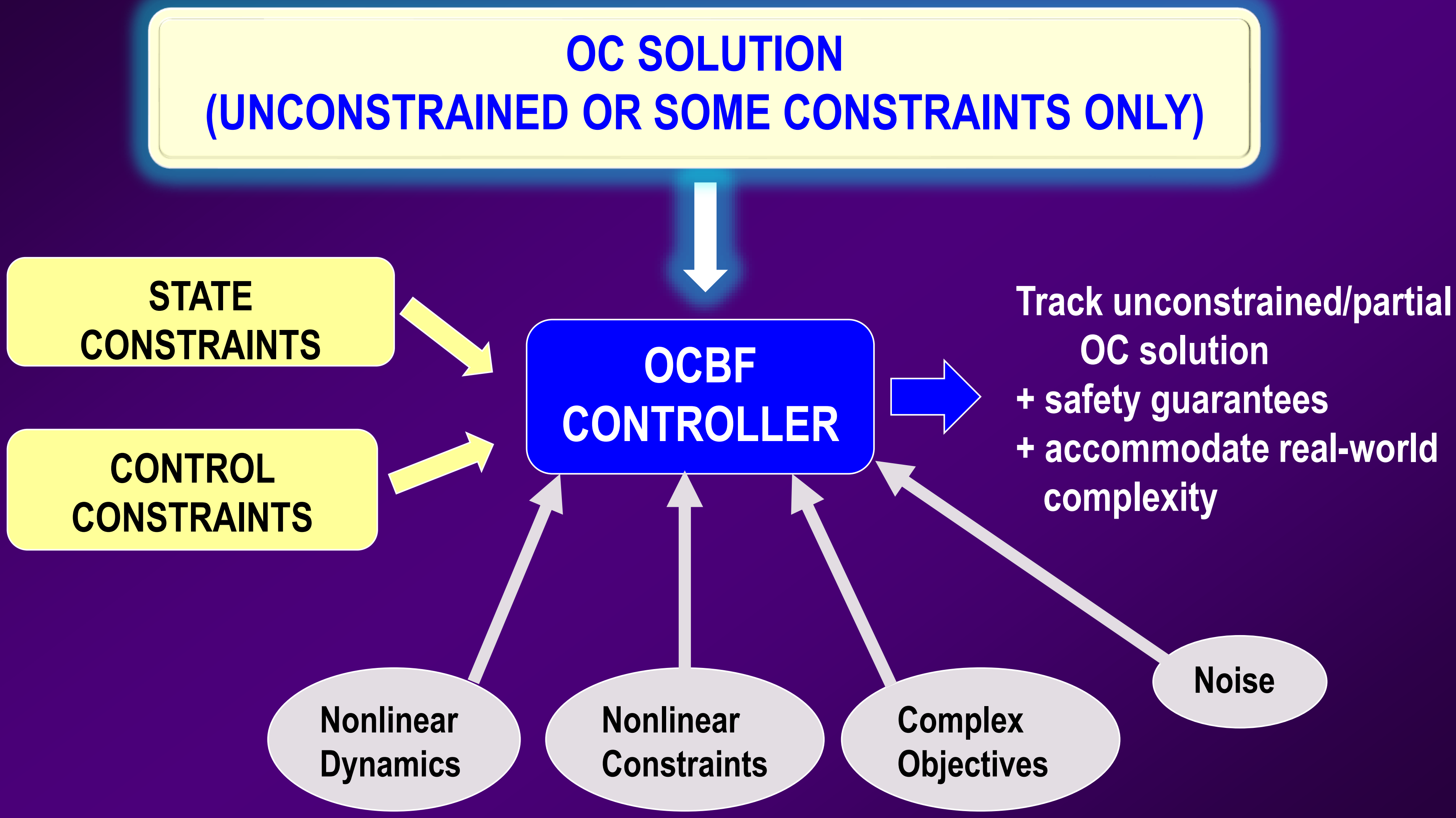
+

CBFs

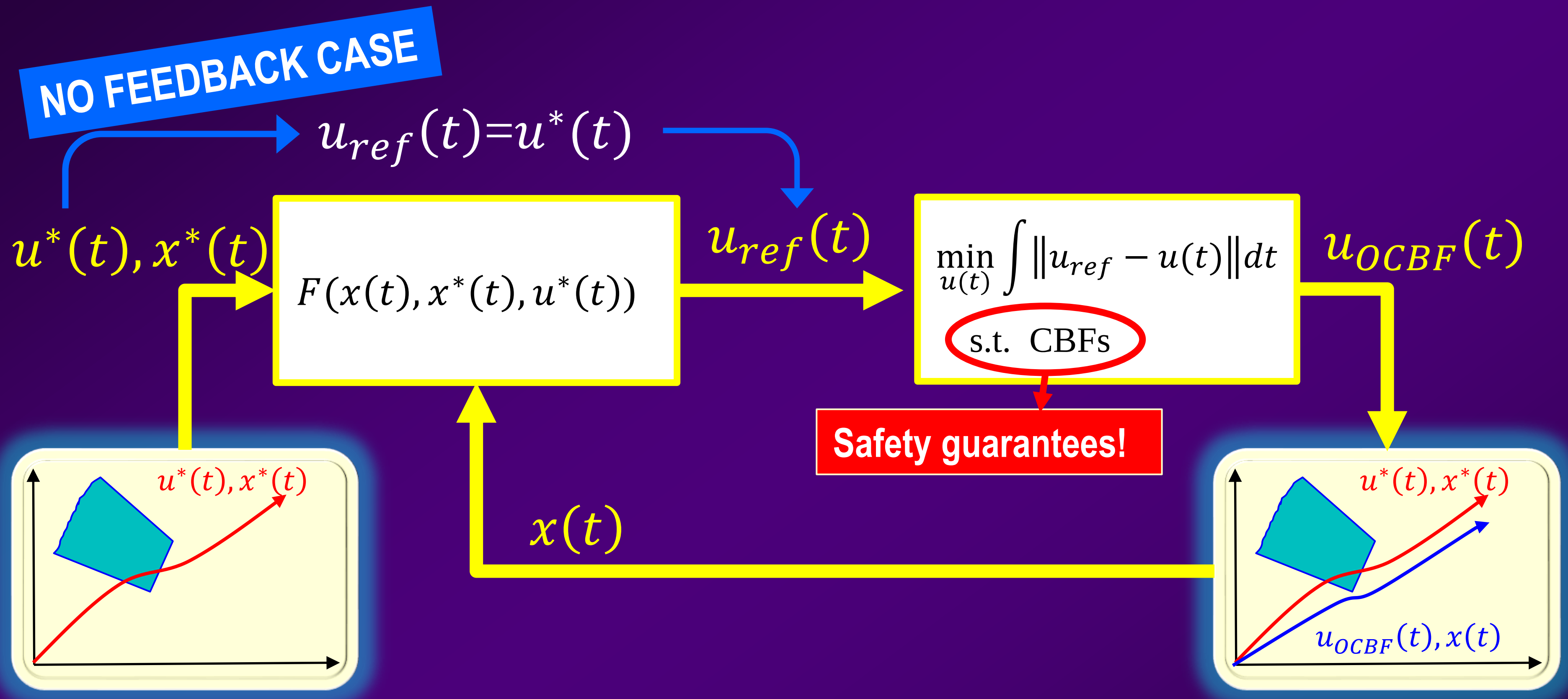
⬇

OCBF CONTROL

OC + CBF APPROACH (OCBF): OVERVIEW



OCBF CONTROL SYSTEM



GET BEST POSSIBLE OCP SOLUTION WITHIN REAL-TIME CONSTRAINTS → GET FEEDBACK → OPT. TRACK WITH GUARANTEED SAFETY

OC + CBF APPROACH (OCBF)

STEP 1: Solve the Optimal Control Problem (OCP)

$$J = \int_{t_0}^{t_f} [\beta + C(\boldsymbol{x}, \boldsymbol{u}, t)]\, dt,$$

subject to:

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}) + g(\boldsymbol{x})\boldsymbol{u}$$

$$\boldsymbol{x}(t_f) = \bar{X}, \quad \text{(or min. } \|\boldsymbol{x}(t_f) - \bar{X}\|^2 \text{)}$$

$$b_j(\boldsymbol{x}(t)) \geq 0, \quad \forall t \in [t_0, t_f]$$

$$\boldsymbol{x}_{\min} \leq \boldsymbol{x}(t) \leq \boldsymbol{x}_{\max}, \forall t \in [t_0, t_f]$$

$$\boldsymbol{u}_{min} \leq \boldsymbol{u} \leq \boldsymbol{u}_{max}$$

…with the least possible amount of simplifications, *e.g., linear dynamics and no constraints*.

$$\boxed{\boldsymbol{u}^*(t) \text{ and } \boldsymbol{x}^*(t),}$$

OC + CBF APPROACH (OCBF)

STEP 2: Observe actual system, allowing for unmodeled disturbances and noise

$$\dot{x} = f(x) + g(x)u + w$$

and define $u_{ref}(t) = h(u^*(t), x^*(t), x(t))$

From OCP

Actual state trajectory

EXAMPLES:

$$u_{ref}(t) = \sum_{j \in \{1,\ldots,n\}} \frac{x_j^*(t)}{x_j(t)} u^*(t)$$

$$u_{ref}(t) = e^{\sum_{j=1}^{n} \frac{x_j^*(t) - x_j(t)}{\sigma_j}} u^*(t)$$

…possibly simply $u_{ref}(t) = u^*(t)$

OC + CBF APPROACH (OCBF)

STEP 3: Optimally track the simplified OCP solution using actual state feedback for improved performance

$$\min_{u_k(t),\delta_k(t)} \int_{t_0}^{t_f} \left(\beta\delta_k^2(t) + \|\boldsymbol{u}(t) - \boldsymbol{u}_{ref}(t)\|^2\right) dt.$$

subject to:

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}) + g(\boldsymbol{x})\boldsymbol{u} + \boldsymbol{w} \qquad \boldsymbol{u}_{min} \leq \boldsymbol{u} \leq \boldsymbol{u}_{max}$$

$$L_f^{m_j} b_j(\boldsymbol{x}) + L_g L_f^{m_j-1} b_j(\boldsymbol{x})\boldsymbol{u} + S(b_j(\boldsymbol{x})) + p_{i,safe}\alpha_{m_j}(\psi_{m_j-1}(\boldsymbol{x})) \geq 0,\ j \in S_o$$

$$L_f^{m_i} b_{i,\max}(\boldsymbol{x}) + L_g L_f^{m_i-1} b_{i,\max}(\boldsymbol{x})\boldsymbol{u} + S(b_{i,\max}(\boldsymbol{x})) + p_{i,\max}\alpha_{m_i}(\psi_{m_i-1}(\boldsymbol{x})) \geq 0$$

$$L_f^{m_i} b_{i,\min}(\boldsymbol{x}) + L_g L_f^{m_i-1} b_{i,\min}(\boldsymbol{x})\boldsymbol{u} + S(b_{i,\min}(\boldsymbol{x})) + p_{i,\min}\alpha_{m_i}(\psi_{m_i-1}(\boldsymbol{x})) \geq 0$$

$$L_f V(y_k(t)) + L_g V(y_k(t))\boldsymbol{u}(t) + \epsilon V(y_k(t)) \leq \delta_k(t)$$

CLF for some states (if desired)
e.g., $y_i(t) := v_i(t) - v_{max}$

HOCBFs corresponding to state and safety constraints: $b_j(\boldsymbol{x}(t)) \geq 0,\ \boldsymbol{x}_{\min} \leq \boldsymbol{x}(t) \leq \boldsymbol{x}_{\max}$

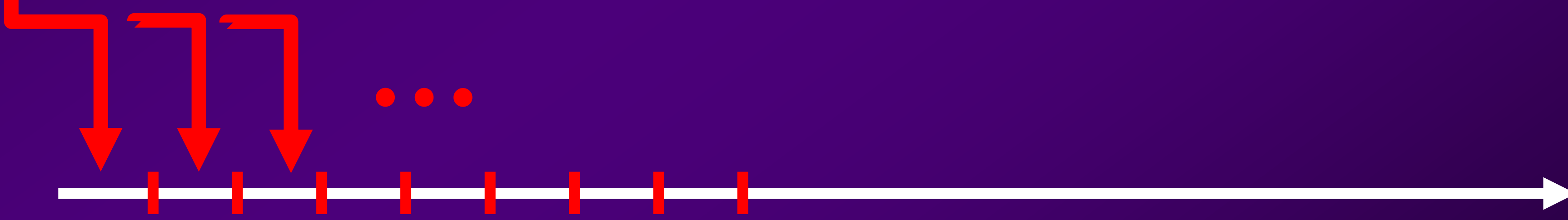
OC + CBF APPROACH (OCBF)

STEP 4: **Solve the optimal tracking problem in real time**

Discretize time and solve a sequence of Quadratic Programs (QPs):

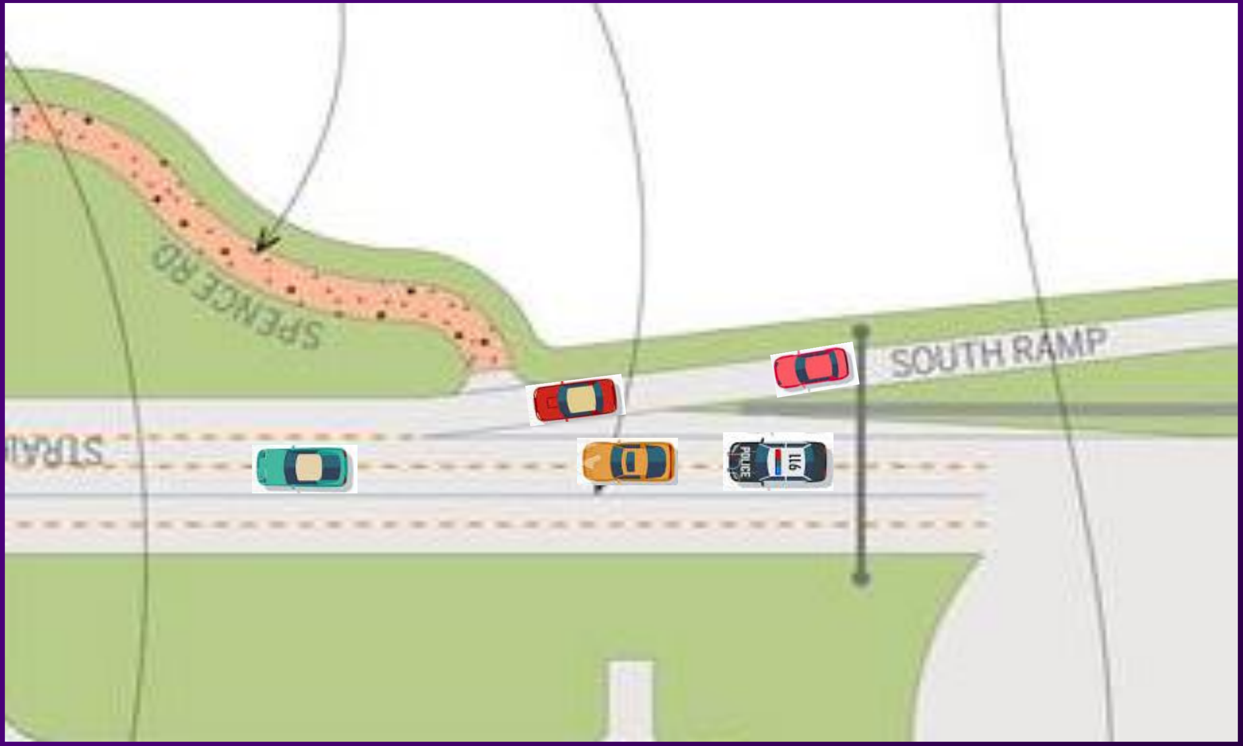
$$(\boldsymbol{u}^*(t_k), \delta^*(t_k)) = \arg \min_{\boldsymbol{u}(t_k), \delta(t_k)} \|\boldsymbol{u}(t_k)\|^2 + p\delta^2(t_k)$$

$$\text{s.t.} \quad \text{Vehicle Dynamics, CBF constraints, CLF constraints}$$

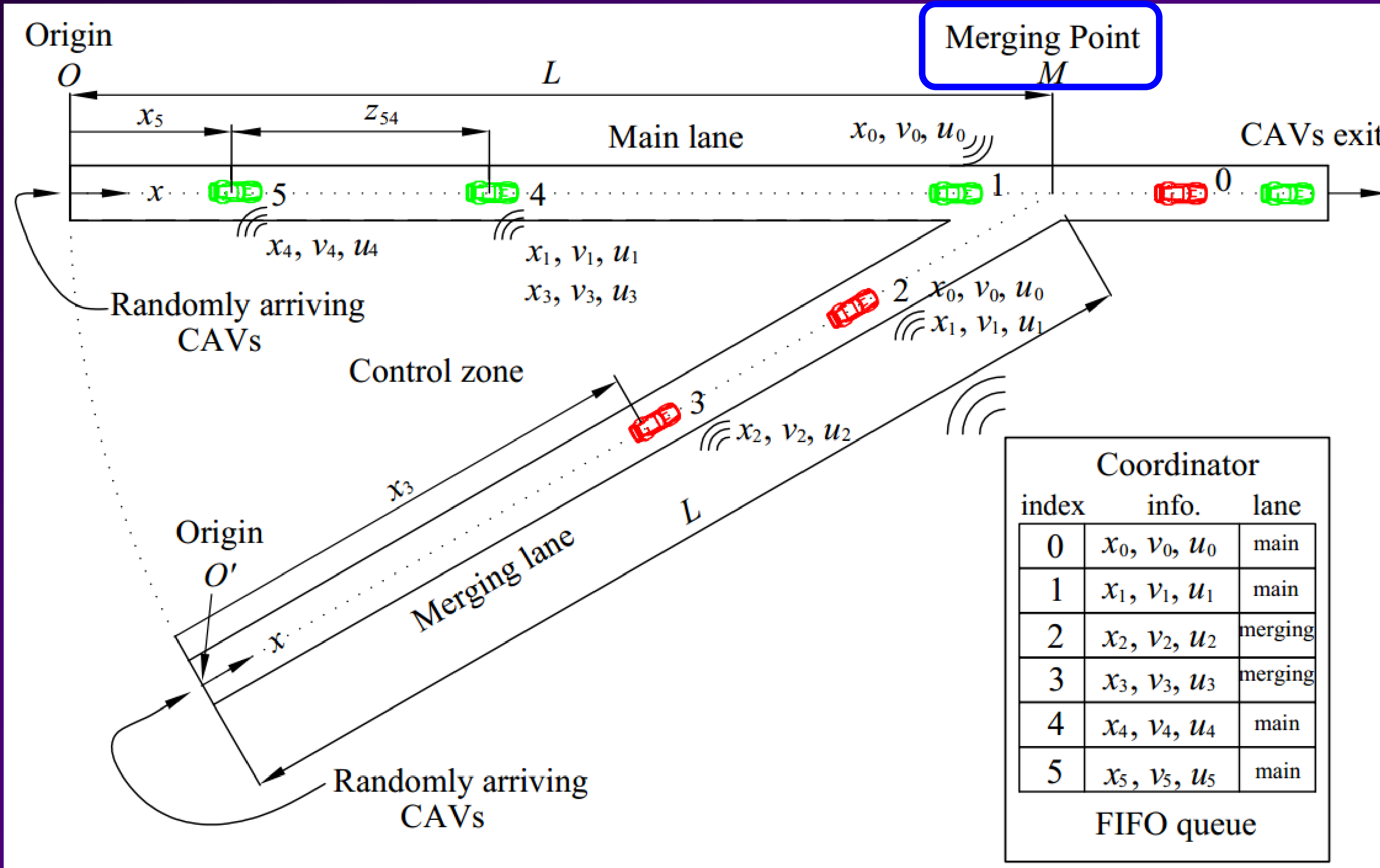


Preferably, using EVENT-DRIVEN control

MERGING AS AN OPTIMAL CONTROL PROBLEM



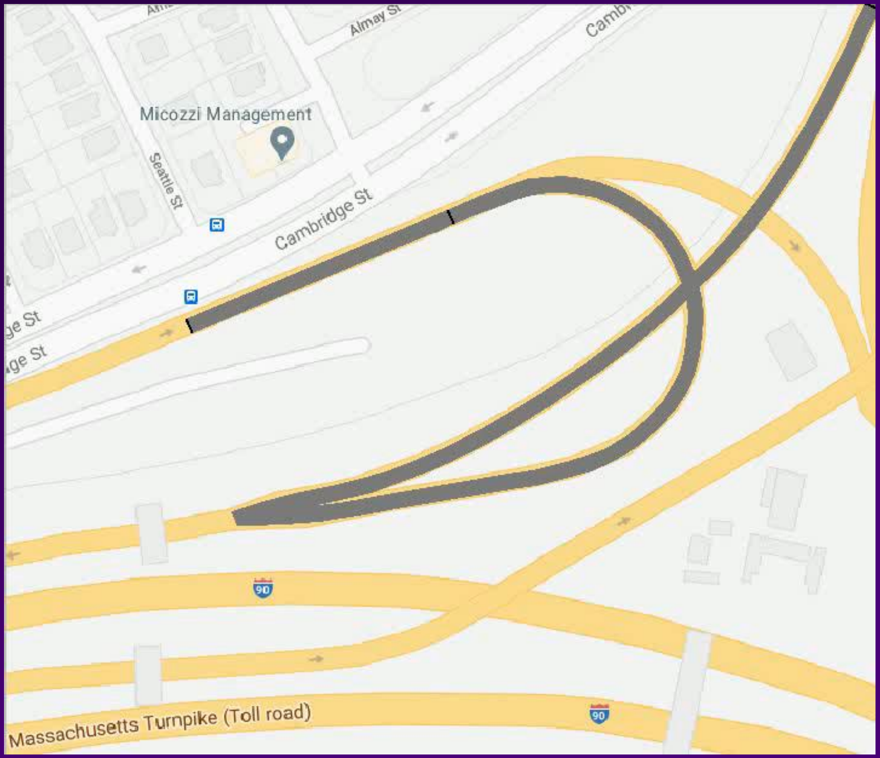
OPTIMAL MERGING

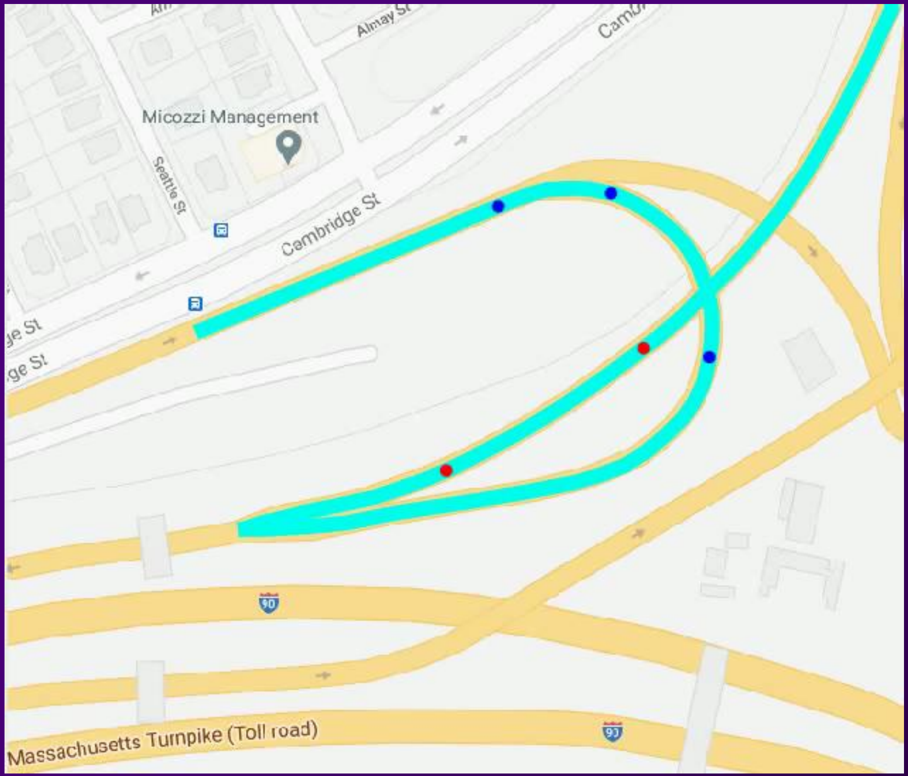


Xiao and Cassandras, *Automatica 2021*

MERGING WITH CURVES

Merging into the Mass Turnpike in Boston, MA



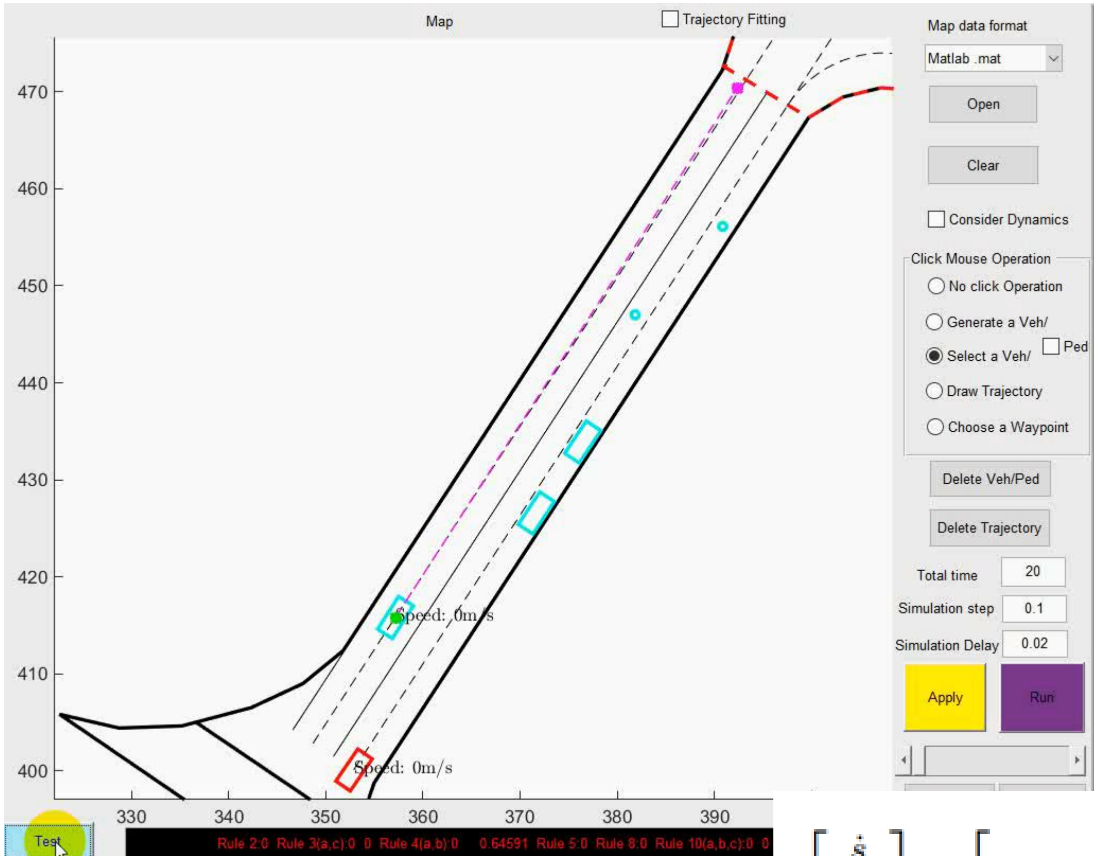


- **Human-driven vehicles**

- **CAVs**

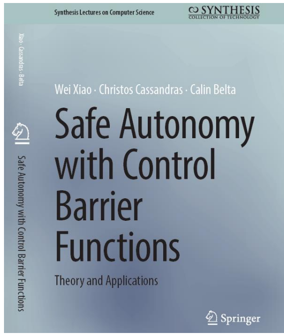
  (with comfort accounted for to accommodate centrifugal forces)

Xiao and Cassandras, *CDC, 2021*



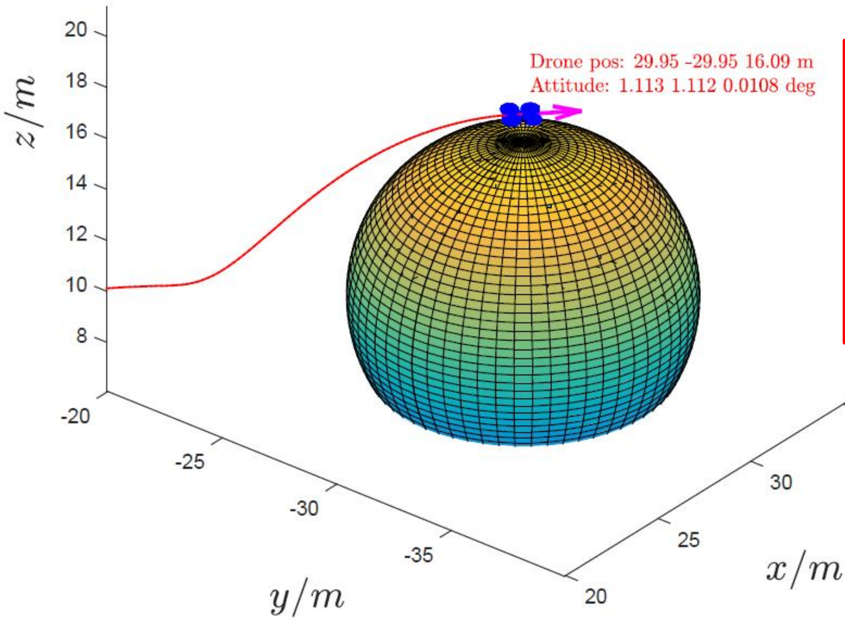
- **Trajectory tracking (mid-lane)**
- **State constraints (steering angle, speed)**
- **Control bounds (jerk, steering accel.)**
- **Obey prioritized list of traffic rules**

Details:
Ch. 10 of …



$$\underbrace{\begin{bmatrix}\dot{s}\\ \dot{n}\\ \dot{\mu}\\ \dot{v}\\ \dot{a}\\ \dot{\delta}\\ \dot{\omega}\end{bmatrix}}_{\dot{x}} = \underbrace{\begin{bmatrix}\frac{v\cos(\mu+\beta)}{1-n\kappa}\\ v\sin(\mu+\beta)\\ \frac{v}{l_r}\sin\beta-\kappa\frac{v\cos(\mu+\beta)}{1-n\kappa}\\ a\\ 0\\ \omega\\ 0\end{bmatrix}}_{f(x)} + \underbrace{\begin{bmatrix}0 & 0\\ 0 & 0\\ 0 & 0\\ 0 & 0\\ 1 & 0\\ 0 & 0\\ 0 & 1\end{bmatrix}}_{g(x)} \underbrace{\begin{bmatrix}u_{jerk}\\ u_{steer}\end{bmatrix}}_{u}$$

QUADROTOR SAFE NAVIGATION



$$\begin{bmatrix} \dot{x} \\ \dot{v}_x \\ \dot{y} \\ \dot{v}_y \\ \dot{z} \\ \dot{v}_z \end{bmatrix} = \begin{bmatrix} v_x \\ 0 \\ v_y \\ 0 \\ v_z \\ -g \end{bmatrix} + \frac{F}{M} \begin{bmatrix} 0 \\ C_\psi S_\theta C_\phi + S_\psi S_\phi \\ 0 \\ S_\psi S_\theta C_\phi - C_\psi S_\phi \\ 0 \\ C_\theta C_\phi \end{bmatrix} - \frac{1}{M} \begin{bmatrix} 0 & 0 & 0 \\ A_x & 0 & 0 \\ 0 & 0 & 0 \\ 0 & A_y & 0 \\ 0 & 0 & 0 \\ 0 & 0 & A_x \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix}$$

16-state quadrotor model

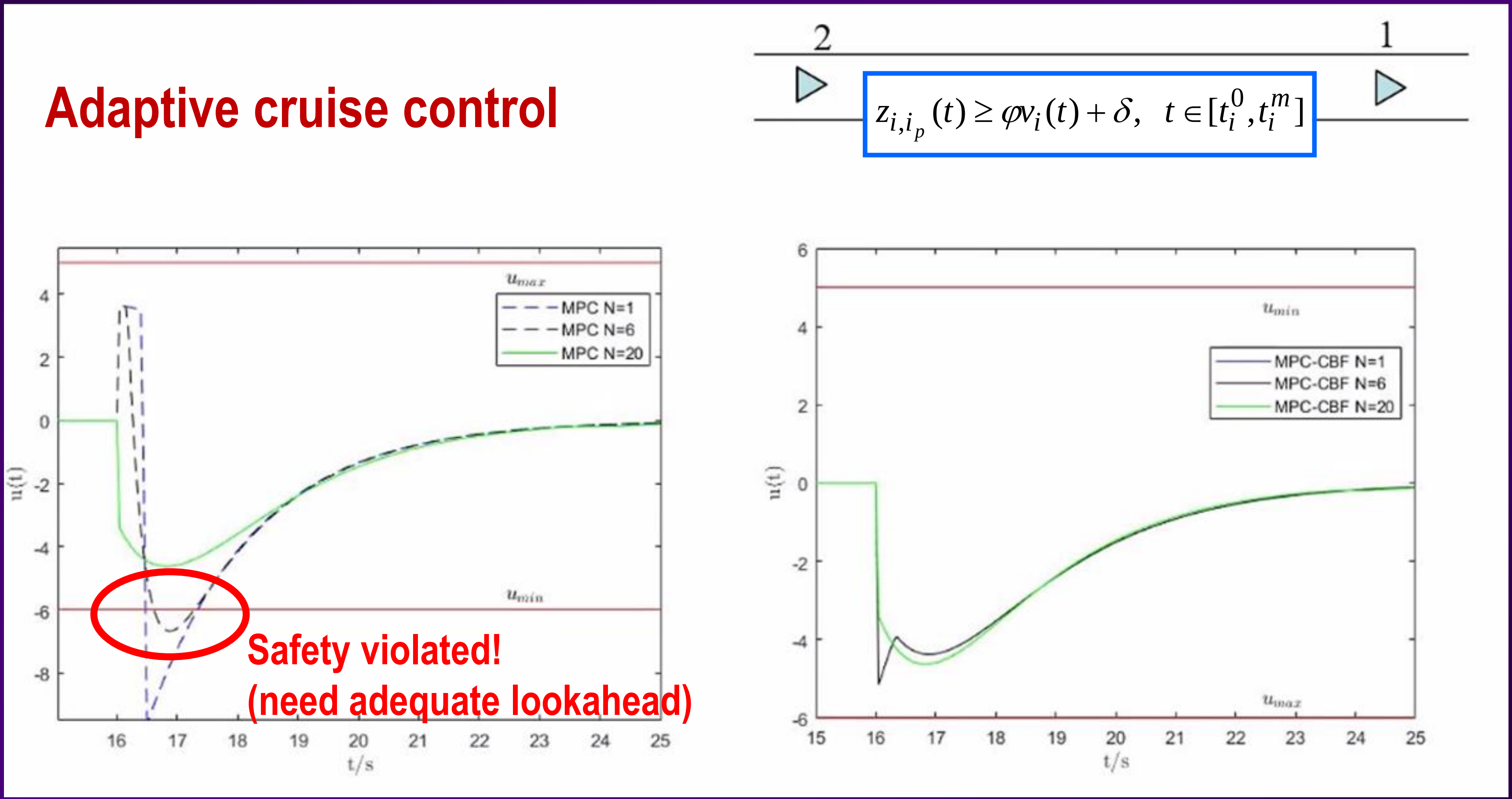
Roll, Pitch, Yaw

$$\begin{bmatrix} \dot{\phi} \\ \dot{\omega}_\phi \\ \dot{\theta} \\ \dot{\omega}_\theta \\ \dot{\psi} \\ \dot{\omega}_\psi \end{bmatrix} = \begin{bmatrix} \omega_\phi & 0 & 0 \\ 0 & \omega_\phi C_\phi T_\theta + \omega_\theta S_\phi / C_\theta^2 & -\omega_\phi S_\phi C_\theta + \omega_\theta C_\phi / C_\theta^2 \\ \omega_\theta & 0 & 0 \\ 0 & -\omega_\phi S_\phi & -\omega_\phi C_\phi \\ \omega_\psi & 0 & 0 \\ 0 & \omega_\phi C_\phi / C_\theta + \omega_\theta S_\phi T_\theta / C_\theta & -\omega_\phi S_\phi / C_\theta + \omega_\theta C_\phi T_\theta / C_\theta \end{bmatrix} \begin{bmatrix} 1 \\ q \\ r \end{bmatrix} + W_\eta^{-1} \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix}$$

Angular speed of each rotor

$$\dot{\omega}_i = b_1 u_i - \beta_0 - \beta_1 \omega_i - \beta_2 \omega_i^2, i \in \{1, 2, 3, 4\}$$

- Make CBF-based methods less myopic, less conservative: MPC + CBFs



Computational cost: Generally, MPC cost > 10 × (CBF-based cost)

THE ROAD AHEAD…

- Resolve all practical issues related to CBFs: conservativeness, feasibility, unknown dynamics, etc
- Using learning methods: Safety guarantees for NNs
- Reduce complexity: Safety Attention Transformations
  - Pay more "attention" to some state variables
  - Replace HOCBFs (can be complex) by CBFs