# Real-Time Attack-Recovery for Cyber-Physical Systems Using Linear Approximations

Lin Zhang[1], Xin Chen[2], Fanxin Kong[1], Alvaro A. Cardenas[3]

[1]Department of Electrical Engineering and Computer Science, Syracuse University, Syracuse NY

[2]Department of Computer Science, University of Dayton, Dayton OH

[3]Department of Computer Science and Engineering, University of California at Santa Cruz, Santa Cruz CA

# Motivation

CPS attacks **cannot** be handled by **classic cyber security mechanisms**

Sensor spoofing attack

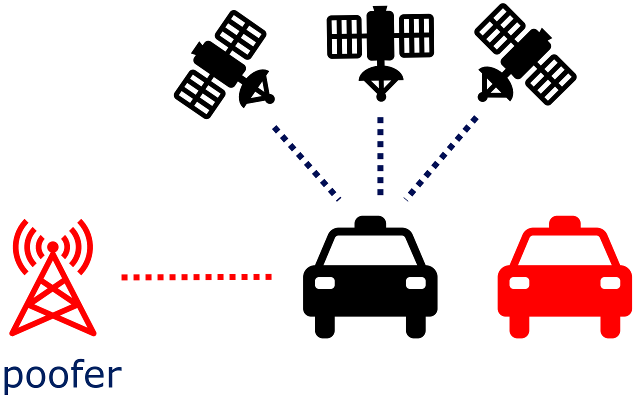
- software attacks

  malicious sensor information

  drive the **physical system** to unsafe state

- transduction attacks

  manipulates a **physical property** that affects sensor reading



# Motivation

Most of the literature focus on attack-detection

- 32 recent CPS security surveys
    - **most of them** talked about **attack-detection**
    - only 8 of them described response to attacks

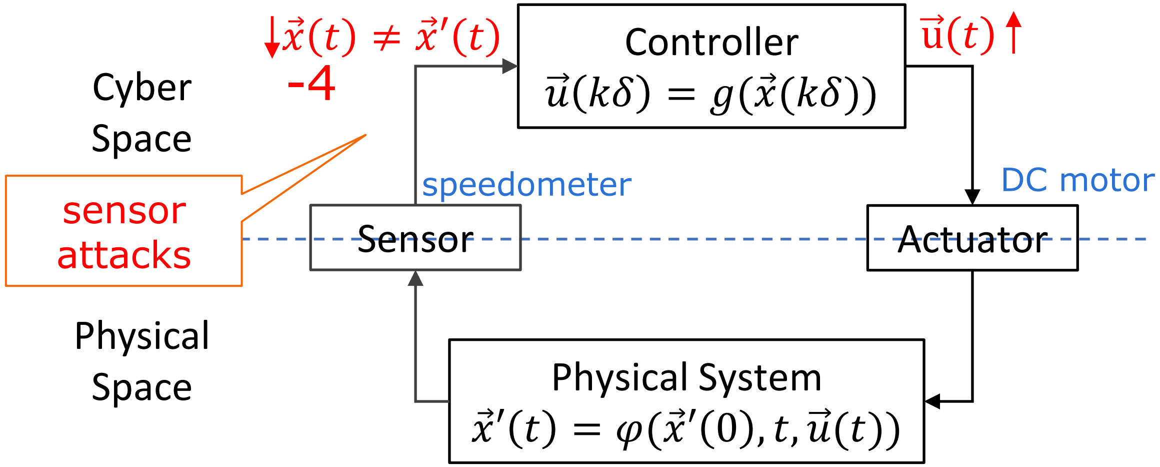
After detecting an attack, what should we do?

This paper focuses on **attack recovery** in a real time manner

# Motivational Example

## Cruise Control

Control stepsize δ: 0.02s



↓$\vec{x}(t) \neq \vec{x}'(t)$
-4
Controller
$\vec{u}(k\delta) = g(\vec{x}(k\delta))$
$\vec{u}(t)$↑
Cyber Space
sensor attacks
speedometer
DC motor
Sensor
Actuator
Physical Space
Physical System
$\vec{x}'(t) = \varphi(\vec{x}'(0), t, \vec{u}(t))$

DC motor speed model

angular velocity

applied voltage

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} -\frac{b}{J} & \frac{K_T}{J} \\ -\frac{K_e}{L} & -\frac{R}{L} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{L} \end{bmatrix} u$$

armature current

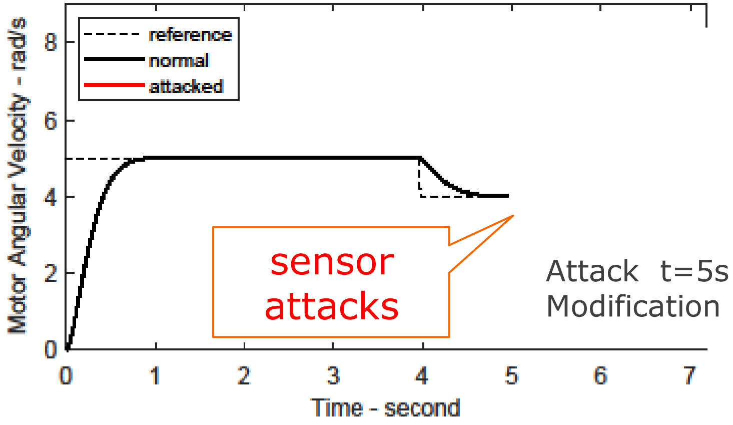
Input: voltage applied to the motor
Output: angular velocity of the motor

**Attack scenarios:**

(1) Modification:
adding/subtracting some values

(2) Replay:
use data from previous time period

(3) Delay:
intentionally delay the data

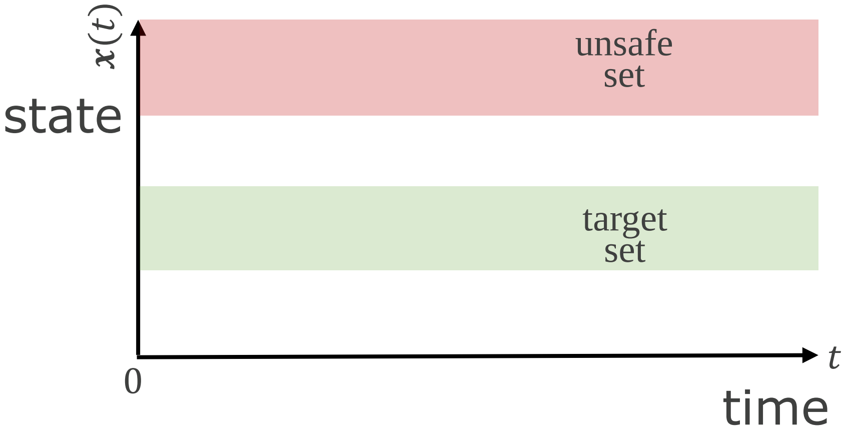


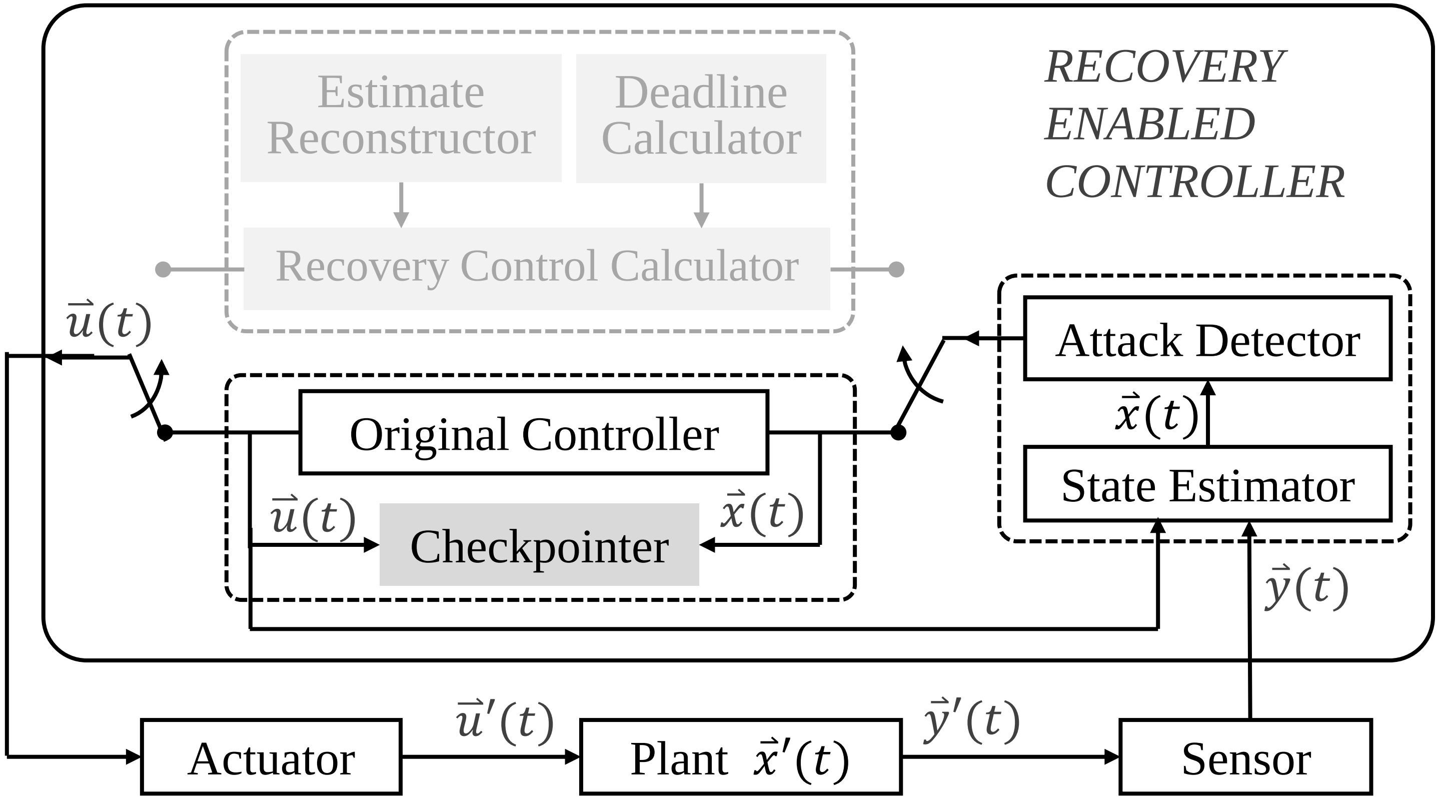
Attack t=5s
Modification attack

# Overview of the Real-Time Recovery Framework

**Unsafe set:** the set of states that define catastrophic events.

**Target set:** the set of desired states. E.g., planned paths, reference values.





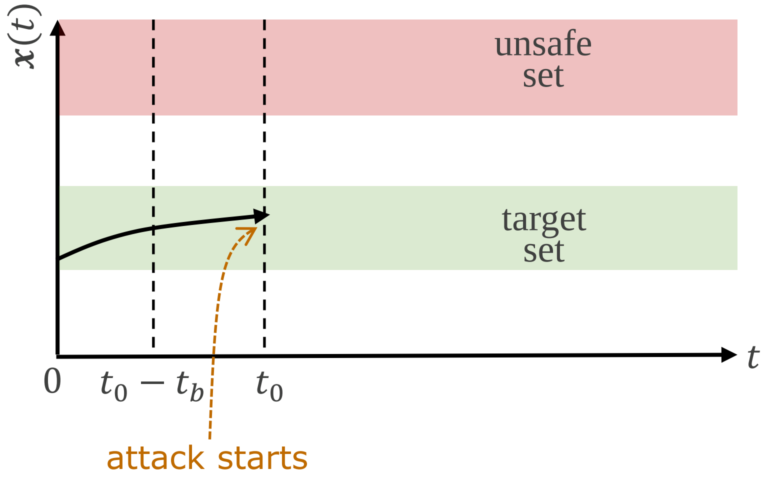
# Overview of the Real-Time Recovery Framework

**original controller**

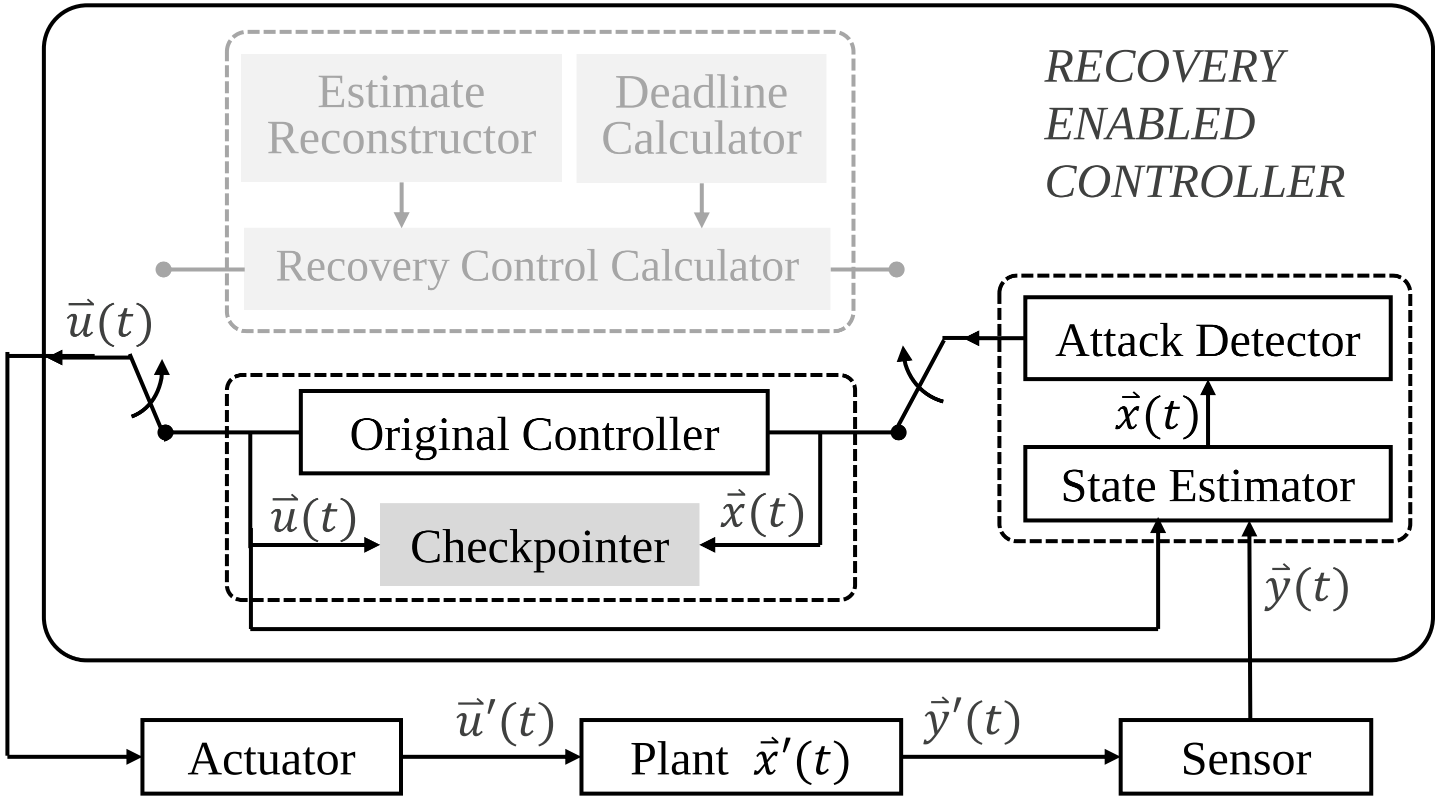
**checkpointer**

- record historical data
  - state estimate $\vec{x}(t)$
  - control input $\vec{u}(t)$

an attack is launched at $t_0$



**Normal** Mode



# Overview of the Real-Time Recovery Framework

attack is detected after at most $\mathrm{t_a}$

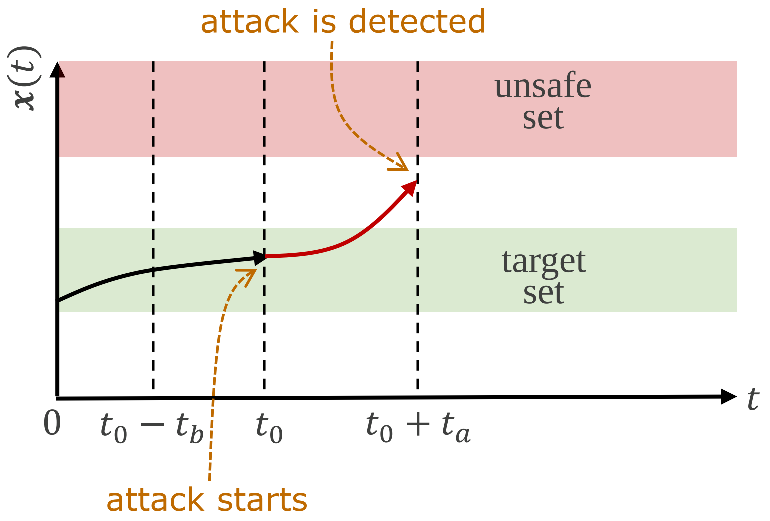
- switch to the recovery mode

**estimate reconstructor**

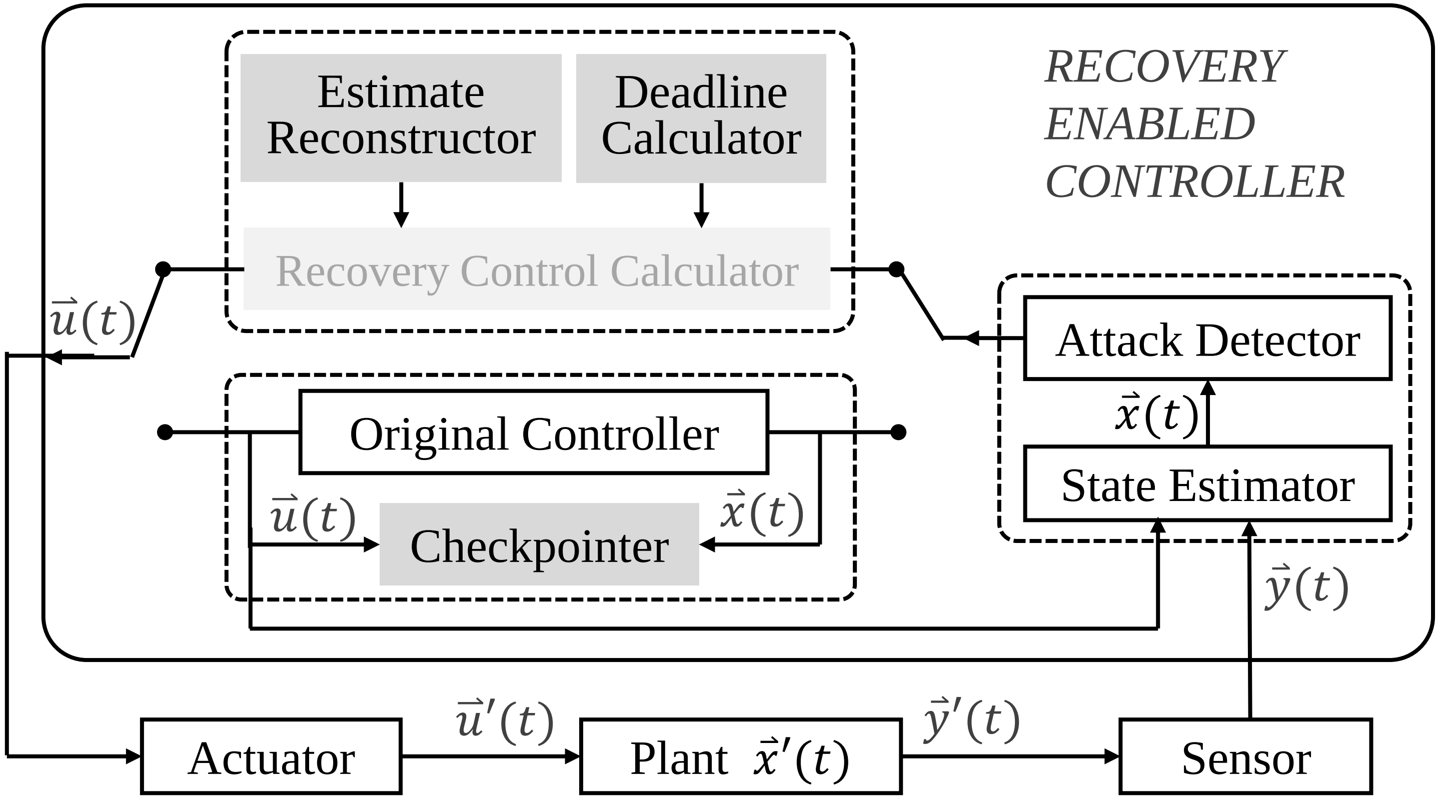
- rebuild state estimate at $t_0 + t_a$

**deadline calculator**

- calculate a safety deadline $t_d$



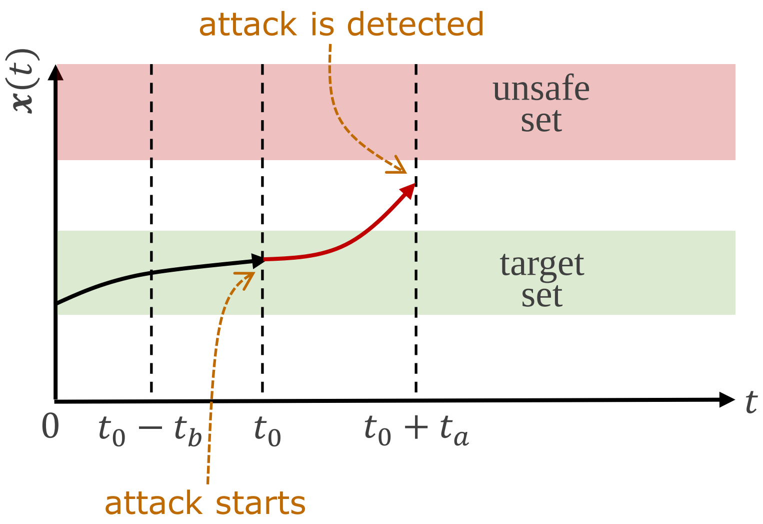
**Recovery** Mode



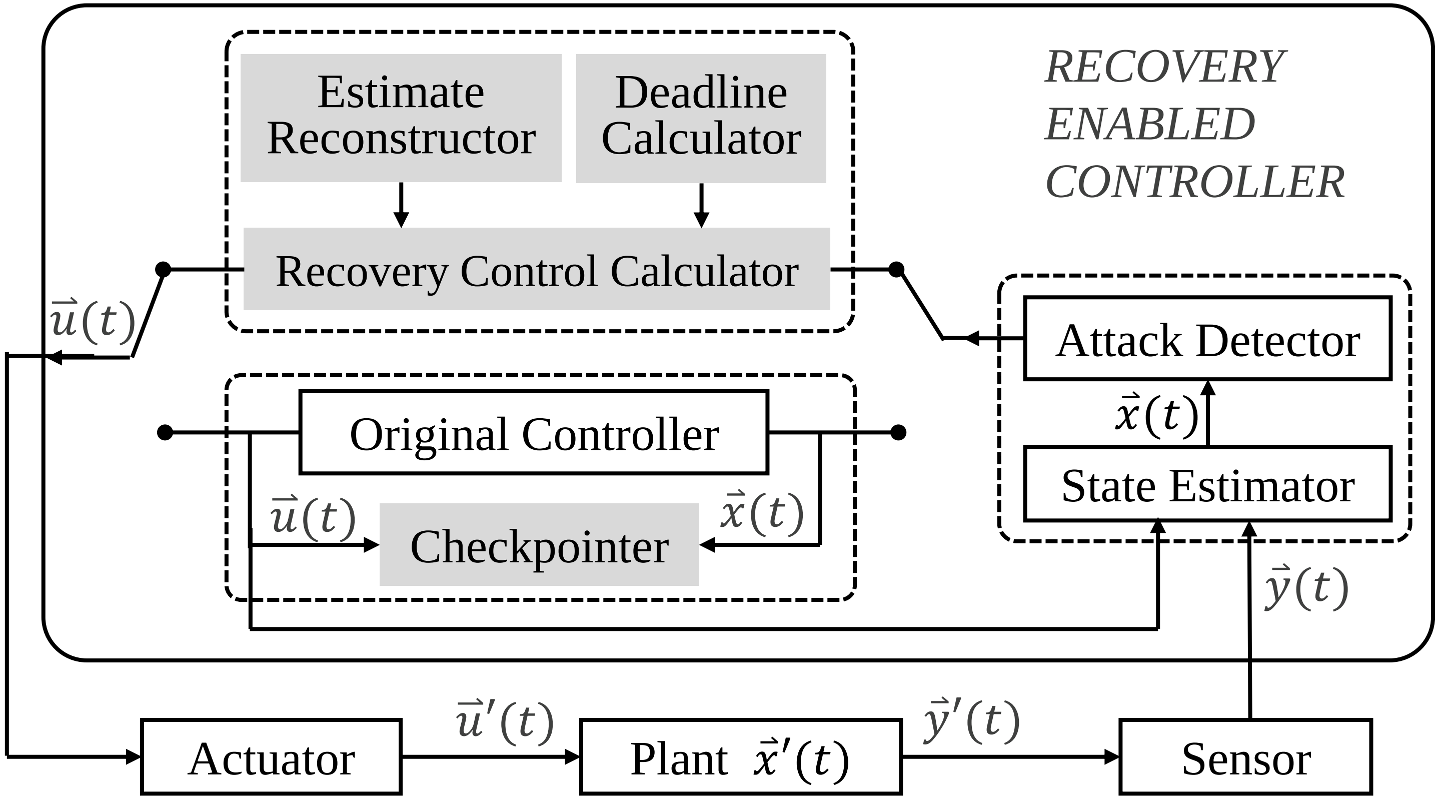
# Overview of the Real-Time Recovery Framework

## recovery control calculator

- compute a Piece-Wise Constant control sequence
    - rebuilt state → target set
    - within safety deadline



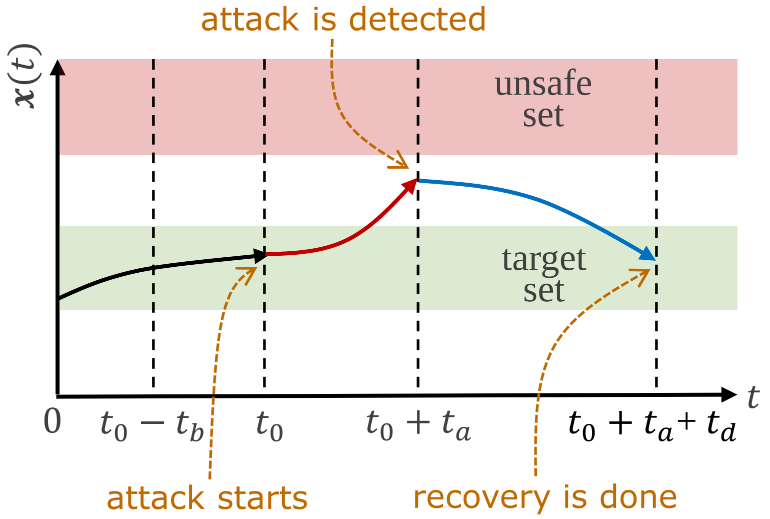
**Recovery** Mode



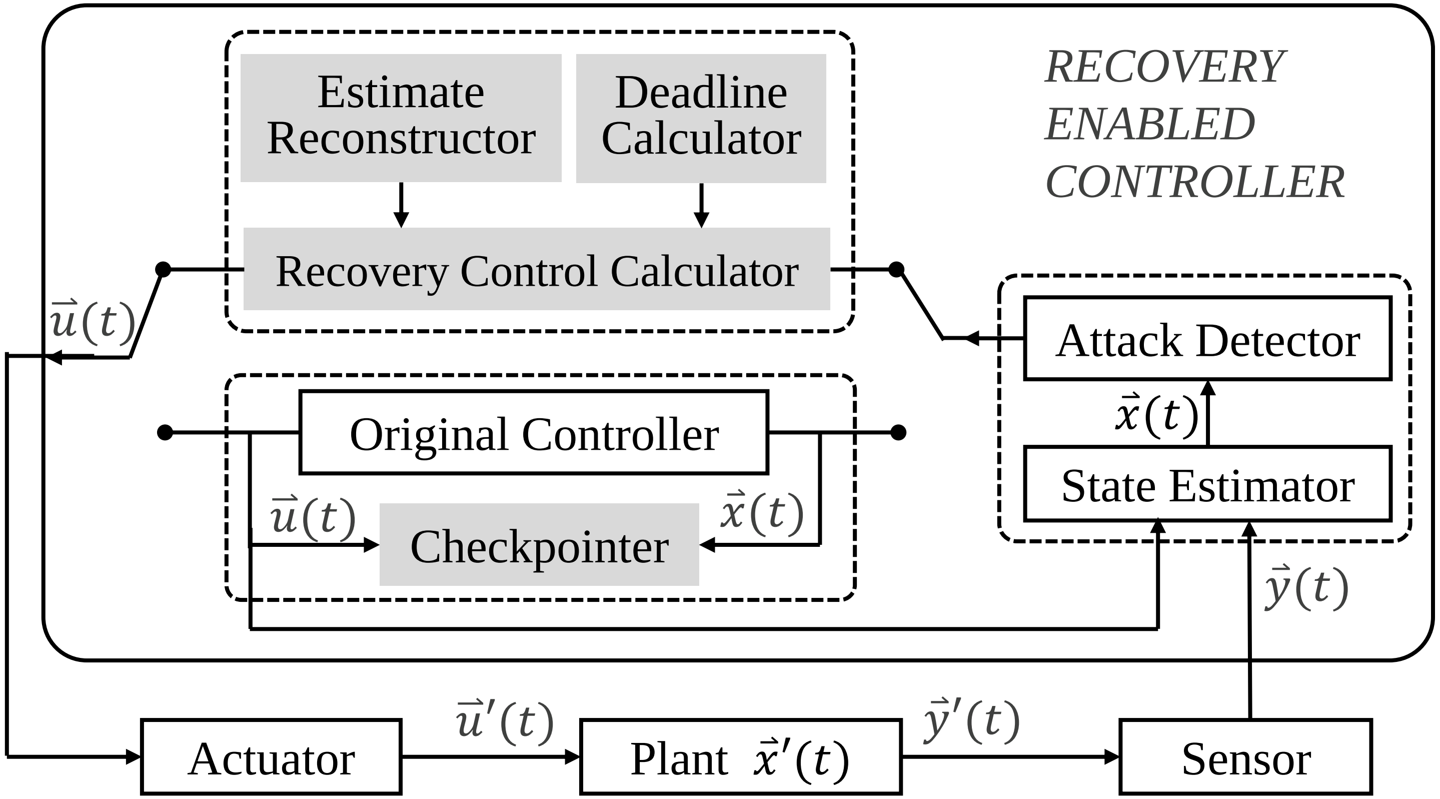
# Overview of the Real-Time Recovery Framework

**recovery controller**

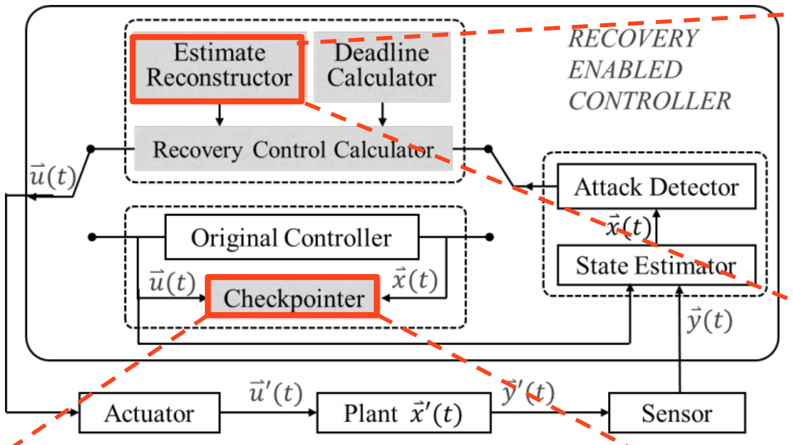
- apply recovery control sequence immediately
- back to target state set before $t_0 + t_a + t_d$



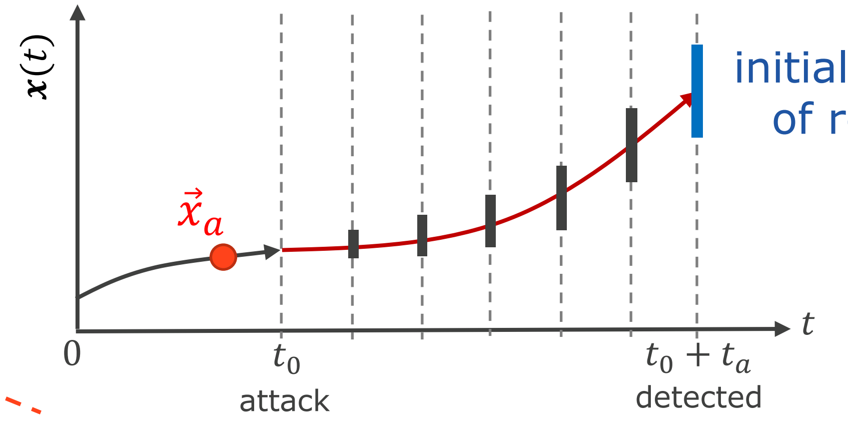
**Recovery** Mode



# Estimate Reconstructor

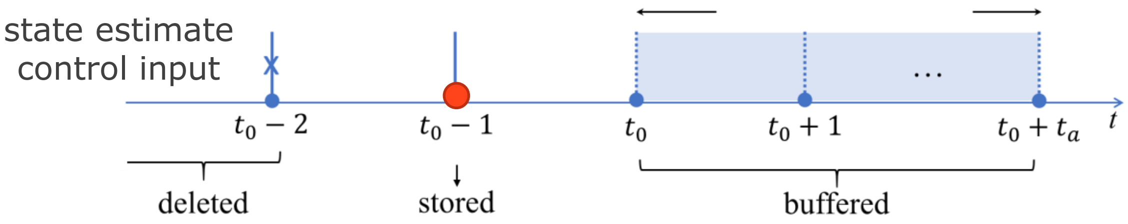


$$\vec{x}_{k+1} = A\,\vec{x}_k + B\,\vec{u}_{k+1}$$



initial state set of recovery

checkpoint protocol



not needed

outside window
last trustful

detection window
may be compromised

plant ε-LTI approximation

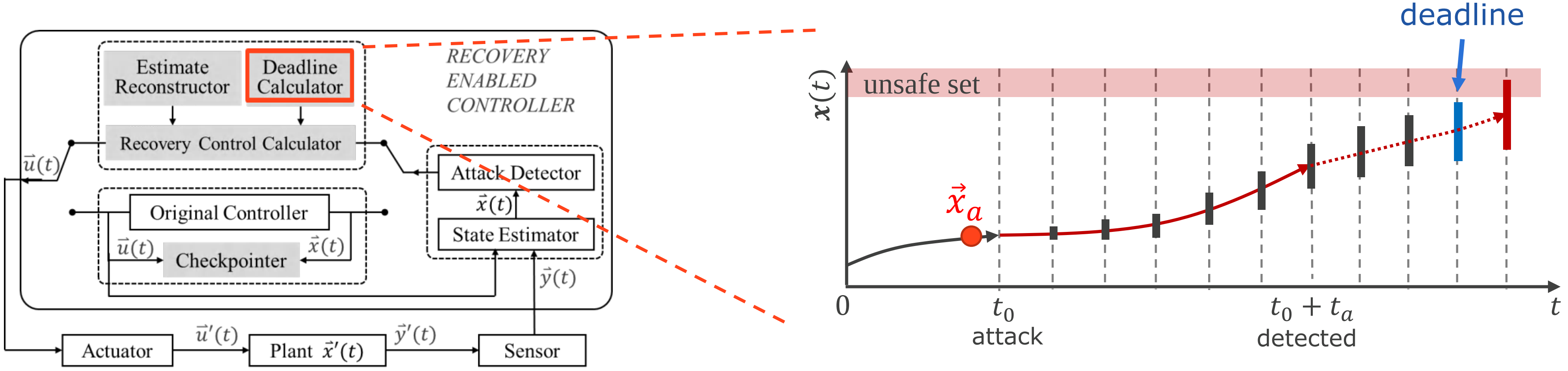
$$\varphi(\vec{s}, \delta, \vec{c}) \in \{A\vec{s} + B\vec{c}\} \oplus \mathcal{B}_\epsilon \quad (2)$$

reachable state set at time $t_0 + t_a$

$$\left\{ A^{N_a}\vec{x}_a + \sum_{j=0}^{N_a-1} A^j B \vec{c}_{N_a-j} \right\} \oplus \bigoplus_{j=0}^{N_a-1} A^j \mathcal{B}_\epsilon \quad (4)$$

box overapproximation – support function method

# Deadline Calculator



Reachability computation:

$$Z_i = \left\{ A^{N_a+i}\vec{x}_a + \sum_{j=i}^{N_a+i-1} A^j B\vec{c}_{N_a+i-j} \right\} \oplus \bigoplus_{j=0}^{N_a+i-1} A^j \mathcal{B}_\epsilon \oplus \underbrace{\left\{ \sum_{j=0}^{i-1} A^j B\vec{c}_{N_a+i-j} \right\}}_{\Phi_i}.$$

where future control input $\vec{c}_{N_a+1}, \ldots, \vec{c}_{N_a+i}$ is pre-assumed

Safety checking based on support function:

$$a_p^T A^{N_a+i}\vec{x}_a + \sum_{j=0}^{N_a+i-1} a_p^T A^j B\vec{c}_{N_a+i-j} + \sum_{j=0}^{N_a+i-1} \sqrt{a_p^T A^j (A^j)^T a_p \epsilon} \overset{?}{>} b.$$

**support function** of a set $S \subseteq \mathbb{R}^n$ according to a vector $\vec{l}$ :

$$\rho_S(\vec{l}) = \sup_{s \in S}\{\vec{l}^T s\}$$

For support function on convex sets

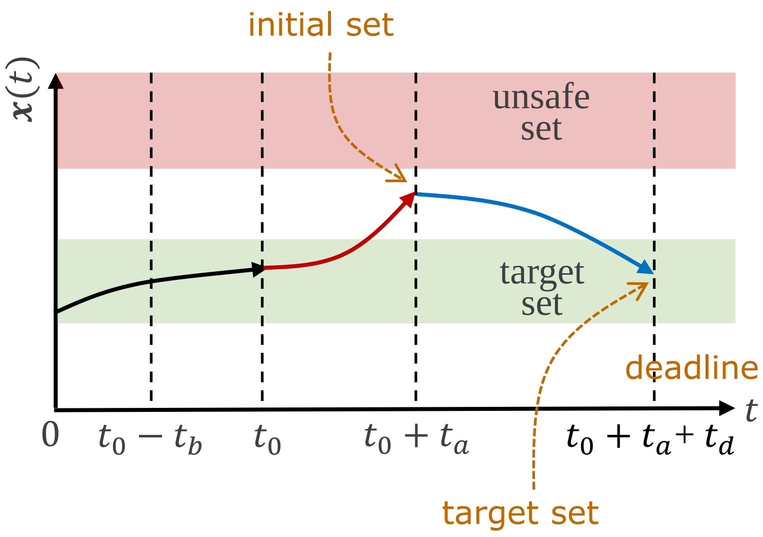
$$\rho_{AS}(\vec{l}) = \rho_S(A^T\vec{l}), \quad \text{for convex set } S$$

$$\rho_{S_1 \oplus S_2}(\vec{l}) = \rho_{S_1}(\vec{l}) + \rho_{S_2}(\vec{l}) \quad \text{for convex sets } S_1, S_2$$

For $\Omega_j = A^j \mathcal{B}_\epsilon$ , we have

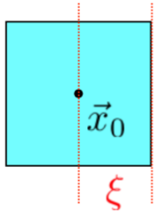
$$\rho_{\Omega_j}(\vec{l}) = \sqrt{\vec{l}^T A^j (A^j)^T \vec{l}}\epsilon$$

# Real-time Recovery using PWC Control



**the recovery problem** asks whether there exists a **recovery control sequence** $\vec{\boldsymbol{u}}_1, \dots, \vec{\boldsymbol{u}}_N$ where $N \leq \boldsymbol{D}$ steering the system from **initial state** $\vec{\boldsymbol{x}}_0$ to a state in **target set** $\boldsymbol{X_T}$ while all reachable states on the way are in a **safe set** $\boldsymbol{X_S}$.

**Linear Programming Problem**



$X_0 = \{\vec{x}_0\} \oplus \mathcal{I}_\xi$

states in target set is maintainable

$\forall \vec{s} \in X_T$ is maintainable, iff.

$(I - A)\vec{s} = B\vec{c}$, i.e., $\vec{s} = A\vec{s} + B\vec{c}$

single initial set

extension to an initial set

Find $\vec{u}_1, \dots, \vec{u}_D \in \mathcal{U}$

s.t. $\vec{x}_D \in X_T \longrightarrow \vec{x}_D \in (X_T \ominus A^D \mathcal{I}_\xi)$

$\bigwedge_{i=0}^{D} (\vec{x}_i \in X_s) \longrightarrow \bigwedge_{i=0}^{D} \vec{x}_i \in (X_S \ominus A^i \mathcal{I}_\xi)$

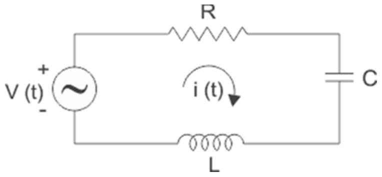
$\bigwedge_{i=0}^{D-1} (\vec{x}_{i+1} = A\vec{x}_i + B\vec{u}_{i+1})$

# Evaluation - Benchmarks

## 1. Vehicle Turning

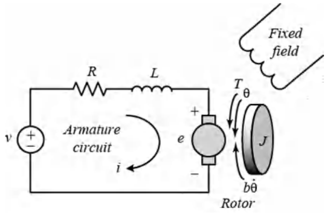
$$\dot{x} = -\frac{25}{3}x + 5u$$

## 2. Series RLC Circuit



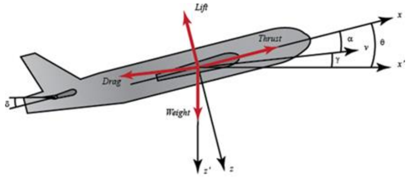
$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & \frac{1}{C} \\ -\frac{1}{L} & -\frac{R}{L} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{L} \end{bmatrix} u$$

## 3. DC Motor Position



$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & -\frac{b}{J} & \frac{K}{J} \\ 0 & -\frac{K}{L} & -\frac{R}{L} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \frac{1}{L} \end{bmatrix} u$$

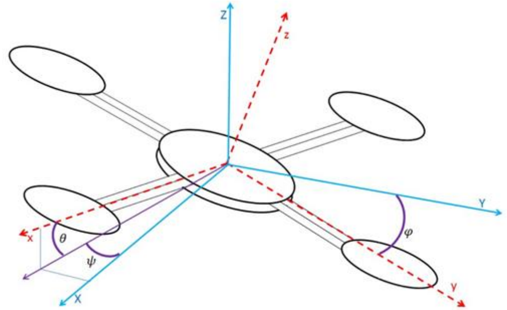
## 4. Aircraft Pitch



$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} -0.313 & 56.7 & 0 \\ -0.0139 & -0.426 & 0 \\ 0 & 56.7 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} 0.232 \\ 0.0203 \\ 0 \end{bmatrix} u$$

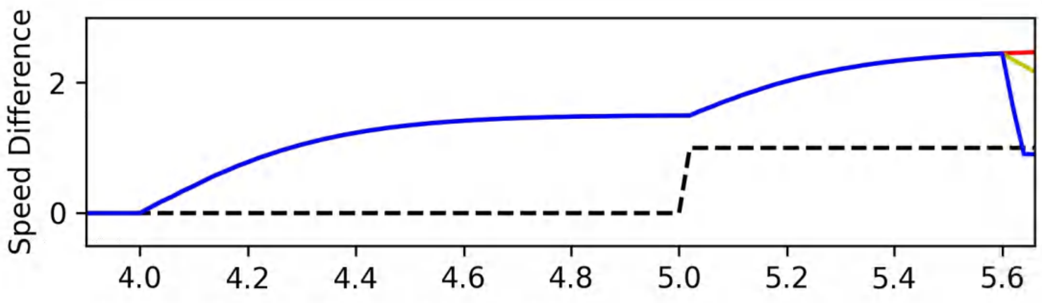
## 5. Quadrotor

$$\begin{cases} \dot{\phi} = p \\ \dot{\theta} = q \\ \dot{\psi} = r \\ \dot{p} = \frac{\tau_x + \tau_{wx}}{I_x} \\ \dot{q} = \frac{\tau_y + \tau_{wy}}{I_y} \\ \dot{r} = \frac{\tau_z + \tau_{wz}}{I_z} \\ \dot{u} = -g\theta + \frac{fwx}{m} \\ \dot{v} = g\phi + \frac{fwy}{m} \\ \dot{w} = \frac{f_{wz} - f_t}{m} \\ \dot{x} = u \\ \dot{y} = v \\ \dot{z} = w \end{cases}$$

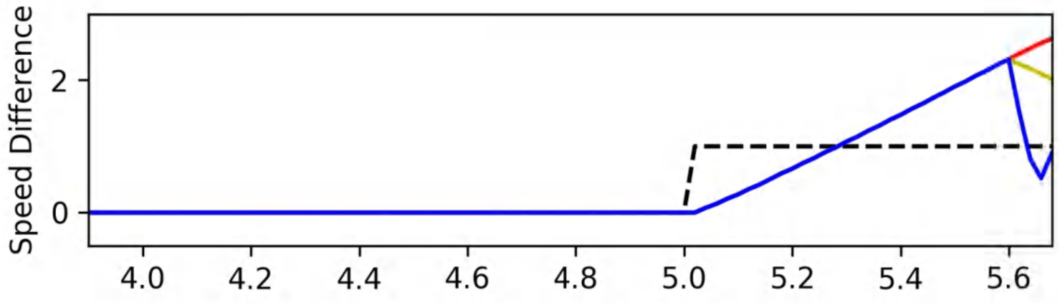


Reference: https://ctms.engin.umich.edu/CTMS/index.php

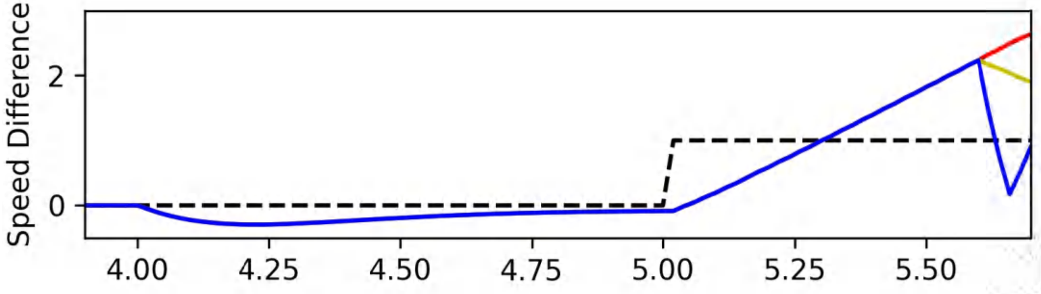
# Evaluation – Results for Vehicle Turning



(a) Vehicle turning & modification attack



(b) Vehicle turning & delay attack



(c) Vehicle turning & replay attack

**our method can do real-time recovery**

## Legend:

Dotted Black: Reference state

Red: No recovery

Yellow: Non-real-time recovery

Blue: Real-time recovery

# Evaluation – Other Results

Modification

Delay

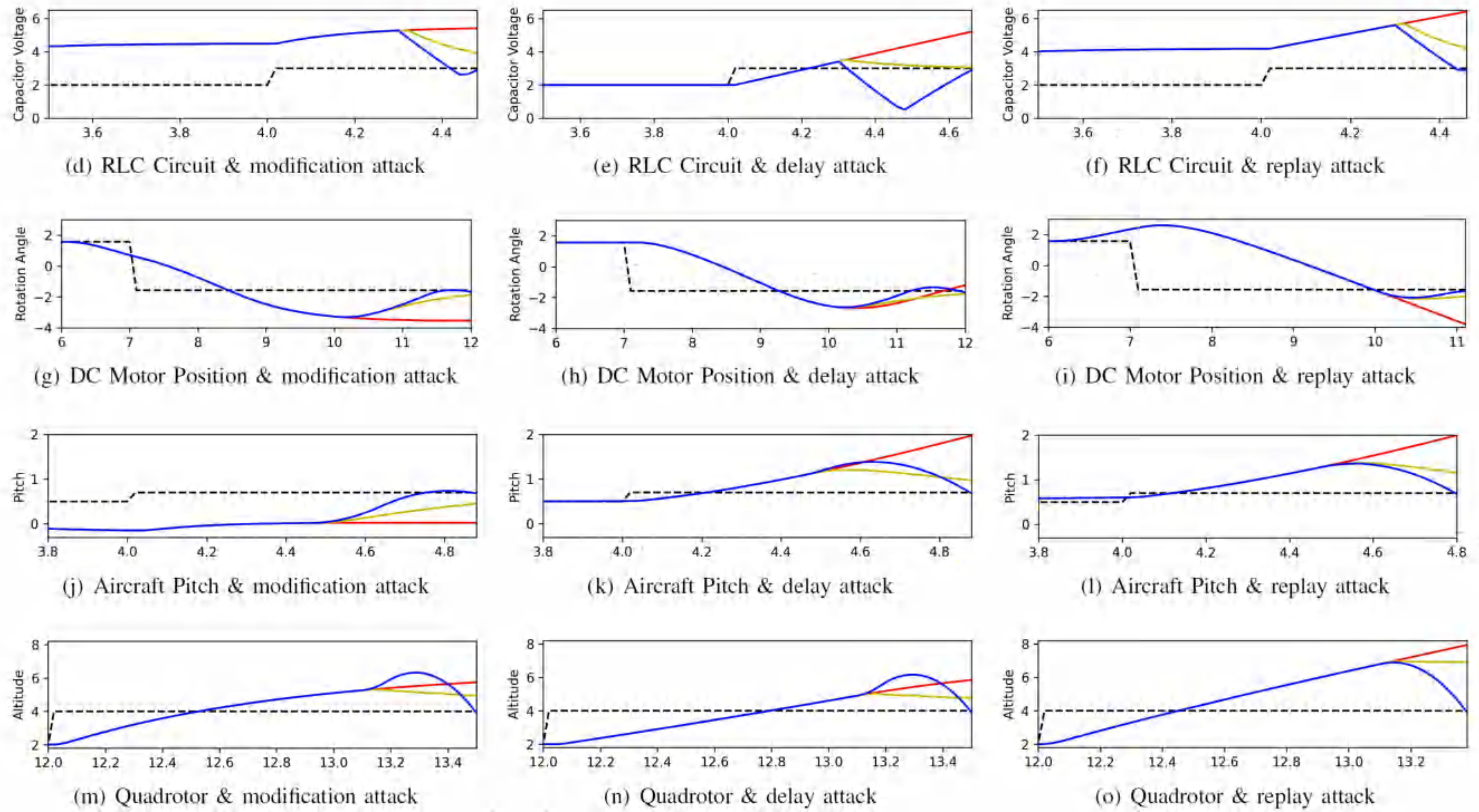
Replay

RLC Circuit

DC Motor Position

Aircraft Pitch

Quadrotor



(d) RLC Circuit & modification attack

(e) RLC Circuit & delay attack

(f) RLC Circuit & replay attack

(g) DC Motor Position & modification attack

(h) DC Motor Position & delay attack

(i) DC Motor Position & replay attack

(j) Aircraft Pitch & modification attack

(k) Aircraft Pitch & delay attack

(l) Aircraft Pitch & replay attack

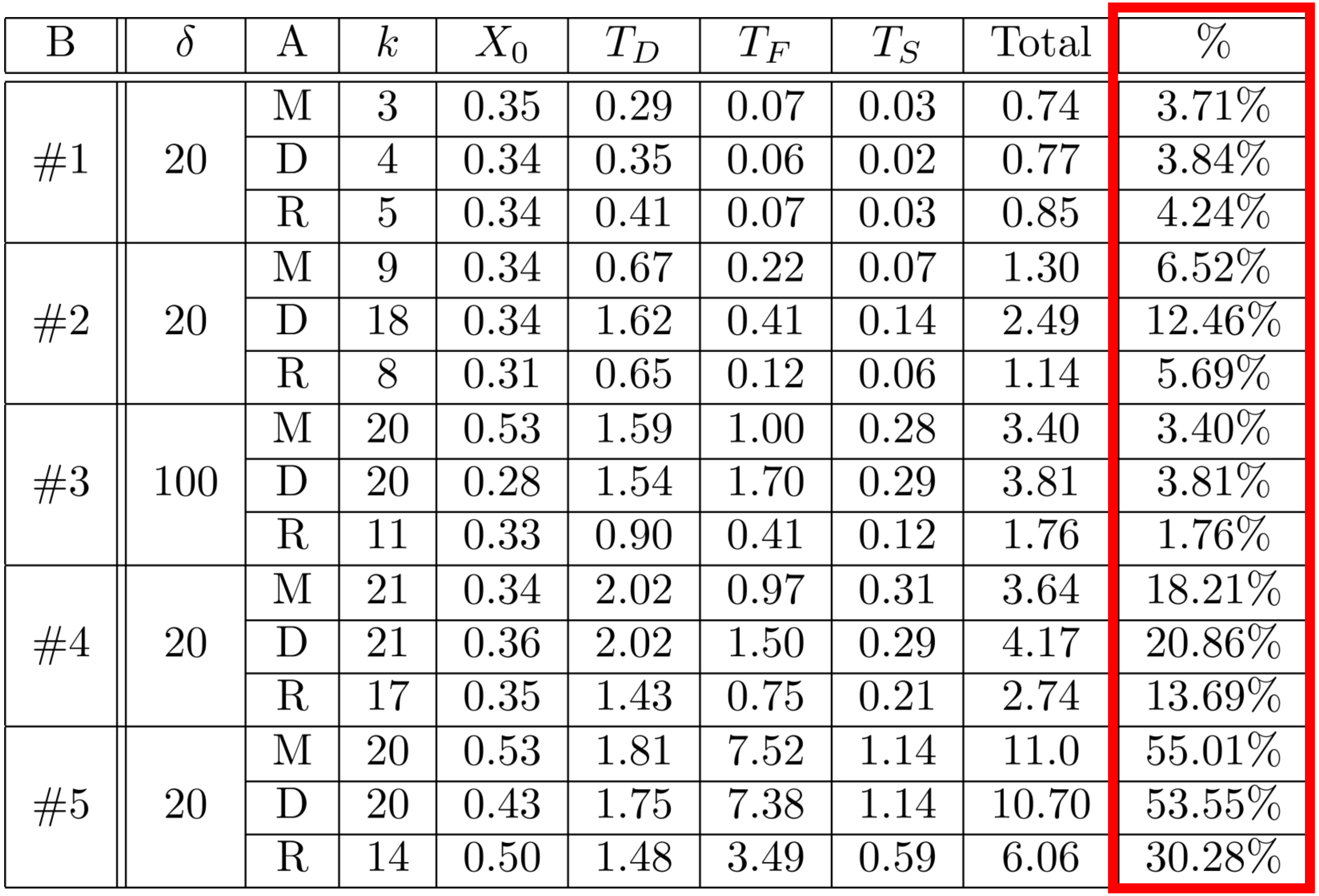
(m) Quadrotor & modification attack

(n) Quadrotor & delay attack

(o) Quadrotor & replay attack

# Evaluation – Time Cost

**overhead is small**



| B | $\delta$ | A | $k$ | $X_0$ | $T_D$ | $T_F$ | $T_S$ | Total | % |
|---|---|---|---|---|---|---|---|---|---|
| #1 | 20 | M | 3 | 0.35 | 0.29 | 0.07 | 0.03 | 0.74 | 3.71% |
| | | D | 4 | 0.34 | 0.35 | 0.06 | 0.02 | 0.77 | 3.84% |
| | | R | 5 | 0.34 | 0.41 | 0.07 | 0.03 | 0.85 | 4.24% |
| #2 | 20 | M | 9 | 0.34 | 0.67 | 0.22 | 0.07 | 1.30 | 6.52% |
| | | D | 18 | 0.34 | 1.62 | 0.41 | 0.14 | 2.49 | 12.46% |
| | | R | 8 | 0.31 | 0.65 | 0.12 | 0.06 | 1.14 | 5.69% |
| #3 | 100 | M | 20 | 0.53 | 1.59 | 1.00 | 0.28 | 3.40 | 3.40% |
| | | D | 20 | 0.28 | 1.54 | 1.70 | 0.29 | 3.81 | 3.81% |
| | | R | 11 | 0.33 | 0.90 | 0.41 | 0.12 | 1.76 | 1.76% |
| #4 | 20 | M | 21 | 0.34 | 2.02 | 0.97 | 0.31 | 3.64 | 18.21% |
| | | D | 21 | 0.36 | 2.02 | 1.50 | 0.29 | 4.17 | 20.86% |
| | | R | 17 | 0.35 | 1.43 | 0.75 | 0.21 | 2.74 | 13.69% |
| #5 | 20 | M | 20 | 0.53 | 1.81 | 7.52 | 1.14 | 11.0 | 55.01% |
| | | D | 20 | 0.43 | 1.75 | 7.38 | 1.14 | 10.70 | 53.55% |
| | | R | 14 | 0.50 | 1.48 | 3.49 | 0.59 | 6.06 | 30.28% |

# Evaluation – Scalability Analysis

### Scalable Heating Model

heating in a point of a rod located at **1/3** of the length

recording the temperature at **2/3** of the length

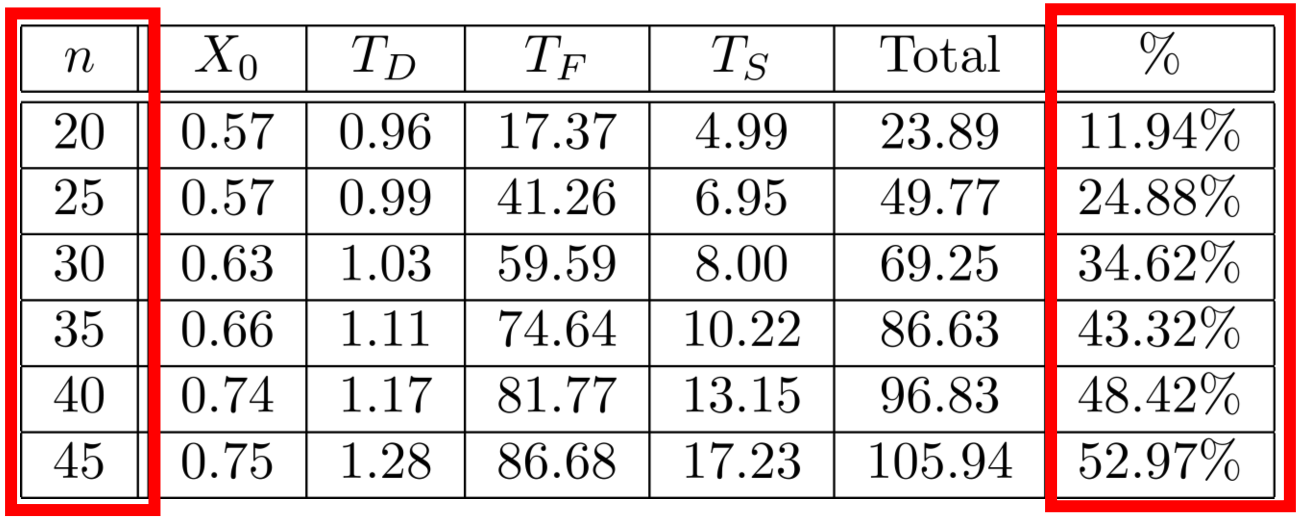
# of variables is scalable   n= 25,30,35,40,45

**overhead increase with # of variables**

The temperatures of the selected points on the rod is described by

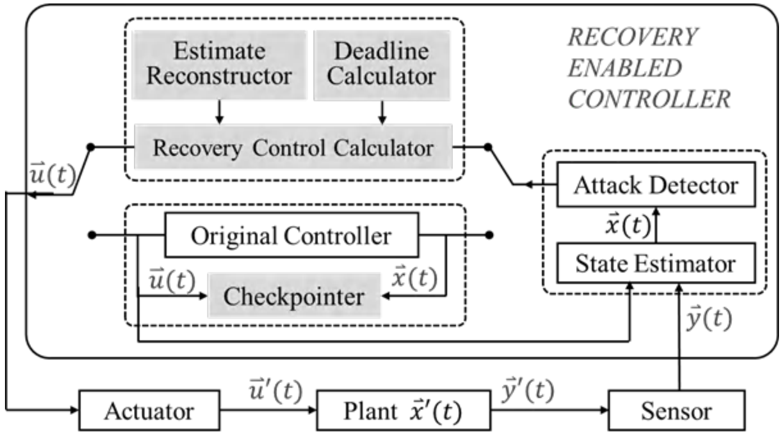
$$\dot{\vec{x}} = A\vec{x} + Bu \qquad \text{such that}$$

$$A = \frac{\alpha}{h^2}\begin{pmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -1 & 2 \end{pmatrix}$$



| $n$ | $X_0$ | $T_D$ | $T_F$ | $T_S$ | Total | % |
|---|---|---|---|---|---|---|
| 20 | 0.57 | 0.96 | 17.37 | 4.99 | 23.89 | 11.94% |
| 25 | 0.57 | 0.99 | 41.26 | 6.95 | 49.77 | 24.88% |
| 30 | 0.63 | 1.03 | 59.59 | 8.00 | 69.25 | 34.62% |
| 35 | 0.66 | 1.11 | 74.64 | 10.22 | 86.63 | 43.32% |
| 40 | 0.74 | 1.17 | 81.77 | 13.15 | 96.83 | 48.42% |
| 45 | 0.75 | 1.28 | 86.68 | 17.23 | 105.94 | 52.97% |

# Summary



- A new attack-recovery architecture
  - estimate reconstructor
  - deadline calculator
  - recovery control calculator
- A formal method to conservatively **estimate** the current and future states with a control **stepwise error bound** $\varepsilon > 0$ based on a Linear Time-Invariant (**LTI**) approximate
- Formulate the **reach-avoid problem** as a **Linear Programming (LP)** restriction with safety and target specifications
- Formal analysis + Simulation + Scalability analysis

Syracuse University

# Real-Time Attack-Recovery

for Cyber-Physical Systems Using Linear Approximations

Thank you.

Q&A