Threshold Strategy for a Leaking Corner-Free Hamilton-Jacobi Reachability with Decomposed Computations

Chong He, Mugilan Mariappan, Keval Vora, and Mo Chen

SFU

CIFAR

BACKGROUND: HAMILTON-JACOBI REACHABILITY

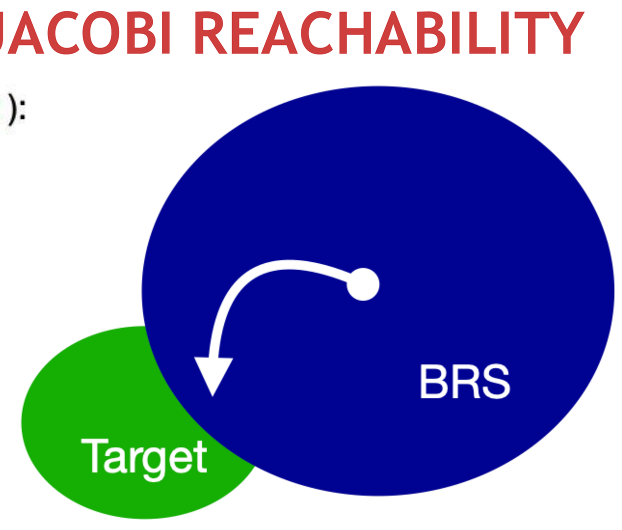
- Implicit Surface Function (0 sublevel set is the target set):

$$z \in \mathcal{T} \leftrightarrow \ell(z) \le 0$$

- System Dynamics: $\dot{z} = f(z) + g(z) \cdot u$
- Trajectory: $\ell(\zeta(0; z, t, u(\cdot))$
- Value Functions:

Liveness problem: $V(z,t) = \min_{u(\cdot)\in\mathbb{U}} \ell(\zeta(0; z, t, u(\cdot)))$

Safety problem: $V(z,t) = \max_{u(\cdot)\in\mathbb{U}} \ell(\zeta(0; z, t, u(\cdot)))$



- Backward computation with HJ-PDE (Grid-based Dynamic Programming):

Liveness problem: $V(z, t-\delta t) = V(z,t) + \min_{\dot{z}} D_z V(z,t)\dot{z}\delta t,$

Safety problem: $V(z, t-\delta t) = V(z,t) + \max_{\dot{z}} D_z V(z,t)\dot{z}\delta t, \quad V(z,0) = l(z)$

- Backward Reachable Set (BRS):

$$z \in \mathcal{R}(t) \leftrightarrow V(z,t) \le 0$$

- Computationally expensive due to the curse of dimensionality

EXAMPLE METHOD SUFFERS FROM THE LEAKING CORNER ISSUE

Self-contained Subsystem Decomposition
- Computation happens in low dimensions
- Solution to the Curse of Dimensionality

- Full-dimensional system: 2D Single Integrator

$$\dot{z} = \begin{bmatrix}\dot{x}\\ \dot{y}\end{bmatrix} = \begin{bmatrix}u_x\\ u_y\end{bmatrix}$$

- Control Input:

$u = (u_x, u_y);$ constrained by $c(u) = \|u\|_2 - \bar{u} \le 0$

- Value Function: $V(z,t)$

- Subsystem 1:

$$\dot{x}_1 = \dot{x} = u_x$$

- Control Input:

$w_x = u_x;$ constrained by $c_1(w_x) = \|u_x\|_2 - \bar{u} \le 0$

- Value Function: $V_1(z,t) = \phi_1(x_1, t)$

- Subsystem 2:

$$\dot{x}_2 = \dot{y} = u_y$$

- Control Input:

$w_y = u_y;$ constrained by $c_2(w_y) = \|u_y\|_2 - \bar{u} \le 0$

- Value Function: $V_2(z,t) = \phi_2(x_2, t)$

PROBLEM: LEAKING CORNER ISSUE

Definition: (Leaking Corners) Suppose we obtain $V(z,t)$ by solving HJ-PDE, and $\hat{V}(z,t)$ by combining value functions. The set of "leaking corners" $\mathcal{L}(t)$ is defined as

$$\mathcal{L}(t) = \{z : V(z,t) \neq \hat{V}(z,t)\}$$

2 Cases will suffer from the issue:

(1) Intersection case for liveness problem: $\hat{V}_R(z,t) = \max\{V_{R,1}(x_1,t), V_{R,2}(x_2,t)\}$

(2) Union case for safety problem: $\hat{V}_A(z,t) = \min\{V_{A,1}(x_1,t), V_{A,2}(x_2,t)\}$

METHOD: THRESHOLD STRATEGY

Theorem 1: We can find the set of leaking corners $\mathcal{L}(t)$ by comparing the (full-dimensional) sub-value functions.

$$\mathcal{L}(t) = \{z : |V_1 - V_2| < \Delta\}.$$

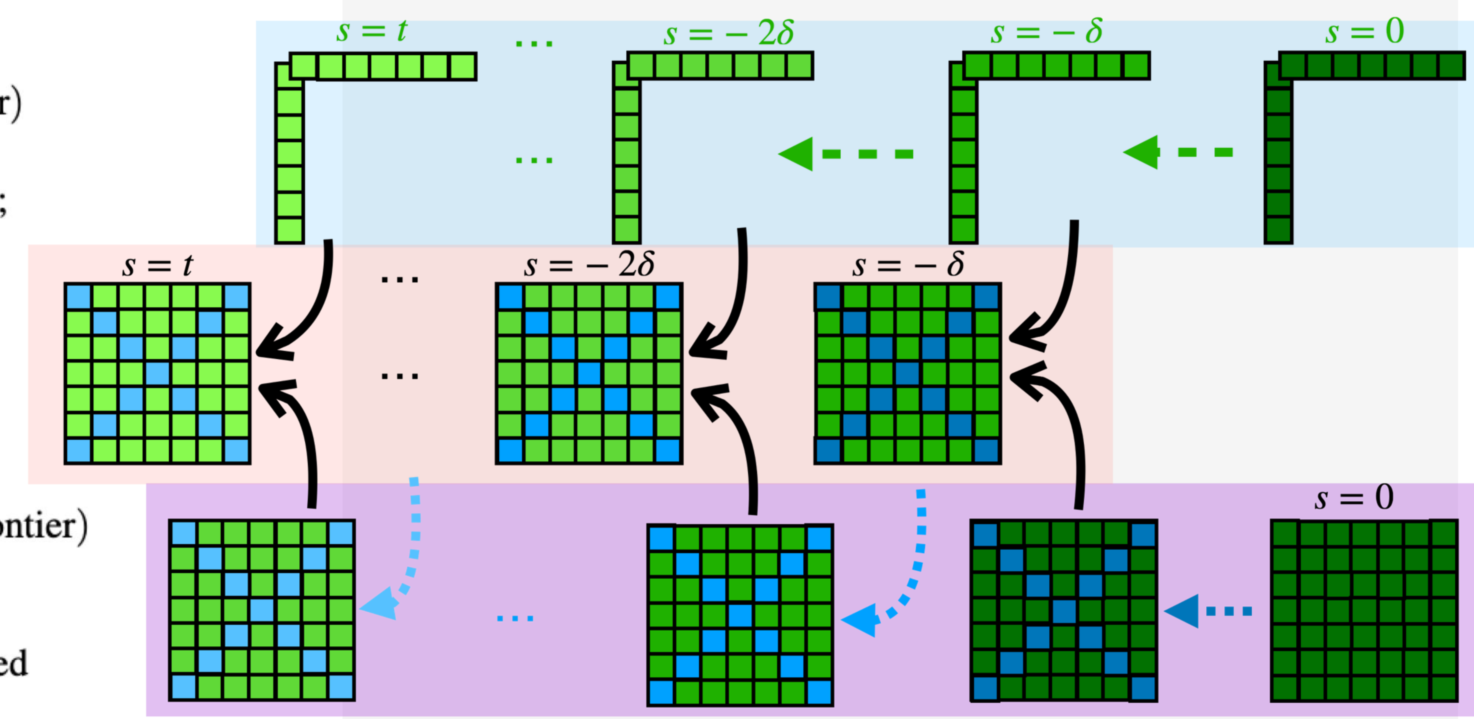
The value of Δ is

$$\Delta = \begin{cases} |\tilde{V}_1^* - V_1|, & \text{if } V_{R,2} \ge V_{R,1} \text{ or } V_{A,1} \ge V_{A,2} \\ |\tilde{V}_2^* - V_2|, & \text{if } V_{R,1} \ge V_{R,2} \text{ or } V_{A,2} \ge V_{A,1}. \end{cases}$$

Algorithm 1: Local updating procedure

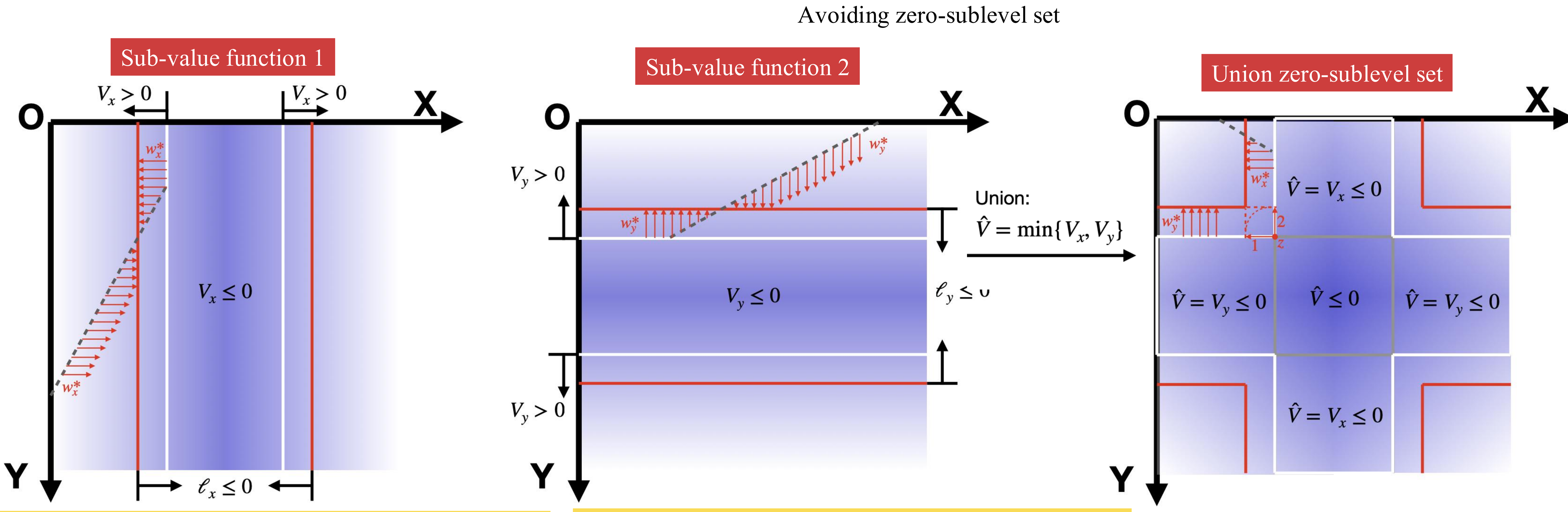
Data: $\hat{V}(\cdot,\cdot), \hat{\mathcal{L}}(\cdot), Z, t_{\text{list}} = [t, t+\delta \ldots, 0]$
Result: $\check{V}(\cdot,\cdot)$
$s \leftarrow 0;$ ▷ Backward Computation
$\check{V}(\cdot, 0) \leftarrow \hat{V}(\cdot, 0);$
Frontier ← nextFrontier ← visited ← {};
while $s > t$ **do**
  for $z \in Z$ **do**
    if $z \in \hat{\mathcal{L}}(s)$ **then**
      updateValue(z, s, δ, Frontier)
    else
      $\check{V}(z, s-\delta) \leftarrow \hat{V}(z, s-\delta);$
    end
  end
  visited ← $\hat{\mathcal{L}}(s)$
  Frontier ← Frontier \ visited
  while *Frontier* $\neq \emptyset$ **do**
    for *z in Frontier* **do**
      updateValue(z, s, δ, nextFrontier)
    end
    visited ← visited ∪ Frontier
    Frontier ← nextFrontier \ visited
    nextFrontier ← {}
  end
  $s \leftarrow s - \delta;$
end
def *updateValue(z, s, δ, Frontier)*:
  $\check{V}(z, s-\delta) \leftarrow$ HJ Update($\check{V}(z,s)$) ▷ Equation 5
  if $\check{V}(z, s-\delta) \neq \hat{V}(z, s-\delta)$ **then**
    Frontier ← Frontier ∪ neighbor(z)
  end

METHOD: LOCAL UPDATING PROCEDURE



CONTROL INCONSISTENCY IN UNION SET- CAUSE OF THE LEAKING CORNER ISSUE

Avoiding zero-sublevel set



$w_x = u_x;$ constrained by $c_1(w_x) = \|u_x\|_2 - \bar{u} \le 0$

$w_y = u_y;$ constrained by $c_2(w_y) = \|u_y\|_2 - \bar{u} \le 0$

When w_x and w_y both acting towards optimal, the full-dimensional control constraint will be violated.

$u = (u_x, u_y);$

Full-dimensional control constraint:

$c(u) = \|u\|_2 - \bar{u} \le 0$

RESULT (2D SINGLE INTEGRATOR)

TABLE I: 2D Accuracy Comparison for One Step

Metric	Before	After
Number of grid points with different values from the ground truth	200	0
Average absolute difference from ground truth	1.2×10^{-4}	9.51×10^{-18}
Maximum absolute difference from ground truth	2×10^{-2}	2.22×10^{-16}

TABLE III: 2D Accuracy Comparison for 10 Steps

Metric	Before	After
Number of states with different values from the ground truth	1344	0
Average absolute difference from ground truth	2.5×10^{-3}	1×10^{-9}
Maximum absolute difference from ground truth	7.39×10^{-2}	2.44×10^{-8}

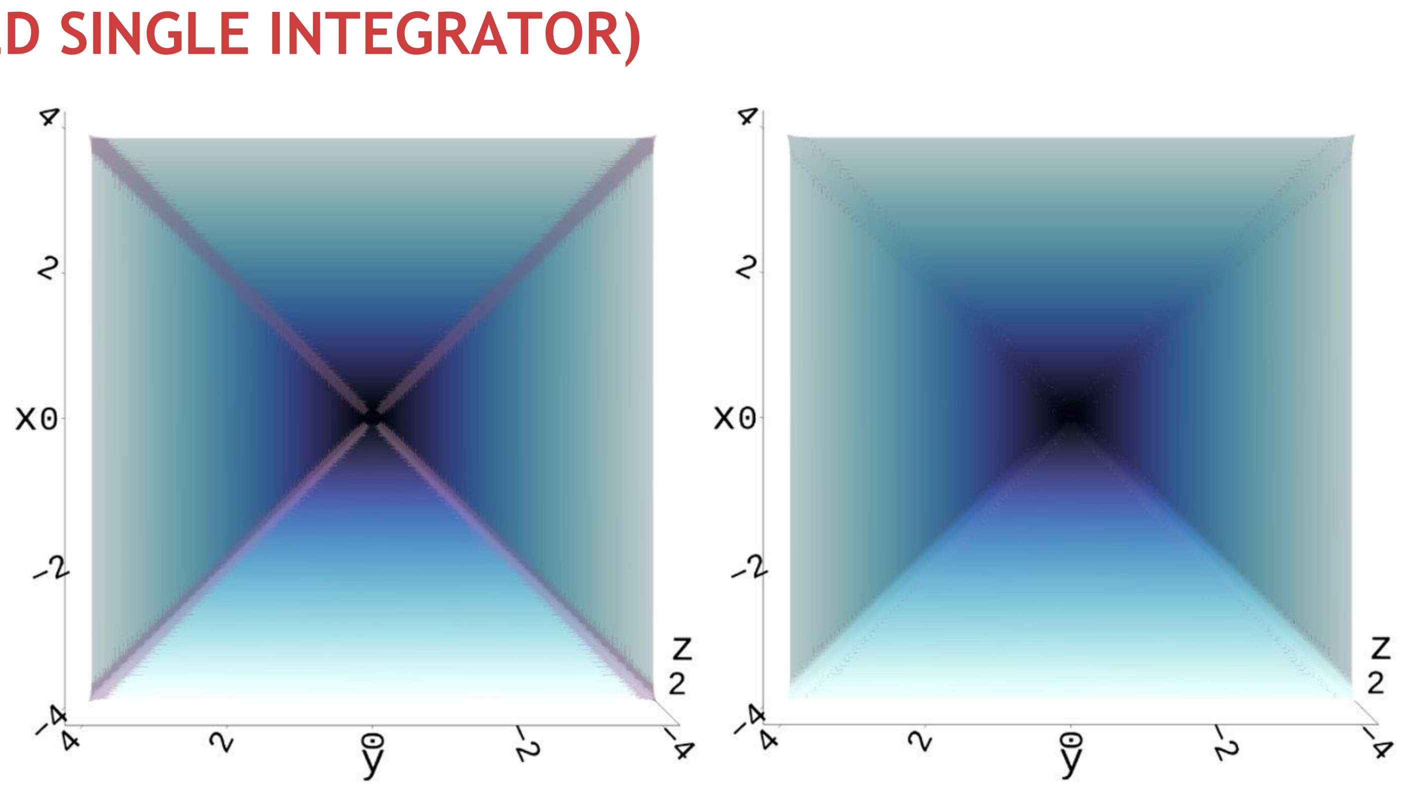


TABLE II: 2D Time Comparison for One Step

Process	Time (seconds)
Direct computation	3.3×10^{-2}
SCSD computation + HJ local update computation	$7 \times 10^{-4} + 1.3 \times 10^{-3} = 2.0 \times 10^{-3}$

TABLE IV: 2D Time Comparison for 10 Steps

Process	Time (seconds)
Direct computation	3.36×10^{-1}
SCSD computation + HJ local update computation	$4.72 \times 10^{-3} + 1.61 \times 10^{-1} = 1.65 \times 10^{-1}$

RESULT (6D PLANAR QUADROTOR)

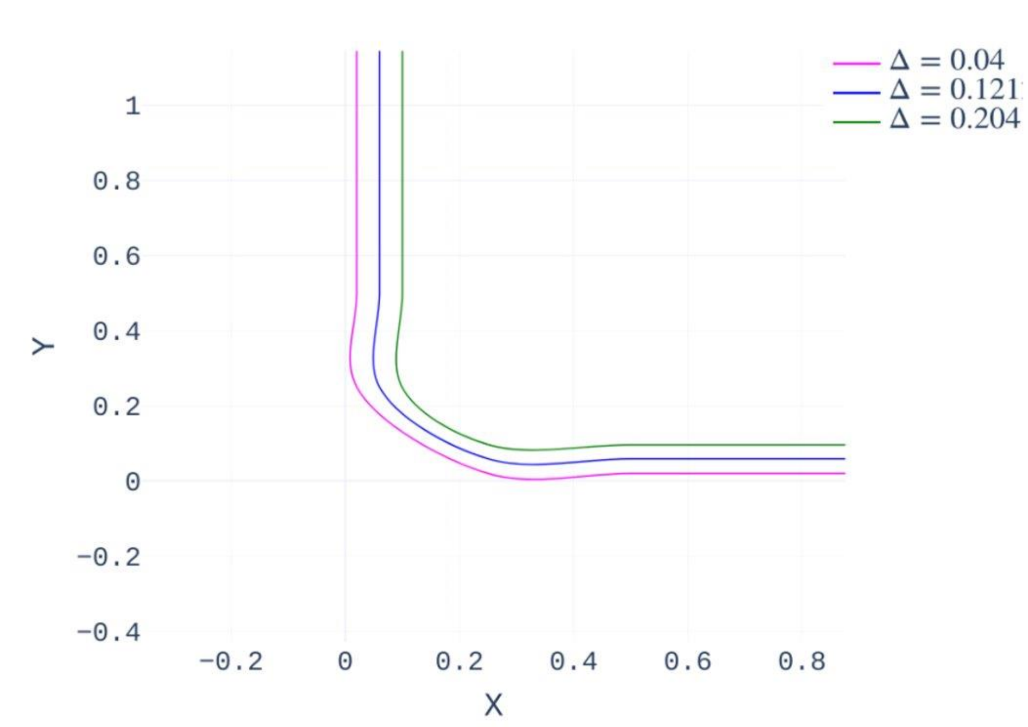


TABLE VI: Computation Time and Delta Value for ts

t (s)	Δ	Decomposition Time + Local Updating Time (seconds)
-0.02	0.04	2.447 + 47.1078 = 49.5548
-0.06	0.1212	6.769 + 157.3528 = 164.1218
-0.1	0.204	11.8038 + 250.7859 = 262.5897

The work was supported by the Canada CIFAR AI Chairs and NSERC Discovery Grants Programs.
Accepted by the Conference on Decision and Control (CDC) 2025
For math proofs, check the paper with QR code.
Contact: chong_he@sfu.ca