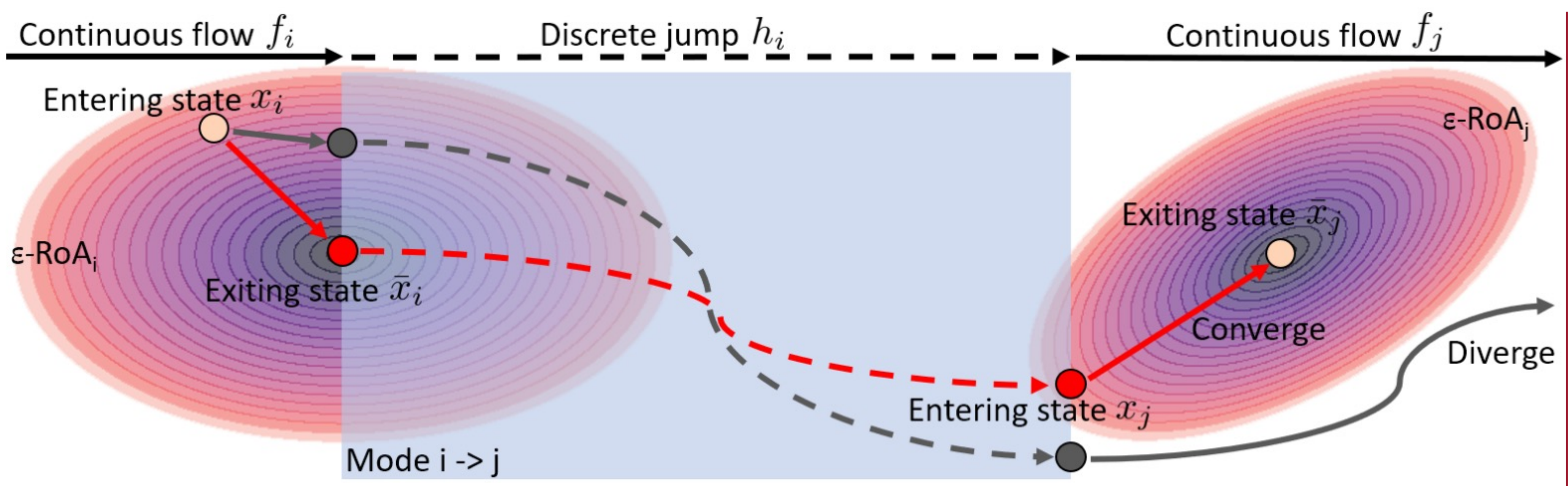
5th L4DC Conference

Hybrid System Neural Control with Region-of-Attraction Planner

Yue Meng Chuchu Fan
Massachusetts Institute of Technology

Introduction

We propose a hierarchical, neural network (NN) method to stabilize hybrid systems via control Lyapunov functions (CLF). **Features:** (1) novel theoretical stability guarantees for hybrid systems (2) strong results in simulations (car tracking control, pogobot navigation and bipedal walker locomotion), with the highest stability/success rate over other baselines such as model-base and model-free reinforcement learning (RL), model predictive control (MPC) and linear quadratic regulator (LQR), less training samples needed compared to RL, and with the computation speed 8-50X faster than MPC.



	Jumps	Guarantee	Scalability
MPC	✓		
RL	✓		✓
Hamilton-Jacobi	✓	✓	
Lyapunov theory		✓	
Our method	✓	✓	✓

Theory

Hybrid system: $\begin{cases} \dot{x} = f_i(x,u), & x \in C_i \quad \text{(flow set)} \\ x^+ = h_{ij}(x,u), & x \in D_{i,j} \quad \text{(jump set)} \end{cases}$

CLF stability: a system under a mode (set point x^*) is stable if: $\exists\, V, \text{s.t.}\, V(x^*) = 0, V(x) > 0 \text{ and } \dot{V}(x) < -\alpha V, \forall x \neq x^*$.

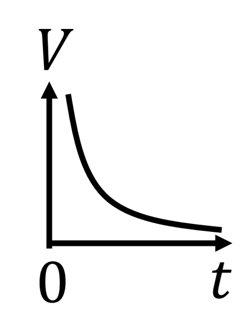
Theorem (hybrid system stability): Assume for each (visited) mode, $\exists\, c_i$, s.t. $\{x | V_i(x) < c_i\}$ can under-approximate the RoA: $S_i = \{x | x(0) = x, |\bar{x}_i - x_i^*| \leq \varepsilon\}$, where $\bar{x}_i$ is exiting mode i. The hybrid system is ε-stable if each mode is CLF-stable, and $\forall i \to j$:

$$V_i(x_i) < c_i \text{ and } V_j(x_j) < \frac{\alpha_j}{\beta_j} c_j - \alpha_j K_{ij} \varepsilon$$

where $x_i (x_j)$ are the entering states for the mode i (j), K_{ij} is the Lipschitz constant for h_{ij} and $\alpha_j |x_j - x_j^*| < V_j(x_j) < \beta_j |x_j - x_j^*|$.

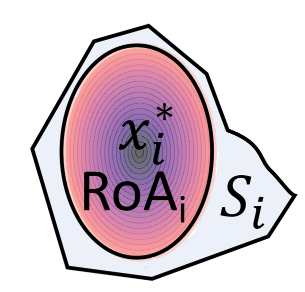
Methodology

Control: For each (sampled) system mode, learn a NN CLF and a NN controller.



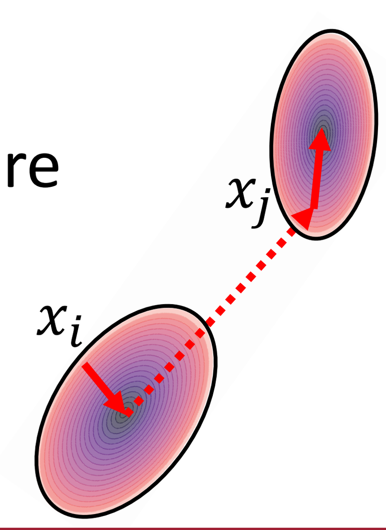
$$\mathcal{L}_{CLF} = \text{ReLU}\left(\gamma V(x_t) + \frac{V(x_{t+1}) - V(x_t)}{\Delta t}\right)$$

RoA: Compute RoA under modes and use NN to predict RoA given the mode.



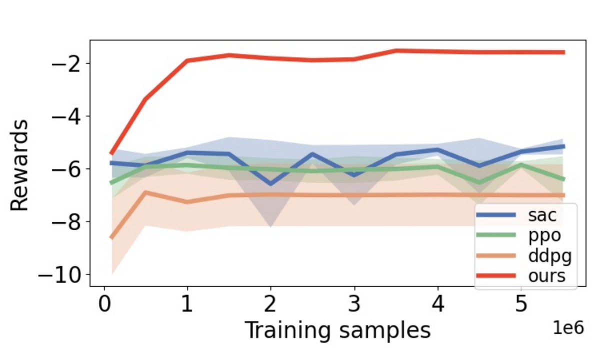
$$\mathcal{L}_{RoA} = \sum_{x_i^*, c_i^*} (R(x_i^*) - c_i^*)^2$$

Planning: Optimize the mode x_i^* to ensure the entering states are in the RoA.

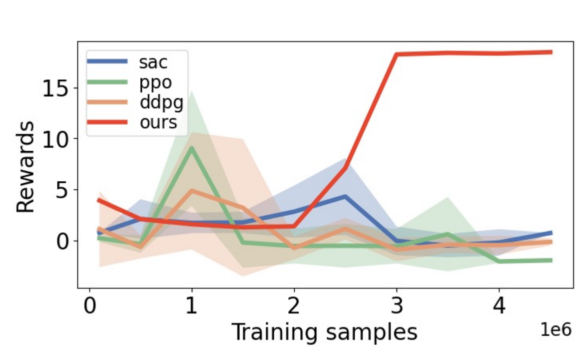


$$\mathcal{L}_{Plan} = \text{ReLU}(V_i(x_i) - R(x_i^*)) + \text{ReLU}(V_j(h_{ij}(x_i^*, u)) - \rho R(x_j^*) + \sigma)$$

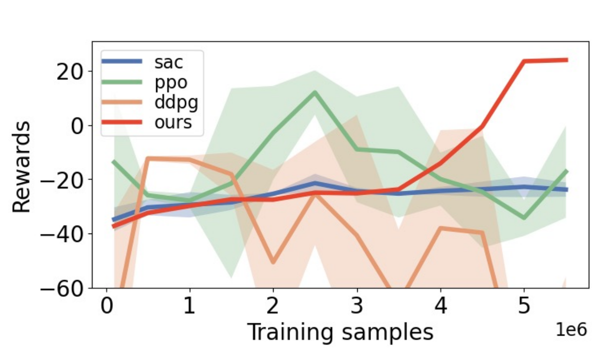
Experimental results



Car tracking control

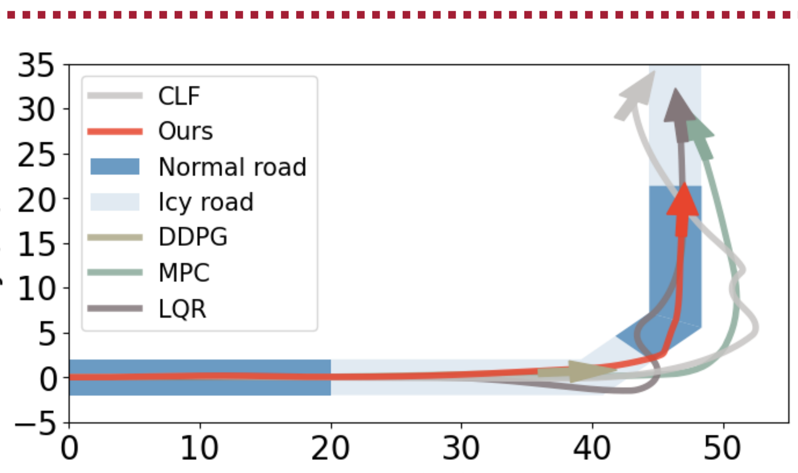


Pogobot navigation

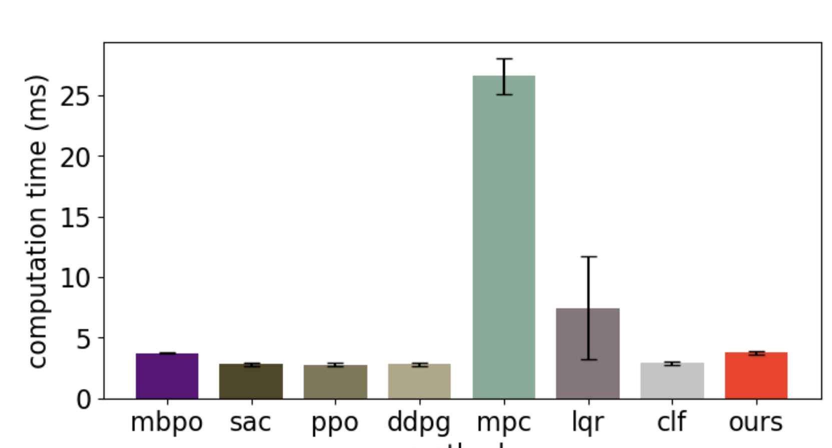


Bipedal locomotion

Reward learning: Compared to RL under the same sample size, we achieve the highest rewards. This is because RL directly interacts with the hybrid systems, while we learn to control the system under each mode, which is easier.



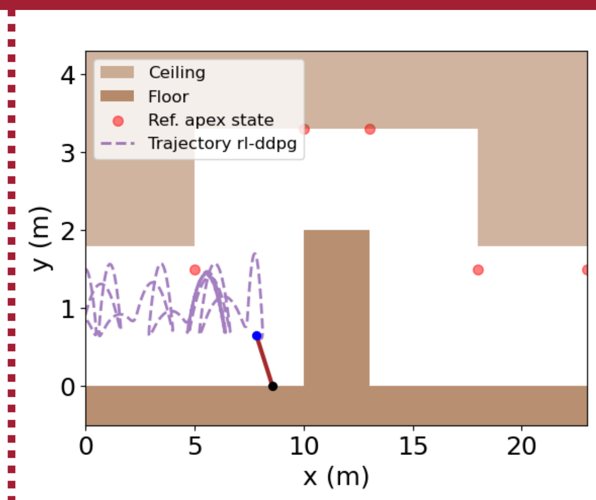
Car trajectory visualization



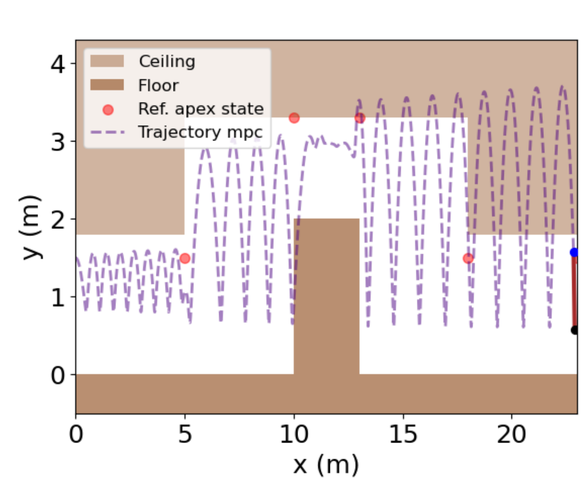
Computation time

Car tracking control results: We control a car on roads with varied frictions. Our approach learns to first turn left and then decelerate to gain more traction for the next icy road segment, whereas other methods fail to keep the car on the road. Our computation speed is 8X faster than the MPC method and close to other learning-based methods.

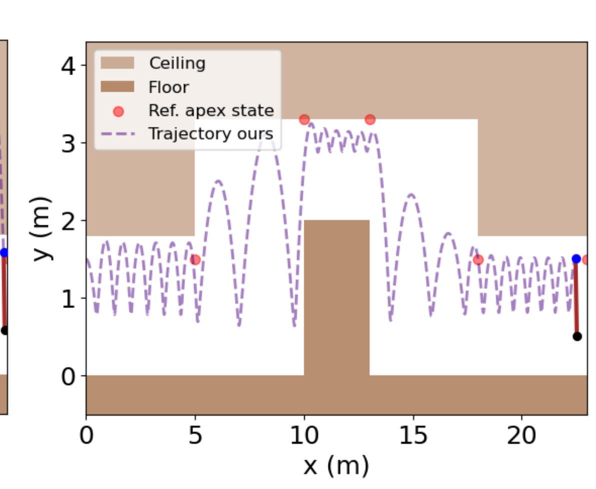
SCAN ME



PPO

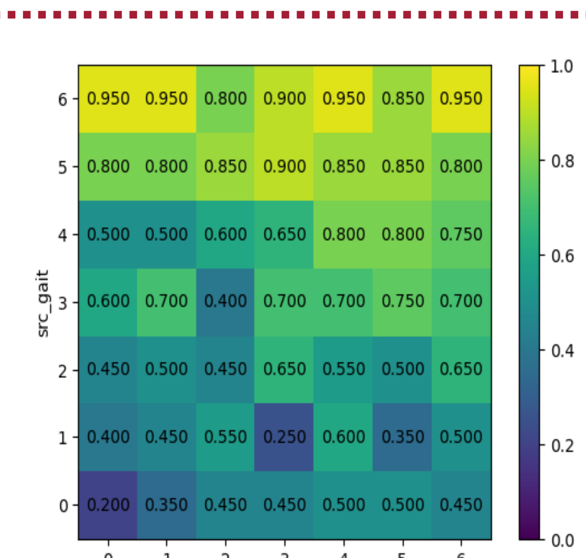


MPC

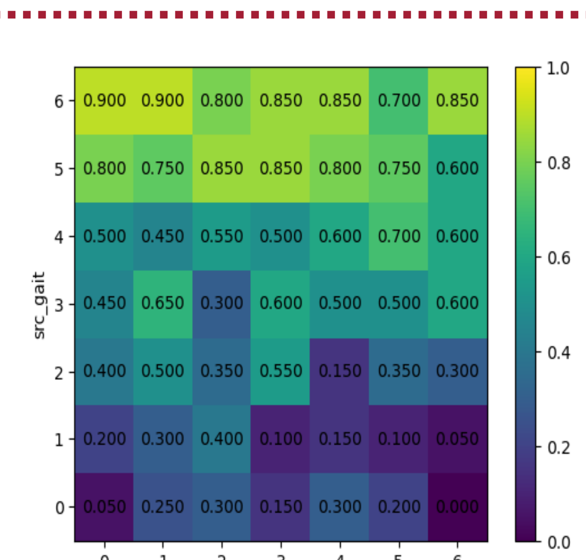


Ours

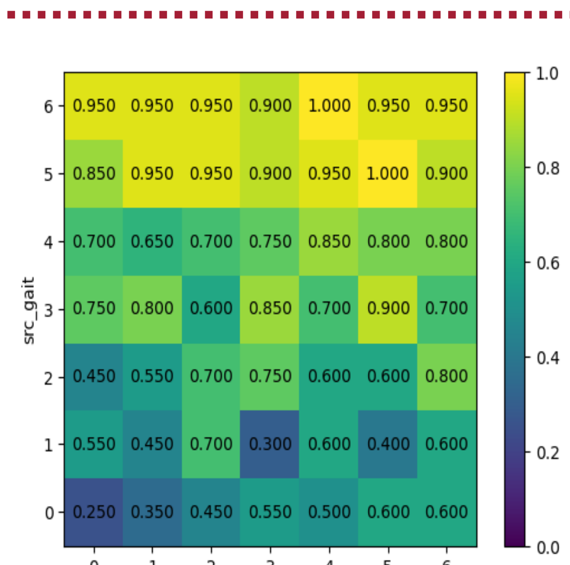
Pogobot navigation visualization: We control a pogobot (Spring-loaded Inverted Pendulum model) to jump through 2D mazes with reference apex states. Our approach can safely finish the task, whereas PPO starts to jump to the left afterwards and MPC causes collisions.



PPO



QP



Ours

Bipedal locomotion success rate heatmap: We control the bipedal robot to reach a target gait (motion pattern). Compared to RL and quadratic program (QP), we obtain a higher success rate over different initial/goal gaits set up.