HyRRT/HySST: Sampling-based Motion Planning Algorithms for Hybrid Dynamical Systems

Nan Wang and Ricardo G. Sanfelice

OCTOBER 1 - 5, 2023
IEEE/RSJ International Conference on Intelligent Robots and Systems

Hybrid Systems Laboratory
University of California Santa Cruz

Overview

Summary

We propose two RRT-type algorithms to address motion planning problems for hybrid dynamical systems, which are characterized by their rapid search capabilities and are accompanied by theoretical guarantees:

1. Our RRT[1] -type algorithm to solve feasible motion planning problems for hybrid systems, which we refer to as HyRRT, is guaranteed to be probabilistically complete.
2. Our SST[2] -type algorithm to solve optimal motion planning problems for hybrid systems, which we refer to as HySST, is guaranteed to be asymptotically near-optimal.
3. Both algorithms possess the ability to rapidly search through high-dimensional problems.

Hybrid Systems

The Modeling Framework [3]

A hybrid system $\mathcal{H}$ with state $x \in \mathbb{R}^n$ and input $x \in \mathbb{R}^m$:

$$\mathcal{H}: \begin{cases} \dot{x} = f(x,u) & (x,u) \in C \\ x^+ = g(x,u) & (x,u) \in D \end{cases} \quad (1)$$

- ➤ C is the flow set
- ➤ f is the flow map
- ➤ D is the jump set
- ➤ g is the jump map

Encapsulates numerous robots' dynamics:

- ➤ Collision-resilient vehicles;
- ➤ Systems with state reset;
- ➤ Robots with multi-modal structure;
- ➤ Purely continuous/discrete-time systems.

The Solution Concept

A solution pair (ϕ, u) of $\mathcal{H}$ is parametrized by $(t,j) \in \mathbb{R}_{\geq 0} \times \mathbb{N}$ on a hybrid time domain dom ϕ. A solution satisfies $\phi(0,0) \in \overline{C} \cup D$ and the dynamics of $\mathcal{H}$. For each j, it satisfies the

➤ Continuous dynamics;

$$\dot{\phi}(t,j) = f(\phi(t,j), u(t,j))$$

for almost all $t \in (t_j, t_{j+1})$, and

$$(\phi(t,j), u(t,j)) \in C \text{ for all } t \in (t_j, t_{j+1}).$$

➤ Discrete dynamics;

$$\phi(t_{j+1}, j+1) = g\left(\phi(t_{j+1}, j), u(t_{j+1}, j)\right)$$

and

$$\left(\phi(t_{j+1}, j), u(t_{j+1}, j)\right) \in D.$$

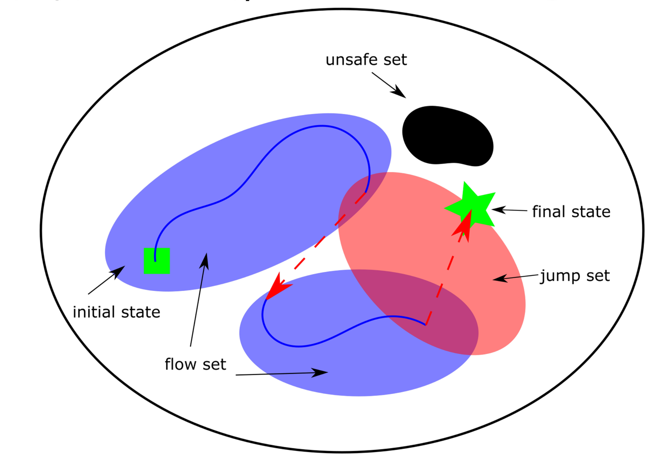
Problem Statement

Problem 1: (Feasible motion planning problem for hybrid systems) Given a hybrid system $\mathcal{H}$ as in (1) with state $x \in \mathbb{R}^n$ and input $u \in \mathbb{R}^m$, the initial state set $X_0 \subset \mathbb{R}^n$, the final state set $X_f \subset \mathbb{R}^n$, the unsafe set $X_u \subset \mathbb{R}^n \times \mathbb{R}^m$, find a pair (ϕ, u) such that for some $(T,J) \in \text{dom}(\phi, u)$, the following hold:

- ➤ $\phi(0,0) \in X_0$;
- ➤ $\phi(T,J) \in X_f$;
- ➤ (ϕ, u) is a solution pair to $\mathcal{H}$;
- ➤ For any $(t,j) \in \text{dom}(\phi, u)$

such that $t + j \leq T + J$,

$(\phi(t,j), u(t,j)) \notin X_u$.



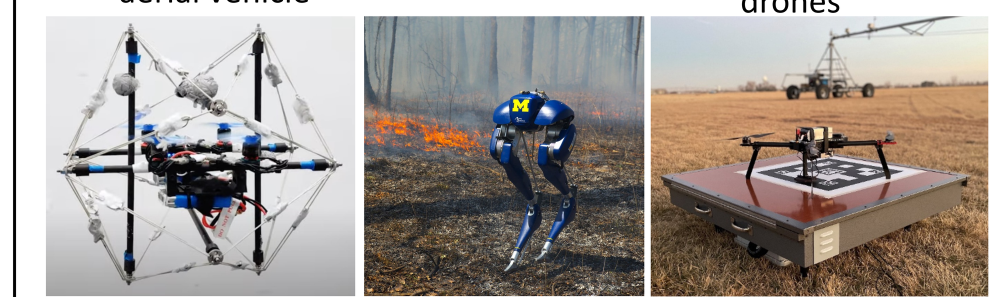
Problem 1 is formulated as $\mathcal{P} = (X_0, X_f, X_u, (C, f, D, g))$

Problem 2: (Optimal motion planning problem for hybrid systems) Given ***Problem 1*** and a cost functional c, find a feasible motion plan (ϕ^*, u^*) to Problem 1 such that $(\phi^*, u^*) = \arg\min_{(\phi,u)} c(\phi)$. Problem 2 is formulated as $\mathcal{P}^* = (X_0, X_f, X_u, (C, f, D, g), c)$.

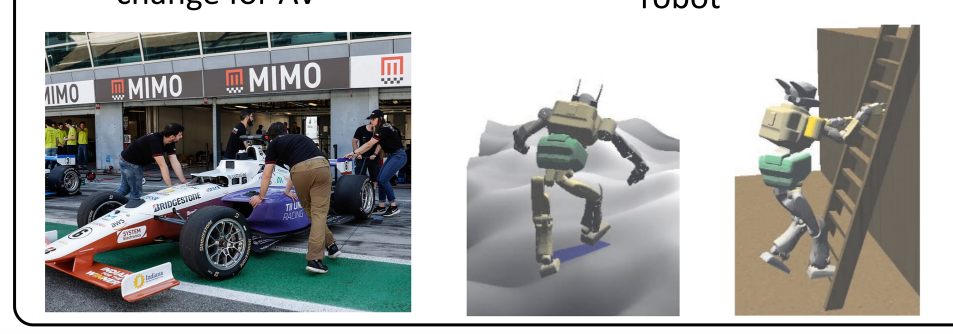
Applications

Possible applications of the proposed motion planning algorithms include:

Collision-resilient aerial vehicle | Walking robot | Recharging for drones



Refuel and tire change for AV | Multimodal robot

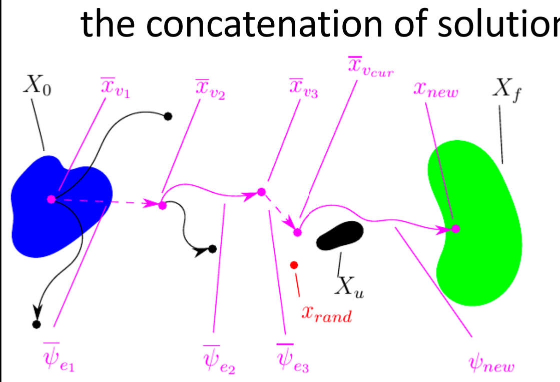


Sampling-based Motion Planning Algorithms for Hybrid Dynamical Systems

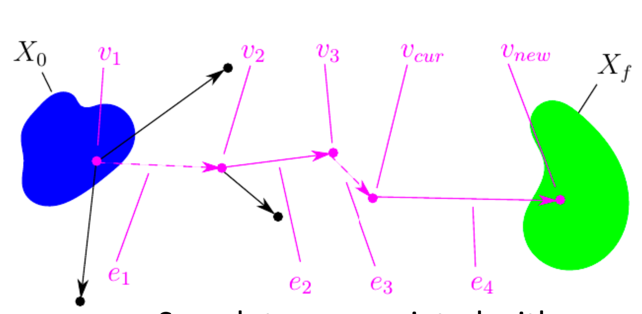
Search Tree Model

The search tree is a pair $\mathcal{T} = (V, E)$, where V is a set whose elements are called vertices, denoted v, and E is a set of paired vertices whose elements are called edges, denoted e. A path in $\mathcal{T}$ is a sequence of vertices $p = (v_1, v_2, \ldots, v_k)$ such that $(v_i, v_{i+1}) \in E$ for all $i \in \{1, 2, \ldots, k-1\}$.

- ➤ Each vertex $v \in V$ in the search tree $\mathcal{T} = (V, E)$ is associated with a state value of $\mathcal{H}$ (and, for HySST, a cost value that, via addition, compounds the cost from the root vertex up to the vertex v)
- ➤ Each edge $e \in E$ in the search tree $\mathcal{T} = (V, E)$ is associated with a solution pair to $\mathcal{H}$.
- ➤ The solution pair that the path $p = (v_1, v_2, \ldots, v_k)$ represents is the concatenation of solution pairs associated with edges therein.



States and solution pairs



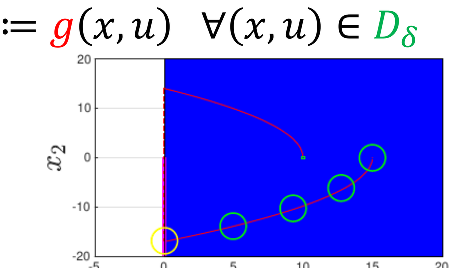
Search tree associated with states and solution pairs

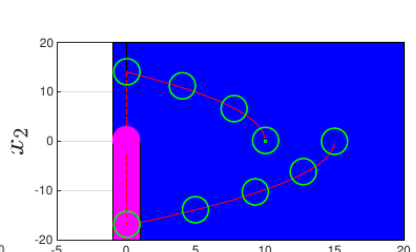
δ —Inflation of Hybrid Systems

Given a hybrid system $\mathcal{H} = (C, f, D, g)$ and $\delta > 0$, the δ-inflation of the hybrid system $\mathcal{H}$, denoted $\mathcal{H}_\delta$ with data $(C_\delta, f_\delta, D_\delta, g_\delta)$, is

$$\mathcal{H}_\delta: \begin{cases} \dot{x} = f_\delta(x,u) & (x,u) \in C_\delta \\ x^+ = g_\delta(x,u) & (x,u) \in D_\delta \end{cases}$$

- ➤ $C_\delta := \{(x,u) \in \mathbb{R}^n \times \mathbb{R}^m : \exists (y,v) \in C : x \in y + \delta\mathbb{B}, u \in v + \delta\mathbb{B}\}$
- ➤ $f_\delta := f(x,u) \quad \forall (x,u) \in C_\delta$
- ➤ $D_\delta := \{(x,u) \in \mathbb{R}^n \times \mathbb{R}^m : \exists (y,v) \in D : x \in y + \delta\mathbb{B}, u \in v + \delta\mathbb{B}\}$
- ➤ $g_\delta := g(x,u) \quad \forall (x,u) \in D_\delta$





Probabilistically Complete HyRRT Algorithm [4]

The inputs to HyRRT are X_0, X_f, X_u, $\mathcal{H}$, the input library $\mathcal{U}_C$ and $\mathcal{U}_D$, a parameter $p_n \in (0,1)$, which tunes the probability of proceeding with the flow regime or the jump regime, and an upper bound $K \in \mathbb{N}$ for the number of iterations to execute.

Algorithm 1 HyRRT algorithm

```
Input: X_0, X_f, X_u, H = (C, f, D, g), (U_C, U_D), p_n ∈ (0,1), K ∈ N_>0
T.init(X_0).
for k = 1 to K do
    randomly select a real number r from [0,1].
    if r ≤ p_n then
        x_rand ← random_state(C̄').
        extend(T, x_rand, (U_C, U_D), H, X_u, X_c).
    else
        x_rand ← random_state(D').
        extend(T, x_rand, (U_C, U_D), H, X_u, X_d).
    end if
end for
return T.
```

Algorithm 2 Extend function

```
function EXTEND((T, x, (U_C, U_D), H, X_u, X_*))
    v_cur ← nearest_neighbor(x, T, H, X_*);
    (is_a_new_vertex_generated, x_new, ψ_new) ← new_state(v_cur, (U_C,
  U_D), H, X_u)
    if is_a_new_vertex_generated = true then
        v_new ← T.add_vertex(x_new);
        T.add_edge(v_cur, v_new, ψ_new);
        return Advanced;
    end if
    return Trapped;
end function
```

Theorem 1. (Probabilistic Completeness of HyRRT)
Suppose there exists a motion plan (ϕ, u) to $\mathcal{P} = (X_0, X_f, X_u, (C, f, D, g))$. When HyRRT is used to solve $\mathcal{P} = (X_0, X_f, X_u, (C_\delta, f_\delta, D_\delta, g_\delta))$, the probability that HyRRT fails to find a motion plan (ϕ', u') such that ϕ' is close to ϕ after k iterations is at most ae^{-bk}, where $a, b > 0$.

Asymptotically Near-Optimal HySST Algorithm [5]

In addition to the inputs to HyRRT, HySST requires parameters $\delta_{BN} > 0$ and $\delta_s > 0$ to tune the optimization of the cost and sparsification of the vertices.

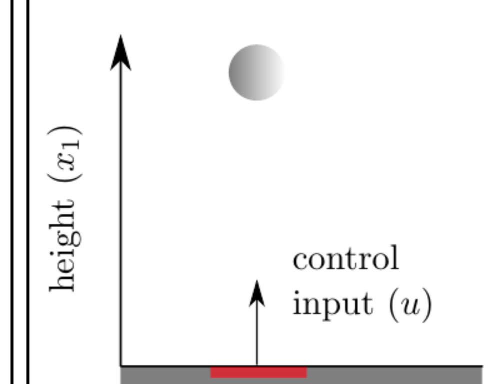
Algorithm 3 HySST algorithm

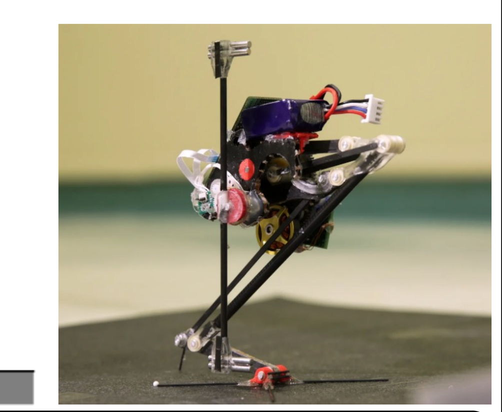
```
Input: X_0, X_f, X_u, c, H = (C, f, D, g), (U_C, U_D), p_n ∈ (0,1),
K ∈ N, X_c, X_d, δ_BN and δ_s
T.init(X_0);
V_active ← V, V_inactive ← ∅, S ← ∅;
for all v_0 ∈ V do
    if is_vertex_locally_the_best(x̄_v0, 0, S, δ_s) then
        (S, V_active, V_inactive, E) ← prune_dominated_
  vertices(v_0, S, V_active, V_inactive, E)
    end if
end for
for k = 1 to K do
    randomly select a real number r from [0,1];
    if r ≤ p_n then
        x_rand ← random_state(C̄');
        v_cur ← best_near_selection(x_rand, V_active, δ_BN,
  X_c);
    else
        x_rand ← random_state(D');
        v_cur ← best_near_selection(x_rand, V_active, δ_BN,
  X_d);
    end if
    (is_a_new_vertex_generated, x_new, ψ_new, cost_new)
  ← new_state(v_cur, (U_C, U_D), H, X_u)
    if is_a_new_vertex_generated & is_vertex_locally
  _the_best(x_new, cost_new, S, δ_s) then
        v_new ← V_active.add_vertex(x_new, cost_new);
        E.add_edge(v_cur, v_new, ψ_new);
        (S, V_active, V_inactive, E) ← prune_dominated_
  vertices(v_new, S, V_active, V_inactive, E);
    end if
end for
return T;
```

Theorem 2. (Asymptotic Near-Optimality of HySST)
Suppose there exists an optimal motion plan (ϕ^*, u^*) to $\mathcal{P}^* = (X_0, X_f, X_u, (C, f, D, g), c)$. When HySST is used to solve $\mathcal{P} = (X_0, X_f, X_u, (C_\delta, f_\delta, D_\delta, g_\delta), c)$, the probability that HySST finds a motion plan (ϕ', u') such that $c(\phi') < (1 + a\delta)c(\phi^*)$ converges to 1 as the number of iterations approaches infinity.

Application: Motion Planning for Jumping Insect Robot

A jumping insect robot can be modeled as a ball bouncing on a fixed horizontal surface. The surface is located at the origin and, through control actions, is capable of affecting the velocity of the ball after the impact.





Hybrid System Model for Jumping Insect Robot

The dynamics of the ball while in the air is given by

$$\dot{x} = \begin{bmatrix} x_2 \\ -\gamma \end{bmatrix} =: f(x,u)$$

where $x := (x_1, x_2) \in \mathbb{R}^2$ and $u \in \mathbb{R}^1$. The height of the ball is denoted by x_1. The velocity of the ball is denoted by x_2. The gravity constant is denoted by γ. The flow is allowed when the ball is above the surface. Hence, the flow set is

$$C := \{(x,u) \in \mathbb{R}^2 \times \mathbb{R} : x_1 \geq 0\}$$

At every impact, the velocity of the ball changes from pointing down to pointing up while the height remains the same. The dynamics at jumps of the actuated bouncing ball system is given as

$$x^+ = \begin{bmatrix} x_1 \\ -\lambda x_2 + u \end{bmatrix} =: g(x,u)$$

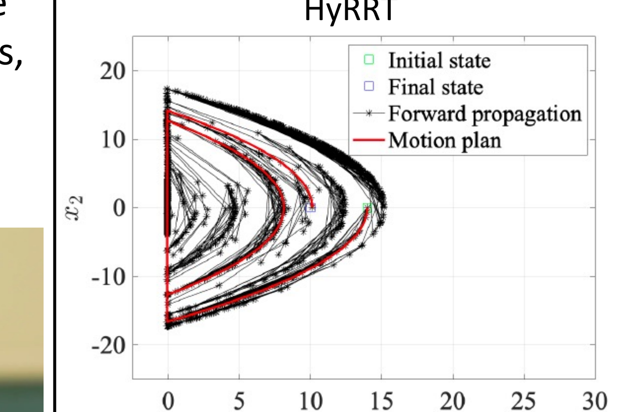
where and $u > 0$ is the input and $\lambda \in (0,1)$ is the coefficient of restitution.

The jump is allowed when the ball is on the surface with negative velocity. Hence, the jump set is

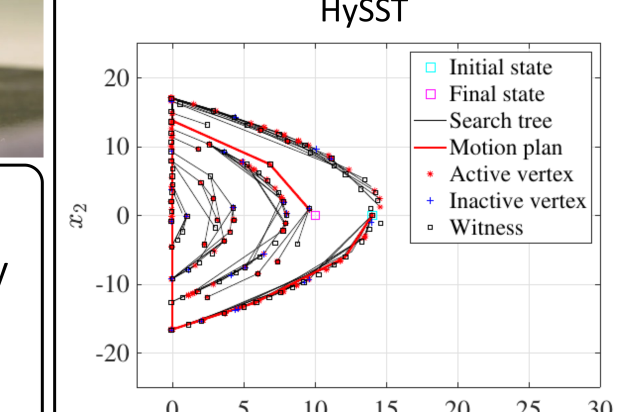
$$D := \{(x,u) \in \mathbb{R}^2 \times \mathbb{R} : x_1 = 0, x_2 \leq 0, u \geq 0\}$$

Simulation Results

HyRRT

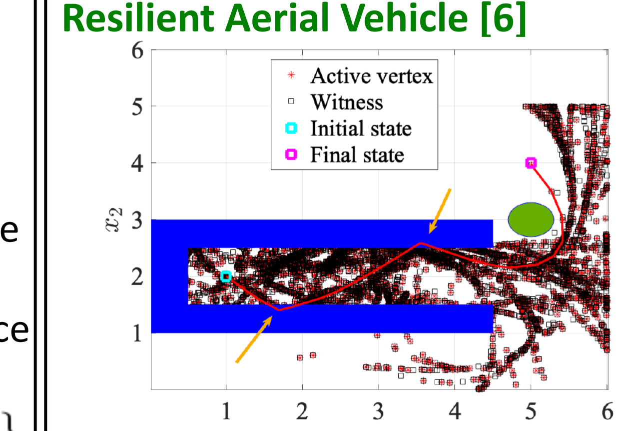


HySST



The HySST creates 154 active vertices and 35 inactive vertices and takes 3.30 seconds, while HyRRT creates 660 vertices in total and takes 18.4 seconds on average.

Other Application: Collision Resilient Aerial Vehicle [6]



Acknowledgement: This research has been partially supported by NSF Grants no. CNS- 2039054 and CNS-2111688, by AFOSR Grants nos. FA9550-19-1-0169, FA9550-20-1-0238, FA9550-23-1-0145, and FA9550-23-1-0313, by AFRL Grant nos. FA8651-22-1-0017 and FA8651-23-1-0004, by ARO Grant no. W911NF-20-1-0253, and by DoD Grant no. W911NF-23-1-0158.

Members of the Hybrid Systems Laboratory (HSL) at the University of California, Santa Cruz, Department of Electrical and Computer Engineering. Principal Investigator: Dr. Ricardo G. Sanfelice

Selected References

[1] LaValle, Steven M., and James J. Kuffner Jr. "Randomized kinodynamic planning." The international journal of robotics research 20.5 (2001): 378-400.
[2] Li, Yanbo, Zakary Littlefield, and Kostas E. Bekris. "Asymptotically optimal sampling-based kinodynamic planning." The International Journal of Robotics Research35.5 (2016): 528-564.
[3] Sanfelice, Ricardo G. Hybrid feedback control. Princeton University Press, 2021.
[4] Wang, Nan, and Ricardo G. Sanfelice. "A rapidly-exploring random trees motion planning algorithm for hybrid dynamical systems." 2022 IEEE 61st Conference on Decision and Control (CDC). IEEE, 2022.
[5] Wang, Nan, and Ricardo G. Sanfelice. "HySST: A Stable Sparse Rapidly-Exploring Random Trees Optimal Motion Planning Algorithm for Hybrid Dynamical Systems." arXiv preprint arXiv:2305.18649 (2023).
[6] J. Zha and M. W. Mueller, "Exploiting collisions for sampling-based multicopter motion planning," in 2021 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2021, pp. 7943–7949.