# Controlled Reach-avoid Set Computation for Discrete-time Polynomial Systems via Convex Optimization

Bai Xue
Joint work with Taoran Wu, Yiling Xue, Dejin Ren, Arvind Easwaran, and Martin Fränzle

Institute of Software, Chinese Academy of Sciences

February 11, 2026

# Outline

# I. Introduction: Motivation & Applications

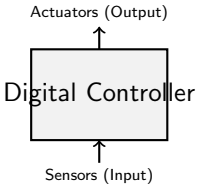
**Why Discrete-time Control?**

**(**$x(k+1) = f(x(k), u(k)), \quad x(0) = x_0$**,** $u \in \mathcal{U}$**)**

- Digital computers and embedded controllers naturally operate in discrete steps.
- Applications in **Consumer Electronics**, **Medical Devices**, and **Autonomous Robotics**.

**Why Polynomial Systems?**

- A highly flexible model class.
- Capable of capturing both **complex nonlinearities** and linear dynamics seen in physical systems.



# I. Introduction: The Reach-avoid Problem

- **Reach Task:** Guarantee that a system reaches a target set $\mathcal{T}$.
- **Safety Constraint:** The system must stay within a safe set $\mathcal{X}$ at every intermediate step.
- **CRAS:** The *Controlled Reach-avoid Set* represents the region of all possible "starts" such that the system completes the reach task safely.

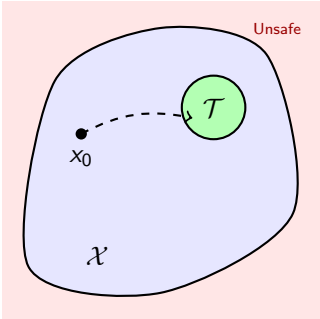



Figure: Safe trajectory from $x_0$ to $\mathcal{T}$.

# I. Introduction: Challenges in Computing CRASs

## Proposition 1 (Lyapunov-like Characterization)

If there exists a Lyapunov-like function $v(x) : \mathbb{R}^n \to \mathbb{R}$, which is bounded and continuous, satisfying:

$$\begin{cases} \max_{u \in \mathcal{U}} v(f(x,u)) - \lambda v(x) \geq 0, & \forall x \in \mathcal{X} \setminus \mathcal{T} \\ v(x) \leq 0, & \forall x \in \hat{\mathcal{X}} \setminus \mathcal{X} \end{cases}$$

where $\lambda > 1$, then $\{x \in \mathcal{X} \mid v(x) > 0\} \neq \emptyset$ is a CRAS.

**Shortcomings of Prop 1**

- **Numerical Intractability:** The *maximum* operator is non-convex.
- **Composition Problem:** Composition of control variables and Lyapunov functions makes direct optimization via standard convex solvers impossible.

# I. Introduction: Challenges in Computing CRASs

## Proposition 2 (Feedback Control Policy)

If there exists a bounded continuous $v(x)$ and a control function $u(x) : \mathbb{R}^n \to \mathcal{U}$, satisfying:

$$\begin{cases} v(f(x, u(x))) - \lambda v(x) \geq 0, & \forall x \in \mathcal{X} \setminus \mathcal{T} \\ v(x) \leq 0, & \forall x \in \hat{\mathcal{X}} \setminus \mathcal{X} \end{cases}$$

where $\lambda > 1$, then $\{x \in \mathcal{X} \mid v(x) > 0\} \neq \emptyset$ is a CRAS.

**Shortcomings of Proposition 2**

- **Bilinear Non-convexity:** Simultaneously fitting $v(x)$ and a controller $u(x)$ is notoriously difficult.
- **Discrete-time Gap:** In discrete systems, $v(f(x, u(x)))$ introduces high-degree terms like $u^2$ or $u^3$.

# I. Introduction: The Discrete-time Composition Problem

**Why is $v(f(x, u(x)))$ so hard to solve?** Consider a simple 1D system and candidate functions:

- **Dynamics:** $x(k+1) = -x + u$
- **Lyapunov Function:** $v(x) = ax^2$ (Find coefficient $a$)
- **Control Policy:** $u(x) = bx$ (Find coefficient $b$)

Evaluating $v$ at the **next step**:

$$v(f(x, u)) = a(-x + bx)^2 = \mathbf{a(1 - b)^2}x^2$$

**The Shortcoming:**

- **High-Degree Coupling:** The decision variables $a$ and $b$ are coupled as $\mathbf{a \cdot b^2}$.
- **Non-convexity:** This interaction is non-convex and high-degree, causing standard SOS solvers to get stuck or fail.

# I. Introduction: Main Contributions

1. **Probabilistic Viewpoint:** We establish for the first time that CRASs are equivalent to **probabilistic 0-reach-avoid sets**.
2. **Convex Framework:** By using an expectation operator, we eliminate non-convex terms, making the problem **computationally tractable** via SOS.
3. **Iterative Expansion:** We propose an $\epsilon$-**greedy inspired algorithm** to progressively enlarge computed sets and reduce conservatism.

# II. Preliminaries: System Dynamics

## Discrete-time State Update

$$x(k+1) = f(x(k), u(k)), \quad x(0) = x_0$$

- **State & Control:** $x(k) \in \mathbb{R}^n$ is the state; $u(k) \in \mathcal{U} \subset \mathbb{R}^m$ is the control input.
- **Input Set:** $\mathcal{U} = \{u \mid \underline{u} \leq u \leq \bar{u}\}$ is a compact set in interval form.
- **Control Policy:** An infinite sequence of control inputs $\pi = \{u(k)\}_{k \in \mathbb{N}}$.

# II. Preliminaries: System Trajectory

Given an initial state $x_0 \in \mathbb{R}^n$ and a control policy $\pi$, the resulting **trajectory** is the sequence $\{\phi_{x_0}^{\pi}(k)\}_{k\in\mathbb{N}}$ satisfying:

1. **Initial Condition:** $\phi_{x_0}^{\pi}(0) = x_0$
2. **Recursive Update:** $\phi_{x_0}^{\pi}(k+1) = f(\phi_{x_0}^{\pi}(k), u(k)), \quad \forall k \in \mathbb{N}$

**Set Constraints:**

- **Safe Set:** $\mathcal{X} = \{x \in \mathbb{R}^n \mid h(x) < 0\}$
- **Target Set:** $\mathcal{T} = \{x \in \mathbb{R}^n \mid g(x) < 0\}$, where $\mathcal{T} \subseteq \mathcal{X}$

# II. Preliminaries: Formal Definition of CRAS

## Definition (Controlled Reach-avoid Set)

A CRAS $\mathcal{R} \subseteq \mathcal{X}$ for system (1) is a set of initial states such that, for any $x_0 \in \mathcal{R}$, there exists a control policy $\pi$ that drives the system to the target set $\mathcal{T}$ in finite time $k_0 \in \mathbb{N}$ while remaining in the safe set $\mathcal{X}$ at all intermediate steps:

$$\exists \pi . \exists k_0 \in \mathbb{N}. \left[ \phi^{\pi}_{x_0}(k_0) \in \mathcal{T} \wedge \forall k < k_0 . \phi^{\pi}_{x_0}(k) \in \mathcal{X} \right]$$

- **Intuition:** It is the set of all "recoverable" or "safely steerable" states.
- **Goal:** We seek a large under-approximation of the maximal CRAS.

# III. Probabilistic Reformulation: Probability Space for Policy $\pi$

To formalize the $p$-reach-avoid sets, we endow the control input with randomness:

- **Base Space:** $u(k)$ are drawn i.i.d. from a probability space $(\Omega, \mathcal{F}, P)$ on $\mathcal{U}$.
- **Stochastic Process:** Policy $\pi$ is a realization of $\{u(i) : \Omega \to \mathcal{U}, i \in \mathbb{N}\}$.
- **Measure:** Associated with the induced probability measure $P^\infty$ on $\mathcal{U}^\infty$.
- **Expectation:** Denoted by $\mathbb{E}^\infty[\cdot]$ over the realization of control inputs.

# III. Probabilistic Reformulation: Definition 2 ($p$-reach-avoid sets)

## Definition (p-reach-avoid sets)

A $p$-reach-avoid set $R_p$ is a set of initial states $x_0$ such that trajectories reach $\mathcal{T}$ safely with probability strictly larger than $p \in [0, 1)$:

$$P^{\infty}\left(\exists k_0 \in \mathbb{N}.[\phi^{\pi}_{x_0}(k_0) \in \mathcal{T} \wedge \forall k < k_0.\phi^{\pi}_{x_0}(k) \in \mathcal{X}]\right) > p$$

- **Bridge:** By setting $p = 0$, we focus on the set where a safe path exists with any non-zero probability.

# III. Probabilistic Reformulation: Theorem 1 (Equivalence)

## Theorem 1

Under Assumption 1 (Full support distribution $P$ on $\mathcal{U}$), a CRAS $R$ is equivalent to a probabilistic 0-reach-avoid set $R_0$.

- **Direction $R \subseteq R_0$:** If a deterministic safe policy exists, **Lipschitz continuity** of $f$ ensures a small "tube" of valid controls exists. Full support of $P$ means this tube is sampled with probability $> 0$.
- **Direction $R_0 \subseteq R$:** If reaching the target safely has probability $> 0$, the set of successful control sequences is non-empty. Thus, at least one deterministic policy must exist.

# III. Probabilistic Reformulation: Proposition 2 (Convexity)

Finding a CRAS is now equivalent to solving for $v(x)$ in:

$$\begin{cases} (\max_{u\in\mathcal{U}} v(f(x,u)) \geq)\mathbb{E}[v(f(x,u))] - \lambda v(x) \geq 0, & \forall x \in \mathcal{X}\setminus\mathcal{T}, \quad \forall x \in \mathcal{X}\setminus\mathcal{T} \\ v(x) \leq 0, & \forall x \in \hat{\mathcal{X}}\setminus\mathcal{X} \end{cases}$$

where $\lambda > 1$, then $\{x \in \mathcal{X} \mid v(x) > 0\} \neq \emptyset$ is a 0-reach-avoid set and thus a CRAS.

- This transformation eliminates the nonlinear control terms via the Expectation operator.
- The optimization for $v(x)$ is now **convex** and solvable via SOS.

$$\max \int_{\mathcal{X}} v(x)dx \tag{1}$$

$$\text{s.t.} \begin{cases} \mathbb{E}[v(f(x,u))] - \lambda v(x) + s_1(x)h(x) - s_2(x)g(x) \in \sum[x], \\ -v(x) + s_3(x)\hat{h}(x) - s_4(x)h(x) \in \sum[x], \\ s_1(x), s_2(x), s_3(x), s_4(x) \in \sum[x], \end{cases} \tag{2}$$

# IV. Iterative Refinement: Strategy Motivation

- **Conservatism:** Since $\mathbb{E}[v(f(x,u))] \leq \max_{u \in \mathcal{U}} v(f(x,u))$, a fixed uniform distribution often underestimates the CRAS.
- **The Goal:** Adjust the control distribution iteratively to concentrate probability on inputs that drive the system toward the target (Refined $v(x)$).
- **Approach:** Use an $\epsilon$-greedy strategy to balance **exploitation** of the best-known controller and **exploration** of the input space.

# IV. Iterative Refinement: $\epsilon$-greedy Probability Density

The control input is assigned a state-dependent probability density $\rho(u)$ at each iteration:

$$\rho(u) = \begin{cases} \frac{1-\epsilon}{Z}, & \text{if } u \in [\tilde{u}(x) - \delta, \tilde{u}(x) + \delta] \subseteq \mathcal{U} \\ \frac{\epsilon}{Z}, & \text{if } u \in \mathcal{U} \setminus [\tilde{u}(x) - \delta, \tilde{u}(x) + \delta] \end{cases}$$

- $\tilde{u}(x)$: Approximate optimal controller fitted from current data.
- $\delta$: Small neighborhood for exploitation.
- $\epsilon$: Weight for global exploration (ensures Theorem 1 holds).
- $Z$: Normalization constant($Z = (1-\epsilon)(2\delta)^m + \epsilon \prod_{j=1}^{m} (\overline{u}(j) - \underline{u}(j))$).

# IV. Iterative Refinement: Controller Fitting Step

To find the polynomial controller $\tilde{u}(a, x)$ for the next distribution:

1. **Sample:** Draw $N$ states $\{x_i\}$ from the set $\mathcal{X} \setminus \mathcal{T}$.
2. **Optimize:** Discretize $\mathcal{U}$ to find $u_i = \arg\max_u v_{l-1}(f(x_i, u))$ for each state $x_i$.
3. **Fit:** Solve an SOS regression problem to find coefficients $a$:

$$\min \sum_{i=1}^{N} \|\tilde{u}(a, x_i) - u_i\|_2^2, \quad \text{s.t. } \tilde{u}(a, x) \in \mathcal{U}$$

# IV. Iterative Refinement: Proposition 3 & Proof Logic

## Proposition 3 (Mixture Condition)

Satisfying the $\epsilon$-greedy mixture expectation is a **formal sufficient condition** for the deterministic reach-avoid property:

$$\mathbb{E}_\rho[v(f(x,u))] - \lambda v(x) \geq 0 \implies \max_{u \in \mathcal{U}} v(f(x,u)) - \lambda v(x) \geq 0,$$

where $\mathbb{E}_\rho[v(f(x,u))] = \frac{1-\epsilon}{Z} \int_{\tilde{u}_0(x)-\delta}^{\tilde{u}_0(x)+\delta} v(f(x,u))du + \frac{\epsilon}{Z} \int_{\underline{u}}^{\overline{u}} v(f(x,u))du$.

**Proof Logic:** The average value (expectation) is always bounded by the maximum value. If the average exceeds the threshold, the maximum must also exceed it.

# IV. Iterative Refinement: Algorithm 1 Walkthrough

1. **Solve initial CRAS:** Use uniform distribution in the first SOS optimization to find $v_0(x)$.
2. **Refine:** For iterations $l = 1 \dots L$:
   - Generate data $\{(x_i, u_i)\}$ maximizing $v_{l-1}$.
   - Fit polynomial controller $\tilde{u}_l(x)$.
   - Solve SOS optimziation with mixture distribution to find expanded $v_l(x)$.
   - Update $\epsilon$ (exploitation focused).
3. **Return:** Final set $\{x \mid v_L(x) > 0\}$.

# V. Experiments: Computational Setup

- **Hardware:** i9-12900H 2.5GHz CPU, 16GB RAM.
- **Volume Estimation:** Monte Carlo sampling with $10^6$ points.
- **Software:** Matlab with YALMIP and Mosek (SDP solver).

# V. Experiments: Example 1 Dynamics (VanderPol)

## System Equations

$$\begin{cases} x(k+1) = x(k) + \tau(-2y(k)) \\ y(k+1) = y(k) + \tau(0.8x(k) - 10(y(k) - 0.21)y(k) + u(k)) \end{cases}$$

- $\tau = 0.01, \mathcal{U} = [-3, 3]$.
- **Safe Set:** $x^2 + y^2 - 1 < 0$; **Target:** $x^2 + y^2 - 0.01 < 0$.

# V. Experiments: Volume Estimation via Monte Carlo

To quantitatively evaluate the expansion of the CRAS, we estimate its volume relative to the safe set $\mathcal{X}$:

- **Sampling:** We sample $N = 10^6$ points uniformly within the safe set $\mathcal{X}$.
- **Membership Test:** For each sampled point $x_i$, we check if it lies within the computed CRAS:

$$x_i \in \text{CRAS} \iff v(x_i) > 0$$

- **Volume Estimate ($\gamma$):** The volume is defined as the fraction of points that satisfy the Lyapunov condition:

$$\gamma = \frac{\text{Number of points where } v(x) > 0}{N}$$

**Normalization:** We normalize the volume of the safe set $\mathcal{X}$ to one, so $\gamma \in [0, 1]$.

# V. Experiments: Example 1 Results

- **Initial Volume:** $\gamma = 0.5428$.
- **Final Volume (Alg 1):** $\gamma = \mathbf{0.9070}$.
- **Greedy Baseline:** $\gamma = 0.7772$.
- **Observation:** $\epsilon$-greedy refinement outperforms standard greedy strategies (i.e.,$\epsilon = 0$).

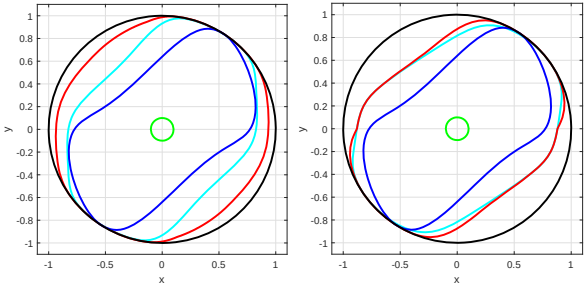



Figure: Left: Alg 1 expansion; Right: Greedy baseline.

# V. Experiments: Table I - Safety Verification Comparison

| Example | Dim ($n$) | Fossil (s) | DeepICBC (s) | **Ours (s)** |
|---|---|---|---|---|
| Ex 3 | 2 | 23.8 | 4.2 | **2.7** |
| Ex 7 | 3 | – | 28452 | **24.6** |
| Ex 8 | 4 | – | 7935 | **7.8** |
| **Ex 9** | **12** | – | – | **4.9** |

Table: Comparison showing speed and scalability (timeout after 8 hours).

Why Safety Verification Comparison?

- **Absence of Direct Competitors**
- **Framework Generality:** Our convex optimization framework is not limited to reach-avoid sets; it can be directly adapted to search for **Control Barrier Certificates (CBCs)** for safety verification.

# V. Experiments: Limitation - Curse of Dimensionality

- Method relies on SOS programming tailored to polynomial systems.
- **Combinatorial growth:** Variables scale as $\binom{n+d}{d}$.
- Restricted direct applicability to very large-scale systems.
- Despite this, it outperforms current state-of-the-art symbolic/neural methods.

# VI. Conclusion: Summary

- Established a convex framework for discrete-time CRAS.
- Theorem 1 ensures equivalence between deterministic and probabilistic sets.
- Proposition 3 reduces conservatism via RL-inspired iterative refinement.
- Outperformed symbolic and neural tools in computation time and scalability.

# Thank You!

**Questions?**

*Contact: xuebai@ios.ac.cn*