Reachability Predictive Control: A Novel Control Method for Systems with Unknown Dynamics

Taha Shafa[1]

[1]Aerospace Engineering Ph.D. Candidate
University of Illinois Urbana-Champaign

Preliminary Exam Presentation

Motivation

- Dynamic models are not always available in every situation where control is needed

Motivation

- Dynamic models are not always available in every situation where control is needed
  - Adverse event creating an abrupt change in the system dynamics

Motivation

- Dynamic models are not always available in every situation where control is needed
  - Adverse event creating an abrupt change in the system dynamics
  - Chaotic systems without existing reliable dynamic models

Motivation

- Dynamic models are not always available in every situation where control is needed
  - Adverse event creating an abrupt change in the system dynamics
  - Chaotic systems without existing reliable dynamic models
- Question: How can we optimally utilize additional knowledge to control a nonlinear system without a dynamics model?

Motivation

- Dynamic models are not always available in every situation where control is needed
  - Adverse event creating an abrupt change in the system dynamics
  - Chaotic systems without existing reliable dynamic models
- Question: How can we optimally utilize additional knowledge to control a nonlinear system without a dynamics model?
  - Onboard learning methods

Motivation

- Dynamic models are not always available in every situation where control is needed
  - Adverse event creating an abrupt change in the system dynamics
  - Chaotic systems without existing reliable dynamic models
- Question: How can we optimally utilize additional knowledge to control a nonlinear system without a dynamics model?
  - Onboard learning methods
  - Known physical laws

Motivation

- Dynamic models are not always available in every situation where control is needed
  - Adverse event creating an abrupt change in the system dynamics
  - Chaotic systems without existing reliable dynamic models
- Question: How can we optimally utilize additional knowledge to control a nonlinear system without a dynamics model?
  - Onboard learning methods
  - Known physical laws
  - Observable behaviors like previous trajectories

Existing Control Methods for Uncertain Systems

- **Robust Control:**
  - Develop control action for worst case unknown disturbances
- **Adaptive Control:**

Existing Control Methods for Uncertain Systems

- **Robust Control:**
  - Develop control action for worst case unknown disturbances
  - Requires a dynamic model and worst-case bounds on the potential disturbances
- **Adaptive Control:**

Existing Control Methods for Uncertain Systems

- **Robust Control:**
  - Develop control action for worst case unknown disturbances
  - Requires a dynamic model and worst-case bounds on the potential disturbances
  - Performance can be sluggish when disturbances bounds are too large
    - Inherently conservative
    - Prioritize safety
- **Adaptive Control:**

Existing Control Methods for Uncertain Systems

- **Robust Control:**
  - Develop control action for worst case unknown disturbances
  - Requires a dynamic model and worst-case bounds on the potential disturbances
  - Performance can be sluggish when disturbances bounds are too large
    - Inherently conservative
    - Prioritize safety
- **Adaptive Control:**
  - Control method which adapts to parametric uncertainty

Existing Control Methods for Uncertain Systems

- **Robust Control:**
  - Develop control action for worst case unknown disturbances
  - Requires a dynamic model and worst-case bounds on the potential disturbances
  - Performance can be sluggish when disturbances bounds are too large
    - Inherently conservative
    - Prioritize safety

- **Adaptive Control:**
  - Control method which adapts to parametric uncertainty
  - Unlike robust control, no a priori information on the bounds of the uncertain parameters needed
    - Control law adapts to parametric change

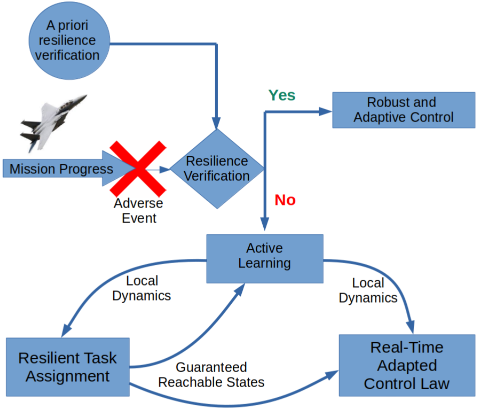
Existing Control Methods for Uncertain Systems

- **Robust Control:**
  - Develop control action for worst case unknown disturbances
  - Requires a dynamic model and worst-case bounds on the potential disturbances
  - Performance can be sluggish when disturbances bounds are too large
    - Inherently conservative
    - Prioritize safety

- **Adaptive Control:**
  - Control method which adapts to parametric uncertainty
  - Unlike robust control, no a priori information on the bounds of the uncertain parameters needed
    - Control law adapts to parametric change
  - Fails if the parametric change is too significant

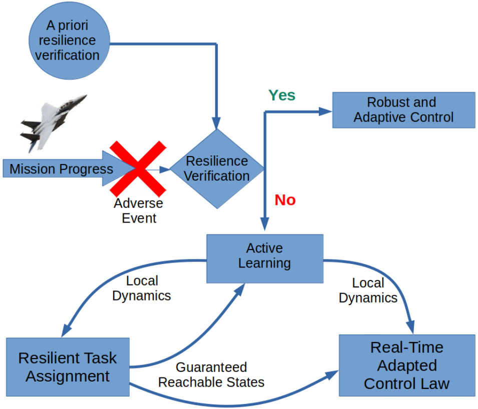
Control Pipeline

- Active learning
  - Learn local dynamics with an arbitrarily small error from test control inputs



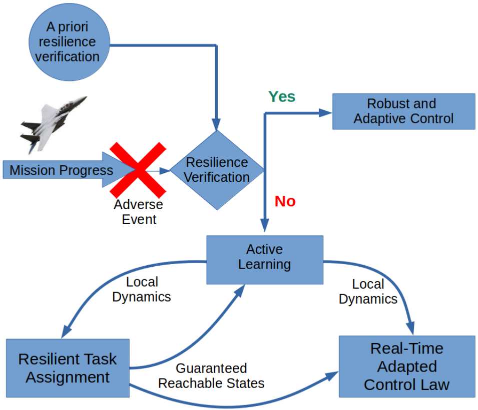
Control Pipeline

- Active learning
  - Learn local dynamics with an arbitrarily small error from test control inputs
- Resilient Task Assignment
  - Determine what you can provably achieve without knowing the true system dynamics



Control Pipeline

- Active learning
  - Learn local dynamics with an arbitrarily small error from test control inputs
- Resilient Task Assignment
  - Determine what you can provably achieve without knowing the true system dynamics
- Controller Synthesis
  - Synthesize a controller using gathered knowledge without a dynamics model



Overview

- **Resilient Task Assignment:**
  - Determine the guaranteed set of reachable states without knowledge of the system dynamics
- **Controller Synthesis:**

Overview

- **Resilient Task Assignment:**
  - Determine the guaranteed set of reachable states without knowledge of the system dynamics
  - Solve for the *Guaranteed Reachable Set* of an unknown system
    - A guaranteed underapproximation of the true reachable set of the unknown system
- **Controller Synthesis:**

Overview

- **Resilient Task Assignment:**
  - Determine the guaranteed set of reachable states without knowledge of the system dynamics
  - Solve for the *Guaranteed Reachable Set* of an unknown system
    - A guaranteed underapproximation of the true reachable set of the unknown system
- We solve for various underapproximations of the true reachable set under different assumptions and detail their advantages

- **Controller Synthesis:**

Overview

- **Resilient Task Assignment:**
  - Determine the guaranteed set of reachable states without knowledge of the system dynamics
  - Solve for the *Guaranteed Reachable Set* of an unknown system
    - A guaranteed underapproximation of the true reachable set of the unknown system
- We solve for various underapproximations of the true reachable set under different assumptions and detail their advantages

- **Controller Synthesis:**
  - Synthesize control action based on the limited knowledge gained from active learning and resilient task assignment

Overview

- **Resilient Task Assignment:**
  - Determine the guaranteed set of reachable states without knowledge of the system dynamics
  - Solve for the *Guaranteed Reachable Set* of an unknown system
    - A guaranteed underapproximation of the true reachable set of the unknown system
- We solve for various underapproximations of the true reachable set under different assumptions and detail their advantages

- **Controller Synthesis:**
  - Synthesize control action based on the limited knowledge gained from active learning and resilient task assignment
  - We consider two methods:

Overview

- **Resilient Task Assignment:**
  - Determine the guaranteed set of reachable states without knowledge of the system dynamics
  - Solve for the *Guaranteed Reachable Set* of an unknown system
    - A guaranteed underapproximation of the true reachable set of the unknown system
- We solve for various underapproximations of the true reachable set under different assumptions and detail their advantages

- **Controller Synthesis:**
  - Synthesize control action based on the limited knowledge gained from active learning and resilient task assignment
  - We consider two methods:
    - Utilize reachable sets to directly perform system identification and learn the model of our unknown system

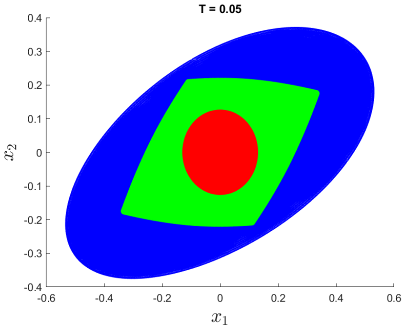
Overview

- **Resilient Task Assignment:**
  - Determine the guaranteed set of reachable states without knowledge of the system dynamics
  - Solve for the *Guaranteed Reachable Set* of an unknown system
    - A guaranteed underapproximation of the true reachable set of the unknown system
- We solve for various underapproximations of the true reachable set under different assumptions and detail their advantages

- **Controller Synthesis:**
  - Synthesize control action based on the limited knowledge gained from active learning and resilient task assignment
  - We consider two methods:
    - Utilize reachable sets to directly perform system identification and learn the model of our unknown system
    - Use a proxy system model derived during resilient task assignment to synthesize a controller for small time intervals

Guaranteed Reachability

- It is impossible to determine the exact set of reachable states without full knowledge of the system dynamics

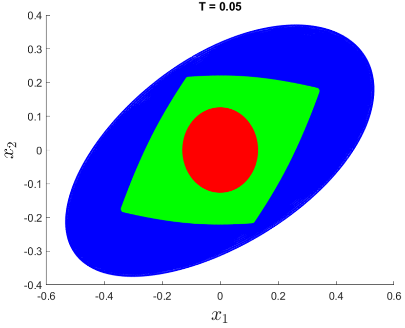


Guaranteed Reachability

- It is impossible to determine the exact set of reachable states without full knowledge of the system dynamics
- To determine what is provably possible, we want to underapproximate such a set

Guaranteed Reachable Set

A set of states that are provably achievable for a system within a given time frame



Framework for Under-Approximating the GRS

- Calculate an ordinary differential inclusion whose right-hand side is the set of all velocities that can be taken by all systems consistent with the assumed knowledge of the dynamics

Framework for Under-Approximating the GRS

- Calculate an ordinary differential inclusion whose right-hand side is the set of all velocities that can be taken by all systems consistent with the assumed knowledge of the dynamics
- Underapproximate said set with another set whose geometric properties allow for it to be represented by a proxy control system

Framework for Under-Approximating the GRS

- Calculate an ordinary differential inclusion whose right-hand side is the set of all velocities that can be taken by all systems consistent with the assumed knowledge of the dynamics
- Underapproximate said set with another set whose geometric properties allow for it to be represented by a proxy control system
  - Simple geometric properties allow for real-time implementation

Framework for Under-Approximating the GRS

- Calculate an ordinary differential inclusion whose right-hand side is the set of all velocities that can be taken by all systems consistent with the assumed knowledge of the dynamics
- Underapproximate said set with another set whose geometric properties allow for it to be represented by a proxy control system
    - Simple geometric properties allow for real-time implementation
    - The reachable set of the proxy system will underapproximate the guaranteed reachable set

Framework for Under-Approximating the GRS

- Calculate an ordinary differential inclusion whose right-hand side is the set of all velocities that can be taken by all systems consistent with the assumed knowledge of the dynamics
- Underapproximate said set with another set whose geometric properties allow for it to be represented by a proxy control system
  - Simple geometric properties allow for real-time implementation
  - The reachable set of the proxy system will underapproximate the guaranteed reachable set
- Calculate the reachable set of the proxy control system

Framework for Under-Approximating the GRS

- Calculate an ordinary differential inclusion whose right-hand side is the set of all velocities that can be taken by all systems consistent with the assumed knowledge of the dynamics
- Underapproximate said set with another set whose geometric properties allow for it to be represented by a proxy control system
  - Simple geometric properties allow for real-time implementation
  - The reachable set of the proxy system will underapproximate the guaranteed reachable set
- Calculate the reachable set of the proxy control system

Proposition

The reachable set of the proxy system is contained in the true reachable set of the unknown system for all time

Formalized Problem Statement

- We have an unknown nonlinear control-affine system of the form:
  - $\dot{x} = f(x) + G(x)\mathcal{U}, \quad x(0) = x_0$
  - $f(x) \in \mathbb{R}^n$ and $G(x) \in \mathbb{R}^{n \times m}$

Formalized Problem Statement

- We have an unknown nonlinear control-affine system of the form:
  - $\dot{x} = f(x) + G(x)\mathcal{U}, \quad x(0) = x_0$
  - $f(x) \in \mathbb{R}^n$ and $G(x) \in \mathbb{R}^{n \times m}$
- We assume knowledge of:

Formalized Problem Statement

- We have an unknown nonlinear control-affine system of the form:
  - $\dot{x} = f(x) + G(x)\mathcal{U}, \quad x(0) = x_0$
  - $f(x) \in \mathbb{R}^n$ and $G(x) \in \mathbb{R}^{n \times m}$
- We assume knowledge of:
  - The initial state x_0

Formalized Problem Statement

- We have an unknown nonlinear control-affine system of the form:
  - $\dot{x} = f(x) + G(x)\mathcal{U}, \quad x(0) = x_0$
  - $f(x) \in \mathbb{R}^n$ and $G(x) \in \mathbb{R}^{n \times m}$
- We assume knowledge of:
  - The initial state x_0
  - The input set $\mathcal{U} = \mathbb{B}^m(0; 1)$

Formalized Problem Statement

- We have an unknown nonlinear control-affine system of the form:
  - $\dot{x} = f(x) + G(x)\mathcal{U}, \quad x(0) = x_0$
  - $f(x) \in \mathbb{R}^n$ and $G(x) \in \mathbb{R}^{n \times m}$
- We assume knowledge of:
  - The initial state x_0
  - The input set $\mathcal{U} = \mathbb{B}^m(0; 1)$
  - Local dynamics $f(x_0)$ and $G(x_0)$
    - Learned within an arbitrarily small error from test control inputs

Formalized Problem Statement

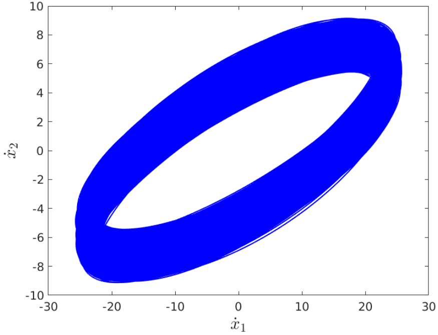
- We have an unknown nonlinear control-affine system of the form:
  - $\dot{x} = f(x) + G(x)\mathcal{U}, \quad x(0) = x_0$
  - $f(x) \in \mathbb{R}^n$ and $G(x) \in \mathbb{R}^{n\times m}$
- We assume knowledge of:
  - The initial state x_0
  - The input set $\mathcal{U} = \mathbb{B}^m(0; 1)$
  - Local dynamics $f(x_0)$ and $G(x_0)$
    - Learned within an arbitrarily small error from test control inputs
  - The maximum growth rate of dynamics given by Lipschitz bounds L_f and L_G
    - Determined from known physical laws
    - Uncertainty quantification

Problem Statement

Determine or underapproximate the guaranteed reachable set

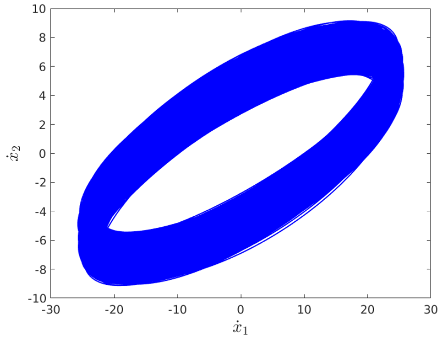
Guaranteed Velocity Set

- We are left with infinitely many candidate systems consistent with our assumptions



Guaranteed Velocity Set

- We are left with infinitely many candidate systems consistent with our assumptions
- Let us denote $\mathcal{D}_{con}$ as the set of all f, G consistent with our assumptions



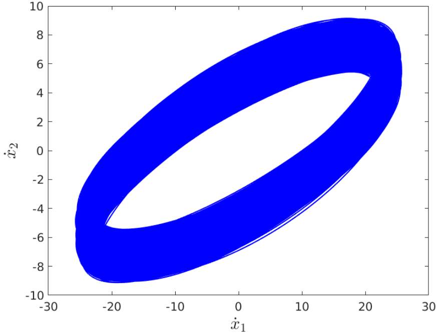
Guaranteed Velocity Set

- We are left with infinitely many candidate systems consistent with our assumptions
- Let us denote $\mathcal{D}_{con}$ as the set of all f, G consistent with our assumptions
- We introduce the following ODI:
  - $\dot{x} \in \mathcal{V}_x = f(x) + G(x)\mathcal{U}, \quad x(0) = x_0$

Guaranteed Velocity Set

The intersection of the set of all velocities whose dynamics are consistent with our assumptions:

$$\mathcal{V}_x^{\mathcal{G}} = \cap_{(f,G)\in\mathcal{D}_{con}} f(x) + G(x)\mathcal{U} \subset \mathcal{V}_x$$



Derived Ball Underapproximation

Ball Underapproximation

Let $\mathcal{U}$, L_f, and L_G be defined as above. Let $x \in \mathbb{R}^n$ satisfy $(L_f + L_G)\|x\| < \|G^\dagger(x_0)\|^{-1}$. Define

$$\bar{\mathcal{V}}_x^{\mathcal{G}} = \mathbb{B}^n(f(x_0); \|G^\dagger(x_0)\|^{-1} - (L_f + L_G)\|x\|) \cap \mathrm{Im}(G(x_0)).$$

Then, $\bar{\mathcal{V}}_x^{\mathcal{G}} \subseteq \mathcal{V}_x^{\mathcal{G}}$.

Derived Convex Underapproximation

Advanced Convex Underapproximation

Let $\mathcal{U}$, L_f, and L_G be defined as above. Let $\mu = 1$ if $\mathrm{rank}(G(x_0)) = m = n$, $\mu = \sqrt{2}$ if $\mathrm{rank}(G(x_0)) = \min(m, n)$ and $m \neq n$, $\mu = \frac{1+\sqrt{5}}{2}$ if $\mathrm{rank}(G(x_0)) < \min(m, n)$, and let x satisfy $(L_f + L_G)\|x\| \leq \|G^\dagger(x_0)\|^{-1}$. If

$$\bar{\bar{\mathcal{V}}}_x^{\mathcal{G}} = \{f(x_0) + kd \mid \|d\| = 1, d \in \mathrm{Im}(R), 0 \leq k \leq K(d)\}$$

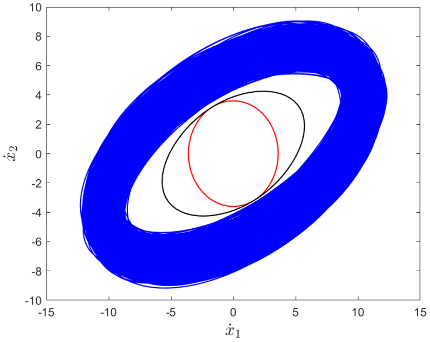
$$\text{s.t.} \quad K(d) = \frac{\|G^\dagger(x_0)\|^{-1} - (L_f + L_G)\|x\|}{\|G^\dagger(x_0)d\|(\|G(x_0)^\dagger\|^{-1} - L_G\|x\|) + \mu\|G(x_0)^\dagger\|L_G\|x\|},$$

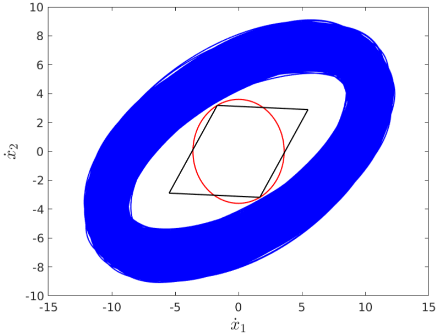
then $\bar{\bar{\mathcal{V}}}_x^{\mathcal{G}} \subseteq \mathcal{V}_x^{\mathcal{G}}$.

Corollary

For invertible matrices $G(x_0)$, $\bar{\mathcal{V}}_x^{\mathcal{G}} \subseteq \bar{\bar{\mathcal{V}}}_x^{\mathcal{G}}$.

Visual Interpretation of GVS Underapproximations





Interpretation of Ball ODI as a Control System

- Let $\mathcal{R}^{\mathcal{G}}(T, x_0)$ be defined as the guaranteed reachable set of our unknown system.

Interpretation of Ball ODI as a Control System

- Let $\mathcal{R}^{\mathcal{G}}(T, x_0)$ be defined as the guaranteed reachable set of our unknown system.
- We interpret the following ODI:

$$\dot{x} \in \bar{\mathcal{V}}_x^{\mathcal{G}}, \quad x(0) = x_0,$$

as a control system

Interpretation of Ball ODI as a Control System

- Let $\mathcal{R}^{\mathcal{G}}(T, x_0)$ be defined as the guaranteed reachable set of our unknown system.
- We interpret the following ODI:

$$\dot{x} \in \bar{\mathcal{V}}_x^{\mathcal{G}}, \quad x(0) = x_0,$$

  as a control system
- The reachable set of such a control system contained in the guaranteed reachable set

Theorem

Let us consider the control system

$$\dot{x} = a + g(\|x\|)\bar{u}, \quad x(0) = x_0,$$

on $\{x \mid \|x\| \le \|G^\dagger(x_0)\|/(L_f + L_G)\}$, with $a = f(x_0)$, $\bar{u} \in \mathbb{B}^n(0;1) \cap \mathrm{Im}(G(x_0))$ and where $g(s) = \|G^\dagger(s_0)\|^{-1} - (L_f + L_G)s$ if $s \le \|G^\dagger(s_0)\|^{-1}/(L_G + L_f)$. If $\overline{\mathcal{R}}(T, x_0)$ is the reachable set of the control system above, then $\overline{\mathcal{R}}(T, x_0) \subseteq \mathcal{R}^{\mathcal{G}}(T, x_0)$.

Interpretation of the Polygon ODI as a Control System

- We interpret the following ODI:

$$\dot{x} \in P(\mathcal{S}(x)), \quad x(0) = x_0,$$

as a control system.

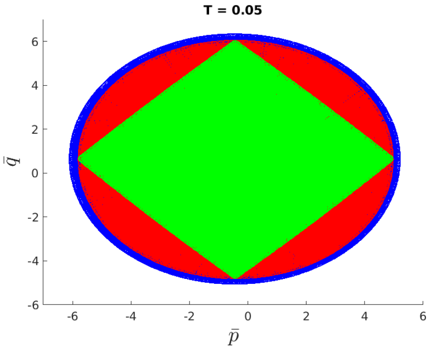
Theorem

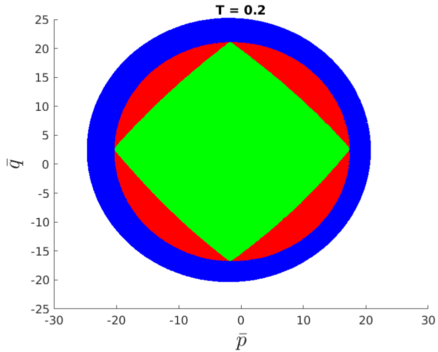
Let $s \in \mathbb{R}$ and $g(s) = \|G(s_0)^{\dagger}\|^{-1} - (L_G + L_f)s$, $\alpha(s) = \|G(s_0)^{\dagger}\|^{-1} - L_G s$, $\beta(s) = \mu\|G(s_0)^{\dagger}\|L_G s$ with μ as defined in Theorem 2. Let $U\Sigma V^T$ be the singular value decomposition of $G(x_0)$ where $U = [\eta_1, ..., \eta_n]$. Let $r = \mathrm{rank}(G(x_0))$; we define $\Lambda(s) \in \mathbb{R}^{n\times m}$ such that $\mathrm{diag}(\Lambda(s)) = \max\{\frac{g(s)}{\alpha(s)\|G(x_0)^{\dagger}\eta_i\|+\beta(s)}, g(s), 0\}$ and $\lambda_{ij}(s) = 0$ elsewhere. The reachable set of $\dot{x} \in P(\mathcal{S}(x))$ equals the reachable set of the control system

$$\dot{x} = a + U\Lambda(\|x\|)u, x(0) = x_0,$$

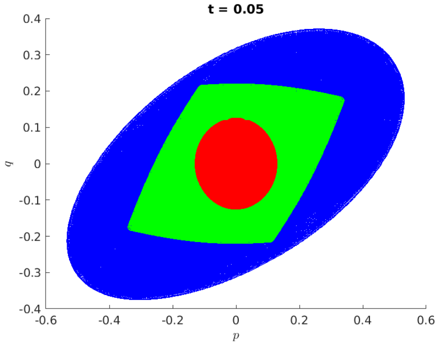
on $\{x \mid \|x\| \leq \|G(x_0)^{\dagger}\|/(L_f + L_G)\}$, with $a = f(x_0)$ and $u \in \{u \mid \|u\|_1 \leq 1\}$. If $\hat{\mathcal{R}}(T, x_0)$ denotes the reachable set of the system above, then $\hat{\mathcal{R}}(T, x_0) \subseteq \mathcal{R}^{\mathcal{G}}(T, x_0)$.

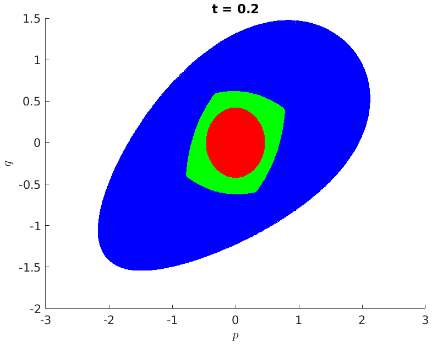
Visual Interpretation of the Results





Visual Interpretation of the Results





Incorporating Additional Knowledge

- We want to determine how we can incorporate additional knowledge to improve our underapproximations
  - Underapproximations should better reflect the complex shape of the interior of the GVS

Incorporating Additional Knowledge

- We want to determine how we can incorporate additional knowledge to improve our underapproximations
  - Underapproximations should better reflect the complex shape of the interior of the GVS
- Assume knowledge of $f(x)$ but $G(x)$ remains unknown

Incorporating Additional Knowledge

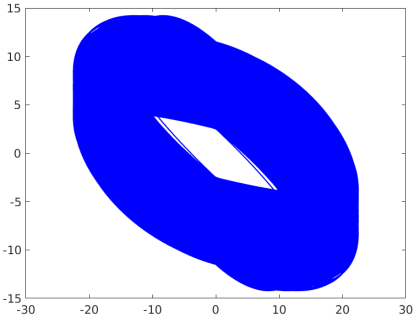
- We want to determine how we can incorporate additional knowledge to improve our underapproximations
  - Underapproximations should better reflect the complex shape of the interior of the GVS
- Assume knowledge of $f(x)$ but $G(x)$ remains unknown
- Assume knowledge of $G(x_0)$ with element-wise perturbations on elements of $G(x_0)$
  - $G(x)$ lives in the space of all viable known elementwise perturbations of $G(x_0)$

Incorporating Additional Knowledge

- We want to determine how we can incorporate additional knowledge to improve our underapproximations
  - Underapproximations should better reflect the complex shape of the interior of the GVS
- Assume knowledge of $f(x)$ but $G(x)$ remains unknown
- Assume knowledge of $G(x_0)$ with element-wise perturbations on elements of $G(x_0)$
  - $G(x)$ lives in the space of all viable known elementwise perturbations of $G(x_0)$
- The problem statement remains consistent with previous work

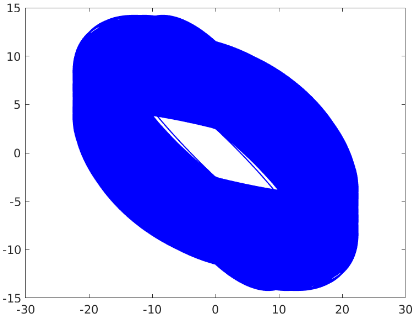
New Method for Underapproximating the GVS

- We are currently trying to solve an optimization problem with infinitely many constraints



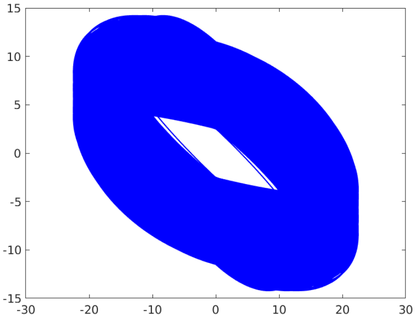
New Method for Underapproximating the GVS

- We are currently trying to solve an optimization problem with infinitely many constraints
    - Simplify the problem to an optimization problem with finitely many constraints



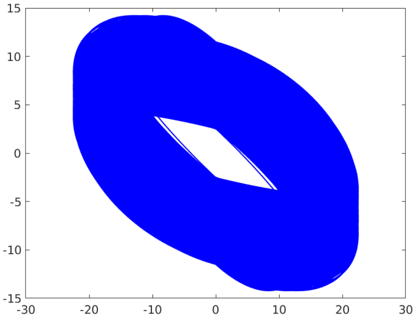
New Method for Underapproximating the GVS

- We are currently trying to solve an optimization problem with infinitely many constraints
  - Simplify the problem to an optimization problem with finitely many constraints
  - Use existing optimization methods to inscribe an ellipse of maximal volume inside the GVS



New Method for Underapproximating the GVS

- We are currently trying to solve an optimization problem with infinitely many constraints
  - Simplify the problem to an optimization problem with finitely many constraints
  - Use existing optimization methods to inscribe an ellipse of maximal volume inside the GVS



Finite Perturbation Theorem

Within a domain largely consistent with previous derivations, we can reduce the infinite constraint optimization problem to one with finitely many constraints

Finite Perturbation Theorem

- Let δ be the maximum perturbation magnitude for an element of $G(x_0)$
- If $\exists u_1, u_2 \in \mathcal{U}$ s.t.

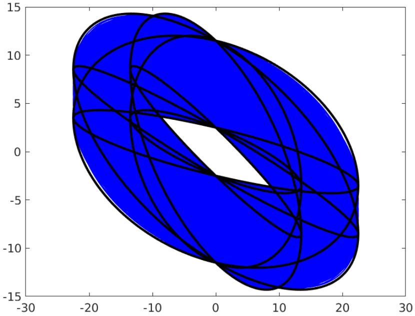
$$v = \left(G(x_0) + \begin{bmatrix} +\delta & 0 \\ 0 & 0 \end{bmatrix}\right) u_1,$$

$$v = \left(G(x_0) + \begin{bmatrix} -\delta & 0 \\ 0 & 0 \end{bmatrix}\right) u_2,$$

then $\forall\, \alpha \in [-1, 1]$, $\exists u^* \in \mathcal{U}$ s.t.

$$v = \left(G(x_0) + \begin{bmatrix} \alpha\delta & 0 \\ 0 & 0 \end{bmatrix}\right) u^*.$$

- Any vector contained in the 2^{n^2} black edges are also contained in the blue

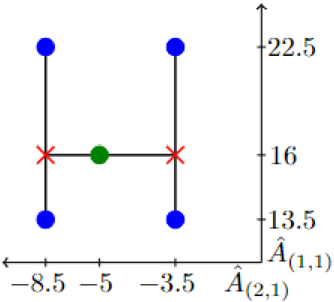


Finite Perturbation Theorem - Example

- Let $\Delta = \begin{bmatrix} 4.5 & 0 \\ 2.5 & 4.5 \end{bmatrix}$ be the matrix of maximal element-wise perturbation magnitudes
- Let $A = \begin{bmatrix} 18 & 0 \\ -6 & 7 \end{bmatrix}$
- Let $C_\Delta(A)$

$$= \left\{ \begin{bmatrix} 18 + \alpha_1 4.5 & 0 \\ -6 + \alpha_2 2.5 & 7 + \alpha_3 4.5 \end{bmatrix} \;\middle|\; \alpha_i \in [-1, 1] \right\}$$

- $\hat{A} = \begin{bmatrix} 16 & 0 \\ -5 & 5.5 \end{bmatrix} \in C_\Delta(A)$



Maximally Inscribed Ellipsoid

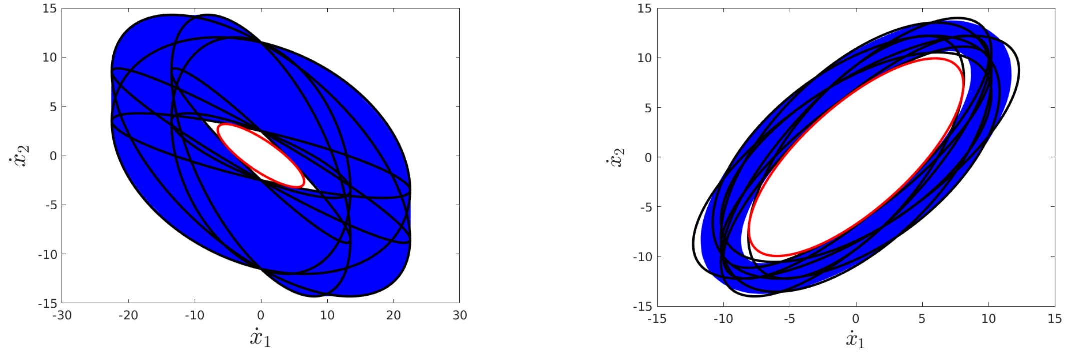
Theorem

Let A and Δ_k be the nominal and perturbation matrices respectively. Let $U\Sigma V^T$ be the singular value decomposition of A and let $\Sigma_k = U^T(A+\Delta_k)V$ and $A_k = (\Sigma_k^{-1})^T\Sigma_k^{-1}$. Let $\mathcal{E}_k = (A+\Delta_k)\mathcal{U}$. Then, an ellipsoid $\mathcal{E}$ of maximal volume such that $\mathcal{E} \subseteq \bigcap_k \mathcal{E}_k$ is given by $\mathcal{E} = UBV^T\mathcal{U}$ where B is the solution to

$$\begin{aligned} \underset{B\in\mathcal{S}^n_{++},\lambda_1,\ldots,\lambda_{2^{n^2}}\in\mathbb{R}}{\text{minimize}} \quad & \log\det B^{-1} \\ \text{subject to} \quad & \begin{bmatrix} -\lambda_k+1 & 0 & 0 \\ 0 & \lambda_k I_n & B \\ 0 & B & A_k^{-1} \end{bmatrix} \geq 0 \\ & \text{for all } k \in [2^{n^2}]. \end{aligned}$$

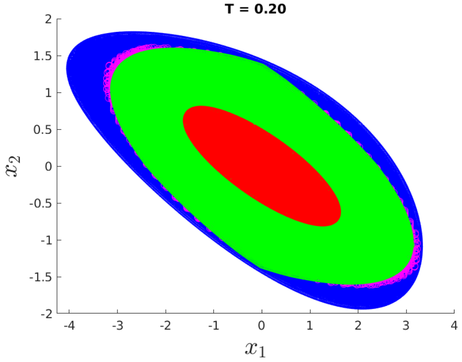
- The 2^{n^2} constraints can potentially be reduced with further analysis

Illustrated Results



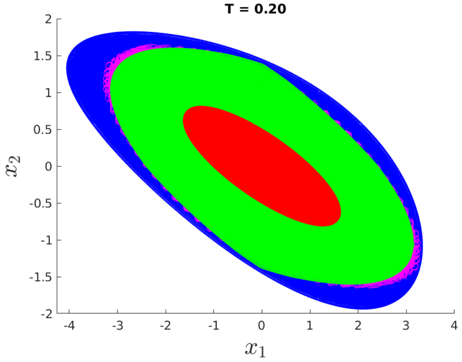
Analysis of the Results

- We are able to characterize the GVS as an intersection of finitely many ellipsoids



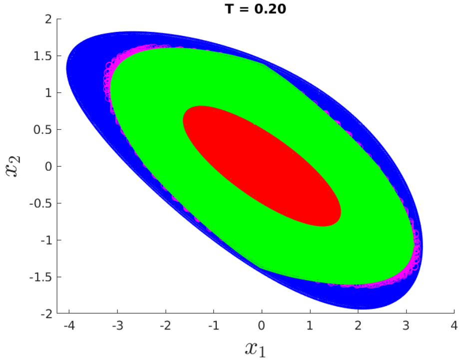
Analysis of the Results

- We are able to characterize the GVS as an intersection of finitely many ellipsoids
  - We can use this characterization to determine an optimally inscribed ellipse as our underapproximated GVS



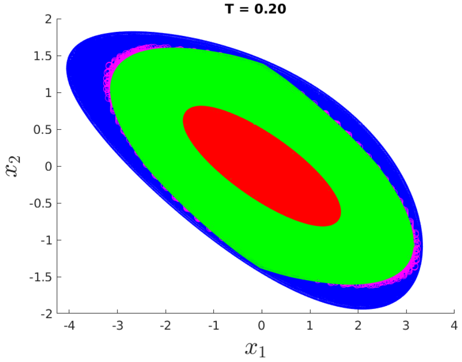
Analysis of the Results

- We are able to characterize the GVS as an intersection of finitely many ellipsoids
  - We can use this characterization to determine an optimally inscribed ellipse as our underapproximated GVS
- If the perturbation matrix is a function of x, this requires solving an optimization problem at every time step
  - Not a realistic method of underapproximation for real-time implementation

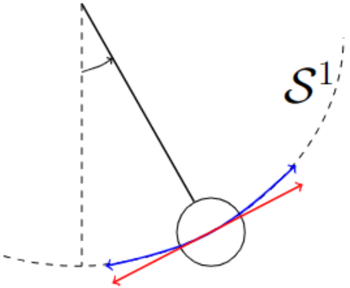


Analysis of the Results

- We are able to characterize the GVS as an intersection of finitely many ellipsoids
  - We can use this characterization to determine an optimally inscribed ellipse as our underapproximated GVS
- If the perturbation matrix is a function of x, this requires solving an optimization problem at every time step
  - Not a realistic method of underapproximation for real-time implementation
- We can take the worst-case perturbation for all x within a domain and solve one optimization problem
  - Real time implementable

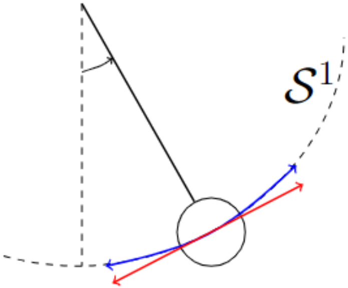


- Control systems can exist on manifolds outside of $\mathbb{R}^n$



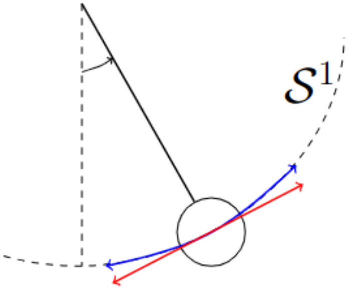
Generalization of Results to Complete Manifolds

- Control systems can exist on manifolds outside of $\mathbb{R}^n$
  - Pendulum

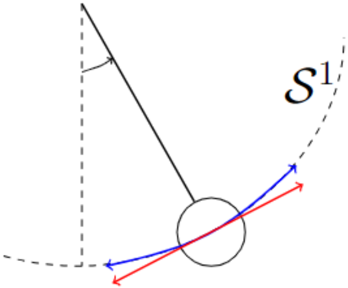


Generalization of Results to Complete Manifolds

- Control systems can exist on manifolds outside of $\mathbb{R}^n$
  - Pendulum
  - Satellite in Orbit

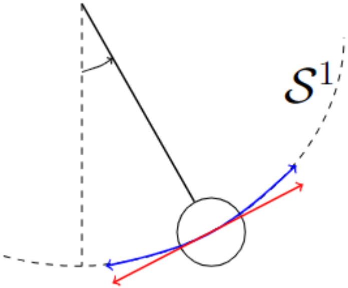


- Control systems can exist on manifolds outside of $\mathbb{R}^n$
  - Pendulum
  - Satellite in Orbit
  - $SO(2)$, $SO(3)$



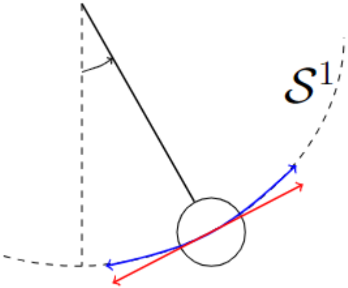
Generalization of Results to Complete Manifolds

- Control systems can exist on manifolds outside of $\mathbb{R}^n$
  - Pendulum
  - Satellite in Orbit
  - $SO(2)$, $SO(3)$
- Now that we have maximized our underapproximation, we want to generalize results to a larger class of systems



Generalization of Results to Complete Manifolds

- Control systems can exist on manifolds outside of $\mathbb{R}^n$
  - Pendulum
  - Satellite in Orbit
  - $SO(2)$, $SO(3)$
- Now that we have maximized our underapproximation, we want to generalize results to a larger class of systems
- We relax assumptions to assume the control system exists on a complete Riemannian manifold



Formal Problem Statement

- We have an unknown nonlinear control-affine system operating on a manifold M of the form:
  - $\dot{x} = f(x) + G(x)\mathcal{U} = f(x) + \sum_l g_l(x)u^l, \quad x(0) = x_0$
  - $f(x) \in M$ and $g_l(x) \in M$
- We assume knowledge of:
  - The initial state x_0
  - The input set $\mathcal{U} = \mathbb{B}^m(0;1)$
  - Local dynamics $f(x_0)$ and $G(x_0)$
    - Learned within an arbitrarily small error from test control inputs
  - The maximum growth rate of dynamics given by *Riemannian Lipschitz* bounds L_f and L_G
    - Determined from known physical laws
    - Uncertainty quantification

Problem Statement

Determine or underapproximate the guaranteed reachable set

Preliminaries for Control Systems on Manifolds

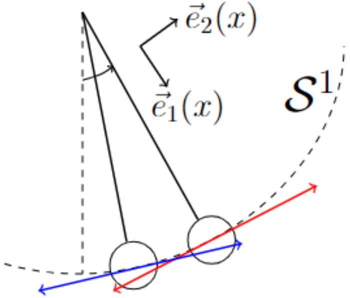
- Operations between vector spaces require connections in the geometric sense

Riemannian Lipschitz

Let V be a continuous vector field on M and τ be the parallel transport. Then L is the *classical Lipschitz constant* on V if

$$L = \sup_{\gamma} \frac{|\tau_\gamma V(\gamma(0)) - V(\gamma(1))|_{h_x}}{\mathrm{Length}(\gamma)}$$

where $\gamma : [0, 1] \to M$ varies over all $\mathcal{C}^1$-paths and τ_γ is shorthand for the parallel transport along the curve γ from $\gamma(0)$ to $\gamma(1)$.



Preliminaries for Control Systems on Manifolds

- The vector space $T_x M$ varies as $x \in M$ varies in general

Preliminaries for Control Systems on Manifolds

- The vector space $T_x M$ varies as $x \in M$ varies in general
- To perform operations between vectors in different tangent spaces, we need to use connections and parallel transport

Preliminaries for Control Systems on Manifolds

- The vector space $T_x M$ varies as $x \in M$ varies in general
- To perform operations between vectors in different tangent spaces, we need to use connections and parallel transport
- The Lipschitz constant may seem difficult to calculate because it depends on varying over all $\mathcal{C}^1$-paths on M.
    - We perform calculations on a compact subset of the manifold

Preliminaries for Control Systems on Manifolds

- The vector space $T_x M$ varies as $x \in M$ varies in general
- To perform operations between vectors in different tangent spaces, we need to use connections and parallel transport
- The Lipschitz constant may seem difficult to calculate because it depends on varying over all $\mathcal{C}^1$-paths on M.
  - We perform calculations on a compact subset of the manifold

Lemma

The supremum of the Lipschitz constant on a compact set can be attained if we vary only over geodesics

Preliminaries for Control Systems on Manifolds

- The vector space $T_x M$ varies as $x \in M$ varies in general
- To perform operations between vectors in different tangent spaces, we need to use connections and parallel transport
- The Lipschitz constant may seem difficult to calculate because it depends on varying over all $\mathcal{C}^1$-paths on M.
  - We perform calculations on a compact subset of the manifold

Lemma

The supremum of the Lipschitz constant on a compact set can be attained if we vary only over geodesics

- Given an appropriate neighborhood, we need only consider one geodesic path
  - Knowledge of Lipschitz constant for general manifolds as reasonable as in the Euclidean case

Guaranteed Underapproximation of GVS

Riemannian Ball Underapproximation

Let $f(x_0)$, $G(x_0)$, L_f, L_G, H_x, Γ_{ij}^k, and g_l^Γ for $l \in [m]$ be defined as above. Let $\gamma : [0,1] \to M$ define a geodesic curve from x_0 to x. Let $\tilde{\tau}$ define the parallel transport using the flat connection. If

$$\overline{\mathcal{V}}_x^{\mathcal{G}} = \mathbb{B}^n \left(\tilde{\tau}_{x_0}^x f(x_0); \alpha(x_0, x)\right) \cap \mathrm{Im}(\tilde{\tau}_{x_0}^x G(x_0))$$

where $\overline{\mathcal{V}}_x^{\mathcal{G}} \in T_x M$, and

$$\begin{gathered} \alpha(x_0, x) = \|\tilde{\tau}_{x_0}^x G^\dagger(x_0)\|^{-1} - \\ (\|H_x^{-1}\|\|H_x\|)^{\frac{1}{2}} \Big(\|H_x\|^{\frac{1}{2}} \left\| \begin{bmatrix} g_1^\Gamma & \dots & g_m^\Gamma \end{bmatrix} \right\| + \\ \left\| \sum_{i,j,k} \dot{\gamma}^i \Gamma_{ij}^k f^j(x_0) \vec{e}_k \right\| + \left(L_g + \|H_x\|^{-\frac{1}{2}} L_f \right) d(x_0, x) \Big), \end{gathered} \tag{1}$$

then $\overline{\mathcal{V}}_x^{\mathcal{G}} \subseteq \mathcal{V}_x^{\mathcal{G}}$.

Calculate the GRS on a Manifold

- We interpret the ODI

$$\dot{x} \in \overline{\mathcal{V}}^{\mathcal{G}}_x$$

  as a control system

Calculate the GRS on a Manifold

- We interpret the ODI
$$\dot{x} \in \overline{\mathcal{V}}_x^{\mathcal{G}}$$
as a control system

Theorem

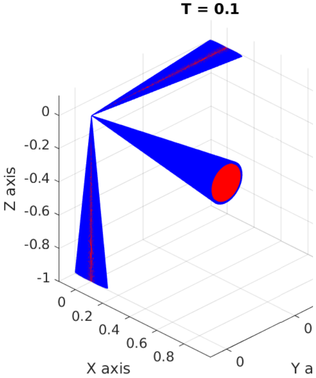
Let $\overline{\mathcal{R}}(T, x_0)$ be defined as the reachable set of

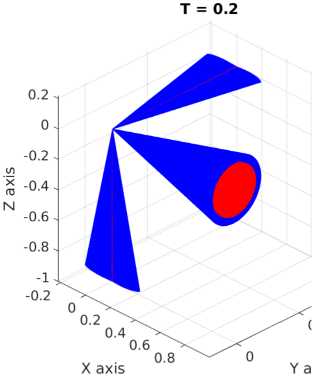
$$\dot{x} = a + g(x_0, x)u, \quad x(0) = x_0,$$

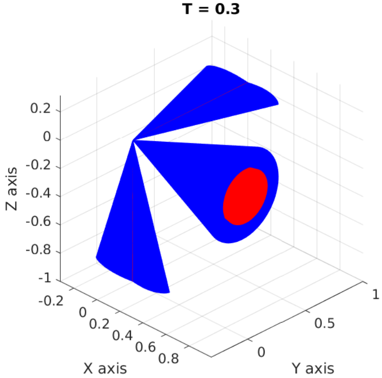
on $\{x \mid d(x_0, x) \leq \overline{d}(x_0, x)\}$, with $a = \tilde{\tau}_{x_0}^{x} f(x_0)$, $u \in \mathbb{B}^n(0; 1)$, and $g(s_0, s) = \alpha(x_0, x)$ if $d(s_0, s) \leq \overline{d}(x_0, x)$. Then $\overline{\mathcal{R}}(T, x_0) \subset \mathcal{R}^{\mathcal{G}}(T, x_0)$.

- The domain $\overline{d}(x_0, x)$ is explicitly defined in the literature

Illustrated Results







Analysis of Results and Next Steps

- We have generalized the capabilities of determining what an unknown nonlinear system can *provably* achieve to systems operating on any complete Riemannian manifold

Analysis of Results and Next Steps

- We have generalized the capabilities of determining what an unknown nonlinear system can *provably* achieve to systems operating on any complete Riemannian manifold
- The results focus on generalizing results for the ball underapproximation

Analysis of Results and Next Steps

- We have generalized the capabilities of determining what an unknown nonlinear system can *provably* achieve to systems operating on any complete Riemannian manifold
- The results focus on generalizing results for the ball underapproximation
  - Generalizations for other derived underapproximations can be calculated in a similar manner

Analysis of Results and Next Steps

- We have generalized the capabilities of determining what an unknown nonlinear system can *provably* achieve to systems operating on any complete Riemannian manifold
- The results focus on generalizing results for the ball underapproximation
  - Generalizations for other derived underapproximations can be calculated in a similar manner
- We have now derived underapproximations aimed at the following:

Analysis of Results and Next Steps

- We have generalized the capabilities of determining what an unknown nonlinear system can *provably* achieve to systems operating on any complete Riemannian manifold
- The results focus on generalizing results for the ball underapproximation
    - Generalizations for other derived underapproximations can be calculated in a similar manner
- We have now derived underapproximations aimed at the following:
    - Containing simple geometric properties that allow for real-time implementation

Analysis of Results and Next Steps

- We have generalized the capabilities of determining what an unknown nonlinear system can *provably* achieve to systems operating on any complete Riemannian manifold
- The results focus on generalizing results for the ball underapproximation
  - Generalizations for other derived underapproximations can be calculated in a similar manner
- We have now derived underapproximations aimed at the following:
  - Containing simple geometric properties that allow for real-time implementation
  - Producing guaranteed reachable sets through optimization methods to calculate the maximal set of reachable states

Analysis of Results and Next Steps

- We have generalized the capabilities of determining what an unknown nonlinear system can *provably* achieve to systems operating on any complete Riemannian manifold
- The results focus on generalizing results for the ball underapproximation
  - Generalizations for other derived underapproximations can be calculated in a similar manner
- We have now derived underapproximations aimed at the following:
  - Containing simple geometric properties that allow for real-time implementation
  - Producing guaranteed reachable sets through optimization methods to calculate the maximal set of reachable states
  - Generalizing the underapproximations to manifolds to incorporate a larger class of systems

Analysis of Results and Next Steps

- We have generalized the capabilities of determining what an unknown nonlinear system can *provably* achieve to systems operating on any complete Riemannian manifold
- The results focus on generalizing results for the ball underapproximation
  - Generalizations for other derived underapproximations can be calculated in a similar manner
- We have now derived underapproximations aimed at the following:
  - Containing simple geometric properties that allow for real-time implementation
  - Producing guaranteed reachable sets through optimization methods to calculate the maximal set of reachable states
  - Generalizing the underapproximations to manifolds to incorporate a larger class of systems
- We now want to determine how we can use this information to synthesize control action for unknown nonlinear systems

System Identification

- Create a mapping from input trajectories to observed output trajectories
  - Recursive least squares approach
  - Neural networks

System Identification

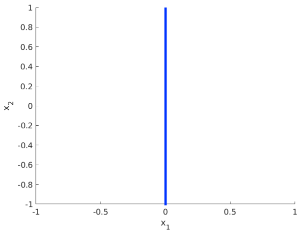
- Create a mapping from input trajectories to observed output trajectories
  - Recursive least squares approach
  - Neural networks
- Requires knowledge of individual trajectories and control of actuators

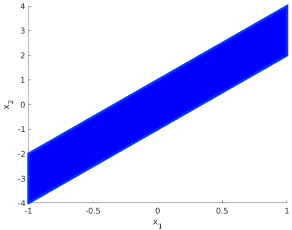
System Identification

- Create a mapping from input trajectories to observed output trajectories
  - Recursive least squares approach
  - Neural networks
- Requires knowledge of individual trajectories and control of actuators
- What if the only information one had access to is the reachable sets?

Reachable Sets

- We have the system dynamics
  - $x[i+1] = Ax[i] + bu[i], \quad x[0] = 0$
- For $i \in \mathbb{Z}_{\geq 0}$, the (forward) reachable set for the system at time i is
  - $\mathcal{R}(i, x[0]) = \{\phi_u(i; x[0]) \mid u : \mathbb{Z}_{\geq 0} \to \mathcal{U}\}$
- Defines the set of states which are reachable at time i for the system using all $u \in \mathcal{U}$
- Calculated through Minkowski sums

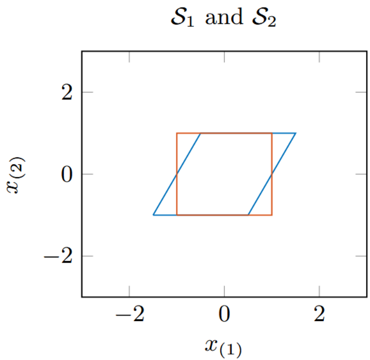


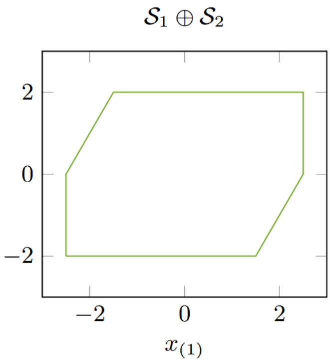


Minkowski Sum

- Given two sets $\mathcal{S}_1, \mathcal{S}_2 \in \mathbb{R}^n$ we denote

$$\mathcal{S}_1 \oplus \mathcal{S}_2 = \{s_1 + s_2 \mid s_1 \in \mathcal{S}_1,\ s_2 \in \mathcal{S}_2\}$$





Technical Assumptions

- Uniquely determine system model under the following assumptions:
  - Single-Input System
  - Generic Properties
  - Discrete Linear System
    - $x[i+1] = Ax[i] + bu[i], \quad x[0] = 0, \quad u \in \mathcal{U}$
    - Consecutive unit-length time steps
  - Fully Controllable System
  - The input set $\mathcal{U}$ is completely known

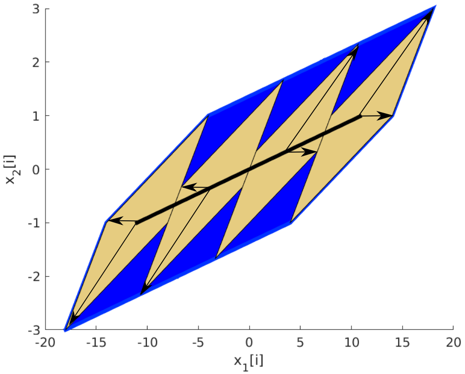
How can we solve for A and b given the assumptions above?

Reachable Sets as Minkowski Sums

- We have the following system:
  - $x[i+1] = Ax[i] + bu[i], \quad x[0] = 0, \quad u \in \mathcal{U}$
- $\mathcal{R}(i,0) =$ the reachable set of the system above at time i
- $\mathcal{R}(1,0) = b\mathcal{U}$
- $x[i] = A^i x[0] + A^{i-1} bu[0] + \ldots + bu[i-1]$
- This implies:
  - $\mathcal{R}(i,0) = A^{i-1} b\mathcal{U} \oplus \ldots \oplus b\mathcal{U}$
  - $\mathcal{R}(i,0) = A^{i-1} b\mathcal{U} \oplus \mathcal{R}(i-1,0)$
- Therefore:
  - $A^{i-1} b\mathcal{U} = \mathcal{R}(i,0) \ominus \mathcal{R}(i-1,0)$

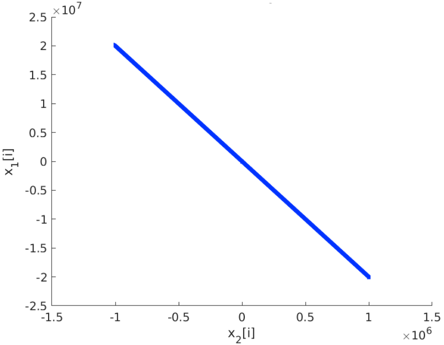
Minkowski Difference

- Given two sets $\mathcal{A}, \mathcal{B} \in \mathbb{R}^n$
  - $\mathcal{A} \ominus \mathcal{B} = \{c \in \mathbb{R}^n \mid c \oplus \mathcal{B} \subseteq \mathcal{A}\}$
- Let $v^{(i)} \in \mathcal{V}$ be the vertices of $\mathcal{R}(i-1,0)$
  - $\mathcal{R}(i,0) \ominus \mathcal{R}(i-1,0) = \bigcap_{v^{(i)} \in \mathcal{V}} (\mathcal{R}(i,0) - v^{(i)})$



Solving for Dynamics

- Solving for the vector b with knowledge of $\mathcal{U}$ is trivial
  - $\mathcal{R}(1,0) = b\mathcal{U}$
- Similarly we can solve for $A^i b$ with knowledge of $\mathcal{U}$
  - $A^{i-1}b\mathcal{U} = \mathcal{R}(i,0) \ominus \mathcal{R}(i-1,0)$
- If we have a controllable system
  - $C_{A,b} = \begin{bmatrix} b & Ab & \dots & A^{i-1}b \end{bmatrix}$
  - $C_{A,b}$ is invertible
  - $A = AC_{A,b}C_{A,b}^{-1}$



Uniqueness

Can we determine if the calculated dynamics A and b are unique?

- In general cases no
  - $A = I, \quad b = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$
  - $A = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}, \quad b = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$

Uniqueness

Can we determine if the calculated dynamics A and b are unique?

- In general cases no
  - $A = I, \quad b = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$
  - $A = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}, \quad b = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$
- Yes under the following conditions:

Uniqueness

Can we determine if the calculated dynamics A and b are unique?

- In general cases no
  - $A = I, \quad b = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$
  - $A = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}, \quad b = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$
- Yes under the following conditions:
  - Generic system properties
    - A is invertible
    - A is diagonalizable
    - A contains distinct eigenvalues
    - First element of right eigenvectors of A are nonzero

Uniqueness

Can we determine if the calculated dynamics A and b are unique?

- In general cases no
  - $A = I, \quad b = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$
  - $A = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}, \quad b = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$
- Yes under the following conditions:
  - Generic system properties
    - A is invertible
    - A is diagonalizable
    - A contains distinct eigenvalues
    - First element of right eigenvectors of A are nonzero
  - Any generic system with an input set $\mathcal{U}$ asymmetric around the origin

Uniqueness

Can we determine if the calculated dynamics A and b are unique?

- In general cases no
  - $A = I, \quad b = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$
  - $A = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}, \quad b = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$
- Yes under the following conditions:
  - Generic system properties
    - A is invertible
    - A is diagonalizable
    - A contains distinct eigenvalues
    - First element of right eigenvectors of A are nonzero
  - Any generic system with an input set $\mathcal{U}$ asymmetric around the origin
  - Generic 2D system with an input set $\mathcal{U}$ symmetric around the origin

Analytical Results for Uniqueness

Theorem

Under generic assumptions, we can **uniquely** identify the unknown dynamics of a discrete linear system with an input set asymmetric around the origin using $n + 1$ reachable sets for unit time intervals

Remark

The theorem above holds for systems with symmetric input sets of dimension 2

Bandpass Filter Circuit Example

- Fourth-order band-pass circuit
  - Controllable canonical representation

$$x[i+1] = Ax[i] + bv_c[i] = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -a_0 & -a_1 & -a_2 & -a_3 \end{bmatrix} x[i] + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} v_c[i]$$

- $v_c[i] \in [0,1]$ for all $i \in \mathbb{Z}_{\geq 0}$
- If $a_0 \neq 0$ then assumptions are likely satisfied
  - A invertible
  - A diagonalizable
- Want to recover the true parameters
  - $a_0 = 3,\ a_1 = 2,\ a_2 = 3,\ a_3 = 6$

Bandpass Filter Circuit Example

$$\mathcal{R}(1,0) = \mathrm{conv}\left(\begin{bmatrix}0\\0\\0\\0\end{bmatrix}, \begin{bmatrix}0\\0\\0\\1\end{bmatrix}\right), \mathcal{R}(2,0) = \mathrm{conv}\left(\begin{bmatrix}0\\0\\1\\-5\end{bmatrix}, \begin{bmatrix}0\\0\\1\\-6\end{bmatrix}, \begin{bmatrix}0\\0\\0\\0\end{bmatrix}, \begin{bmatrix}0\\0\\0\\1\end{bmatrix}\right)$$

$$\mathcal{R}(3,0) = \mathrm{conv}\left(\begin{bmatrix}0\\0.86\\-6.02\\33.00\end{bmatrix}, \begin{bmatrix}0\\-0.14\\0.98\\-6.00\end{bmatrix}, \begin{bmatrix}0\\-0.14\\0.98\\-5.00\end{bmatrix}, \begin{bmatrix}0\\0.86\\-6.02\\34.00\end{bmatrix}\right)$$

$$\mathcal{R}(4,0) = \mathrm{conv}\left(\begin{bmatrix}-0.15\\1.05\\-5.99\\33.00\end{bmatrix}, \begin{bmatrix}0.85\\-5.95\\34.01\\-188\end{bmatrix}, \begin{bmatrix}0.85\\-5.95\\34.01\\-187\end{bmatrix}, \begin{bmatrix}-0.15\\1.05\\-5.99\\34.00\end{bmatrix}\right)$$

$$\mathcal{R}(5,0) = \mathrm{conv}\left(\begin{bmatrix}-5.93\\33.88\\-188.02\\1035.00\end{bmatrix}, \begin{bmatrix}1.07\\-6.12\\33.98\\-188.00\end{bmatrix}, \begin{bmatrix}1.07\\-6.12\\33.98\\-187.00\end{bmatrix}, \begin{bmatrix}-5.93\\33.88\\-188.02\\1036.00\end{bmatrix}\right)$$

Bandpass Filter Circuit Example

- $\mathcal{R}(1,0) = b\mathcal{U}$ where $\mathcal{U} = [0,1]$
  - b can be trivially computed
  - $b = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}^T$

Bandpass Filter Circuit Example

- $\mathcal{R}(1,0) = b\mathcal{U}$ where $\mathcal{U} = [0,1]$
  - b can be trivially computed
  - $b = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}^T$
- Recall $A^{i-1}b\mathcal{U} = \mathcal{R}(i,0) \ominus \mathcal{R}(i-1,0)$

$$Ab = \begin{bmatrix} 0 \\ 0 \\ 1 \\ -6 \end{bmatrix}, \; A^2 b = \begin{bmatrix} 0 \\ 1 \\ -6 \\ 33 \end{bmatrix}, \; A^3 b = \begin{bmatrix} 1 \\ -6 \\ 33 \\ -182 \end{bmatrix}, \; A^4 b = \begin{bmatrix} -6 \\ 33 \\ -182 \\ 1002 \end{bmatrix}$$

Bandpass Filter Circuit Example

- $\mathcal{R}(1,0) = b\mathcal{U}$ where $\mathcal{U} = [0,1]$
  - b can be trivially computed
  - $b = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}^T$
- Recall $A^{i-1}b\mathcal{U} = \mathcal{R}(i,0) \ominus \mathcal{R}(i-1,0)$

$$Ab = \begin{bmatrix} 0 \\ 0 \\ 1 \\ -6 \end{bmatrix},\ A^2b = \begin{bmatrix} 0 \\ 1 \\ -6 \\ 33 \end{bmatrix},\ A^3b = \begin{bmatrix} 1 \\ -6 \\ 33 \\ -182 \end{bmatrix},\ A^4b = \begin{bmatrix} -6 \\ 33 \\ -182 \\ 1002 \end{bmatrix}$$

- $A = AC_{A,b}C_{A,b}^{-1} = \begin{bmatrix} A^4b & A^3b & A^2b & Ab \end{bmatrix} \begin{bmatrix} A^3b & A^2b & Ab & b \end{bmatrix}^{-1}$

Bandpass Filter Circuit Example

- $\mathcal{R}(1,0) = b\mathcal{U}$ where $\mathcal{U} = [0,1]$
  - b can be trivially computed
  - $b = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}^T$
- Recall $A^{i-1}b\mathcal{U} = \mathcal{R}(i,0) \ominus \mathcal{R}(i-1,0)$

$$Ab = \begin{bmatrix} 0 \\ 0 \\ 1 \\ -6 \end{bmatrix}, \; A^2b = \begin{bmatrix} 0 \\ 1 \\ -6 \\ 33 \end{bmatrix}, \; A^3b = \begin{bmatrix} 1 \\ -6 \\ 33 \\ -182 \end{bmatrix}, \; A^4b = \begin{bmatrix} -6 \\ 33 \\ -182 \\ 1002 \end{bmatrix}$$

- $A = AC_{A,b}C_{A,b}^{-1} = \begin{bmatrix} A^4b & A^3b & A^2b & Ab \end{bmatrix} \begin{bmatrix} A^3b & A^2b & Ab & b \end{bmatrix}^{-1}$
- If we want to find A for an n dimensional system, we can do so with $n+1$ reachable sets

Further Analysis

- To apply this method to our previous results, we need to generalize these results to the continuous domain for multi-input systems
  - Can potentially avoid generalizing to nonlinear systems for short-time intervals
  - Zonotopes are not closed under Minkowski difference when $\mathcal{U} \subseteq \mathbb{R}^m$ and $m > 2$

Further Analysis

- To apply this method to our previous results, we need to generalize these results to the continuous domain for multi-input systems
  - Can potentially avoid generalizing to nonlinear systems for short-time intervals
  - Zonotopes are not closed under Minkowski difference when $\mathcal{U} \subseteq \mathbb{R}^m$ and $m > 2$
- We ideally want to develop a method that can use all the existing developments in resilient task assignment to synthesize control action for a large class of systems

Further Analysis

- To apply this method to our previous results, we need to generalize these results to the continuous domain for multi-input systems
  - Can potentially avoid generalizing to nonlinear systems for short-time intervals
  - Zonotopes are not closed under Minkowski difference when $\mathcal{U} \subseteq \mathbb{R}^m$ and $m > 2$
- We ideally want to develop a method that can use all the existing developments in resilient task assignment to synthesize control action for a large class of systems
- **New Strategy:**
  - Utilize the proxy system we use to calculate the guaranteed reachable set to synthesize control action with an arbitrarily small error for short-time intervals

Controller Synthesis for Reachable Paths

- We do not have the dynamics of the unknown system

Controller Synthesis for Reachable Paths

- We do not have the dynamics of the unknown system
- During resilient task assignment, we derived a proxy system whose reachable set converges to the true reachable set as $t \to 0$

Controller Synthesis for Reachable Paths

- We do not have the dynamics of the unknown system
- During resilient task assignment, we derived a proxy system whose reachable set converges to the true reachable set as $t \to 0$
- We want to use this proxy system to synthesize control action

Controller Synthesis for Reachable Paths

- We do not have the dynamics of the unknown system
- During resilient task assignment, we derived a proxy system whose reachable set converges to the true reachable set as $t \to 0$
- We want to use this proxy system to synthesize control action
- Challenges:

Controller Synthesis for Reachable Paths

- We do not have the dynamics of the unknown system
- During resilient task assignment, we derived a proxy system whose reachable set converges to the true reachable set as $t \to 0$
- We want to use this proxy system to synthesize control action
- Challenges:
    - Learn how u affects the unknown system based on information from proxy control system

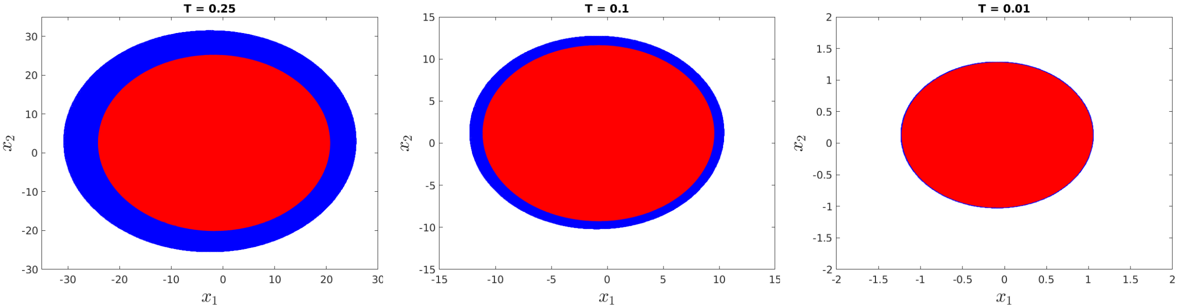
Controller Synthesis for Reachable Paths

- We do not have the dynamics of the unknown system
- During resilient task assignment, we derived a proxy system whose reachable set converges to the true reachable set as $t \to 0$
- We want to use this proxy system to synthesize control action
- Challenges:
  - Learn how u affects the unknown system based on information from proxy control system
  - Develop convergence guarantees
    - True state using proxy control must converge to the desired state

Controller Synthesis for Reachable Paths

- We do not have the dynamics of the unknown system
- During resilient task assignment, we derived a proxy system whose reachable set converges to the true reachable set as $t \to 0$
- We want to use this proxy system to synthesize control action
- Challenges:
  - Learn how u affects the unknown system based on information from proxy control system
  - Develop convergence guarantees
    - True state using proxy control must converge to the desired state
  - Synthesize control action capable of maneuvering an unknown system in real time

Reachable Sets as $t \to 0$



Review of Original Assumptions with New Problem Statement

- We have an unknown nonlinear control-affine system of the form:
  - $\dot{x} = f(x) + G(x)\mathcal{U}, \quad x(0) = x_0$
  - $f(x) \in \mathbb{R}^n$ and $G(x) \in \mathbb{R}^{n \times m}$
- We assume knowledge of:
  - The initial state x_0
  - The input set $\mathcal{U} = \mathbb{B}^m(0; 1)$
  - Local dynamics $f(x_0)$ and $G(x_0)$
    - Learned within an arbitrarily small error from test control inputs
  - The maximum growth rate of dynamics given by Lipschitz bounds L_f and L_G
    - Determined from known physical laws
    - Uncertainty quantification

Problem Statement

Synthesize control action for an unknown nonlinear system

Control Framework and Strategy

- Previous work showed that with these assumptions, we can produce some guaranteed set of reachable states

Control Framework and Strategy

- Previous work showed that with these assumptions, we can produce some guaranteed set of reachable states
- The guaranteed reachable set $\partial\hat{\mathcal{R}}(T, x_0)$ is found by finding the reachable set of a proxy system of the form
$$\dot{\hat{x}}(t) = a + (b - c|\hat{x}(t)|)\hat{u}(t), \quad \hat{x}(0) = x_0,$$
on the domain $\mathbb{B}$ where $a = f(x_0)$, $b = \|G^\dagger(x_0)\|^{-1}$, $c = L_f + L_G$, and $\hat{\phi}_{\hat{u}}(\cdot; x_0) : [0, \infty) \to \mathbb{R}^d$ is the controlled flow map (solution) under $\hat{u}$.

Control Framework and Strategy

- Previous work showed that with these assumptions, we can produce some guaranteed set of reachable states
- The guaranteed reachable set $\partial\hat{\mathcal{R}}(T, x_0)$ is found by finding the reachable set of a proxy system of the form
$$\dot{\hat{x}}(t) = a + (b - c|\hat{x}(t)|)\hat{u}(t), \quad \hat{x}(0) = x_0,$$
on the domain $\mathbb{B}$ where $a = f(x_0)$, $b = \|G^\dagger(x_0)\|^{-1}$, $c = L_f + L_G$, and $\hat{\phi}_{\hat{u}}(\cdot; x_0) : [0, \infty) \to \mathbb{R}^d$ is the controlled flow map (solution) under $\hat{u}$.

Proposition

Suppose $x_0 \in \mathrm{Int}(\mathbb{B})$ and $y \in \partial\hat{\mathcal{R}}(T, x_0)$. Then, $\hat{u} \equiv \frac{y - aT - x_0}{|y - aT - x_0|}$ is the unique control (almost everywhere) such that $\hat{\phi}_{\hat{u}}(T) = y$. Consequently, there exists a unique controlled path from x_0 to y.

Control Framework and Strategy

- Previous work showed that with these assumptions, we can produce some guaranteed set of reachable states
- The guaranteed reachable set $\partial\hat{\mathcal{R}}(T, x_0)$ is found by finding the reachable set of a proxy system of the form

$$\dot{\hat{x}}(t) = a + (b - c|\hat{x}(t)|)\hat{u}(t), \quad \hat{x}(0) = x_0,$$

  on the domain $\mathbb{B}$ where $a = f(x_0)$, $b = \|G^{\dagger}(x_0)\|^{-1}$, $c = L_f + L_G$, and $\hat{\phi}_{\hat{u}}(\cdot; x_0) : [0, \infty) \to \mathbb{R}^d$ is the controlled flow map (solution) under $\hat{u}$.

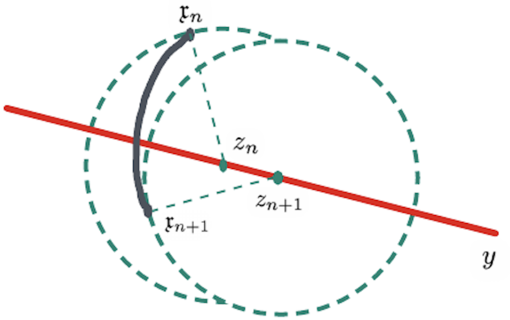
Proposition

Suppose $x_0 \in \mathrm{Int}(\mathbb{B})$ and $y \in \partial\hat{\mathcal{R}}(T, x_0)$. Then, $\hat{u} \equiv \frac{y - aT - x_0}{|y - aT - x_0|}$ is the unique control (almost everywhere) such that $\hat{\phi}_{\hat{u}}(T) = y$. Consequently, there exists a unique controlled path from x_0 to y.

- We can determine the GRS at some state, reach some $y \in \partial\hat{\mathcal{R}}(T, x_0)$, then repeat the process to follow some piecewise linear path and reach some eventual desired end state

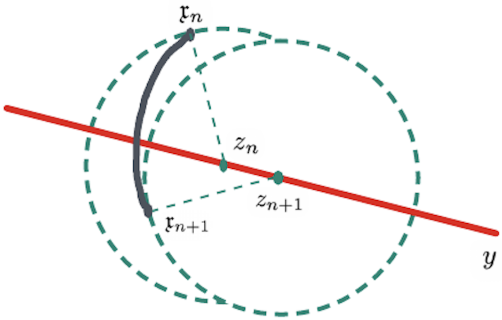
Control Framework and Strategy

- The pipeline is as follows:



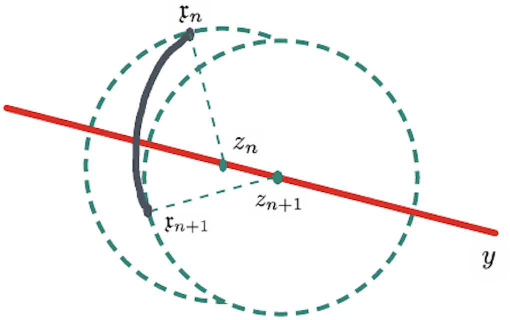
Control Framework and Strategy

- The pipeline is as follows:
    - We learn the system dynamics using $m + 1$ affinely independent constant inputs



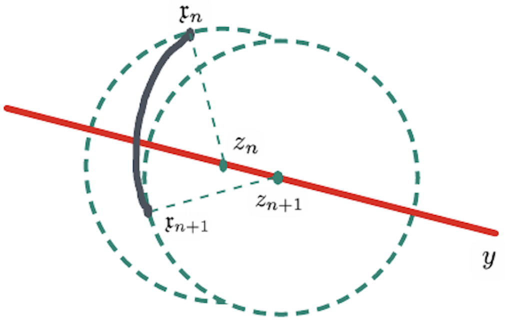
Control Framework and Strategy

- The pipeline is as follows:
  - We learn the system dynamics using $m+1$ affinely independent constant inputs
  - We execute an initial control $u = (1-\epsilon)\frac{G^{\dagger}(x_0)(y)}{\|G^{\dagger}(x_0)\||y|}$ to send the system towards y



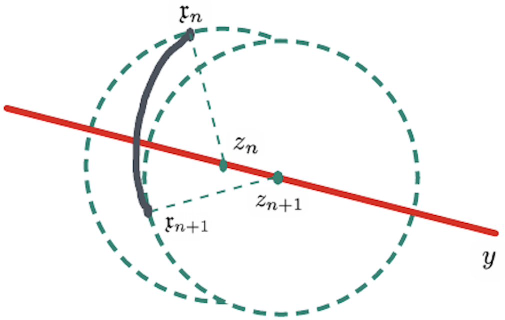
Control Framework and Strategy

- The pipeline is as follows:
  - We learn the system dynamics using $m+1$ affinely independent constant inputs
  - We execute an initial control $u = (1-\epsilon)\frac{G^{\dagger}(x_0)(y)}{\|G^{\dagger}(x_0)\||y|}$ to send the system towards y
  - We create a sequence $\{z_n\}$ such that $z_n = \theta_n y$ and $\{\theta_n\} \to 1$ is an increasing sequence where $\theta_n \in [0,1]$



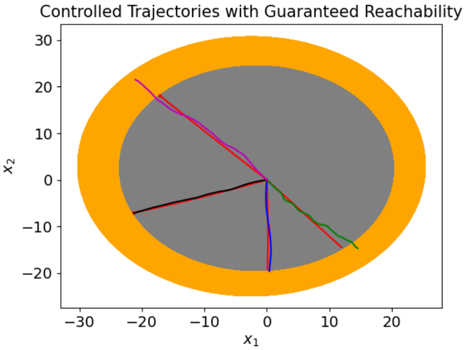
Control Framework and Strategy

- The pipeline is as follows:
  - We learn the system dynamics using $m+1$ affinely independent constant inputs
  - We execute an initial control $u = (1-\epsilon)\frac{G^{\dagger}(x_0)(y)}{||G^{\dagger}(x_0)|||y|}$ to send the system towards y
  - We create a sequence $\{z_n\}$ such that $z_n = \theta_n y$ and $\{\theta_n\} \to 1$ is an increasing sequence where $\theta_n \in [0,1]$
  - Each input u_n forms a direct path towards z_n with an arbitrarily small error
    - As $z_n \to y$, so to does the control system



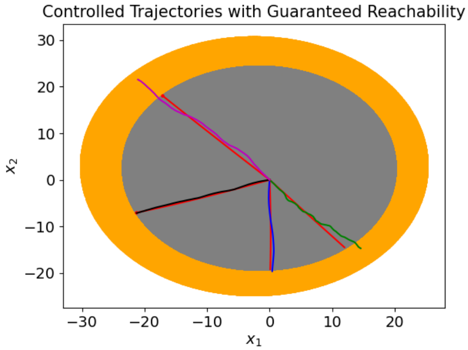
Implementation of Control Strategy

- The algorithm tuning parameters are δt, ϵ, k so that the appropriate construction of $\{z_n\}$ can be initialized



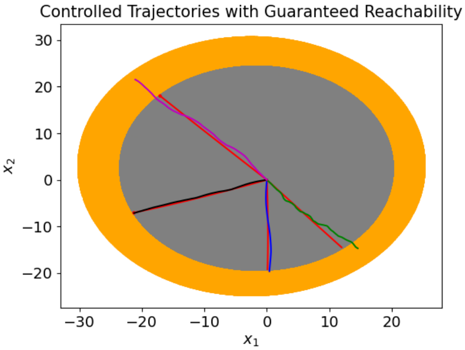
Implementation of Control Strategy

- The algorithm tuning parameters are δt, ϵ, k so that the appropriate construction of $\{z_n\}$ can be initialized
  - δt: Frequency at which algorithm chooses new z_n



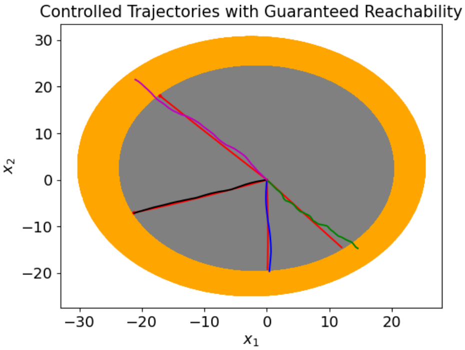
Implementation of Control Strategy

- The algorithm tuning parameters are δt, ϵ, k so that the appropriate construction of $\{z_n\}$ can be initialized
    - δt: Frequency at which algorithm chooses new z_n
    - ϵ: Ensures the sequence $\{u_{0,j}\}$ is a subset of $\mathcal{U}$
        - an ϵ too large creates a large error and if too small then $\{u_{0,j}\} \notin \mathcal{U}$



Implementation of Control Strategy

- The algorithm tuning parameters are δt, ϵ, k so that the appropriate construction of $\{z_n\}$ can be initialized
  - δt: Frequency at which algorithm chooses new z_n
  - ϵ: Ensures the sequence $\{u_{0,j}\}$ is a subset of $\mathcal{U}$
    - an ϵ too large creates a large error and if too small then $\{u_{0,j}\} \notin \mathcal{U}$
  - k: Determines the size of your convergence radius r
    - Large $k \implies$ large r



Reachable Predictive Control

- We can use proxy system dynamics to synthesize control action for small time intervals

Reachable Predictive Control

- We can use proxy system dynamics to synthesize control action for small time intervals
- We now have a verifiable method of controlling an unknown nonlinear system

Reachable Predictive Control

- We can use proxy system dynamics to synthesize control action for small time intervals
- We now have a verifiable method of controlling an unknown nonlinear system
- The pipeline is as follows:

Reachable Predictive Control

- We can use proxy system dynamics to synthesize control action for small time intervals
- We now have a verifiable method of controlling an unknown nonlinear system
- The pipeline is as follows:
    - Active learning methods learn the local dynamics with an arbitrarily small error

Reachable Predictive Control

- We can use proxy system dynamics to synthesize control action for small time intervals
- We now have a verifiable method of controlling an unknown nonlinear system
- The pipeline is as follows:
    - Active learning methods learn the local dynamics with an arbitrarily small error
    - Calculate the GRS using the local dynamics and additional information from known physical laws and a priori quantization techniques

Reachable Predictive Control

- We can use proxy system dynamics to synthesize control action for small time intervals
- We now have a verifiable method of controlling an unknown nonlinear system
- The pipeline is as follows:
    - Active learning methods learn the local dynamics with an arbitrarily small error
    - Calculate the GRS using the local dynamics and additional information from known physical laws and a priori quantization techniques
    - Synthesize a learned controller capable of navigating to any state in the GRS

Reachable Predictive Control

- We can use proxy system dynamics to synthesize control action for small time intervals
- We now have a verifiable method of controlling an unknown nonlinear system
- The pipeline is as follows:
  - Active learning methods learn the local dynamics with an arbitrarily small error
  - Calculate the GRS using the local dynamics and additional information from known physical laws and a priori quantization techniques
  - Synthesize a learned controller capable of navigating to any state in the GRS
  - Reinitialize the local dynamics around the newly reached state and repeat until some final goal state is reached

Future Work

- Given the existing accomplishments through existing publications, we would like to achieve the following:

Future Work

- Given the existing accomplishments through existing publications, we would like to achieve the following:
  - Utilize machine learning techniques to create bounds consistent with assumptions in resilient task assignment

Future Work

- Given the existing accomplishments through existing publications, we would like to achieve the following:
  - Utilize machine learning techniques to create bounds consistent with assumptions in resilient task assignment
  - Generalize results to a larger class of underactuated systems

Future Work

- Given the existing accomplishments through existing publications, we would like to achieve the following:
  - Utilize machine learning techniques to create bounds consistent with assumptions in resilient task assignment
  - Generalize results to a larger class of underactuated systems
  - Utilize multiple trajectories to improve results derived in resilient task assignment
    - Develop a notion of long-term stability guarantees for RPC

Future Work

- Given the existing accomplishments through existing publications, we would like to achieve the following:
  - Utilize machine learning techniques to create bounds consistent with assumptions in resilient task assignment
  - Generalize results to a larger class of underactuated systems
  - Utilize multiple trajectories to improve results derived in resilient task assignment
    - Develop a notion of long-term stability guarantees for RPC
  - Demonstrate RPC effectively on a system without knowledge of the system dynamics

Future Work

- Given the existing accomplishments through existing publications, we would like to achieve the following:
  - Utilize machine learning techniques to create bounds consistent with assumptions in resilient task assignment
  - Generalize results to a larger class of underactuated systems
  - Utilize multiple trajectories to improve results derived in resilient task assignment
    - Develop a notion of long-term stability guarantees for RPC
  - Demonstrate RPC effectively on a system without knowledge of the system dynamics
- The ultimate goal is to illustrate by example how RPC can control an unknown nonlinear system and analyze its efficacy

Resilient Task Assignment for Underactuated Systems

- Emphasize that the ultimate goal is to make RPC real-time implementable

Resilient Task Assignment for Underactuated Systems

- Emphasize that the ultimate goal is to make RPC real-time implementable
- A large class of practical systems are underactuated
    - Motivates us to broaden the existing results from resilient task assignment to include a larger class of underactuated systems

Resilient Task Assignment for Underactuated Systems

- Emphasize that the ultimate goal is to make RPC real-time implementable
- A large class of practical systems are underactuated
  - Motivates us to broaden the existing results from resilient task assignment to include a larger class of underactuated systems
- Challenge:
  - The guaranteed velocity set is empty for general cases when $\mathrm{Im}(f(x)) \not\subset \mathrm{Im}(G(x))$
    - We cannot repeat the same method as before to solve for the GRS
    - The same method would result in the empty set

Resilient Task Assignment for Underactuated Systems

- Emphasize that the ultimate goal is to make RPC real-time implementable
- A large class of practical systems are underactuated
  - Motivates us to broaden the existing results from resilient task assignment to include a larger class of underactuated systems
- Challenge:
  - The guaranteed velocity set is empty for general cases when $\mathrm{Im}(f(x)) \not\subset \mathrm{Im}(G(x))$
    - We cannot repeat the same method as before to solve for the GRS
    - The same method would result in the empty set
- We use the notion of *eventual reachabilty*

Resilient Task Assignment for Underactuated Systems

Eventual (Forward) Reachability

We define the eventual forward reachable set of $\mathcal{M}(f, G)$ as

$$\mathcal{R}^{f,G}(T, x_0) = \left\{ \bigcup_{t \in [0,T]} \phi_u^{f,G}(t; x_0) \mid u : [0, T] \to \mathcal{U},\ t \in [0, T] \right\}$$

where $\mathcal{R}^{f,G}(T, x_0)$ represents the set of states that can be reached within some time T.

- For short time intervals the exact amount of time it takes to reach a state is less significant
  - Interested in determining if a state is provably reachable
  - Not important to know the exact time that state is reached

Resilient Task Assignment

Lemma

Let

$$\dot{x} = f(x(t)), \quad x(0) = x_0 \tag{2}$$

and let $\lambda : \mathbb{R}^n \to [1, +\infty)$ such that

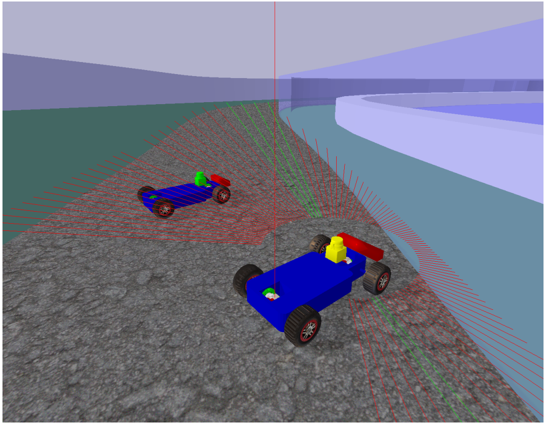
$$\dot{z} = \lambda(z(t)) f(z(t)), \quad z(0) = z_0. \tag{3}$$

If $x_0 = z_0$ and $\lim_{t\to\infty} \int_0^t \frac{1}{\lambda(x(s))}\, ds = \infty$, then there exists $t \in [0, T]$ such that $z(t) = x(T)$.

- If two velocities are colinear and the initial state $x(0) = z(0)$, then any state reached by (2) at some time T would have been reached by (3) for some time $t \in [0, T]$

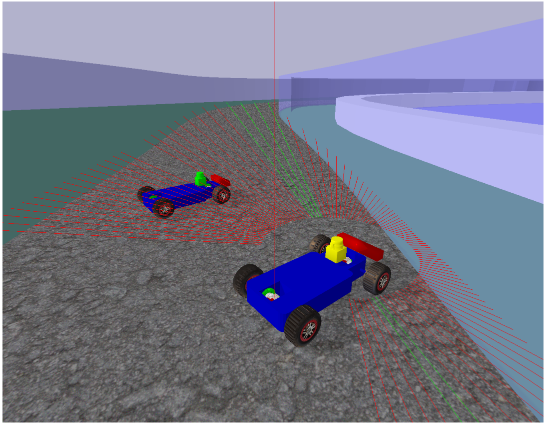
Autonomous Vehicle Application

- We want to demonstrate the ability for RPC control a system

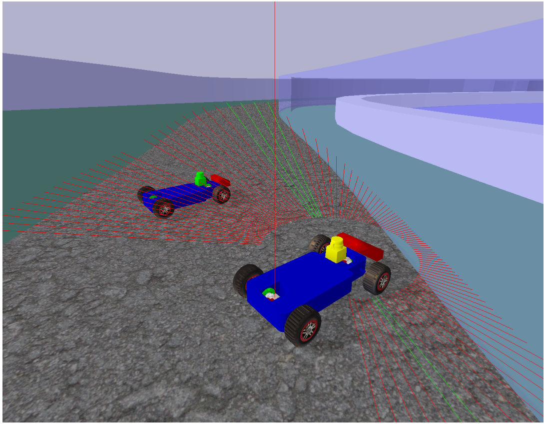


Autonomous Vehicle Application

- We want to demonstrate the ability for RPC control a system
- We can use the proxy system to develop control action aimed at navigating an autonomous system with short-term performance guarantees

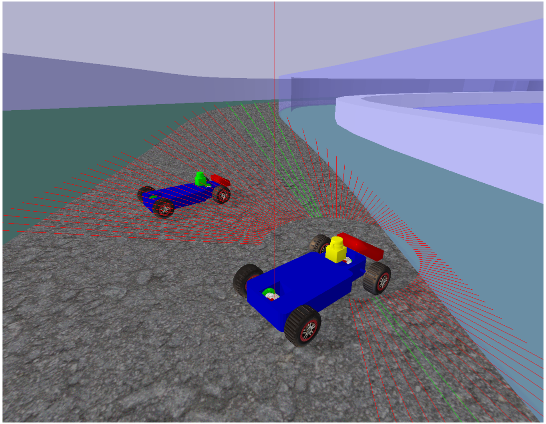


- We want to demonstrate the ability for RPC control a system
- We can use the proxy system to develop control action aimed at navigating an autonomous system with short-term performance guarantees
- Challenge:



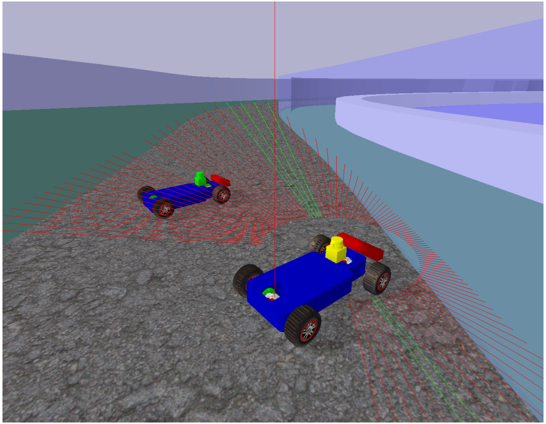
Autonomous Vehicle Application

- We want to demonstrate the ability for RPC control a system
- We can use the proxy system to develop control action aimed at navigating an autonomous system with short-term performance guarantees
- Challenge:
  - Use onboard sensors to detect obstacles and navigate around an unknown environment with the ultimate goal of reaching some state using RPC



Autonomous Vehicle Application

- We want to demonstrate the ability for RPC control a system
- We can use the proxy system to develop control action aimed at navigating an autonomous system with short-term performance guarantees
- Challenge:
    - Use onboard sensors to detect obstacles and navigate around an unknown environment with the ultimate goal of reaching some state using RPC
    - Determine how to choose desired reach states within the underapproximated GRS to reach eventual goal state



Soft Robotics Application

- Soft robots are difficult to model
  - Physics-based models are limited due to the chaotic nature of soft material

Soft Robotics Application

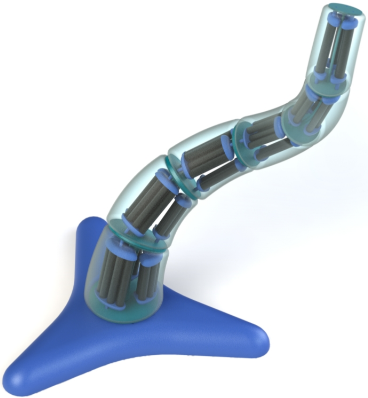
- Soft robots are difficult to model
  - Physics-based models are limited due to the chaotic nature of soft material
- Use machine learning methods to quantify a relationship between position and stiffness

Soft Robotics Application

- Soft robots are difficult to model
  - Physics-based models are limited due to the chaotic nature of soft material
- Use machine learning methods to quantify a relationship between position and stiffness
- Use information from machine learning to gain knowledge consistent with assumptions outlined in previous work for resilient task assignment

Soft Robotics Application

- Soft robots are difficult to model
  - Physics-based models are limited due to the chaotic nature of soft material
- Use machine learning methods to quantify a relationship between position and stiffness
- Use information from machine learning to gain knowledge consistent with assumptions outlined in previous work for resilient task assignment
- Apply RPC to accomplish a provably achievable performance objective



Control of Unknown Systems Using the Koopman Operator

- Goal: Utilize observable trajectories to perform system identification within some quantifiable error
  - We can use dynamics recovered through SysID using the Koopman operator to perform reachability analysis and synthesize control action

Control of Unknown Systems Using the Koopman Operator

- Goal: Utilize observable trajectories to perform system identification within some quantifiable error
  - We can use dynamics recovered through SysID using the Koopman operator to perform reachability analysis and synthesize control action
- Consider a complete normed function space $(\mathcal{F}, \|\cdot\|_{\mathcal{F}})$ of real-valued observable functions $h : \mathcal{X} \to \mathbb{R}$.
  - In the context of operator learning, observable functions refer to test functions for operators
  - Not equal to control systems which refer to the concept of observability

Control of Unknown Systems Using the Koopman Operator

- Goal: Utilize observable trajectories to perform system identification within some quantifiable error
  - We can use dynamics recovered through SysID using the Koopman operator to perform reachability analysis and synthesize control action
- Consider a complete normed function space $(\mathcal{F}, \|\cdot\|_{\mathcal{F}})$ of real-valued observable functions $h : \mathcal{X} \to \mathbb{R}$.
  - In the context of operator learning, observable functions refer to test functions for operators
  - Not equal to control systems which refer to the concept of observability

Koopman Operator Family

The Koopman operator family $\{\mathcal{K}\}_{t\geq 0}$ of the nonlinear system is a collection of maps $\mathcal{K}_t : \mathcal{F} \to \mathcal{F}$ defined by

$$\mathcal{K}_t h = h \circ \phi(t, \cdot), \quad h \in \mathcal{F}$$

where $\circ$ denotes the composition of ϕ with $\mathcal{F}$.

Control of Unknown Systems Using the Koopman Operator

- For autonomous systems

$$\dot{x} = f(x),$$

  there is a semigroup of Koopman operators $\mathcal{K}_{\Delta t}$ associated with the flow map for time interval Δt

  - The infinitesimal generator for this semigroup is defined as

$$\mathcal{L}h(x) := \lim_{t \to 0} \frac{\mathcal{K}_t h(x) - h(x)}{t} \approx \int_0^{\tau} e^{-\lambda s} h(\phi(s, x)) \, ds$$

    with any fixed $\tau > 0$ for a sufficiently large λ

Control of Unknown Systems Using the Koopman Operator

- For autonomous systems
$$\dot{x} = f(x),$$
there is a semigroup of Koopman operators $\mathcal{K}_{\Delta t}$ associated with the flow map for time interval Δt
  - The infinitesimal generator for this semigroup is defined as
$$\mathcal{L}h(x) := \lim_{t\to 0}\frac{\mathcal{K}_t h(x) - h(x)}{t} \approx \int_0^{\tau} e^{-\lambda s} h(\phi(s,x))\, ds$$
with any fixed $\tau > 0$ for a sufficiently large λ
- Define $p_j : \mathcal{X} \to \mathbb{R}$ as the projection function to the j^{th} dimension

Control of Unknown Systems Using the Koopman Operator

- For autonomous systems
$$\dot{x} = f(x),$$
there is a semigroup of Koopman operators $\mathcal{K}_{\Delta t}$ associated with the flow map for time interval Δt
  - The infinitesimal generator for this semigroup is defined as
$$\mathcal{L}h(x) := \lim_{t\to 0} \frac{\mathcal{K}_t h(x) - h(x)}{t} \approx \int_0^{\tau} e^{-\lambda s} h(\phi(s,x))\, ds$$
with any fixed $\tau > 0$ for a sufficiently large λ
- Define $p_j : \mathcal{X} \to \mathbb{R}$ as the projection function to the j^{th} dimension
- We can learn f_j at each j at samples points $\{x^m\}$ with
$$f_j(x^{(m)}) \approx \mathcal{L}p_j(x) = \int_0^{\tau} e^{-\lambda s} p_j(\phi(s,x))\, ds$$

Control of Unknown Systems Using the Koopman Operator

- We have the unknown nonlinear system

$$\dot{x} = f(x) + G(x)u = f(x) + \sum_{l} g_l(x)u^l, \quad x(0) = x_0$$

  - We can use the method outlined in the previous slide to learn f and g_l using the Koopman operator

Control of Unknown Systems Using the Koopman Operator

- We have the unknown nonlinear system

$$\dot{x} = f(x) + G(x)u = f(x) + \sum_{l} g_l(x)u^l, \quad x(0) = x_0$$

  - We can use the method outlined in the previous slide to learn f and g_l using the Koopman operator
- With enough data, we can perform system identification to learn f and g_l within some error bound
  - Additionally, we can determine long-term stability guarantees

Control of Unknown Systems Using the Koopman Operator

- We have the unknown nonlinear system

$$\dot{x} = f(x) + G(x)u = f(x) + \sum_{l} g_l(x)u^l, \quad x(0) = x_0$$

  - We can use the method outlined in the previous slide to learn f and g_l using the Koopman operator
- With enough data, we can perform system identification to learn f and g_l within some error bound
  - Additionally, we can determine long-term stability guarantees
- Further improvement involves adaptation of onboard capabilities
  - Determining how information gathered onboard can improve error bounds and performance

Publications

[1] T. Shafa and M. Ornik, "Reachability of nonlinear systems with unknown dynamics," *IEEE Transactions on Automatic Control*, vol. 68, no. 4, pp. 2407–2414, 2022.

[2] ——, "Maximal ellipsoid method for guaranteed reachability of unknown fully actuated systems," in *2022 IEEE 61st Conference on Decision and Control (CDC)*. IEEE, 2022, pp. 5002–5007.

[3] ——, "Guaranteed reachability on riemannian manifolds for unknown nonlinear systems," *arXiv preprint arXiv:2404.09850*, 2024.

[4] T. Shafa, R. Dong, and M. Ornik, "Identifying single-input linear system dynamics from reachable sets," in *2023 62nd IEEE Conference on Decision and Control (CDC)*. IEEE, 2023, pp. 3527–3532.

[5] Y. Meng, T. Shafa, J. Wei, and M. Ornik, "Online learning and control synthesis for reachable paths of unknown nonlinear systems," *arXiv preprint arXiv:2403.03413*, 2024.

Questions

Taha Shafa

tahaas2@illinois.edu