Ensuring Safety for Sampled Data Systems

An Efficient Algorithm for Filtering Potentially Unsafe Input Signals

Ian M. Mitchell[1], Jeffrey Yeh[1], Forrest J. Laine[2], Claire J. Tomlin[2]

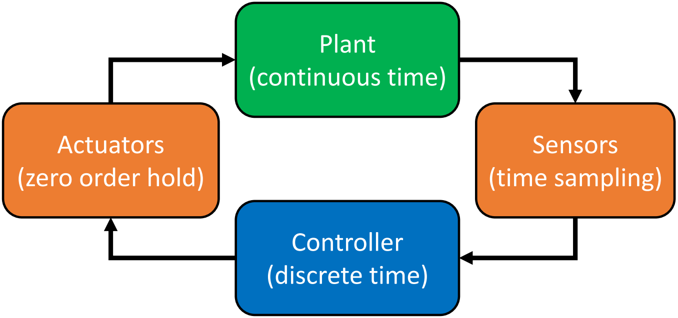
[1]Department of Computer Science
The University of British Columbia
[2]Department of Electrical Engineering & Computer Science
University of California, Berkeley

December 2016

mitchell@cs.ubc.ca
http://www.cs.ubc.ca/~mitchell

Motivation: Sampled Data Systems

A common design pattern for cyber-physical systems:



Traditional models of time evolution miss important features of this design:

- Continuous time models ignore the periodic nature of feedback.
- Discrete time models ignore plant evolution between samples.

The sampled data model captures these features.

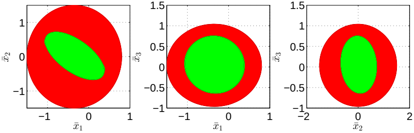
Previous Work

[Mitchell, Kaynama, Chen & Oishi, NAHS 2013]:

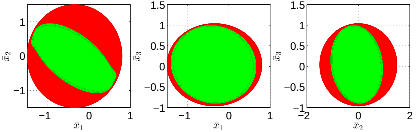
- Developed an algorithm to approximate sampled data discriminating kernels.
- Demonstrated on toy examples.

[Mitchell & Kaynama, HSCC 2015]:

- Described an algorithm to more accurately approximate sampled data discriminating kernels robust to sample time jitter.
- Demonstrated on a partially nonlinear three dimensional model of quadrotor altitude maintenance.



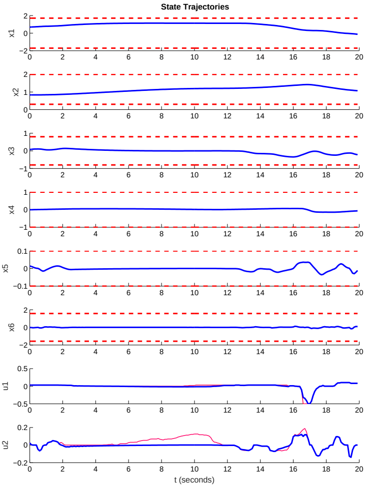
[Mitchell et al, NAHS 2013]



[Mitchell & Kaynama, HSCC 2015]

Contributions

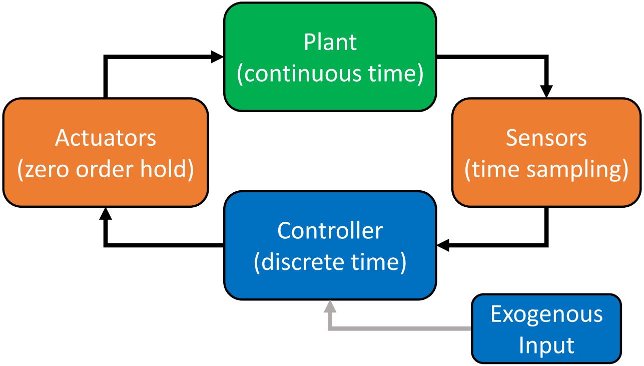
- Adapt algorithm to fixed time capture basins.
- Construct discrete state automaton / look-up table for controller to synthesize (set-valued) safe feedback control signals.
- Demonstrate on a partially nonlinear six dimensional longitudinal model of quadrotor flight.



Set-Valued Safe Control?

But the plant requires a single control signal!

- Proposed automaton represents a verified control envelope [Aréchiga & Krogh, ACC 2014] which could be used to more efficiently design, modify or tune proposed controllers to ensure safety.
- Set-valued constraints can be used online to check and possibly modify exogenous input signal, such as human-in-the-loop or legacy controller



Outline

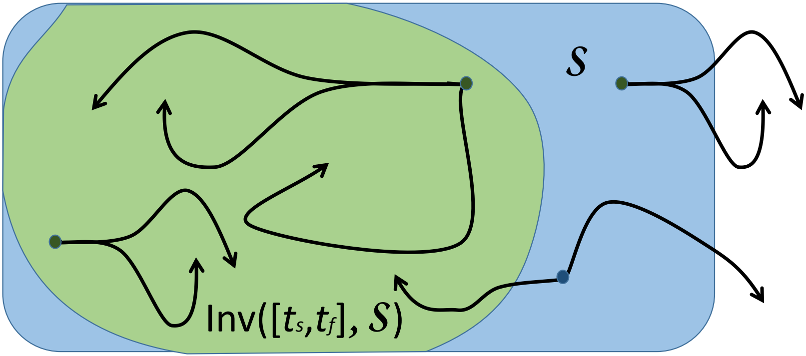
1. Motivation & Contributions
2. Constructs
3. Models & Algorithms
4. Control Filtering Hybrid Automaton
5. Quadrotor Flight Envelope Maintenance

Outline

1. Motivation & Contributions
2. Constructs
3. Models & Algorithms
4. Control Filtering Hybrid Automaton
5. Quadrotor Flight Envelope Maintenance

Invariance Kernel

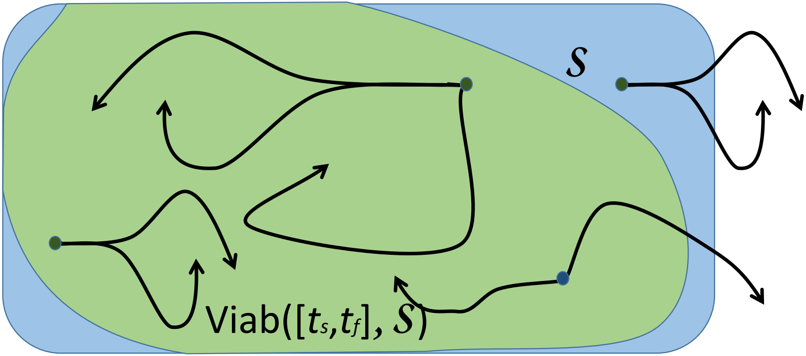
$$\text{Inv}([t_s, t_f], \mathcal{S}) \triangleq \{x(t_s) \in \mathcal{S} \mid \forall u(\cdot), \forall t \in [t_s, t_f], x(t) \in \mathcal{S}\},$$



- What states will remain safe despite input uncertainty.
- Inputs treated in a worst-case fashion.

Viability Kernel

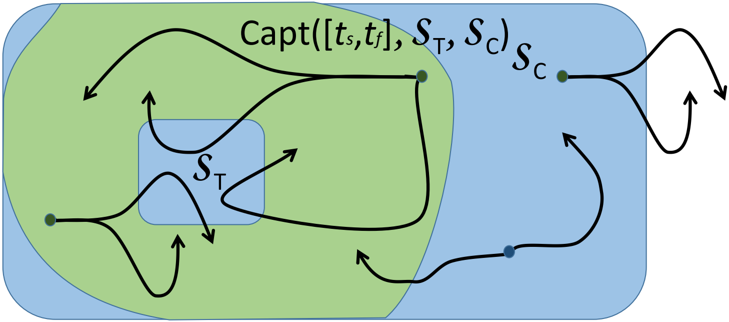
$$\text{Viab}\,([t_s, t_f], \mathcal{S}) \triangleq \{x(t_s) \in \mathcal{S} \mid \exists u(\cdot), \forall t \in [t_s, t_f], x(t) \in \mathcal{S}\},$$



- Also called controlled invariant set.
- Inputs treated in a best-case fashion.

Capture Basin

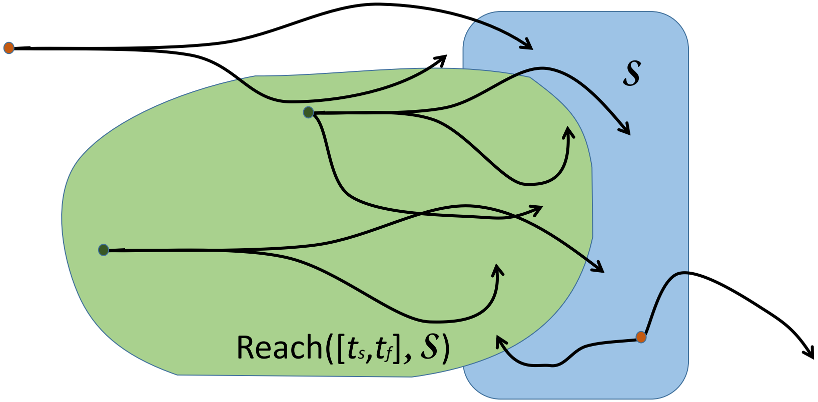
$$\text{Capt}\left([t_s, t_f], \mathcal{S}_T, \mathcal{S}_C\right) \triangleq \left\{ x(t_s) \in \mathcal{S}_C \;\middle|\; \begin{array}{c} \exists u(\cdot), \exists t_T \in [t_s, t_f], \forall t \in [t_s, t_T], \\ x(t) \in \mathcal{S}_C \, \wedge \, x(t_T) \in \mathcal{S}_T \end{array} \right\},$$



- Trajectories must stay inside constraint $\mathcal{S}_C$ until they reach target $\mathcal{S}_T$
- Inputs treated in a best-case fashion.

Robust Reach Set

$$\text{Reach}\left([t_s, t_f], \mathcal{S}\right) \triangleq \{x(t_s) \in \Omega \mid \forall v(\cdot), x(t_f) \in \mathcal{S}\}$$



- Not a reach tube: Trajectories must reach $\mathcal{S}$ at exactly t_f.
- Reach tube may not be the union of these reach sets [Mitchell 2007].

Discriminating Kernel

$$\mathsf{Disc}\left([t_s, t_f], \mathcal{S}\right) \triangleq \{x(t_s) \in \mathcal{S} \mid \exists u(\cdot), \forall v(\cdot), \forall t \in [t_s, t_f], x(t) \in \mathcal{S}\},$$

That is hard to draw...

- Also called robust controlled invariant set.
- Two inputs "control" $u(\cdot)$ and "disturbance" $v(\cdot)$ treated adversarially.

The Challenge: Efficient Parametric Representations

Existing algorithms used non-parametric representations; complexity is exponential in state space dimension.

- Viability algorithms: for example [Saint-Pierre 1994; Cardaliaguet et al 1999].
- Level set methods: for example [Mitchell et al 2005].
- New: Outer approximation of capture basin ("region of attraction") using occupational measures and SDP for polynomial dynamics (no disturbance inputs) [Henrion & Korda, IEEE TAC 2014].

In contrast, algorithms using parametric representations for reachable sets are widely available.

- Ellipsoids: for example [Kurzhanski & Valyi 1996; Kurzhanski & Varaiya 2000; Kurzhanskiy & Varaiya 2006].
- Support functions / vectors: for example [Le Guernic 2009; Le Guernic & Girard 2010; Frehse et al 2011].

Outline

1. Motivation & Contributions
2. Constructs
3. Models & Algorithms
4. Control Filtering Hybrid Automaton
5. Quadrotor Flight Envelope Maintenance

Discrete and Continuous Time

Discrete time:

$$x(t+1) = f(x(t), u(t), v(t)) \qquad \text{general dynamics}$$
$$x(t+1) = \mathrm{A}x(t) + \mathrm{B}u(t) + \mathrm{C}v(t) \qquad \text{linear dynamics}$$

- Assume state feedback: Choose $u(t)$ knowing $x(t)$.
- Conservative treatment of uncertainty: Choose $v(t)$ knowing $x(t)$ and $u(t)$.

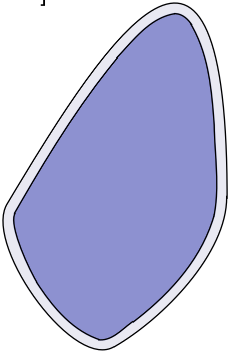
Continuous time:

$$\dot{x}(t) = f(x(t), u(t), v(t)) \qquad \text{general dynamics}$$
$$\dot{x}(t) = \mathrm{A}x(t) + \mathrm{B}u(t) + \mathrm{C}v(t) \qquad \text{linear dynamics}$$

- "Non-anticipative strategies" rigorously resolve input ordering issue; equivalent to state feedback in all but artificially constructed examples.
- Optimal input signals often have little regularity and hence may not be physically realizable.

Continuous-Time Viability Algorithm

[Maidens et al, Automatica 2013], [Kaynama et al, HSCC 2012]



- Let ρ be a small computational timestep and M a uniform bound on f.
- Start with an under-approximation $\mathcal{K}_\downarrow$ of $\mathcal{K}$

$$\mathcal{K}_\downarrow := \{x \in \mathcal{K} \mid \operatorname{dist}(x, \mathcal{K}^c) \geq \rho M\}$$

- Iteratively compute K_{n+1}:

$$\mathcal{K}_0 = \mathcal{K}_\downarrow,$$
$$\mathcal{K}_{n+1}(P) = \mathcal{K}_0 \cap \mathsf{Reach}\left([0, \rho], \mathcal{K}_n\right)$$

- Discriminating kernel algorithm is straightforward, albeit notationally complicated.
- Discrete time algorithm omits initial erosion: $\mathcal{K}_0 = \mathcal{K}$.

Continuous-Time Viability Algorithm

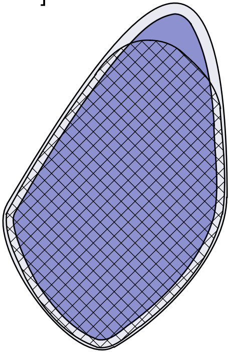
[Maidens et al, Automatica 2013], [Kaynama et al, HSCC 2012]

- Let ρ be a small computational timestep and M a uniform bound on f.
- Start with an under-approximation $\mathcal{K}_\downarrow$ of $\mathcal{K}$

$$\mathcal{K}_\downarrow := \{x \in \mathcal{K} \mid \operatorname{dist}(x, \mathcal{K}^c) \geq \rho M\}$$

- Iteratively compute K_{n+1}:

$$\begin{aligned} \mathcal{K}_0 &= \mathcal{K}_\downarrow, \\ \mathcal{K}_{n+1}(P) &= \mathcal{K}_0 \cap \mathsf{Reach}\left([0, \rho], \mathcal{K}_n\right) \end{aligned}$$



- Discriminating kernel algorithm is straightforward, albeit notationally complicated.
- Discrete time algorithm omits initial erosion: $\mathcal{K}_0 = \mathcal{K}$.

Continuous-Time Viability Algorithm

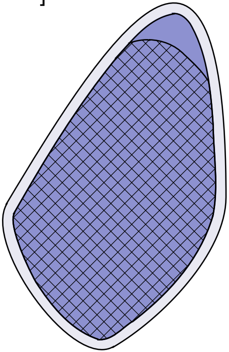
[Maidens et al, Automatica 2013], [Kaynama et al, HSCC 2012]

- Let ρ be a small computational timestep and M a uniform bound on f.
- Start with an under-approximation $\mathcal{K}_\downarrow$ of $\mathcal{K}$

$$\mathcal{K}_\downarrow := \{x \in \mathcal{K} \mid \mathrm{dist}(x, \mathcal{K}^c) \geq \rho M\}$$

- Iteratively compute K_{n+1}:

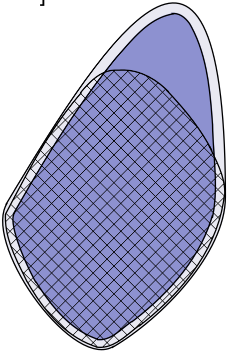
$$\begin{aligned} \mathcal{K}_0 &= \mathcal{K}_\downarrow, \\ \mathcal{K}_{n+1}(P) &= \mathcal{K}_0 \cap \mathsf{Reach}\left([0,\rho], \mathcal{K}_n\right) \end{aligned}$$



- Discriminating kernel algorithm is straightforward, albeit notationally complicated.
- Discrete time algorithm omits initial erosion: $\mathcal{K}_0 = \mathcal{K}$.

Continuous-Time Viability Algorithm

[Maidens et al, Automatica 2013], [Kaynama et al, HSCC 2012]



- Let ρ be a small computational timestep and M a uniform bound on f.
- Start with an under-approximation $\mathcal{K}_\downarrow$ of $\mathcal{K}$

$$\mathcal{K}_\downarrow := \{x \in \mathcal{K} \mid \mathrm{dist}(x, \mathcal{K}^c) \geq \rho M\}$$

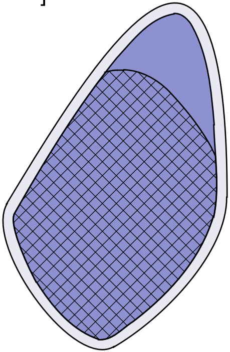
- Iteratively compute K_{n+1}:

$$\begin{aligned} \mathcal{K}_0 &= \mathcal{K}_\downarrow, \\ \mathcal{K}_{n+1}(P) &= \mathcal{K}_0 \cap \text{Reach}\left([0, \rho], \mathcal{K}_n\right) \end{aligned}$$

- Discriminating kernel algorithm is straightforward, albeit notationally complicated.
- Discrete time algorithm omits initial erosion: $\mathcal{K}_0 = \mathcal{K}$.

Continuous-Time Viability Algorithm

[Maidens et al, Automatica 2013], [Kaynama et al, HSCC 2012]



- Let ρ be a small computational timestep and M a uniform bound on f.
- Start with an under-approximation $\mathcal{K}_\downarrow$ of $\mathcal{K}$

$$\mathcal{K}_\downarrow := \{x \in \mathcal{K} \mid \operatorname{dist}(x, \mathcal{K}^c) \geq \rho M\}$$

- Iteratively compute K_{n+1}:

$$\begin{aligned} \mathcal{K}_0 &= \mathcal{K}_\downarrow, \\ \mathcal{K}_{n+1}(P) &= \mathcal{K}_0 \cap \mathsf{Reach}\left([0, \rho], \mathcal{K}_n\right) \end{aligned}$$

- Discriminating kernel algorithm is straightforward, albeit notationally complicated.
- Discrete time algorithm omits initial erosion: $\mathcal{K}_0 = \mathcal{K}$.

Continuous-Time Viability Algorithm

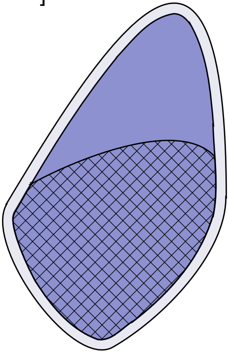
[Maidens et al, Automatica 2013], [Kaynama et al, HSCC 2012]

- Let ρ be a small computational timestep and M a uniform bound on f.
- Start with an under-approximation $\mathcal{K}_{\downarrow}$ of $\mathcal{K}$

$$\mathcal{K}_{\downarrow} := \{x \in \mathcal{K} \mid \mathrm{dist}(x, \mathcal{K}^c) \geq \rho M\}$$

- Iteratively compute K_{n+1}:

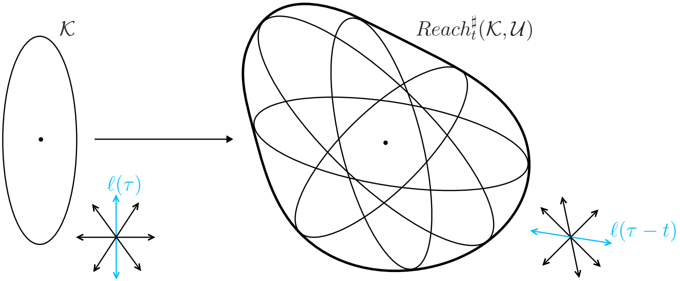
$$\mathcal{K}_0 = \mathcal{K}_{\downarrow},$$
$$\mathcal{K}_{n+1}(P) = \mathcal{K}_0 \cap \mathsf{Reach}\left([0, \rho], \mathcal{K}_n\right)$$



- Discriminating kernel algorithm is straightforward, albeit notationally complicated.
- Discrete time algorithm omits initial erosion: $\mathcal{K}_0 = \mathcal{K}$.

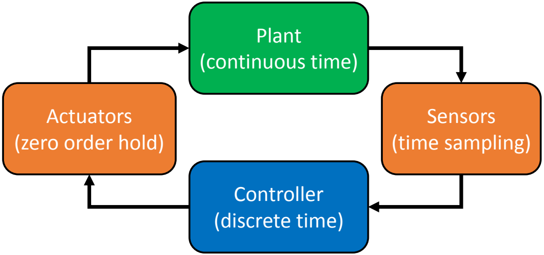
Ellipsoidal Representations

Ellipsoidal techniques (under-)approximating the maximal reach set:



- Key operations (set evolution, intersection) are accomplished through ODEs and convex optimization.
- Class of ellipsoids are not closed under these operations, so underapproximations must be used.
- Set evolution for linear dynamics possible in discrete or continuous time.
- Control and/or disturbance inputs can be treated.

Sampled Data Model of Time



- Use continuous time model of the plant

$$\dot{x}(t) = f(x(t), u(t), v(t)) \qquad \text{general dynamics;}$$

$$\dot{x}(t) = \mathrm{A}x(t) + \mathrm{B}u(t) + \mathrm{C}v(t) \qquad \text{linear dynamics.}$$

- However, control input is piecewise constant in time

$$u_{\mathsf{pw}}(t) = u_{\mathsf{fb}}(x(t_k)) \text{ for } t_k \leq t < t_{k+1}$$

where $u_{\mathsf{fb}} : \Omega \to \mathcal{U}$ is a feedback control policy.

- Disturbance input is allowed to vary (measurably) continuously.

Sampled Data Formulation

- Assume fixed sample time, but can be extended to handle timing jitter.
- Sampled data algorithm uses continuous time algorithm in an augmented state space

$$\tilde{x} \triangleq \begin{bmatrix} x \\ u \end{bmatrix} \qquad \tilde{f}(\tilde{x}, v) \triangleq \begin{bmatrix} f(x, u, v) \\ 0 \end{bmatrix}.$$

- Move between original and augmented state space with tensor products and projections

$$\mathsf{Proj}_x\left(\tilde{\mathcal{X}}\right) \triangleq \left\{ x \in \Omega \;\middle|\; \exists u, \begin{bmatrix} x \\ u \end{bmatrix} \in \tilde{\mathcal{X}} \right\},$$

$$\mathsf{Proj}_u\left(\tilde{\mathcal{X}}, x\right) \triangleq \left\{ u \in \mathbb{U} \;\middle|\; \begin{bmatrix} x \\ u \end{bmatrix} \in \tilde{\mathcal{X}} \right\}.$$

Finite Horizon Sampled Data Capture Basin

Define

- Sample period δ
- Horizon $T = \bar{N}\delta$
- Constraint set S_C
- Target set $S_T \subset S_C$
- Finite horizon sampled data capture basin

$$\text{Capt}_{\text{sd}}\left([0,T], \mathcal{S}_T, \mathcal{S}_C\right) \triangleq \left\{ x_0 \in \mathcal{S}_C \;\middle|\; \begin{array}{c} \exists u_{\text{pw}}(\cdot), \exists i \in \{0, 1, \dots, \bar{N}\}, \\ \forall v(\cdot), \forall t \in [0, i\delta], \\ x(t) \in \mathcal{S}_C \wedge x(i\delta) \in \mathcal{S}_T \end{array} \right\}.$$

If a safe infinite horizon feedback controller $u_{\text{fb}}^{\text{inf}}(x)$ is available for $x \in \mathcal{S}_T$, then capture basin is also infinite horizon safe.

Capture Basin Algorithm

- For $i = 1, 2, \dots, \bar{N}$

$$
\begin{aligned}
\mathcal{E}_i &\triangleq \mathcal{E}(\mathsf{Capt}_i\,(\mathcal{S}_T, \mathcal{S}_C)) \\
\mathcal{E}_0 &= \mathcal{E}(\mathcal{S}_T) \\
\mathcal{E}(\mathcal{I}_1) &\triangleq \mathcal{E}(\mathsf{Inv}\,([0,\delta], \mathcal{S}_C \times \mathbb{U})), \\
\mathcal{E}(\mathcal{R}_i) &\triangleq \mathcal{E}(\mathsf{Reach}\,([0,\delta], \mathcal{E}_{i-1} \times \mathbb{U})), \\
\mathcal{E}(\mathcal{C}_i) &\triangleq \mathsf{Inscribed}_\alpha\left(\mathcal{E}(\mathcal{R}_i) \cap \mathcal{E}(\mathcal{I}_1)\right), \\
\mathcal{E}_i &= \mathsf{Proj}_x\left(\mathsf{Inscribed}_0\left(\mathcal{E}(\mathcal{C}_i) \cap \mathcal{E}(\Omega \times \mathcal{E}(\mathcal{U}))\right)\right),
\end{aligned}
$$

- Overapproximates the sampled data capture basin

$$
\bigcup_{i=0}^{\bar{N}} \mathcal{E}_i \subseteq \mathsf{Capt}_{\mathsf{sd}}\left([0,T], \mathcal{S}_T, \mathcal{S}_C\right).
$$

- Provides a safe control policy

$$
\mathcal{U}_{\mathsf{ctrl}}(x,i) \triangleq \mathsf{Proj}_u\left(\mathcal{E}(\mathcal{C}_i), x\right) \cap \mathcal{E}(\mathcal{U}).
$$

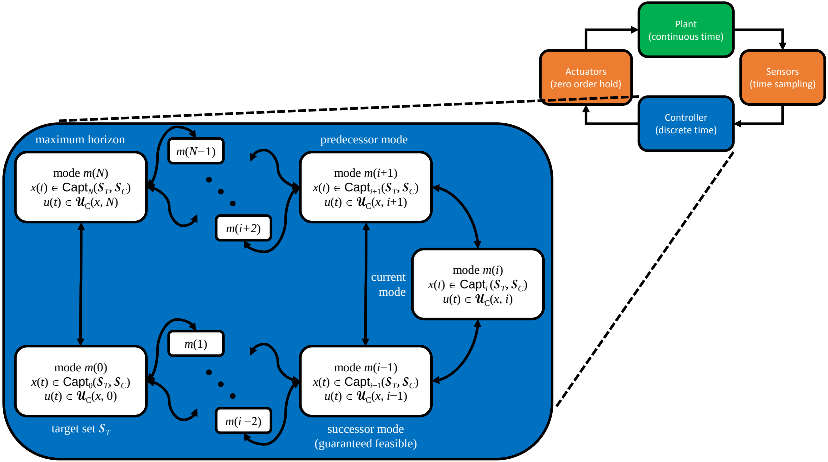
- All operations can be efficiently implemented for ellipsoids.

Outline

1. Motivation & Contributions
2. Constructs
3. Models & Algorithms
4. Control Filtering Hybrid Automaton
5. Quadrotor Flight Envelope Maintenance

Discrete Control Automaton Ensures Runtime Safety

Create a mode for each horizon $i = 0, 1, \ldots, \bar{N}$.



- Not every mode transition is shown; in fact, every mode is connected to every other node (including self-loops).

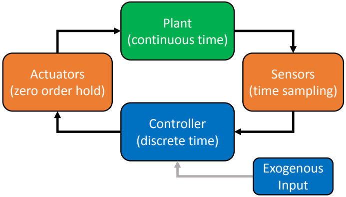
Look-Up Table Ensures Runtime Safety

Mode	Valid States	Safe Inputs
$m(\bar{N})$	$\mathsf{Capt}_{\bar{N}}\left(\mathcal{S}_T, \mathcal{S}_C\right)$	$\mathcal{U}_{\mathsf{ctrl}}(x, \bar{N})$
$\vdots$	$\vdots$	$\vdots$
$m(i+1)$	$\mathsf{Capt}_{i+1}\left(\mathcal{S}_T, \mathcal{S}_C\right)$	$\mathcal{U}_{\mathsf{ctrl}}(x, i+1)$
$m(i)$	$\mathsf{Capt}_{i}\left(\mathcal{S}_T, \mathcal{S}_C\right)$	$\mathcal{U}_{\mathsf{ctrl}}(x, i)$
$m(i-1)$	$\mathsf{Capt}_{i-1}\left(\mathcal{S}_T, \mathcal{S}_C\right)$	$\mathcal{U}_{\mathsf{ctrl}}(x, i-1)$
$\vdots$	$\vdots$	$\vdots$
$m(0)$	$\mathcal{S}_T$	$u_{\mathsf{fb}}^{\mathsf{inf}}(x)$

- Table data $\mathsf{Capt}_i\left(\mathcal{S}_T, \mathcal{S}_C\right)$ and $\mathcal{U}_{\mathsf{ctrl}}(x, i)$ are computed offline.
- At sample time t_k, choose a row for which $x(t_k)$ is in the valid states to find a safe set of input values.
- If $x(t_{k-1})$ was valid for mode $m(i)$, then $x(t_k)$ is guaranteed to be valid for mode $m(i-1)$.

Filtering an Exogenous Input

Let $\tilde{u}(\cdot)$ be the exogenous input signal.



- Upon choosing mode $m(i)$ at time t_k, let

$$u_{\text{pw}}(t) = \begin{cases} \tilde{u}(t_k), & \text{if } \tilde{u}(t_k) \in \mathcal{U}_{\text{ctrl}}(x(t_k), i); \\ \bar{u}, & \text{otherwise;} \end{cases}$$

- The clipped input $\bar{u} \in \mathcal{U}_{\text{ctrl}}(x, i)$ is chosen "near" the value $\tilde{u}(t_k)$ in some sense; for example

$$\bar{u} = q + \frac{\tilde{u}(t_k) - q}{\|\mathrm{L}(\tilde{u}(t_k) - q)\|_2}$$

where L is the Cholesky factorization of Q^{-1}, Q is the shape matrix for $\mathcal{U}_{\text{ctrl}}(x(t_k), i)$ and q is its center vector.

Other mechanisms for filtering the exogenous input are possible.

Related Work

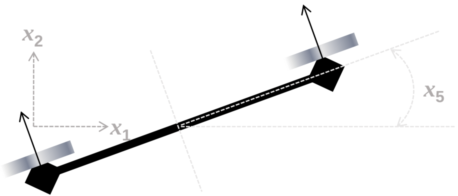
- Much work on traditional control objectives; for example [Goodwin et al, IEEE Control Systems Magazine 2013], [Karafyllis & Krstic, IEEE TAC 2012], [Monaco & Normand-Cyrot, Euro. J. Control 2007], [Nešić & Teel, IEEE TAC 2004].
- In [Tsuchie & Ushio, ADHS 2006]: Controller determines switches, more restrictive (but more realistic?) class of jitter, requires trajectory solutions.
- In [Karafylllis & Kravaris, Int. J. Control 2009]: Define r-robust reachability, but requires Lyapunov-like function.
- In [Simko & Jackson, HSCC 2014]: Taylor models and SMT solver, but only initial state is nondeterministic.
- In [Gillula, Kaynama & Tomlin, HSCC 2014]: Sampled data viability kernel (no disturbance input) with polytopic set representation.
- In [Aréchiga & Krogh, ACC 2014]: Theorem prover to verify invariants and control envelopes robust to parameter variations and sample time uncertainty.
- In [Kaynama, Michell, Oishi & Dumont, IEEE TAC 2015]: Discrete control automaton built from ellipsoidal approximations of discriminating kernels to ensure safety for continuous time systems.
- In [Dabadie, Kaynama & Tomlin, IROS 2014]: robust sampled data reach set is complement of (jitter-free) discriminating kernel.

Outline

Nonlinear Longitudinal Model of a Quadrotor

- From [Bouffard 2012]

$$
\begin{aligned}
\dot{x}_1 &= x_3, \\
\dot{x}_2 &= x_4, \\
\dot{x}_3 &= u_1 K \sin x_5, \\
\dot{x}_4 &= -g + u_1 K \cos x_5, \\
\dot{x}_5 &= x_6, \\
\dot{x}_6 &= -d_0 x_5 - d_1 x_6 + n_0 u_2,
\end{aligned}
$$



- Inputs: total thrust u_1 and desired roll angle u_2

Constraints

Safety constraint set $\mathcal{S}_C$:

$$\begin{aligned} x_1 &\in [-1.7, +1.7], \\ x_2 &\in [+0.3, +2.0], \\ x_3 &\in [-0.8, +0.8], \\ x_4 &\in [-1.0, +1.0], \\ x_5 &\in [-0.15, +0.15], \\ x_6 &\in [-\tfrac{\pi}{2}, +\tfrac{\pi}{2}]. \end{aligned}$$

LQR controller experimentally known to stablize from states in $\mathcal{S}_T$:

$$\begin{aligned} x_1 &\in [-1.2, +1.2], \\ x_2 &\in [+0.5, +1.7], \\ x_3 &\in [-0.5, +0.5], \\ x_4 &\in [-0.8, +0.8], \\ x_5 &\in [-0.1, +0.1], \\ x_6 &\in [-0.3, +0.3]. \end{aligned}$$

Linearized Model

- For ellipsoidal analysis, linearize dynamics about $\bar{u}_1$ and $\bar{x}_5$

$$
\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \\ \dot{x}_5 \\ \dot{x}_6 \end{bmatrix} = \overbrace{\begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2} K \bar{u}_1 \cos \bar{x}_5 & 0 \\ 0 & 0 & 0 & 0 & -\frac{1}{2} K \bar{u}_1 \sin \bar{x}_5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & -d_0 & -d_1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ K(\sin \bar{x}_5 - \frac{1}{2} \bar{x}_5 \cos \bar{x}_5) & 0 \\ K(\cos \bar{x}_5 + \frac{1}{2} \bar{x}_5 \sin \bar{x}_5) & 0 \\ 0 & 0 \\ 0 & n_0 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}}^{\text{linear}}
$$

$$
+ \underbrace{\begin{bmatrix} 0 \\ 0 \\ -\frac{1}{2} \bar{u}_1 K(\bar{x}_5 \cos \bar{x}_5) \\ \frac{1}{2} \bar{u}_1 K(\bar{x}_5 \sin \bar{x}_5) - g \\ 0 \\ 0 \end{bmatrix}}_{\text{constant}} + \underbrace{\begin{bmatrix} 0 \\ 0 \\ \frac{1}{2} K x_5 u_1 \cos \bar{x}_5 - \frac{1}{2} K (x_5 - \bar{x}_5)^2 \bar{u}_1 \sin \xi \\ -\frac{1}{2} K x_5 u_1 \sin \bar{x}_5 - \frac{1}{2} K (x_5 - \bar{x}_5)^2 \bar{u}_1 \cos \xi \\ 0 \\ 0 \end{bmatrix}}_{\text{linearization error}}
$$

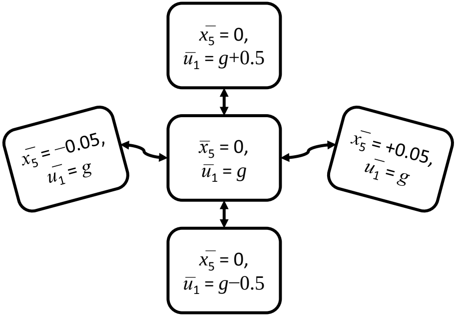
for some ξ in the range of possible values of x_5.

- Compute capture basins robust to bound on the linearization error.

Hybridization to Reduce Error Bound

- Leading error term is $\frac{1}{2} K x_5 u_1 \cos \bar{x}_5$.
- To reduce range of error, use hybrid model with values of $\bar{u}_1 \in \{g - 0.5, g, g + 0.5\}$ and $\bar{x}_5 \in \{-0.05, 0.00, +0.05\}$ for each mode.
- Adjust $\mathcal{S}_C$ and range of inputs for each model hybridization mode as well.

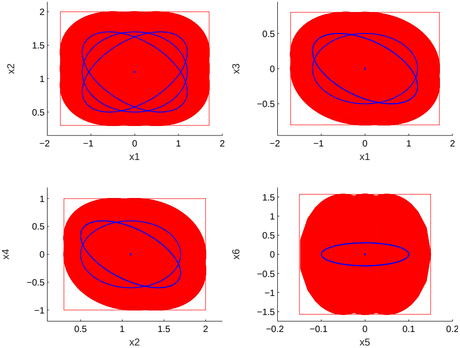
$$x_5 \in [-0.1, +0.1] + \bar{x}_5$$
$$u_1 \in [-0.5, +0.5] + \bar{u}_1$$
$$u_2 \in \left[-\tfrac{\pi}{16}, +\tfrac{\pi}{16}\right] + \bar{x}_5$$



Capture Basin Calculation

- Create three pairs of $\mathcal{S}_C$ and $\mathcal{S}_T$ to better fill box constraints with ellipsoids.
- Could also use multiple direction vectors for ellipsoidal reachability calculations, but a single vector did a good job.

$$\ell = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}^T$$



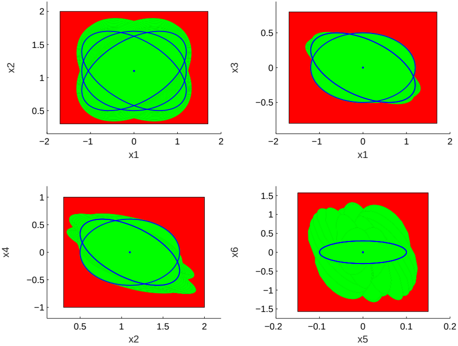
Red line: $\mathcal{S}_C$
Red fill: $\mathcal{E}(\mathcal{S}_C)$
Blue line $\mathcal{E}(\mathcal{S}_T)$

Capture Basin Results

Compute capture basin approximations over

- 5 hybridization modes.
- 3 constraint set approximations.
- 1 direction vector.
- 10 sample periods with $\delta = 0.1$s.

Computation takes ~ 15s for each combination of mode, constraint set and direction vector over 10 sample periods.



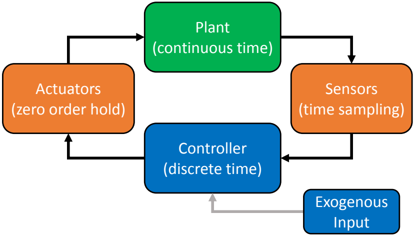
Red fill: $\mathcal{S}_C$
Green fill: $\bigcup_{i=0}^{10} \mathcal{E}_i$
Blue line: $\mathcal{E}(\mathcal{S}_T)$

Runtime Application

Compare exogenous pilot input with $\mathcal{U}_C(x(t), m)$ for several modes m.

Heuristic for selecting modes:

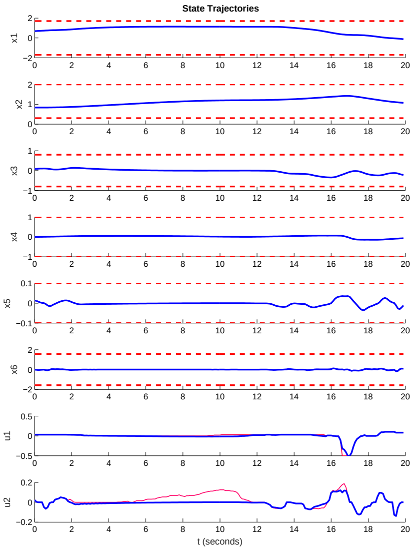
- Current hybridization and constraint set with horizons $\{i-1, i, i+1, i+2\}$ (4 modes).
- Current horizon i with all hybridizations and contraint sets (14 modes).



- If pilot input is inside $\mathcal{U}_C(x(t), m)$, choose m with largest horizon.
- If pilot input is not inside, choose m that gets closest and project input onto $\mathcal{U}_C(x(t), m)$.

Runtime Results

- Each mode comparison requires evaluating a quadratic function (18 modes takes ~ 0.03s).
- Input u_2 is clipped for $t \in [6, 12]$ because of threat of exceeding bounds on x_1.
- Input u_2 is allowed much higher value for $t \approx 16$ without clipping.
- LQR controller is not invoked for $t \in [0, 20]$ even though capture basin horizon is $T = 1$.



Limitations

- No formal proof of LQR controller's infinite horizon safety.
- Worst case treatment of linearization error leads to overly conservative results.
- Ellipsoids offer poor approximation of boxes, which leads to overly conservative results.
- Algorithm does not account for feedback delay or state uncertainty.
- Input clipping may not be the appropriate shared control strategy.

Conclusions & Future Work

In this paper we

- Described a method to construct a control automaton / look-up table returning set-valued safe control inputs for a sampled data system.
- Implemented an efficient algorithm constrained to linear dynamics but able to handle some nonlinearity through robust analysis.
- Demonstrated technique on a six dimensional nonlinear longitudinal quadrotor model with a human-in-the-loop pilot providing an exogenous input signal.

In the future we plan to

- Investigate methods to handle realistic signal delay and timing jitter.
- Seek more accurate representations able to handle more general dynamics.
- Adapt techniques to learned models.
- Identify methods of sharing control which humans find more suitable than clipping.