Ensuring Safety for Sampled Data Systems
An Efficient Algorithm for Filtering Potentially Unsafe Input Signals

Ian M. Mitchell\textsuperscript{1}, Jeffrey Yeh\textsuperscript{1}, Forrest J. Laine\textsuperscript{2}, Claire J. Tomlin\textsuperscript{2}

\textsuperscript{1}Department of Computer Science
The University of British Columbia
\textsuperscript{2}Department of Electrical Engineering & Computer Science
University of California, Berkeley

December 2016

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

Copyright 2016 by Ian M. Mitchell
This work is made available under the terms of the Creative Commons Attribution 4.0 International license
http://creativecommons.org/licenses/by/4.0/
Motivation: Sampled Data Systems

A common design pattern for cyber-physical systems:

- **Plant** (continuous time)
- **Controller** (discrete time)
- **Sensors** (time sampling)
- **Actuators** (zero order hold)

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

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

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

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

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

\[
\text{Capt}([t_s, t_f], S_T, S_C) \triangleq \left\{ x(t_s) \in S_C \mid \exists u(\cdot), \exists t_T \in [t_s, t_f], \forall t \in [t_s, t_T], x(t) \in S_C \land x(t_T) \in S_T \right\},
\]

- Trajectories must stay inside constraint $S_C$ until they reach target $S_T$.
- Inputs treated in a best-case fashion.
Robust Reach Set

\[
\text{Reach}([t_s, t_f], S) \triangleq \{ x(t_s) \in \Omega \mid \forall \nu(\cdot), x(t_f) \in S \}
\]

- Not a reach tube: Trajectories must reach \( S \) at exactly \( t_f \).
- Reach tube may not be the union of these reach sets [Mitchell 2007].
Discriminating Kernel

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

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.

- 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:

\[ x(t + 1) = f(x(t), u(t), v(t)) \]

general dynamics

\[ x(t + 1) = Ax(t) + Bu(t) + Cv(t) \]

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:

\[ \dot{x}(t) = f(x(t), u(t), v(t)) \]

general dynamics

\[ \dot{x}(t) = Ax(t) + Bu(t) + Cv(t) \]

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 $\rho$ be a small computational timestep and $M$ a uniform bound on $f$.

• Start with an under-approximation $\mathcal{K}_\downarrow$ of $\mathcal{K}$

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

• Iteratively compute $K_{n+1}$:

$$\mathcal{K}_0 = \mathcal{K}_\downarrow,$$

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

• 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 $\rho$ be a small computational timestep and $M$ a uniform bound on $f$.
- Start with an under-approximation $K_\downarrow$ of $K$

\[ K_\downarrow := \{ x \in K \mid \text{dist}(x, K^c) \geq \rho M \} \]

- Iteratively compute $K_{n+1}$:

\[
K_0 = K_\downarrow, \\
K_{n+1}(P) = K_0 \cap \text{Reach}([0, \rho], K_n)
\]

- Discriminating kernel algorithm is straightforward, albeit notationally complicated.
- Discrete time algorithm omits initial erosion: $K_0 = K$. 
Continuous-Time Viability Algorithm

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

- Let $\rho$ be a small computational timestep and $M$ a uniform bound on $f$.
- Start with an under-approximation $K_\downarrow$ of $K$
  \[
  K_\downarrow := \{ x \in K \mid \text{dist}(x, K^c) \geq \rho M \}
  \]
- Iteratively compute $K_{n+1}$:
  \[
  K_0 = K_\downarrow, \\
  K_{n+1}(P) = K_0 \cap \text{Reach}([0, \rho], K_n)
  \]
- Discriminating kernel algorithm is straightforward, albeit notationally complicated.
- Discrete time algorithm omits initial erosion: $K_0 = K$. 

Continuous-Time Viability Algorithm

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

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

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

• Iteratively compute $K_{n+1}$:

$$K_0 = K_{\downarrow},$$
$$K_{n+1}(P) = K_0 \cap \text{Reach}([0, \rho], K_n)$$

• Discriminating kernel algorithm is straightforward, albeit notationally complicated.
• Discrete time algorithm omits initial erosion: $K_0 = K$. 
Continuous-Time Viability Algorithm

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

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

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

• Iteratively compute $\mathcal{K}_{n+1}$:

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

• 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 $\rho$ be a small computational timestep and $M$ a uniform bound on $f$.
- Start with an under-approximation $K_\downarrow$ of $K$
  \[
  K_\downarrow := \{ x \in K \mid \text{dist}(x, K^c) \geq \rho M \} 
  \]
- Iteratively compute $K_{n+1}$:
  \[
  K_0 = K_\downarrow, \\
  K_{n+1}(P) = K_0 \cap \text{Reach}([0, \rho], K_n) 
  \]
- Discriminating kernel algorithm is straightforward, albeit notationally complicated.
- Discrete time algorithm omits initial erosion: $K_0 = K$. 
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.
• Use continuous time model of the plant
\[ \dot{x}(t) = f(x(t), u(t), v(t)) \] general dynamics;
\[ \dot{x}(t) = Ax(t) + Bu(t) + Cv(t) \] linear dynamics.

• However, control input is piecewise constant in time
\[ u_{pw}(t) = u_{fb}(x(t_k)) \text{ for } t_k \leq t < t_{k+1} \]
where \( u_{fb} : \Omega \to \mathcal{U} \) 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
  \[
  \tilde{x} \triangleq \begin{bmatrix} x \\ u \end{bmatrix} \quad \tilde{f}(\tilde{x}, v) \triangleq \begin{bmatrix} f(x, u, v) \\ 0 \end{bmatrix}.
  \]
- Move between original and augmented state space with tensor products and projections
  \[
  \text{Proj}_x (\tilde{x}) \triangleq \left\{ x \in \Omega \mid \exists u, \begin{bmatrix} x \\ u \end{bmatrix} \in \tilde{x} \right\},
  \]
  \[
  \text{Proj}_u (\tilde{x}, x) \triangleq \left\{ u \in \mathcal{U} \mid \begin{bmatrix} x \\ u \end{bmatrix} \in \tilde{x} \right\}.
  \]
Finite Horizon Sampled Data Capture Basin

Define

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

$$\text{Capt}_{sd} ([0, T], S_T, S_C) \triangleq \left\{ x_0 \in S_C \mid \exists u_{pw}(\cdot), \exists i \in \{0, 1, \ldots, \bar{N}\}, \right.$$ 
$$\forall v(\cdot), \forall t \in [0, i\delta],$$ 
$$x(t) \in S_C \land x(i\delta) \in S_T \right\}.$$ 

If a safe infinite horizon feedback controller $u_{fb}^{\inf}(x)$ is available for $x \in S_T$, then capture basin is also infinite horizon safe.
Capture Basin Algorithm

• For \( i = 1, 2, \ldots, \bar{N} \)

\[
\mathcal{E}_i \triangleq \mathcal{E}(\text{Capt}_i (\mathcal{S}_T, \mathcal{S}_C)) \\
\mathcal{E}_0 = \mathcal{E}(\mathcal{S}_T) \\
\mathcal{E}(\mathcal{I}_1) \triangleq \mathcal{E}(\text{Inv} ([0, \delta], \mathcal{S}_C \times \mathcal{U})), \\
\mathcal{E}(\mathcal{R}_i) \triangleq \mathcal{E}(\text{Reach} ([0, \delta], \mathcal{E}_{i-1} \times \mathcal{U})), \\
\mathcal{E}(\mathcal{C}_i) \triangleq \text{Inscribed}_\alpha (\mathcal{E}(\mathcal{R}_i) \cap \mathcal{E}(\mathcal{I}_1)), \\
\mathcal{E}_i = \text{Proj}_x (\text{Inscribed}_0 (\mathcal{E}(\mathcal{C}_i) \cap \mathcal{E}(\Omega \times \mathcal{E}(\mathcal{U})))) ,
\]

• Overapproximates the sampled data capture basin

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

• Provides a safe control policy

\[
\mathcal{U}_{ctrl}(x, i) \triangleq \text{Proj}_u (\mathcal{E}(\mathcal{C}_i), x) \cap \mathcal{E}(\mathcal{U}).
\]

• 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
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

<table>
<thead>
<tr>
<th>Mode</th>
<th>Valid States</th>
<th>Safe Inputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>$m(\bar{N})$</td>
<td>Capt$_\bar{N}$ ($S_T, S_C$)</td>
<td>$U_{ctrl}(x, \bar{N})$</td>
</tr>
<tr>
<td>$m(i+1)$</td>
<td>Capt$_{i+1}$ ($S_T, S_C$)</td>
<td>$U_{ctrl}(x, i+1)$</td>
</tr>
<tr>
<td>$m(i)$</td>
<td>Capt$_i$ ($S_T, S_C$)</td>
<td>$U_{ctrl}(x, i)$</td>
</tr>
<tr>
<td>$m(i-1)$</td>
<td>Capt$_{i-1}$ ($S_T, S_C$)</td>
<td>$U_{ctrl}(x, i-1)$</td>
</tr>
<tr>
<td>$m(0)$</td>
<td>$S_T$</td>
<td>$u_{fb}^\inf(x)$</td>
</tr>
</tbody>
</table>

- Table data Capt$_i$ ($S_T, S_C$) and $U_{ctrl}(x, i)$ 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

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

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

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

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

Other mechanisms for filtering the exogenous input are possible.
Related Work

- 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

1. Motivation & Contributions

2. Constructs

3. Models & Algorithms

4. Control Filtering Hybrid Automaton

5. Quadrotor Flight Envelope Maintenance
Nonlinear Longitudinal Model of a Quadrotor

From [Bouffard 2012]

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

• Inputs: total thrust \( u_1 \) and desired roll angle \( u_2 \)
Constraints

Safety constraint set $S_C$:

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

LQR controller experimentally known to stabilize from states in $S_T$:

$x_1 \in [-1.2, +1.2]$,  
$x_2 \in [+0.5, +1.7]$,  
$x_3 \in [-0.5, +0.5]$,  
$x_4 \in [-0.8, +0.8]$,  
$x_5 \in [-0.1, +0.1]$,  
$x_6 \in [-0.3, +0.3]$.
For ellipsoidal analysis, linearize dynamics about $\bar{u}_1$ and $\bar{x}_5$.

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

constant

linearization error

for some $\xi$ 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 \( S_C \) and range of inputs for each model hybridization mode as well.

\[
\begin{align*}
x_5 &\in [-0.1, +0.1] + \bar{x}_5 \\
u_1 &\in [-0.5, +0.5] + \bar{u}_1 \\
u_2 &\in \left[-\frac{\pi}{16}, +\frac{\pi}{16}\right] + \bar{x}_5
\end{align*}
\]
Capture Basin Calculation

- Create three pairs of $S_C$ and $S_T$ 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.

$$\ell = [0 \ 0 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 0]^T$$

Red line: $S_C$
Red fill: $\mathcal{E}(S_C)$
Blue line $\mathcal{E}(S_T)$
Capture Basin Results

Compute capture basin approximations over

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

Computation takes $\sim 15\text{s}$ for each combination of mode, constraint set and direction vector over 10 sample periods.

Red fill: $S_C$
Green fill: $\bigcup_{i=0}^{10} E_i$
Blue line: $E(S_T)$
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 constraint 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 $\sim 0.03s$).
- 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.