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

Ian M. Mitchell, Jeffrey Yeh, Forrest J. Laine, Claire J. Tomlin

***Abstract*— A common design pattern in cyber-physical systems features a continuous plant and a discrete controller in a feedback loop. Sampled data analysis attempts to take into consideration both the continuous and discrete time elements of such a design. In this paper we adapt an earlier algorithm for efficient ellipsoidal approximation of robust sampled data finite horizon viability kernels to compute capture basins for systems with linear dynamics. Using these capture basins, we construct a hybrid automaton which can verify and if necessary modify an exogenous input signal to ensure safety. The hybrid automaton can be run online in the controller so that it can handle exogenous input signals arriving in real time, such as might be generated by human-in-the-loop control. The technique is demonstrated on a six dimensional nonlinear longitudinal model of a quadrotor with a human pilot in the loop. The capture basins' robustness is used to handle the model nonlinearity in a sound fashion.**

I. Introduction

The design of cyber-physical systems often follows the basic block diagram shown in figure 1, where a continuous time physical plant is controlled by a discrete time cyber controller and the interface between the two is provided by periodically sampled sensors and actuators that maintain a constant setting over each sample period. Traditional approaches to safety verification of such systems often ignore important behaviours: a discrete time analysis can miss failures caused by the continuous evolution of the plant between sample times, while a continuous time analysis may fail to account for the sampled nature of the feedback loop.

In previous work [1], [2] we have proposed algorithms to compute viability and discriminating kernels for sampled data systems; however, the main goal was to determine from which states it was possible to remain safe. In this paper we focus on a more proactive approach: We construct a hybrid automaton which can generate a set of known safe input values at every plant state known to be safe. If in addition to the plant state the controller has access to an exogenous input signal—such as might be generated by a human-in-the-loop—this set of known safe input values can be used to check and if necessary modify the exogenous signal to ensure safety.

This work was supported in part by National Science and Engineering Council of Canada (NSERC) Discovery Grant #298211 (IMM & JY), an NSERC Undergraduate Student Research Award (JY), and ONR under grants N00014-12-1-0609 and N000141310341 (Embedded Humans MURI) (FJL & CJT). Author Affiliations are the Department of Computer Science, the University of British Columbia, Vancouver, Canada (IMM & JY) and the Department of Electrical Engineering & Computer Science, University of California, Berkeley, California (FJL & CJT). Email addresses are mitchell@cs.ubc.ca, jeff_8514@hotmail.com, forrest.laine@berkeley.edu and tomlin@eecs.berkeley.edu.

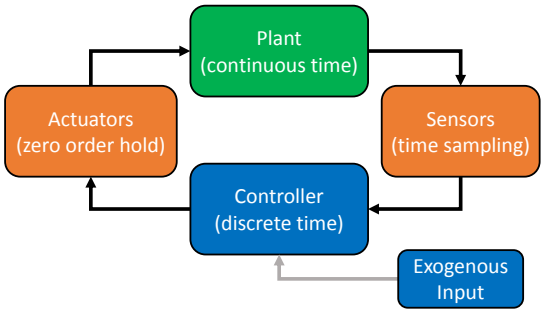

Fig. 1. A closed loop sampled data system. In this paper we assume that a continuous time ordinary differential equation model of the plant is given, and we focus on verification and/or synthesis of discrete time controllers which ensure safety of the plant at all (continuous) times.

A second shortcoming of the previous work was that safety was guaranteed only over a finite horizon. While the computational algorithms proposed below continue to apply over a finite horizon, we propose to ensure infinite-horizon safety of the resulting system using a two-stage process akin to that frequently used to achieve infinite-horizon stability through model predictive control (MPC) [3]: Design a stabilizing controller for some (relatively small) subset of the safety constraints, and then compute a finite horizon capture basin within the safety constraints using this stabilizable subset as a target. In the common case where the capture basin is larger than the target, we are able to ensure safety over an infinite horizon by utilizing the input signals derived from the capture basin calculation to drive the plant into the stabilizable region. In fact, in some situations it may never be necessary to invoke the stabilizing controller.

The contributions of this paper are:

- Adaptation of the sampled data discriminating kernel algorithm [1] to robust fixed time capture basins. We describe the adaptation for systems with linear dynamics because we can efficiently implement it with ellipsoidal set representations.
- Construction of a discrete state automaton for the discrete time controller which, when combined with the continuous time plant to form a hybrid automaton, can verify and if necessary filter an exogenous input signal to ensure infinite horizon safety, provided that a safe stabilizing controller can be designed for some subset of the safety constraints.
- Demonstration of these techniques on a partially nonlinear longitudinal model of a quadrotor with six state dimensions and two input dimensions.

A. Related Work

Since we are borrowing a technique from MPC to achieve infinite horizon guarantees, why not use MPC to directly tackle the safety constraint problem in the first place? First, our implementation is efficient compared to the optimizations typically required by MPC; for example, the offline computation for the quadrotor requires on the order of minutes to complete, while the online effort involves evaluating a handful of quadratic functions at each sample time. Second, the proposed hybrid automaton is a representation of a *verified control envelope*, which could be used to more efficiently design, modify or tune proposed controllers to ensure safety [4].

In addition to the algorithms from [1], [2] that we extend here, one of us (CJT) has also studied alternative mechanisms for verifying sampled data systems. In [5] a sampling based approach to approximation of sampled data viability kernels for systems with linear dynamics is considered. That scheme is more accurate than the ellipsoidal implementations used here, but cannot yet provide results robust to uncertainty in the dynamics or synthesize control signals. The paper [6] focuses on a ground robot collision avoidance scenario; although the Hamilton-Jacobi techniques used therein are very similar to the high level ideas outlined in [2], successful application to a particular problem requires working out many details (we tackle a similar challenge in the example below). In [7] another one of us (IMM) used ellipsoidal approximations of discriminating kernels to construct a hybrid control automaton to ensure system safety; however, that system assumed a continuous time controller.

In [8] the authors combine SMT and Taylor models to bound the reach tubes of sampled data hybrid systems, and apply their technique to models of adaptive cruise control, glucose control through insulin, and watertanks. The approach is more rigorous numerically, but it is not clear how it might be adapted to synthesize controllers or operate online to analyze exogenous input signals.

In [9] the authors construct outer approximations of the capture basin (called therein the region of attraction) by solving (in the end) a single semi-definite program. Their approach, based on occupational measures, allows treatment of a much more general class of dynamics but does not provide inner approximations or handle disturbance inputs.

We note that there has been considerable work done on stabilization of sampled data systems; for example, see [10] and the citations therein. These techniques are less relevant to the problem of interest in this paper because they cannot easily enforce general state and input constraints.

B. Problem Definition

We will focus in this paper on systems with linear dynamics because we are able to efficiently implement the resulting capture basin algorithm. We assume that the plant's dynamics take the form

$$\dot{x} = \mathrm{A}x + \mathrm{B}u + \mathrm{C}v \tag{1}$$

where $x \in \Omega \subseteq \mathbb{R}^{d_x}$ is the state, $u \in \mathcal{U}$ is the control input constrained to compact subset $\mathcal{U} \subset \mathbb{U}$ of a control input subspace $\mathbb{U} \subseteq \mathbb{R}^{d_u}$, $v \in \mathcal{V}$ is the disturbance input constrained to compact set $\mathcal{V} \subseteq \mathbb{R}^{d_v}$, and the matrices A, B and C are known.

We study a safety problem in which the control input seeks to keep the plant state within a set of prespecified constraints $\mathcal{S}_C \subset \Omega$. The disturbance input is used to model uncertainty, nonlinearity or error in the dynamics. We assume that it seeks to drive the plant state outside of $\mathcal{S}_C$; in other words, that it adversarially chooses the worst possible value for safety. In this manner we ensure that the resulting analysis is robust to any modeled disturbance input.

While the plant evolves in continuous time and state according to (1), as shown in figure 1 the controller only receives state feedback and can set the control signal at sampling times $t_k = k\delta$ for some sample time period $\delta > 0$. For simplicity we only consider fixed sample time period δ in this paper, although it is straightforward to use the techniques from [1] to adapt the algorithms described below to handle some forms of sample time jitter. Taking into account the sample times, the closed loop dynamics of the system take the form

$$\dot{x}(t) = \mathrm{A}x(t) + \mathrm{B}u_{\text{pw}}(t) + \mathrm{C}v(t) \tag{2}$$

where the piecewise constant input signal $u_{\text{pw}}(\cdot)$ is chosen according to

$$u_{\text{pw}}(t) = u_{\text{fb}}(x(t_k)) \text{ for } t_k \leq t < t_{k+1} \tag{3}$$

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

In order to extend our safety guarantee to an infinite horizon, we will assume the existence of a feedback controller $u_{\text{fb}}^{\text{inf}} : \mathcal{S}_C \to \mathcal{U}$ which can ensure safety for the sampled data system if $x \in \mathcal{S}_T \subset \mathcal{S}_C$; for example, $u_{\text{fb}}^{\text{inf}}(x)$ might be a controller which stabilizes to an equilibrium in $\mathcal{S}_T$. For technical reasons [11], we assume $\mathcal{S}_C$ and $\mathcal{S}_T$ are the complements of open sets.

In the rest of the paper, algorithms will be described as *online* if they are designed to run in the controller block (and must hence satisfy its periodic cycle time), or *offline* if they can be run in advance and provide fixed data to be incorporated into the controller block. Offline algorithms are assumed to have access to the plant model (1), sample period δ, constraint set $\mathcal{S}_C$ and target set $\mathcal{S}_T$. Online algorithms will have access to this information, the current state $x(t)$, and must also be able to evaluate the exogenous input signal and possibly the feedback controller $u_{\text{fb}}^{\text{inf}}(x)$ at the sample times.

II. Computing Capture Basins

In this section we adapt the algorithm for sampled data discriminating kernels from [1] to robust sampled data capture basins. As in [1] an abstract version can be formulated for general nonlinear dynamics, but here we present only the version for linear systems which uses the efficient and conservative ellipsoidal representation of sets. The algorithm in this section is intended to be run offline, and scales polynomially (roughly cubicly) with $d_x + d_u$ and linearly with the time horizon.

A. Prelminary Definitions

Define the robust fixed time sampled data capture basin

$$\mathsf{Capt}_{\mathsf{sd}}\left([0,T],\mathcal{S}_T,\mathcal{S}_C\right) \triangleq \left\{ x_0 \in \mathcal{S}_C \;\middle|\; \begin{array}{c} \exists u_{\text{pw}}(\cdot), \exists i \in \{0,1,\ldots,\bar{N}\}, \\ \forall v(\cdot), \forall t \in [0, i\delta], \\ x(t) \in \mathcal{S}_C \wedge x(i\delta) \in \mathcal{S}_T \end{array} \right\}, \quad (4)$$

where $T = \bar{N}\delta$ and $x(\cdot)$ solves (2) with initial condition $x(0) = x_0$. We call this construct a "fixed time" capture basin because we fix the maximum time at which the trajectory must achieve the target set $\mathcal{S}_T$. The sampled data capture basin must also achieve the target set at a sample time; passing through the target set between sample times is insufficient. This fixed time sampled data construct is in contrast to the standard capture basin (trajectories may reach the target set at any finite time) or the viability / discriminating kernel with target (trajectories may either reach the target set at any finite time or remain within the constraint $\mathcal{S}_C$ for infinite time) [11].

We now repeat some definitions from [1], [2]. The approximation of (4) is performed in an augmented state space

$$\tilde{x} \triangleq \begin{bmatrix} x \\ u \end{bmatrix} \in \tilde{\Omega} \triangleq \Omega \times \mathbb{U}$$

with dynamics

$$\frac{d}{dt}\tilde{x} = \frac{d}{dt}\begin{bmatrix} x \\ u \end{bmatrix} = \begin{bmatrix} \mathrm{A}x(t) + \mathrm{B}u_{\text{pw}}(t) + \mathrm{C}v(t) \\ 0 \end{bmatrix}. \quad (5)$$

Projection operators move from $\tilde{\Omega}$ back into Ω or $\mathbb{U}$

$$\mathsf{Proj}_x\left(\tilde{\mathcal{X}}\right) \triangleq \left\{ x \in \Omega \;\middle|\; \exists u, \begin{bmatrix} x \\ u \end{bmatrix} \in \tilde{\mathcal{X}} \right\}, \quad (6)$$

$$\mathsf{Proj}_u\left(\tilde{\mathcal{X}}, x\right) \triangleq \left\{ u \in \mathbb{U} \;\middle|\; \begin{bmatrix} x \\ u \end{bmatrix} \in \tilde{\mathcal{X}} \right\}, \quad (7)$$

for $\tilde{\mathcal{X}} \subseteq \tilde{\Omega}$ and $x \in \Omega$. We apply two reachability constructs in this augmented state space: An invariance kernel

$$\mathsf{Inv}\left([t_s,t_f],\mathcal{S}\right) \triangleq \left\{ \tilde{x}(t_s) \in \mathcal{S} \;\middle|\; \begin{array}{c} \forall v(\cdot), \forall t \in [t_s,t_f], \\ \tilde{x}(t) \in \mathcal{S} \end{array} \right\} \quad (8)$$

and a robust reach set

$$\mathsf{Reach}\left([t_s,t_f],\mathcal{S}\right) \triangleq \{\tilde{x}(t_s) \in \tilde{\Omega} \mid \forall v(\cdot), \tilde{x}(t_f) \in \mathcal{S}\}. \quad (9)$$

Note that this latter construct is *not* a reach tube.

Sets are represented by ellipsoids. An ellipsoid $\mathcal{E} \subset \mathbb{R}^d$ is parameterized by center vector $q \in \mathbb{R}^d$ and symmetric positive definite shape matrix $\mathrm{Q} \in \mathbb{R}^{d\times d}$ as the set

$$\{y \in \mathbb{R}^d \mid (y-q)^T \mathrm{Q}^{-1}(y-q) \leq 1\},$$

Let $\mathcal{E}(\mathcal{S})$ denote the ellipsoidal approximation of a set $\mathcal{S}$. In order to ensure conservativeness, we choose an ellipsoidal *over*approximation of the set of possible disturbance inputs $\mathcal{V}$ (in other words, $\mathcal{V} \subseteq \mathcal{E}(\mathcal{V})$), while all other sets will be *under*approximated (for example, $\mathcal{E}(\mathcal{U}) \subseteq \mathcal{U}$).

Finally, we use the operator $\mathsf{Inscribed}_\alpha\left(\cap_i \mathcal{Y}_i\right)$ to compute a maximum *weighted* trace ellipsoid contained within the intersection of a set of ellipsoids $\{\mathcal{Y}_i\}$ which lie in the space $\Omega \times \mathbb{U}$. The weight factor $\alpha \in [0,1]$ determines whether the result should favour dimensions in Ω ($\alpha \to 0$), dimensions in $\mathbb{U}$ ($\alpha \to 1$), or all dimensions equally ($\alpha \to 0.5$). For a full definition of this operator and how to implement it with a convex optimization, see [1].

B. Capture Basin Algorithm

We adapt the iterative algorithm from [1] to compute the finite time robust sampled data capture basin.

$$\mathcal{E}_i \triangleq \mathcal{E}(\mathsf{Capt}_i\left(\mathcal{S}_T,\mathcal{S}_C\right)) \quad (10a)$$

$$\mathcal{E}_0 = \mathcal{E}(\mathcal{S}_T) \quad (10b)$$

$$\mathcal{E}(\mathcal{I}_1) \triangleq \mathcal{E}(\mathsf{Inv}\left([0,\delta],\mathcal{S}_C \times \mathbb{U}\right)), \quad (10c)$$

$$\mathcal{E}(\mathcal{R}_i) \triangleq \mathcal{E}(\mathsf{Reach}\left([0,\delta],\mathcal{E}_{i-1} \times \mathbb{U}\right)), \quad (10d)$$

$$\mathcal{E}(\mathcal{C}_i) \triangleq \mathsf{Inscribed}_\alpha\left(\mathcal{E}(\mathcal{R}_i) \cap \mathcal{E}(\mathcal{I}_1)\right), \quad (10e)$$

$$\mathcal{E}_i = \mathsf{Proj}_x\left(\mathsf{Inscribed}_0\left(\mathcal{E}(\mathcal{C}_i) \cap \mathcal{E}(\Omega \times \mathcal{E}(\mathcal{U}))\right)\right), \quad (10f)$$

for $i = 1,2,\ldots,\bar{N}$. We note in passing that the ellipsoidal underapproximations of the invariance kernel and robust reach set are parameterized by a direction vector $\ell \in \mathbb{R}^{d_x+d_u}$, and so given a set of such vectors the algorithm above can be used separately on each vector to generate a set of ellipsoids whose *union* is an underapproximation of the capture basin. To avoid notational complexity we omit the direction vector parameter in the remainder of the discussion.

The only significant difference between this capture basin version and the discriminating kernel version in [1] is that computations start from the target set $\mathcal{S}_T$ instead of the entire constraint set $\mathcal{S}_C$. However, this small change has the important implication that intermediate sets are not monotonic in time: It is not necessarily true that $\mathcal{E}_i$ contains or is contained by $\mathcal{E}_{i+1}$. Because of this (lack of) relationship, we define a set-valued capture horizon function $\mathcal{N} : \Omega \to \mathcal{P}(\{0,1,2,\ldots,\bar{N}\})$, where $\mathcal{P}(\mathcal{S})$ is the power set of $\mathcal{S}$

$$i \in \mathcal{N}(x) \text{ iff } x \in \mathcal{E}_i. \quad (11)$$

For any $i \in \mathcal{N}(x)$, a control policy is given by

$$\mathcal{U}_{\text{c}}(x,i) \triangleq \mathsf{Proj}_u\left(\mathcal{E}(\mathcal{C}_i),x\right) \cap \mathcal{E}(\mathcal{U}). \quad (12)$$

The proofs of the following two claims about the capture basin approximations are straightforward modifications of the proofs in [1] about discriminating kernel approximations.

Lemma 1: If $x(t_k) \in \mathcal{E}_i$ for some $i \in 1,\ldots,\bar{N}$, then applying any $u_{\text{pw}}(t) \in \mathcal{U}_{\text{c}}(x(t_k),i)$ for $t_k \leq t < t_{k+1}$ will result in $x(t_{k+1}) \in \mathcal{E}_{i-1}$ for any disturbance input $v(\cdot)$.

Proposition 2: If $T = \bar{N}\delta$ for some integer $\bar{N} > 0$ then

$$\bigcup_{i=0}^{\bar{N}} \mathcal{E}_i \subseteq \mathsf{Capt}_{\mathsf{sd}}\left([0,T],\mathcal{S}_T,\mathcal{S}_C\right) \quad (13)$$

III. Control Signal Filtering

In this section we use the capture basin computed in the previous section to analyze and if necessary modify an exogenous input signal $\tilde{u} \in \mathbb{U}$. The algorithm in this section is designed to be run online so that it could be applied to an exogenous signal only available in real-time, such as might

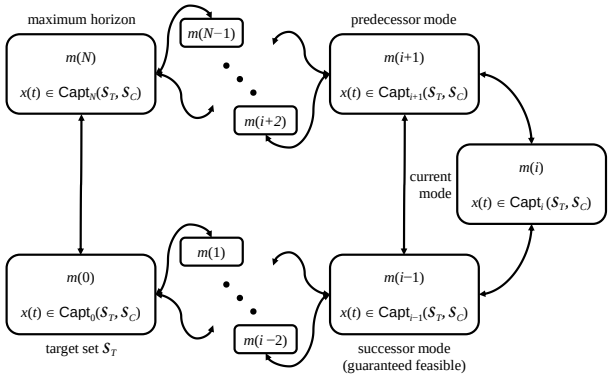

Fig. 2. Discrete automaton for the controller block. Not every mode transition is shown; in fact, every mode is connected to every other mode (including self-loops). When combined with the continuous plant (following figure 1), the result is the control filtering hybrid automaton (CFHA).

come from a human-in-the-loop scenario; however, it could also be used in a more comprehensive offline analysis such as proposed for verified control envelopes [4].

For notational convenience, we will extend the definition of $\mathcal{U}_{\mathrm{c}}(x,i)$ from (12) to include $i=0$:

$$\mathcal{U}_{\mathrm{c}}(x, i=0) \triangleq u_{\mathrm{fb}}^{\mathrm{inf}}(x), \tag{14}$$

where $u_{\mathrm{fb}}^{\mathrm{inf}}(x)$ is the feedback controller which ensures the infinite horizon safety of the system for $x \in \mathcal{S}_T$.

Because the capture basins are defined by the union of a discrete collection of sets, we find it convenient to implement our controller as a discrete time, discrete state automaton. Note that this controller automaton requires the periodically sampled state $x(t_k)$ at $t_k = k\delta$ as input, and generates a control value $u(t_k)$ as output. A graphical sketch of the automaton is shown in figure 2, and it consists of:

- Discrete modes: One for each sample time in the fixed horizon plus one for the end of the horizon. We denote modes by $m(i)$ for $i = 0, 1, \ldots, \bar{N}$.
- Invariants: Mode $m(i)$ has invariants S_T for $i = 0$ and $\mathcal{E}_i$ for $i = 1, 2, \ldots, \bar{N}$.
- Edges: Every pair of modes is joined by an edge (so that we do not have to explicitly keep track of edges). Note that edges may be infeasible if the outgoing mode's invariant does not overlap the edge's guard. Every mode also has a self-loop.
- Guards: The guard for an edge entering mode $m(i)$ is the invariant of mode $m(i)$.

Note that the discrete evolution of this automaton will be nondeterministic if there exists any i and $\tilde{i} \neq i-1$ such that $\mathcal{E}_{\tilde{i}} \cap \mathcal{E}_{i-1} \neq \emptyset$.

To generate the control value from this automaton, first define

$$\mathsf{Clip}\left(\mathcal{U}_{\mathrm{c}}(x,i), \tilde{u}\right) = \begin{cases} \tilde{u}, & \text{if } \tilde{u} \in \mathcal{U}_{\mathrm{c}}(x,i); \\ \bar{u}, & \text{otherwise;} \end{cases} \tag{15}$$

where the value $\bar{u} \in \mathcal{U}_{\mathrm{c}}(x,i)$ is "near" the value $\tilde{u}$ in some sense; for example, choose $\bar{u}$ to minimize $\|\tilde{u} - \bar{u}\|$ or $\|\mathrm{B}\tilde{u} - \mathrm{B}\bar{u}\|$ for some norm. When the controller automaton transitions—which it must do at every sample time t_k—the control signal generated upon transitioning into mode $m(i)$ at time t_k is

$$u_{\mathrm{pw}}(t) = \mathsf{Clip}\left(\mathcal{U}_{\mathrm{c}}(x(t_k), i), \tilde{u}(t_k)\right) \tag{16}$$

for $t_k \leq t < t_{k+1}$, where $\mathcal{U}_{\mathrm{c}}(x(t_k), i)$ is given in (12) for $i = 1, 2, \ldots, \bar{N}$ and (14) for $i = 0$.

In what follows we use

$$\bar{u} = q + \frac{\tilde{u} - q}{\|\mathrm{L}(\tilde{u} - q)\|_2} \tag{17}$$

where L is the Cholesky factorization of Q^{-1}, Q is the shape matrix for $\mathcal{U}_{\mathrm{c}}(x(t_k), i)$ and q is its center vector. This choice is cheap to evaluate but other choices of $\bar{u}$ are possible.

When we combine this discrete automaton with the continuous state, continuous time plant through the time sampling sensors and zero-order hold actuators shown in figure 1, we arrive at the *control filtering hybrid automaton* (CFHA).

Proposition 3: Let plant trajectory $x(\cdot)$ solve (2)–(3) with initial condition $x(0) = x_0$. If

$$x_0 \in \bigcup_{i=0}^{\bar{N}} \mathcal{E}_i$$

and (16) is used to generate the control signal, then $x(t) \in \mathcal{S}_C$ for all $t > 0$. Furthermore, for any $i \in \mathcal{N}(x_0)$, there exists a control signal which can be generated by the CFHA such that $x(i\delta) \in \mathcal{S}_T$.

Proof: We prove the second claim first. Choose any $i \in \mathcal{N}(x_0)$. We show inductively that the CFHA can generate a control signal $u_{\mathrm{pw}}(\cdot)$ such that $x(t_k) \in \mathcal{E}_{i-k}$ for all $k = 0, 1, \ldots, i$. The base case is true by observing that $x_0 = x(t_0) \in \mathcal{E}_i$. Now assume that $x(t_k) \in \mathcal{E}_{i-k}$ and that the CFHA is in mode $m(i-k)$. By (16), $u_{\mathrm{pw}}(t) \in \mathcal{U}_{\mathrm{c}}(x(t_k), i-k)$, which by lemma 1 implies that $x(t_{k+1}) \in \mathcal{E}_{i-k-1}$. In particular, $x(i\delta) \in \mathcal{E}_0 = \mathcal{E}(\mathcal{S}_T) \subseteq \mathcal{S}_T$.

Now we show by induction that all control signals (16) generated by the CFHA maintain safety for all $t > 0$. If $x(t_k) \in \mathcal{E}_i$ for $i > 0$, then for any choice of input allowed by (12), lemma 1 ensures that $x(t_{k+1}) \in \mathcal{E}_{i-1}$. By (10f) and (10e), $\begin{bmatrix} x(t_k) & u_k \end{bmatrix}^T \in \mathcal{I}_1$; and by (8) and (10c), $x(t) \in \mathcal{S}_C$ for all $t \in [t_k, t_{k+1}]$. Finally, consider the base case $x(t_k) \in \mathcal{E}_0 = \mathcal{E}(\mathcal{S}_T) \subseteq \mathcal{S}_T$. By (14) $u_{\mathrm{pw}}(t) = u_{\mathrm{fb}}^{\mathrm{inf}}(x(t_k))$ for $t_k \leq t < t_{k+1}$; furthermore, by the assumption made in section I-B, continued use of $u_{\mathrm{fb}}^{\mathrm{inf}}(x(t_{\hat{k}}))$ for $\hat{k} > k$ will ensure that the trajectory remains in $\mathcal{S}_C$ for all $t > t_k$. ■

Proposition 4: If there exists $i \geq 1$ such that

$$\mathcal{E}_{i-1} \subseteq \mathcal{E}_i \tag{18}$$

and $x(t_k) \in \mathcal{E}_i$ for any k, then the CFHA can ensure satisfaction of the safety constraint for all $t > 0$ without use of $u_{\mathrm{fb}}^{\mathrm{inf}}(x)$.

Proof: We show that the CFHA can choose mode $m(i)$ at all sample times $t_{\hat{k}}$ for $\hat{k} \geq k$. For the base case, assumption $x(t_k) \in \mathcal{E}_i$ implies the CFHA can be in mode $m(i)$. By lemma 1 any $u_{\mathrm{pw}}(t)$ allowed by (16) will yield $x(t_{k+1}) \in \mathcal{E}_{i-1}$, which implies by (18) that $x(t_{k+1}) \in \mathcal{E}_i$.

Consequently, the CFHA can take the self-loop back to $m(i)$, completing the inductive step. Because this is a valid behaviour of the CFHA, proposition 3 guarantees infinite horizon safety; however, only mode $m(i)$ for $i > 0$ is visited, so $u_{\text{fb}}^{\text{inf}}(x)$ is never invoked. ■

If the conditions of proposition 4 hold and $\mathcal{E}_i$ is large enough to serve as an operational envelope for the system, then the discrete components of CFHA can be collapsed to the single mode $m(i)$ and a self-loop transition. This option may be convenient when online memory and/or processing resources are limited.

The computational cost of evaluating the CFHA is relatively modest: Transitioning from mode $m(i)$ requires an evaluation of the invariant ellipsoid (a quadratic function of state) for each alternative transition considered other than mode $m(i-1)$ (which is guaranteed to be feasible by lemma 1 and the choice of invariants for modes $m(i)$ and $m(i-1)$), plus the cost of projecting the exogenous input into the set of safe inputs for the chosen mode (17). However, the computational cost of considering these alternative transitions is likely to be a key driver in resolving the remaining degrees of freedom made available to the designer by the discrete nondeterminism that remains in the CFHA. We further explore one point in this space of safe designs in section IV-D.

IV. Quadrotor Control Example

In this section we demonstrate the application of the algorithms described above to a longitudinal nonlinear model of a quadrotor.

A. Modeling

The state of the longitudinal model is six dimensional:

- Horizontal position x_1 [m] (positive rightward),
- vertical position x_2 [m] (positive upward),
- horizontal velocity x_3 [m/s],
- vertical velocity x_4 [m/s],
- roll x_5 [rad] (positive clockwise),
- roll velocity x_6 [rad/s].

The control input is two dimensional:

- Total thrust u_1,
- Desired roll angle u_2.

We use the model of plant dynamics derived in [12]:

$$\dot{x}_1 = x_3, \tag{20a}$$
$$\dot{x}_2 = x_4, \tag{20b}$$
$$\dot{x}_3 = u_1 K \sin x_5, \tag{20c}$$
$$\dot{x}_4 = -g + u_1 K \cos x_5, \tag{20d}$$
$$\dot{x}_5 = x_6, \tag{20e}$$
$$\dot{x}_6 = -d_0 x_5 - d_1 x_6 + n_0 u_2, \tag{20f}$$

where constant K is a gain relating input u_1 and the quadrotor mass. Note that although input u_2 appears on the right hand side of (20f) and hence is related to roll acceleration $\ddot{x}_5$, it is *not* differential thrust. The constants d_0, d_1 and n_0 are the gains in a low-level PD controller for the roll angle which runs at high frequency on board the quadrotor; consequently, the pilot and/or CFHA provide desired roll angle as u_2.

The ellipsoidal implementation requires a linear model, so we must linearize the nonlinear terms (20d) and (20e). The two variables in these equations are x_5 and u_1, so we construct a second order Taylor series expansion about fixed values $\bar{x}_5$ and $\bar{u}_1$ respectively and derive affine dynamics (19). Given a bounded range of x_5 and u_1 it is possible to bound the linearization error, treat the error as a disturbance input v, and thereby construct a conservative capture basin using the affine dynamics.

It turns out that the leading term of the linearization error is the $(\frac{K}{2})x_5 u_1 \cos \bar{x}_5$ term in the error for $\dot{x}_3$. To reduce the size of this error, we must bound the range of x_5 and u_1; however, stringent bounds on x_5 requires the quadrotor to stay almost level, while stringent bounds on u_1 make it hard to drive the system to a desired vertical position and velocity. In order to avoid overly stringent bounds and yet still keep the linearization error small, we construct a hybrid automaton model of the plant as shown in figure 3, where each mode corresponds to linearization about a different pair of $\bar{x}_5$ and $\bar{u}_1$ values. We will define constraint and target sets for each of these modes appropriate to keep the bounds on the linearization error small, and then compute a capture basin for each mode independently. For the experiments conducted below, we chose $\bar{x}_5 = 0$ or $\bar{x}_5 = \pm 0.05$ and $\bar{u}_1 = g$ or $\bar{u}_1 = g \pm 0.5$ (a total of five modes).

B. Safety Constraints and Target Sets

While our implementation uses constraint sets that are ellipsoidal, humans typically prefer to describe constraints in box form. For our experiments, the state constraint set was chosen as

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

The ranges of x_1 and x_2 were chosen based on a conservative estimate of the region of the flight room in which accurate state estimates were available. The ranges of x_3 and x_4 were chosen because it was felt that high linear velocities were too likely to lead to a crash. The range of x_6 was chosen based on angular velocities that were experimentally observed to be safe.

The only state constraints which are not driven by physical considerations are those on x_5 (roll angle). We have observed experimentally that this quadrotor can recover a stable hover from roll angles as large as 0.5 radians, but we choose a much smaller range for our safety analysis in order to keep the linearization error bound small. In addition, we take advantage of the hybridized dynamics and split the range of the constraint on x_5 into patches centered on the

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\\ \dot{x}_3\\ \dot{x}_4\\ \dot{x}_5\\ \dot{x}_6\end{bmatrix} = \overbrace{\begin{bmatrix}0&0&1&0&0&0\\ 0&0&0&1&0&0\\ 0&0&0&0&\frac{1}{2}K\bar{u}_1\cos\bar{x}_5&0\\ 0&0&0&0&-\frac{1}{2}K\bar{u}_1\sin\bar{x}_5&0\\ 0&0&0&0&0&1\\ 0&0&0&0&-d_0&-d_1\end{bmatrix}\begin{bmatrix}x_1\\ x_2\\ x_3\\ x_4\\ x_5\\ x_6\end{bmatrix} + \begin{bmatrix}0&0\\ 0&0\\ K(\sin\bar{x}_5-\frac{1}{2}\bar{x}_5\cos\bar{x}_5)&0\\ K(\cos\bar{x}_5+\frac{1}{2}\bar{x}_5\sin\bar{x}_5)&0\\ 0&0\\ 0&n_0\end{bmatrix}\begin{bmatrix}u_1\\ u_2\end{bmatrix}}^{\text{linear}} + \underbrace{\begin{bmatrix}0\\ 0\\ -\frac{1}{2}\bar{u}_1K(\bar{x}_5\cos\bar{x}_5)\\ \frac{1}{2}\bar{u}_1K(\bar{x}_5\sin\bar{x}_5)-g\\ 0\\ 0\end{bmatrix}}_{\text{constant}} + \underbrace{\begin{bmatrix}0\\ 0\\ \frac{1}{2}Kx_5u_1\cos\bar{x}_5-\frac{1}{2}K(x_5-\bar{x}_5)^2\bar{u}_1\sin\xi\\ -\frac{1}{2}Kx_5u_1\sin\bar{x}_5-\frac{1}{2}K(x_5-\bar{x}_5)^2\bar{u}_1\cos\xi\\ 0\\ 0\end{bmatrix}}_{\text{linearization error}} \quad \text{for some } \xi \text{ in the range of possible values of } x_5. \tag{19}$$

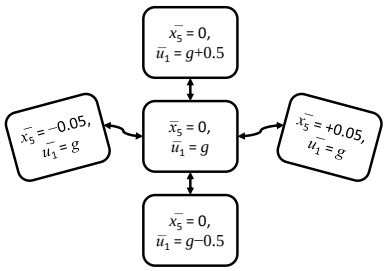

Fig. 3. Hybridization of the plant dynamics. Each mode corresponds to dynamics (19) with a different $\bar{x}_5$ and $\bar{u}_1$ pair.

corresponding $\bar{x}_5$:

$$\begin{aligned}\bar{x}_5 &= -0.05 \quad \text{with } x_5 \in [-0.15, +0.05]\\ \bar{x}_5 &= 0 \quad \text{with } x_5 \in [-0.10, +0.10]\\ \bar{x}_5 &= +0.05 \quad \text{with } x_5 \in [-0.05, +0.15]\end{aligned}$$

The experimental control constraint set also depends on the hybridization mode:

$$\begin{aligned}u_1 &\in [-0.5, +0.5] + \bar{u}_1,\\ u_2 &\in [-\tfrac{\pi}{16}, +\tfrac{\pi}{16}] + \bar{x}_5.\end{aligned}$$

The range of u_1 was also chosen to keep the linearization error bound small; the quadrotor is actually capable of total thrust in the range $[0, g+2]$. The range of u_2 was chosen to be slightly inside the constraint set of x_5 for each hybridization mode because x_5 tracks u_2.

Unfortunately, ellipsoids make poor approximations of box constraints. The single largest volume ellipsoid fitting within the state constraint box for any particular hybridization mode turns out to contain less than 20% of the volume of that box. In order to capture more of the safe set, we use two other constraint sets in addition to this maximum volume ellipsoid. Both have smaller volume, but are specifically chosen to stretch into the anti-diagonal corners in $x_1 - x_3$ and $x_2 - x_4$ space (for example, the corners where x_1 is small and x_3 is large, and vice versa), because such a constraint is favourable to double integrators. One of the sets is also chosen to stretch into the anti-diagonal corners in $x_1 - x_2$ space, while the other into the diagonal corners in $x_1 - x_2$ space. All three constraint sets treat the x_5 and x_6 constraints identically. Figure 4 shows the relevant projections of these constraint sets.

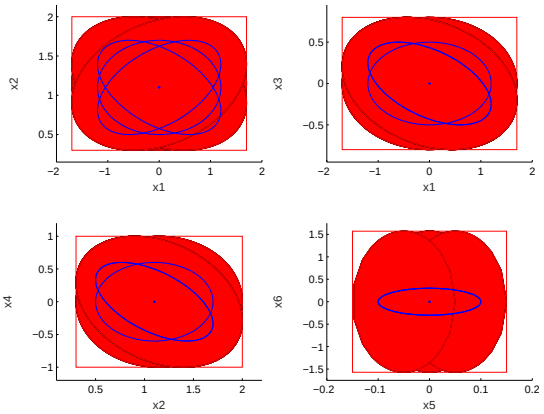

Fig. 4. Projections of the state constraint approximations. The red rectangles are the safety constraints, the solid red regions are the union of the three ellipsoids which are used as constraint sets by the capture basin algorithm, and the blue outlines are the corresponding target sets.

Using a linearization about $\bar{x}_5 = 0$ and $\bar{u}_1 = g$ (flat hover) and ignoring the linearization error, a stabilizing LQR feedback controller was designed and tested. The cost matrices in the LQR design were chosen to more heavily penalize deviations in x_2 and u_2 in order to promote fast action in the vertical direction while keeping the angular velocity x_6 from becoming too large. Experiments determined that this LQR controller was capable of stabilizing the quadrotor from the set

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

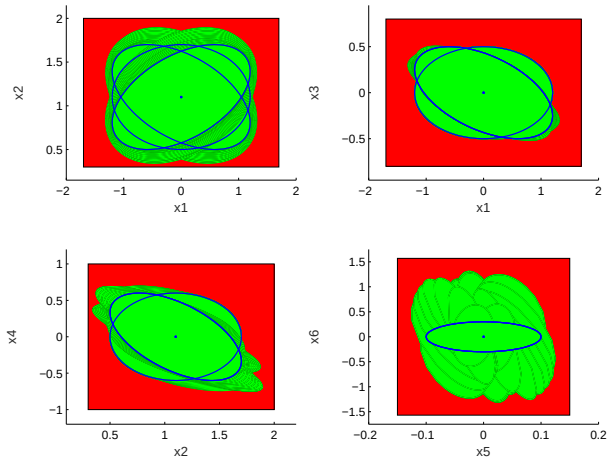

Fig. 5. Projections of the piecewise ellipsoidal approximation of the capture basin. The red rectangles are the safety constraints, the solid green region is the union of the ellipsoids comprising the capture basin approximation, and the blue contours are the target set(s).

Ellipsoidal underapproximations of this set were therefore used as the target for the capture basin computations. We used multiple underapproximations for the same reasons and stretched in the same fashion as described above for the constraint set ellipsoids. We note that these ranges ensure that the quadrotor is well away from the safety constraints, particularly in x_6 (whose dynamics have by far the smallest time constant), when the LQR controller is activated.

C. Capture Basin Computation

Although it is possible to use multiple direction vectors ℓ in the capture basin approximation, we determined empirically that a single direction vector

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

did a surprisingly good job of finding reasonably large capture basin approximations. Furthermore, it was found that among the index sets of the piecewise ellipsoidal representation of the capture basin, increasing the number of direction vectors was the least effective use of computational effort.

The computation of the piecewise ellipsoidal approximation of the capture basin indexed over five hybridization modes, three approximations of the constraint set, a single direction vector and ten sample periods was performed following the steps from section II-B. Projections of the result are shown in figure 5. It was run on a Lenovo Thinkpad Yoga with with an Intel Core i7-4600U CPU running at 2.1 GHz, 8 GB of RAM, 64-bit Windows 8.1 Pro, MATLAB R2014a, Ellipsoidal Toolbox 1.1.3 [13], SeDuMi 1.3 [14] and YALMIP 3 [15]. Computation during this offline phase took about 15 seconds for each combination of hybridization, constraint set and direction vector over ten sample periods, for a total offline computation time of less than five minutes.

D. Controller Execution

The exogenous input is generated by sampling the deflection of a joystick in real-time. The vertical deflection is mapped to the range of u_1 (zero deflection corresponding to $u_1 = g/K$), and the horizontal deflection is mapped to the full range of u_2.

We implement the CFHA from section III. At each sample time t_k the current state $x(t_k)$ is read, and then the exogenous input signal $\tilde{u}(t_k)$ is compared to the control envelope $\mathcal{U}_c(x(t_k), m)$ for multiple modes m.

If the exogenous input lies within the control envelope for one or more modes, then we choose the mode with largest horizon i. If the largest horizon is shared among multiple modes, we choose the mode for which the state $x(t_k)$ is deepest inside the mode's invariant.

If the exogenous input does not lie within any control envelope, then we choose the mode whose control envelope lies closest to the exogenous input; in other words, we choose a mode to minimize the projection error $\|\bar{u} - \tilde{u}(t_k)\|$ in (17).

Evaluating each alternative mode takes time, so we are constrained in how many can be checked in the online environment. We have adopted the following heuristic to prioritize which modes will be tested at each sample time. We consider the mode indexes in two groups: current horizon i and everything else (direction vector, hybridization and constraint approximation). Fixing the other indexes, we check modes i (the current mode), $i-1$ (guaranteed to be feasible by lemma 1) and also $i+1$ and $i+2$ (to see if it is possible to increase the horizon). We then fix the horizon to i and consider all possible combinations of the other indexes.

When running on the same machine described above, searching through a total of eighteen modes (each of five hybridizations and three constraint approximations at horizon i, as well as the current hybridization and constraint approximation at horizons $i-1$, $i+1$ and $i+2$ took an average of 0.03 seconds.

E. Simulation Results

The target flight hardware is an Ascending Technologies Pelican quadrotor running ROS on an Intel Atom processor. Flights are performed indoors with highly accurate, low latency state estimates generated by a VICON motion capture system. Experiments were performed to estimate model parameters $K = 0.89/1.4$, $d_0 = 70$, $d_1 = 17$ and $n_0 = 55$ for the Pelican. The sample period was chosen as 0.1 seconds, so the ten sample period capture basin horizon corresponds to one second of flight.

A soft real-time MATLAB simulator was developed using the nonlinear model (20) to drive system evolution and a USB joystick to provide the exogenous input from a human pilot. Figure 6 shows simulation results where the controller was running the CFHA described in section IV-D based on the piecewise ellipsoidal approximation of the capture basin described in section IV-C. During the time period between 6 and 12 seconds the pilot attempts to roll the quadrotor clockwise with input $u_2 > 0$, which would result in the horizontal position x_1 exceeding its upper bound; consequently, the CFHA clips u_2 and safety is maintained. In contrast, around time 16 seconds input u_2 is allowed a much larger positive value before clipping is initiated because x_1

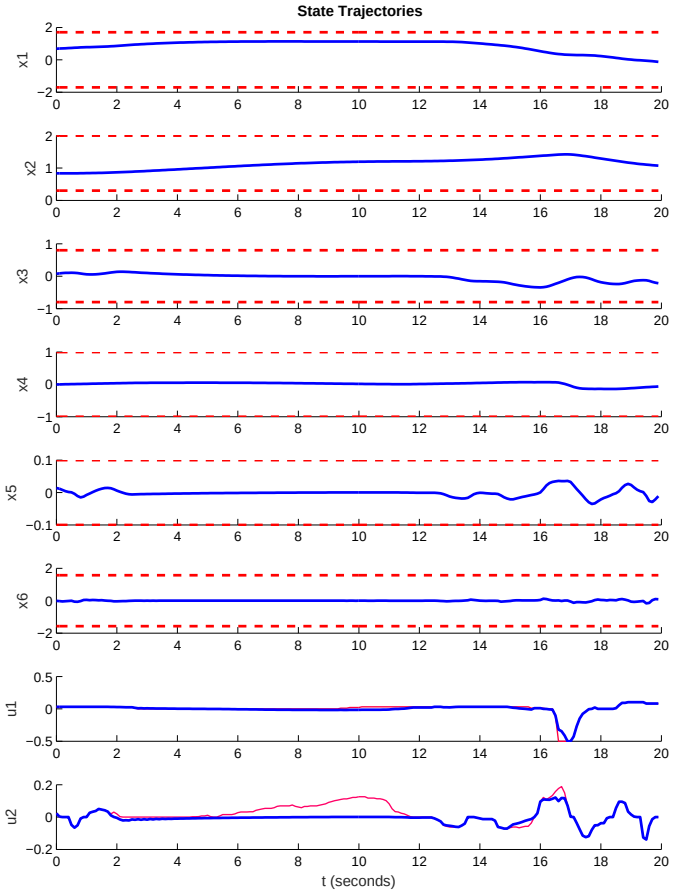

Fig. 6. Human-in-the-loop control trajectories of the simulated quadrotor using the CFHA. For the state variables, trajectories are shown in solid blue and state constraints in dashed red. For the input variables the user input $\tilde{u}$ is shown in thin red, and the clipped input $\bar{u}$ is shown in solid blue.

is well below its upper bound at this time. Notice that the 20 second simulated time period is much longer than the 1 second capture basin horizon, yet the system is never forced to invoke the LQR controller because a mode $m(i)$ with $i > 0$ is always found.

The assumption of an infinite horizon stabilizing controller was not overly strong for this system: as can be seen in the simulation results, the CFHA rarely exhausts even a short horizon. On the other hand, the time constants of the horizontal and vertical velocity are large compared with the horizon, and the linearization errors for these components of the dynamics are large enough that it was essentially impossible to get the capture basin to grow in these dimensions; consequently, we cannot invoke proposition 4 to rigorously prove that the LQR controller will never be needed.

V. Conclusions

We have described a method to construct a control automaton for a sampled data cyber-physical system which can ensure safe maintenance of state space constraints by checking and if necessary modifying an exogenous input signal which is only known at runtime. The algorithm is restricted to systems with linear dynamics but uses robust capture basins and hence can handle some nonlinearity through a worst-case analysis. The technique is demonstrated on a longitudinal model of an indoor quadrotor with a human-in-the-loop pilot providing the exogenous input signal. In the future we plan to investigate methods of handling signal delay and jitter, and fly the algorithm on the target hardware.

VI. Acknowledgments

The authors would like to thank Anayo Akametalu for his extensive help with the flight platform and some late night algebraic derivations, as well as Shahab Kaynama for his insight into the Ellipsoidal Toolbox and the discriminating kernel algorithm.

References

[1] I. M. Mitchell and S. Kaynama, "An improved algorithm for robust safety analysis of sampled data systems," in *Hybrid Systems: Computation and Control (HSCC)*, 2015, pp. 21–30.

[2] I. M. Mitchell, S. Kaynama, M. Chen, and M. Oishi, "Safety preserving control synthesis for sampled data systems," *Nonlinear Analysis: Hybrid Systems*, vol. 10, pp. 63–82, 2013.

[3] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, 2000.

[4] N. Aréchiga and B. Krogh, "Using verified control envelopes for safe controller design," in *Proceedings of the American Control Conference*, June 2014, pp. 2918–2923.

[5] J. H. Gillula, S. Kaynama, and C. J. Tomlin, "Sampling-based approximation of the viability kernel for high-dimensional linear sampled-data systems," in *Hybrid Systems: Computation and Control (HSCC)*, 2014, pp. 173–182.

[6] C. Dabadie, S. Kaynama, and C. J. Tomlin, "A practical reachability-based collision avoidance algorithm for sampled-data systems: Application to ground robots," in *International Conference on Intelligent Robots and Systems (IROS)*, 2014.

[7] S. Kaynama, I. M. Mitchell, M. M. K. Oishi, and G. A. Dumont, "Scalable safety-preserving robust control synthesis for continuous-time linear systems," *IEEE Transactions on Automatic Control*, vol. 60, no. 11, pp. 3065–3070, November 2015.

[8] G. Simko and E. K. Jackson, "A bounded model checking tool for periodic sample-hold systems," in *Hybrid Systems: Computation and Control (HSCC)*, 2014, pp. 157–162.

[9] D. Henrion and M. Korda, "Convex computation of the region of attraction of polynomial control systems," *IEEE Transactions on Automatic Control*, vol. 59, no. 2, pp. 297–312, Feb 2014.

[10] I. Karafyllis and M. Krstic, "Nonlinear stabilization under sampled and delayed measurements, and with inputs subject to delay and zero-order hold," *IEEE Transactions on Automatic Control*, vol. 57, no. 5, pp. 1141–1154, May 2012.

[11] J.-P. Aubin, A. M. Bayen, and P. Saint-Pierre, *Viability Theory: New Directions*, ser. Systems & Control: Foundations & Applications. Springer, 2011.

[12] P. Bouffard, "On-board model predictive control of a quadrotor helicopter: Design, implementation, and experiments," Department of Electrical Engineering and Computer Science, University of California at Berkeley, Tech. Rep. UCB/EECS-2012-241, December 2012. [Online]. Available: http://www.eecs.berkeley.edu/Pubs/TechRpts/2012/EECS-2012-241.html

[13] A. A. Kurzhanskiy and P. Varaiya, "Ellipsoidal toolbox," Department of Electrical Engineering and Computer Science, University of California, Berkeley, Tech. Rep. UCB/EECS-2006-46, May 2006. [Online]. Available: http://www.eecs.berkeley.edu/Pubs/TechRpts/2006/EECS-2006-46.html

[14] J. F. Sturm, "Using SeDuMi 1.02, a Matlab toolbox for optimization over symmetric cones," *Optimization methods and software*, vol. 11, no. 1–4, pp. 625–653, 1999.

[15] J. Löfberg, "YALMIP : a toolbox for modeling and optimization in MATLAB," in *Computer Aided Control Systems Design*, September 2004, pp. 284–289.