Numerical Simulation and Zero-Crossing Detection of Hybrid Dynamical Systems

Mitch Wilson

May 28, 2010

As more and more processes become digitally controlled, there is an increasing need to balance the continuous dynamics of analog systems with the discrete characteristics of these controllers, as in [2]. These types of systems, known as hybrid systems or embedded systems, are a current area of interest with respect to Mathematical modeling. The applications are numerous, from physics phenomena to aircraft electronics. In this paper, we will introduce the reader to the framework of hybrid dynamical systems. We will then introduce the topic of numerical integration as a means for simulation. Next, the concept of zero-crossing detection (ZCD) will be introduced, culminating in current progress and buildup for future work. Examples will be illustrated throughout.

1 Introduction

A hybrid system is described as any system that contains both continuous and discrete dynamics. Following the notation used by [7], we have a hybrid system H as follows:

Definition 1. *A hybrid system, H, is constructed by a data set (C, D, F, G) where*

- *A set, $C \subset \mathbb{R}^n$, the flow set, which is our domain for the continuous dynamics*
- *A set, $D \subset \mathbb{R}^n$, the jump set, which is our domain for the discrete dynamics*
- *A set-valued map, $F : \mathbb{R}^n \to \mathbb{R}^n$, pertaining to the continuous dynamics*
- *A set-valued map, $G : \mathbb{R}^n \to \mathbb{R}^n$, pertaining to the discrete dynamics*

The sets C and D do not have to be disjoint; there may be points where the system can both jump and flow. F can be seen as a set of differential inclusions, and G as a set of difference inclusions, written in the form:

$$\begin{aligned} \dot{x} &\in F(x), x \in C \\ x^+ &\in G(x), x \in D. \end{aligned}$$

Thus, if either F or G is empty, then we are simply left with a discrete system or a continuous ODE system of inclusions, respectively.

Example 1. *Consider the simulation of a bouncing ball. The ball is dropped from a height x_1 with a velocity x_2 under the force of gravity g. Thus we have a simple system of differential equations: $\dot{x_1} = x_2, \dot{x_2} = -g$ while $x_1 > 0$. Of course, once the ball touches the ground, a number of different events occur. The velocity of the ball changes sign as it bounces back up, but also with less energy due to deformation or absorption by the ground. Thus, for the event when $x_1 = 0$ and $x_2 \leq 0$, we*

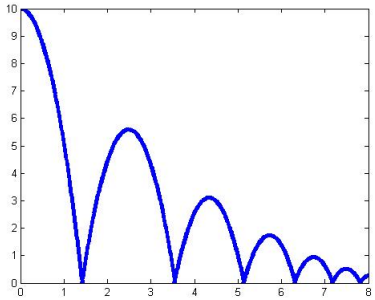

Figure 1: The evolution of a bouncing ball over time, illustrated as height x_1 over time t.

implement our discrete map G. In this case, we update x_2 *as* $x_2^+ = -\epsilon x_2$, *where* $0 \leq \epsilon < 1$ *is the coefficient of restitution. We then return to our differential equation in F. See Figure 1 for a typical 2-d graph. Thus its hybrid system is defined as:*

$$\begin{aligned} x = [x_1, x_2]', \quad F(x) &= [x_2, -g], C = \{(x_1, x_2) \in \mathbb{R}^2 | x_1 > 0\} \\ G(x) &= [0, -\epsilon x_2], D = \{(x_1, x_2) \in \mathbb{R}^2 | x_1 = 0, x_2 \leq 0\} \end{aligned}$$

To track the evolution of our system, we now introduce the concept of a hybrid time domain and particular paths which adhere to specific properties, as well as a few more definitions.

Definition 2. *Given an integer J and times* $0 = t_0 \leq t_1 \leq t_2 \leq ... \leq t_{J-1} \leq t_J$,*and* $\mathbb{N}$ *being the set of natural numbers, the set* $E = \bigcup_{j=0}^{J-1}([t_j, t_{j+1}], j)$, *where* $E \in \mathbb{R}_{\geq 0} \times \mathbb{N}$, *is a compact hybrid time domain. It is a hybrid time domain if for all* $(T, J) \in E, (T, J) \cap ([0, T] \times \{0, 1, 2, ..., J\})$ *is a compact hybrid time domain.*

The compact hybrid time domain will keep track of both our continuous and discrete evolutions. The value of t relates to the elapsed time in the continuous mappings, while j is used to indicate the number of times the discrete mapping has been used.

Definition 3. *A hybrid arc is a function x :dom x (the domain of x)* $\rightarrow \mathbb{R}^n$, *where dom x is a hybrid time domain. Also, for each* $j \in \mathbb{N}$ *the function mapping t to* $x(t, j)$ *is absolutely continuous* [1] *on the interval* $I_j = \{t | (t, j) \in$ *domain of* $x\}$.

In simpler terms, a hybrid arc is a curve with a hybrid time domain and is continuous in the interval (t_j, t_{j+1}) where j is constant.

Definition 4. *A hybrid arc x is a solution to the hybrid system H given x(0,0)* $\in C \cup D$ *and*

- *for each* $j \in \mathbb{N}$ *where the interval* I_j *has a nonempty interior,* $x(t, j) \in C$ *for all* $t \in (t_j, t_{j+1})$, *and for almost all* $t \in I_j, \dot{x}(t, j) \in F(x(t, j))$;
- *for each* (t, j) *in the domain of x such that* $(t, j + 1)$ *is also in the domain,* $x(t, j) \in D$ *and* $x(t, j + 1) \in G(x(t, j))$.

Definition 5. *A set-valued map* $u(x)$ *is outer-semicontinuous if for each* $x_i \rightarrow x$, *there is* $y_i \in u(x_i)$ *such that as* $y_i \rightarrow y, y \in u(x)$. *This means that any convergent sequence behaves like its limit point.*

Example 2. *Going back to the bouncing ball, we select the initial conditions* $x(0, 0) = [10, 0]', \epsilon = 0.5$. *Accounting the number of jumps into the system, its graph now looks like that in Figure 2. The first jumps occur at about 1.43, 2.86, 3.65, and 3.92 seconds. Note how the time interval between bounces becomes smaller over time.*

To ensure robustness against perturbation, it is useful, though not always feasible, that the hybrid system adheres to some very useful properties:

[1] Meaning that the function is differentiable almost everywhere and Lebesgue integrable

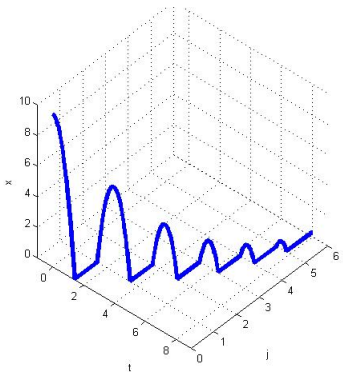

Figure 2: The evolution of a hybrid bouncing ball system over time, indexing the jump count, j.

- C and D are closed sets. This helps them become robust against perturbation. This means that if the point x is slightly changed to $x + \epsilon$ for some small ϵ, then the point will more likely stay within the domain and continue its evolution.
- F is outer-semicontinuous and locally bounded, and for all $x \in C$, $f(x)$ is nonempty and convex[2].
- G is outer-semicontinuous, and for all $x \in D$, $g(x)$ is nonempty.

2 Numerical Integration and Simulation

In order to simulate these systems, we must investigate the properties of numerical integration. This involves implementing our continuous dynamics into a discrete domain given a specific numerical scheme. The most basic simulation can be performing using a forward Euler method. In evaluating the evolution of a differential equation $\dot{y} = f(x)$ with y_0 given, the method can be written as

$$y_{n+1} = y_n + hf(y_n), \tag{1}$$

where h is a constant step size and f is the right-hand side of the differential equation. There are other possible numerical methods available, including some of higher accuracy like the Runge-Kutta method. For the purposes of this paper, only the Forward Euler, Backwards Euler, and trapezoidal methods are considered. Other methods are discussed in [5] and [3]. We define the numerical operator, $N_h y(t_n)$, as follows for the forward Euler method for a given time $t_n \in [0, t_J]$:

$$N_h y(t_n) = \frac{y(t_n) - y(t_{n-1})}{h} - f(t_{n-1}, y_{n-1}) \tag{2}$$

The numerical operator is interpreted as the error between the actual derivative of a system and its estimated derivative based on its points. We now introduce a few more definitions that illustrate the power of these numerical schemes.

Definition 6. *A numerical scheme with step size h is consistent of an order, p, if its error from the actual solution is on the order of h^p, written as $O(h^p)$.*[3]

Definition 7. *A numerical scheme is 0-stable if there exists an upper bound on the difference between two mesh functions with the same domain. Generally, this is written for functions x and z as :*

$$|x_n - z_n| \leq K(|x_0 - z_0| + \max_{i \leq j \leq n} |N_h x_h(t_j) - N_h z_h(t_j)|) \tag{3}$$

where x_n and z_n are the evaluations of x and z at the nth time step, and $N_h x_h(t_j)$ and $N_h z_h(t_j)$ are the numerical operators for x and z, respectively.

[2]For a definition of convex, refer to [6]

[3]We use the notation $O(h^p)$ to mean that it can be bounded by a finite multiple of h^p, so error $\leq Ch^p$ for some large, positive C

In practice, these two functions would be the actual solution and our simulated result.

If a numerical scheme is is consistent to an order p and also 0-stable, then it is said to be *convergent* of order p.

We can show that the forward Euler method is consistent to an order of 1, as well as 0-stable, thus making it first-order convergent.

Following the setup by [5], we can show consistency by using a Taylor expansion for $y(t_n)$. Letting $h = t_n - t_{n-1}$, we find:

$$\begin{aligned} N_h y(t_n) &= \frac{y(t_n) - y(t_{n-1})}{h} - f(y(t_{n-1})) \\ &= \frac{(y(t_{n-1}) + hf(t_{n-1}) + \frac{h^2}{2}y''(t_{n-1}) + ...) - y(t_{n-1})}{h} - f(y(t_{n-1})) \\ &= \frac{hf(y(t_{n-1})) + \frac{h^2}{2}y''(t_{n-1}) + ...}{h} - f(y(t_{n-1})) \\ &= f(y(t_{n-1})) + \frac{h}{2}y''(t_{n-1}) + ... - f(y(t_{n-1})) \\ &= \frac{h}{2}y''(t_{n-1}) + ... = \frac{h}{2}y''(t_{n-1}) + O(h^2) \end{aligned}$$

To show 0-stability we start off with a couple simplifications. Using x and z as our functions:

$$\begin{aligned} s_n &= x_n - z_n, \\ \theta &= \max_{1 \le n \le N} |N_h x_h(t_n) - N_h z_h(t_n)|. \end{aligned}$$

Then, for any n,

$$\begin{aligned} |N_h x_h(t_n) - N_h z_h(t_n)| &\le \theta \\ |(\frac{x_n - x_{n-1}}{h_n} - f(x_{n-1})) - (\frac{z_n - z_{n-1}}{h_n} - f(z_{n-1}))| &\le \theta \\ |\frac{x_n - z_n}{h_n} - \frac{x_{n-1} - z_{n-1}}{h_n} - (f(x_{n-1}) - f(z_{n-1}))| &\le \theta. \\ |\frac{s_n}{h_n} - \frac{s_{n-1}}{h_n} - (f(x_{n-1}) - f(z_{n-1}))| &\le \theta. \end{aligned}$$

We next recall two more things, the triangle inequality and Lipschitz continuity. We recall that for scalars a and $b \in \mathbb{R}$, we have $|a + b| \le |a| + |b|$. If we substitute $a = A - B$ and $b = B$, we get $|A - B + B| \le |A - B| + |B|$, or $|A| \le |A - B| + |B|$, resulting in the similar inequality $|A| - |B| \le |A - B|$. Plugging in $\frac{s_n}{h_n}$ for A and $\frac{s_{n-1}}{h_n} + (f(x_{n-1}) - f(z_{n-1}))$ for B, we get

$$\begin{aligned} &|\frac{s_n}{h_n}| - |\frac{s_{n-1}}{h_n} + (f(x_{n-1}) - f(z_{n-1}))| \\ \le &|\frac{s_n}{h_n} - \frac{s_{n-1}}{h_n} - (f(x_{n-1}) - f(z_{n-1}))| \le \theta. \end{aligned}$$

We next recall Lipschitz continuity.

Definition 8. *A function is Lipschitz continuous if there exists a constant $L \ge 0$ such that for every y and $\hat{y}$,*

$$d_X(f(y), f(\hat{y})) \le L d_Y(y, \hat{y}).$$

where d_X and d_Y represent some metric. For our purposes, our domain is the time interval $[t_0, t_N] = [0, b]$, $f(y) = f(t, y)$ is the right-hand side of our differential equation. This relates to our hybrid set-valued map $F(x)$ as two arcs would have this same property for a given fixed interval. Furthermore, d_X and d_Y are the metrics given by the absolute value. Thus our Lipschitz inequality is now written

$$|f(y) - f(\hat{y})| \le L|y - \hat{y}|. \tag{4}$$

We can now return to our proof of 0-stability for the Euler method. Using Lipschitz continuity, we note

$$\begin{aligned}
|\frac{s_{n-1}}{h_n} + (f(x_{n-1}) - f(z_{n-1}))| &\leq |\frac{s_{n-1}}{h_n}| + |(f(x_{n-1}) - f(z_{n-1}))| \\
&\leq |\frac{s_{n-1}}{h_n}| + L|x_{n-1} - z_{n-1}| = |\frac{s_{n-1}}{h_n}| + L|s_{n-1}| \\
&= |s_{n-1}|(\frac{1}{h_n} + L).
\end{aligned}$$

We then return to our previous equation and simplify.

$$\begin{aligned}
|\frac{s_n}{h_n}| - |\frac{s_{n-1}}{h_n} + (f(x_{n-1}) - f(z_{n-1})|) &\leq \theta \\
|\frac{s_n}{h_n}| &\leq |\frac{s_{n-1}}{h_n} + (f(x_{n-1}) - f(z_{n-1}))| + \theta \\
|\frac{s_n}{h_n}| &\leq |s_{n-1}|(\frac{1}{h_n} + L) + \theta \\
|s_n| &\leq h_n\theta + |s_{n-1}|(1 + h_nL)
\end{aligned}$$

So far the computation has been modestly easy to follow. However, in the next few lines, a number of large jumps, with minimal explanation, are made in [5], culminating in

$$\begin{aligned}
|s_n| &\leq h_n\theta + |s_{n-1}|(1 + h_nL) \\
&\leq h_n\theta + |h_{n-1}\theta + |s_{n-2}|(1 + h_{n-1}L)|(1 + h_nL) \\
&\vdots \\
&\leq |s_0| \prod_{j=1}^{N} (1 + h_jL) + \theta \sum_{j=1}^{N} h_j(1 + h_{j+1}L)(1 + h_{j+2}L) \dots (1 + h_NL) \\
&\leq e^{Lt_n}|s_0| + \frac{1}{L}(e^{Lt_n} - 1)\theta.
\end{aligned}$$

We will now go through each of the steps in greater detail. We note that in the second line, $|s_{n-1}|$ is substituted by $h_{n-1}\theta + |s_{n-2}|(1 + h_{n-1}L)$. This can be seen as we have a iterative process where each s_n is bounded by a function of s_{n-1}. This process is repeated until our s_n is only a function of s_0 and all the h's along the way, resulting in our product of $(1 + h_jL)$'s and s_0.

Petzold quickly asserts that this product for indices j through N is less than the value $e^{L(t_N - t_j)}$. This step requires a bit of reorganizing.

It is easy to see that the first terms of the Taylor series for a value $e^{h_jL} = 1 + h_jL + O((h_jL)^2)$, so it is easy to assert that $1 + h_jL \leq e^{h_jL}$. Comparing the product of multiple terms seems a little harder, but we will extend this to the case of multiplying out two terms, and it will be coercive that we have a solid statement.

Consider $(1 + h_1L)(1 + h_2L) = (1 + (t_1 - t_0)L)(1 + (t_2 - t_1)L)$. Upon multiplying terms, we have a telescoping series in the linear L term, leaving only the first and the last t_j, and the L^2 term is bounded by the expansion of the square of the sums of h_j,

$$\begin{aligned}
(1 + t_1L - t_0L)(1 + t_2L - t_1L) &= 1 + L(t_2 - t_1 + t_1 - t_0) + h_1h_2L^2 \\
&\leq 1 + L(t_2 - t_0) + \frac{L^2}{2}(h_1 + h_2)^2 \\
&= 1 + L(t_2 - t_0) + \frac{L^2}{2}(t_2 - t_0)^2 \leq e^{L(t_2 - t_0)}
\end{aligned}$$

Thus, with our $|s_0|$ coefficient,

$$(1 + h_1L)(1 + h_2L) \dots (1 + h_NL) \leq e^{L(t_N - t_0)} = e^{L(b-0)} = e^{Lb}.$$

Likewise, we can also say

$$\theta\sum_{j=1}^{N} h_j(1+h_{j+1}L)(1+h_{j+2}L)\dots(1+h_NL) \leq \theta\sum_{j=1}^{N} h_j e^{L(t_N-t_j)}.$$

The last substitution seems to be the most obscure. Petzold makes the following claim:

$$\sum_{j=1}^{N} h_j e^{L(t_N-t_j)} \leq \sum_{j=1}^{N}\int_{t_{j-1}}^{t_j} e^{L(t_N-t)}dt.$$

Thus for each index, j, we have $h_j e^{L(t_N-t_j)} \leq \int_{t_{j-1}}^{t_j} e^{L(t_N-t)}dt$. Let us reorganize this, one side at the time, starting with the left.

$$\begin{aligned} h_j e^{L(t_N-t_j)} &= h_j e^{L(t_N-t_{N-1}+t_{N-1}-t_{N-2}\dots+t_{j+1}-t_j)} \\ &= h_j e^{L(h_N+h_{N-1}+\dots+h_{j+1})} \end{aligned}$$

On the right side, we make a substitution, $u = t_N - t, du = -dt$, changing our integral to

$$\begin{aligned} \int_{t_{j-1}}^{t_j} e^{L(t_N-t)}dt &= -\int_{t_N-t_{j-1}}^{t_N-t_j} e^{Lu}du \\ &= -\frac{1}{L}(e^{L(t_N-t_j)} - e^{L(t_N-t_{j-1})}) \\ &= \frac{1}{L}(e^{L(h_N+h_{N-1}+\dots+h_j)} - e^{L(h_N+h_{N-1}+\dots+h_{j+1})}) \\ &= \frac{1}{L}e^{L(h_N+h_{N-1}+\dots+h_{j+1})}(e^{Lh_N}-1). \end{aligned}$$

Putting the left and right-hand sides together, we obtain

$$\begin{aligned} h_j e^{L(h_N+h_{N-1}+\dots+h_{j+1})} &\leq \frac{1}{L}e^{L(h_N+h_{N-1}+\dots+h_{j+1})}(e^{Lh_j}-1) \\ h_j &\leq \frac{1}{L}(e^{Lh_j}-1) \\ 1+Lh_j &\leq e^{Lh_j} \end{aligned}$$

which we know to be true. Thus, the statement by Petzold is verified, and we can finish our check of 0-stability.

$$\begin{aligned} \sum_{j=1}^{N} h_n e^{L(t_N-t_j)} \leq \sum_{j=1}^{N}\int_{t_{j-1}}^{t_j} e^{L(t_N-t)}dt &= e^{Lt_N}\int_0^{t_N} e^{-Lt}dt \\ &= \frac{1}{L}(e^{Lt_N}-1) = \frac{1}{L}(e^{Lb}-1) \end{aligned}$$

Thus our verification of 0-stability is complete, as long as our $K = \max\{e^{Lb}, \frac{1}{L}(e^{Lb}-1)\}$. For the case where x is our numerical solution and z is the actual solution, we can place greater bounds on this inequality. If we have the same starting point, then $|s_0| = 0$, and likewise $N_h x_h(t_n) = 0$ based on our Euler method. Thus,

$$|x_n - z_n| \leq \frac{1}{L}(e^{Lb}-1)|N_h z_h(t_n)|.$$

Since we are now confident in the implementation of our numerical integrators, we introduce the notions of hybrid simulators, and their arcs and solutions.

Definition 9. *The hybrid simulator, $\hat{H}$, is defined the same way we defined our original hybrid system. The set is comprised of $\hat{H} = \{\hat{C}, \hat{D}, \hat{F}, \hat{G}\}$, where $\hat{C}, \hat{D}, \hat{F}$, and $\hat{G}$ are the discrete equivalents of their respective counterparts.*

$\hat{H}$ can thus be set up using a system of difference inclusions:

$$\begin{aligned} x^+ &\in \hat{F}(x), x \in \hat{C} \\ x^+ &\in \hat{G}(x), x \in \hat{D}. \end{aligned}$$

Definition 10. *Given an integer J and times $0 = K_0 \leq K_1 \leq K_2 \leq ... \leq K_{J-1} \leq K_J$,and $\mathbb{N}$ being the set of natural numbers, a subset $E \in \mathbb{N} \times \mathbb{N}$ is a compact discrete time domain if $E = \bigcup_{j=0}^{J-1} \bigcup_{k=K_j}^{K_{j+1}} (k, j)$. Also like before, It is a discrete time domain if $\forall (K, J) \in E, E \cap (\{0, 1, ..., K\} \times \{0, 1, ..., J\})$ is a compact discrete time domain.*

Definition 11. *A function $\hat{x}$:dom $\hat{x} \to \mathbb{R}^n$, is a discrete hybrid arc if dom $\hat{x}$ is a discrete time domain.*

Definition 12. *A discrete arc $\hat{x}$ is a simulated solution to the hybrid system H with with a hybrid simulator $\hat{H}$ given*

- *for all $k, j \in \mathbb{N} \times \mathbb{N}$ such that (k, j) and $(k + 1, j) \in$dom $\hat{x}$, $\hat{x}(k, j) \in \hat{C}, \hat{x}(k + 1, j) \in \hat{F}(\hat{x}(k, j))$*
- *for all $k, j \in \mathbb{N} \times \mathbb{N}$ such that (k, j) and $(k, j + 1) \in$dom $\hat{x}$, $\hat{x}(k, j) \in \hat{D}, \hat{x}(k, j + 1) \in \hat{G}(\hat{x}(k, j))$*

3 Observations of Simulations and Results with Zero-Crossing Detection

Upon implementation of these simulators, a number of acute phenomena can occur. These can have minimal or profound effect on the simulated solutions, especially when comparing them to the original solution.

It may be possible that after every time step, the arc alternates between being in $\hat{C}$ and $\hat{D}$. This effectively skyrockets the index j as the discrete arc is constantly jumping. This event is known as *chattering* and is noted by [4].

Example 3. *Consider the system*

$$\begin{aligned} x^+ \in \hat{F}(x) = x - 1.9hx, \quad \hat{C} = \{x | x \geq 0\} \\ x^+ \in \hat{G}(x) = x - 1.9hx, \quad \hat{D} = \{x | x < 0\} \end{aligned}$$

with a step size $h = 1$ and $x(0) = 1$. An illustration is shown in Figure 3. The original hybrid system would quickly decay to zero but never cross that point if $x(0) > 0$, whereas the simulated solution will alternate regardless of $x(0)$.

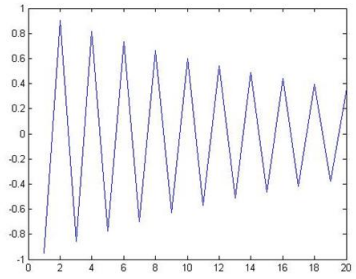

Figure 3: A discrete arc may constantly jump back and forth between $\hat{C}$ and $\hat{D}$.

One more troublesome occurrence is known as Zeno behavior. A solution is said to be Zeno if it jumps an infinite number of times within a finite time interval. Thus during this phenomenon, the time between jumps become infinitesimally smaller, and the system is unable to progress past a finite time because the amounts at which the system is evolving is within the machine error of the simulator (for example, MATLAB has a machine error $O(10^{-16})$), so any evolution is on the same likelihood as having added a small random perturbation. This is clearly counterintuitive with our normal perception of these hybrid systems.

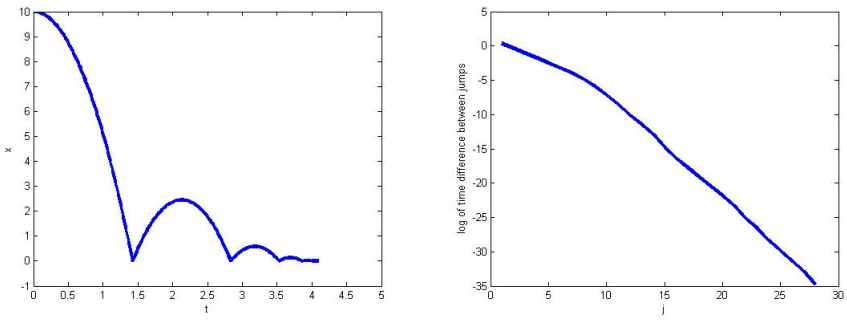

Figure 4: On the left, the discrete bouncing ball system experiences Zeno behavior, causing it to end prematurely. On the right, the logarithm of the time between jumps is graphed against its index j. It quickly approaches machine error until the system reaches an infinite number of jumps while increasing the elapsed time by 0.

Example 4. *Revisiting the bouncing ball example, we see that it has Zeno behavior. As each subsequent bounce yields a lower maximum point, the time between bounces becomes ever smaller until the time is no longer able to be managed by the simulator. In reality, the ball would stay on the ground for an indefinite length of time. However, as shown in Figure 4, the simulator is unable to proceed past a time frame of a few seconds.*

Zeno behavior in itself is under research, as in [4]. A few solutions have been offered to eliminate Zeno behavior. One of them involves detecting "Zeno neighborhoods." or regions where the system encounters a large number of jumps in a short period of time. Upon recognizing said Zeno neighborhood, a possible solution is to redeclare the system variables so that the system may continue to evolve over time. In doing so, one would have to introduce a third index, z, and write the arcs as $x(t, j, z)$.

Depending on how the original sets C and D are defined, it may also be difficult, if not near impossible, to recognize a jump in the system. That is, D may be ill-defined and easily missable.

Example 5. *Consider the system of a particle rotating around the origin in $\mathbb{R}^2$. See Figure 5. Suppose we wanted to count how many times the particle has crossed the y-axis. We can write our system as:*

$$\begin{aligned} x = [x_1, x_2]', F(x) = [-x_2, x_1]', \quad C = \quad & \{(x_1, x_2) \in \{(-\infty, 0) \cup (0, \infty)\} \times \mathbb{R}\} \\ G(x) = [0, x_2]', \quad D = \quad & \{(x_1, x_2) \in \mathbb{R}^2 | x_1 = 0\} \end{aligned}$$

In a solution to the hybrid system, the particle would rotate counterclockwise at a fixed radius. However, it could be the case

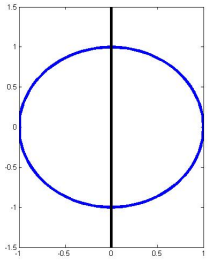

Figure 5: A rotating particle of fixed radius would have no trouble counting how many times it has crossed the line $x = 0$, but simulators have a lot more setbacks.

that, based on our simulation, we may never hit $x = 0$ exactly and the system may never jump. Another possibility is that the arc does land exactly on 0 and never resumes flowing.

Perhaps, then, the domains for sets $\hat{C}$ and $\hat{D}$ should be rewritten to ensure robustness. This leads into the discussion of zero-crossing detection, or ZCD. That is, determining exactly when and where the simulation reaches the threshold between

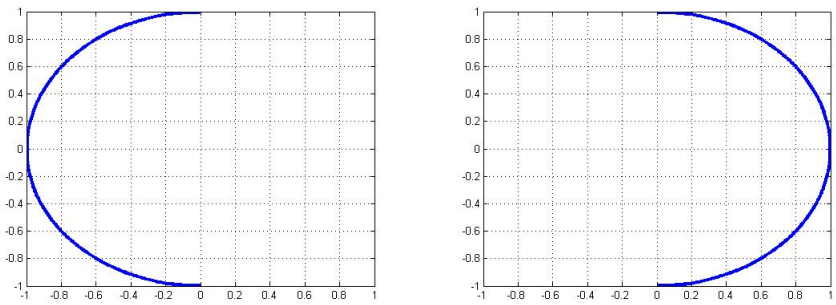

Figure 6: By utilizing the logic variable into ZCD, we can generate distinct results for the different values of q. The left side refers to when $q = -1$. The right side is for $q = 1$.

continuous and discrete, thus switching from $\hat{F}$ to $\hat{G}$ or vice-versa. We will start with a general form of ZCD, and then go into its discrete version. In order to properly model this behavior, we introduce a few new components.

Definition 13. *The threshold function $h(x)$ is used to isolate the boundary between $\hat{C}$ and $\hat{D}$. The boundary is defined as all points where $h(x) = 0$.*

Definition 14. *The logic variable q is taken from the set $\{-1, 1\}$ and is used to ensure proper crossing over the boundary.*

Since our domain depends on the hybrid arc, we write our ZCD domains for C and D as follows:

$$C_{ZCD} = \{(x, q) \in \mathbb{R}^n \times \{-1, 1\} \quad |h(x)q \geq 0\}$$
$$D_{ZCD} = \{(x, q) \in \mathbb{R}^n \times \{-1, 1\} \quad |h(x)q \leq 0\}$$

Likewise, our set-valued maps may be written:

$$F_{ZCD} = [F(x), 0], (x, q) \in C_{ZCD}$$
$$G_{ZCD} = [G(x), l(q)], (x, q) \in D_{ZCD}$$

where $l(q)$ is a function which switches the sign of q. The benefit of the ZCD model is that it adheres to some those useful properties which we described earlier. These conditions require a couple of definitions.

Example 6. *Revisiting the rotating particle system, we can rewrite our ZCD system as follows:*

$$\dot{x_1} = -x_2, \dot{x_2} = x_1, \dot{q} = 0, \quad (x, q) \in C = \{\mathbb{R}^2 \times \{-1, 1\} \quad |xq \geq 0\}$$
$$x_1^+ = 0, x_2^+ = x_2, q^+ = -q, \quad (x, q) \in D = \{\mathbb{R}^2 \times \{-1, 1\} \quad |xq \leq 0\}$$

Illustrations for when $q = -1$ and $q = 1$ are shown in Figure 6.

In extending this to a discrete ZCD system, we use F' to indicate the progression using some numerical integration scheme, and h as the step size. The data can be written as:

$$x_1^+ = x_1 - hF'x_2, x_2^+ = x_2 + hF'x_1, q+ = q, \quad (x, q) \in \hat{C} = \{\mathbb{R}^2 \times \{-1, 1\} \quad |xq \geq 0\}$$
$$x_1^+ = 0, x_2^+ = x_2, q^+ = -q, \quad (x, q) \in \hat{D} = \{\mathbb{R}^2 \times \{-1, 1\} \quad |xq \leq 0\}$$

It is easy to see the benefits of ZCD into many of our situations. Its primary benefit is it allows us to detect our zero-crossings without concern. It maintains the properties for robustness, and ensures more smooth transitions between jumps and flows. However, ZCD is still limited to its implementation. For discrete ZCD, the error between the simulated result and its true

solution is still subject to the integration method used. It turns out that the choice of numerical integrator used can affect the number of jumps counted on a large time scale. For example, using forward Euler we would have:

$$\begin{aligned} x_{1new} &= x_{1old} - hx_{2old} \\ x_{2new} &= x_{2old} + hx_{1old} \end{aligned}$$

It is easy to see then that the radius no longer remains constant as

$$\begin{aligned} r_{new}^2 &= x_{1new}^2 + x_{2new}^2 \\ &= (x_{1old} - hx_{2old})^2 + (x_{2old} + hx_{1old})^2 \\ &= x_{1old}^2 + x_{2old}^2 + h^2(x_{1old}^2 + x_{2old}^2) \\ &= (1+h^2)r_{old}^2, \end{aligned}$$

and thus the system would spiral outward. Only as we take the limit $h \to 0$ will the path remain circular. Likewise, if we were to use the Backward Euler method for some discrete arc y,

$$y_{n+1} = y_n + hf(y_{n+1}),$$

and applying it to our model we would have

$$\begin{aligned} x_{1new} &= x_{1old} - hx_{2new} \\ x_{2new} &= x_{2old} + hx_{1new}. \end{aligned}$$

We find that for backward Euler the radius changes by a rate of $\frac{1}{1+h^2}$, so the curve spirals inwards. This is because the backward Euler method, in a way, tends to underestimate the actual next value of x and y. One way to remedy this is to use the trapezoidal method, which includes a half-step of forward Euler and a half step of backward Euler. It is written as

$$y_{n+1} = y_n + \frac{h}{2}(f(y_n) + f(y_{n+1})),$$

and applying it to our model we would have

$$\begin{aligned} x_{1new} &= x_{1old} - \frac{h}{2}(x_{2new} + x_{2old}) \\ x_{2new} &= x_{2old} + \frac{h}{2}(x_{1new} + x_{1old}). \end{aligned}$$

In addition to being consistent on the order h^2 and also 0-stable, it can be shown that, for this model, the radius remains constant by using the trapezoidal method. One may also try to implement a forward/backward Euler method in the following way:

$$\begin{aligned} x_{1new} &= x_{1old} - hx_{2old} \text{ forward Euler step} \\ x_{2new} &= x_{2old} + hx_{1new} \text{ backward Euler step.} \end{aligned}$$

It turns out that this causes the discrete arc to become elliptic in nature, more noticeably so as the step size increases. Fortunately, regardless of the simulation method used, it is relatively simple to stabilize the orbit radius, if desired, by realigning the values of x and y at every time step by

$$\begin{aligned} x_1 &= r_0 \frac{x_1}{x_1^2 + x_2^2} \\ x_2 &= r_0 \frac{x_2}{x_1^2 + x_2^2} \end{aligned}$$

where r_0 is the original radius. To compare the behavior of the four systems, we ran them all for a finite time domain and initial value $(x_1, x_2)(0) = (1, 0)$. For a small time step $h = 0.01$ with time domain [0,100] all four graphs behaved the same minus the aforementioned spiraling behaviors. All four systems counted the same number of crossings. See the left half of Figure 7. However, with a larger step of 0.5 and domain [0,200], more troublesome behavior arrived, as seen on the

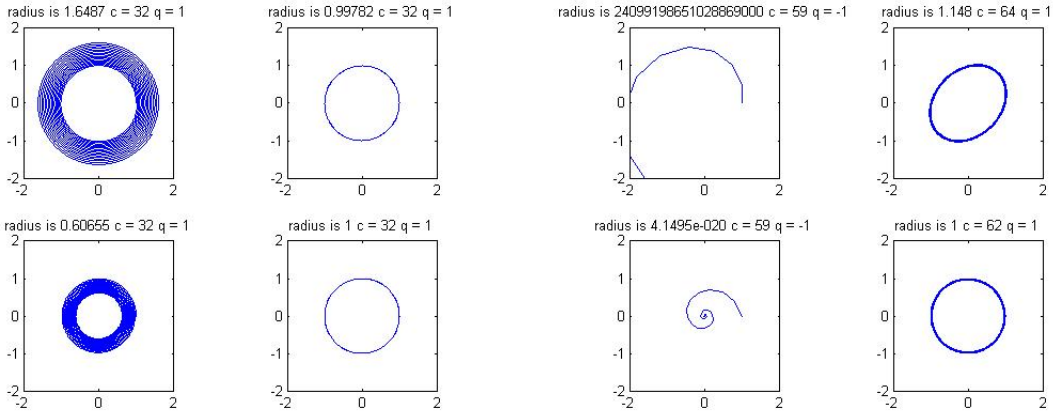

Figure 7: With h sufficiently small, all the simulators exhibited similar behavior, as shown by the four diagrams on the left. The graphs represent, starting from the upper-left and going clockwise, the Forward Euler, Forward/Backward Euler, trapezoidal, and Backward Euler methods. For each subplot, c refers to the number of crossings. With step size h too large, a number of problems arise. Note the more elliptic shape of the Forward/Backward Euler.

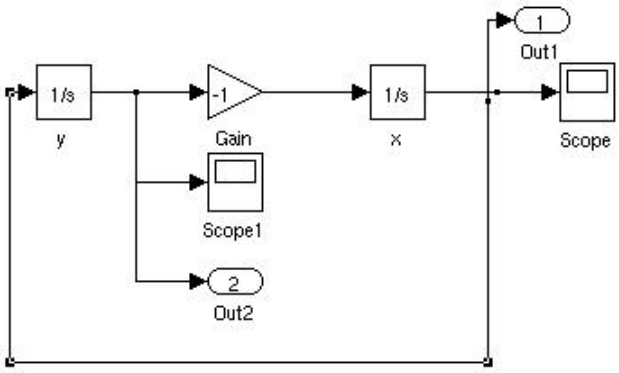

Figure 8: A Simulink model of the rotating particle example. Pre-existing software can be tested to see how it handles ZCD.

rest of Figure 7. The biggest concern comes from the difference in number of zero crossings. The Euler methods may have zero-crossing problems as the values of x and y are becoming quite different than what they started as, particularly the implicit case as the radius goes below machine error. The other two methods also seem to be mismatched. It appears as though the trapezoidal method goes slower than it should, resulting in fewer crossings than there actually are. For the time domain [0,200], there should be 64 crossings. Fortunately, these problem appear to and should go away as $h \to 0$ but they may still be of concern.

Likewise, using mathematical software such as MATLAB or Simulink can help test the ZCD systems under a number of different possible schemes. One cause of concern regarding our ZCD arcs is their uniqueness. This was especially critical at points in which $\hat{C}$ and $\hat{D}$, our simulator domains, had nonzero intersection, as the system could either flow or jump, branching out into multiple possible solutions. We concluded that as long as both our flow and jump functions transposed our hybrid arc to somewhere where the intersection was empty, that is:

$$\begin{aligned} \hat{F}(\hat{C} \cap \hat{D}) \cap (\hat{C} \cap \hat{D}) &= \emptyset \\ \hat{G}(\hat{C} \cap \hat{D}) \cap (\hat{C} \cap \hat{D}) &= \emptyset, \end{aligned}$$

then we should be okay with the uniqueness of our solutions.

Example 7. *Relating this uniqueness to our rotating particle example, we see that* $\hat{C} \cap \hat{D} = \{x | h(x) = 0\}$. *Thus for our*

solutions to be unique, both our jump and flow maps must send the point elsewhere. The flow map does this without a problem. The jump map sends x_1 to zero, so it is possible that the system could try and jump again, but then it could also flow again and no longer be in the intersection and the system is able to evolve.

4 Future Work and Conclusions

In constructing models for our hybrid systems, we would like to have a way to compare the closeness of a discrete arc to its original solution. We know that the numerical methods used are convergent to a specific order for continuously flowing systems, but now that we have the notion of jumps as well, do these ideas of 0-stability and convergence still hold? Intuitively, the answer is "probably," since as we make the step size h closer to zero, we would expect the simulator solution to behave like the true solution. The difficulty lies in the construction of said intuition. How do we account for the displacements caused by the jumps? Also, how can we say that solutions are "close" if they are on different indices j at a given time t? It is understood that while on the same index j one would expect their arcs to be relatively close to each other, but this notion of "discontinuity time," where one arc is on a different index j than another arc, brings about discrepancies and raises the question about convergence. More research needs to be done to consider the hybrid versions of 0-stability and convergence.

Another area worth looking into is the use of an adaptive step size rather than a fixed length, as suggested by [1]. By introducing better refinement where the hybrid arc has more distinct behavior or approaches a zero, we can more accurately model the arc in our simulator. Consider the following pseudocode:

```
if (next iteration crosses a zero)
 then
 go back to previous location
 halve the step size
 if (step size down to machine precision)
  then
  proceed with the next step
  acknowledge zero-crossing
repeat
```

Such an algorithm can be implemented into $\hat{H}$ for analysis.

In conclusion, these hybrid systems implement two different structures which, by themselves, we have come to understand to a certain extent. However, the act of combining them has shown that there are a number of particular instances, such as their numerical simulation, which we must tread over very carefully. The introduction of Zero-Crossing Detection systems helps promote a couple of nice assurances, but not without some minor hiccups. There is room for much more research in this field, and people like Sanfelice, Mosterman, and others will continue to make contributions.

I would sincerely like to thank Dr. Ricardo Sanfelice for taking the time to work with me on this project, as well as the University of Arizona GIDP in Applied Mathematics for making this research possible.

References

[1] Mosterman; Biswas. A hybrid modeling and simualtion methodolgy for dynamic physical systems. *Simulation*, Volume 79, Number 1, 5-17.

[2] Franklin; et al. Feedback control of dynamic systems. Pearson Prentice Hall, 2006.

[3] Stuart; Humphries. Dynamical systems and numerical analysis, volume 8. Cambridge University Press, 1998.

[4] Zhang; Mosterman. Zero-crossing location and detection algorithms for hybrid system simulation. World Congress, Volume 17, Part 1, 2008.

[5] Ascher; Petzold. Computer methods for ordinary differential equations and differential-algebraic equations. SIAM, 1998.

[6] Rockafellar and Wets. Variational analysis. Springer, 1997.

[7] Sanfelice; Teel. Dynamical properties of hybrid systems. *Automatica* 46 236-248, 2010.