15-424: Foundations of Cyber-Physical Systems

Lecture Notes on Events & Responses

André Platzer

Carnegie Mellon University

Lecture 8

1 Introduction

Lecture 3 on Choice & Control demonstrated the importance of control and loops in CPS models, Lecture 5 on Dynamical Systems & Dynamic Axioms presented a way of unwinding loops iteratively to relate repetition to runs of the loop body, Lecture 6 on Truth & Proof showed a corresponding way of unwinding loops in sequent calculus, and Lecture 7 on Control Loops & Invariants finally explained the central proof principle for loops based on induction using invariants.

That has been a lot of attention on loops, but there are even more things to be learned about loops. Not by coincidence, because loops are one of the difficult challenges in CPS. The other difficult challenge comes from the differential equations. If the differential equations are simple and there are no loops, CPS suddenly become easy (they are even decidable [Pla12a]).

This lecture will focus on how these two difficult parts of CPS interact: how loops interface with differential equations. That interface is ultimately the connection between the cyber and the physical part, which, as we know since Lecture 2 on Differential Equations & Domains, is fundamentally represented by the evolution domain constraints that determine when physics pauses to let cyber look and act.

Today's and the next lecture focuses on two important paradigms for making cyber interface with physics to form cyber-physical systems. Both paradigms played an equally important role in classical embedded systems. One paradigm is that of *event-driven control*, where responses to events dominate the behavior of the system and an action is taken whenever one of the events is observed. The other paradigm is *time-triggered control*, which uses periodic actions to affect the behavior of the system at certain frequencies. Both paradigms follow naturally from an understanding of the hybrid program principle for CPS. Event-driven control will be studied in this lecture, while time-triggered control will be pursued in the next lecture.

These lecture notes are loosely based on [Pla12b, Pla10].

Based on the understanding of loops from Lecture 7 on Loops & Invariants, the most important learning goals of this lecture are:

Modeling and Control: Today's lecture provides a number of crucial lessons for modeling CPS. We develop an understanding of one important design paradigm for control loops in CPS: event-driven control. This lecture studies ways of developing models and controls corresponding to this feedback mechanism, which will turn out to be surprisingly subtle to model. Today's lecture focuses on CPS models assuming continuous sensing, which assumes that sensor data is available and can be checked all the time.

Computational Thinking: This lecture uses the rigorous reasoning approach from Lecture 5 on Dynamical Systems & Dynamic Axioms and particularly the rigorous reasoning approach for CPS loops from Lecture 7 on Loops & Control to study CPS models with event-driven control. A discussion of reasoning for CPS models with time-triggered control will be pursued in the next lecture. As a running example, the lecture continues to develop the bouncing ball that has served us so well for conveying subtleties of hybrid system models in an intuitive example. This time, we add control decisions to the bouncing ball, turning it into a ping pong ball, which retains the intuitive simplicity of the bouncing ball, while enabling us to develop generalizable lessons about how to design event-driven control systems correctly. The lecture will also crucially study invariants and show a development of the powerful technique of design-by-invariant in a concrete example. While the lecture could hardly claim showing how to verify CPS models of appropriate scale, the basics laid in this lecture definitely carry significance for numerous practical applications.

CPS Skills: This lecture develops an understanding for the precise semantics of event-driven control, which can often be surprisingly subtle even if superficially simple. This understanding of the semantics will also guide our intuition of the operational effects caused by event-driven control. Finally, the lecture shows a brief first glimpse of higher-level model-predictive control, even if that topic will have to be followed up on in much more detail later in the course.

2 The Need for Control

Having gotten accustomed to the little bouncing ball, this lecture will simply stick to it. Yet, the bouncing ball asks for more action, for it had so far no choice but to wait until it was at ground height $x = 0$. And when its patience paid off so that it finally observed height $x = 0$, then its only action was to make its velocity bounce back up. Frustrated by this limited menu of actions to choose from, the bouncing ball begs for a ping pong paddle. Thrilled at the opportunities opened up by a ping pong paddle, the bouncing ball first performs some experiments and then settles on using the ping pong paddle high up in the air to push itself back down again. It had high hopes that proper control exerted by the ping pong paddle at just the right moments would allow the ball to go faster without risking the terrified moments inflicted on it by its acrophobic attitude to heights. Setting aside all Münchausian concerns about how effective ping pong paddles can be for the ball if the ball is using the paddle on itself in light of Newton's third law about opposing forces, let us investigate this situation regardless.[1] After all, the ping-pong-crazy bouncing ball still has what it takes to make control interesting: the dynamics of a physical system and decisions on when to react and how to react to the observed status of the system.

Lecture 7 on Loops & Invariants developed a sequent proof of the undamped bouncing ball with repetitions:

$$
\begin{aligned}
&0 \le x \land x = H \land v = 0 \land g > 0 \land 1 = c \rightarrow \\
&\qquad [\big(x' = v, v' = -g \,\&\, x \ge 0; (?x = 0; v := -cv \cup ?x \ge 0)\big)^*](0 \le x \land x \le H) \quad (1)
\end{aligned}
$$

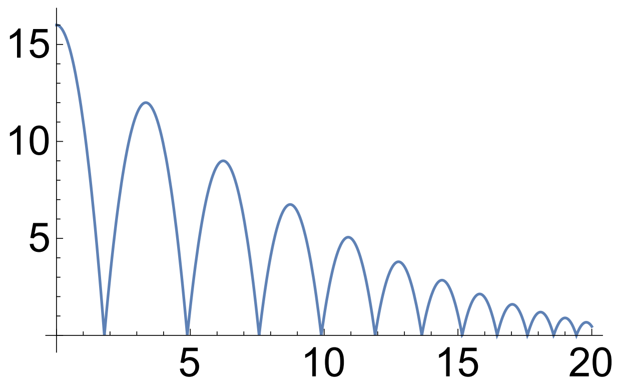

Figure 1: Sample trajectory of a bouncing ball (plotted as position over time)

With this pretty complete understanding of (undamped) bouncing balls, let's examine how to turn the simple bouncing ball into a fancy ping pong ball using clever actuation of a ping pong paddle. The bouncing ball tried to actuate the ping pong paddle in all kinds of directions. But it never knew where it was going to land if it tried the ping pong paddle sideways. So it quickly gave up the thought of using the ping pong paddle sideways. The ball probably got so accustomed to its path of going up and down

[1] If you find it hard to imagine a bouncing ball that uses a ping pong paddle to pad itself on its top to propel itself down to the ground again, just step back and consider the case where the ping pong ball has a remote control to activate a device that moves the ping pong paddle. That will do as well, but is less fun. Besides, Baron Münchhausen would be horribly disappointed if we settled for such a simple explanation for the need of control.

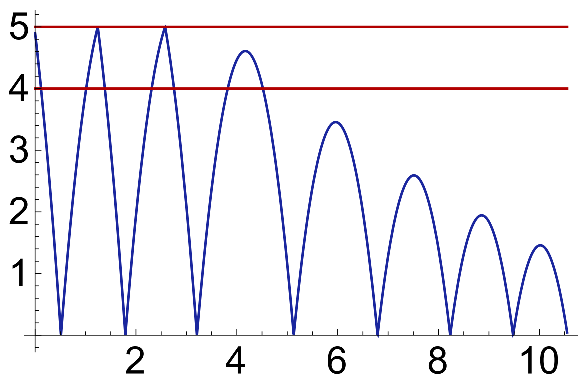

Figure 2: Sample trajectory of a ping pong ball (plotted as position over time) with the indicated ping pong paddle actuation range

on the spot that it embraced the thought of keeping it that way. With the ping pong paddle, it wanted to do the same, just faster.

By making the ping pong paddle move up and down, the bouncing ball ultimately figured out that the ball would go back down pretty fast as soon as it got a pat on the top by the paddle. It also learned that the other direction turned out to be not just difficult but also rather dangerous. Moving the ping pong paddle up when the ball was above it to give it a pat on the bottom was first of all rather tricky, but when it worked would, furthermore, make the ball fly up even higher than before. Yet, that is what the acrophobic bouncing ball did not enjoy at all, so it tries to control the ping pong paddle so that the ping pong paddle only ever bounces the ball down, never up.

As a height that the bouncing ball feels comfortable with, it chooses the magic number 5 and so it wants to establish $0 \leq x \leq 5$ to always hold as its favorite safety condition. The ball further installs the ping pong paddle at a similar height so that it can actuate somewhere between 4 and 5. It exercises great care to make sure it would ever only move the paddle downwards when the ball is underneath, never above, because that would take it frightfully high up. Thus, the effect of the ping pong paddle will only be to reverse the ball's direction. For simplicity, the ball figures that being hit by a ping pong paddle might have a similar effect as being hit by the floor, except with a possibly different bounce factor $f \geq 0$ instead of the damping coefficient c.[2] So the paddle actuated this way is simply assumed to have the effect $v := -fv$. Since the bouncing ball can decide to use the ping pong paddle as it sees fit (within the ping pong paddle's reach between height 4 and 5), the ping pong model is obtained from the bouncing ball model by adding this additional (nondeterministic) choice to the HP. A sample trajectory for the ping pong ball, where the ping pong paddle is used twice is illustrated in Fig. 2. Observe how the use of the ping pong paddle (here only at height $x = 5$) makes the ball bounce back faster.

Taking these thoughts into account, the ball devises an HP and conjectures safety as

[2]The real story is quite a bit more complicated, but the bouncing ball does not know any better.

expressed in the following $\mathsf{d}\mathcal{L}$ formula:

$$\begin{aligned}&0 \le x \wedge x \le 5 \wedge v \le 0 \wedge g > 0 \wedge 1 \ge c \ge 0 \wedge f \ge 0 \to\\ &\quad \big[\big(x' = v, v' = -g \,\&\, x \ge 0;\\ &\qquad (?x = 0; v := -cv \cup ?4 \le x \le 5; v := -fv \cup ?x \ge 0)\big)^*\big](0 \le x \le 5)\end{aligned} \tag{2}$$

Having taken the *Principle of Cartesian Doubt* from Lecture 4 on Safety & Contracts to heart, the aspiring ping-pong ball first scrutinizes conjecture (2) before setting out to prove it. What could go wrong?

For one thing, (2) allows the right control options of using the paddle by $?4 \le x \le 5; v := -fv$ but it also always allows the wrong choice $?x \ne 0$ when above ground. Remember that nondeterministic choices are just that: nondeterministic. So if the bouncing ball is unlucky, the HP in (2) could run so that the middle choice is never chosen and, if the ball has a large downwards velocity v initially, it will jump back up higher than 5 even if it was below 5 initially. That scenario falsifies (2) and a concrete counterexample can be constructed correspondingly, e.g., from initial state ν with

$$\nu(x) = 5, \nu(v) = -10^{10}, \nu(c) = \frac{1}{2}, \nu(f) = 1, \nu(g) = 10$$

Despite this setback in its first control attempt, the bouncing ball is thrilled by the extra prospects of a proper control decision for it to be made. So the bouncing ball "only" needs to figure out how to restrict the control decisions such that nondeterminism will only ever take one of the (possibly many) correct control choices, quite a common problem in CPS control. How can the bouncing ball fix this bug in its control and turn itself into a proper ping pong ball? The problem with the controller in (2) is that it permits too much choice, some of which are unsafe. Restricting these choices and making them more deterministic is what it takes to ensure the ping pong paddle is actuated as intended.

$$\begin{aligned}&0 \le x \wedge x \le 5 \wedge v \le 0 \wedge g > 0 \wedge 1 \ge c \ge 0 \wedge f \ge 0 \to\\ &\quad \big[\big(x' = v, v' = -g \,\&\, x \ge 0;\\ &\qquad (?x = 0; v := -cv \cup ?4 \le x \le 5; v := -fv \cup ?x \ge 0 \wedge x < 4 \vee x > 5)\big)^*\big](0 \le x \le 5)\end{aligned} \tag{3}$$

Recalling the $\mathtt{if}(E)\,\alpha\,\mathtt{else}\,\beta$ statement, the same system can be modeled equivalently:

$$\begin{aligned}&0 \le x \wedge x \le 5 \wedge v \le 0 \wedge g > 0 \wedge 1 \ge c \ge 0 \wedge f \ge 0 \to\\ &\quad \big[\big(x' = v, v' = -g \,\&\, x \ge 0;\\ &\qquad (?x = 0; v := -cv \cup ?x \ne 0; \mathtt{if}(4 \le x \le 5)\, v := -fv)\big)^*\big](0 \le x \le 5)\end{aligned}$$

Or, even shorter as the equivalent

$$\begin{aligned}&0 \le x \wedge x \le 5 \wedge v \le 0 \wedge g > 0 \wedge 1 \ge c \ge 0 \wedge f \ge 0 \to\\ &\quad \big[\big(x' = v, v' = -g \,\&\, x \ge 0;\\ &\qquad \mathtt{if}(x = 0)\, v := -cv\, \mathtt{else}\, \mathtt{if}(4 \le x \le 5)\, v := -fv\big)^*\big](0 \le x \le 5)\end{aligned} \tag{4}$$

Is conjecture (4) valid?

Before you read on, see if you can find the answer for yourself.

3 Events in Control

The problem with the controller in (4) is that, even though it exercises the appropriate control choice whenever the controller runs, the model does not ensure the controller would run at all when needed. The paddle control only runs after the differential equation stops, which could be almost any time. The differential equation is only guaranteed to stop when the ball bounces down to the ground ($x = 0$), because its evolution domain constraint $x \geq 0$ would not be satisfied any longer on its way down. Above ground, the differential equation model does not provide any constraints on how long it might evolve. Recall from Lecture 2 on Differential Equations & Domains that the semantics of differential equations is nondeterministic in that the system can follow a differential equation *any amount of time*, as long as it does not violate the evolution domain constraints. In particular, the HP in (4) could miss the interesting *event* $4 \leq x \leq 5$ that the ping pong ball's paddle control wanted to respond to. The system might simply skip over that region by following the differential equation $x' = v, v' = -g \,\&\, x \geq 0$ obliviously until the event $4 \leq x \leq 5$ has passed.

How can the HP from (4) be modified to make sure that the event $4 \leq x \leq 5$ will always be noticed and never missed?

Before you read on, see if you can find the answer for yourself.

The "only" way to prevent the system from following a differential equation for too long is to restrict the evolution domain constraint, which is the predominant way to make cyber and physics interact. Indeed, that is what the evolution domain constraint $\ldots \,\&\, x \geq 0$ in (4) did in the first place. Even though this domain was introduced for different reasons (first principle arguments that light balls never fall through solid ground), its secondary effect was to make sure that the ground controller $?x = 0; v := -cv$ will never miss the right time to take action and reverse the direction of the ball from falling to climbing.

Note 1 (Evolution domains detect events)**.** *Evolution domain constraints of differential equations in hybrid programs can detect events. That is, they can make sure the system evolution stops whenever an event happens on which the control wants to take action. Without such evolution domain constraints, the controller is not necessarily guaranteed to execute but may miss the event.*

Following these thoughts further indicates that the evolution domain somehow ought to be augmented with more constraints that ensure the interesting event $4 \leq x \leq 5$ will never be missed accidentally. How can this be done? Should the event be conjoined to the evolution domain as follows

$$
\begin{aligned}
&0 \leq x \land x \leq 5 \land v \leq 0 \land g > 0 \land 1 \geq c \geq 0 \land f \geq 0 \rightarrow \\
&\qquad \big[\big(x' = v, v' = -g \,\&\, x \geq 0 \land \boldsymbol{4 \leq x \leq 5};\\
&\qquad\quad \mathtt{if}(x = 0)\, v := -cv \,\mathtt{else}\,\mathtt{if}(4 \leq x \leq 5)\, v := -fv\big)^*\big](0 \leq x \leq 5)
\end{aligned}
$$

Before you read on, see if you can find the answer for yourself.

Of course not! This evolution domain would be entirely counterfactual and require the ball to always be at height between 4 and 5, which is hardly the right physical model. How could the ball ever fall on the ground and bounce back, this way? It couldn't.

Yet, on second thought, the way the event $\ldots \,\&\, x = 0$ got detected by the HP in the first place was not by including $x = 0$ in the evolution domain constraint, but by including the inclusive limiting constraint $\ldots \,\&\, x \geq 0$, which made sure the system could perfectly well evolve before the event domain $x = 0$, but that it just couldn't miss the event rushing past the event $x = 0$. What would the inclusion of such an inclusive limiting constraint correspond to for the intended ping pong paddle event $4 \leq x \leq 5$?

When the ball is hurled up into the sky, the last point at which action has to be taken to make sure not to miss the event $4 \leq x \leq 5$ is $x = 5$. The corresponding inclusive limiting constraint $x \leq 5$ thus should be somewhere in the evolution domain constraint.

$$\begin{aligned}
&0 \leq x \land x \leq 5 \land v \leq 0 \land g > 0 \land 1 \geq c \geq 0 \land f \geq 0 \rightarrow \\
&\quad \big[\big(x' = v, v' = -g \,\&\, x \geq 0 \land \boldsymbol{x \leq 5}; \\
&\quad\quad \mathtt{if}(x = 0)\, v := -cv \;\mathtt{else}\; \mathtt{if}(4 \leq x \leq 5)\, v := -fv\big)^*\big](0 \leq x \leq 5)
\end{aligned} \tag{5}$$

Is this the right model? Is dL formula (5) valid? Will its HP ensure that the critical event $4 \leq x \leq 5$ will not be missed out on?

Before you read on, see if you can find the answer for yourself.

Formula (5) is valid. And, yet, (5) *is not at all the appropriate formula to consider!* It is crucial to understand why.

First, however, note that the hybrid program in (5) allows the use of the ping pong paddle anywhere in between the height range $4 \leq x \leq 5$. Its evolution domain constraint enforces that this event $4 \leq x \leq 5$ will be noticed at the latest at height $x = 5$. So when exactly the ping pong paddle is exercised in that range is nondeterministic (even if the control is written deterministically), because the duration of the differential equation is still chosen nondeterministically. This allows the ping pong paddle to be controlled at the last height $x = 5$ or before it reaches height $x = 5$ as in Fig. 3.

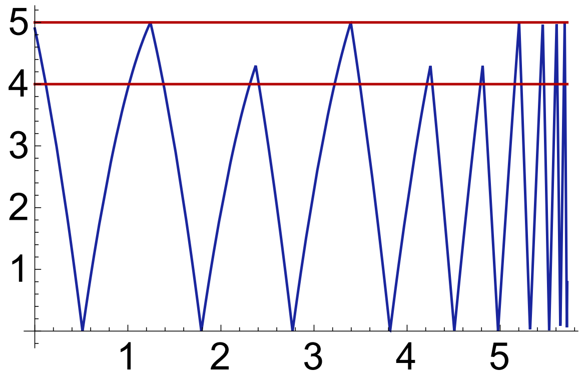

Figure 3: Sample trajectory of a ping pong ball (plotted as position over time) with the indicated ping pong paddle actuation range, sometimes actuating early, sometimes late

Notice, however, that (5) does not make sure that the critical event $4 \leq x \leq 5$ will not be missed out on in the case of a ball that is bouncing up above the lower trigger 4 but starts falling down again already before it exceeds the upper trigger 5 of the event. Such a possible behavior of the ping pong ball was already shown in Fig. 2. Yet, this is not actually problematic, because missing out on the chance of actuating the ping pong paddle in a situation where it is not needed to ensure height control is just missing an opportunity for fun, not missing a critical control choice.

But there is a much deeper problem with (5). So, formula (5) is perfectly valid. But why? Because all runs of the differential equation $x' = v, v' = -g \,\&\, x \geq 0 \land x \leq 5$ remain within the safety condition $0 \leq x \leq 5$ *by construction*. None of them are ever allowed to leave the region $x \geq 0 \land x \leq 5$, which, after all, is their evolution domain constraint. So formula (5) is trivially safe, because it says that a system that is constrained to not leave $x \leq 5$ cannot leave $x \leq 5$, which is a rather trivial insight since $5 = 5$. A more careful argument involves that, every time around the loop, the postcondition holds trivially, because the differential equation's evolution constraint maintains it by definition, the subsequent discrete control never changes the only variable x on which the postcondition depends. Hold on, the loop does not have to run but could be skipped over by zero iterations as well. Yet, in that case, the precondition ensures the postcondition, so, indeed, (5) is valid, but only trivially so.

Note 2 (Non-negotiability of Physics)**.** *It is a good idea to make systems safe by construction. For computer programs, that is a great idea. But we need to remember that physics is unpleasantly non-negotiable. So if the only reason why a CPS model is safe is because we forgot to model all relevant behavior of the real physical system, then correctness statements about those inadequate models are not particularly applicable to reality.*

One common cause for counterfactual models are too restrictive evolution domain constraints that rule out physically realistic behavior.

And that is what happened in (5). The bouncing ball got so carried away with trying not to miss the event $4 \leq x \leq 5$ that it forgot to include a behavior in the model that takes place after the event has happened.

Contrast this with the role of the evolution domain constraint $\dots \,\&\, x \geq 0$, which came into the system because of physics: to model the guaranteed bouncing back on the ground and to prevent the ball from falling through the ground. The constraint $x \geq 0$ is there for physical reasons. It models the physical limitations of balls which cannot fall through solid soil. The evolution domain constraint $\dots \,\&\, x \leq 5$ got added to the ping pong HP for an entirely different reason. It came into play to model what our controller does, and inaptly so, because our feeble attempt ruled out physical behavior that could actually have happened in reality. There is no reason to believe that physics would be so kind to only evolve within $x \leq 5$ just because our controller model wants to respond to an event then. Remember never to do that ever.

Note 3 (Physical constraints versus control constraints)**.** *Some constraints of the system models are included for physical reasons, other constraints are added later to describe the controllers. Take care to ensure not to accidentally limit the behavior of physics when all you meant to do is impose a constraint on your system controller. Physics will not listen to your desire. This applies to evolution domain constraints but also other aspects of your system model such as tests. It is fine, for example, to limit the force that the ping pong paddle is exerting, because that is for the controller to decide. But it is not necessarily a good idea for a controller to limit or change the values of gravity or damping coefficients, because that is rather hard to implement without leaving the planet.*

Let's make up for this modeling mishap by developing a model that has both behaviors, the behaviors before and after the event, just in different continuous programs so that the decisive event in the middle could not accidentally have been missed.

$$\begin{aligned}&0 \leq x \wedge x \leq 5 \wedge v \leq 0 \wedge g > 0 \wedge 1 \geq c \geq 0 \wedge f \geq 0 \rightarrow \\ &\quad \big[\big(((x' = v, v' = -g \,\&\, x \geq 0 \wedge x \leq 5) \cup (\boldsymbol{x' = v, v' = -g \,\&\, x > 5}));\\ &\quad\quad \mathtt{if}(x = 0)\, v := -cv \;\mathtt{else}\;\mathtt{if}(4 \leq x \leq 5)\, v := -fv\big)^*\big](0 \leq x \leq 5)\end{aligned} \tag{6}$$

Instead of the single differential equation with a single evolution domain constraint in (5), the HP in (6) has a (nondeterministic) choice between two differential equations, actually both the same, with two different evolution domain constraints. The left continuous system is restricted to the lower physics space $x \geq 0 \land x \leq 5$, the right continuous system is restricted to the upper physics space $x > 5$. Every time the loop repeats, there is a choice of either the lower physics equation or the upper physics equation. But the system can never stay in these differential equations for too long, because, e.g., when the ball is below 5 and speeding upwards very fast, then it cannot stay in the left differential equation above height 5, so it will have to stop evolving continuously and give the subsequent controller a chance to inspect the state and respond in case the event $4 \leq x \leq 5$ happened.

Now d$\mathcal{L}$ formula (6) has a much better model of events than the ill-advised (5). Is formula (6) valid?

Before you read on, see if you can find the answer for yourself.

The model in (6) is, unfortunately, quite horribly broken. We meant to split the continuous evolution space into the regions before and after the event $4 \leq x \leq 5$. But we overdid it, because the space is now fractured into two disjoint regions, the lower physics space $x \geq 0 \land x \leq 5$ and the upper physics space $x > 5$. How could the ping pong ball ever transition from one to the other? Certainly, as the ball moves upwards within lower physics space $x \geq 0 \land x \leq 5$, it will have to stop evolving at $x = 5$ at the latest. But then even if the loop repeats, the ball still could not continue in the upper physics space $x > 5$, because it is not quite there yet. It is an infinitesimal step away from $x > 5$. Of course, the bouncing ball will only ever move continuously along a differential equation. There is no continuous motion that would take the ball from the region $x \geq 0 \land x \leq 5$ to the disjoint region $x > 5$. In other words, the HP in (6) has accidentally modeled that there will never ever be a transition from lower to upper physics space nor the other way around, because of an infinitesimal gap in between.

Note 4 (Connectedness & disjointness in evolution domains)**.** *Evolution domain constraints need to be thought out carefully, because they determine the respective regions within which the system can evolve. Disjoint or unconnected evolution domain constraint regions often indicate that the model will have to be thought over again, because there cannot be any continuous transitions from one domain to the otherif they are not connected.*

Let's close the infinitesimal gap between $x \geq 0 \land x \leq 5$ and $x > 5$ by including the boundary $x = 5$ in both domains:

$$\begin{aligned} &0 \leq x \land x \leq 5 \land v \leq 0 \land g > 0 \land 1 \geq c \geq 0 \land f \geq 0 \rightarrow \\ &\quad \big[\big(((x' = v, v' = -g \,\&\, x \geq 0 \land x \leq 5) \cup (x' = v, v' = -g \,\&\, \boldsymbol{x \geq 5})); \\ &\quad\quad \texttt{if}(x = 0)\, v := -cv \,\texttt{else}\, \texttt{if}(4 \leq x \leq 5)\, v := -fv\big)^*\big](0 \leq x \leq 5) \end{aligned} \tag{7}$$

Now there is a proper separation into lower physics $x \geq 0 \land x \leq 5$ and upper physics $x \geq 5$ but the system can be in either physics space at the switching boundary $x = 5$. This makes it possible for the ball to pass from lower physics into upper physics or back, yet only on the boundary $x = 5$, which, in this case, is the only point that the two evolution domain constraints have in common.

Now d$\mathcal{L}$ formula (7) has a much better model of events than the ill-advised (5). Is formula (7) valid?

Before you read on, see if you can find the answer for yourself.

When the ball is jumping up from the ground, the model in (7) makes it impossible for the controller to miss the event $4 \leq x \leq 5$, because the only evolution domain constraint in the HP that applies at the ground is $x \geq 0 \land x \leq 5$. And that evolution domain stops being true above 5. Yet, suppose the ping pong ball was jumping up from the ground following the continuous program in the left choice and then stopped its evolution at height $x = 4.5$, which always remains perfectly within the evolution domain $x \geq 0 \land x \leq 5$ and is, thus, allowed. Then, after the sequential composition between the middle and last line of (7), the controller in the last line of (7) runs, notices that the formula $4 \leq x \leq 5$ for the event checking is true, and changes the velocity according to $v := -fv$, corresponding to the assumed effect of a pat with the paddle. That is actually its only choice in such a state, because the controller is deterministic, much unlike the differential equation. Consequently, the velocity has just become negative since it was positive before as the ball was climbing up. So the loop can repeat and the differential equation runs again. Yet, then the differential equation might evolve until the ball is at height $x = 4.25$, which will happen since its velocity is negative. If the differential equation stops then, the controller will run again, determine that $4 \leq x \leq 5$ is true still and so take action to change the velocity to $v := -fv$ again. That will, however, make the velocity positive again, since it was previously negative as the ball was in the process of falling. Hence, the ball will keep on climbing now, which, again, threatens the postcondition $0 \leq x \leq 5$. Will this falsify (7) or is it valid?

Before you read on, see if you can find the answer for yourself.

On second thought, that alone still will not cause the postcondition to evaluate to $false$, because the only way the bouncing ball can evolve continuously from $x = 4.25$ is still by the continuous program in the left choice of (7). And that differential equation is restricted to the evolution domain $x \geq 0 \land x \leq 5$, which causes the controller to run before leaving $x \leq 5$. That is, the event $4 \leq x \leq 5$ will again be noticed by the controller so that the ball is ping pong paddle pats the ball back down; see Fig. 4.

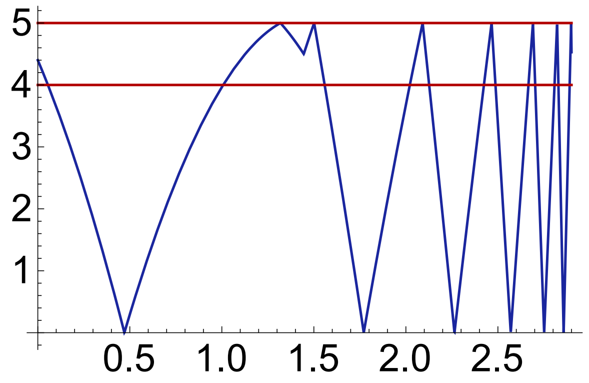

Figure 4: Sample trajectory of a ping pong ball (plotted as position over time) with the controller firing multiple times for the same event

However, the exact same reasoning applies also to the case where the ball successfully made it up to height $x = 5$, which is the height at which any climbing ball has to stop its continuous evolution, because it would otherwise violate the evolution domain $x \geq 0 \land x \leq 5$. As soon as that happens, the controller runs, notices that the event $4 \leq x \leq 5$ came true and responds with a ping pong paddle to cause $v := -fv$. If, now, the loop repeats, yet the continuous evolution evolves for duration zero only, which is perfectly allowed, then the condition $4 \leq x \leq 5$ will still be true so that the controller again notices this "event" and responds with ping pong paddle $v := -fv$. That will make the velocity positive, the loop can repeat, the continuous program on the right of the choice can be chosen since $x \geq 5$ holds true, and then the bouncing ball can climb and disappear into nothingness high up in the sky if only its velocity has been large enough. Such a behavior is shown in Fig. 5. The second illustration in Fig. 5 uses the artistic liberty of delaying the second ping pong paddle use just a tiny little bit to make it easier to see the two ping pong paddle uses separately, even if that is not actually quite allowed by the HP model, because such behavior would actually be reflected by a third ping pong paddle use as in Fig. 4.

Ergo, (7) is not valid. What a pity! The poor bouncing ball would still have to be afraid of heights when following the control in (7). How can this problem be resolved?

Before you read on, see if you can find the answer for yourself.

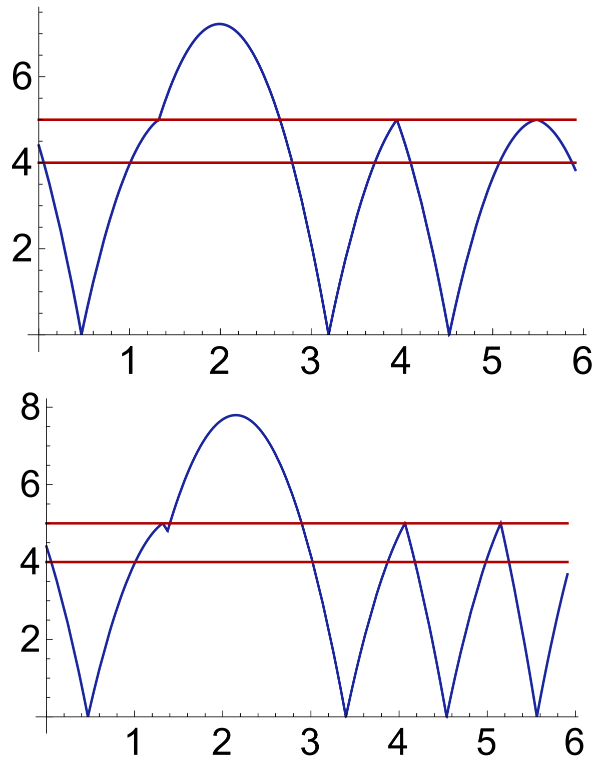

Figure 5: Sample trajectory of a ping pong ball (plotted as position over time) with the controller firing multiple times for the same event on the event boundary $x = 5$ within lower and upper physics

The problem in (7) is that its left differential equation makes sure never to miss out on the event $4 \le x \le 5$ but its control may respond to it multiple times. It is not even sure whether each occasion of $4 \le x \le 5$ should be called an event. But certainly repeated responses to the same event according to control (7) causes trouble.

Note 5 (Multi-firing of events)**.** *In event-driven control, exercise care to ensure whether you want events to fire only once when they occur for the first time, or whether the system stays safe even if the same event is detected and responded to multiple times in a row.*

One way of solving this problem is to change the condition in the controller to make sure it only responds to the $4 \le x \le 5$ event when the ball is on its way up, i.e. when its velocity is not negative ($v \ge 0$). That is what the bouncing ball wanted to ensure in any case. The ping pong paddle should only be actuated downwards when the ball is flying up.

These thoughts lead to the following variation:

$$\begin{aligned}&0 \le x \land x \le 5 \land v \le 0 \land g > 0 \land 1 \ge c \ge 0 \land f \ge 0 \to \\ &\quad \big[\big(\big((x' = v, v' = -g \,\&\, x \ge 0 \land x \le 5) \cup (x' = v, v' = -g \,\&\, x \ge 5)\big); \\ &\qquad \mathtt{if}(x = 0)\, v := -cv \,\mathtt{else}\, \mathtt{if}(4 \le x \le 5 \land \boldsymbol{v \ge 0})\, v := -fv\big)^*\big](0 \le x \le 5)\end{aligned} \tag{8}$$

Because the paddle action $v := -fv$ will disable the condition $v \ge 0$ for nonzero velocities (Exercise 1), the controller in (8) can only respond once to the event $4 \le x \le 5$ to turn the upwards velocity into a downwards velocity, scaled by f. Unlike in (7), this control decision cannot be reverted inadvertently by the controller.

Is dL formula (8) valid?

Before you read on, see if you can find the answer for yourself.

Yes, formula (8) is valid. Note that it is still the case in (8) that, every time around the loop, there will be a nondeterministic choice to evolve within lower physics $x \geq 0 \land x \leq 5$ or within upper physics $x \geq 5$. This choice is nondeterministic, so any outcome will be possible. If the left differential equation is chosen, the subsequent continuous evolution must be confined to $x \geq 0 \land x \leq 5$ and stop before leaving that lower physics region to give the controller a chance to check for events and respond. If the right differential equation is chosen, the subsequent continuous evolution must be limited to $x \geq 5$ and must stop before leaving that upper physics region to give the controller a chance to inspect. In fact, the only way of leaving the upper physics space is downwards (with velocity $v < 0$), which, unlike in (7), will not trigger a response from the subsequent control in (8), because that controller checks for $v \geq 0$.

How could $\mathsf{d}\mathcal{L}$ formula (8) be proved, so that we have unquestionable evidence that it is, indeed, valid? The most critical element of a proof is finding a suitable invariant. What could be the invariant for proving (8)?

Before you read on, see if you can find the answer for yourself.

The formula

$$5 \geq x \geq 0 \tag{9}$$

is an obvious candidate for an invariant. If it is true, it trivially implies the postcondition $0 \leq x \leq 5$ and it holds in the initial state. It is not inductive, though, because a state that satisfies (9) could follow the second differential equation if it satisfies $x \geq 5$. In that case, if the velocity is positive, the invariant (9) would be violated immediately. Hence, at the height $x = 5$, the control has to make sure that the velocity is negative, so that the right differential equation in (8) has to stop immediately. Could (9) be augmented with a conjunction $v \leq 0$ to form an invariant?

$$5 \geq x \geq 0 \land v \leq 0$$

No, that would not work either, because the bounce on the ground immediately violates that invariant, because the whole point of bouncing is that the velocity will become positive again. In fact, the controller literally only ensures $v \leq 0$ at the event, which is detected at $x = 5$ at the latest. Gathering these thoughts, it turns out that the d$\mathcal{L}$ formula (8) can be proved in the d$\mathcal{L}$ calculus using the invariant:

$$5 \geq x \geq 0 \land (x = 5 \to v \leq 0) \tag{10}$$

This invariant retains that the possible range of x is safe but is just strong enough to also remember the correct control choice at the event boundary $x = 5$. It expresses that the ball is either in lower physics space or at the boundary of both physics spaces. But if the ball is at the boundary of the physics spaces, then it is moving downward.

That is the reason why (10) is easily seen to be an invariant of (8). The invariant (8) is initially true, because the ball is initially in range and moving down. The invariant trivially implies the postcondition, because it consists of the postcondition plus an extra conjunction. The inductive step is most easily seen by considering cases. If the position before the loop body ran was $x < 5$, then the only physics possible to evolve is lower physics, which, by construction, implies the conjunct $5 \geq x \geq 0$ from its evolution domain constraint. The extra conjunct $x = 5 \to v \leq 0$ is true after the loop body ran, because, should the height actually be 5, which is the only case for which this extra conjunct is not already vacuously true, then the controller made sure to turn the velocity downwards by checking $4 \leq x \leq 5 \land v \geq 0$ and negating the velocity. If the position before the loop body ran was $x \geq 5$ then the invariant (10) implies that the only position it could have had is $x = 5$ in which case either differential equation could be chosen. Except, if the first differential equation is chosen, the reasoning for inductiveness is as for the case $x < 5$. If the second differential equation is chosen, then the invariant (10) implies that the initial velocity is $v \leq 0$, which implies that the only possible duration that keeps the evolution domain constraint $x \geq 5$ of the upper physics true is duration 0, after which nothing changed so the invariant still holds.

Observe how the scrutiny of a proof, which necessitated the transition from the broken invariant (9) to the provable invariant (10), would have pointed us to subtleties with events and how ping pong balls would become unsafe if they fired repeatedly. We found these issues out by careful formal modeling with our "safety first" approach and a good dose of Cartesian Doubt. But had we not noticed it, the proof would have not let us get away with such oversights, because the (unreflected) invariant candidate (9) would not have worked, nor would the broken controller (7) have been provable.

Finally, recall that (global) invariants need to be augmented with the usual mundane assumptions about the unchanged variables, like $c \geq 0 \land g > 0 \land f \geq 0$.

See ≪Event-driven ping pong ball KeYmaera model≫

The model that (8) and the other controllers in this section adhere to is called event-driven control or sometimes also called event-driven architecture.

Note 6 (Event-driven control)**.** *One common paradigm for designing controllers is event-driven control, in which the controller runs in response to certain events that happen in the system. The controller could possibly run under other circumstances as well—when in doubt, the controller simply skips over without any effect if it does not want to change anything about the behavior of the system. But event-driven controllers assume they will run for sure whenever certain events in the system happen.*

These events cannot be all too narrow, or else the system will not be implementable, though. For example, it is nearly impossible to build a controller that responds exactly at the point in time when the height of the bouncing ball is $x = 4.12345$. Chances are high that any particular execution of the system will have missed this particular height. Care must be taken in event-driven design models also that the events do not inadvertently restrict the evolution of the system to the behavioral cases outside or after the events have happened. Those executions must still be verified.

Are we sure in model (8) that events are taken into account faithfully? That depends on what exactly we mean by an event like $4 \leq x \leq 5$. Do we mean that this event happens for the first time? Or do we mean every time this event happens? If multiple successive runs of the ping pong ball's controller see this condition satisfied, do these count as the same or separate instances of that event happening? Comparing the validity of (7) with the non-validity of (7) illustrates that these subtleties can have considerable impact on the system. Hence, a precise understanding of events and careful modeling is required.

The controller in (8) only takes an action for event $4 \leq x \leq 5$ when the ball is on the way up. Hence, the evolution domain constraint in the right continuous evolution is $x \geq 5$. Had we wanted to model the occurrence of event $4 \leq x \leq 5$ also when the ball is on its way down, then we would have to have a differential equation with evolution domain $x \geq 4$ to make sure the system does not miss $4 \leq x \leq 5$ when the ball is on its way down either, without imposing that it would have to notice $x = 5$ already. This could be achieved by splitting the evolution domain regions appropriately, but was not necessary for (8) since it never responds to balls falling down, only those climbing up.

Note 7 (Subtleties with events)**.** *Events are a slippery slope and great care needs to be exercised to use them without introducing an inadequate executional bias into the model.*

There is a highly disciplined way of defining, detecting, and responding to general events in differential dynamic logic based on the there and back again axiom of differential dynamic logic [Pla12a]. That is, however, much more complicated than the simpler account shown here.

Finally, notice that the proof for (8) was entirely independent of the differential equation and just a consequence of the careful choices of the evolution domain constraint to reflect the events of interest as well as getting the controller responses to these events right. That is, ultimately, the reason why the invariant (10) could be so simple. This also often contributes to making event-driven controllers are easier to get right.

Note 8 (Correct event-driven control)**.** *As long as the controller responds in the right ways to the right events, event-driven controllers can be built rather systematically and are easier to prove correct. But beware! You have to get the handling of events right, otherwise you only end up with a proof about counterfactual physics, which is not at all helpful since your actual CPS then follows an entirely different kind of physics.*

Note 9 (Physics versus control)**.** *Observe that some parts of hybrid program models represent facts and constraints from physics, other parts represent controller decisions and choices. It is a good idea to keep the facts straight and remember which part of a hybrid program model comes from which. Especially, whenever a constraint is added, because of a controller decision, it is good practice to carefully think through what happens if this is not the case. That is how we ended up splitting physics into different evolution domain constraints, for example.*

Partitioning the hybrid program in (8) into the parts that come from physics (typographically marked like physics) and the parts that come from control (typographically marked like control) leads to:

$$\begin{aligned}&0 \le x \wedge x \le 5 \wedge v \le 0 \wedge g > 0 \wedge 1 \ge c \ge 0 \wedge f \ge 0 \rightarrow\\ &\quad \big[\big(((x' = v, v' = -g \,\&\, x \ge 0 \wedge x \le 5) \cup (x' = v, v' = -g \,\&\, x \ge 5));\\ &\qquad \texttt{if}(x = 0)\, v := -cv \;\texttt{else}\; \texttt{if}(4 \le x \le 5 \wedge v \ge 0)\, v := -fv\big)^*\big](0 \le x \le 5)\end{aligned} \tag{8}$$

Note that there could have been a second evolution domain constraint $x \ge 0$ for the physics in the second differential equation, but that evolution domain constraint was elided, because it is redundant in the presence of the evolution domain constraint $x \ge 5$ coming from the controller. Observe that only controller constraints have been added compared to the initial physical model of the bouncing ball (1) that was entirely physics.

4 Summary

This lecture studied event-driven control, which is one important principle for designing feedback mechanisms in CPS and embedded systems. The lecture illustrated the most important aspects for a running example of a ping pong ball. Even if the ping pong ball may not be the most exciting application of control in the world, the effects and pitfalls of events in control were sufficiently subtle to merit focusing on a simple intuitive case.

Event-driven control assumes that all events are detected perfectly and right away. The event-driven controller in (8) took some precautions by defining the event of interest for using the ping pong paddle to be $4 \leq x \leq 5$. This may look like a big event in space to be noticed in practice, except when the ball moves too quickly, in which case the event $4 \leq x \leq 5$ is over rather quickly. However, the model still has $x \leq 5$ as a hard limit in the evolution domain constraint to ensure that the event would never be missed in its entirety as the ball is rushing upwards.

Event-driven control assumes permanent continuous sensing of the event of interest, because the hard limit of the event is ultimately reflected in the evolution domain constraint of the differential equation. This evolution domain constraint is checked permanently according to its semantics (Lecture 3 on Choice & Control).

Exercises

Exercise 1. Can the ping pong paddle in (8) ever respond to the event $4 \leq x \leq 5$ twice in a row? What would happen if it did?

Exercise 2. Is the following formula an invariant for proving (8)?

$$0 \leq x \leq 5 \land (x = 5 \to v \leq 0) \land (x = 0 \to v \geq 0)$$

Exercise 3. Would the invariant (10) succeed in proving a variation of (8) in which the controller conjunction $\land v \geq 0$ is removed? If so explain why. If not, explain which part of the proof will fail.

Exercise 4. Would a generalization of formula (8) be valid in which the assumption $v \leq 0$ on the initial state is dropped? If yes, give a proof. If not, show a counterexample and explain how to fix this problem in a way that leads to a generalization of (8) that is still a valid formula.

Exercise 5. Could we replace the two differential equations in (8) with a single differential equation and a disjunction of their evolution domain constraints instead to retain a valid formula?

$$\begin{aligned}
&0 \leq x \land x \leq 5 \land v \leq 0 \land g > 0 \land 1 \geq c \geq 0 \land f \geq 0 \to \\
&\qquad \big[\big((x' = v, v' = -g \,\&\, (x \geq 0 \land x \leq 5) \lor x \geq 5);\\
&\qquad\quad \texttt{if}\,(x = 0)\, v := -cv \,\texttt{else}\,\texttt{if}\,(4 \leq x \leq 5 \land \boldsymbol{v \geq 0})\, v := -fv\big)^*\big](0 \leq x \leq 5)
\end{aligned}$$

Exercise 6. Conduct a sequent proof proving the validity of d$\mathcal{L}$ formula (8). Track which assumptions are used for which case.

Exercise 7 (*). Design a variation of the event-driven controller for the ping pong ball that is allowed to use the ping pong paddle within height $4 \leq x \leq 5$ but has a relaxed safety condition that accepts $0 \leq x \leq 2 \cdot 5$. Make sure to only force the use of the ping pong paddle when necessary. Find an invariant and conduct a proof.

References

[Pla10] André Platzer. *Logical Analysis of Hybrid Systems: Proving Theorems for Complex Dynamics*. Springer, Heidelberg, 2010. `doi:10.1007/978-3-642-14509-4`.

[Pla12a] André Platzer. The complete proof theory of hybrid systems. In *LICS*, pages 541–550. IEEE, 2012. `doi:10.1109/LICS.2012.64`.

[Pla12b] André Platzer. Dynamic logics of dynamical systems. *CoRR*, abs/1205.4788, 2012. `arXiv:1205.4788`.