15-824: Logical Foundations of Cyber-Physical Systems Proposal Fall 2018

Locked-on: verifying controls for aircraft tracking

Name: Chun Kai Ling, Daniel Wong Andrew ID: chunkail, dlwong

Contents

1 Abstract

Radar tracking of aircraft is a standard task for both civilians (air traffic control) and the military (ground-based air surveillance). This project focused on aircraft tracking using a ground-based radar, and the physical control needed to keep the radar beam locked on the aircraft. Our goal was to verify basic controls and protocols for continuous single-target tracking given reasonable flight parameters. We successfully modeled and verified controllers for the 1D and 2D tracking scenarios, which we believe may be relevant to other tracking tasks with a similar control component as well.[1]

2 Introduction

Radar is often used to monitor the distance and velocity of moving targets, including cars, planes and ships. Here, we focus on the tracking of aircraft using a ground-based radar as a motivating problem. (That being said, our control can easily be adapted for an aircraft-mounted radar that is tracking a speeding car or an unidentified ship.) In the air defence context, anti-aircraft weapons typically have their own radar system, which needs to remain locked on to a target before firing. In the civilian context, aircraft need to be tracked in order for air traffic control to ensure aircraft maintain safe separation, and to track uncooperative or hijacked aircraft. (Notably, after the Sept 11 attacks, major investments were made to upgrade radar systems throughout the country.) In both cases, the target may be trying to evade detection and execute evasive maneuvers. Thus, having a verified controller that is able to maintain *safety* (not lose track of the target) and be *efficient* (do so with a narrow radar beam radius and realistic reaction speeds) would be useful for administrators of such aircraft tracking radars.

Once the target has been acquired, the radar beam needs to stay on the target. From the radar operator's perspective, the center of the beam cannot stray too far from the aircraft. (Note that our focus is on physical control of the radar; the electromagnetic aspects are irrelevant to this project.) Keeping track of a moving target in this way is a recurring control problem in cyber-physical systems (e.g., a self-driving car following another car that wants to be neither too near nor far), and we found that the challenges we faced were more similar than anticipated to other tracking challenges in cyber-physical systems. We are hopeful that the proof techniques and intuition that we developed may also be useful in those tracking scenarios.

3 System Description

The operation of the radar is split into three consecutive phases: (P1) acquisition, (P2) tracking, and (P3) re-acquisition. In the acquisition phase (P1), the radar performs acquisition by scanning the airspace in a predefined pattern (e.g., a S-shaped pattern). Here, the radar is scanning the airspace to find its target. Once the target is found, we enter the tracking phase (P2), where the radar is shining at the aircraft and its goal is to maintain tracking. While locked on, the controller has access to the velocity of the aircraft. (P3) Sometimes, tracking is broken midway due to various reasons such as clutter or electronic jamming. In this case, operators have to re-acquire the target using stale information such as the last location of the aircraft, its velocity and knowledge of possible flight profiles (aerodynamics). These scenarios are shown in Figures 1, 2, and 3 [2]. In the rest of this project, we focus only on (P2) tracking.

[1] Both group members contributed equally to this work.

[2] Yes, we know these are satellite dishes and not tracking radars...

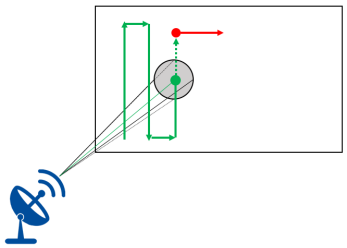

Figure 1: (P1)

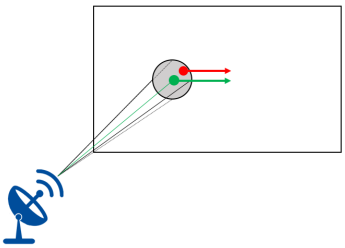

Figure 2: (P2)

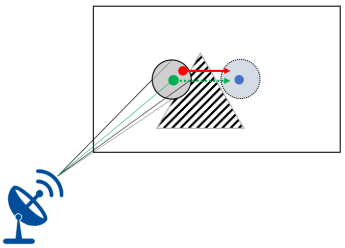

Figure 3: (P3)

4 Model

We make the following assumptions to make working with KeYmaeraX easier.

Assumption A0 We model the 'playing airspace' as a 2d plane, with y and x describing *height* and *location* respectively. The radar is modelled as a pencil beam (cone-shaped), while the aircraft is a single (x, y) coordinate.

We find that this assumption is reasonable as it models the perspective of the tracking radar. Assuming the distance from the radar to the aircraft is far enough to be considered approximately constant over time, movement directly towards or directly away from the radar is not relevant to the control of the radar as the aircraft will remain within the tracking beam.

Assumption A1 The aircraft is far from the radar relative to the time-frame of the tracking process.

Assumption A1 allows us to utilize small-angle approximations, that is, we can assume the radar controls are with respect to (x_a, y_a) coordinates as opposed to adjusting the angular velocities (which is what happens in practice). This assumption also reflects the reality of tracking aircraft, in that it is most difficult but also most important to track aircraft near the edge of the radar's operating range. Being able to track aircraft from a distance is especially important in the military context both to give enough time for anti-aircraft measures to be deployed and to serve the purpose of detecting hostile aircraft long before they are close to locations that the radar is responsible for protecting.

Furthermore, the distance from the radar to the aircraft, throughout the entire process, is approximately constant and is far greater than its height. This allows us to make the assumption that in the 2-d plane, the region that the pencil beam cuts the 2d playing field is approximately a circle with a fixed radius (see Figure 1), i.e., this region is described as an (x_r, y_r) point and some constant radius R. Hence, the radar would lock on when the (x_a, y_a) point of the aircraft lies within (Euclidean) distance R of the radar's position. Modelling aerodynamics is complicated and cumbersome, for example, fixed wing aircraft have very different profiles from rotory-winged aircraft. Hence, we will make some reasonable model reasonable flight dynamics such as those used by Jeannin et al. [3].

Assumption A2 The radar beam has unbounded acceleration.

Given that the distances that these radars are used may range from 10-40 miles, the problem of tracking is more akin to a human trying to point a laser pointer and keep a steady beam on a distant target than a spotlight following a human actor on stage. As a small angular velocity and

acceleration of the radar corresponds to a large velocity and acceleration of the radar beam at a distance of 10 miles (that could easily far exceed the maximum acceleration of the plane), we find this assumption reasonable.

Assumption A3 The aircraft and radar beam move in straight lines for the 2D plane.

One of the axes on the 2D plane corresponds to altitude. In flight, it is more common for aircraft to be flying level, or to maintain a constant pitch when climbing or descending (which better correspond to straight lines), as opposed to doing the acrobatics corresponding to arcs, which place a strain on both pilots and their aircraft, and may lead to undesirable scenarios such as spins.

4.1 Flight Profile Dynamics

For 1D profiles, we assumed that the aircraft has a constant altitude and may only move from left to right across the horizon from the perspective of the tracking radar. We ignored the axis corresponding to the distance of the aircraft from the radar as it is less relevant for the tracking problem as the aircraft will generally still remain within the radar beam, and the distance travelled by the aircraft in this direction should be a small fraction of the distance between aircraft and radar.

For non-constant velocity and acceleration in the 2D space, the aircraft might speed up, slow down, or change its pitch angle (with the ground). The aircraft will have a maximum airspeed, and maximum rate of climb as per its flight envelope. The minimum speed (the stall speed) is dependent on the angle of attack of the aircraft, and in real life is also affected by the altitude of the aircraft. Currently, we model this by bounding the minimum and maximum speed. As shown in Figures 4 and 5, civilian and military aircraft have different operating envelopes and the tracking radars needed will thus differ. Future work could have more realistic flight dynamics to incorporate the relationship between ground speed and the rate of climb/descent and more tightly bound the range of operating scenarios for the aircraft, which may lead to a more efficient controller – however we did not need this for our current controllers.

4.2 Controller

We model the radar with a time-triggered controller. In real life, the time limit T is a physical characteristic of the radar system used. In our model, the maximum T for which the tracking can be proved is dependent on the radius of the radar beam. Our model can thus be useful to someone designing a radar tracking system, to know the frequency at which the tracking system needs to run in order to track a target at a specified range, with different parameters possible to track civilian and military aircraft.

For the scenarios where the aircraft does not have constant acceleration / velocity, we use non-deterministic aircraft control for the aircraft.

5 Deliverables

As this is an implementation project, our deliverables are in the form of our model (.kyx) and proof (.kya) files. Assuming that we already know the aircraft's previous position and velocity, we want to position the beam to fall as close to the aircraft as possible such that we do not lose track. With reference to Figure 2, the red dot should always fall within the grey region. We successfully delivered the following proofs:

1. 1D controller using a ghost and restrictive assumptions (e.g., positive initial velocity)
2. 1D controller which permits the tracker to change its velocity between positive and negative

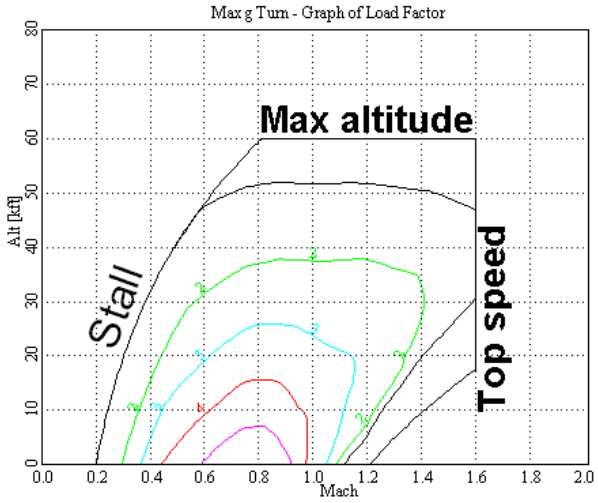

Figure 4: Sample Height-Velocity Diagram for a civilian plane [5]

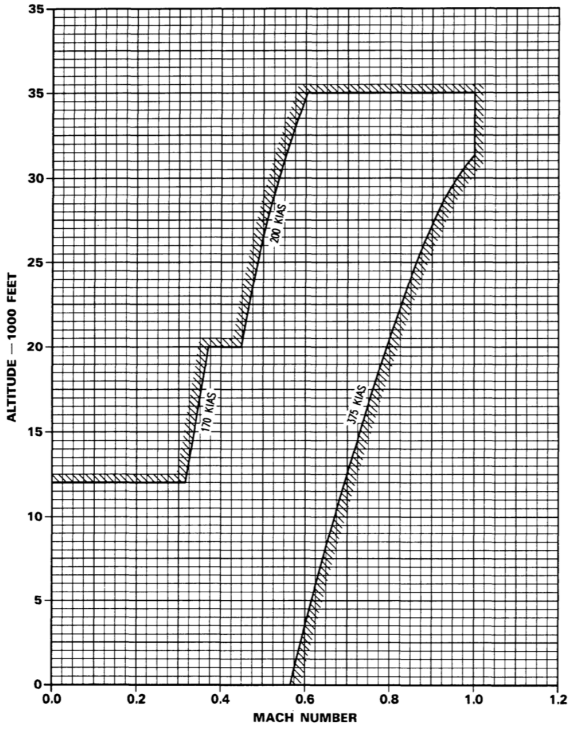

Figure 5: F16 Flight Envelope [1]. Outside of the Flight Envelope, the aircraft may stall or break up due to mechanical stress

3. 1D controller with constant known acceleration of the target
4. A more general proof for a class of 1D controllers that provide a continuum between the exponentially decaying controller and a controller with maximum oscillation
5. 2D controller that extends the 1D controller

We also delivered partial proofs for:

1. Non-deterministic 1D controller, where the adversary target has more complicated behaviour (modelled by introducing uncertainty into the acceleration (*'noisy dRatio'*))

6 Related work

It appears at first that this work is a trivial extension of Lab 2 (and 3). Here we detail why this is *not* so; in fact, our belief is that the tracking problem is significantly more challenging. These problems come to light in the *Spotlight* project [4], which uses the following template (pseudocode only to highlight relevant portions)

```
light_on_actor(t, va) === abs(x(t) - xa(t, va)) <= r - ra
Preconditions ->
[
```

```
{
 t := 0; va := *; ?(abs(va) <= V); a :=*;
 ?(light_on_actor(T, V) & light_on_actor(T, -V)
 & light_on_actor(text(V), V) & light_on_actor(text(-V), -V));
 {x' = v, v' = a, xa' = va, t' = 1 & t <= T}
 }* @invariant(light_on_actor)
}
]light_on_actor(t, va)
```

The predicate LIGHT__ON__ACTOR just says that the target and tracker are sufficiently close. To summarize, the program allows the arbitrary assignment of velocity to the target (actor in their case), and allows the controller to assign any acceleration to the tracker (spotlight in their case). The controller's acceleration is restricted by means of an input guard, which checks to make sure that the distance between the target and tracker is not too large. Since this distance is a quadratic, the authors check that the bounds hold for time T and the turning point of that quadratic, making sure that both of these quantities are not too large in magnitude. Intuitively, something is wrong when the invariant makes no reference to initial velocities, nor T. Indeed, **their model is incorrect**. Here are 2 examples of when their program ought to have no solution (i.e., no controller may satisfy their safety condition).

1) When the target is just at the edge of the tracker which is moving *backwards*. This satisfies the LIGHT__ON__ACTOR predicate. In the next instant, we would lose track, and regardless of how high we set acceleration, there is always some non-zero time where the spotlight is not on the target. This problem can possibly be fixed by changing the inequality in LIGHT__ON__ACTOR to a strict one.

2) Same as (1) but when the target is ϵ away from the boundary of the circle. Then, the tracker would have to accelerate very fast. However, if T is large (their program did not have any restrictions on T), then it is possible that we may move too far ahead and go out of bounds again.

The second problem appears to be caught by the input guards. However, the more serious point is that there could be no control which satisfies the input guard, making it a potentially **vacuous controller**! The object of this whole discussion is that unlike the labs, *there is no clear 'safe-by-default' action like braking here.*

Another prior project that may seem similar is Safe Robot: Follow-the-Leader in the Plane[2]. However, in that project, the follower (the radar beam in our case) can instantaneously adjust its velocity, which makes safe control much simpler as the follower can simply set its velocity to be the same as the tracked object's velocity. In contrast, to ensure safety in our project, we need to devise a loop invariant that considers both the difference in displacement as well as the difference in velocity in ensuring safety.

7 Simplified 1-D Problem

Let us consider a simpler 1-d problem with fewer variables. The proof follows in other cases without loss of generality, as we will explain in Section 9. These are the assumptions

1. The target is not moving.
2. The tracker is centered on the radar.
3. The tracker is moving at positive velocity v_0.

The general case is obtained by merging assumptions (1) and (3), which when combined together, says that the relative velocity of the tracker with respect to the target is v_0. Assumption (2) is made for simplicity; our proposed solutions will work in the general case with a few modifications.

Remark. This problem is similar to the follow-the-leader lab assignment *with efficiency constraints*.

The problem may be interpreted as follows. For each cycle, the controller chooses an acceleration. Then, an adversary chooses the duration of the evolution of the DE for time t, between $[0, T]$. After time t, we are allowed to choose a new acceleration. Our goal is to make sure that the magnitude area underneath the graph (potentially negative) does not exceed some value.

Consider Figure 6 for an example of how a controller may behave. Each color represents a control cycle (also referred to as a run). Dotted lines denote the trajectory had the controller not been interrupted. Notice that the time between 2 reactions is non-deterministic, but no greater than T. The area under the (piece-wise linear) curve represents the displacement of the controller relative to the target. Our goal is to design controllers such that the area under the (solid) curve is bounded in terms of v_0, given that the upper bound on reaction time, T is known.

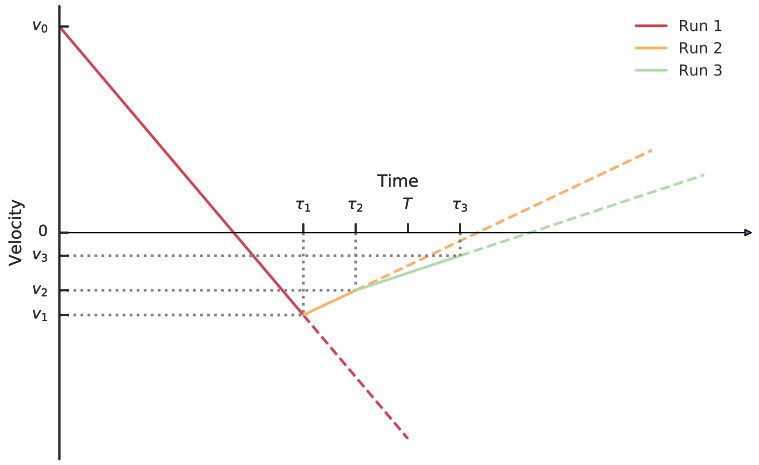

Figure 6: Velocity-time graph for a possible controller.

Reminder regarding notation x, v, a refer to displacement, velocity, and acceleration. t refers to the time passed in a particular control cycle. T is the upper bound on reaction time ($t \leq T$ always). R is the size of the pencil beam, i.e, the maximum distance from the aircraft in order to maintain lock.

7.1 Connections to stability and control

With these simplifications, the problem may appear to be a straightforward control problem, where our goal is to stabilize the displacement of the controller. For example, textbook dynamical systems

allow for dynamics of the following form

$$\begin{aligned} x' &= f(x,t) \qquad \text{(Continuous systems)} \\ x(t+1) &= g(x,t) \qquad \text{(Discrete systems)}, \end{aligned}$$

where f and g are dynamics and transition functions respectively, and the dependence on t in f, g denotes the distinction between autonomous and non-autonomous systems, with the latter allowing for dynamics that vary according to time (resp. iteration). In our case, we would like our relative displacement to eventually become stable (or at least, not exceed a certain magnitude). In control theory, there are several different notions of stability. In our project, we will only concern ourselves with three of them.

1. *Stability in the sense of Lyapunov* , commonly abbreviated as i.s.L . Informally, this means that the system, if starting close enough to an equilibrium point, will continue to remain close to equilibrium. For example, undamped oscillation is stable i.s.L. As far as radar tracking is concerned i.s.L is 'good enough', in that things are acceptable as long as relative displacements are not too high.
2. *Asymptotic Stability.* Informally, this means that the system has to eventually converge. This typically includes damped systems or oscillatons, for example, a damped harmonic oscillator eventually returns to rest. Undamped oscillators do not satisfy this property, since they go on forever.
3. *Exponential Stability.* A special case of Asymptotic stability, where the system is further required to converge at an exponential rate, e.g., $||x(t) - x_{\text{eqm}} \leq \alpha||x(0) - x_{\text{eqm}}|| \exp(-\beta t)$ for some positive constants α, β.

We stress that while these notions of stability are *relevant* and give inspiration as to what is typically desired of controllers, the exact control problem we are trying to solve is, to the best of our knowledge, not common. **The crux of the issue here is that the response time t is *not* a fixed constant.** In a sense, our desired controller has to be stable even if the response time was picked adversarially (not necessarily stochastic). In fact, the equilibrium point itself depends on the reaction times, which is somewhat atypical. Another way of looking at our project is to determine entire classes of dynamical systems $\mathcal{G}$ where for each $g \in \mathcal{G}$, the system $x(t+1) = g(x,t)$ is stable (possibly with different equilibrium points). Here, g summarizes both the controller's decisions, and the infinitely many choices of reaction times $t_0, t_1 \ldots$.

Despite these fundamental differences, we will see later that there are analogies between our proposed controllers and the different notions of damping, as well as the different types of stability. Furthermore, the backbone of our proofs, specifically, our loop invariant, is precisely a Lyapunov (energy) function.

7.2 Why is this problem challenging

One may (incorrectly) think that the issue with nondeterministic reaction times is a small one. Indeed, if a controller works in the 'worst-case' scenario where reaction times are $t = T$ for all cycles, then it *ought* to work in the 'better' case where reaction times are faster. Unfortunately, this line of thought is erroneous.

For example, consider the controller which sets its acceleration to 1 if its velocity is negative and -1 otherwise. If reaction time was deterministic at T, then this controller simply oscillates and is

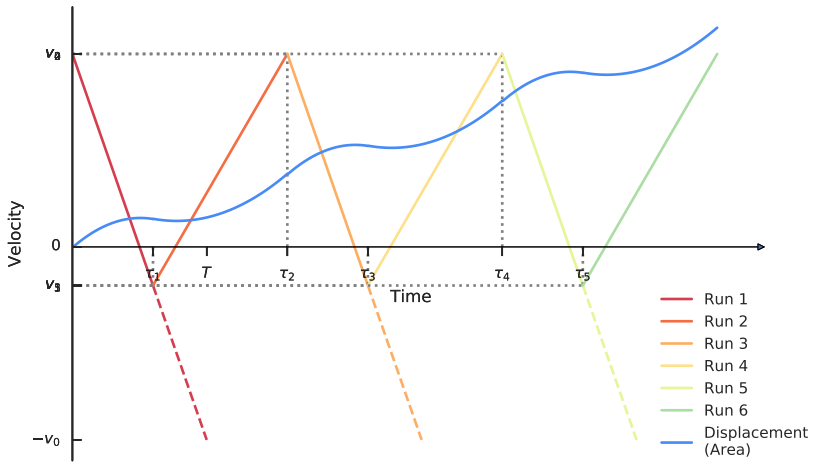

Figure 7: Piecewise linear segments denote velocity. Each color is one control cycle. Dark blue curved line denotes displacement (area under the straight lines). Observe that the blue lines extends to infinity.

stable i.s.L. However, nastier things can happen if reaction times were chosen at inconvenient times. Consider Figure 7. When the velocity is positive, the reaction times are chosen to be as short as is required to achieve negative velocity. However, when velocity is negative, then reaction times are chosen to be as long as possible. This can create a divergent behavior as shown. This example again stresses the importance of dealing with all possible edge cases. These reasons, together with the potential hiccups faced by the SPOTLIGHT paper suggest that even this simplified problem is worth solving.

8 Creating a working controller

A little intuition is in order. First, observe that we *should* have a solution – if our reaction time T is 'infinitesimally' small, then we can set $a \propto -x$, where x is displacement, i.e., we get some variant of simple harmonic motion, which under no other external forces has bounded displacement. As long as the oscillation boundaries are less than the safe buffer radius R, we would be safe. Of course, we are making the unjustified claim that our understanding from the continuous control case carries over to the discrete control, which is not necessarily true.

The second observation is that if we are guaranteed that the control would trigger in *exactly* time $t = T$, then a a straightforward solution which does not oscillate is available. Simply set $a := -v_0/T$. After exactly T time, velocity is 0 and as long as we are still within the tracking radius, we may subsequently set $a = 0$ and remain safe forever. Unfortunately, the same argument does not work when the reaction time *could* be $t < T$.

8.1 Exponential Decaying Controllers (A)

It turns out that the aforementioned controller does indeed work, albeit with more analysis. We call this controller (A) for brevity. It simply sets $a := -v/T$ at all control iterations, where v is

the tracker's velocity at the start of each control iteration. (Obviously, when the reaction time t is deterministic and equal to T, we are done.) We describe this controller as 'exponentially decaying', since it represents some form of halving process, but with nondeterministic time periods.

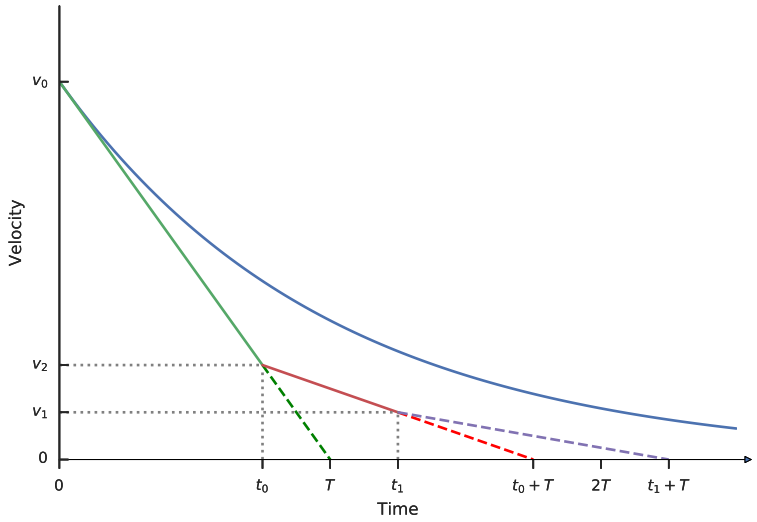

Figure 8: Blue: upper envelope on velocity. Green, Red, Teal: velocities for the first, second and third control cycles respectively. Observe that 1) velocities are always nonnegative, and 2) upper bounded by the blue-curve. Areas under the curve (or the piecewise graph formed by actual velocities) correspond to the (nonnegative) distance travelled. The area under the blue curve is finite, even as $t \to \infty$.

The key here is to observe that the area under the tracker's velocity-time graph is upper bounded by an exponentially decaying curve. See Figure 8 for a graphical illustration of the bound. Since negative-exponential curves have finite area underneath, we are done. An informal, pen-and-paper proof of this is shown below.

Theorem 1. *The total displacement at any time $0 \leq t \leq T$ is no greater than $v_0 \cdot T$, where v_0 is the initial velocity of the tracker.*

Proof. Let $t_i, i \in \{1, 2, \dots\}$ be the i-th reaction time (i.e., the for the i-th period) and v_i be the *initial* velocity of that time period. Note that by definition, $t_i \leq T$. Then

$$v_{i+1} = v_i \left(1 - t_i/T\right) \quad \forall i > 0$$

Unravelling gives us

$$\begin{aligned}
v_1 &= v_0\left(1 - t_0/T\right)\\
v_2 &= v_0\left(1 - t_1/T\right)\left(1 - t_0/T\right)\\
v_3 &= v_0\left(1 - t_2/T\right)\left(1 - t_1/T\right)\left(1 - t_0/T\right)\\
&\cdots\\
v_i &= v_0\prod_{j=0}^{i-1}\left(1 - t_j/T\right)\\
&\leq v_0\prod_{j=0}^{i-1}\exp\left(t_j/T\right)\\
&= v_0\exp\left(\frac{1}{T}\sum_{j=0}^{i-1}t_j\right)\\
&= v_0\exp\left(\frac{\tau_{i-1}}{T}\right)
\end{aligned}$$

where for simplicity, we have defined $\tau_i = \sum_{j=0}^{i}$. Furthermore, the exact velocity at any time t is given by

$$v(t) = \begin{cases} v_0\left(1 - \frac{t}{T}\right) & 0 \leq t \leq t_0\\ v_1\left(1 - \frac{t-t_0}{T}\right) & t_0 \leq t \leq t_0 + t_1\\ \cdots & \\ v_i\left(1 - \frac{t-\tau_{i-1}}{T}\right) & \tau_{i-1} \leq t \leq \tau_i \end{cases}$$

Crucially, we my observe that by induction, the velocities are always non-negative (this is also apparent from the way we select acceleration). Combining the the expressions for v_i, we may obtain the inequalities

$$\begin{aligned}
v(t) &\leq \begin{cases} v_0\left(1 - \frac{t}{T}\right) \leq v_0\exp\left(-\frac{t}{T}\right) & 0 \leq t \leq t_0\\ v_0\exp\left(\frac{t_0}{T}\right)\left(1 - \frac{t-t_0}{T}\right) \leq v_0\exp(-\frac{t}{T}) & t_0 \leq t \leq t_0 + t_1\\ \cdots & \\ v_0\exp\left(\frac{\tau_{i-1}}{T}\right)\left(1 - \frac{t-\tau_{i-1}}{T}\right) \leq v_0\exp\left(-\frac{t}{T}\right) & \tau_{i-1} \leq t \leq \tau_i \end{cases}\\
&= v_0\exp\left(-\frac{t}{T}\right)
\end{aligned}$$

The total displacement at any time t is given by

$$\begin{aligned}
\int_0^t v(t)dt &\leq \int_0^t v_0\exp\left(-\frac{t}{T}\right)dt\\
&\leq \int_0^\infty v_0\exp\left(-\frac{t}{T}\right)dt\\
&\leq v_0 \cdot T
\end{aligned}$$

Furthermore, since $v(t) \geq 0$ for all t, the lower bound for this quantity is also 0. Thus, $v_0 \cdot T$ bounds the absolute total distance travelled by our controller. We stress again that the t_i's were *not* chosen by the controller but rather by an adversary, as long as they are bounded by $[0, T]$. ■

Thus, our controller is safe as long as the safe radius R is greater than $v_0 \cdot T$. We may also observe that the bound on the maximum displacement may be made arbitrarily tight by decreasing making all t_i's arbitrarily small. This makes the piecewise linear graph track the exponential curve more tightly. Now, the only outstanding issue is to express this in d$\mathcal{L}$.

As one may expect, the formal proof almost literally follows the velocity graph in Figure 8. In our differential equation, we provide a 'ghost' which evolves based on exponentially decaying dynamics. This ghost quantities and the horizontal line 0 upper and lower bound the velocity of the controller. Concretely, we have

```
...
Real Xub;              /* Upper bound for displacement */
Real Vub;              /* Upper bound for velocity */
/* Preconditions and other parts of the program */
...
[
    {
        ... /* Control strategy */ ...
        {
        ... /* Aicraft dynamics */
        ...
        Xub' = Vub,
        Vub' = -Vub/T,
        ...
        }
    }*invariant(...
                cX <= Xub &
                R - Xub >= Vub * T &
                Vub > 0
                ...
                )
] ... /* Postconditions */ ...
```

where Xub and Vub are the exponential bounds and cX is the controller's displacement. The portion of interest is the loop invariant, which says that (1) the area travelled by the controller is no greater than that of the upper bound, (2) the area underneath the upper bound is no greater than that of buffer and the distance already travelled, and (3) the area. It also turns out that KeYmaeraX is able to figure out that for exponentially decaying dynamics, the area under the curve is finite. The actual steps in the proof are not too complicated, but involve several differential cuts in order to get certain bounds into the evolution domain constraints. As with our pen-and-paper proof, we use the linearization of $\exp(\cdot)$ as a bound for the exponential function; this is proven easily by repeatedly applying dI, identical to what we did in assignments.

An interesting and useful freebie that we obtain in the process of the proof: our controller is exponentially stable. At the very least, the ghost shows that velocity clearly decays exponentially. A similar case may be made for displacement (to the eventual equilibrium), implying that the whole system is exponentially stable. Furthermore, our controller's acceleration is clearly bounded (by construction, and the fact that the velocity at every iteration is upper bounded by the initial

velocity). In practice, this means our controller may be physically implementable.

8.2 A more aggressive controller (B)

Looking at Figure 8, it is easy to propose different classes of algorithms based on the level of overshoot in velocity. In our initial work, we set our 'target' velocity at time $t = T$ to be equal to 0. The other alternative is to select the target velocity to be the negative of current velocity, which also implies that (assuming maximum reaction time) the incurred displacement for each control cycle is 0. In fact, this controller may be expressed easily with a simple modification. This is illustrated in Figure **??**. We refer to this as controller type (B).

Unlike our initial controller, controller (B) allows for negative velocities, that is, the controller may potentially swing back and forth between cycles. Observe however, that because of nondeterministic reaction times, it is not necessarily the case that the sign of velocity changes *every* control cycle.

Unlike controller (A), controller B is not as easily analyzed. Intuitively, this should be stable (at least) in the Lyapunov sense, since the magnitude of velocity never increases, and velocity keeps trying to 'compensate for itself' by trying to flip signs. However, note that it cannot possibly be *asymptotically* stable, since if reaction times are precisely T for all control cycles, undamped oscillatory behavior results (see Figure 9).

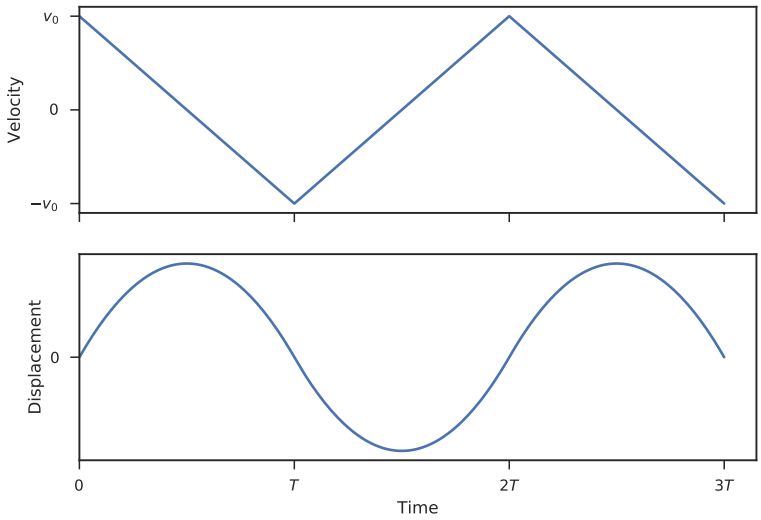

Figure 9: Oscillatory behavior when reaction time is equal to T for every control cycle. The system is stable i.s.L, but does not converge to a point. Note that the displacement graph is a series of quadratics, and *not* a sinosoid.

Unfortunately, the 'picture proof' of Figure 8 does not apply here easily. This arises from the overshoot of velocity, which makes accounting for positive and negative velocities separately difficult. The key to overcoming this is to look at the velocity-displacement graph, rather than velocity-time graph (and the area underneath it).

Indeed analyzing controller (B) requires the use of a state-space diagram, where state refers to both velocity and acceleration. For example, the trajectory of the case where reaction time is exactly $t = T$ is shown in Figure 10. Each of the left and right components of the closed curve is a fragment

of a quadratic, and coresspond to a single control cycle each. Note further that the curve is not smooth at the edges where the displacement is 0.

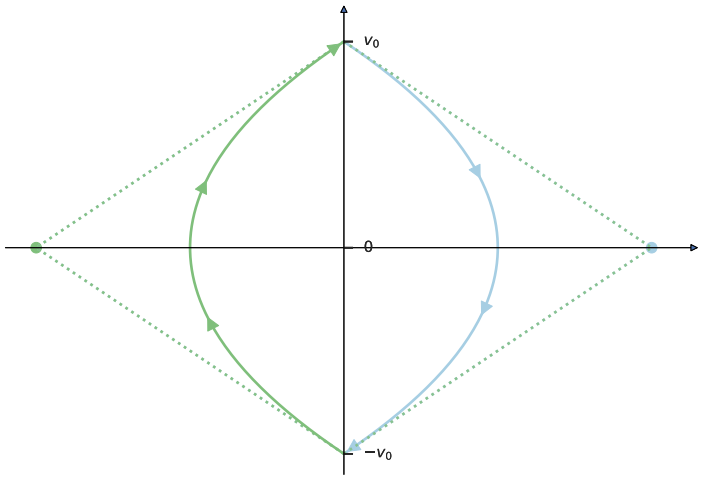

Figure 10: Thick lines: velocity-displacement graph when reaction time is deterministic. Each color is exactly one control cycle. The trajectory of the oscillation is 'clockwise' in the velocity-displacement graph. Note that the dotted diamonds completely enclose the curve.

What could happen when reaction times could be less than T? Now, each control cycle occupy only a portion of the quadratic (with reference to Figure 10). Then, we would begin a new control cycle and start afresh from a new quadratic, centered at the current displacement. Our analysis technique is standard in control – we find region of the state space that we will never leave, and show that this region is 'small' or at least bounded. This would imply that each quantity is also bounded. Looking at Figure 10, it may seem hopefully that the region bounded by the curve would serve as such a region. Unfortunately, this is incorrect, we need a larger region.

It turns out that the outermost red 'diamond' in Figure 11 is precisely the shape that we need. In particular, the red diamond encompasses all the possible curves that we may encounter. Since it itself is a finite geometric object, it has a maximum displacement coordinate, which in turn forms a bound for safety.

We now need to show that each curve is completely contained within the diamond of the same color. Informal proof of this is surprisingly trivial. Consider the *gradient of velocity with respect to displacement*. This is equivalent to

$$\begin{aligned}\frac{dv}{dx} &= \frac{\frac{dv}{dt}}{\frac{dx}{dt}} \\ &= \frac{a}{v} \\ &= -\frac{2\text{old}(v)/T}{v}\end{aligned}$$

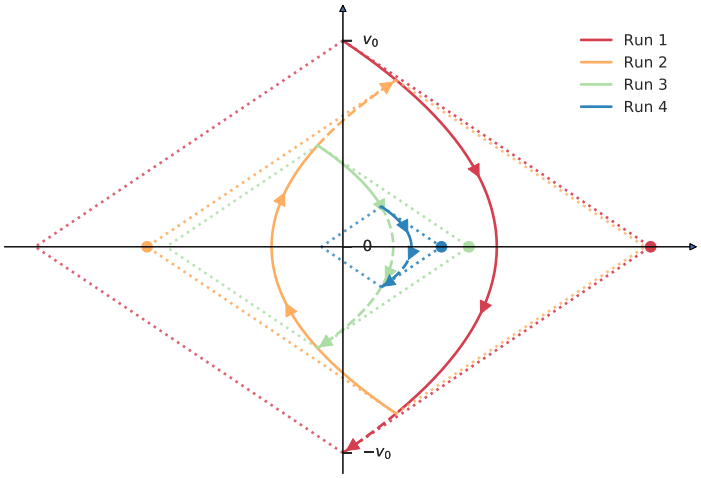

Figure 11: Velocity displacement graph. Light dotted straight lines denote the upper bound for each iteration, observe that the outermost diamond contains all other diamonds (which in turn contain the curved portion). Thick dotted lines denote where the trajectory had it not been terminated early. Note that the thick lines do not necessarily cross the x axis at every control cycle. Furthermore, observe that each diamond encloses the corresponding (solid) curve of the same color, and thus, the outermost diamond encloses everything else.

Here old(v) refers to the velocity at the beginning of the current control cycle. Now, if the control cycle just started, then old(v) $= v$ and this reduces to $-\frac{2}{T}$. Crucially, the $v - x$ gradient at the beginning of each control cycle is independent of both v and x. Furthermore, since $v \leq$ old(v) in magnitude, the gradient becomes sharper over time, i.e., future trajectories would remain within the diamond, *which has the same shape/aspect ratio regardless of iteration count.*

What about a formal proof? It turns out that having this geometric intuition tells us exactly which region we will never leave and is hence an excellent choice for a loop invariant. In fact, our program is of the following form.

```
( ... /*Preconditions */ ...)
  ->
  [
    {
      /* Controller */
      cA := -(cV*2)/T;
      ...
      /* ODEs */
      {cX' = cV,
       cV' = cA,              /* Controller dynamics */
       t' = 1 &               /* Time indicator */
       t <= T}                /* Reaction time bounds */
```

```
  }*@invariant( abs(cX-old(cX))*2/T + abs(cV) <= abs(old(cV)) &
                abs(cX) + abs(cV)*T/2 <= R
              )
]( ... /* Postcondition */ ... )
```

Here, cA, cV, cX are the controller's acceleration, velocity, and displacement respectively. The key here is the invariant, where in the first line, we are saying that at the end of the control cycle, we remain in the diamond centered on the previous displacement. The second line says that the *next* diamond centered at our current displacement does not exceed our buffer R, this may then be used to prove our postcondition.

As may be expected, once armed with this simple invariant, the proof itself becomes extremely straightforward. All one needs to do is to first apply cuts to force the DE solutions into evolution domain constraints. The additional branches created but the cut are trivially proven via dI. Once the solutions for the differential equations are in place, just apply dW. The rest is simply an application of QE, which is simple as we can just offload the work to the Mathematica backend.

8.3 A generalization of our controllers

Recall that our controllers so far are

$$a := -v/T \qquad \text{(Controller } A\text{)}$$
$$a := -2v/T \qquad \text{(Controller } B\text{)}$$

Naturally, this leads to the question: are there a range of such controllers which are in fact stable? So far, both controllers are chosen out of convenience of analysis. Could we have in fact selected $a := -1.5/T$ and still be safe? Unsurprisingly, the answer is *yes*. This leads us to the more general class of controllers,

$$a := -\gamma v/T \qquad \text{(Controller } C\text{)}$$

where $0 < \gamma \leq 2$ is an arbitrary parameter. Figures 13 and 12 illustrate the case for $0 < \gamma < 1$ and $1 < \gamma < 2$. Note that the we assume that reaction time is precisely $t = T$.

First, observe that for $1 < \gamma < 2$, we have a similar situation as when $\gamma = 2$, i.e. Controller (B). However, now, we do not have a closed loop, but rather, a spiral that collapses inward towards an equilibrium point. This is because each quadratic is followed for an 'irregular' amount of time, before moving on to the next. This is as opposed to Controller (B), which follows the quadratic one end to the other. Regardless of the precise calculation, we can immediately see that a similar invariant will hold. In fact, the outermost 'diamond' encompasses all inner diamonds, which share the same aspect ratio. Using similar arguments as for $\gamma = 2$, we may convince ourselves that the outermost diamond continues to be an upper bound even if reaction time was non-deterministic.

The other case for $0 < \gamma < 1$ is slightly more interesting. Observe that the state remains in the top-right quadrant throughout. Velocity monotonically decreases, but never goes below 0. Regardless, the same argument still applies – the outermost diamond (not fully shown) serves as a bound on the reachable states.

In all cases, the exact same proof may be applied, with some slight adjustments where factors of 2 are replaced with γ. In fact, the .kyx, tactics, and .kya files are virtually identical; the major difference is simply the time required for the final QE step. In our case, it took around 10-15 minutes, although we believe this could be sped up but cutting down on the number of abs() operators.

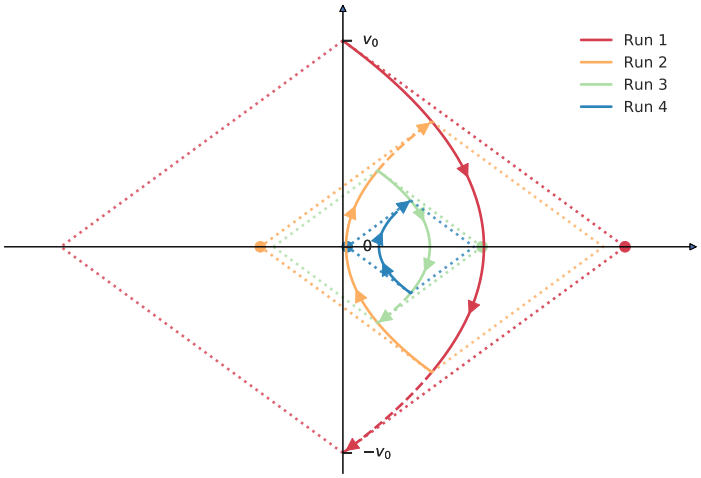

Figure 12: Velocity-displacement graph. Trajectory when $\gamma = 1.5$. Light dotted straight lines denote the upper bound for each iteration, observe that the outermost diamond contains all other diamonds (which in turn contain the curved portion). Thick dotted lines denote where the trajectory would have went had $\gamma = 2$, assuming reaction time was deterministic.

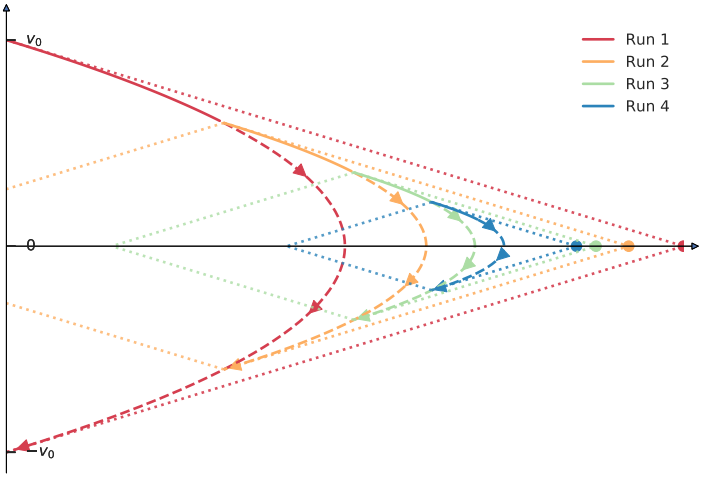

Figure 13: Velocity-displacement graph. Trajectory when $\gamma = 0.5$. Light dotted straight lines denote the upper bound for each iteration, observe that the outermost diamond contains all other diamonds (which in turn contain the curved portion). Thick dotted lines denote where the trajectory would have went had $\gamma = 2$, assuming reaction time is deterministic. Note: only the right half of the 'diamond' is shown.

There is one final point pertaining to which is the 'most efficient' choice of γ. Some simple calculations reveal that $dy/dx = -\gamma \text{old}(v)/Tv$. That is, when γ is larger, the diamond is in fact, smaller (i.e., we may allow for a smaller buffer or a larger v_0). This suggests that as far as safety constraints are concerned, setting $\gamma = 2$ is the most efficient. However, note that this allows for overshoots, and is not even asymptotically stable. Setting $1 < \gamma < 2$ will still suffer from overshoots, but is at least asymptotically stable. Setting $0 < \gamma < 1$ has no overshoots, but does have a weaker bound. Setting $\gamma = 0$ is the most efficient controller which does not suffer from overshoots. It is interesting to note that this is analogous to the idea of undamped ($\gamma = 2$), underdamped ($1 < \gamma < 2$), critically damped ($\gamma = 1$) and overdamped ($0 < \gamma < 1$) oscillators.

9 Extensions

9.1 Relaxing unnecessary assumptions

In the previous section, we made several assumptions on the initial conditions of the problem, for example, that $v_0 \geq 0$. In reality, these are strictly tighter than the loop invariant itself, and we may do without them. The ghostly proof (Section 8.1) with the exponential upper bound on velocity will obviously hold with the graph flipped along the x axis. Similarly, the proof involving $v - x$ graphs will also hold, since we are merely concerned with absolute values in all subexpressions. Nonetheless, for completeness, we proved some of the above, with these constraints relaxed. The proofs are not novel and the reader is directed to the relevant .KYA files instead.

9.2 Moving targets with constant acceleration

This extension is, with some thought, trivial. In fact, one only needs to replace the terms 'displacement', 'velocity', and 'acceleration' with their relative counterparts (e.g., relative displacement) and all the proofs will still hold. For example, if the aircraft is accelerating at $\tilde{a}$ and the controller at $\hat{a}$, then the effective acceleration, using the controller as a reference point, is $\tilde{a} - \hat{a}$. This similarly holds for the other 2 quantities. This is illustrated for clarity in Figure 14, and extends the proof in Section 8.1. In our actual proofs, we did not make us of this meta argument (QE works well all the same). The main point however, is that we do not need to restart the search for a new loop invariant just to handle this extension.

9.3 2D-trajectories

Extension to the 2D plane is in fact trivial. If we may guarantee that the relative displacement in each coordinate is bounded, then the distance in L2 norm is also bounded (up to a factor of $\sqrt{2}$). From a practical standpoint, we simply have to apply a cut and apply the 1d proofs as lemmas.

9.4 Non-determinism in dynamics/aircraft acceleration

Let us suppose that the aircraft may in fact change its acceleration during the flight. That is, when we set relative acceleration to a, it may actually be $a + \epsilon$, where ϵ is bounded in magnitude. There are several possible variants on this which we describe below in *decreasing* difficulty.

1. Relative acceleration (the aircraft) may change its acceleration arbitrarily, even within the control cycle. This is the most realistic scenario, and corresponds to the case when the tracked aircraft is adversarial (e.g., in military settings).
2. Relative acceleration may change only between each control cycle. This would happen not because of some coordination between the controller and the aircraft, but rather due to 'noise' in the selection of acceleration. For example, mechanical faults may cause us to effect an

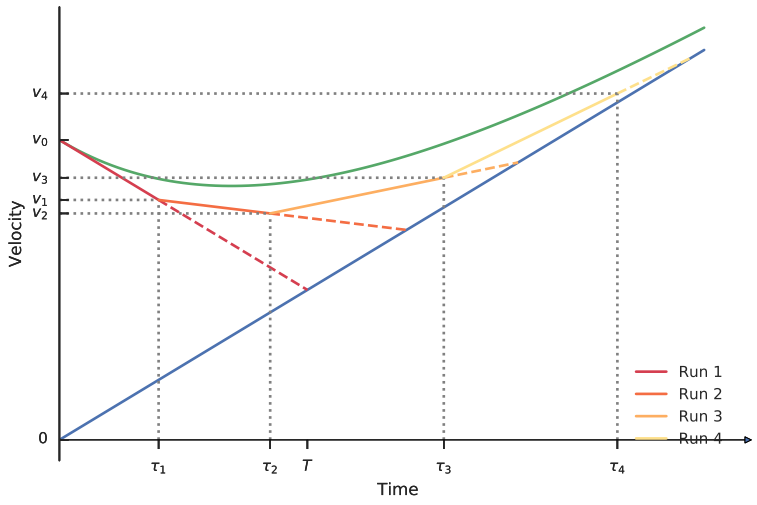

Figure 14: Velocity-time graph (non-relative). The straight line starting from the origin is the velocity of the aircraft. The controller simply has to compensate for the aircraft's velocity and acceleration in his control. Note that relative displacement is equal to the area *between* the blue line and the piecewise linear graph.

acceleration slightly different from what we would have liked to. This is a fairly realistic and important case, since nobody can guarantee absolute precision in physical controllers.

3. Relative acceleration may change, but only in a multiplicative sense with respect to relative velocity (as opposed to additive non-determinism). Recall that our controllers are of the form $a := -\gamma v/T$. If we enforce ϵ to be of the form $\kappa \cdot v$, where κ is small, then this would reduce to $a := -(\gamma + \kappa)v/T$.

Unfortunately, we only managed to prove the third scenario [3]. This is the least realistic scenario, but also the only one which follows directly from our previous proofs. Intuitively, this makes the 'diamond' for each iteration vary in size for each iteration. Thus, a safe bound would the the largest such diamond. More concretely, suppose that $\kappa = 0.5$. Then if we adopt $\gamma = 1$, the largest diamond that may be formed is that corresponding to a gradient of $0.5/T$. This case is illustrated in Figure 15 – the outermost diamond (not fully shown) completely encompasses inner ones.

Why are the first 2 scenarios challenging? We believe that reasonable controllers *have* to exist, since the problem is not complicated at all. However, the problem is that the 'diamond' region argument will never work in this situation. This is because if we are stationary but at the end of the boundary (i.e., at the left or rightmost edges of the diamond), then any acceleration would cause velocity to shoot up perpendicularly (in the $v - x$ graph) to the x axis. In the next instant, we will certainly not be within the diamond. However, we know that the diamond 'safe' area is tight even when there is no non-determinism in acceleration. These 2 traits are unfortunately incompatible.

[3] In practice, there was some issue with KeYmaeraX and QE. However, we are sure that with some tinkering and unravelling of expressions, the proof would follow through.

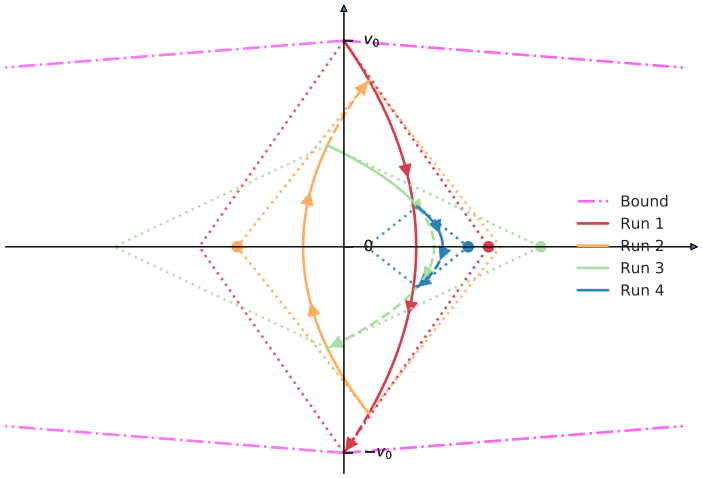

Figure 15: Velocity-displacement graph. This is for Scenario 3, when aircraft acceleration is non-deterministic multiplicatively. Here, each diamond varies in size, but the pink diamond enjoys the widest possible in aspect ratio. This is then used as the outer bound.

Our believe is that the controller would (finally) have to take into account relative displacement into account. This however, is a significantly larger can of worms compared to the relatively simple arguments we have made so far.

10 Proofs included

10.1 1D cases

The following have been proven. This forms the bulk of our project

1. `1d_target_stationary`
2. `1d_target_stationary_neg`
3. `1d_target_stationary_overshoot`
4. `1d_target_acc`
5. `1d_target_acc_overshoot`
6. `1d_target_acc_general`

The descriptions are as follows.

1. Contains the fancy proof using a ghost. We make some assumptions, such as initial velocity being positive. Proving the negative case is trivial and not included (we just change the signs of some values and directions of inequalities).
2. Contains the case where initial velocity in (1) can be negative, just to show it may trivially be done.
3. Contains the case where we have a different type of controller, where the velocity may oscillate between positive and negative. The controller is almost identical to (1) up to a factor, but the

proof technique is far simpler and general, and does not require nasty ghosts. We essentially solved the DE's explicitly and use DW and QE to complete.
4. Contains an extension of (1) or (4) where the target aircraft is allowed to have a constant, known acceleration.
5. Contains an extension of (3), where the aircraft is allowed to have a constant known acceleration.
6. Contains a most general case where we allow for a spectrum of controllers which 'interpolate' between controllers (1) and (3), and even beyond based on a parameter dRatio (γ in this report). (1) corresponds to dRatio = 1 and (2) to dRatio = 2.

10.2 2D cases

1. `2d_target`
2. `2d_target_l1`

These are their descriptions.

1. Contains the completed proof for bounds on both the x and y distances, i.e. bounds in L1 norm in the (velocity-displacement) state diagram.
2. Contains the partial proof (which *should* complete easily, and is trivial anyway) bounds in L2 norm in the (velocity-displacement) state diagram.

For some reason (2) doesn't prove (stuck in a QE step), however, it is obvious that bounds in L1 implies bounds in L2 up to a factor of $\sqrt{2}$. With some tweaks, this should work.

10.3 Non-deterministic acceleration

We only have partial proofs and models here.

1. `1d_target_acc_nondet`
2. `1d_target_acc_nondet_fake`

Descriptions:

1. We have the case where dRatio may change between 0 and 2, between each iteration. This coressponds to the case where we may have 'noisy' dRatios.
2. We have the case where the aircraft may change its acceleration non-deterministically, and not necessarily at the same time as the controller. This is the most difficult case and extremely difficult.

11 Conclusion

We have analyzed, modeled and verified controllers for simple tracking problems. In the case of aircraft with constant acceleration, we have proven the correctness of our controller. Even though we have primarily focused on the tracking of aircraft, we believe our work is relevant for other tracking or surveilance tasks as well, as well as other stability related systems with nondeterministic response times. Future work includes devising and verifying stable controllers for the nondeterministic case.

References

[1] Lockheed Martin Corporation. USAF F-16A/B Flight Manual. 1995.

[2] Austin Davis and David Wise. Safe robot follow-the-leader in the plane. 2014.

[3] Jean-Baptiste Jeannin, Khalil Ghorbal, Yanni Kouskoulas, Aurora Schmidt, Ryan Gardner, Stefan Mitsch, and André Platzer. A formally verified hybrid system for safe advisories in the next-generation airborne collision avoidance system. *International Journal on Software Tools for Technology Transfer*, 19(6):717–741, 2017.

[4] Jesurum. In the spotlight: Verifying automated theatrical follow-spots. *CPS VV Grand Prix*, 2017.

[5] Wikimedia Commons. AtltitudeEnvelope.png — Wikimedia Commons, 2015. [Online; accessed 17-November-2018]. URL: `https://commons.wikimedia.org/w/index.php?title=File:AtltitudeEnvelope.png&oldid=149838781`.